

HEWLETT-PACKARD



User's Reference

Notice

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

Hewlett-Packard makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

IBM is a U.S. registered trademark of International Business Machines Corporation.

MS-DOS is a U.S. registered trademark of Microsoft, Incorporated.

Vectra is a U.S. registered trademark of Hewlett-Packard Company.

Printing History

First Edition -- September, 1985
Second Edition ---- June, 1987

Printed in Singapore
Printed in Singapore

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.



MS-DOS 3.2 Features

New MS-DOS Commands

- **FOR150**

This new command prepares a 720 Kb 3.5-inch disc for use with an HP150 computer. It is similar in function to the **FORMAT** command.

- **KEYBxx**

This new command replaces the **KEYBUS** command. It provides support for non-U.S. keyboards.

- **REPLACE**

This new command selectively replaces files on a disc. It can also selectively add files to a disc.

- **SELECT**

This new command installs **MS-DOS** on a new disc with the country and keyboard codes of your choice.

- **XCOPY**

This new command copies groups of files, including files in subdirectories and the subdirectories, themselves.

Revised MS-DOS Commands

■ ATTRIB

Two new options have been added to this command: the +A and -A options. They set and clear the archive attribute of a file.

■ COMMAND

One new option has been added to this command: the /E option. It sets the size of the MS-DOS environment.

■ FORMAT

Two new options have been added to this command: the /T and /N options. They format a 3.5-inch 720 Kb disc in a 1.44 Mb drive.

One option has been removed from this command: the /P option. It's been replaced by the new PAMINSTL utility, which is described in the *Installing Your Operating System* manual.

■ SYS

One option has been removed from this command: the /P option. It's been replaced by the new PAMINSTL utility, which is described in the *Installing Your Operating System* manual.

Other New Features

■ DRIVER.SYS

This new device driver provides the ability to access a physical drive by referencing a logical drive. It allows you to copy files to and from the same disc drive. It also loads support for external disc drives that don't come with their own software drivers. (Typically, HP external disc drives come with their own software drivers.)

■ STACKS

This new configuration command overrides the default number of stack frames and the default size of each stack frame.

Other Revised Features

■ COUNTRY

Three new country codes have been added to this configuration command:

002 (Canadian-French)

351 (Portugal)

785 (Middle-East)

USER'S REFERENCE

© 1995 Hewlett-Packard Company



Manual Part No. 45951-90077

Table of Contents

USER'S REFERENCE

Notice.....	1
Printing History	1

MS-DOS 3.2 Features

New MS-DOS Commands	1
Revised MS-DOS Commands.....	2
Other New Features.....	2
Other Revised Features.....	3

Chapter 1: Introduction

What is MS-DOS?	1-1
What is in This Manual	1-1

Chapter 2: Understanding MS-DOS Concepts

What is a File?	2-1
Data And Program Files.....	2-2
Naming a File	2-3
Valid Filenames and Extensions	2-4
Reserved Filenames and Extensions.....	2-7
Wildcard Characters	2-10

File Attributes.....	2-11
Organizing Files	2-12
Hierarchical Directory Structure	2-12
The Root Directory	2-13
Subdirectories	2-13
File Allocation Tables (FATs)	2-15
Active Drive and Current Directory	2-15
Active Drive.....	2-15
Current Directory.....	2-16
Files, Directories, and Paths.....	2-17

Chapter 3:

MS-DOS Command Basics

What is an MS-DOS Command?.....	3-1
Internal Commands	3-1
External Commands.....	3-1
The MS-DOS Command Line	3-2
The MS-DOS Prompt.....	3-2
Constructing A Command Line	3-3
Executing a Command Line	3-5
MS-DOS Commands.....	3-6
File-Oriented Commands	3-6
Disc-Oriented Commands.....	3-7
Peripheral Commands.....	3-9
System Commands	3-9

Chapter 4:

MS-DOS Command Descriptions

Introduction	4-1
Command Descriptions	4-1
Sample Directory Structure	4-6
ASSIGN.....	4-7
ATTRIB.....	4-9
BACKUP.....	4-12
BREAK	4-17

CHDIR.....	4-19
CHKDSK	4-21
CLS	4-25
COMMAND	4-26
COMP.....	4-30
COPY	4-34
CTTY	4-41
DATE.....	4-43
DEL.....	4-46
DIR	4-49
DISKCOMP	4-54
DISKCOPY	4-57
ERASE.....	4-60
EXE2BIN	4-63
EXIT	4-67
FC	4-68
FDISK	4-75
FIND.....	4-83
FOR150.....	4-86
FORMAT	4-88
GRAFTABL	4-94
GRAPHICS	4-96
JOIN	4-99
KEYBxx.....	4-103
LABEL.....	4-106
MKDIR	4-109
MODE	4-112
MORE	4-120
PATH	4-122
PRINT	4-125
PROMPT.....	4-131
RECOVER.....	4-135
RENAME	4-139
REPLACE	4-141
RESTORE.....	4-146
RMDIR	4-149
SELECT.....	4-151
SET	4-156
SHARE	4-159
SORT	4-162

SUBST	4-165
SYS	4-170
TIME	4-172
TREE	4-175
TYPE	4-178
VER.....	4-180
VERIFY.....	4-181
VOL.....	4-183
XCOPY	4-185

Chapter 5:

Batch Processing

What is Batch Processing?.....	5-1
Valid Batch Filenames.....	5-2
Creating a Batch File.....	5-2
Executing a Batch File.....	5-3
Chaining Batch Files.....	5-5
The AUTOEXEC.BAT File.....	5-6
Batch Files with Replaceable Parameters	5-6
Using %0 - %9	5-7
Using the SET Command	5-9
Batch Commands	5-10
ECHO.....	5-11
Syntax.....	5-11
Operation	5-11
Notes	5-13
FOR.....	5-14
Syntax.....	5-14
Operation	5-14
Notes	5-15
GOTO.....	5-16
Syntax.....	5-16
Operation	5-16
Notes	5-17
IF	5-18
Syntax.....	5-18
Operation	5-18
Notes	5-19

PAUSE.....	5-20
Syntax.....	5-20
Operation	5-20
Notes.....	5-21
REM.....	5-22
Syntax.....	5-22
Operation	5-22
SHIFT.....	5-23
Syntax.....	5-23
Operation	5-23

Chapter 6:

System Configuration

Overview.....	6-1
Configuration Commands	6-2
BREAK	6-3
Syntax.....	6-3
Operation	6-3
Notes.....	6-4
BUFFERS.....	6-5
Syntax.....	6-5
Operation	6-5
Notes.....	6-6
COUNTRY	6-7
Syntax.....	6-7
Operation	6-8
Notes.....	6-8
DEVICE	6-9
Syntax.....	6-9
Operation	6-9
ANSI.SYS	6-10
DRIVER.SYS.....	6-11
VDISK.SYS.....	6-14

FCBS.....	6-18
Syntax.....	6-18
Operation	6-18
If File-Sharing is Loaded	6-19
If File-Sharing isn't Loaded	6-19
Notes.....	6-19
FILES.....	6-20
Syntax.....	6-20
Operation	6-20
Notes.....	6-20
LASTDRIVE.....	6-22
Syntax.....	6-22
Operation	6-22
SHELL.....	6-23
Syntax.....	6-23
Operation	6-23
Notes.....	6-24
STACKS	6-25
Syntax.....	6-25
Operation	6-25
Notes.....	6-26
Device Drivers.....	6-27
Installable Device Drivers	6-28
Installable Character Device Drivers	6-28
Installable Block Device Drivers	6-30

Chapter 7:

Redirecting Input and Output

Standard Input and Output Devices.....	7-1
Redirecting Input and Output.....	7-2
Redirecting Input	7-3
Redirecting Output	7-4
Piping Input and Output.....	7-5
Using Filters	7-7
Tips on Using Redirection, Piping, and Filters.....	7-8

Chapter 8:

EDLIN

Introduction to EDLIN.....	8-1
How To Start EDLIN.....	8-2
Information Common To All EDLIN Commands....	8-4
Examples	8-5
EDLIN Syntax	8-7
EDLIN Command Parameters	8-8
EDLIN Commands.....	8-10
A (Append).....	8-11
Syntax	8-11
Operation	8-11
Example	8-12
C (Copy).....	8-13
Syntax	8-13
Operation	8-13
Examples	8-14
D (Delete)	8-16
Syntax	8-16
Operation	8-16
Examples	8-17
Edit	8-20
Syntax	8-20
Operation	8-20
Example	8-21
E (End).....	8-23
Syntax	8-23
Operation	8-23
I (Insert).....	8-24
Syntax	8-24
Operation	8-24
Examples	8-25
L (List)	8-27
Syntax	8-27

Operation	8-27
Examples	8-28
M (Move)	8-31
Syntax	8-31
Operation	8-31
Example	8-32
P (Page)	8-33
Syntax	8-33
Operation	8-33
Example	8-33
Q (Quit)	8-34
Syntax	8-34
Operation	8-34
R (Replace)	8-35
Syntax	8-35
Operation	8-35
Example	8-37
S (Search)	8-39
Syntax	8-39
Operation	8-39
Examples	8-41
T (Transfer)	8-43
Syntax	8-43
Operation	8-43
Example	8-44
W (Write)	8-45
Syntax	8-45
Operation	8-45
EDLIN Error Messages	8-46

Chapter 9:

MS-LINK

What MS-LINK is and What it Does	9-1
Definition of Terms	9-4
How MS-LINK Combines and Arranges Segments	9-7
Files that MS-LINK Uses	9-12
Input Files	9-13
Output Files	9-14
Virtual Memory File (VM.TMP File)	9-15

Running MS-LINK.....	9-16
Working With MS-LINK.....	9-16
Using the Text Prompt Method.....	9-17
Using the Command Line Method.....	9-17
Using the Response File Method.....	9-20
Creating a Response File.....	9-21
Command Prompts.....	9-23
Object Modules [.OBJ]	9-25
Run File [First-Object-filename.EXE]:	9-25
List File [NUL.MAP]:.....	9-26
Libraries [.LIB]:.....	9-26
Command Characters	9-28
The Plus-Sign	9-28
The Semicolon.....	9-29
CONTROL C	9-30
Optional Switches.....	9-30
The /DSALLOCATE Switch.....	9-32
The /HIGH Switch	9-33
The /LINENUMBERS Switch	9-34
The /MAP Switch	9-34
The /PAUSE Switch	9-36
The /STACK:<number> Switch.....	9-36
Error Messages.....	9-38
MS-LINK Examples	9-44
Example 1:	9-44
Example 2:	9-45
Example 3:	9-46
Example 4:	9-47
Example 5:	9-48
Example 6:	9-49
Example 7:	9-51
Example 8:	9-52
Example 9:	9-53
Example 10:.....	9-54
Example 11:.....	9-56
Example 12:.....	9-57
Example 13:.....	9-59
Example 14:.....	9-60

Chapter 10:

DEBUG

Introduction	10-1
DEBUG Commands.....	10-1
DEBUG Command Parameters.....	10-3
Memory Addressing on Your Computer.....	10-6
How to Start DEBUG	10-9
DEBUG Command Descriptions.....	10-11
A (Assemble)	10-12
Syntax.....	10-12
Operation	10-12
C (Compare).....	10-15
Syntax.....	10-15
Operation	10-15
Examples	10-15
D (Dump)	10-16
Syntax.....	10-16
Operation	10-16
Examples	10-16
E (Enter)	10-18
Syntax.....	10-18
Operation	10-18
Examples	10-19
F (Fill).....	10-20
Syntax.....	10-20
Operation	10-20
Examples	10-20
G (Go).....	10-21
Syntax.....	10-21
Operation	10-21
Examples	10-22
H (Hex).....	10-24
Syntax.....	10-24
Operation	10-24

Examples	10-24
I (Input)	10-25
Purpose	10-25
Syntax	10-25
Operation	10-25
Examples	10-25
L (Load)	10-26
Syntax	10-26
Operation	10-26
Examples	10-27
M (Move)	10-28
Syntax	10-28
Operation	10-28
Examples	10-28
N (Name)	10-29
Syntax	10-29
Operation	10-29
Examples	10-31
O (Output)	10-32
Syntax	10-32
Operation	10-32
P (Procedure)	10-33
Syntax	10-33
Operation	10-33
Q (Quit)	10-34
Syntax	10-34
Operation	10-34
Examples	10-34
R (Register)	10-35
Syntax	10-35
Operation	10-35
Examples	10-37
S (Search)	10-39
Syntax	10-39
Operation	10-39
Examples	10-39
T (Trace)	10-40
Syntax	10-40

Operation	10-40
Examples	10-40
U (Unassemble).....	10-42
Syntax	10-42
Operation	10-42
Examples	10-42
W (Write)	10-44
Syntax	10-44
Operation	10-44
Debug Error Messages	10-46

Appendix A: MS-DOS Message Directory

Disc and Device Errors.....	A-1
MS-DOS Command Messages.....	A-4

Appendix B: The MS-DOS Keyboard

Editing a Command Line.....	B-1
Other Useful Functions.....	B-5
Canceling a Command.....	B-5
Closing a File	B-6
Logging the Screen.....	B-6
Pausing the Screen	B-6
Printing the Screen	B-7
Unused Keys.....	B-7

Appendix C:

Extended Screen and Keyboard Control (ANSI.SYS)

Control Sequence Syntax	C-2
Control Sequences	C-3
Cursor Position	C-4
Syntax	C-4
Example	C-4
Cursor Up	C-5
Syntax	C-5
Example	C-5
Cursor Down	C-6
Syntax	C-6
Example	C-6
Cursor Forward	C-7
Syntax	C-7
Example	C-7
Cursor Backward	C-8
Syntax	C-8
Example	C-8
Horizontal and Vertical Position	C-9
Syntax	C-9
Example	C-9
Cursor Position Report	C-10
Syntax	C-10
Example	C-10
Device Status Request	C-11
Syntax	C-11
Example	C-11
Save Cursor Position	C-12
Syntax	C-12
Example	C-12
Restore Cursor Position	C-13
Syntax	C-13
Example	C-13

Erase in Display	C-14
Syntax	C-14
Example	C-14
Erase in Line	C-15
Syntax	C-15
Example	C-15
Set Graphics Rendition (SGR)	C-16
Syntax	C-17
Example	C-17
Set Mode	C-18
Syntax	C-18
Example	C-19
Reset Mode	C-20
Examples	C-20
Keyboard Key Reassignment	C-21
Syntax	C-21
Examples	C-21

Appendix D: Disc and Disc Drive Compatibility

Appendix E: Character Codes and Keystrokes

Appendix F: The SORT Command Collating Sequences

INDEX

1

Introduction

What is MS-DOS?

MS-DOS stands for "MicroSoft Disc Operating System". It is the operating system available with HP computers.

MS-DOS is the most widely used operating system for 16-bit personal computers. Literally thousands of applications are designed to run in the MS-DOS environment, enabling you to perform virtually any task with your HP computer.

MS-DOS performs two vital functions for you:

- **Application Program Support.** MS-DOS is used by applications for basic data file operations and character input and output.
- **Disc, File, and System Management.** MS-DOS has a full range of commands and utilities to perform the disc and file management operations for your computer and peripherals.

What is in This Manual

This manual is written for programmers and users who want to use MS-DOS for system and application management. It is designed to be a reference tool, and as such, it assumes some familiarity with computer systems and terminology.

This User's Reference is divided into ten chapters, each of which addresses a different set of information.

■ **Chapter 1: *Introduction***

Defines MS-DOS and explains in detail what is in this manual.

■ **Chapter 2: *Understanding MS-DOS Concepts***

Describes the MS-DOS concepts of files, directories, and paths.

■ **Chapter 3: *MS-DOS Command Basics***

Covers creating an MS-DOS command line and provides information you must know to use MS-DOS commands.

■ **Chapter 4: *MS-DOS Command Descriptions***

Lists the MS-DOS commands in alphabetical order. Each command description includes the complete syntax, typical uses, examples, and notes.

■ **Chapter 5: *Batch Processing***

Explains MS-DOS Batch Processing. Topics include creating a batch file, the AUTOEXEC.BAT file, and detailed descriptions of the batch commands.

■ **Chapter 6: *System Configuration***

Describes the MS-DOS Configuration commands. You also learn how to install device drivers in order to add system components.

■ **Chapter 7: *Redirecting Input and Output***

Documents the MS-DOS capabilities to redirect and pipe input and output, and the three MS-DOS filters, FIND, SORT, and MORE.

■ Chapter 8: *EDLIN*

Discusses EDLIN, the MS-DOS line editor.

■ Chapter 9: *MS-LINK*

Documents the MS-DOS object module linker (used primarily by programmers to link program modules into one program).

■ Chapter 10: *DEBUG*

This final chapter provides complete documentation on DEBUG, the program debugging tool provided by MS-DOS.

APPENDICES. Includes supplementary information on MS-DOS System and Error Messages, the MS-DOS Keyboard, Extended Screen and Keyboard Control for the ANSI device driver, Discs and Disc Drive Compatibility, Character Codes and Keystrokes, and the SORT Command Collating Sequences.

INDEX. An alphabetical listing of subjects with page number references.

Understanding MS-DOS Concepts

In this chapter we discuss basic MS-DOS concepts: files and filenames, directories and subdirectories, and MS-DOS commands directly related to these topics. This material, as well as that contained in the chapter entitled "MS-DOS Command Basics", provides the foundation for our later discussions of MS-DOS.

File and disc management is at the heart of MS-DOS. We therefore begin this introduction to MS-DOS and its capabilities by exploring files.



What is a File?

If you look up the word "file" in a computer dictionary, you'll find it defined as a collection of related records, such as names and addresses, treated as a unit and stored on a disc. A file on a disc is very much like a file in a filing cabinet. Both contain data and both must have an identifier (or filename) so you can locate them.

Each time you use your computer to write a letter or create a spreadsheet, the text from your letter or data from your spreadsheet is held in your computer's internal memory while you are working on it. However, this information is erased when you turn your computer off *unless* you save it as a file on a disc. Discs hold information whether your computer's power is on or not.

If you save a file on a disc, you can later have your computer copy the file back into its internal memory. There, you can add information to the file, change the information that is there, or delete information. When you're finished, you can save the new version of the file with a new filename or you can overwrite the old version of the file by using the same filename.

Data And Program Files

Most application programs use data files to store information. Word processors store text in data files, spreadsheets store numbers and equations, accounting programs store financial information, etc. In short, data files are an integral part of almost any application program.

Although the actual format of data within files varies from application to application, MS-DOS does not concern itself with the contents of the files. Its commands are designed to operate on the files themselves. Thus, a file containing spreadsheet data and one containing word processing text are indistinguishable to MS-DOS.

A program is a set of instructions that tells the computer how to perform a certain task. Programs are also stored on discs in files.

In this manual, when a file is referred to, it can be either a data or program file, unless stated otherwise. The design of the MS-DOS file structure allows both types of files to be treated almost identically.

Naming a File

Whether a file contains data or a program, it must have a name. A name consists of two parts: a filename and an optional filename extension. The filename can be up to 8 characters in length, the filename extension up to 3 characters. The filename and extension must be separated by a period (.). A filename can't consist of an extension only, such as .BAK; if an extension is used, it must be preceded by a filename. Figure 2-1 outlines the requirements of a filename.

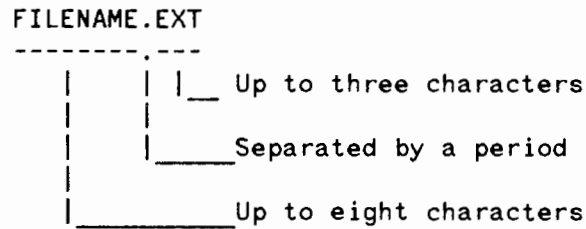


Figure 2-1. MS-DOS Filenames

Selecting proper filenames is fundamental to organizing data on your disc. Some application programs, such as word processors, place the entire responsibility of selecting a filename with the user. Other programs ask the user to provide a filename, then append a specific extension automatically. Still other programs name their data files without consulting you. For example, most accounting programs name their data files internally, rather than asking the user for a name.

Valid Filenames and Extensions

While the MS-DOS file naming conventions allow the use of most characters in filenames and extensions, there are some restrictions.

Table 2-1 lists the characters which **can** be used in MS-DOS filenames and extensions; these characters are valid.

Table 2-2 lists the characters which **cannot** be used in MS-DOS filenames and extensions; these characters are invalid.

Table 2-1. Valid Characters for Filenames & Extensions

Char	Description
A-Z	Alpha characters
a-z	Alpha characters
0-9	Numeric characters
\$	Dollar sign
&	Ampersand
#	Pound sign
%	Percent sign
'	Apostrophe
!	Exclamation point
()	Parentheses
-	Hyphen
___	Underscore
@	"At" sign
^	Carat
{ }	Braces
~	Tilde
'	Single quotation

Table 2-2. Invalid Characters for Filenames & Extensions

Char	Description
.	Period
[]	Brackets
?	Question mark
\	Backslash
/	Forward slash
=	Equal sign
"	Quote
,	Comma
+	Plus Sign
*	Asterisk
:	Colon
;	Semicolon
	Space
< >	Angle Brackets
	Vertical Bar

Note



A period can be used only to separate a filename and an extension; it cannot be used within either.

The question mark and asterisk are interpreted as "wildcard" characters by MS-DOS (see the section entitled *Wildcard Characters* in this chapter).

Reserved Filenames and Extensions

In addition to invalid characters that cannot be used in filenames and extensions, there are certain "reserved" filenames and extensions that cannot be used. These reserved filenames and extensions are used by MS-DOS, exclusively. If you use a reserved filename or extension, MS-DOS will either ignore it or display an error message.

Table 2-3 lists the MS-DOS reserved filenames and table 2-4 lists the MS-DOS reserved extensions.

Note



Some application programs also have filenames and extensions that shouldn't be used. Be sure to check the manuals that come with your application programs for additional reserved filenames and extensions.

Table 2-3. MS-DOS Reserved Filenames

Filename	Function
AUX	Auxiliary device.
CLOCK\$	This is a special purpose device used to read or set the system time and date. Unlike the other device names, CLOCK\$ cannot be used in an MS-DOS command.
COM1	Serial Port 1.
COM2	Serial Port 2.
COM3	Serial Port 3.
COM4	Serial Port 4.
CON	Console. This device name represents the keyboard for input and the screen for output. If used as an input filename in an MS-DOS command, press F6 , then Enter to signify the end of input.
LPT1	Parallel Port 1. This device supports output only.
LPT2	Parallel Port 2. This device supports output only.
LPT3	Parallel Port 3. This device supports output only.

Vectra MS-DOS 3.2 Manual Update Packet

This packet contains update pages for manuals in both the *Vectra MS-DOS Volume I* and *Volume II* binders. Open this update packet and insert these pages where they belong in each manual. Discard the pages that they replace because they do not apply to your computer.

Contents of This Packet

This packet is divided into the following three sections:

- Update pages for the *Installing Your Operating System* manual which is located in the *Volume I* binder.
- Update Pages for the *Using PAM* manual which is also located in the *Volume I* binder.
- Update pages for the *User's Reference* manual which is located in the *Volume II* binder.

Printed in Singapore 8/87
Part Number 45951-90173

Table 2-3. MS-DOS Reserved Filenames (Cont.)

Filename	Function
NUL	Dummy device. When used as an input device, it immediately returns an end-of-file condition. As an output device, it accepts data, but immediately discards it (i.e., no data is written).
PRN	Printing Device.

Table 2-4. MS-DOS Reserved Extensions

Extension	Function
.BAK	Backup file created by EDLIN (the MS-DOS editor) and several word processing programs.
.BAT	MS-DOS uses this extension for all batch files.
.CHK	The extension assigned files that are recovered with the CHKDSK command.
.COM	MS-DOS program file.
.EXE	MS-DOS program file.
.MAP	The default extension for list files created by MS-LINK.

Table 2-4. MS-DOS Reserved Extensions (Cont.)

Extension	Function
.MSG	Applications from Hewlett-Packard use this extension for message files.
.OVL	MS-DOS uses this extension for overlay files.
.REC	MS-DOS assigns this extension for files that are recovered with the RECOVER command.
.SYS	This extension is typically used for device drivers.
.\$\$\$	MS-DOS uses this extension for temporary files.

Wildcard Characters

The last topic pertaining to the naming of a file is wildcard characters. Wildcards are used with many MS-DOS commands to reference multiple files with one filename or extension. Wildcards are introduced here, and also demonstrated later in this manual when MS-DOS commands are discussed.

There are two wildcard characters: the question mark (?) and the asterisk (*). The ? wildcard means "substitute any single valid character".

The * wildcard character operates in a similar manner. It is interpreted by MS-DOS to mean "fill the remaining characters in the filename or extension with the ? wildcard".

The * wildcard is quite flexible. It can be used with fixed characters in either the filename or extension. One common use of this wildcard is the global filename "*. *". This means "all files", and it can be used with many MS-DOS commands. For example, the command line:

```
A>COPY *.* B:
```

copies all of the files in the current directory on drive A: to the current directory on drive B:.

File Attributes

File attributes are special characteristics assigned to individual files. Each MS-DOS file can be assigned up to four file attributes. Because file attributes are assigned on an individual file basis, one file might have two file attributes, while another file might not have any file attributes. The four file attributes are explained below.

Read-only	This attribute marks a file as read-only. Files with this attribute cannot be erased or updated. However, they can be renamed. The ATTRIB command modifies the read-only attribute.
Hidden	This attribute marks a file as Hidden. Files with the hidden attribute are normally excluded from directory searches by MS-DOS and most application programs. Your MS-DOS Master Disc contains three hidden files: the two MS-DOS files IBMBIO.COM and IBMDOS.COM plus the PAM file PAM.CIF. Any attempt to use an MS-DOS command on a hidden file causes an error message similar to the

following to appear:

File Not Found

System	This attribute is similar to the hidden attribute. All files with this attribute are excluded from directory searches by most application programs.
Archive	This attribute is used by BACKUP , RESTORE , and XCOPY to perform incremental file back-ups. This attribute is set by MS-DOS anytime a file is updated and cleared by BACKUP and XCOPY .

Organizing Files

MS-DOS allows you to organize the files on your disc into one or more directories, depending on your needs. For example, you may want all of your accounting files in one directory, all of your **BASIC** program files in another directory, and so on. Any one directory can contain a number of files, and it can also contain other directories. This method of organizing your files is called a Hierarchical Directory Structure.

Hierarchical Directory Structure

A Hierarchical Directory Structure can be pictured as an "inverted tree", with the root at the top of the tree, the directories as the branches, and the files as the leaves. The root directory is the first, and therefore, topmost level in the directory structure. The root directory is created automatically when you format a disc.

The remainder of the inverted tree grows as you create new directories for groups of files or for other users on

your system. Within each new directory, files can be added and new directories (called subdirectories) can be created.

The Root Directory

Each disc has one Root Directory. As mentioned, the root directory is created automatically when you format a disc. Once you format a disc, you can add files and create new subdirectories in the root directory. However, there is a limit to the number of directory entries (files and subdirectories) that the root directory can contain. This limitation is based on the type of disc media. The maximum number of directory entries for each of the available disc types is shown in Table 2-5.

Table 2-5. Directory Entries in Root Directory

Disc	Capacity	# of Entries
Flexible	360 Kb	112
Flexible	720 Kb	112
Flexible	1.2 Mb	224
Flexible	1.44 Mb	224
Hard	20 Mb	512
Hard	40 Mb	512

Subdirectories

On discs containing many files, the root directory may not be sufficient to adequately organize your data. Thus, MS-DOS provides the ability to add additional directories, called Subdirectories, to your disc to solve this problem.

Subdirectories are similar to the root directory. A subdirectory has a directory entry for each of its files and subdirectories. (This helps MS-DOS in locating and accessing files.) However, unlike the root directory, there is no limit to the number of directory entries (files and subdirectories) that a subdirectory can have.

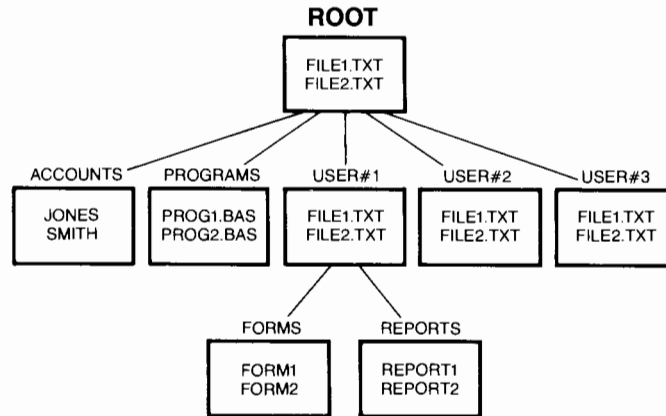


Figure 2-2. MS-DOS Directory Structure

As part of the hierarchical directory structure, subdirectories are "nested". That means that when you create a subdirectory, you create it within a directory that already exists. The end result is a series of directory levels. Each level indicates how far away a subdirectory is from the root directory. For example, in Figure 2-2, the first level contains the root directory. The second level contains the subdirectories **ACCOUNTS**, **PROGRAMS**, **USER#1**, **USER#2**, and **USER#3**, which are all nested in the root directory. The third level contains the subdirectories **FORMS** and **REPORTS**, which are both nested in the subdirectory **USER#1**.

Subdirectories use the same naming convention as files: an eight character name and an optional three character extension, separated by a period. Subdirectories are created with the **MKDIR** command.

File Allocation Tables (FATs)

MS-DOS uses File Allocation Tables (FATs) to keep track of (or manage) files. When information is added to a file, disc space is allocated to that file in units called "clusters". The size of a cluster is dependent on the size of the disc, but generally it's 1-8 Kb. If a file uses all the space in a cluster, another cluster is allocated to it. However, adjacent parts of a file are not necessarily stored in physically adjacent clusters. Therefore, MS-DOS creates a File Allocation Table on each disc to locate all the clusters which comprise each file on the disc. Because the integrity of the FAT is so important to the integrity of all the files on a disc, MS-DOS creates two copies of the FAT on each disc.

MS-DOS provides a command to check the integrity of the FATs on a disc: the CHKDSK command. When you execute the CHKDSK command, it reports any problems and attempts to fix them. As a general rule, it's a good idea to run CHKDSK periodically to make sure the two FATS are intact.

Active Drive and Current Directory

Now that we've discussed files and directories, we need to define two terms: Active Drive and Current Directory. These terms are used extensively throughout this manual.

Active Drive

MS-DOS assigns one of the disc drives as the Active Drive. In order to locate data or a program, MS-DOS must not only know the name of the file, but also the drive on which the file resides. A drive designator can be entered in front of a filename to explicitly tell MS-DOS where the file is located. The following tells MS-DOS that the file FILE1.TXT is located in the root directory on drive C:

C:\FILE1.TXT

The active drive allows the drive designator to be implicitly referenced. If MS-DOS encounters a filename that isn't preceded by a drive designator on a command line, it assumes that the file is on the active drive. If the active drive is A:, MS-DOS interprets:

FILE2.TXT

as

A:\FILE2.TXT

When you start your system, MS-DOS sets the active drive as the drive from which you started your system. For example, if you start your system from flexible disc drive A:, it is the active drive. Starting your system from your hard disc makes C: the active drive.

The default MS-DOS prompt displays the active drive. This reminds you of the active drive as you use your system. The default MS-DOS prompt can be changed with the **PROMPT** command.

The active drive can be changed by entering the drive designator of the new active drive at the MS-DOS prompt. To change the active drive from A: to C:, enter the following command line at the MS-DOS prompt (A>):

C:

MS-DOS responds with the new prompt:

C>

showing C: as the active drive.

Current Directory

The Current Directory is the directory (root directory or subdirectory) that MS-DOS assumes is being referenced, unless a filename is preceded by a path (see the next section in this chapter).

Each drive has a current directory. When you start your system, the current directory for each drive is the root directory. The current directory for a drive can be changed with the CHDIR command.

Files, Directories, and Paths

When you use nested directories, you must tell MS-DOS where the files (or subdirectories) are located in the directory structure. If two users have duplicate filenames, each user must tell MS-DOS which series of subdirectories to search for their file. This is done by giving MS-DOS the Path to the file. The path is literally the route MS-DOS must travel from the root directory to reach a specific file (or subdirectory).

As already stated, the name of a file is a sequence of characters that can optionally be preceded by a drive designator and followed by a filename extension. The path to that file is the sequence of subdirectory names separated by backslashes (\).

Let's use an example from Figure 2-2 to illustrate paths. The root directory doesn't have a name (unlike subdirectories), and therefore is noted only with the backslash character (\). Suppose you ask MS-DOS to locate the following file:

`C:\USER#1\REPORTS\REPORT1`

You're telling MS-DOS that REPORT1 is located on the disc in drive C:. In addition, the path to REPORT1 from the root directory of drive C: is \USER#1\REPORTS.

Note



The backslash character (\) serves a dual purpose: it indicates the root directory and it separates subdirectories from each other and from files.

Part of the path can be "implicitly" referenced. If the current directory is USER#1, you need only supply the missing portion of the path (\REPORTS) to get to the file REPORT1.

There isn't a limit on the depth to which subdirectories can be nested; however, a path can't be longer than 64 characters. This limit on path size includes the entire path from the root directory, even if part of the path is implicitly referenced.

If you attempt to reference a file or subdirectory using a path in excess of 64 characters, the following error message is displayed:

Invalid Directory

3

MS-DOS Command Basics

In this chapter, we'll examine the basics of MS-DOS commands. We'll discuss the types of MS-DOS commands and the MS-DOS command line, then we'll list the MS-DOS commands by function.



What is an MS-DOS Command?

There are two types of MS-DOS commands: internal and external. Internal commands reside in system memory and are always available for use. External commands, on the other hand, are stored on disc as files. External commands are really no different than your application programs except that they manage your files and system rather than process words, create spreadsheets, etc.

Internal Commands

The MS-DOS internal commands are part of the larger, executable file called `COMMAND.COM`. They are available any time the MS-DOS prompt is displayed on the screen.

External Commands

External commands operate the same as internal commands, with the exception that they must first be loaded into memory from disc. In order to execute an external MS-DOS command, it must be on one of the discs in the system. External MS-DOS commands have one of two possible filename extensions: `.COM` or `.EXE`.

All of the MS-DOS commands are listed at the end of this chapter and described in detail in the chapter entitled "MS-DOS Command Descriptions".

The MS-DOS Command Line

MS-DOS commands are entered via a command line. In this section we'll examine the MS-DOS command line in detail.

The MS-DOS Prompt

MS-DOS commands can only be entered when the MS-DOS prompt is displayed on the screen. The standard MS-DOS prompt consists of the active drive designator followed by the greater than (>) character. If the active drive is A:, the MS-DOS prompt is:

A>

MS-DOS has the capability of displaying the prompt in different formats; the example above is only the default. The **PROMPT** command can be used to modify the prompt to include the time, date, current directory, MS-DOS version number, or any characters you want. For example, the **PROMPT** command could produce the following prompt:

A:\USER#1>

which indicates the current directory.

Refer to the chapter entitled "MS-DOS Command Descriptions" for specifics on the use of the **PROMPT** command.

Throughout this manual, the examples show the default prompt.

Constructing A Command Line

An MS-DOS command line consists of several distinct parts. To illustrate the components of a command line, we'll use the **FORMAT** command, as shown in Figure 3-1.

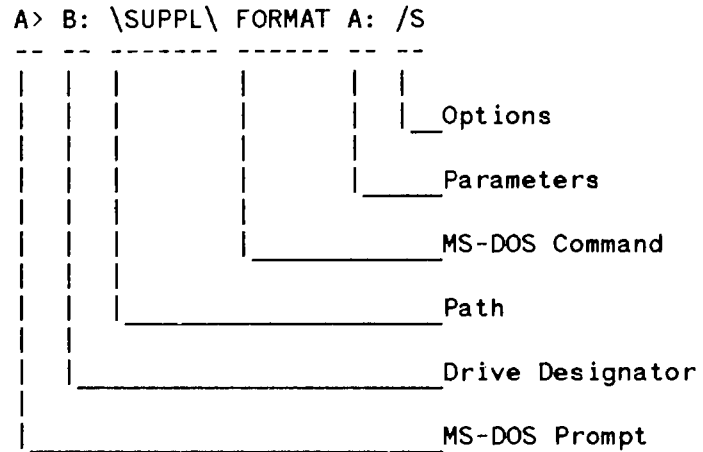


Figure 3-1. MS-DOS Command Line

Let's briefly review each of the components.

- The MS-DOS prompt must be displayed to indicate that MS-DOS is ready to accept and execute a command line.
- The drive designator can be included to tell MS-DOS which drive contains the command file. In the above example, drive B: contains the **FORMAT** command.

- The path can be included to tell MS-DOS the path to the command file. In the above example, the path to the **FORMAT** command is **\SUPPL**. The path is not allowed with internal commands. If it is specified, the following error message is displayed:

Bad command or filename

If no path is included on a command line for an external command, MS-DOS searches the accessible directories to find it. Accessible directories are the current directory, plus any additional directories specified with the **PATH** command. For more information, refer to the **PATH** command in the chapter entitled "MS-DOS Command Descriptions".

- The MS-DOS command follows the optional drive designator and path. This is simply the name of the MS-DOS command.
- The last two components, parameters and options, tell the command what you want accomplished. The parameters (or arguments) depend on the command, but typically contain filename, drive designator, path, etc. Many MS-DOS commands support options (or switches). They consist of the forward slash character (/) and a letter.

The command line can be typed in upper- or lower-case characters, or a mix of both.

Executing a Command Line

After a command is typed on the command line, it is executed by pressing **Enter**.

Many MS-DOS commands prompt you during execution, and some ask for additional parameters. When responding, type information in the format required for the command. Several commands have "safety checks" built into them. For example, if you enter the command line:

```
DEL *.*
```

(erase all files from the current directory), DEL makes sure by asking:

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

If you enter "Y", the command executes. If you enter "N" or any other character, the command terminates and the MS-DOS prompt reappears. This allows you to double check potentially damaging commands before they are executed.

At times, you may be instructed to perform a task and then notify MS-DOS when you've completed it. For example, FORMAT instructs you to insert the disc you want to format. You are asked to press any key when ready to indicate you have completed the task. You should press the space bar, **Enter**, or any of the alphanumeric keys (A-Z, 0-9).

MS-DOS Commands

In this section, the MS-DOS commands are briefly outlined by their functions. A detailed description of each command appears in the chapter entitled "MS-DOS Command Descriptions".

File-Oriented Commands

Name	Description
ATTRIB	Changes a file's archive and read-only attributes.
BACKUP	Backs up files from one disc to another.
COMP	Compares two files.
COPY	Copies a file.
DEL	Removes a file from a disc.
DIR	Performs a directory listing.
ERASE	Removes a file from a disc.
EXE2BIN	Converts a file from .EXE to binary object file format.
FC	Compares two files.
FIND	Searches one or more files for a specified search string.
MORE	Displays output one screen at a time.

PATH	Sets alternate directory searches for a command or program.
RECOVER	Recovers damaged files or directories from a disc.
RENAME	Renames a file.
REPLACE	Updates previous versions of files.
RESTORE	Restores files to a disc after they have been backed up with the BACKUP command.
SHARE	Loads MS-DOS support for shared files in a network environment.
SORT	Sorts data.
TREE	Lists all subdirectories on a disc by path.
TYPE	Displays the contents of a file on the standard output device.
XCOPY	Copies files and directories, including lower-level directories, if they exist.

Disc-Oriented Commands

Name	Description
ASSIGN	Assigns one logical disc drive to another.
CHDIR	Changes the current directory for a disc.

CHKDSK	Checks the validity of a disc, and reports disc and system memory statistics.
DISKCOMP	Compares the contents of two different discs.
DISKCOPY	Copies the contents of one disc to another.
FDISK	Manages MS-DOS and non-MS-DOS partitions on a hard disc.
FOR150	Formats a disc for use by MS-DOS on an HP150 computer.
FORMAT	Formats a disc for use by MS-DOS.
JOIN	Allows a disc drive to be logically connected to another disc drive via a subdirectory.
LABEL	Places a volume label on a disc, or changes an existing volume label.
MKDIR	Creates a subdirectory.
RMDIR	Removes a subdirectory.
SELECT	Installs MS-DOS on a new disc with the keyboard layout, date and time format you select.
SUBST	Assigns a subdirectory to a drive designator.
SYS	Transfers a copy of MS-DOS to a disc.
VOL	Displays the volume label of a disc.

Peripheral Commands

Name	Description
CTTY	Changes the physical device assigned to the Standard Input and Output devices.
GRAFTABL	Loads additional ASCII characters for use by a color/graphics adapter card.
GRAPHICS	Loads support for the printing of display screens in the Graphics mode on the system printer.
KEYBxx	Provides support for non-U.S. language keyboards.
MODE	Configures video, parallel, and serial adapter cards for use with your computer.
PRINT	Prints the contents of a file(s) as a background task.

System Commands

Name	Description
BREAK	Enables or disables extended CTRL Break and CTRL C checking by MS-DOS.
CLS	Clears the display screen.

COMMAND	Loads a second version of the MS-DOS command processor.
DATE	Displays and/or sets the current system date.
EXIT	Exits the current version of the MS-DOS command processor and returns to the previous version.
PROMPT	Specifies the format of the prompt.
SET	Sets or displays parameters in the MS-DOS environment.
TIME	Displays and/or sets the current system time.
VER	Returns the current version number of MS-DOS.
VERIFY	Enables or disables disc write verification by MS-DOS.

forward slash followed by a single letter (for example, /P). In most cases, you can specify multiple options in a command line.

argument additional information that frequently requires you to make a choice (for example, between ON and OFF).

The following symbols are used when describing command syntax in this chapter:

CAPS A word in capital letters is a "keyword" that represents a command or an argument that must be entered exactly as shown.

[] A word enclosed by square brackets is optional.

<> A word enclosed by angle brackets represents data you must enter. For example, <filename> means that you must replace <filename> with the name of your file on the command line. If a word is enclosed by square and angle brackets (for example, [<filename>]), it is optional. However, if you choose to specify it on the command line, you must replace <filename> with the name of your file.

{ } A word enclosed by braces indicates that you have a choice between two or more entries. At least one of the entries must be selected unless the entries are also enclosed by square brackets.

- | A vertical bar indicates an argument which means you must make a choice (for example, ON|OFF). When used with an MS-DOS filter, the vertical bar indicates a pipe. (See the chapter entitled "Redirecting Input and Output" for additional information on Filters and Piping.)
- ... An ellipsis indicates that an entry can be repeated as many times as needed or desired.

All other punctuation, such as commas, colons and equal signs, must be entered exactly as shown.

Commands take effect when you press **Enter**.

Information Common to All MS-DOS Commands

The following information is common to all MS-DOS commands:

- The backspace key **←** is a delete key. If you make an error while entering a command, simply backspace over the incorrect character(s). Always read the syntax on a command line carefully before pressing **Enter**.
- Commands can be followed by one or more parameters. Commands and parameters can be entered in upper- or lower-case letters, or any combination of the two.
- Commands and parameters must be separated by delimiters. Because they are easiest, the space and the comma are the most frequently used delimiters. For example:

```
COPY FILE1.OLD FILE1.NEW  
Del,File1.Old File1.New
```

Notice that both the space and the comma were used in the second command. You can also use the semicolon (;), the equal sign (=), and the tab key (**Tab**) as delimiters. The space is used as the delimiter in this manual.

- When MS-DOS commands instructs you to **Press any key when ready...**, you can press any alphanumeric key (A-Z or 0-9).
- When referring to a file, if its filename includes a filename extension, you must specify the filename and filename extension on the command line.
- Wildcards cannot be used in command names.
- In some commands, drive designators are referred to as *source* drives and *target* drives. The source drive is the drive from which you will be reading information, and the target drive is the drive to which you will be writing information.
- Commands can be aborted while they are executing by holding down **CTRL** and then pressing **C**.

Sample Directory Structure

The sample directory structure shown below is used in examples throughout this chapter. Refer to this sample directory structure if you need help understanding any of the command examples.

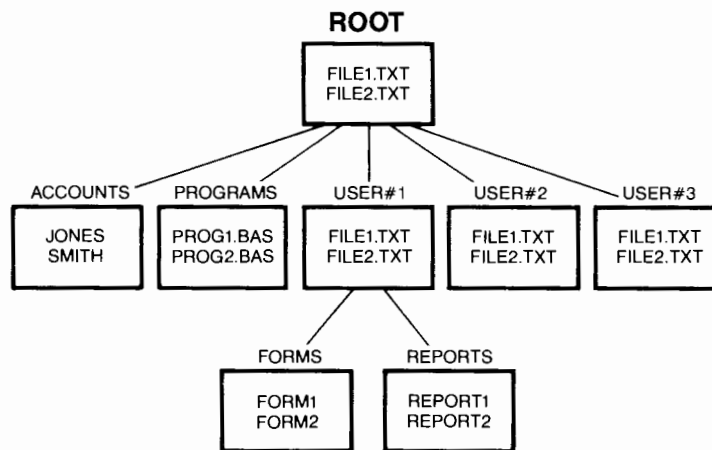


Figure 4-1. Sample Directory Structure

ASSIGN

The ASSIGN command reassigns all disc read and write requests from one drive to another drive.

Type

ASSIGN is an EXTERNAL command.

Syntax

[<d>:][<path>]ASSIGN [<d1>[=<d2> [...]]

- | | |
|------|----------------------------------------------------------------------------------------------------------------------|
| d: | the drive that contains the ASSIGN command. |
| path | the path to the ASSIGN command. |
| d1 | the drive to which read and write requests are currently sent. These requests are to be reassigned to another drive. |
| d2 | the drive to which read and write requests are to be sent. |

Operation

The ASSIGN command is used to instruct MS-DOS to reassign all disc operations from one disc drive to another. This is useful if you have an application program that was originally designed to run only on specific drives (for example, A: and B:), yet you want to run the application program on another drive (for example, C:). The command line:

ASSIGN A=C

redirects all disc read and write requests for drive A: to drive C:. If you issue the command:

DIR A:

you now see the display of the directory on drive C:. If you want to redirect the disc operations from both drives A: and B: to drive C:, enter the command:

ASSIGN A=C B=C

To cancel all drive reassignments, issue the **ASSIGN** command without parameters.

Notes

1. Notice that the above examples do not use a colon after the drive designator. If you enter the colon, the following error message is displayed:

Invalid parameter

2. The **ASSIGN** command should only be used with application programs. Also, all drive reassignments should be canceled before you use the following commands: **BACKUP**, **JOIN**, **LABEL**, **PRINT**, **RESTORE** and **SUBST**.
3. The **DISKCOMP** and **DISKCOPY** commands ignore any drive reassignments.
4. The **FORMAT** command issues an error message if you attempt to format an assigned drive.
5. Only resident drives can be assigned. For example, on a system with two flexible drives (A: and B:), a hard drive (C:), and a virtual drive created by **VDISK** (D:), you can assign drives A: through D:. However, if you attempt to assign drive E:, **ASSIGN** issues an error message.

ATTRIB

The ATTRIB command sets or clears the read-only and archive attributes for one or more files. ATTRIB can also be used to determine the current setting of the read-only and archive attributes for a specific file or set of files.

Type

ATTRIB is an EXTERNAL command.

Syntax

```
[<d>:][<path>]ATTRIB [+R|-R] [+A|-A]  
[<d1>:][<path1>]  
    <filename1>
```

d:	the drive that contains the ATTRIB command.
path	the path to the ATTRIB command.
+R	the option that sets the read-only attribute of a file.
-R	the option that clears the read-only attribute.
+A	the option that sets the archive attribute of a file.
-A	the option that clears the archive attribute.
d1:	the drive that contains the file to be changed.

path1 the path to the file to be changed.

filename1 the file whose attribute is to be changed.

Operation

The ATTRIB command allows you to mark a file so that it can't be changed (modified) or deleted. It also allows you to mark a file for archiving by other commands, such as BACKUP and XCOPY.

The file FILE1.TXT can be protected with the command line:

```
ATTRIB +R FILE1.TXT
```

To return the file to its original state so it can be changed or deleted, enter the command line:

```
ATTRIB -R FILE1.TXT
```

To protect a file and mark it as archived, enter the command line:

```
ATTRIB +R +A FILE1.TXT
```

The ATTRIB command without parameters returns the current state of the read-only and archive attributes for the specified file. To determine the state of a group of files, wildcards can be used. For example, the command line:

```
ATTRIB *.TXT
```

results in the following listing:

```
R A  C:\FILE1.TXT
      C:\FILE2.TXT
R    C:\FILE3.TXT
A    C:\FILE4.TXT
```

Files with the letter "R" in the first column are read-only files and files with the letter "A" in the second column are archived files. Notice that the path to each file is displayed in the listing.

Notes

1. If you use the COPY command to copy read-only files, the read-only attribute isn't passed on to the new files.
2. The BACKUP command with the /M option and the XCOPY command with the /M option are both affected by the archive attribute. If the archive attribute is set for a specified file, that file is copied; if it isn't set, that file isn't copied.
3. If an application opens a file with read and write permission in a network environment, this command enforces a read-only mode to permit file-sharing over the network.

BACKUP

The **BACKUP** command makes backup copies of the files on a disc. The backup copies are created and stored on another disc.

Type

BACKUP is an **EXTERNAL** command.

Syntax

```
[<d>:][<path>]BACKUP <d1>:[<path1>][<filename1>]  
                        <d2>:[/S][/M][/A]  
                        [/D:<mm-dd-yy>]
```

d:	the drive that contains the BACKUP command.
path	the path to the BACKUP command.
d1:	the source drive. It contains the file(s) to be backed up.
path1	the path to the file(s) to be backed up.
filename1	the name(s) of the file(s) to be backed up. If you don't specify a filename, all the files in the specified directory are backed up.
d2:	the target drive. The drive that you want to contain the backed up files. The source drive and the target drive cannot be the same.

/S	the option that backs up the files in all subdirectories of the specified directory, in addition to the files in the specified directory.
/M	the option that backs up only those files that have been modified since the last BACKUP occurred.
/A	the option that adds the files to be backed up to those files already present on the backup disc. It does not erase any old files that are already present on the backup disc.
/D	the option that backs up only those files that have been modified on or after a certain date. The date format is dependent on the country code you select using the COUNTRY command in the CONFIG.SYS file. <mm-dd-yy> is the date format for the U.S.

Operation

The BACKUP command backs up files from one disc to another. It displays the name of each file as it is being backed up. Each backup disc should be labeled and numbered consecutively. This procedure helps you to properly restore the backed up files since the RESTORE command prompts you to insert the backup discs in the correct order.

The BACKUP command is not the same as the COPY command which makes an exact duplicate of a file. Files that have been backed up have control data included that the RESTORE command uses. Therefore, you cannot use backed up files as data files until they have been restored. Refer to the RESTORE command description in this chapter for further information.

In its most commonly used form, the **BACKUP** command copies files from a hard disc onto one or more flexible discs. For example, to back up all the files in the root directory of the hard disc in drive C: to a series of flexible discs in drive A:, enter the following command line:

```
BACKUP C: A:
```

You can specify a certain directory or even an individual file to backup, such as:

```
BACKUP C:\LETTERS A:
```

where **LETTERS** is either a subdirectory or an individual file.

BACKUP supports a set of options to provide additional flexibility. These options are **/S**, **/M**, **/A** and **/D**. The **/S** option backs up the files in all the subdirectories of the specified directory, in addition to the files in the specified directory. If the **/S** option is omitted, only the files in the specified directory are backed up. The **/S** option is commonly used because it's required if you want to backup an entire disc that contains subdirectories. To backup the entire contents of the hard disc in drive C: (including subdirectories) to a series of flexible discs in drive A:, enter the following command line:

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

The **/M** option only copies those files that have been modified since the last **BACKUP** occurred. This is referred to as an "incremental" backup. This option reduces the number of files that are copied.

The **/A** option adds the files being copied to the files already on the backup disc. If this option isn't specified, all existing files on the backup disc are erased before any new files are added.

The /D option copies only those files that have been modified on or after the specified date (<mm-dd-yy>). This is another form of an incremental backup.

The BACKUP command returns several error codes. The error codes and their descriptions are listed below.

- | | |
|---|---------------------------------------------------------------------------------------------|
| 0 | Normal Completion |
| 1 | No files were found to back up |
| 2 | Some files not backed up due to file-sharing conflicts |
| 3 | Terminated by user <input type="button" value="CTRL"/> <input type="button" value="Break"/> |
| 4 | Terminated due to error |

These error codes can be used with the ERRORLEVEL option of the batch processing IF command. Refer to the chapter entitled "Batch Processing" for additional information.

Notes

1. BACKUP accepts wildcards in the filename. This allows you to backup groups of files. To backup all the files that have a filename extension of .TXT in the root directory on drive C: to drive A:, enter the command line:

```
BACKUP C:\*.TXT A:
```

2. All flexible discs must be formatted before you can use them with the BACKUP command. These discs should be formatted without the /S option since the system files are erased by BACKUP (unless the /A option is specified with BACKUP).

3. If the target drive contains a hard disc, the backup files are automatically placed in a subdirectory called \BACKUP. If the target drive contains a flexible disc, the backup files are automatically placed in the root directory.
4. If the source drive contains a flexible disc, the flexible disc should not be write-protected because BACKUP needs to mark the files as being backed up (/M).
5. The archive attribute, which can be set with the ATTRIB command, affects the /M option of the BACKUP command. If the archive attribute is set for a specified file, that file is backed up; if it isn't set, that file isn't backed up.
6. DO NOT use the BACKUP command on a drive that has been ASSIGNed, JOINed, or SUBSTituted. These commands hide the true drive designator that BACKUP uses.

BREAK

The BREAK command extends **CTRL Break** and **CTRL C** checking to include all MS-DOS operations. Normally, MS-DOS only checks for a **CTRL Break** or **CTRL C** during keyboard input, screen output, printer output, and auxiliary input and output.

Type

BREAK is an INTERNAL command.

Syntax

BREAK [ON|OFF]

- ON this causes MS-DOS to check for **CTRL Break** and **CTRL C** during all MS-DOS operations.
- OFF this causes MS-DOS to check for **CTRL Break** and **CTRL C** only when it is performing keyboard, screen, printer, or auxiliary input and output operations. OFF is the default setting.

Operation

When BREAK is set to ON, **CTRL Break** and **CTRL C** can be used to stop the execution of an MS-DOS command or an application program that performs few or no keyboard, screen, printer, or auxiliary input and output operations.

To set extended checking, enter the command line:

BREAK ON

To cancel extended checking, enter the command line:

BREAK OFF

To determine whether extended checking is in effect, enter **BREAK** without a parameter. MS-DOS then displays the current state of the extended checking.

Notes

1. When **BREAK** is set to **ON**, system performance may be slower due to frequent checking. Therefore, a setting of **OFF** is recommended.
2. The **BREAK** command can be included in the system configuration file (**CONFIG.SYS**). To turn extended checking **ON**, include the command:

BREAK ON

anywhere in the file.

3. Some programs bypass MS-DOS for character and disc operations. When these programs are running, a **CTRL Break** or **CTRL C** might not interrupt them, even though you have issued the **BREAK ON** command.

CHDIR (Change Directory)

The CHDIR command changes the current directory on the active or specified drive. The CHDIR command can also be used to display the path to the current directory.

Type

CHDIR is an INTERNAL command.

Syntax

CHDIR [**<d>** :] [**<path>**]

d: the drive that contains the new directory.

path the path to the new directory.

Operation

The CHDIR command changes the current directory. If the optional drive designator is omitted, the current directory on the active drive (also referred to as the "working" directory) is changed. If the optional drive designator is included, the current directory on an inactive drive is changed.

CHDIR entered without parameters displays the active drive and the path from the root directory to the current directory. For example, if you enter the command line:

CHDIR

and MS-DOS displays the following information:

C:\USER#1\REPORTS

this tells you that the active drive is C:, the current directory on the active drive is **REPORTS**, and the path to the current directory from the root directory is **\USER#1\REPORTS**.

To change the current directory from **REPORTS** to **USER#1**, enter:

```
CHDIR \USER#1
```

or use the shorthand notation for "moving up" one directory:

```
CHDIR ..
```

Using the **CHDIR** command with two dots (..) when you're in a subdirectory moves you up to that subdirectory's parent directory. This same command entered from the root directory (which has no parent) results in the following error message:

```
Invalid directory
```

To change the current directory back to **REPORTS** from **USER#1**, enter the command:

```
CHDIR REPORTS
```

To return to the root directory from any subdirectory, enter the command:

```
A>CHDIR \
```

Notes

1. **CD** is an alternate form of the **CHDIR** command. The command line **CD \USER#1\REPORTS** is the same as the command line **CHDIR \USER#1\REPORTS**.

CHKDSK (Check Disc)

The CHKDSK command tests the integrity of a disc or certain file(s) on a disc. Specifically, it scans the directory of the specified disc, checks it for consistency, and displays a status report. CHKDSK also reports information about files on the disc and system memory. Finally, if errors are encountered, CHKDSK can be used to correct some, but not all, of them.

Type

CHKDSK is an EXTERNAL command.

Syntax

```
[<d>:][<path>]CHKDSK [<d1>:][<path1>]  
[<filename1>][ /F ][ /V]
```

d:	the drive that contains the CHKDSK command.
path	the path to the CHKDSK command.
d1:	the drive that contains the disc or file(s) to be checked.
path1	the path to the file(s) to be checked.
filename1	the file(s) to be checked.
/F	the option that instructs MS-DOS to fix any errors, if possible.
/V	the option that instructs MS-DOS to display the names of all files and subdirectories.

Operation

The CHKDSK command is used to determine the status of a disc or file(s). The command line:

```
CHKDSK C:
```

instructs CHKDSK to examine the disc in drive C:. CHKDSK cross-checks the root directory with all of the subdirectories and the two copies of the File Allocation Table (FAT). If any errors are found, an error message is displayed. The error messages, along with the recommended steps to correct them, are listed in the appendix entitled "MS-DOS Message Directory" at the back of this manual.

When CHKDSK completes its inspection, it displays several statistics about the disc, its files, and system memory. An example of this listing is shown below:

```
Volume HARDDISC                      Created AUG 12, 1987

10592256 bytes total disk space
   40960 bytes in 2 hidden files
   45056 bytes in 11 directories
  4435968 bytes in 391 user files
   24576 bytes in bad sectors
  6045696 bytes available on disk

   655360 bytes total memory
   220000 bytes free
```

This is only a representative listing of a CHKDSK display. The entries for volume, hidden files, directories, user files, and bad sectors won't be listed if they don't exist on the disc.

If you specify a filename instead of a disc drive, CHKDSK determines how the file is stored on disc. In addition to the information shown above, it reports the number of non-contiguous blocks contained in the file. You'll see the message:

```
All specified file(s) are contiguous
```

or

```
<filename> contains <n> non-contiguous  
blocks
```

This information is important because system performance is degraded if the number of fragmented files on a disc (particularly a hard disc) grows too large.

Wildcards can be used in the filename to determine the status of more than one file at a time. However, CHKDSK only examines the files in one directory--either the current directory or the one specified with the optional path.

CHKDSK supports two options: /F and /V. The /F (fix) option instructs CHKDSK to correct any errors it finds, if possible. If CHKDSK is able to fix an error, it asks you if you want the error fixed before correcting it. You can answer "Yes" or "No", but you must have specified the /F option to actually correct the errors.

The /V (view) option instructs CHKDSK to display the names of all the files in the root directory and the subdirectories. This information is, in effect, a "diagram" of your hierarchical directory structure.



Notes

1. CHKDSK won't allow you to specify a path without a filename on the command line because it attempts to interpret the path as a file. For example, if you want to determine the amount of fragmentation in the files contained in the subdirectory USER#1, do not enter the command line:

```
CHKDSK C:\USER#1
```

CHKDSK incorrectly interprets this command to mean, "report on the file USER#1 in the root directory" and returns the error message:

```
File not found
```

Instead, enter the command line:

```
CHKDSK C:\USER#1\*.*
```

and CHKDSK executes correctly.

2. You can redirect CHKDSK's output to a device or file for later use. However, you can't redirect the output and use the /F option simultaneously. See the chapter entitled "Redirecting Input and Output" for more information on redirecting command output.
3. When CHKDSK encounters lost clusters, it asks you if you want them recovered. If you enter "Yes" and the /F option has been specified, CHKDSK places each recovered allocation chain (one or more clusters) in a file with the name FILEnnnn.CHK.

The nnnn portion of the filename is a number starting with "0000". These files are placed in the root directory. You can look at them to see if the information recovered is of any value, and erase them if not.

CLS (Clear Screen)

The CLS command moves the cursor to the upper left corner of the display screen and then clears the entire screen.

Type

CLS is an INTERNAL command.

Syntax

CLS

Operation

To clear the screen of all characters, enter the command line:

CLS

COMMAND

This command loads a second version of the MS-DOS command processor, COMMAND.COM, into system memory.

Type

COMMAND is an EXTERNAL command.

Syntax

```
[<d>:][<path>]COMMAND [<d1>:][<path1>][ /P]
                               [/E:<nnnn>]
                               [/C<command>]
```

d:	the drive that contains the COMMAND command.
path	the path to the COMMAND command.
d1:	the drive that contains the COMMAND.COM file if MS-DOS needs to reload the transient part into memory.
path1	the path to the COMMAND.COM file if MS-DOS needs to reload the transient part into memory.
/P	the option that instructs MS-DOS to make the second version of COMMAND.COM permanent. This version can only be removed by restarting your system.

<code>/E:nnnnn</code>	the option that instructs MS-DOS to set the size of the MS-DOS environment. <code>nnnnn</code> is a base 10 integer from 160 to 32768 that is rounded up to the nearest paragraph boundary. The default value is 160.
<code>/C</code>	the option that tells the second version of the command processor to execute the MS-DOS command (either internal or external) following the <code>/C</code> parameter. Then, return to the original version of the command processor. If both the <code>/C</code> and <code>/P</code> options are used, the <code>/P</code> option is ignored.
command	the MS-DOS internal or external command to be executed under the <code>/C</code> parameter.

Operation

The `COMMAND` command loads a second version of the command processor, `COMMAND.COM`, into system memory.

The original version of `COMMAND.COM` contains all the MS-DOS internal commands. This file resides on your MS-DOS disc. Its contents are loaded into memory every time you load your operating system. When `COMMAND.COM` is in memory, part of it is in resident memory and part of it is in transient memory.

If sufficient memory is available, you can load a new, second version of `COMMAND.COM` by entering the following command line:

`COMMAND`

This second command processor "inherits" the MS-DOS environment of the original command processor (the two environments are the same).

At this point, if the SET command is used to change the MS-DOS environment, only the MS-DOS environment used by the second command processor is affected. The MS-DOS environment used by the original command processor remains unchanged. Then, when you EXIT the second command processor and return to the original, the environment used by the original command processor resumes operation.

If the /P option is specified, the second command processor becomes permanent. That is, the EXIT command can't be used to return to the original command processor. The /P option increases the resident size of MS-DOS in memory.

If the /C option is specified, the second command processor is only active during the execution of a specific command. Then, control is returned to the original command processor. For example, if you enter the following command line from drive C:

```
COMMAND /C DIR A:
```

MS-DOS loads the second command processor. Then, the second command processor executes the command DIR A: and exits back to the original command processor.

Notes

1. The /C option, if used, must be the last option specified on the command line.

2. **COMMAND** can be used in a batch file to avoid "chaining" (refer to the chapter entitled "Batch Processing" for further information on chaining batch files). For example, if you create a batch file called **BATCH#1.BAT** that contains the following information:

```
\COMMAND /C BATCH#2.BAT  
\COMMAND /C BATCH#3.BAT
```

when you execute this batch file, a number of things happen. First, MS-DOS loads the new, second command processor. The second command processor executes **BATCH#2.BAT** and exits to the original command processor. Then, MS-DOS loads the second command processor again. It executes **BATCH#3.BAT** and exits to the original command processor for the final time.

COMP (Compare)

The **COMP** command compares the contents of two files and reports any differences.

Type

COMP is an **EXTERNAL** command.

Syntax

```
[<d>:][<path>]COMP [<d1>:][<path1>][<filename1>]  
[<d2>:][<path2>][<filename2>]
```

d:	the drive that contains the COMP command.
path	the path to the COMP command.
d1:	the drive that contains the first of two files to be compared.
path1	the path to the first of two files to be compared.
filename1	the first of two files to be compared.
d2:	the drive that contains the second of two files to be compared.
path2	the path to the second of two files to be compared.
filename2	the second of two files to be compared.

Operation

The COMP command compares the files specified on the command line, and reports any differences. For example, the command line:

```
COMP C:\USER#1\FILE1.TXT B:\FILE2.TXT
```

compares the contents of the file FILE1.TXT in the subdirectory USER#1 on drive C: with the file FILE2.TXT in the root directory on drive B:. If the filenames are not entered on the command line, COMP prompts for them as follows:

```
Enter primary file name
```

```
Enter 2nd file name or drive id
```

If the files are identical, the message:

```
Files compare OK
```

is displayed. If differences are discovered, they are displayed in the following format:

```
Compare error at OFFSET xxxxx  
File1 = yy  
File2 = zz
```

where xxxxx is the byte offset from the start of the files, yy is the hex value of the byte in the first file, and zz is the hex value of the byte in the second file.

After ten differences are found, COMP stops comparing the files, displays the following message:

```
10 Mismatches - ending compare
```

and then displays the prompt:

Compare more files (Y/N)?

If you enter "N", you are returned to the MS-DOS prompt. If you enter "Y", COMP prompts you for the names of the new files as follows:

Enter primary file name

Enter 2nd file name or drive id

After you type a response to these prompts, COMP compares the files.

Notes

1. If the two files do not contain the same number of bytes (as indicated in their respective directory entries), COMP doesn't compare them. Instead, the following error message is displayed:

Files are different sizes

2. Wildcards can be used with COMP to compare more than two files. For example, the command line:

COMP *.TXT *.BAK

compares all the files with an extension of .TXT with all the files with an extension of .BAK. The names of the files are listed as they are compared.

3. If the first filename is omitted (only a drive designator and/or a path are specified), **COMP** assumes the first filename is **"*.***". For example, if you enter the command line:

```
COMP A: C:\USER#1\FILE1.TXT
```

COMP compares every file in the root directory of drive A: with the file **FILE1.TXT** in the subdirectory **USER#1** on drive C:.

4. If the second filename is omitted, **COMP** assumes it is the same as the first filename.
5. If a file isn't terminated with an end-of-file marker (**CTRL** **(Z)**), the following message is displayed:

```
EOF mark not found
```

However, **COMP** still compares the two files.

COPY

The COPY command is used to copy one or more files from one disc or directory to another. COPY can also be used to make a copy of a file in the same directory, but with a different name, or to combine two or more files into one file.

Type

COPY is an INTERNAL command.

Syntax

To copy a file:

```
COPY [/A][/B]
      [<d1>:][<path1><filename1>[/A][/B]
      [<d2>:][<path2><filename2>
      [/A] [/B] [/V]
```

/A the option that treats a file like an ASCII file. When this option is specified for a source file, the file is copied up to the first end-of-file marker (**CTRL Z**). The remainder of the file, if any, isn't copied. When this option is specified for a target file, an end-of-file marker is added to the file.

/B the option that treats a file like a Binary file. When this option is specified for a source file, the entire file, based on the directory file size, is copied (including the end-of-file marker). When this option is specified for a target file, no end-of-file marker is added to the file.

d1: the source drive. It contains the source file (the file to be copied from).

path1 the path to the source file.

filename1 the name of the source file.

d2: the target drive. It contains the target file (the file to be copied to).

path2 the path to the target file.

filename2 the name of the target file.

/V the option that verifies the target file as it is written. This option performs the same function as the VERIFY command, except that it only works while the COPY command is executing.

To combine (concatenate) two or more files into one file:

```
COPY [/A][/B][<d1>:][<path1>]<filename1>[/A][/B]
      [+<d2>:][<path2>]<filename2>[/A][/B]... ]
      [<d n>:][<path n>]<filename n>[/A][/B][/V]
```

/A the option that treats a file like an ASCII file. When this option is specified for a source file, the file is combined up to the first end-of-file marker (CTRL Z). The remainder of the file, if any, isn't combined. When this option is specified for a target file, an end-of-file marker is added to the file.

/B	the option that treats a file like a binary file. When this option is specified for a source file, the entire file, based on directory file size, is combined (including the end-of-file marker). When this option is specified for a target file, no end-of-file marker is added to the file.
d1:	the first source drive. It contains the first source file (the first file to be combined).
path1	the path to the first source file.
filename1	the name of the first source file.
d2:	the second source drive. It contains the second source file (the second file to be combined).
path2	the path to the second source file.
filename2	the name of the second source file.
d n	the target drive. It contains the target file (the new file to contain the combined files).
path n	the path to the target file.
filename n	the name of the target file. If the target filename is omitted, the source files are combined under the first source filename.
/V	the option that verifies the target file after it is written. This option performs that same function as the VERIFY command, except that it only works while the COPY command is executing.

Note



The position of the /A and /B options on the command line determines the files affected by them. The /A and /B options affect the file immediately preceding them and any files following them until another /A or /B is encountered.

Operation

Copying a File. The COPY command copies the source file(s) to the desired target (or destination). If the target filename isn't included on the command line, COPY assumes the target file has the same name as the source file. For example, to copy the file FILE1.TXT from the active drive C: to drive B: and keep it under the same name, enter the command line:

```
COPY FILE1.TXT B:FILE1.TXT
```

or

```
COPY FILE1.TXT B:
```

In either case, a duplicate of FILE1.TXT is created on drive B:. However, MS-DOS won't let you make a duplicate of a file with the same filename in the same directory. If you attempt to do this, the following error message is displayed:

```
File cannot be copied onto itself
0 File(s) copied
```

To make a duplicate of a file in the same directory, you must give the target file a new name. For example, the command line:

```
COPY FILE1.TXT FILE2.TXT
```

copies the file FILE1.TXT to FILE2.TXT in the same directory.

Both the source and target can be extended with drive designators and paths to operate on files in directories other than the current directory. The following command line copies FILE1.TXT from the subdirectory USER#1 on drive C: to the subdirectory REPORTS on drive A:.

```
COPY C:\USERS\USER#1\FILE1.TXT A:\REPORTS
```

Wildcards can be used to copy groups of files. To copy all the files from the current directory of drive C: to drive A:, enter the command line:

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

COPY can be used to transfer data between system devices. You can copy information from a device, such as your console (terminal), to another device, such as a file on a disc. A valid device name can be used as the source and/or the target filename. For example, the following command line:

```
COPY CON FILE1.TXT
```

allows you to enter data directly into a file called FILE1.TXT from your console. Lines are typed one at a time and are entered by pressing the **Enter** key. When you are finished entering data into the file, hold down **CTRL** and press **Z**, then press **Enter** to write the file out to your disc.

Note



The COPY CON command can be used to create batch files. See the chapter entitled "Batch Processing" for an example using COPY CON to create a batch file.

In another example, the command line:

```
COPY FILE2.TXT PRN
```

copies the disc file **FILE2.TXT** to the printer configured as the **PRN** device.

Combining (Concatenating) Two or More Files. **COPY** can also be used to combine two or more files into one file. A good example of the usefulness of this form of the command is a manuscript that contains many chapters. Each chapter is stored in its own separate file for ease of use. If you need to combine them all, **COPY** accomplishes this with the following command:

```
COPY CHAP1.TXT+CHAP2.TXT+CHAP3.TXT BOOK.TXT
```

The individual chapters (files) are combined into a new file called **BOOK.TXT**. Any or all of the source and target filenames can be preceded by an optional drive designator and/or path.

If you don't want to combine the files into a new file, omit the target filename. Then, the files are combined into the first source file. For example, the command line:

```
COPY CHAP1.TXT+CHAP2.TXT+CHAP3.TXT
```

combines the individual chapters into the first source file, **CHAP1.TXT**.

Wildcards can be used with **COPY** to combine files. The first example above could have been entered as:

```
COPY CHAP?.TXT BOOK.TXT
```



Notes

1. When **COPY** is used to copy a file, the source and target files are treated as **Binary** if no option is specified. When **COPY** is used to combine files, the files are treated as **ASCII** if no option is specified.
2. The attributes of the source file (for example, read-only) are not transferred to the target file.
3. If you use wildcards with the **COPY** command, make sure that the target filename is **NOT** included in the source filename reference. For example, the command line:

```
COPY *.TXT BOOK.TXT
```

violates this restriction. An error will occur; however, it won't be detected until after the file **BOOK.TXT** is destroyed!

CTTY

The CTTY command allows you to change the standard input and output device (CON) to another valid I/O (input and output) device.

Type

CTTY is an INTERNAL command.

Syntax

CTTY <device>

device the device you want to specify as the standard I/O device.

Operation

The CTTY command allows any valid character I/O device to be used as the MS-DOS Console. The device can be any standard MS-DOS I/O device, such as AUX, COM1, COM2, COM3, and COM4. Or, it can be an optional character device driver installed with the DEVICE command in the CONFIG.SYS file.

The command line:

CTTY AUX

moves all command input and output from the current device (the console) to the device configured as AUX (for example, another terminal).

To return command input and output to the console, enter the following command from the AUX device:

CTTY CON

Notes

1. Many programs do not use MS-DOS for standard input and output routines. These programs bypass the software and access (communicate with) the hardware directly, or use ROM BIOS I/O routines. The CTTY command has no effect on these programs.
2. The assigned device must be a character I/O device. In addition, it must be capable of receiving input and sending output. Therefore, a device such as PRN (a printer) cannot be used as the assigned device because it can't send output to MS-DOS.

DATE

The DATE command is used to set or display the system date.

Type

DATE is an INTERNAL command.

Syntax

DATE [<mm>-<dd>-<yy>]

mm a 1- or 2-digit number from 1 to 12 representing the month.

dd a 1- or 2-digit number from 1 to 31 representing the day of the month.

yy a 2-digit number from 80 to 99 representing the year (century 19 is assumed), or a 4-digit number from 1980 to 2099 representing the year.

A hyphen (-), slash (/), or period (.) can be used as a separator.

Operation

The DATE command allows the MS-DOS system date to be set or displayed. MS-DOS uses the date and time values to record the exact time files are created or last updated. These values are affected **ONLY** by modifications to your files. As you build your library of files, you will find this information both necessary and useful.

To set the system date, enter the command line:

```
DATE 10-15-87
```

To display the current system date, enter the command line:

```
DATE
```

MS-DOS responds by displaying the current system date and asking for a new date as follows:

```
Current date is Thu 10-15-87
Enter new date [mm-dd-yy]:
```

You can enter a new system date or press to retain the current system date.

MS-DOS checks for invalid dates and separators. If you enter an invalid date or use an incorrect separator, the system responds with:

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

This response is repeated until you enter a valid date in the correct format.

The DATE command sets the MS-DOS system date, but it doesn't change the date kept by the "system clock". The system clock is a battery-powered clock that keeps track of the date as well as the time. It increments the date and time even when the system is turned off. The system clock resets the system date and time each time you start (or restart) your system. To change the system clock, refer to the *Setting Up* manual that came with your computer.

Notes

1. The DATE display format is determined by the COUNTRY command in the CONFIG.SYS file.
2. MS-DOS is programmed to change months and years correctly, including months with 29, 30, and 31 days, and leap years.
3. If you leave your computer on for 24 hours without using it, the system date may not be updated properly.
4. If you start your system without PAM and an AUTOEXEC.BAT file, you are prompted to enter the date and time every time you start it. However, if you have an AUTOEXEC.BAT file, you aren't prompted for the date and time unless your AUTOEXEC.BAT file includes the DATE and TIME commands.

DEL (Delete)

The DEL command removes unwanted files from a disc.

Type

DEL is an INTERNAL command.

Syntax

DEL [<d>:][<path>]<filename>

Parameters

d:	the drive that contains the file(s) to be deleted.
path	the path to the file(s) to be deleted.
filename	the file(s) to be deleted.

Operation

The DEL command deletes one or more files from a disc. The command line consists of the DEL command, followed by an optional drive designator and/or path, and a filename. If the filename doesn't contain wildcards, a single file is deleted. The command line:

DEL FILE1.TXT

deletes the file FILE1.TXT in the current directory.

The command line:

DEL *.TXT

deletes all of the files with the extension of .TXT in the current directory. To delete all of the files in the current directory, enter the command line:

DEL *.*

As a precaution, MS-DOS asks the following question before executing the command:

Are you sure (Y/N)?

If you enter "N", the command is canceled and the files are left intact. If you enter "Y", all the files in the current directory are deleted.

The DEL command removes unwanted files from a directory, but it doesn't remove the directory, itself. To remove a directory, use the RMDIR (Remove Directory) command described in this chapter. Note that a directory must be empty before it can be removed. Thus, if you want to remove a directory, use the DEL command first to delete all of the files in it; then, use the RMDIR command.

Notes

1. The DEL command is identical in function to the ERASE command.
2. The DEL command can't be used to delete files with hidden or read-only attributes.

3. Specifying a path without a filename is the same as specifying a path with the filename "*.*". Therefore, if you enter the command line:

```
DEL A:\
```

MS-DOS asks you if you're sure and deletes all the files in the root directory if you respond "Y".

DIR (Show Directory)

The DIR command lists the names of the files in a directory.

Type

DIR is an INTERNAL command.

Syntax

DIR [**<d>**:][**<path>**][**<filename>**][**/P**][**/W**]

d: the drive that contains the filename(s) to be displayed.

path the path to the filename(s) to be displayed.

filename the filename(s) to be displayed.

/P the option that instructs MS-DOS to pause when the screen is filled. It is also referred to as Page mode.

/W the option that instructs MS-DOS to list filenames only, five filenames per line. It is also referred to as Wide Display mode.

Operation

The DIR command is used to determine what files are in a certain directory. The DIR command by itself lists the entire contents of a directory. Files are listed with their size in bytes, and the date and time they were last updated. Any subdirectories are also listed, but file size is omitted and the designation **<DIR>** appears after the subdirectory name. The listing concludes with the total

number of files and subdirectories displayed, and the number of free bytes remaining on the disc. For example, the following command issued from the root directory of drive C:

```
DIR
```

results in a listing similar to the following:

```
Volume in drive C is HARDDISC
Directory of C:\

FILE1      TXT      23210   10-07-87   1:43p
ACCOUNTS   <DIR>          10-12-87   8:03a
PROGRAMS   <DIR>          10-12-87   8:05a
USER#1     <DIR>          10-12-87   8:10a
USER#2     <DIR>          10-12-87   8:11a
USER#3     <DIR>          10-12-87   8:13a
FILE2      TXT       4329   10-13-87   3:45p
7 File(s)  6348800 bytes free
```

The contents of a directory are listed in the order in which they were entered in the directory. For this reason, files and subdirectories are displayed initially in the order they were created or modified. Then, as files are deleted, new files fill the vacated directory locations, and the directory is no longer in its original date/time order. The SORT command, described later in this chapter, can be used to display a directory in alphabetical order.

A directory other than the current one can be specified on the command line. For example, if the current directory is the root directory on drive A:, the following command line lists the files and subdirectories in the subdirectory USER#1 on drive C:

```
DIR C:\USER#1
```

A filename can also be specified on the command line. If the filename contains no wildcards, only the file with that name is listed. Wildcards can be used in the filename to list a group of files. For example, the command line:

```
DIR *.TXT
```

lists all the files in the current directory with a filename extension of .TXT.

The DIR command supports two options: the /P (Page) option and the /W (Wide Display) option. The /P option is useful for directories that contain a large number of files. It instructs MS-DOS to pause when 23 lines of data have been displayed. The message:

```
Strike a key when ready...
```

appears at the end of the display. The listing resumes when any key is pressed.

The /W option displays the filenames in wide format. The filenames are displayed left to right, five per line. Information about file size and date/time of last update are omitted. As with other directory listings, the list concludes with the total number of files and the number of free bytes.

To display particularly large directories, the /P and /W options can be specified together to produce a listing in wide format that pauses at the end of every page.

The directory listing for a subdirectory has "special" entries. The following example shows the directory listing for the subdirectory USER#1. The "." entry indicates the current directory, which is subdirectory USER#1, and the ".." entry indicates its parent directory, which is the root directory. In addition, USER#1 has two subdirectories of its own called FORMS and REPORTS.

```
Volume in drive C is HARDDISC
Directory of C:\USER#1
```

```

.           <DIR>      10-12-87  8:10a
..          <DIR>      10-07-87  1:43p
FORMS       <DIR>      10-12-87  8:17a
REPORTS     <DIR>      10-12-87  8:18a
FILE1      TXT        206   10-15-87  3:45p
FILE2      TXT        310   10-17-87  9:28a
6 file(s)  7518208 bytes free
```

Notes

1. The number of free bytes is always a multiple of 1024 bytes.
2. The date and time display format are determined by the COUNTRY command in the CONFIG.SYS file.
3. While you can use wildcards with the DIR command, in many cases they are unnecessary. That's because DIR makes the following assumptions if a filename is specified without a filename extension or a filename extension is specified without a filename:

```
DIR filename = DIR filename.*
DIR .extension = DIR *.extension
```

Thus, to list all the files with the filename **FILE1**, enter the following command line:

DIR FILE1

MS-DOS displays all the files with the name **FILE1**, including the various filename extensions.

To list all the files with an extension of **.TXT**, enter the command line:

DIR .TXT

4. Hidden files are not displayed in any directory listing.

DISKCOMP

The DISKCOMP command compares the contents of two flexible discs, sector by sector.

Type

DISKCOMP is an EXTERNAL command.

Syntax

[<d>:][<path>]DISKCOMP [<d1>: [<d2>:]][/1]/8]

Parameters

d:	the drive that contains the DISKCOMP command.
path	the path to the DISKCOMP command.
d1:	the first of two flexible discs to be compared.
d2:	the second of two flexible discs to be compared.
/1	the option that compares only the first side of the flexible discs, even if they are double-sided discs.
/8	the option that compares only 8 sectors per track, even if the flexible discs have more sectors per track.



Operation

The DISKCOMP command compares the contents of the flexible disc in the first drive with the flexible disc in the second drive. The command line:

`DISKCOMP A: B:`

compares the flexible disc in drive A: with the flexible disc in drive B: on a sector-by-sector basis.

When DISKCOMP starts, it displays the message:

Insert FIRST diskette in drive A:

Insert SECOND diskette in drive B:

Press any key when ready . . .

If a compare error is discovered, the track and sector containing the mismatch are displayed in the following format:

Compare error side xx, sector yy

where xx is the side number (0 or 1) and yy is the sector number.

After the compare operation is complete, DISKCOMP asks if you want to compare more discs with the message:

Compare more diskettes (Y/N)?

If you enter "Y", you are prompted to insert the new discs. If you enter "N", you are returned to the MS-DOS prompt.

Notes

1. DISKCOMP cannot be used on hard or virtual discs.
2. If you omit both drive designators from the command line, DISKCOMP uses only one drive (the active drive) to compare the contents of the two flexible discs.
3. If you omit the second drive designator from the command line, DISKCOMP uses the active drive as the second drive.
4. Two discs are usually identical only if one has been made from the other using the DISKCOPY command. If the COPY command has been used to transfer files from one disc to another, the discs probably won't compare. The reason is that COPY doesn't necessarily put data in the same place on the target drive as it is on the source drive. To compare discs made with the COPY command, refer to the FC and COMP commands.
5. DISKCOMP cannot be used with ASSIGNED, JOINed, or SUBSTITuted drives.
6. If the /1 and /8 options aren't specified, DISKCOMP determines the number of sides and sectors per track based on the format of the source disc.
7. If the target and source discs have been formatted differently, DISKCOMP displays the following error message:

Drive types or diskette types are not compatible

DISKCOPY

The DISKCOPY command copies the contents from one flexible disc to another flexible disc.

Type

DISKCOPY is an EXTERNAL command.

Syntax

[<d>:][<path>]DISKCOPY [<d1>: [<d2>:]] [/1]

d: the drive that contains the DISKCOPY command.

path the path to the DISKCOPY command.

d1: the source drive. It contains the source disc (the disc to be copied from).

d2: the target drive. It contains the target disc (the disc to be copied to).

/1 the option that only copies the first side of the disc, whether the disc is single- or double-sided.

Operation

The DISKCOPY command makes an exact copy of one flexible disc on another flexible disc. The command line:

DISKCOPY A: B:

makes an exact copy of the disc in drive A: on the disc in drive B:.

When DISKCOPY starts, it displays the following message:

Insert SOURCE disk in drive A:

Insert TARGET disk in drive B:

Press any key when ready . . .

When the copy is complete, DISKCOPY asks:

Copy another (Y/N)?

If you enter "Y", DISKCOPY instructs you to insert the next set of discs. If you enter "N", you are returned to the MS-DOS prompt.

Notes

1. DISKCOPY cannot be used on hard or virtual discs.
2. If you omit both drive designators from the command line, DISKCOPY uses only one drive (the active drive) to copy the contents of the two flexible discs.
3. If you omit the second drive designator from the command line, DISKCOPY uses the active drive as the second drive.

4. **DISKCOPY** cannot be used with **ASSIGNed**, **JOINed**, or **SUBSTituted** drives.
5. If the target disc has never been formatted, **DISKCOPY** formats it during the copy process. The target disc is formatted the same as the source disc.
6. If disc errors are encountered on either the source or target disc, **DISKCOPY** reports the track, sector, and size of the error; then, it proceeds with the copy process. The target disc may or may not be usable in this situation. Use **DISKCOMP** or **COMP** to verify the state of the target disc.

ERASE

The ERASE command removes unwanted files from a disc.

Type

ERASE is an INTERNAL command.

Syntax

ERASE [**<d>:**][**<path>**]**<filename>**

d: the drive that contains the file(s) to be erased.

path the path to the file(s) to be erased.

filename the file(s) to be erased.

Operation

The ERASE command erases one or more files from a disc. The command line consists of the ERASE command, followed by an optional drive designator and/or path, and a filename. If the filename doesn't contain wildcards, a single file is erased. The command line:

ERASE FILE1.TXT

erases the file FILE1.TXT in the current directory.

The command line:

```
ERASE *.TXT
```

erases all of the files with the extension of .TXT in the current directory. To erase all the files in the current directory, enter the command line:

```
ERASE *.*
```

As a precaution, MS-DOS asks the following question before executing the command:

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

If you enter "N", the command is canceled and the files are left intact. If you enter "Y", all the files in the current directory are erased.

The ERASE command removes unwanted files from a directory, but it doesn't remove the directory, itself. To remove a directory, use the RMDIR (Remove Directory) command described in this chapter. Note that a directory must be empty before it can be removed. Thus, if you want to remove a directory, use the ERASE command first to erase all of the files in it; then, use the RMDIR command.

Notes

1. The ERASE command is identical in function to the DEL command.
2. The ERASE command can't be used to erase files with hidden or read-only attributes.

3. Specifying a path without a filename is the same as specifying a path with the filename "*.*". Therefore, if you enter the command line:

```
ERASE A:\
```

MS-DOS asks you if you're sure and erases all the files in the root directory if you respond "Y".

EXE2BIN (Executable to Binary)

The EXE2BIN command converts files from executable format (.EXE) to binary format (.BIN or .COM).

Type

EXE2BIN is an EXTERNAL command.

Syntax

```
[<d>:][<path>]EXE2BIN [<d1>:][<path1>]<filename1>  
[<d2>:][<path2>][<filename2>]
```

d:	the drive that contains the EXE2BIN command.
path	the path to the EXE2BIN command.
d1:	the source drive. It contains the source file (the file to be converted).
path1	the path to the source file.
filename1	the name of the source file. This is also referred to as the input file. You must specify a source filename; however, the filename extension is optional. If you don't specify a filename extension, .EXE is used.
d2:	the target drive. It contains the target file (the converted file).
path2	the path to the target file.

filename2 the name of the target file. This is also referred to as the output file. If you don't specify an output filename, the input filename is used. If you don't specify a filename extension, .BIN is used.

Operation

The EXE2BIN command can be used to convert executable files to binary format. This conversion saves disc space and allows programs to load more quickly.

The command line specifies the name of the source file (the file to be converted) and optionally, the name of the target file (the new file once it's been converted). If the target filename isn't specified, EXE2BIN uses the source filename with a filename extension of .BIN as the target filename.

To convert the file MYPROG.EXE from .EXE format to a file called MYPROG.COM in .COM format, enter the following command line:

```
EXE2BIN MYPROG MYPROG.COM
```

Notice that the filename extension of the source file doesn't need to be specified--.EXE is presumed.

Notes

1. The source file must be in valid .EXE format as produced by the linker. The resident (actual) code and data portions of the file must be less than 64 Kb combined, and there must be no STACK segment. If the source file is not in valid .EXE format, EXE2BIN issues the following error message:

File can't be converted

2. EXE2BIN produces two different types of target files depending on whether the initial CS:IP (Code Segment:Instruction Pointer) is specified in the .EXE file. The two types are described below:

- **Binary Conversion:** If CS:IP is not specified in the .EXE file (the .EXE file contains a CS:IP value of 0:0), a pure binary conversion is assumed. This is the format that must be used to produce installable device drivers.

If segment fixups are necessary (if the program contains instructions requiring segment relocation), you are prompted for the fixup value. This value is the absolute segment where the program is to be loaded. As a result, the program is only usable when loaded at the absolute memory address specified by a user application. The MS-DOS command processor will not be able to properly load the program.

- **.COM File Conversion:** 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. No segment fixups are allowed, as .COM files must be segment relocatable; that is, they must assume the entry conditions explained in the *MS-DOS 3.2 Programmer's Reference Manual*. Once the conversion is complete, you can rename the file with a .COM extension. Then, the MS-DOS command processor is able to load and execute the program the same way it loads and executes the .COM programs supplied on your MS-DOS disc.

If CS:IP does not meet either of these criteria, or if it meets the .COM file criterion but has segment fixups, the following message is displayed:

File cannot be converted

EXIT

The EXIT command exits the MS-DOS command processor (COMMAND.COM) and returns you to the previous command processor, if one exists.

Type

EXIT is an INTERNAL command.

Syntax

EXIT

Operation

The MS-DOS command processor, COMMAND.COM, can be loaded by PAM, an application program, or COMMAND.COM, itself. For example, when it's loaded by PAM, the MS-DOS command processor appears as the application MS-DOS COMMANDS. In order to return to PAM, enter the command line:

EXIT

Notes

1. EXIT doesn't perform any operation if a previous command processor doesn't exist.

FC (File Compare)

The FC command compares the contents of two files.

Type

FC is an EXTERNAL command.

Syntax

```
[<d>:][<path>]FC [/A][/B][/C][/L][/LB<n>]  
                [/W][/T][/N][/<nnnn>]  
                [<d1>:][<path1>]<filename1>  
                [<d2>:][<path2>]<filename2>
```

- d:** the drive that contains the FC command.
- path** the path to the FC command.
- /A** the option that abbreviates the output of an ASCII (text) comparison. Instead of displaying all the lines that are different, only the first and last lines of each set of differences are displayed. The lines in between are represented by ellipses (...).
- /B** the option that does a binary comparison of both files, byte-for-byte, with no attempt to re-synchronize after a mismatch. This is the default for files with filename extensions of .EXE, .COM, .SYS, .OBJ, .LIB, and .BIN.
- /C** the option that instructs MS-DOS to ignore

	the case of letters and consider all letters as upper-case. This option should only be used for ASCII comparisons.
/L	the option that does an ASCII comparison of both files. This is the default for files with filename extensions <i>other than</i> .EXE, .COM, .SYS, .OBJ, .LIB, and .BIN.
/LB<n>	the option that sets the internal buffer to <n> during an ASCII comparison. The default length of the internal buffer is 100 lines. <n> has to exceed the number of differing lines by 2. If not, FC aborts and the results of the file comparison are invalid.
/W	the option that compresses the white spaces created by tabs and spaces during the comparison. Thus, multiple contiguous white spaces on a line are considered one white space. Note that although FC compresses white spaces, it doesn't ignore them, except the beginning and ending white spaces on a line. This option should only be used for ASCII comparisons.
/T	the option that doesn't expand tabs to spaces during the comparison. The default is to treat tabs as spaces to 8-column positions.
/N	the option that displays line numbers during an ASCII comparison.
/ <nnnn>	the option that specifies the number of lines that must match after a difference is found for the files to be considered as matching again. That is, FC will continue to list lines that are different between the two files until it encounters <nnnn> number of lines that are the same. The default is 2.

d1:	the drive that contains the first file to be compared.
path1	the path to the first file to be compared.
filename1	the name of the first file to be compared.
d2:	the drive that contains the second file to be compared.
path2	the path to the second file to be compared.
filename2	the name of the second file to be compared.

Operation

The FC command allows you to compare the contents of two files. Two types of comparisons can be made:

- ASCII or text comparison (line-by-line)
- Binary comparison (byte-by-byte)

FC compares the first file to the second and reports any differences between them. Both filenames can have optional drive designators and/or paths attached to them. For example, the command line:

```
FC C:\USER#1\FILE1.TXT B:\FILE2.TXT
```

compares FILE1.TXT in the subdirectory USER#1 on drive C: with FILE2.TXT in the root directory on drive B:.

FC displays the following information on a file comparison:

- the filename
- the last matching line preceding a difference
- the different lines
- the next matching line

Thus, if FILE1.TXT and FILE2.TXT contain the following information:

FILE1.TXT

all
good
things
must
eventually
come
to
some
kind
of
an
end

FILE2.TXT

all
good
things
come
to
some
kind
of
end

the ASCII comparison of the two files shows the following:

```
***** C:\USER#1\FILE1.TXT
things
must
eventually
come
```

```
***** B:FILE2.TXT
things
come
*****
```

```
***** C:\USER#1\FILE1.TXT
of
an
end
```

```
***** B:FILE2.TXT
of
end
*****
```

To force a binary comparison of the two files, enter the following command line:

```
FC /B C:\USER#1\FILE1.TXT B:\FILE2.TXT
```

The results are displayed in three columns of hexadecimal numbers:

First Column	the position (in the files) of each pair of mismatched bytes. The first byte in each file is byte number 0.
Second Column	the values of the mismatched bytes in <filename1>.

Third Column the values of the mismatched bytes in
 <filename2>.

A portion of a binary comparison of the two sample files is shown below. Note that if one file contains more data than the other, a message is displayed at the end of the comparison.

```
00000013: 6d 63
00000014: 75 6f
00000015: 73 6d
.         .  .
.         .  .
.         .  .
```



fc: C:\USER#1\FILE1.TXT longer than B:FILE2.TXT

The following example demonstrates the use of line buffer option (/LB<n>). The line buffer is set to three lines and the two sample files are compared with the command line:

```
FC /LB3 C:\USER#1\FILE1.TXT B:\FILE2.TXT
```

Because the sample files have more consecutive, differing lines than the buffer can hold, the comparison aborts and the following message is displayed:

```
Resync failed. Files are too different.
```

The following example demonstrates the use of the line number option (/<nnnn>). The line number option is set to six lines, and once again the two sample files are compared with the command line:

```
FC /6 C:\USER#1\FILE1.TXT B:\FILE2.TXT
```


In this case, FC doesn't encounter six consecutive matching lines after the first difference is found. So, FC continues to list the lines until it reaches the end of one of the files as shown below.

```
***** C:\USER#1\FILE1.TXT
things
must
eventually
come
to
some
kind
of
an
end
***** B:FILE2.TXT
things
come
to
some
kind
of
end
*****
```

Notes

1. Options placed after filenames are ignored, and the default values are used instead. Incorrect or unrecognized parameters are also ignored and do not prevent the FC command from executing.

FDISK

The FDISK command creates an MS-DOS partition on a hard disc. The MS-DOS partition is necessary for storing the MS-DOS programs on a hard disc. It is also necessary if you want to start your computer from a hard disc and if you want to use MS-DOS as your principal operating system.

FDISK is one step in a multi-step process which includes:

- Running the SETUP program described in the *Setting Up* manual that came with your computer.
- Running FDISK.
- Using the FORMAT command.

FDISK operates slightly differently from other MS-DOS commands. It is menu-driven rather than command-driven. As such, there are no parameters or options to place on the command line.

To access the FDISK Options Menu, type FDISK on the command line. The FDISK Options Menu shown on the following page is displayed.

Fixed Disk Setup Program

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:

Press ESC to return to DOS.

Note



Option 5 (Select Next Fixed Disk Drive) is displayed only if you have more than one internal hard disc. If it is displayed, you must select option 5 before you can create an MS-DOS partition and perform other FDISK operations on internal hard discs other than the one in drive C:.

Creating An MS-DOS Partition

Select option 1 from the FDISK Options Menu to create an MS-DOS partition. FDISK asks you if you want to use the entire hard disc for MS-DOS with the following prompt:

Create DOS Partition

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

The default response is "Y".

CHOOSE "YES":

If your hard disc is 32 Mb or SMALLER (MS-DOS can only access a total of 32 Mb on a hard disc), and you want to use your entire hard disc with the MS-DOS operating system. To choose "Yes", simply press . FDISK creates the partition and displays the following message:

```
System will now reboot
Insert DOS diskette in drive A:
Press any key when ready . . .
```

If necessary, re-insert your MS-DOS disc and press to restart your system.

Your hard disc now has an MS-DOS partition and it is ready to be formatted using the MS-DOS FORMAT command.

CHOOSE "NO":

If your hard disc is GREATER than 32 Mb or you don't want to use the entire hard disc for the MS-DOS operating system (i.e., you want to put another operating system on your hard disc). To choose "No", type "N" and press . Then, read the next section on *Creating an MS-DOS Partition on Part of a Hard Disc*.

Creating an MS-DOS Partition on Part of a Hard Disc

After you type "N" and press , FDISK displays the total disc space in the following format:

```
Total fixed disk space is n cylinders.
Maximum available space is n
Cylinders at n.
```

- The first number is the total number of cylinders on the hard disc.
- The second number is the largest *contiguous* block of cylinders that can fit into the MS-DOS partition. If the hard disc is blank, this number represents up to 32 Mb of cylinders. If there are other partitions already on the disc, this number represents the largest contiguous block of cylinders *not* presently used in a partition.
- The last number is the cylinder where the block of usable cylinders starts.

After displaying the total disc space, FDISK prompts you to:

Enter partition size:[n]

The number displayed is the size of the largest block. If you want to allocate the entire block to the MS-DOS partition, press . Otherwise, type the number of cylinders you want to allocate to the MS-DOS partition and press .

FDISK then prompts you to:

Enter starting cylinder number.....:[n]

Again, the number displayed is the beginning cylinder number of the largest available block discovered by FDISK. Press to select that number, or type one of your own choosing and press .

FDISK now creates the MS-DOS partition. If there's already a non-MS-DOS partition on the disc, it's probably the active partition. If you want to start your system from the hard disc using the MS-DOS operating system, you'll need to change the active partition to the MS-DOS partition. To do this, follow the steps in the next section. Once the MS-DOS partition is the active partition, the hard disc is ready to be formatted using the MS-DOS FORMAT command.

Note



FDISK can ONLY create the MS-DOS partition. If your hard disc is greater than 32 Mb or you want to create partitions for other operating systems, you MUST use the Hard Disc Multi-Voluming Utility described in the *Installing Your Operating System* manual. This utility gives instructions for creating an MS-DOS partition *and* a non-MS-DOS partition.

Changing the Active Partition

Select option 2 from the FDISK Options Menu to change the active partition. A Change Active Partition screen similar to the one shown below is displayed.

Change Active Partition

Partition	Status	Type	Start	End	Size
1	A	DOS	0	769	770
2	N	non-DOS	770	975	206

Total disk space is 976 cylinders

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

Two partitions are listed on this sample screen. Partition 1 is the MS-DOS partition. It is currently the active partition (status = A) and it occupies 770 cylinders on the disc. Partition 2 is a non-MS-DOS partition. It is currently inactive (status = N) and it occupies the remaining 206 cylinders.

In order to change the active partition from Partition 1 to Partition 2, type "2" in response to the prompt and press **Enter**. The non- MS-DOS partition is now active. If the hard disc drive is used as the drive from which you start your system, the operating system in Partition 2 now starts your system instead of MS-DOS.

Deleting the MS-DOS Partition

Select option 3 from the FDISK Options Menu to delete the MS-DOS partition. A Delete DOS Partition screen similar to the one shown below is displayed.

Caution

Deleting the MS-DOS partition destroys all the information stored in it. Be certain that all data and programs have been transferred from the MS-DOS partition to a backup disc(s) before you delete it.

Delete DOS Partition

Partition	Status	Type	Start	End	Size
1	A	DOS	0	769	770
2	N	non-DOS	770	975	206

Total disk space is 976 cylinders

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

Displaying Partition Information

Remember, the FDISK command can't create or delete non-MS-DOS partitions. Thus, FDISK doesn't ask you for the partition number, only if you want to delete the MS-DOS partition.

The default response is "N". Therefore, if you make a mistake and don't want to delete the MS-DOS partition, simply press **Enter**. Otherwise, type "Y" and press **Enter** to delete the MS-DOS partition. Once the MS-DOS partition has been deleted, you cannot access the hard disc or the data on it with the MS-DOS operating system. Also, your computer will no longer be able to access that portion of the hard disc that was in the deleted partition *unless* it is included in some other partition.

Select option 4 from the FDISK Options Menu to display the current partition information for a hard disc. A Display Partition Information screen similar to the one shown below is displayed.

Display Partition Information

Partition	Status	Type	Start	End	Size
1	A	DOS	0	769	770
2	N	non-DOS	770	975	206

Total disk space is 976 cylinders

The partition number, status (active or non-active), type (MS-DOS or non-MS-DOS), starting cylinder, ending cylinder, and size (in cylinders) are listed for each partition. In addition, the total number of cylinders on the hard disc is displayed.

Selecting the Next Fixed Disc Drive

Select option 5 from the FDISK Options Menu to switch all FDISK operations from drive C: to the next internal hard disc drive. Then, select options 1 through 4 depending on the type of FDISK operation you want to perform on the disc in the next internal hard disc drive.

FIND

The **FIND** command searches one or more files for a specified text string.

Type

FIND is an **EXTERNAL** command.

Syntax

```
[<d>:][<path>]FIND [/V][/C][/N]"<text string>"  
                [[<d1>:][<path1>]  
                <filename1>...>]
```

d: the drive that contains the **FIND** command.

path the path to the **FIND** command.

/V the option that displays each line **NOT** containing the specified text string.

/C the option that counts the number of lines containing the specified text string, instead of displaying the lines.

/N the option that displays each line containing the specified text string, preceded by its line number in the file.

text string the string to be located. It must be enclosed by quotation marks.

d1: the drive that contains the first file to be searched.

path1 the path to the first file to be searched.
filename1 the name of the first file to be searched.

Operation

FIND is one of three MS-DOS filters. For a detailed explanation of filters, refer to the chapter entitled "Redirecting Input and Output".

FIND looks through one or more specified files and attempts to locate the specified <text string>. The text string must be exact as to case (upper- and lower-case letters) and spacing. In addition, it must be enclosed by quotation marks. Quotation marks should be used even if the string to be located is already enclosed in quotation marks. For example, to locate the following string in FILE1.TXT:

`"just in case"`

enter the command line:

`FIND ""just in case"" FILE1.TXT`

In another example, to locate the following string in FILE1.TXT:

`He said, "Hello" to me.`

enter the command line:

`FIND "He said, ""Hello"" to me."`

Notice that "Hello" is surrounded by double quotes.

To locate a string in two or more files, enter the specific filenames on the command line. For example, to locate the string "My name is John" in the files FILE1.TXT and FILE2.TXT, enter the command line:

```
FIND "My name is John" FILE1.TXT FILE2.TXT
```

FIND supports three options which allow additional flexibility in searches: /V, /C and /N. The /C option takes precedence over the /N option. If both /C and /N are specified, FIND ignores the /N option. If both the /C and /V options are specified, FIND counts the number of lines *not* containing the string.

If no filename is specified, input from the standard input device (the console) is used as the source, and its output is used as input to the FIND filter. For example, the command line:

```
C>DIR | FIND "<DIR>"
```

displays all of the subdirectories within the current directory. In this example, the DIR command is input from the standard input device, and its output is piped to the FIND filter. As a result, the directory entries containing the string <DIR> are displayed.

Notes

1. Wildcards *cannot* be used with the FIND command to reference a group of files. Instead, the files must be referenced individually.
2. The specified file (or input from the standard input device) is searched up to the first end-of-file marker ((**CTRL**) (**Z**)).
3. The options must precede the text string and filename on the command line.

FOR150

The FOR150 command prepares a 720 Kb 3.5" disc for use with an HP150 computer.

Caution



The FOR150 command destroys all the data on a disc. Therefore, make sure the disc doesn't contain valuable data before you format it.

Syntax

[<d>:][<path>]FOR150 <d1>:[/V]

- | | |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| d: | the drive that contains the FOR150 command. |
| path | the path to the FOR150 command. |
| d1: | the drive that contains the 720 Kb 3.5" disc to be formatted. |
| /V | the option that writes a volume label on the disc. If this option is specified, FOR150 prompts for a volume label after the disc is formatted. A volume label can be up to 11 characters in length. It's displayed when the DIR command is used. |

Operation

The FOR150 command allows you to prepare a 720 Kb 3.5" disc for use by MS-DOS on an HP150 computer. A disc must be formatted before it can be used. To format a 720 Kb 3.5" disc, enter the FOR150 command followed by the drive designator. For example, the following command line:

FOR150 A:

formats the 720 Kb 3.5" disc in drive A:.

FOR150 performs several tasks during the format process:

- It places tracks and sectors on the disc
- It marks defective tracks to prevent MS-DOS from placing data in them
- It builds the root directory and the FATs (File Allocation Tables)
- It prints a status report

Notes

1. The FOR150 command only works on double-sided 720 Kb 3.5" discs. It requires MS-DOS 3.2 and the 3.5" installable device driver.
2. The FOR150 command can't put the "system" on a disc. Therefore, a disc formatted with this command can't be used to start your system.

FORMAT

The FORMAT command prepares a disc for use.

To format a 3.5-inch 720 Kb disc for use with an HP150 computer, use the FOR150 command instead of the FORMAT command.

Caution

The FORMAT command destroys all of the data on a disc. Therefore, make sure the disc doesn't contain valuable data before you format it.

Type

FORMAT is an EXTERNAL command.

Syntax

```
[<d>:][<path>]FORMAT <d1>:[/S][/V][/B][/1]  
[ /4][/B][/T:<tt>][ /N:<nn>]
```

- | | |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| d: | the drive that contains the FORMAT command. |
| path | the path to the FORMAT command. |
| d1: | the drive that contains the disc to be formatted. |
| /S | the option that puts the "system" on the disc. The system consists of the two MS-DOS hidden system files, plus the MS-DOS command processor (COMMAND.COM). |

- /V** the option that writes a volume label on the disc. If this option is specified, **FORMAT** prompts for a volume label after the disc is formatted. A volume label can be up to 11 characters in length. It's displayed when the **DIR** command is used.
- /8** the option that formats a disc with only 8 sectors per track. This option can only be used to format single-sided flexible discs (160/180 Kb) and double-sided flexible discs (320/360 Kb). It *cannot* be used to format hard discs, high capacity flexible discs (1.2 Mb), or 3.5-inch flexible discs (720 Kb and 1.44 Mb).
- /1** the option that formats a disc single-sided, regardless of disc drive type. This option can only be used to format single-sided flexible discs (160/180 Kb) and double-sided flexible discs (320/360 Kb). It *cannot* be used to format hard discs, high capacity flexible discs (1.2 Mb), or 3.5-inch flexible discs (720 Kb or 1.44 Mb).
- /4** the option that formats single-sided and double-sided flexible discs (160/180 Kb and 320/360 Kb) in a high capacity (1.2 Mb) disc drive. This option *cannot* be used to format hard discs, high capacity flexible discs (1.2 Mb), or 3.5-inch flexible discs (720 Kb or 1.44 Mb).

/B	the option that formats a disc for backward compatibility. This option formats a disc with 8 sectors per track and reserves space for the MS-DOS hidden system files. The hidden system files aren't actually transferred to the disc; that is accomplished with the SYS command. This option allows a disc to be formatted, and an earlier version of MS-DOS to be installed on it. This option <i>cannot</i> be used to format hard discs, high capacity flexible discs (1.2 Mb), or 3.5-inch flexible discs (720 Kb or 1.44 Mb).
/T	the option that specifies the number of tracks per side. This option must be used in conjunction with /N. To format a 720 Kb disc, specify /T with a value of 80 (/T:80).
/N	the option that specifies the number of sectors per track. This option must be used in conjunction with /T. To format a 720 Kb disc, specify /N with a value of 9 (/N:9).

Operation

The FORMAT command allows you to prepare a disc for use by MS-DOS. A disc must be formatted before it can be used. To format a disc, enter the FORMAT command followed by the drive designator.

Formatting a Flexible Disc

To format the flexible disc in drive A:, enter the following command:

FORMAT A:

Note



In this example, the disc type must match the drive type (you must be formatting a 360 Kb disc in a 360 Kb drive, or a 720 Kb disc in a 720 Kb drive, etc.).

To format a 360 Kb flexible disc in a 1.2 Mb flexible disc drive, enter the following command line:

```
FORMAT A: /4
```

To format a 720 Kb flexible disc in a 1.44 Mb flexible disc drive, enter the following command line:

```
FORMAT A: /T:80 /N:9
```

Formatting a Hard Disc

To format the hard disc in drive C:, enter the following command line:

```
FORMAT C:
```

If you're formatting a hard disc that's been formatted before, FORMAT prompts you with the following precautionary message before it formats the disc:

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

If your hard disc doesn't have a volume label, press . Otherwise, type the volume label and press . If you type the wrong volume label, FORMAT displays the following error message:

```
Invalid Volume ID Format Failure
```

and you must enter the **FORMAT** command again.
Otherwise, **FORMAT** displays this message:

```
WARNING, ALL DATA ON NON-REMOVABLE DISK  
DRIVE C: WILL BE LOST! Proceed with Format  
(Y/N)?
```

To format the hard disc, type **"Y"** and press **Enter**. To cancel the **FORMAT** command and return to the MS-DOS prompt, type **"N"** and press **Enter**.

FORMAT performs several tasks during the format process:

- It places tracks and sectors on the disc
- It marks defective tracks to prevent MS-DOS from placing data in them
- It builds the root directory and FATs (File Allocation Tables)
- It prints a status report indicating:
 - Total disc space
 - Defective tracks (if any)
 - Space allocated to MS-DOS hidden system files (if any)
 - Available disc space

FORMAT supports the following options: **/S**, **/V**, **/8**, **/1**, **/4**, **/B**, **/T** and **/N**. These options can be entered in any order on the command line. However, if two or more options conflict, the last option specified on the command line is used.

In addition, not all of the options are compatible with each type of disc. The following table shows which options are compatible with each disc type:

Disc Type	Valid Options
160/180 Kb	/S, /V, /8, /1, /4, /B
320/360 Kb	/S, /V, /8, /1, /4, /B
720 Kb	/S, /V, /T, /N
1.2 Mb	/S, /V
1.44 Mb	/S, /V
Hard Disc	/S, /V

The **FORMAT** command returns several error codes. The error codes and their descriptions are listed below.

0	Successful completion
3	Terminated by user CTRL Break
4	Terminated due to error
5	Terminated due to "N" response on a hard disc

These error codes can be used with the **ERRORLEVEL** option of the batch processing **IF** command. Refer to the chapter entitled "Batch Processing" for additional information.

Notes

1. A hard disc must be reformatted if the size of the MS-DOS partition is changed with the FDISK command.
2. A flexible disc is formatted according to disc drive type, *not* disc type, unless the /1, /4, /8, or the /T and /N options are specified. In addition, not all flexible discs are compatible with each type of disc drive. Refer to the appendix entitled "Disc and Disc Drive Compatibility" for additional information.
3. If the /S option is specified, the hidden system files and the MS-DOS command processor (COMMAND.COM) are transferred from the root directory of the disc in the active drive. If the active drive contains a hard disc and these files aren't present on the hard disc, FORMAT prompts you to insert the MS-DOS disc in drive A:.
4. DO NOT use the FORMAT command on a drive that has been ASSIGNED, JOINed, or SUBSTITuted. Also, don't use it on a network drive.
5. If you're formatting an application work disc and you want to use it to start your system, specify the /S option.



(This page has been left blank intentionally.)

GRAFTABL

The GRAFTABL command loads additional ASCII characters into memory for use by a color/graphics adapter card in graphics mode.

Type

GRAFTABL is an EXTERNAL command.

Syntax

[<d>:][<path>]GRAFTABL

d: the drive that contains the GRAFTABL command.

path the path to the GRAFTABL command.

Operation

The GRAFTABL command loads additional ASCII characters into memory. Specifically, it loads character codes 128 through 255. This command is useful for displaying extended characters (non-U.S., math, and line drawing) while in graphics mode. For a list of all the character codes and the characters they represent, see the appendix entitled "Character Codes and Keystrokes".

When the following command line is entered:

GRAFTABL

MS-DOS loads the resident portion of GRAFTABL into memory and displays the following message:

Graphics character loaded

GRAFTABL should only be loaded once. If you attempt to load it more than once, the following error message is displayed:

Graphics characters already loaded

Notes

1. The resident size of MS-DOS in memory increases when you use this command.

GRAPHICS

The GRAPHICS command allows you to print the contents of a graphics screen when you're using a color/graphics adapter card.

Note



Don't use the GRAPHICS command with HP printers in HP mode. Instead, install the Print Screen Utility described in the *Using the Multiple Character Set Utilities* manual.

Type

GRAPHICS is an EXTERNAL command.

Syntax

```
[<d>:][<path>]GRAPHICS [<printer>][/R][/B]
```

d:	the drive that contains the GRAPHICS command.
path	the path to the GRAPHICS command.
printer	the type of printer. Valid values are COLOR1, COLOR4, COLOR8, COMPACT, and GRAPHICS. The default is GRAPHICS.
/R	the option that prints the colors exactly as they appear on the display screen. If you don't specify this option, black is printed as white and white is printed as black.

/B the option that instructs GRAPHICS to print the background color. If you don't specify this option, the background color *isn't* printed. This option is only valid with two types of printers: COLOR4 and COLOR8.

Operation

The GRAPHICS command allows you to send the contents of a graphics screen to a printer. After you issue the GRAPHICS command, press **Shift** **Print Screen** (or **Print Screen**) depending on your computer) to print the contents of the screen.

The GRAPHICS command can be used with the following 5 types of printers:

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

To select one of the above printers, specify the printer type on the command line. If you don't specify a printer type, GRAPHICS uses the IBM Personal Graphics printer (GRAPHICS) as the default.

Notes

1. The resident size of MS-DOS in memory increases when you use this command.
2. If the printer type is COLOR1 or GRAPHICS, and the screen images are displayed in 320x200 color graphics mode, the screen contents are printed in up to four shades of gray. If the screen images are displayed in 640x200 mode, the screen contents are printed sideways on the printer. All other graphics modes are printed normally.

JOIN

The JOIN command allows a disc drive to be logically connected to another disc drive via a subdirectory.

Type

JOIN is an EXTERNAL command.

Syntax

[<d>:][<path>]JOIN <d1>: <d2>:<dirname>

or

[<d>:][<path>]JOIN <d1>: /D

or

[<d>:][<path>]JOIN

d: the drive that contains the JOIN command.

path the path to the JOIN command.

d1: the source drive. This is the drive to be joined to a subdirectory on the target drive. You can't specify the active drive as the source drive.

d2: the target drive. The source drive is to be joined to this drive.

dirname the subdirectory on the target drive. The source drive is to be joined to the target drive via this subdirectory. It must be a subdirectory of the root directory.

/D the option that "unjoins" (separates) two drives that have been joined. When this option is used, the combined drive becomes two physically addressable drives again.

Operation

The JOIN command allows you to combine two physical disc drives into one logical disc drive. This is done by making the source drive a subdirectory of the target drive.

The JOIN command has three forms. The first form joins two disc drives. For example, the command line:

```
C>JOIN A: C:\DRIVEA
```

connects drive A: to drive C: via a subdirectory on drive C: called DRIVEA. As a result, all the files on the disc in the drive A: now "logically" reside on the disc in drive C:, in the subdirectory DRIVEA. Any attempt to reference drive A: at this point, results in the following error message:

Invalid drive specification

The subdirectory used to join the source drive to the target drive *must* be a subdirectory of the root directory. It cannot be a subdirectory of another subdirectory. For example, the command line:

```
JOIN A: C:\DRIVEA\SUB1
```

causes the following error message to be displayed:

Invalid parameter

A new or existing subdirectory can be specified on the command line. If a new subdirectory is specified, the JOIN command automatically creates the subdirectory before it joins the two drives. If an existing subdirectory is specified, it *must* be empty. If it isn't empty, the following error message is displayed:

Directory not empty

The second form of the JOIN command unjoins the source drive from the target drive. For example, the following command line unjoins drive A: from drive C:

C>JOIN A: /D

Finally, the third form of the JOIN command displays all the drives currently joined. For example, if drive A: is joined to drive C:, the command line:

C>JOIN

causes the following information to be displayed:

A: => C:\DRIVEA

Notes

1. A JOIN command is in effect until another JOIN command is issued with the /D option, or until the system is reset.
2. When you unjoin two drives, the subdirectory used to join them doesn't automatically get deleted. As a result, we recommend that you use the same subdirectory each time you join two drives. This prevents the root directory from becoming cluttered with empty subdirectories.

3. DO NOT use the JOIN command on a drive that has been ASSIGNED or SUBSTITuted. Also, don't use it on a network drive.
4. DO NOT use the BACKUP, FORMAT, DISKCOMP, DISKCOPY, or RESTORE command on a drive that has been JOINed.

KEYBxx

The KEYBxx command replaces the ROM BIOS keyboard routine to provide support for non-U.S. keyboards. The "xx" in the command represents one of five keyboard programs provided on the MS-DOS disc.

Type

KEYBxx is an EXTERNAL command.

Syntax

[<d>:][<path>]KEYBUK

[<d>:][<path>]KEYBGR

[<d>:][<path>]KEYBFR

[<d>:][<path>]KEYBIT

[<d>:][<path>]KEYBSP

d: the drive that contains the KEYBxx command.

path the path to the KEYBxx command.

Operation

The KEYBxx command loads support for the U.S. keyboard and one of five non-U.S. keyboards. There are five different commands; each command is associated with a specific country. The commands and the countries associated with them are listed in the following table.

KEYBxx Command	Country
KEYBUK	United Kingdom
KEYBGR	Germany
KEYBFR	France
KEYBIT	Italy
KEYBSP	Spain

The **KEYBxx** command that is executed determines which country's keyboard layout controls the keyboard. In other words, it establishes the primary keyboard layout for your system. For example, the following command line loads support for the Spanish keyboard.

KEYBSP

Any keys typed on the keyboard are automatically interpreted as Spanish keys.

Once you've executed the appropriate **KEYBxx** command, MS-DOS allows you to switch between the primary keyboard layout and the U.S. keyboard layout. To change from the primary keyboard layout to the U.S. keyboard layout, press **CTRL** **Alt** **F1**. To return to the primary keyboard layout, press **CTRL** **Alt** **F2**.

Not all non-U.S. characters appear as keys on the keyboard; some accented characters must be "built". To build them, you need to press and release the appropriate accent key followed by the appropriate letter key. These accent keys are called "dead keys" because they don't produce characters unless they're used in combination with

a letter key. The allowable dead key combinations are listed below.

Germany	á é É í ó ú à è ì ò ù
France	ä Ä ë ì ö Ö ü Ü ÿ â é ê ô û
Spain	ä Ä ë ì ö Ö ü Ü ÿ á é É í ó ú à è ì ò ù â ê î ô û
UK	dead key not supported
Italy	dead key not supported

If you press an invalid dead key combination (a combination that isn't listed above), you'll see the accent followed by the letter and you'll hear a beep.

Notes

1. The **KEYBxx** code is loaded above MS-DOS and remains resident in memory until the system is turned off or restarted.
2. Some non-U.S. keyboards have keys with 3 characters on them. To access the third character on a key, hold down **CTRL** and **Alt**, and then press the key.
3. To automatically load support for a non-U.S. keyboard every time you start your system, include the appropriate **KEYBxx** command in your **AUTOEXEC.BAT** file. For additional information on the **AUTOEXEC.BAT** file, see the chapter entitled "Batch Processing".

LABEL

The LABEL command adds, changes, or deletes the volume label on a disc.

Type

LABEL is an EXTERNAL command.

Syntax

[<d>:][<path>]LABEL [<d1>:][<volume label>]

d: the drive that contains the LABEL command.

path the path to the LABEL command.

d1: the drive that contains the disc you want to label. If a drive is not specified, the active drive is assumed.

volume label the volume label used to identify the disc. It can be up to 11 characters in length; additional characters are truncated. The following characters cannot be used in a volume label:

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

Also, spaces and tabs can't be included.

Operation

The LABEL command allows you to add, change, or delete the volume label on a disc. In many cases, the volume label is placed on the disc at the time it is formatted (see the /V option of the FORMAT command for additional information). If a disc isn't labeled at the time it is formatted, the LABEL command can be used to add one. It can also be used to change or delete an existing volume label on a disc.

To add or change the volume label on a disc, enter the LABEL command followed by the drive designator and the new volume label. For example, the command line:

```
LABEL C:HARDDISC
```

puts the volume label HARDDISC on the disc in drive C:. If you enter the LABEL command and drive designator, but not the new volume label, LABEL prompts you for it with the following message:

```
Volume label (11 characters, ENTER for  
none)?
```

Enter a new volume label and press .

To delete the existing volume label on a disc, enter the LABEL command followed by the drive designator. Then, when LABEL prompts you for a new volume label, simply press . As a precaution, the following message is displayed:

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

Enter "N" to cancel the command and keep the current volume label intact. Enter "Y" to delete the current volume label.

To display the existing volume label on a disc, enter the **DIR** command followed by the drive designator. For example, the command line:

DIR C:

displays the volume label on disc C: followed by a directory listing of the files on C:.

Notes

1. **DO NOT** use the **LABEL** command on a drive that has been **ASSIGNed**, **JOINed**, or **SUBSTituted**. Also, don't use it on a network drive.

MKDIR (Make Directory)

The MKDIR command creates a new subdirectory on a disc.

Type

MKDIR is an INTERNAL command.

Syntax

MKDIR [**<d>:**][**<path>**]**<dirname>**

d: the drive to contain the new subdirectory. If a drive isn't specified, the active drive is assumed.

path the path to the new subdirectory.

dirname the name of the new subdirectory.

Operation

The MKDIR command allows you to build a hierarchical directory structure on your disc by creating new subdirectories. For additional information on the MS-DOS hierarchical directory structure, see the chapter entitled "Understanding MS-DOS Concepts".

You create subdirectories one level at a time, beginning with the second level (the root directory is the first level). In addition, there are two ways to create subdirectories. You can specify the path from the root directory to the new subdirectory when you enter the MKDIR command. Or, you can use the CHDIR (Change Directory) command to move to a directory and enter the MKDIR command from there. Examples of both methods are listed below.

Specifying the Path from the Root Directory. To create a new subdirectory with this method, enter the **MKDIR** command followed by the path from the root directory and the new subdirectory name. For example, to create a new subdirectory called **USER#1** within the root directory of drive C:, enter the following command line:

```
C>MKDIR \USER#1
```

Then, to create a new subdirectory called **REPORTS** within the subdirectory **USERS**, enter the following command line:

```
C>MKDIR \USER#1\REPORTS
```

Using the CHDIR Command. To create a new subdirectory with this method, use the **CHDIR** command to move to the directory you want to create a subdirectory within, and then enter the **MKDIR** command followed by the new subdirectory name from there. You do not need to enter the path from the root directory. For example, to create a new subdirectory called **USER#1** within the root directory of drive C:, enter the following two command lines:

```
C>CHDIR \
```

```
C>MKDIR USER#1
```

Then, to create a new subdirectory called **REPORTS** within the subdirectory **USER#1**, enter the following two command lines:

```
C>CHDIR USER#1
```

```
C>MKDIR REPORTS
```

Notes

1. The root directory has a predetermined number of directory entries (files and subdirectories) that it can have, based on the type and capacity of your disc drive. Refer to the chapter entitled "Understanding MS-DOS Concepts" for additional information.
2. Unlike the root directory, there is no limit to the number of directory entries that a subdirectory can have.
3. Subdirectories can be nested any level deep. However, the path from the root directory to a subdirectory, plus the subdirectory name, is limited to a maximum of 64 characters.
4. MD is an alternate form of the MKDIR command. For example, the command line MD \USER#1\REPORTS is equivalent to the command line MKDIR \USER#1\REPORTS.

MODE

The MODE command is used to configure video, parallel, and serial (RS-232/422) adapter cards for use with an HP computer.

Type

MODE is an EXTERNAL command.

Syntax

Video Display Mode:

```
[<d>:][<path>]MODE <display>
```

or

```
[<d>:][<path>]MODE [<display>],<adjust>[,T]
```

d:	the drive that contains the MODE command.
path	the path to the MODE command.
display	the display mode. Valid display modes are 40, 80, BW40, BW80, CO40, CO80, and MONO.
adjust	the direction that the display is to be shifted. Valid directions are R (right) and L (left).
T	the option that produces a test pattern. The test pattern can be used to align the display.

Serial Communication Mode:

```
[<d>:][<path>]MODE COMn:<baud>  
          [,<parity>[,<databits>  
          [,<stopbits>[,P]]]]
```

d:	the drive that contains the MODE command.
path	the path to the MODE command.
COMn	the asynchronous communications port (COM1, COM2, COM3, or COM4).
baud	the baud (transmission) rate. Valid rates are 110, 150, 300, 600, 1200, 2400, 4800, 9600, and 19,200. Only the first two digits need to be specified; the remaining digits are ignored. The default baud rate is 2400.
parity	the parity. Parity is N (none), O (odd) or E (even). The default parity is "E".
databits	the number of databits in each character transmitted (either 7 or 8). The default number of databits is 7.
stopbits	the number of stopbits (either 1 or 2). The default number of stopbits is 2 if the baud rate is 110, and 1 for all other baud rates.
P	the option that causes MODE to continuously retry sending output to a printer if time-out errors occur.

Parallel Printer Mode:

```
[<d>:][<path>]MODE LPTn:[<horz>]  
[,[<vert>][,P]]
```

d:	the drive that contains the MODE command.
path	the path to the MODE command.
LPTn	the parallel printer port (LPT1, LPT2, or LPT3).
horz	the horizontal spacing (either 80 or 132 characters per line).
vert	the vertical spacing (either 6 or 8 lines per inch).
P	the option that causes MODE to continuously retry sending output to a printer if time-out errors occur.

Redirecting Parallel Printer Output:

```
[<d>:][<path>]MODE LPTn:[=COMn:]
```

d:	the drive that contains the MODE command.
path	the path to the MODE command.
LPTn	the parallel printer port (LPT1, LPT2, or LPT3).
COMn	the asynchronous communications port (COM1, COM2, COM3, or COM4).

Operation

The **MODE** command sets the operating parameters for the system video display, and the serial and parallel ports. **MODE** has a different syntax for each device.

Video Display Mode. **MODE** performs the following two operations on a video display adapter:

- It selects the display mode
- It aligns the display on the screen

To select the display mode, enter the command line:

MODE <display>

where <display> is one of the valid display modes. Each display mode is described on the next page.

40	This sets the number of characters per line to 40 for a CGA-compatible video adapter.
80	This sets the number of characters per line to 80 for a CGA-compatible video adapter.
BW40	This uses the CGA-compatible video adapter as the active display adapter, disables color or grey scale, and sets the number of characters per line to 40.
BW80	This uses the CGA-compatible video adapter as the active display adapter, disables color or grey scale, and sets the number of characters per line to 80.

CO40	This uses the CGA-compatible video adapter as the active display adapter, enables color or grey scale, and sets the number of characters per line to 40.
CO80	This uses the CGA-compatible video adapter as the active display adapter, enables color or grey scale, and sets the number of characters per line to 80.
MONO	If two adapters are attached, this selects the monochrome display adapter as the active display adapter. Monochrome display adapters only support 80 characters per line.

In addition to selecting the display mode, the **MODE** command also provides a means of aligning the display on the screen and generating a test pattern. The display can be adjusted to the left or right on the screen (one character at a time in 40 column mode and two in 80 column mode). The following command line sets the display to CO80 mode, shifts the display 2 characters to the right, and displays a test pattern.

```
MODE CO80,R,T
```

Because the **T** option is specified, after each shift right, **MODE** asks if you want to continue adjusting the display. Enter **"Y"** to continue shifting the display in the same direction or **"N"** to stop. To shift the display to the left, specify the **L** option instead of the **R** option.

Serial Communication Mode. The **MODE** command is used to change the parameters and protocol for the serial (sometimes called RS-232) port(s) on your adapter cards. There are four serial ports: COM1, COM2, COM3, and COM4.

The following command line sets the serial port, COM1, to 1200 baud, even parity, 8 databits, and 1 stopbit.

```
MODE COM1:12,E,8,1
```

MODE supports one additional option for the serial port: the P option. This option indicates that the serial port is being used as the system printer port, and therefore all time-outs are continuously retried. The following command line sets the same parameters as the previous example, plus specifies the P option.

```
MODE COM1:12,E,8,1,P
```

Parallel Printer Mode. MS-DOS supports three parallel printer devices: LPT1, LPT2, and LPT3. The MODE command can be used to set certain parameters for each of these devices. The number of characters per line (<horz>) and the number of lines per inch (<vert>) can be specified.

The following command line sets the horizontal spacing of LPT2 to 132 characters per line and the vertical spacing of LPT2 to 6 lines per inch.

```
MODE LPT2:132,6
```

The valid values for these two parameters are limited: the number of characters per line can be either 80 or 132 and the number of lines per inch can be either 6 or 8. If any other values are entered, MODE ignores them and preserves the current values.

The P option instructs MS-DOS to continuously retry all time-outs. The following command line sets LPT2 to the spacing values in the previous example and enables the P option.

```
MODE LPT2:132,6,P
```

Redirecting Parallel Printer Output. The final MODE syntax variation allows data originally intended for the parallel printer port to be redirected (sent) to one of the serial ports (COM1, COM2, COM3, or COM4).

Note



Before you can redirect parallel printer output to a serial port, you must set the serial communication mode (described on the previous page) for that port. Also, if the serial device is a printer, we recommend you specify the **P** option.

The following command line redirects data from LPT1 to COM1.

```
MODE LPT1:=COM1:
```

To cancel this redirection, enter:

```
MODE LPT1
```

Notes

1. A continuous retry loop can be broken by pressing **CTRL Break**. To disable the **P** option for a device, re-enter the MODE command and original parameters without the **P** option. For example,

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

enables the **P** option for LPT1 and

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

disables it.

2. When you (1) adjust the display right or left, (2) specify the P option, or (3) redirect parallel printer output (LPTn=COMn), MODE loads resident code into memory. This increases the size of MS-DOS. The resident code remains in memory until you turn off or restart your system.

MORE

The MORE command displays output on your screen, one screen (24 lines) at a time.

Type

MORE is an EXTERNAL command.

Syntax

[<d>:][<path>] MORE

d: the drive that contains the MORE command.

path the path to the MORE command.

Operation

MORE is one of three filters provided with the MS-DOS operating system. For a detailed explanation of filters, refer to the chapter entitled "Redirecting Input and Output".

MORE displays the first screenful of data (24 lines), displays the message --MORE--, and pauses. It waits until any key is pressed, and then displays the next screenful of data. This process continues until all the data has been read.

The **MORE** filter is useful for viewing the contents of a lengthy file or directory, one screen at a time. For example, the command line:

```
TYPE FILE1.TXT | MORE
```

displays the contents of **FILE1.TXT**, one screen at a time. Note the use of the "|" character to separate the MS-DOS command from **MORE**.

Another way to display the contents of **FILE1.TXT** a screen at a time is to redirect it to the **MORE** filter as shown below.

```
MORE<FILE1.TXT
```

For a detailed explanation on redirection, refer to the chapter entitled "Redirecting Input and Output".

PATH

The PATH command sets search path(s) to one or more directories for external MS-DOS commands, program files, and batch files. It also displays and deletes existing search paths.

Type

PATH is an INTERNAL command.

Syntax

PATH [[<d1>:]<path1>[;[<d2>:]<path2>...]]

d1: the drive that contains the first search path.

path1 the first search path.

d2: the drive that contains the second search path.

path2 the second search path.

Operation

The PATH command allows you to specify a list of directories to be searched every time you execute an external MS-DOS command, a program file, or a batch file. Then, MS-DOS automatically searches this list when it is unable to find an external command, a program file, or a batch file in the current directory. Each directory in the list is referred to as a "search path". One or more search paths can be set with the PATH command. The PATH command can also be used to display or cancel existing search paths.

MS-DOS only searches one directory at a time. It always starts by searching the current directory. Then, it searches the directories specified by the PATH command (in the order they appear on the command line). Once MS-DOS finds the command or file it's looking for, it stops the search. It doesn't search the remaining directories.

The PATH command should be used at the beginning of a work session (i.e., just after you turn your system on). This sets the search paths for your entire work session. MS-DOS uses these search paths until you end your work session (turn your system off or restart it) or issue another PATH command.

To set the root directory and one subdirectory (USER#1) on drive C: as search paths, enter the following command line:

```
PATH C:\;C:\USER#1
```

To set the same search paths as the previous example, plus add the subdirectory REPORTS (which is a subdirectory of USER#1), enter the command line:

```
PATH C:\;C:\USER#1;C:\USER#1\REPORTS
```

Note that the search path to each subdirectory must be individually specified.

To display existing search paths, enter the command line:

```
PATH
```

To cancel existing search paths, enter the command line:

```
PATH ;
```

Notes

1. MS-DOS only searches for program files and batch files when you execute them, not when you perform any other tasks on them (such as editing).
2. If you want to establish a permanent search path, include the PATH command along with the search path in your AUTOEXEC.BAT file. For additional information on the AUTOEXEC.BAT file, refer to the chapter entitled "Batch Processing".
3. MS-DOS can't detect an invalid search path until it attempts to use it. If MS-DOS detects an invalid search path (due to incorrect syntax or a non-existent subdirectory), MS-DOS ignores it and continues on with the next search path.
4. If you execute commands that load resident code (such as GRAPHICS, KEYBxx, MODE, or PRINT) before you execute the PATH command, there may not be enough room in the MS-DOS environment to store the list of directories you specified. If there isn't enough room, the following message is displayed:

Out of environment space

To correct this situation, execute the PATH command first.

PRINT

The **PRINT** command prints the contents of a file (or a list of files) on your printer while you're performing other tasks on your system. This process is frequently referred to as "background printing".

Type

PRINT is an **EXTERNAL** command.

Syntax

```
[<d>:][<path>]PRINT [/D:<device>][ /B:<buffsize>]
                               [/U:<busyticks>][ /M:<maxticks>]
                               [/S:<timeslice>]
                               [/Q:<queuesize>][ /C][ /T][ /P]
                               [[<d1>:][<path1>][<filename1>]...]
```

d: the drive that contains the **PRINT** command.

path the path to the **PRINT** command.

/D:<device> the option that specifies the list (output) device. A valid list device is any character device driver supported by the system. For example, **AUX**, **COM1**, **COM2**, **COM3**, **COM4**, **LPT1**, **LPT2**, **LPT3**, or **PRN**. The default list device is **PRN**. This option must be the first option specified on the command line.

<code>/B:<buffsize></code>	the option that sets the size of the internal buffer in bytes. The default internal buffer size is 512 bytes. Increasing the size of the internal buffer may improve the performance of the PRINT command.
<code>/U:<busyticks></code>	the option that specifies the number of computer clock ticks that PRINT will wait until the printer is available. If PRINT has to wait a longer period of time, it gives up its time slice. The default value is 1.
<code>/M:<maxticks></code>	the option that specifies the number of computer clock ticks that PRINT can have to send characters to the printer. The number of computer clock ticks can range from 1 to 255. The default value is 2.
<code>/S:<timeslice></code>	the option that specifies the time slice (the amount of time). The time slice can range from 1 to 255. The default value is 8.
<code>/Q:<queuesize></code>	the option that specifies the number of files allowed in the print queue. The number of files can range from 4 to 32. The default value is 10.

/C



the option that cancels specific file(s) in the print queue. If the file is in the process of being printed, the printing stops and a cancellation message is automatically printed. The preceding file and all of the following files that were entered on the command line are removed from the print queue until a /P is found on the command line, or until you press .

/T

the option that cancels or "terminates" all files in the print queue. If a file is in the process of being printed, the printing stops and a cancellation message is automatically printed.

/P

the option that turns on the print mode. The preceding file and all of the following files are added to the print queue until a /C is found on the command line, or until you press .

d1

the drive that contains the file(s) to be printed.

path1

the path to the file(s) to be printed.

filename1

the name(s) of the file(s) to be printed.

Operation

The PRINT command creates and maintains a print queue. A print queue is a list of files to be printed as a background task (while you use your system to perform some other task). The PRINT command provides a wide range of options which allow you to control the print queue system parameters and the print queue, itself.

Files can be added to the print queue by entering the names of the files on the command line. For example, to queue the files FILE1.TXT and FILE2.TXT for printing, enter the command line:

```
PRINT FILE1.TXT FILE2.TXT
```

The files are added to the print queue in the order they were entered on the command line. After each file is printed, the printer paper is advanced to the next page.

Caution



The print queue only holds the names of the files to be printed, and not the actual contents of those files. Therefore, the disc(s) containing the files to be printed *must* remain in its drive(s) until the print queue is empty. Also, the files to be printed *must not* be changed or erased while their names are in the print queue.

The PRINT command provides six options for controlling the print queue's system parameters. They are /D, /B, /U, /M, /S and /Q. If you want to use these options, you *must* specify them the first time you execute the PRINT command after you start your system. Otherwise, the default parameters are automatically set. Also, once the print queue's system parameters are set, they remain in effect until you turn off or restart your system.

If the /D option isn't specified the first time the PRINT command is executed, PRINT prompts for the list (print)

device with the message:

Name of list device [PRN]:

To select PRN as the list device, simply press **Enter**. To select another character device as the list device, type the device name and press **Enter**.

The PRINT command also provides three options for controlling the print queue and the files in it. They are /C, /T and /P.

To remove the files FILE1.TXT and FILE2.TXT from the print queue and add the file FILE3.TXT to the print queue, enter the following command line:

```
PRINT FILE1.TXT/C FILE2.TXT/P FILE3.TXT
```

To remove all of the files currently in the print queue, enter the PRINT command with the /T option. For example, the command line:

```
PRINT /T
```

clears the print queue.

To determine which files are currently in the print queue, enter the PRINT command without parameters.

Notes

1. The printer cannot be used by any other MS-DOS command or application program while PRINT is queuing and printing files. If another MS-DOS command or application program attempts to use the printer, the following error message is displayed:

Out of paper

2. **PRINT** accepts wildcards in filenames. This allows you to queue a group of files for printing. For example, to add all the files with the filename extension of .TXT to the print queue, enter the command line:

```
PRINT *.TXT
```

3. If you want to permanently set the print queue's system parameters, include the **PRINT** command along with the appropriate options in your **AUTOEXEC.BAT** file. For additional information on the **AUTOEXEC.BAT** file, refer to the chapter entitled "Batch Processing".
4. Tab characters are expanded to the next 8-column boundary with blanks.
5. The path to each file in the print queue can contain a maximum of 64 characters, including the drive designator and the filename. Therefore, you may need to change directories if you want to print a file in a subdirectory that is many levels deep.
6. The **PRINT** command increases the resident size of MS-DOS in memory the first time it is executed.
7. If **PRINT** encounters a disc error while printing a file, the printing process stops, a message is printed on the printer, and then **PRINT** continues on with the next file in the queue.

PROMPT

The PROMPT command changes the MS-DOS prompt.

Type

PROMPT is an INTERNAL command.

Syntax

PROMPT [<text>]

text the text and special characters for the new MS-DOS prompt.

Operation

The PROMPT command allows you to change the MS-DOS prompt to any prompt of your choice. The default prompt is the active drive designator (the active drive letter followed by the greater than (>) character). The new MS-DOS prompt can consist of any combination of text and special characters.

The text characters can consist of any of the characters permissible in MS-DOS filenames. For example, to create the friendly MS-DOS prompt HELLO, enter the following command line:

```
PROMPT HELLO
```

The letters HELLO are displayed instead of the default MS-DOS prompt.

In addition to text characters, there are special characters that allow more than just text strings to be inserted in the MS-DOS prompt. These characters are all preceded by the dollar sign (\$) character to differentiate them from text strings. The table below lists the special characters and their corresponding MS-DOS prompts.

Character	MS-DOS Prompt
\$\$	The dollar sign (\$) character
\$B	The vertical bar () character
\$D	The current date
\$E	The ASCII escape code (1BH)
\$G	The greater than (>) character
\$H	The backspace character
\$L	The less than (<) character
\$N	The active drive
\$P	The current directory of the active drive
\$Q	The equal sign (=) character
\$T	The current time
\$V	The MS-DOS version number
\$_(underline)	A carriage return/line feed sequence (go to the next line)

You can use either upper-case or lower-case letters as special characters.

To set the MS-DOS prompt to the current time, enter the following command line:

```
PROMPT $T
```

MS-DOS responds with a prompt similar to this:

```
11:36:45:00
```

To set a two-line prompt that prints "TIME =" followed by the current time, "DATE =" followed by the current date, and the greater than (>) sign, enter the command line:

```
PROMPT TIME = $T$ _DATE = $D$G
```

Your new MS-DOS prompt looks similar to this:

```
TIME = 11:42:08.00  
DATE = Mon 1-5-1987>
```

To set the MS-DOS prompt to the current directory of the active drive followed by the greater than (>) sign, enter the command line:

```
PROMPT $P$G
```

If the active drive is C: and the current directory is the root directory, MS-DOS responds with the following prompt:

```
C:\>
```

If ANSI terminal support is installed, the ASCII escape code special character (`$E`) can be used to produce inverse video, cursor positioning, and other video attributes in the MS-DOS prompt. For example, to set the prompt in inverse video mode and return to video mode for other text, enter the following command line:

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

Notes

1. The **PROMPT** command is stored as a string in the MS-DOS environment. Refer to the **SET** command for additional information.
2. If you execute commands that load resident code (such as **GRAPHICS**, **KEYBxx**, **MODE**, or **PRINT**) before you execute the **PROMPT** command, there may not be enough room in the MS-DOS environment to store the prompt you specified. If there isn't enough room, the following message is displayed:

Out of environment space

To correct this situation, execute the **PROMPT** command first.

RECOVER

The **RECOVER** command recovers a single file or all of the files from a disc containing bad sectors. The data in the bad sector(s) cannot be recovered.

Caution



When you recover all of the files from a disc, the **RECOVER** command renames every file and subdirectory on that disc and puts them all into the root directory. Also, **RECOVER** puts additional characters at the end of all your text files, which means you'll have to edit them to remove these characters. Therefore, *only* recover all of the files from a disc if it's absolutely necessary (for example, if the disc has a bad root directory).

Type

RECOVER is an **EXTERNAL** command.

Syntax

```
[<d>:][<path>]RECOVER [<d1>:][<path1>]<filename1>
```

or

```
[<d>:][<path>]RECOVER <d1>:
```

d: the drive that contains the **RECOVER** command.

path the path to the **RECOVER** command.

d1:	the drive that contains the file(s) to be recovered.
path1	the path to the file(s) to be recovered.
filename1	the name(s) of the file(s) to be recovered.

Operation

If a sector on a disc is causing read/write errors, the **RECOVER** command can be used to recover either the file containing that sector (minus the bad sector), or the entire disc (if the bad sector is in the root directory).

To recover a file called **FILE1.TXT** located in the root directory of drive **B:**, enter the command line:

```
RECOVER B:FILE1.TXT
```

MS-DOS responds with the following prompt:

```
Press any key to begin recovery of the
file(s) on drive B:
```

After any key is pressed, MS-DOS reads the file sector-by-sector, skipping over the bad sector(s). The good sectors are placed in a file with the same name. The bad sectors are marked as defective so that MS-DOS won't allocate new data to them.

After all the good sectors have been recovered, the following message is displayed:

```
nnnnnn of mmmmmm bytes recovered
```

where nnnnnn is the number of bytes recovered, and
mmmmmm is the original size of the file.

To recover all the files located on the disc in drive B:,
enter the command line:

RECOVER B:

MS-DOS responds with the following prompt:

**Press any key to begin recovery of the
file(s) on drive B:**

After any key is pressed, MS-DOS searches the File
Allocation Tables (FATs) for chains of allocation units (or
clusters). Then, MS-DOS creates a file in the root
directory of drive B: for each chain found. MS-DOS uses
the following format to name the files:

FILEnnnn.REC

where nnnn is a sequential number starting with 0001.

After all the files are recovered, the following message is
displayed:

xxxx file(s) recovered

where xxxx is the number of files recovered.

For example, if you recover the files located on the disc in
drive B: and then enter the **DIR** (Directory) command,
you'll see a display similar to the one on the following
page.

RECOVER B:

Press any key to begin recovery of the
file(s) on drive B:

6 file(s) recovered

DIR B:

Volume in drive B has no label
Directory of B:\

FILE0001	REC	5120	10-18-87	9:55a
FILE0002	REC	2048	10-18-87	9:55a
FILE0003	REC	1024	10-18-87	9:55a
FILE0004	REC	2048	10-18-87	9:55a
FILE0005	REC	2048	10-18-87	9:55a
FILE0006	REC	2048	10-18-87	9:55a
		6 File(s)	249856	bytes free

After the files on a disc have been recovered, the **RENAME** command can be used to rename each file back to its original filename. The **RENAME** command is covered in this chapter.

Notes

1. If there isn't enough room in the root directory for the recovered files, **RECOVER** displays a message to that effect and stores information about the extra files in the File Allocation Table. If this happens, you must delete some files from your root directory and run **RECOVER** again. Before you delete files from your root directory, you may want to copy them to another disc.

RENAME

The RENAME (or REN) command changes the name of a file.

Type

RENAME is an INTERNAL command.

Syntax

RENAME [*<d1>*:[*<path1>*]*<filename1>* *<filename2>*

d1: the drive that contains the file to be renamed.

path1 the path to the file to be renamed.

filename1 the file to be renamed.

filename2 the new filename.

Operation

The RENAME command changes the name of a file (*<filename1>*) to a new filename (*<filename2>*). An optional drive designator and/or path can be specified for *<filename1>*, but not for *<filename2>*. Thus, the file remains on the same drive and in the same directory after its name has been changed.

The following command line renames the file **FILE1.TXT** to **MEMO1.TXT**:

```
RENAME FILE1.TXT MEMO1.TXT
```

The **RENAME** command accepts wildcards, which allows more than one file to be renamed at a time. For example, the command line:

```
RENAME *.TXT *.DOC
```

renames all the files with the extension of .TXT to .DOC. The actual filenames remain the same, whereas the filename extensions change.

In another example using wildcards, the command line:

```
RENAME FILE1 MEMO?
```

renames the file **FILE1** to **MEMO1**.

Any attempt to rename a file that doesn't exist or to rename a file to a name already present in the directory results in the following error message:

```
Duplicate file name or File not found
```

REPLACE

The **REPLACE** command selectively replaces target files with source files of the same name. It also selectively adds files from the source to the target.

Type

REPLACE is an **EXTERNAL** command.

Syntax

```
[<d>:][<path>]REPLACE [<d1>:][<path1>]<filename1>  
                        [<d2>:][<path2>][ /A ][ /P ]  
                        [ /R ][ /S ][ /W ]
```

d:	the drive that contains the REPLACE command.
path	the path to the REPLACE command.
d1:	the source drive. It contains the source file(s) (the files to be used to replace the files on the target drive or the files to be added to the target drive).
path1	the path to the source file(s).
filename1	the name(s) of the source file(s).
d2:	the target drive. It contains the target file(s) (the files to be replaced or added to).
path2	the path to the target file(s).

/A	the option that adds source files to the target instead of replacing existing files on the target. Only source files that don't already exist on the target are added. This option cannot be used with the /S option.
/P	the option that prompts you before replacing a target file or adding a source file to the target. This option allows you to selectively replace and add files.
/R	the option that replaces read-only files as well as unprotected files. If you don't specify this option, any attempt to replace a read-only file creates an error which stops the REPLACE process altogether.
/S	the option that searches all subdirectories on the target while replacing matching files. Only subdirectories on the target are searched; subdirectories on the source are not searched when this option is specified. This option cannot be used with the /A option.
/W	the option that waits for you to press any key before replacing or adding files. This allows you to switch flexible discs, if necessary. If this option isn't specified, the REPLACE command immediately begins replacing or adding files to the target.

Operation

The **REPLACE** command allows you to selectively replace the files on a disc or add files to it. As each file is replaced or added, its filename is displayed on the screen. At the conclusion of this process, one of the following messages is displayed:

XXX File(s) replaced/added

or

No files replaced/added

Examples of replacing and adding files are listed below.

Replacing Files. To replace a file on the target drive with a file on the source drive, enter the **REPLACE** command followed by the source drive and path, the source filename, and the target drive and path. For example, the following command line:

```
REPLACE A:\FILE1.TXT C:\
```

replaces the file **FILE1.TXT** in the root directory of drive **C:** with the file **FILE1.TXT** in the root directory of drive **A:**. If the **/S** option had been specified, all of the subdirectories on drive **C:** would have been searched for the file **FILE1.TXT** (in addition to the root directory).

Wildcards can be used to replace a group of files. For example, the command line:

```
REPLACE A:\*.TXT C:\
```

replaces all of the files with the filename extension of **.TXT** in the root directory of drive **C:** with all of the files with the same filename extension in the root directory of drive **A:**.

If you want **REPLACE** to prompt you before replacing files on the target drive, specify the **/P** option on the command line. For example, the command line:

```
REPLACE A:\*.TXT C:\ /P
```

prompts you with the following message:

```
REPLACE FILE1.TXT? (Y/N)
```

before replacing the first file with the filename extension of **.TXT** on the target drive. After you enter **"Y"** (Yes) or **"N"** (No) and press **Enter**, **REPLACE** prompts you with a second message:

```
REPLACE FILE2.TXT? (Y/N)
```

REPLACE continues to prompt you until all of the files with the filename extension of **.TXT** in the root directory of drive **A:** have been displayed.

Adding Files. To add a file to the target drive from the source drive, enter the **REPLACE** command followed by the source drive and path, the source filename, the target drive and path, and the **/A** option. For example, the command line:

```
REPLACE A:\FILE1.TXT C:\ /A
```

adds the file **FILE1.TXT** from the root directory of drive **A:** to the root directory of drive **C:**. Once again, wildcards can be used to add a group of files to the target drive.

If you want **REPLACE** to prompt you before adding files to the target drive, specify the **/P** option and the **/A** option on the command line.

The **REPLACE** command returns several error codes. The error codes and their descriptions are listed below.

1	Command Line Error
2	File Not Found
3	Path Not Found
5	Access Denied
8	Insufficient Memory
15	Invalid Drive
Other	Standard MS-DOS Error

These error codes can be used with the **ERRORLEVEL** option of the batch processing **IF** command. See the chapter entitled "Batch Processing" for additional information.

Notes

1. You can't use the **REPLACE** command to replace or add hidden or system files.

RESTORE

The **RESTORE** command restores one or more files that have been backed up with the **BACKUP** command.

Type

RESTORE is an **EXTERNAL** command.

Syntax

```
[<d>:][<path>]RESTORE <d1>:  
[<d2>:][<path2>]  
[<filename2>][/S][/P]
```

d: the drive that contains the **RESTORE** command.

path the path to the **RESTORE** command.

d1: the source drive. It contains the file(s) to be restored.

d2: the target drive. The drive that you want to contain the restored file(s). This is usually the drive from which the files were backed up originally.

path2 the path to the subdirectory that you want to contain the restored file(s).

filename2 the name(s) of the file(s) to be restored.

/S the option that restores the files in all subdirectories of the specified

directory, in addition to the files in the specified directory.

/P

the option that prompts you before restoring a file that (1) has been changed since it was backed up with the BACKUP command or (2) has a hidden or read-only attribute set. This prevents the loss of the most recent version of the file by inadvertently restoring an older version.

Operation

The RESTORE command restores files from one disc to another. It can be used to restore an entire disc or a group of files, depending on which files were originally copied using the BACKUP command.

In its most commonly used form, RESTORE restores files from one or more flexible discs onto a hard disc. For example, to restore all of the files on the disc in drive A: to drive C:, enter the following command line:

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

If the /S option isn't specified, only the files in the specified directory are restored.

Also, if more than one source disc is required to restore the contents of drive C:, RESTORE prompts you to insert the other discs. The discs must be inserted into drive A: in the order they were originally created.

To restore specific files, instead of an entire disc, enter the RESTORE command followed by the source and target drive designators, the path, and the names of the files you want to restore. For example, the command line:

```
RESTORE A: C:\USER#1\*.TXT
```

restores all of the files with the filename extension of .TXT on drive A: that were originally backed up from the subdirectory USER#1. The files are restored into the subdirectory on drive C: from which they were originally backed up (USER#1).

The RESTORE command returns the following error codes:

- | | |
|---|--------------------------------------------------|
| 0 | Normal completion |
| 1 | No files were found to restore |
| 2 | Some files not restored due to sharing conflicts |
| 3 | Restore terminated by user |
| 4 | Restore terminated due to error |

These error codes can be used with the ERRORLEVEL option of the batch processing IF command. See the chapter entitled "Batch Processing" for additional information.

Notes

1. You can't change the directory structure when restoring files. Files must be restored into the same directory from which they were originally backed up.
2. You can't restore shared files that you don't have access to. If you attempt to do so, RESTORE displays an error message.

RMDIR (Remove Directory)

The RMDIR command is used to remove a subdirectory from a disc.

Type

RMDIR is an INTERNAL command.

Syntax

RMDIR [**<d>:**][**<path>**]**<dirname>**

d: the drive that contains the subdirectory to be removed.

path the path to the subdirectory to be removed.

dirname the name of the subdirectory to be removed.

Operation

The RMDIR command allows you to remove an unwanted subdirectory from a disc. Before a subdirectory can be removed, it must be empty. That means it can't contain any subdirectories of its own or files. If you attempt to remove a subdirectory that isn't empty, the following error message is displayed:

Invalid path, not directory, or directory
not empty

To remove a subdirectory from a disc, enter the **RMDIR** command, the drive designator and path (both are optional), and the subdirectory name. For example, to remove the subdirectory **REPORTS** from drive C:, enter the following command line from the root directory or any subdirectory *except* **REPORTS**:

```
RMDIR \USER#1\REPORTS
```

Notes

1. **RD** is an alternate form of the **RMDIR** command. The command line **RD \USER#1\REPORTS** is the same as the command line **RMDIR \USER#1\REPORTS**.
2. You can't remove the root directory or the current directory.
3. A directory is empty if, when you enter the **DIR** command, the **.** and **..** entries are the only two entries listed. You can't remove these two entries.
4. A directory isn't empty if it contains hidden files.

SELECT

The SELECT command installs MS-DOS on a new disc with the country and keyboard codes of your choice.

Caution



DO NOT use the SELECT command on a disc that already has information on it. The format portion of the SELECT command erases the entire disc.

Type

SELECT is an EXTERNAL command.

Syntax

```
[<d>:][<path>]SELECT [[<d1>:]<d2>:[<path2>]]  
yyy xx
```

- d: the drive that contains the SELECT command.
- path the path to the SELECT command.
- d1: the source drive. A: and B: are the only valid source drives. If you don't specify a source drive, SELECT uses drive A: as the source drive.

d2:	the target drive. This is the drive to which the MS-DOS command files are to be copied. The target drive cannot be the same as the source drive. If you don't specify a target drive, SELECT uses drive B: as the target drive.
path2	the target path. This allows you to put the MS-DOS command files in the subdirectory of your choice. If you don't specify a target path, SELECT puts the command files in the root directory.
yyy	the country code. This tells MS-DOS the format for the date and time, the currency symbol, and the decimal separator.
xx	the keyboard code. This tells MS-DOS which keyboard layout to use.

Operation

The SELECT command allows you to install MS-DOS on a new disc with the country and keyboard codes of your choice. First, SELECT formats your new disc using the FORMAT command. Then, it copies the MS-DOS command files to your new disc using the XCOPY command. Finally, SELECT creates AUTOEXEC.BAT and CONFIG.SYS files on your new disc.

The country code is used to specify the date and time format, currency symbol and decimal separator. The keyboard code is used to specify the keyboard layout. For additional information about keyboard layouts, refer to the *Setting Up* manual that came with your computer.

The following table lists the country and keyboard code combinations offered by Hewlett-Packard. Additional **country codes** are listed under the COUNTRY command in

the chapter entitled "System Configuration".

Language	Country Code	Keyboard Code
Danish	045	DK**
Dutch	031	US*
English/UK	044	UK*
English/US	001	US*
Finnish	358	SU**
Flemish	032	FR*
French	033	FR*
German	049	GR*
Italian	039	IT*
Norwegian	047	NO**
Spanish/ European	034	SP*
Spanish/ Latin American	052	SP*
Swedish	046	SV**
Swiss/German	041	SD**
Swiss/French	041	SF**

- * These keyboard codes are available on the U.S. version of the MS-DOS disc.
- ** These keyboard codes are not available on the U.S. version of the MS-DOS disc. If you have a non-U.S. version of the MS-DOS disc, you can determine which keyboard codes are available by using the DIR command and looking for KEYBxx.COM files.

To use the SELECT command, make sure the MS-DOS disc is in the source drive. Then, enter the SELECT command, followed by the source drive, the target drive, the country code, and the keyboard code. For example, the following command line:

```
SELECT A: C: 001 US
```

formats the disc in drive C:, copies the MS-DOS command files from drive A: to drive C:, and creates AUTOEXEC.BAT and CONFIG.SYS files on drive C:. In addition, it specifies the country and keyboard codes as US.

When you execute the SELECT command, the following message is displayed:

```
This utility installs DOS on the target
disk for the first time. The target disk
will be erased. Proceed (Y/N)?
```

If you want to continue, enter "Y" and follow the instructions on your display screen. To leave the SELECT command, enter "N".

Notes

1. The CONFIG.SYS file created contains the following information:

COUNTRY=yyy

where yyy is the country code specified.

2. The AUTOEXEC.BAT file created contains the following information:

PATH \;[<path2>;]

KEYBxx

ECHO OFF

CLS

DATE

TIME

VER

where xx is the keyboard code specified. If the keyboard code is US, the KEYBxx command isn't placed in the file.

SET

The SET command is used to add, change, display and delete strings from the MS-DOS environment. The MS-DOS environment is used by the MS-DOS command processor and application programs during a work session.

Type

SET is an INTERNAL command.

Syntax

SET [<name>=<value>]]

name the name of a value in the MS-DOS environment to be set for the current work session.

value the value to be assigned to name.

Operation

The SET command allows you to set (add), change, display, and delete values that affect the environment used by the MS-DOS command processor and application programs during a work session. The values set are stored in the system's RAM; thus, they're removed each time you restart your system.

The MS-DOS environment consists of a series of character strings. Each string consists of a <name> and a <value>, separated by an equal (=) sign. To insert a string into the MS-DOS environment, enter the SET command followed by a name, the equal sign, and a value. For example, to insert the string FILE=FILE1.TXT into the MS-DOS

environment, enter the following command line:

```
SET FILE=FILE1.TXT
```

To change the value of an existing string, enter the SET command followed by the name, the equal sign, and the new value. For example, the command line:

```
SET FILE=LETTERS
```

changes the value of FILE from FILE1.TXT to LETTERS.

To display the current strings, enter the SET command without parameters. For example, the command line:

```
SET
```

causes MS-DOS to display the following information:

```
COMSPEC=d:\COMMAND.COM  
FILE=LETTERS
```

Notice that the MS-DOS environment contains the string COMSPEC=d:\COMMAND.COM (d: is the drive from which you started your system). This is a permanent string; the MS-DOS environment always contains it. It's used to reload the MS-DOS command processor, whenever necessary.

To delete a string, enter the SET command followed by the name and the equal sign. Don't enter a value. For example, the command line:

```
SET FILE=
```

deletes the string created in the first example.

Notes

1. When you execute the **PATH** and **PROMPT** commands, strings are placed in the MS-DOS environment. As a result, the command line **SET PATH=** has the same effect as issuing the **PATH** command, and the command line **SET PROMPT=** has the same effect as issuing the **PROMPT** command.
2. The **SET** command can pass variable names to a batch file. Refer to the section on *Batch Files with Replaceable Parameters* in the chapter entitled "Batch Processing" for additional information.
3. If you execute commands that load resident code (such as **GRAPHICS**, **KEYBxx**, **MODE**, or **PRINT**) before you execute the **SET** command, there may not be enough room in the MS-DOS environment to store the strings you specified. If there isn't enough room, the following message is displayed:

Out of environment space

To correct this situation, execute the **SET** command first.

4. On the other hand, a copy of the environment is saved with commands that load resident code. Therefore, if you execute these commands before you execute the **SET** command and create a large MS-DOS environment, you'll save usable memory.

SHARE

The SHARE command loads support for file sharing in a network environment.

Type

SHARE is an EXTERNAL command.

Syntax

```
[<d>:][<path>]SHARE [/F:<filesize>]  
                        [/L:<locks>]
```

- | | |
|------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| d: | the drive that contains the SHARE command. |
| path | the path to the SHARE command. |
| /F: | the option that reserves space (in bytes) for the names of the shared files. Each file opened in share mode requires 11 bytes, plus the number of bytes (characters) in the path and filename. The default value is 2048 bytes. |
| /L: | the option that reserves space for the number of locks allowed. The default value is 20 locks. |

Operation

The SHARE command allows file sharing in a network environment. It can be executed from the MS-DOS prompt or included in the AUTOEXEC.BAT file on your network software disc.

Caution



Before attempting to use the SHARE command, check the documentation that came with your network to make sure this command is allowed. If it isn't allowed, the use of this command may cause you to lose data.

To load support for file sharing, enter the following command line:

```
SHARE
```

Because the /F and /L options aren't specified, the default values for the number of shared files and the number of locks allowed are used.

Once support for file sharing is loaded, all read and write requests are validated against the file sharing code.

Support for file sharing can only be loaded once. If you attempt to load it more than once, the following error message is displayed:

```
SHARE already installed
```

Notes

1. When you execute the SHARE command, MS-DOS checks the FCB (File Control Block) control table. If you specified FCBS=4,0 (the default value) in the CONFIG.SYS file, the table is adjusted to 16,8. For additional information, refer to the FCBS command in the chapter entitled "System Configuration".

SORT

The SORT command reads data from the standard input device, sorts it, and writes the sorted data out to the standard output device.

Type

SORT is an EXTERNAL command.

Syntax

[<d>:][<path>]SORT [/R][/+<n>]

d: the drive that contains the SORT command.

path the path to the SORT command.

/R the option that sorts data in reverse order (i.e., from Z to A instead of A to Z).

/+<n> the option that specifies the column in which the sort is to begin. <n> must be a positive integer; the default for <n> is column 1.

Operation

SORT is one of three filters provided with the MS-DOS operating system. For a detailed explanation of filters, refer to the chapter entitled "Redirecting Input and Output".

SORT reads the data from the standard input device (such as output from a command), sorts it, and then writes it out to the standard output device (usually, a file or the display screen). SORT can read up to 63 Kb of input data at one time.

SORT is designed to sort text. Therefore, it works with lines of data (all of the characters on a line, including the CR/LF sequence). SORT arranges the lines of data such that the characters in column <n> ascend (or descend if the /R option is specified) in alphabetical order. If the nth character in two or more lines is the same, SORT compares the nth+1 character. SORT continues this compare process until the characters are different or the end of the line of data (CR/LF) is encountered.

The following command line takes the output from the DIR command, sorts it by size (file size starts in column 14) and displays it on the screen (your terminal).

```
DIR | SORT /+14
```

The output might look like this:

1313195D		0	12-09-87	3:44p
1313963		0	12-09-87	3:44p
FILE3	TXT	140	10-04-87	1:37p
FILE1	TXT	230	11-16-87	10:53a
MEM01		261	11-16-87	2:17a
FILE2	TXT	676	03-03-87	1:39p

Assume you have a list of names stored in a file called UNSORT.TXT, and you want to sort them in reverse alphabetical order and place them in a new file called SORTED.TXT. You can do this by sorting UNSORT.TXT and then redirecting the output to SORTED.TXT. The command line to do this is:

```
SORT /R <UNSORT.TXT >SORTED.TXT
```

MS-DOS opens the new file SORTED.TXT and directs the results of the reverse sort of UNSORT.TXT to this file.

Notes

1. Non-U.S. characters (128-255) are translated to U.S. characters before they are sorted. For a list of these translations, refer to the appendix entitled "The SORT Command Collating Sequences".
2. SORT ignores upper- and lower-case.

SUBST

The SUBST command allows a new drive, called a "virtual drive", to be substituted for an existing drive and path.

Type

SUBST is an EXTERNAL command.

Syntax

[<d>:][<path>]SUBST <d1>: <d2>:<path2>

or

[<d>:][<path>]SUBST <d1>: /D

or

[<d>:][<path>]SUBST

d:	the drive that contains the SUBST command.
path	the path to SUBST command.
d1:	the virtual drive. This is the new drive to be substituted for the existing drive and path. This drive can't be the active drive.
d2:	the existing drive. The virtual drive is to be substituted for this drive.
path2	the existing path to a subdirectory. The virtual drive is to be substituted for this path.
/D	the option that deletes a substitution (a virtual drive).

Operation

The SUBST command allows you to substitute a virtual drive for an existing drive and path. This capability is useful if you have subdirectories that you use frequently or application programs that don't recognize paths.

The SUBST command creates a new drive, called a virtual drive, and substitutes it for an existing disc drive and path specified. As a result, the existing subdirectory becomes the root directory of the virtual drive.

To create a virtual drive, enter the SUBST command followed by the virtual drive, the existing drive, and the existing path. For example, the following command line:

```
SUBST E: C:\USER#1\REPORTS
```

creates the virtual drive E:. The subdirectory REPORTS on drive C: is now the root directory of drive E:. To see the contents of the file REPORT1 in the subdirectory REPORTS, all you have to enter is:

```
TYPE E:\REPORT1
```

instead of:

```
TYPE C:\USER#1\REPORTS\REPORT1
```

Thus, you can think of a virtual drive as a "nickname" for an existing drive and path.

To display the current substitutions, enter the SUBST command without parameters. For example, the command line:

```
SUBST
```

causes MS-DOS to display the following information:

```
E: => C:\USER#1\REPORTS
```

To cancel a substitution, enter the SUBST command followed by the virtual drive and the /D option. For example, the command line:

```
SUBST E: /D
```

cancels the substitution made in the first example.

Notes



1. The value assigned to the virtual drive (<d1>) must be greater than or equal to the value assigned to the LASTDRIVE command in your CONFIG.SYS file. If you don't have a LASTDRIVE command in your CONFIG.SYS file, the default value for LASTDRIVE is drive E:. This means that you can specify drive A:, B:, C:, D:, or E: as the virtual drive. If you want to specify a drive greater than E:, you need to change the value assigned to LASTDRIVE first.
2. <d1> and <d2> cannot be the same; they must be different. Also, <d1> can be the same as the active drive.

3. As mentioned, when a drive and path are substituted, the subdirectory specified by <path2> becomes the root directory of the virtual drive. However, the relationship between the specified subdirectory and its subdirectories remains the same. Therefore, in the first example, if the subdirectory REPORTS had a subdirectory called PAYROLL, you could reference PAYROLL with the path E:\PAYROLL.
4. Even though a drive and path are substituted, you can still reference them by specifying the old drive and path. Using the first example, you can still reference the subdirectory REPORTS with the path C:\USER#1\REPORTS.
5. If you specify an existing physical drive as the virtual drive (<d1>), that physical drive is inaccessible while the substitution is in effect. For example, if you specify drive B: as the virtual drive, it is inaccessible while the substitution is in effect.
6. You can't use the SUBST command with a drive and/or path that has been redirected over a network. If you attempt to do this, the following error message is displayed:

Cannot SUBST a network drive.
7. Use the following commands with care on a substituted drive:

CHDIR	PATH
MKDIR	RMDIR

8. DO NOT use the following commands on a substituted drive:

ASSIGN	FORMAT
BACKUP	JOIN
DISKCOMP	LABEL
DISKCOPY	RESTORE
FDISK	

SYS (System)

The SYS (System) command transfers the hidden MS-DOS files to the disc in the specified target drive.

Type

SYS is an EXTERNAL command.

Syntax

[<d>:][<path>]SYS [<d1:>]

d: the drive that contains the SYS command.

path the path to the SYS command.

d1: the target drive. The drive to which the system files are to be transferred.

Operation

The SYS command transfers the two hidden MS-DOS files to the disc in the target drive. This command allows you to:

- Put the MS-DOS operating system on an application work disc that was FORMATED with the /B option.
- Upgrade or replace the MS-DOS operating system on a disc.

To transfer the two hidden MS-DOS files, enter the SYS command followed by the target drive. For example, to transfer the two hidden MS-DOS files from the disc in drive A: to the disc in drive C:, enter the following command line:

```
A>SYS C:
```

Notes

1. The two hidden MS-DOS files (IBMBIO.COM and IBMDOS.COM) must be the first two directory entries in the root directory of the active drive (the source drive).
2. The first two allocatable contiguous sectors of the disc in the target drive must be unused (they're automatically left unused when you FORMAT a disc with the /B option).

If you attempt to transfer the files to a disc whose first two sectors are NOT contiguous, the following error message is displayed:

```
Destination disk is too fragmented
```

If you attempt to transfer the files to a disc that doesn't have enough room to hold the two files, the following error message is displayed:

```
No room for system on destination disk
```

3. The SYS command doesn't transfer the COMMAND.COM file. To copy this file to the disc in the target drive, use the COPY command.

TIME

The TIME command is used to set or display the system time.

Type

TIME is an INTERNAL command.

Syntax

TIME [**<hh>**:[**<mm>**:[**<ss>**[**<.xx>**]]]]

hh a 1- or 2-digit number from 0 to 23
 representing the hour.

mm a 1- or 2-digit number from 0 to 59
 representing minutes.

ss a 1- or 2-digit number from 0 to 59
 representing seconds.

.xx a 1- or 2-digit number from 0 to 99
 representing hundredths of a second.

Operation

The TIME command allows the system time to be set or displayed. MS-DOS uses the TIME and DATE values to record the exact time files are created or last updated. These values are affected **ONLY** by modifications to your files. As you build your library of files, you will find this information both necessary and useful.

In order to set the time, the hours (hh:) must be entered. The minutes, seconds and hundredths of a second

(mm:ss.xx) are optional; MS-DOS sets them to 00:00.00 if they aren't entered. To set the system time to 1:30 p.m., enter the following command line:

```
TIME 13:30:00.00
```

Note that 1:30 p.m. is entered as 13:30. That's because MS-DOS uses a 24-hour clock.

To display the current system time, enter the command line:

```
TIME
```

MS-DOS responds by displaying the current system time and asking for a new time as follows:

```
Current time is 13:30:00.00
Enter new time:
```

You can enter a new system time or press **Enter** to retain the current system time.

MS-DOS checks for invalid times and separators. If you enter an invalid time or use an incorrect separator, the system responds with:

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

This response is repeated until you enter a valid time in the correct format.

The **TIME** command sets the MS-DOS system time, but it doesn't change the time kept by the "system clock". The system clock is a battery-powered clock that keeps track of the time as well as the date. It increments the time and date even when the system is turned off. The system clock resets the system time and date each time you start (or restart) your system. To change the system clock, refer to the *Setting Up* manual that came with your computer.

Notes

1. The **TIME** display format is determined by the **COUNTRY** command in the **CONFIG.SYS** file. To see the **TIME** display format for your computer, enter the **TIME** command without parameters.
2. If you start your system without **PAM** or an **AUTOEXEC.BAT** file, you are prompted to enter the time and date every time you start it. However, if you have an **AUTOEXEC.BAT** file, you aren't prompted for the time and date unless your **AUTOEXEC.BAT** file includes the **TIME** and **DATE** commands.

TREE

The **TREE** command displays the directory structure of a disc (the root directory and all of the subdirectories). It can also be used to display all of the files in the root directory and the subdirectories.

Type

TREE is an **EXTERNAL** command.

Syntax

`[<d>:][<path>]TREE [<d1>:][/F]`

- | | |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| d: | the drive that contains the TREE command. |
| path | the path to the TREE command. |
| d1: | the drive that contains the directory structure you want to display. If you don't specify a drive, the directory structure of the disc in the active drive is displayed. |
| /F | the option that displays the names of the files in the root directory and the subdirectories. |

Operation

The **TREE** command allows you to view the directory structure of a disc. If the **/F** option is specified, the names of the files in the root directory and the subdirectories are also displayed.

To display the directory structure and the names of the files on the disc in drive C:, enter the command line:

```
TREE C:/F
```

For each directory found, MS-DOS displays the path to the directory, the names of its subdirectories, and the names of its files.

The information on each directory is displayed in the following format:

```
Path: C:\USER#1
```

```
Sub-directories: FORMS  
                  REPORTS
```

```
Files:           FILE1.TXT  
                  FILE2.TXT
```

In this particular example, the root directory has a subdirectory called **USER#1**. The subdirectory **USER#1**, in turn, has two subdirectories of its own: **FORMS** and **REPORTS**. In addition, the subdirectory **USER#1** contains two files: **FILE1.TXT** and **FILE2.TXT**. The filenames are only displayed if the **/F** option is specified.

To redirect the output from the **TREE** command so that it goes to a printer instead of the display screen, enter the following command line:

```
TREE C:/F >PRN
```

Notes

1. The output from the **TREE** command can be quite lengthy, particularly if the **/F** option is specified. If you want the display to pause after each page full of data, pipe the output from **TREE** through the **MORE** filter, as in:

```
TREE C:/F | MORE
```

TYPE

The **TYPE** command displays the contents of a file on the screen.

Type

TYPE is an **INTERNAL** command.

Syntax

TYPE [**<d>:**][**<path>**]**<filename>**

d: the drive that contains the file to be displayed.

path the path to the file to be displayed.

filename the name of the file to be displayed.

Operation

The **TYPE** command allows you to view the contents of a file; you can see it, but you can't modify it in any way. To alter the contents of a file, use **EDLIN** or one of the HP text processing applications.

To display the contents of a file, enter the **TYPE** command followed by the filename. For example, the following command line:

```
TYPE FILE1.TXT
```

displays the contents of the file **FILE1.TXT** on the screen.

The TYPE command displays a file exactly as it was entered, with one exception. TAB characters are expanded to the nearest 8-character column (8, 16, 24, etc.).

The TYPE command is primarily intended to display ASCII text files. As a result, a program or data file that contains non-ASCII characters produces an unintelligible display.

Notes

1. Wildcards can't be used in the filename because TYPE can only display the contents of one file at a time. If you attempt to use wildcards, TYPE displays an error message.
2. If you want to send the contents of a file to a printer, redirect the output to the PRN device, as in:

```
TYPE FILE1.TXT>PRN
```

To pause the display, pipe the output through the MORE filter, as in:

```
TYPE FILE1.TXT | MORE
```

For additional information on redirection and piping, refer to the chapter entitled "Redirecting Input and Output".

VER (Version)

The VER command displays the MS-DOS version number on the screen.

Type

VER is an INTERNAL command.

Syntax

VER

Operation

The VER command displays the current MS-DOS version number. For example, the command line:

VER

produces a display similar to the following:

MS-DOS Version 3.20

VERIFY

The VERIFY command turns the disc write verify switch on and off.

Type

VERIFY is an INTERNAL command.

Syntax

VERIFY [ON|OFF]

ON this turns the verify switch ON. When the verify switch is ON, MS-DOS verifies that data is correctly written to disc.

OFF this turns the verify switch OFF. OFF is the default setting.

Operation

The VERIFY command allows you to turn the disc write verify switch ON and OFF, and to display the current status of the switch.

When the verify switch is ON, every time MS-DOS writes data to a disc during a work session, it verifies that the data is intact. To turn the verify switch ON, enter the following command line:

VERIFY ON

If MS-DOS is unable to successfully write data to disc when the verify switch is ON, an error message is displayed.

The **VERIFY ON** setting remains in effect until you turn off or restart your system, you enter the **VERIFY** command with the **OFF** parameter, or an application program resets it (see the *Notes* section below for additional information).

To check the current status of the verify switch, enter the **VERIFY** command without parameters. For example, if the verify switch is **ON** and you enter the following command line:

```
VERIFY
```

MS-DOS responds with:

```
VERIFY is off
```

Notes

1. The verify switch can be turned **ON** and **OFF** through a system call from an application program. Thus, it's possible for **VERIFY** to be **ON**, even though you haven't issued the **VERIFY** command with the **ON** parameter.
2. Writing to disc takes longer when the verify switch is **ON**.

VOL (Volume)

The Volume command displays the volume label of a disc.

Type

VOL is an INTERNAL command.

Syntax

VOL [<d>:]

d: the drive containing the disc whose label you
 want to display.

Operation

The VOL command displays the volume label of the disc in the specified drive. If no drive is specified, MS-DOS displays the volume label of the disc in the active drive. For example, the command line:

```
C>VOL
```

results in a display similar to the following:

```
Volume in drive C is HARDDISC
```

If the disc doesn't have a label, the following message is displayed:

```
Volume in drive C has no label
```


Notes

1. You can create a volume label with the **FORMAT** and **LABEL** commands. For additional information, see the **FORMAT** and **LABEL** command descriptions in this chapter.

XCOPY

The XCOPY command copies groups of files, including files in subdirectories. It can also copy the subdirectories, themselves.

Type

XCOPY is an EXTERNAL command.

Syntax

```
[<d>:][<path>]XCOPY [<d1>:][<path1>][<filename1>]
                    [<d2>:][<path2>][<filename2>]
                    [/A][/D:<mm-dd-yy>][/E][/M]
                    [/P][/S][/V][/W]
```

d:	the drive that contains the XCOPY command.
path	the path to the XCOPY command.
d1:	the source drive. This is the drive that contains the source file(s) (the files to be copied).
path1	the path to the source file(s). If a path isn't specified, source files in the current directory are copied.
filename1	the name(s) of the source file(s). If a filename isn't specified, *.* is assumed.
d2:	the target drive. This is the drive to which the source file(s) are to be copied.

path2	the path to the target file(s).
filename2	the name(s) of the target file(s). If a filename isn't specified, *.* is assumed.
/A	the option that only copies source files whose archive bits are set to one. This option, unlike the /M option, does <i>not</i> modify the archive bits of the source files.
/D	the option that only copies source files that have been modified on or after the date specified by <mm-dd-yy>. The date syntax (<mm-dd-yy>) can vary depending upon the COUNTRY command in the CONFIG.SYS file.
/E	the option that copies subdirectories even if they don't contain any files (empty subdirectories). This option must be used in conjunction with the /S option.
/M	the option that only copies source files whose archive bits are set to one. This option, unlike the /A option, clears (sets to zero) the archive bits of the source files after they are copied. Thus, this option allows XCOPY to be used to backup files.
/P	the option that prompts you with Y/N? before copying each source file. This allows you to confirm whether or not you want to create each target file.
/S	the option that copies source files in all the subdirectories of the specified directory, in addition to the files in the specified directory. The subdirectory structure (with the exception of empty

subdirectories) is copied along with the files. Specify the /E option to copy empty subdirectories. If you omit the /S option, only the files in the specified directory are copied.

/V the option that verifies data as it is written to the disc in the target drive.

/W the option that causes XCOPY to wait before starting to copy source files. XCOPY displays the following message:

Press any key to begin copying
file(s)

You can insert a source and/or target disc and then press any key to continue. Or, you can press **CTRL** **C** to cancel the XCOPY command and return to the MS-DOS prompt.

Operation

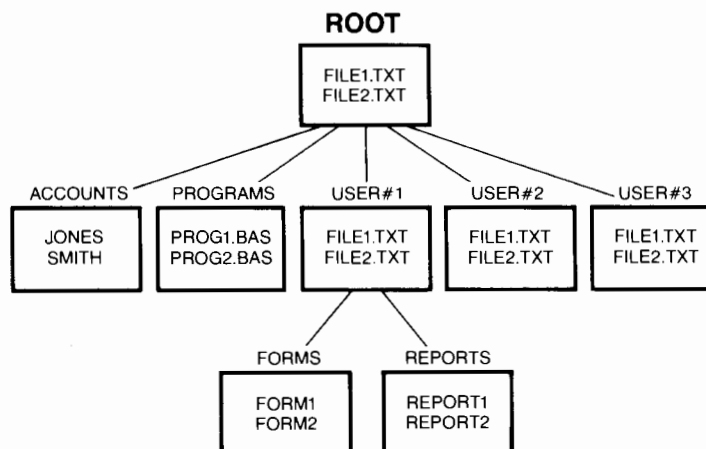
The XCOPY command allows you to copy files and subdirectories from one disc to another. For example, the following command line:

```
XCOPY A:\ B:\ /S /E
```

copies all of the files and subdirectories from the disc in drive A: to the disc in drive B:, including empty subdirectories.

To copy a portion of the directory structure of a disc, enter the XCOPY command followed by the path to the portion of the directory structure you want to copy.

For example, suppose the directory structure of the disc in drive A: looks like this:



and you want to copy everything in the subdirectory **USER#1** to the disc in drive B:. To do this, enter the following command line:

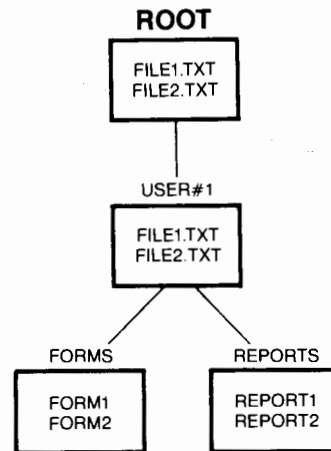
```
XCOPY A:\USER#1 B:\USER#1 /E /S
```

MS-DOS displays the following prompt:

```
Does USER#1 specify a filename or a
directory name on the target (F=file,
D=directory)?
```

Type **D** to indicate a directory and press **Enter** .

The directory structure of the disc in drive B: now looks like this:



The XCOPY command doesn't disturb the existing directory structure on the target disc. In other words, the new files and subdirectories are added to any existing files and subdirectories.

Notes

1. You must specify either the source drive (<d1>), the source path (<path1>), or the source filename (<filename1>) on the command line. You cannot omit all three.
2. If the path to the target files doesn't exist, XCOPY automatically creates it before it copies any files.
3. The XCOPY command, like the COPY command, allows you to rename files as you copy them. For example, the command line:

```
XCOPY A:\*.TXT B:\*.LST
```

copies all of the files with the filename extension of .TXT in the root directory of the disc in drive A: to the disc in drive B:. At the same time, it changes the filename extension from .TXT to .LST.

4. Unlike the COPY command, you can't use the XCOPY command to copy to or from system devices (for example, CON or LPT1).
5. The XCOPY command doesn't copy hidden source files or copy over read-only target files.
6. XCOPY *may* prompt you to specify whether the target is a file or a directory. Type F to indicate a file or D to indicate a directory, and then press to continue the XCOPY process.



5

Batch Processing

In this Chapter we'll examine a powerful feature of MS-DOS: Batch Processing. As you use MS-DOS, you may find yourself executing certain command sequences frequently. You can automate these sequences by using the MS-DOS Batch Processing capability.

What is Batch Processing?

Batch Processing is a two step process. The first step involves putting MS-DOS commands and/or the names of application programs into a single file, called a "batch" file. The second step involves executing the batch file as a command on the MS-DOS command line. When you execute the batch file, the batch processor in MS-DOS executes the commands and programs in the batch file one at a time; thus, saving you from individually executing them yourself.

In addition to MS-DOS commands and the names of application programs, the batch processor has its own set of special commands that can be included in batch files. These special commands provide a rudimentary "program language" which further enhances the capabilities of MS-DOS batch processing. We'll discuss them later in this chapter.

There is one limitation to using batch processing that you should be aware of: if an MS-DOS command prompts you for a response, that response must still be typed at the keyboard. You can't include the response in your batch file. This prevents you from using batch files for

unattended system operations which require responses to command prompts. Even so, batch files that include these commands are still useful for eliminating the keyboard entry of command lines.

Valid Batch Filenames

A batch filename can be any valid MS-DOS filename. However, it must have a filename extension of .BAT.

In addition, the batch filename shouldn't be the same as an MS-DOS command (internal or external) or an application program. If two files in the same directory have the same name, and one has an extension of .COM (or .EXE) and the other has an extension of .BAT, MS-DOS always executes the former instead of the latter. Thus, MS-DOS allows you to create a batch file with the same filename as an MS-DOS command or application program, but it won't allow you to execute it.

Creating a Batch File

There are a number of ways to create a batch file. You can use:

- Any word processor or editor that stores text in ASCII format
- EDLIN, the MS-DOS editor
- The MS-DOS COPY command

We'll use the COPY command in our examples, since it's usually the easiest way to create a batch file.

You can use the COPY command to create a file directly from the keyboard. To do this, enter the COPY command followed by the CON (console) device and the batch filename (with a filename extension of .BAT) on the MS-DOS command line. This command syntax is shown below.

```
COPY CON <filename>.BAT
```

Then, type each command on a separate line. When you're finished, hold down **CTRL** and press **Z**, or press **F5**.

Let's create a sample batch file with the MS-DOS commands DIR and CHKDSK in it. We'll name the file BATCH1.BAT and use COPY to create it on drive C:.

```
COPY CON BATCH1.BAT
DIR
CHKDSK
CTRL Z
1 File(s) copied
```

Executing a Batch File

Executing a batch file is similar to executing an MS-DOS command. You simply enter the filename (without the .BAT extension) on the MS-DOS command line. For example, to execute the batch file we just created, enter the command line:

```
BATCH1
```

This batch file lists the names of the files in the current directory on drive C:, then executes CHKDSK on drive C: to determine the amount of disc space available.

C>DIR

Volume in drive C is HARDDISC
Directory of C:\

COMMAND	COM	23210	9-12-86	12:00a
ANSI	SYS	2501	9-12-86	10:28p
VDISK	SYS	2626	9-27-86	4:55p
.
.
.
CONFIG	SYS	128	9-30-86	8:49a
21 File(s)		153600 bytes free		

C>CHKDSK

362496 bytes total disk space
38912 bytes in 3 hidden files
169984 bytes in 21 user files
153600 bytes available on disk

655360 bytes total memory
215744 bytes free

You can terminate the execution of a batch file by pressing either of the MS-DOS abort key sequences, **CTRL** **Break** or **CTRL** **C**. However, you are not returned directly to the MS-DOS prompt. Instead, the prompt:

Terminate batch job (Y/N)?

appears on the screen. If you enter "Y", you are returned to the MS-DOS prompt. If you enter "N", execution of

the batch file resumes with the next command line in the batch file. The command line that was executing when **CTRL Break** or **CTRL C** was entered isn't resumed.

You can change the current directory and/or active drive during the execution of a batch file. MS-DOS "remembers" the directory and drive that contain the batch file. However, you can't remove the disc that contains the batch file while the batch file is executing. If you do remove the disc, MS-DOS issues the following error message:

```
Not ready error reading drive <d>:  
Abort, Retry, Ignore?
```

To continue executing the batch file, re-insert the disc into its original drive and enter "R" (retry). To stop executing the batch file and return to the MS-DOS prompt, enter "A" (abort). Entering "I" (ignore) isn't recommended.

Chaining Batch Files

Batch files can be "chained" for execution; that is, a batch file can contain the name of another batch file as a command line. This allows a long string of command lines to be assembled from several smaller batch files. There is no limit to the number of batch files that you can chain together.

There is one limitation you should be aware of when chaining batch files: the command line containing the name of the batch file you want to chain to should be the last command line in the batch file. This is because any command lines after the command line that contains the name of a batch file are not executed. Once another batch file is invoked by the first batch file, there is no way to return to that first batch file (except to chain to it).

The AUTOEXEC.BAT File

MS-DOS allows you to create a special batch file called AUTOEXEC.BAT. This file contains MS-DOS commands you want executed every time you start (or restart) your system. You create this file in the root directory of the drive from which you start your system. Then, when you start your system, MS-DOS looks for this file. If MS-DOS finds it, the commands in it are executed before the MS-DOS prompt is displayed.

The AUTOEXEC.BAT file is very useful for executing MS-DOS commands which initialize or configure the system in some way. For example, if you want the printer connected to serial port 1 (COM1) to be active all the time, you can put a command similar to the following:

```
MODE LPT1:=COM1,96,E,8,1
```

in the AUTOEXEC.BAT file. Then, MS-DOS executes this command every time you start (or restart) your system.

Batch Files with Replaceable Parameters

You can create batch files with replaceable parameters (or variables). These replaceable parameters allow you to alter the data used by the batch file each time it is executed. When you enter the batch filename on the MS-DOS command line, you specify the data you want to use. Then, as the batch file is executed, the data is substituted for the replaceable parameters.

You can specify replaceable parameters in two ways:

- The %0 - %9 positional parameters
- The SET Command

Using %0 - %9

Ten positional parameters (%0 - %9) can be inserted into batch files to specify replaceable parameters. %0 is always replaced by the drive designator (optional) and the name of the batch file. Note, however, that the batch filename that replaces %0 does not include the filename extension (.BAT). %1 is replaced by the first data value specified on the command line, %2 is replaced by the second data value, and so on.

To illustrate the use of replaceable parameters, let's create a batch file called BATCH2.BAT on drive C:

```
COPY CON BATCH2.BAT
TYPE %0.BAT
DIR %1
CHKDSK %2
COPY %1*.* %2
CTRL (Z)
1 File(s) copied
```

To execute it, you must specify the batch filename and two data values on the command line. The batch filename will be substituted for %0 and the two data values will be substituted for %1 and %2. For example, if you enter the command line:

```
BATCH2 A: B:
```

%0 is replaced by BATCH2, %1 is replaced by A: and %2 by B: As a result, the following commands are executed:

```
C>TYPE BATCH2.BAT
C>DIR A:
C>CHKDSK B:
C>COPY A: *.* B:
```

The batch file itself isn't modified.

If the % character is contained in a name in the batch file, it must be entered as a double % (%%) to differentiate it from a replaceable parameter. For example, to specify the file FILE1%.EXE in a batch file, enter FILE1%%.EXE.

If there are more replaceable parameters in the batch file than data values on the command line, the unspecified parameters are replaced with blanks. This may or may not cause problems as the batch file executes. For example, if the batch file BATCH2.BAT is invoked with the following command line:

```
C>BATCH2 A:
```

the second data value is omitted. Therefore, the batch file executes the following commands:

```
C>TYPE BATCH2.BAT
C>DIR A:
C>CHKDSK
C>COPY A:*.*
```

In this instance, drive designator B: (the second data value) is omitted. As a result, CHKDSK checks drive C: instead of drive B:, and all the files in the current directory of drive A: are copied to drive C:.

The number of positional parameters can be increased beyond ten by using the batch command SHIFT. This command is discussed later in this chapter.

Using the SET Command

Another method of replacing parameters is to use the MS-DOS SET command. The SET command can be used to place any number of name and parameter pairs in the MS-DOS environment (refer to the chapter entitled "System Configuration" for additional details). The name is enclosed by percent sign characters (%), and inserted into the batch file. When the batch file is executed, the batch processor searches the MS-DOS environment for any names it encounters. Then, the corresponding parameters are substituted.

Let's start by assigning the parameter EXAMPLE.TXT to the name FILE. The following command line accomplishes this.

```
C>SET FILE=EXAMPLE.TXT
```

Then, the name FILE is inserted in the batch file called BATCH3.BAT as shown below:

```
C>COPY CON BATCH3.BAT
TYPE %FILE%
  CTRL  Z
1 File(s) copied
```

When this batch file is executed, the file EXAMPLE.TXT is TYPED.

Note



The MS-DOS environment is lost when the system is turned off or restarted. Thus, after you start or restart your system *and* before you execute a batch file, you must use the SET command to assign any names contained in that batch file.

Batch Commands

This section discusses the batch commands. These commands can be included in batch files along with regular MS-DOS commands and application program names to expand the capabilities of the batch processor. These commands should be used only within batch files, unless otherwise noted.

ECHO

The ECHO command turns the screen display on or off during the execution of batch commands, or displays a message.

Syntax

ECHO [ON|OFF|<message>]

Operation

ECHO ON instructs the batch processor to display all command lines as they are executed. This is the default state.

ECHO OFF suppresses the display of command lines as they are executed. However, displays resulting from the execution of commands and application programs, themselves, still appear.

ECHO followed by a message displays the message regardless of the current ECHO state (ON or OFF). This form of the command functions identically to the REM command, with the exception that the message is displayed even in the ECHO OFF state.

If the ECHO command is entered without parameters, the current state (ECHO ON or ECHO OFF) is displayed.

The following batch file illustrates all four forms of the ECHO command:

```
ECHO OFF
ECHO DRIVE B:
DIR B:/W
ECHO ON
ECHO
DIR A:/W
```

When this file is executed, the following display occurs:

```
A>ECHO OFF
DRIVE B:

Volume on drive B has no label
Directory of B:\

FILE1.TXT      FILE2.TXT

2 file(s)      265234 bytes free

A>ECHO
ECHO is on
A>DIR A:/W

Volume on drive A has no label
Directory of A:\

FILE1.TXT      FILE2.TXT

2 file(s)      265234 bytes free
```

Notes

1. The ECHO command can be executed from the MS-DOS command line.

FOR

The FOR command allows the iterative execution of MS-DOS commands.

Syntax

FOR %%<variable> IN (<set>) DO <command>

Operation

The FOR command accepts three arguments: variable, set, and command.

<variable> is a positional variable consisting of any single character. (To avoid a conflict with the positional parameters %0 - %9, don't use digits 0 - 9.) The variable is made equal to each value in the set.

<set> consists of a series of values (such as filenames) separated by spaces, or a filename with one or more wildcards. Two examples of valid sets are shown below:

(CHPT1.TXT CHPT2.TXT INDEX.TXT)

(* .TXT)

<command> is a command line that invokes an MS-DOS command or an application program. Each time the variable is made equal to a value in the set, the command line after DO is executed. In most cases, the variable is included on the command line.

For example, the following command line finds all of the chapter files for a book, and then runs them sequentially through an application program called INDEX.

```
FOR %%A IN (CHPT?.TXT) DO INDEX %%A
```

In this example, a variable called "A" is made equal to the filename of the first chapter (CHPT1.TXT) and the command line INDEX CHPT1.TXT is executed. Then, A is made equal to the filename of the second chapter (CHPT2.TXT) and the command line INDEX CHPT2.TXT is executed. This iterative process continues until A is made equal to the filename of the last chapter.



Notes

1. The FOR command cannot be nested. Only one FOR command can be included on each command line.
2. Paths are allowed in the set of variables.
3. The FOR command can be executed from the MS-DOS command line. The only difference is that the variable need only be preceded by one "%".

GOTO

The GOTO command transfers control in a batch file to the line following the line identified by a label.

Syntax

GOTO <label>

Operation

The <label> can be any string of characters. However, only the first eight characters are significant (i.e., used to differentiate it from other labels). A label must be preceded by a colon (:) and there shouldn't be anything else on the line with it. The GOTO command executes the command on the line following the label.

The following example transfers control from the line with the command GOTO TWO to the line following the :TWO label:

```
GOTO TWO
.
.
.
:TWO
REM This is a comment line.
```

As a result, the following remark is displayed:

```
REM This is a comment line.
```

Notes

1. The GOTO command can be used with the IF command. This provides batch files with IF-THEN branching capabilities.
2. Labels within a batch file are not displayed when the batch file is executed. Anything on the same line as the label is ignored. Thus, a label which is not referenced may be used as a comment line and is not displayed when the batch file is executed.

IF

The IF command allows the conditional execution of MS-DOS commands in batch files.

Syntax

IF [NOT] <condition> <command>

Operation

The IF command examines the <condition>, and executes the <command> if either (1) the condition is true or (2) the <condition> is false and NOT is included on the command line. The IF command can test for the following conditions:

ERRORLEVEL
<number>

This condition is true if the previous application program executed had an error code of <number> or higher. An error code is set by a specific application program. It is returned to the operating system when the application program is finished.

<string1>==
<string2>

This condition is true if the two strings are identical. Either or both strings may be replaceable parameters. For example, both of the following can be included in a batch file:

```
IF %1==EXAMPLE
IF %1==%2
```

EXIST [<d>:] [<path>] <filename>	This condition is true if the specified file exists. The filename can include wildcards.
----------------------------------------	------------------------------------------------------------------------------------------

Notes

1. **BACKUP, FORMAT, REPLACE and RESTORE** are the only MS-DOS commands that return an **ERRORLEVEL**. This facility is provided for application programs that set error codes. These error codes can be tested by the **ERRORLEVEL** parameter of the **IF** command.

PAUSE

The PAUSE command suspends the execution of a batch file and allows you to display a message (or an instruction to be performed) before resuming execution.

Syntax

PAUSE [<message>]

Operation

The PAUSE command temporarily suspends the execution of a batch file. An optional message of up to 121 characters can be displayed. This is followed by the prompt:

Strike a key when ready...

Pressing **CTRL Break** or **CTRL C** terminates the execution of the batch file. Pressing any other key resumes execution of the batch file.

The PAUSE command is useful for issuing instructions to the batch file user and for allowing sufficient time to execute those instructions. For example, the command line:

PAUSE Insert disc in drive A:

displays the message and prompt:

PAUSE Insert disc in drive A:
Strike a key when ready...

This instructs the user to place a disc in drive A:, and waits until the user has pressed a key to indicate the task is complete.

Notes

1. If you press **CTRL Break** or **CTRL C**, the following message is displayed:

Terminate batch job (Y/N)?

If you answer "Y", you're returned to the MS-DOS prompt. If you answer "N", execution of the batch file resumes. If you enter any other characters, the message is repeated.

2. If ECHO is OFF, the optional message isn't displayed. However, you'll still see the **Strike a key when ready** message.

REM

The REM command displays the message entered after it.

Syntax

REM [<message>]

Operation

The <message> entered with the REM command is displayed when the batch processor reaches that line. The message can contain up to 123 characters. If ECHO is OFF, the message isn't displayed. For example, the command line:

```
REM Copy successfully completed.
```

causes the following message to be displayed:

```
REM Copy successfully completed.
```

when the batch file is executed.

SHIFT

The **SHIFT** command allows access to more than 10 replaceable parameters.

Syntax

SHIFT

Operation

Each time the **SHIFT** command is executed, the values assigned to the replaceable parameters are shifted; %1 becomes %0, %2 becomes %1, and so on. The tenth parameter entered on the command line when the batch file was invoked is assigned to %9.

With careful planning and use of the **SHIFT** command, the limit of 10 replaceable parameters can effectively be removed. Note however, that even with the **SHIFT** command, only 10 parameters can be active at any one time. In addition, there is no way to retrieve parameters which have been "shifted out" (i.e., once a **SHIFT** is executed, the parameter %0 that existed before the shift cannot be recovered). The following example demonstrates the **SHIFT** command. The values assigned to each replaceable parameter are shown below:

%0 = Alpha
%1 = Beta
%2 = Gamma
%3 = Delta

After the SHIFT command, the values are as follows:

%0 = Beta
%1 = Gamma
%2 = Delta

The Alpha is lost, %3 is set to the previous value of %4, and if more than 9 parameters were entered on the command line, %9 is set to the 10th one.



6

System Configuration

In this chapter, we'll examine the various ways MS-DOS can be expanded to suit the different system configurations and requirements of your HP computer.

Overview

MS-DOS is an extremely flexible operating system. It provides support for many different system components and configurations. This flexibility is the result of two different MS-DOS features.

The first feature is the set of MS-DOS configuration commands. These commands allow some of the parameters in MS-DOS to be altered to suit specific system requirements. You can put the configuration commands into a system file called CONFIG.SYS. (To create a CONFIG.SYS file, use the MS-DOS COPY CON command, EDLIN, or any other word processor that creates unformatted files.) Then, each time you start your system, the commands in the CONFIG.SYS file are executed. We'll examine these commands in the next section in this chapter.

The second feature is the ability to add installable device drivers. Installable device drivers allow peripheral devices not already supported by MS-DOS to be added to the system. They also provide enhanced support for certain standard system components. We'll examine device drivers in greater detail later in this chapter.

Configuration Commands

There are nine MS-DOS configuration commands. Each operates in a manner similar to the MS-DOS internal commands. Each command is followed by a set of parameters. The syntax is slightly different in that the command and parameters must be separated by an equal sign (=). With the exception of the **BREAK** command, configuration commands cannot be used outside of the **CONFIG.SYS** file. If you enter a configuration command on the MS-DOS command line, the following error message is displayed:

Bad command or file name

BREAK

The BREAK command enables or disables extended **CTRL Break** and **CTRL C** checking by MS-DOS.

Syntax

BREAK=ON

or

BREAK=OFF

ON this causes MS-DOS to check for **CTRL Break** and **CTRL C** during all MS-DOS operations.

OFF this causes MS-DOS to check for **CTRL Break** and **CTRL C** only during standard input/output, standard printer, and standard auxiliary operations. OFF is the default setting.

Operation

MS-DOS automatically checks for a **CTRL Break** or **CTRL C** keyboard sequence during all standard I/O, printer, and auxiliary operations. If extended **CTRL Break** and **CTRL C** checking are enabled, MS-DOS checks during all MS-DOS operations.

To enable extended checking each time you start your system, enter the following command in the CONFIG.SYS file:

```
BREAK=ON
```

Notes

1. This command is functionally identical to the MS-DOS **BREAK** command.

BUFFERS

The **BUFFERS** command specifies the number of disc I/O buffers.

Syntax

BUFFERS=<n>

n the number of buffers. Valid values are 1 to 99; the default value is 2 or 3 buffers, depending upon your computer.

Operation

To increase the number of buffers to 10, enter the following command in the **CONFIG.SYS** file:

BUFFERS=10

The use of buffers can increase the speed of certain applications. As data is read from the disc, it is stored in a buffer. The buffers are used by MS-DOS in a manner that ensures that the most recently read data is in one of the buffers.

The performance of applications that perform a large number of random reads and writes on a data file can be increased by using additional buffers. As the number of buffers increases, the chances increase that the data requested by the application program is already in one of the buffers. Thus, MS-DOS can retrieve the data from memory instead of disc, which speeds up program performance.

For applications that perform mostly sequential read and write operations, there is little performance gain from an increased number of buffers.

There isn't a precise formula for calculating the optimum number of buffers. You need to try different values to determine the number of buffers that yields the maximum performance. In general, applications that perform large amounts of random data access perform best with between 10 and 25 buffers. Systems with a large number of subdirectories perform best with between 20 and 30 buffers.

Notes

1. Each buffer increases the resident size of MS-DOS by 528 bytes and reduces the amount of memory available to the application program by the same amount.
2. If the number of buffers is too large, it takes more time to search through the buffers than to read the data from disc. As a result, in certain cases, additional buffers actually decrease system performance instead of increase it.

COUNTRY

The COUNTRY command selects the display format for the system time and date, currency symbol, and decimal separator based on the specified country.

Syntax

COUNTRY=<nnn>

nnn the three-digit country code. The default value is 001. The corresponding code for each country is listed below.

Country	Code
Argentina	054
Australia	061
Belgium	032
Canadian-French	002
Denmark	045
Finland	358
France	033
Germany	049
Italy	039
Israel	972

Mexico	052
Middle East	785
Netherlands	031
Norway	047
Portugal	351
Spain	034
Sweden	046
Switzerland	041
United Kingdom	044
United States	001
Venezuela	058

Operation

To select the United Kingdom display format for time, date, decimal separator, and currency symbol, enter the following command in the CONFIG.SYS file:

```
COUNTRY=044
```

Notes

1. The COUNTRY command doesn't translate MS-DOS messages or prompts, or change the video or keyboard character set.
2. If your country isn't listed above, pick the country code that most closely matches your needs.

DEVICE

The DEVICE command allows MS-DOS to load an installable device driver onto the system by specifying the file that contains it.

Syntax

DEVICE=[<d>:][<path><filename>[<parm1...>]

d: the drive that contains the device driver.

path the path to the file that contains the device driver.

filename the name of the file that contains the device driver.

parm1 the first parameter to be passed to the device driver.

Operation

MS-DOS provides built-in device drivers. They support standard input/output, standard printer, and disc operations. MS-DOS also allows the installation of additional device drivers, called installable device drivers. Installable device drivers allow you to support additional devices and reconfigure existing devices. For additional information on device drivers, refer to the section in this chapter entitled *Device Drivers*.

The DEVICE command specifies the file containing an installable device driver. MS-DOS loads the device driver

from this file and incorporates it into the resident portion of MS-DOS.

MS-DOS provides the following three files which contain installable device drivers:

- ANSI.SYS contains a character device driver that allows you to use the ANSI standard terminal escape sequences to extend the screen and keyboard display features of your system.
- DRIVER.SYS contains a block device driver that allows you to access a physical drive by referencing a logical drive.
- VDISK.SYS contains a block device driver that allows you to convert system RAM into a virtual disc.

In addition, you can create your own files which contain installable device drivers. Refer to the *MS-DOS 3.2 Programmer's Reference Manual* for additional information.

Note



The size of the resident portion of MS-DOS is increased by the size of each device driver installed.

ANSI.SYS

ANSI.SYS provides support for extended screen and keyboard display features. To install ANSI.SYS, enter the following command in the CONFIG.SYS file:

```
DEVICE=ANSI.SYS
```

When this device driver is installed, it replaces the built-in CON device driver. As a result, ANSI standard terminal escape sequences are executed by the standard input and output devices. The escape sequences are described in the appendix entitled "Extended Screen and Keyboard Control".

DRIVER.SYS

```
DEVICE=[<d>:][<path>]DRIVER.SYS /D:<ddd>
[/T:<ttt>][/S:<ss>]
[/H:<hh>][C][N][F:<f>]
```

d: the drive that contains DRIVER.SYS.

path the path to DRIVER.SYS.

/D the option that specifies the physical drive number. Valid values are 0 to 255. Flexible and hard disc drives are numbered separately. Flexible disc drive numbers start with 0; hard disc drive numbers start with 128. The most commonly used drive numbers are listed below.

Drive 0 specifies the first internal flexible disc drive.

Drive 1 specifies the second internal flexible disc drive.

Drive 2 specifies the first external flexible disc drive.

Drive 3 specifies the second external flexible disc drive.

Drive 128 specifies the first hard drive.

Drive 129 specifies the second hard drive.

/T the option that specifies the number of tracks per side. Valid values are 1 to 999; the default value is 80 tracks per side.

/S	the option that specifies the number of sectors per track. Valid values are 1 to 99; the default value is 9 sectors per track.
/H	the option that specifies the maximum number of heads. Valid values are 1 to 99; the default value is 2 heads.
/C	the option that indicates changeline support is required.
/N	the option that indicates the physical device is a hard drive (or any other non-removable block device).
/F	the option that specifies the device type. Valid values are 0, 1, 2, and 7; the default value is device type 2. Device type 0 specifies a 160/180 or 320/360 Kb drive. Device type 1 specifies a 1.2 Mb drive. Device type 2 specifies a 720 Kb drive. Device type 7 specifies a 1.44 Mb drive.

DRIVER.SYS provides the ability to access a physical drive by referencing a logical drive. It allows you to copy files to and from the same disc drive. This is useful if you have more than one flexible disc drive on your system and the drive types are different. It also allows you to load support for external disc drives that don't come with their own software drivers.

To install **DRIVER.SYS**, enter the **DEVICE=DRIVER.SYS** command followed by the appropriate parameters in the **CONFIG.SYS** file. For example, assume you have two flexible disc drives and a hard drive on your system. The first flexible disc drive (drive A:) is a 1.2 Mb drive; the second (drive B:) is a 360 Kb drive. Drive C: is a hard drive. To be able to reference the first flexible disc drive as drive A: or drive D:, enter the following command in the **CONFIG.SYS** file:

```
DEVICE=DRIVER.SYS /D:0 /T:80 /S:15 /H:2 /C /F:1
```

Now, you can copy files from drive A: to drive D: by simply switching the flexible discs in the 1.2 Mb drive. For example, if you enter the command line:

```
A>COPY FILE1.TXT D:
```

MS-DOS copies the file FILE1.TXT from drive A: to drive D:. It does this by first prompting you to insert the flexible disc for drive A: and then prompting you to insert the flexible disc for drive D:.

To load support for a 720 Kb drive, if it's your first external flexible disc drive, enter the following command in the CONFIG.SYS file:

```
DEVICE=DRIVER.SYS /D:2 /T:80 /S:9 /H:2 /C /F:2
```

MS-DOS assigns the next available drive letter to the external drive.

To be able to copy to and from this external 720 Kb drive, enter the following two commands in the CONFIG.SYS file:

```
DEVICE=DRIVER.SYS /D:2 /T:80 /S:9 /H:2 /C /F:2  
DEVICE=DRIVER.SYS /D:2 /T:80 /S:9 /H:2 /C /F:2
```

MS-DOS assigns the next two available drive letters to the external drive.

VDISK.SYS

```
DEVICE=[<d>:][<path>]VDISK.SYS [<bbb>]  
[<sss>][<ddd>][</E[:<m>]>]]
```

d: the drive that contains VDISK.SYS.

path the path to VDISK.SYS.

bbb the option that specifies the virtual disc size in Kb. Valid values are 1 to the amount of memory you have in your system; the default value is 64 Kb. VDISK.SYS automatically adjusts the virtual disc size in the following situations:

If the size you specify is less than 1 Kb or greater than the amount of memory in your system, VDISK.SYS uses the default value.

If your system has less than 64 Kb of available memory at the time you're attempting to install VDISK, an error message is displayed and VDISK isn't installed.

sss the option that specifies the sector size in bytes. Valid values are 128, 256, and 512; the default value is 128 bytes. Smaller sector sizes conserve space; larger sector sizes increase performance. Thus, if you're going to use VDISK to store many small files, consider specifying one of the smaller sector sizes.

ddd the option that specifies the number of directory entries that the root directory of the virtual disc can contain. Valid values are 2 to 512; the default value is 64 directory entries.

VDISK.SYS adjusts the number of directory entries upward to the nearest sector size boundary. Thus, if you specify a sector size of 256 and the number of directory entries as 17, 24 directory entries are generated. 24 directory entries at 32 bytes per entry occupy 768 bytes, which is a multiple of the sector size.

If you specify a virtual disc size that's too small to hold the FATs (File Allocation Tables), the root directory, and two additional sectors, VDISK.SYS decreases the directory size until these conditions are met. If the directory size reaches one sector and the conditions still aren't met, an error message is displayed and VDISK isn't installed.

/E the option that instructs VDISK.SYS to create the virtual disc in extended memory (memory at or above 1 Mb), even though the driver code is to be installed in low memory.

This option is valid only for systems equipped with optional extended memory. If you specify this option and your system doesn't have extended memory, an error message is displayed and VDISK isn't installed.

m the option that specifies the maximum number of data sectors (of size <sss>) that VDISK.SYS can transfer to and from extended memory at one time. Valid values are 1 to 8; the default value is 8 data sectors. This option must be used in conjunction with the /E option.

VDISK.SYS allows a portion of your system's RAM memory to be used as a disc drive. RAM memory used to emulate or act like a physical disc drive is referred to as a "RAM Disc" or "Virtual Disc". A virtual disc can be used in the same manner as a physical disc drive, with the exception of a few MS-DOS commands (such as CHKDSK, DISKCOPY, and FORMAT). VDISK.SYS allows either RAM memory or extended memory (at or above 1 Mb) to be used as a virtual disc. In addition, you can install more than one virtual disc at a time.

There is one big advantage and one big disadvantage to using a virtual disc instead of a flexible or hard disc. The advantage is speed. Disc operations on a virtual disc are significantly faster than on a hard disc. The improvement in performance is even more dramatic when compared with a flexible disc. The disadvantage is the inability to permanently keep the contents of a virtual disc. Whenever you restart or turn off your system, the contents of the virtual disc are lost.

Since the contents of a virtual disc are lost when you restart or turn off your system, you need to transfer files to the virtual disc when you start your system. Then, you need to transfer the files back to a flexible or hard disc before you restart or turn off your system. Use the MS-DOS COPY or XCOPY command to transfer files to and from a virtual disc.

Note



We recommend that you only put applications and MS-DOS commands on a virtual disc. Don't put data files on it. Also, you can use the AUTOEXEC.BAT file to load commonly used applications onto a virtual disc every time you start your system.

To install a virtual disc with default parameters, enter the following command line in the CONFIG.SYS file:

```
DEVICE=VDISK.SYS
```

To install more than one virtual disc, insert additional DEVICE=VDISK.SYS commands in the CONFIG.SYS file. Once a virtual disc has been installed, the following sign-on message appears each time you start your system:

```
VDISK Version 3.20  virtual disk x
```

where x is the drive assigned to the virtual disc. Below this message, the values of the three parameters are displayed as shown:

```
Buffer size:      <bbb>
Sector size:      <sss>
Directory entries: <ddd>
```

When a virtual disc is operating in extended memory, interrupt servicing is suspended during data transfers. If frequent interrupts occur during data transfers, some of the interrupts can be lost. If you experience this problem, try installing the virtual disc in non-extended memory.

- If this solves the problem, try installing it in extended memory again, but this time reduce the value of <m> until no interrupts are lost. If an <m> of 1 and an <sss> of 128 doesn't improve the situation, then you can't use the virtual disc in extended memory in that particular environment.
- If this doesn't solve the problem, investigate other areas that might be involved.

FCBS

The FCBS command specifies the number of files that can be concurrently open via FCBs (File Control Blocks).

Syntax

FCBS= <x>,<y>

- x the maximum number of files that can be open at any one time via FCBs. Valid values are 1 to 255; the default value is 4 files. <x> must be greater than or equal to <y>.
- y the number of open files that MS-DOS cannot close automatically if an application program tries to open more than <x> files via FCBs. This is also known as the number of "protected" files because they're protected from being closed. Valid values are 1 to 255; the default value is 0 files.

Operation

MS-DOS provides two ways to open and access files: FCBs and Handles. The FCBS command allows you to specify the number of files that can be open (accessed) by way of FCBs. The FILES command allows you to specify the number of files that can be open by way of Handles. For additional information on FCBs and Handles, refer to the *MS-DOS 3.2 Programmer's Reference Manual*.

To allow 10 files to be open concurrently via FCBs, plus "protect" the first 5 files, enter the following command line in the CONFIG.SYS file:

FCBS=10,5

MS-DOS keeps track of which file opened via an FCB is the least recently used. Then, if an application program attempts to open more than <x> files, the action taken by MS-DOS depends on whether file-sharing is loaded.

If File-Sharing is Loaded

If file-sharing is loaded (the SHARE command has been executed), MS-DOS closes the least recently used file prior to opening the new file. The <y> parameter protects the first <y> files from being closed in this manner. If <y> is set equal to <x>, MS-DOS doesn't close any files when an application program tries to open more than <x> files. Instead, the application program is unable to open the new file.

If File-Sharing isn't Loaded

If file-sharing isn't loaded, there isn't a limit to the number of files that can be open concurrently via FCBs. Thus, no action is taken by MS-DOS provided the SHARE command hasn't been executed.

Notes

1. If an application program uses more than one FCB to refer to the same file, MS-DOS still considers this as only one FCB used.
2. The FCBS command increases the size of the resident portion of MS-DOS.

FILES

The FILES command specifies the number of files that can be concurrently open via Handles.

Syntax

FILES= <n>

n the maximum number of files that can be open at any one time via handles. Valid values are 8 to 255; the default value is 8 files.

Operation

MS-DOS provides two ways to open and access files: Handles and FCBs (File Control Blocks). The FILES command allows you to specify the number of files that can be open by way of handles. The FCBs command allows you to specify the number of files that can be open by way of FCBs. For additional information on handles and FCBs, refer to the *MS-DOS 3.2 Programmer's Reference Manual*.

To allow 10 files to be concurrently open via handles, enter the following command line in the CONFIG.SYS file:

```
FILES=10
```

Notes

1. Each handle over 8 increases the resident size of MS-DOS by 48 bytes.

2. There isn't a limit to the number of files an application program can have open when those files are opened through FCBs instead of handles, unless the SHARE command has been executed. Then, the maximum is set by the FCBS command.
3. <n> is the total number of handles open for the entire system. Each process is limited to a maximum of 20 files open at any one time. This includes the 4 files MS-DOS opens for the standard input/output, standard error, standard printer, and standard auxiliary devices.

LASTDRIVE

The LASTDRIVE command specifies the last valid drive. This sets the maximum number of drives that can be accessed on the system.

Syntax

LASTDRIVE=<d>

d the last valid drive. Valid values are A to Z; the default value is drive E (5 drives).

Operation

To specify drive G: as the last valid drive on your system, enter the following command line in the CONFIG.SYS file:

LASTDRIVE=G

This tells MS-DOS that you have 7 drives on your system, A: through G:.

Each block device driver, disc drive, and subdirectory assigned via the SUBST command requires a drive designator. If the total of these exceeds 5, you need to put a LASTDRIVE command in the CONFIG.SYS file to extend the last valid drive.

If the last valid drive isn't high enough to accommodate the block device drivers and disc drives, MS-DOS ignores the specified value and determines the last valid drive for you. However, when determining the value of the last valid drive, MS-DOS doesn't take into consideration any subdirectories assigned via the SUBST command.

SHELL

The SHELL command allows you to specify an alternate command processor.

Syntax

SHELL=[<d>:][<path><filename>[<parm1>...]

d: the drive that contains the alternate command processor.

path the path to the file that contains the alternate command processor.

filename the name of the file that contains the alternate command processor.

parm1 the first parameter to be passed to the alternate command processor.

Operation

The SHELL command substitutes an alternate command processor for COMMAND.COM. For example, PAM is an alternate command processor. To substitute PAM for COMMAND.COM, include the following command line in the CONFIG.SYS file:

SHELL=PAMCODE.COM ROOT

If this command line isn't included, MS-DOS starts the default command interpreter, COMMAND.COM.

Notes

1. The word "ROOT" in the example is a parameter passed to the PAM command processor.
2. If you're a system programmer and you're developing your own top-level command processor, remember to include provisions for handling interrupts 22H, 23H, and 24H, and for reading and executing commands. The batch processor and internal commands reside in COMMAND.COM. As a result, they must be duplicated in your command processor or they won't be available to the user.
3. COMSPEC= isn't affected by the SHELL command. To make sure that the same command processor is used for reloading (the transient portion only), use the SET command with COMSPEC= to point to that command processor.

STACKS

The STACKS command allows you to override the default number of stack frames and the default size of each stack frame.

Syntax

STACKS=n,s

- n the number of stack frames. Valid values are 8 to 64; the default value is 9 stack frames.
- s the size (in bytes) of each stack frame. Valid values are 32 to 512; the default value is 128 bytes.

Operation

Each time a hardware interrupt occurs, MS-DOS uses a stack frame from the stack pool. After MS-DOS finishes processing the interrupt, it returns the stack frame to the stack pool. If a number of hardware interrupts occur in rapid succession, MS-DOS may need to use more stack frames than it has available. The STACKS command allows you to increase the number and size of stack frames, and thus resolve this resource problem.

For example, to increase the number of stack frames to 20, include the following command line in the CONFIG.SYS file:

STACKS=20,128

Caution

DO NOT decrease the number of stack frames or the size of each stack frame below the default value. Doing so could cause a system failure or unexpected results.

Notes

1. When you increase stack resources, you decrease the amount of available memory. For this reason, we recommend that you increase the number of stacks first, without increasing the size of each stack frame. Then, if that doesn't solve the problem, increase the size of each stack frame.

Device Drivers

Installable device drivers allow additional peripheral devices to be added to your system. As we mentioned earlier in this chapter, installable device drivers provide the second major method for expanding MS-DOS. Before we discuss installable device drivers in detail, let's review some of the characteristics of device drivers in general.

There are two types of device drivers: character device drivers and block device drivers. Character device drivers provide support for character devices, such as keyboards, display screens, modems, printers and plotters. They're identified by device driver names, such as CON and PRN.

Block device drivers provide support for block devices, such as disc drives and devices which emulate disc drives (virtual discs). Unlike character device drivers, block device drivers don't have names. Instead, they're identified by drive designators, such as A: and B:.

MS-DOS provides built-in device drivers. They support standard input/output, standard printer and disc operations. In addition, MS-DOS provides the following three files which contain installable device drivers:

- ANSI.SYS (a character device driver)
- DRIVER.SYS (a block device driver)
- VDISK.SYS (a block device driver)

For additional information on these three files, refer to the DEVICE command in this chapter.

Finally, you can create your own files which contain installable device drivers. For details on the structure of installable device drivers, refer to the *MS-DOS 3.2 Programmer's Reference Manual*.

Installable Device Drivers

MS-DOS uses the DEVICE command to add installable device drivers, both character and block, to your system. We've already discussed the syntax of the DEVICE command in this chapter. Now we'll look a little deeper into *how* MS-DOS uses the DEVICE command to add installable device drivers.

MS-DOS uses two different methods to add installable device drivers, depending upon whether they're character or block device drivers.

Installable Character Device Drivers

MS-DOS provides the built-in character device drivers listed in table 6-1 below.

Table 6-1. Built-in Character Device Drivers

Name	Description
AUX	Auxiliary device (same as COM1).
CLOCK\$	A special purpose device used to read or set the system time and date. Unlike the other devices, CLOCK\$ can't be used in an MS-DOS command.
COM1	Serial port 1.
COM2	Serial port 2.
COM3	Serial port 3.

Table 6-1. Built-in Character Device Drivers (Cont.)

Name	Description
COM4	Serial port 4.
CON	Console. This device represents the keyboard for input and the display screen for output.
LPT1	Parallel port 1. This device supports output only.
LPT2	Parallel port 2. This device supports output only.
LPT3	Parallel port 3. This device supports output only.
NUL	Dummy device. When used as an input device, it immediately returns an end-of-file condition. When used as an output device, it accepts data, but immediately discards it (no data is written).
PRN	System printer (same as LPT1).

When you start your system, MS-DOS reads the CONFIG.SYS file. When it finds a DEVICE command specifying an installable *character* device driver, MS-DOS compares the driver's name with the names of the built-in character device drivers. If a match is found, MS-DOS loads the installable driver in place of the built-in driver with the same name. This feature allows you to replace the built-in character device drivers with installable character device drivers.

For example, if the following command line:

```
DEVICE=MOUSE.SYS
```

is added to the CONFIG.SYS file, and MOUSE.SYS is a file that contains an installable character device driver called COM1, then it is loaded in place of the built-in driver COM1 as serial port 1. The single exception is the NUL device driver; it can't be replaced.

If a match isn't found, MS-DOS loads the installable character device driver as an extension to the system. Its name is added to the list of character device drivers. In addition, its name becomes part of the list of reserved MS-DOS filenames.

Installable Block Device Drivers

MS-DOS provides the following built-in block device drivers for the internal disc drives in your system:

- A: is your first internal flexible disc drive.
- B: is your second internal flexible disc drive. Drive B: is automatically used, even if your system has only one internal flexible disc drive. In this case, logical drives A: and B: are both associated with your single internal flexible disc drive.
- C: is your first internal hard disc drive, if your system has a hard disc drive.
- D: is your second internal hard disc drive, if your system has two hard disc drives.

As mentioned, when you start your system, MS-DOS reads the CONFIG.SYS file. If it finds a DEVICE command specifying an installable *block* device driver, MS-DOS assigns it the next available drive designator on the system.

For example, if your system has two internal flexible disc drives (A: and B:) and an internal hard disc drive (C:), and MS-DOS finds a **DEVICE** command specifying an installable block device driver, it assigns drive designator D: to that driver.

If more than one installable block device driver is specified in the **CONFIG.SYS** file, MS-DOS assigns drive designators to those drivers in the order they appear in the file.

Also, a single block device driver can contain support for more than one logically contiguous block device. For example, if a single block device driver contains support for four block devices, they are assigned the next four available drive designators.

Finally, unlike installable character device drivers, installable block device drivers can't replace the built-in block device drivers.



7

Redirecting Input and Output

In this chapter, we'll discuss redirecting input and output, as well as piping and filters. Throughout this chapter, we'll refer to the standard input and output devices. Therefore, to start, let's review a few of the important characteristics of these devices.

Standard Input and Output Devices

Input always comes from the standard input device and output always goes to the standard output device. When you start your computer, the keyboard is assigned as the default standard input device and the display screen is assigned as the default standard output device. The keyboard and display screen together are referred to as the Console or CON device.

MS-DOS allows you to change the standard input and output devices from the default to other physical devices, or even files. There are a number of ways to change the standard input and output devices. In this chapter, we'll discuss changing them via redirection.

Redirecting Input and Output

It is often advantageous to receive input from a file or a physical device other than the keyboard, or to send output to a file or a physical device other than the display screen. This process is called "redirection".

MS-DOS allows you to redirect input so that it doesn't come from the default standard input device; it comes from a file or another physical device instead. The same is true of output. MS-DOS allows you to redirect output so that it doesn't go to the default standard output device; it goes to a file or another physical device instead. In both cases, the file (or physical device) becomes the new standard input or output device, temporarily replacing the default standard input or output device.

Redirection has the following important characteristics:

- Input and Output are redirected on the MS-DOS command line.
- Input and Output are redirected independently of each other.
- Redirection is temporary. It's only in effect while an MS-DOS command or application program is executing.

Redirecting input and output has many uses. You can put an application program's keyboard input into a file and then redirect the input so that it comes from that file instead of the keyboard. Thus, the file acts somewhat like a batch file. Also, you can redirect output so that it goes to a printer or a file instead of the display screen. If output goes to a printer, you get a hard copy; if output goes to a file, you get a permanent copy that you can use in the future.

Redirecting Input

Input can be redirected so that it comes from a file or a physical device instead of the keyboard. The syntax for redirecting input is shown below.

```
<[<d>:][<path>]<filename>
```

```
<<devicename>
```

If input comes from a file, MS-DOS reads characters from the specified file as they are requested by an MS-DOS command (or an application program) until the command finishes executing. When using this form of redirection, it is important that all input required by the command is contained in the file. If the command attempts to read more input than is contained in the file, it stops executing, and you need to exit by holding down **CTRL** and pressing **Break**.

If input comes from a physical device, MS-DOS reads characters from the specified device. Input can be redirected from any character device capable of input (e.g., AUX, COM1, etc.).

The following two examples redirect input so that it comes from a file and a physical device instead of the default standard input device (i.e., the keyboard). In the first example, the input to the MS-DOS SORT command comes from a file called CLIENT.LST. In the second example, the input comes from the AUX device.

```
SORT <CLIENT.LST
```

```
SORT <AUX
```

In both examples, the default standard output device (i.e., the display screen) remains unchanged. As a result, the sorted output appears on the display screen.

Redirecting Output

Output can be redirected so that it goes to a file or a physical device instead of the display screen. The syntax for redirecting output is shown below.

```
>[<d>:][<path>]<filename>
```

```
><devicename>
```

If output is redirected to a file, MS-DOS creates the specified file and stores all of the characters sent to the standard output device in that file. If the file already exists on the disc, MS-DOS overwrites its original contents.

If output is redirected to a physical device, MS-DOS sends characters to the specified device. Output can be redirected to any character device capable of output (e.g., PRN, AUX, LPT1, COM1, etc.).

The following two examples redirect output so that it goes to a file and physical device instead of the default standard output device (i.e., the display screen). In the first example, the output from the DIR command goes to a file called DISCDIR.LST. In the second example, the output goes to PRN, the system printer.

```
DIR >DISCDIR.LST
```

```
DIR >PRN
```

In both examples, the default standard input device (i.e., the keyboard) remains unchanged. As a result, the input comes from the keyboard.

To redirect output to an existing file and *not* overwrite its original contents, use the syntax shown below.

```
>>[<d>:][<path>]<filename>
```

The double angle brackets (>>) allow you to append data to the end of an existing file, rather than overwrite it. For example, the following command line:

```
DIR >>DISCDIR.LST
```

appends the output from the MS-DOS DIR command to the contents (if any) of the file DISCDIR.LST.

Piping Input and Output

Piping is an extension of redirection. Piping allows two or more MS-DOS commands (or application programs) to be "chained" together. That is, the output from the first command serves as the input to the next.

Piping occurs when MS-DOS encounters two or more commands (or programs) separated by the vertical bar (|) character on the command line. For example, the following command line pipes the output from the DIR command to the SORT command. The output from the SORT command, in turn, is piped to the MORE command. As a result, the sorted directory listing appears on the display screen, one screen at a time.

```
DIR | SORT | MORE
```

When piping is used, MS-DOS executes the commands in the order they're encountered on the command line. In addition, MS-DOS creates a temporary file to store the output from each command except the last command. It assigns the filename extension .\$\$\$ to each temporary file. When piping is complete, MS-DOS deletes the temporary files.

Thus, in the above example, MS-DOS creates a temporary file to store the output from the **DIR** command. This temporary file is used as input to the **SORT** command. Then, MS-DOS creates another temporary file to store the output from the **SORT** command. This second temporary file is used as input to the **MORE** command. Finally, MS-DOS deletes both temporary files. Piping and the use of temporary files is illustrated in Figure 7-1 below.

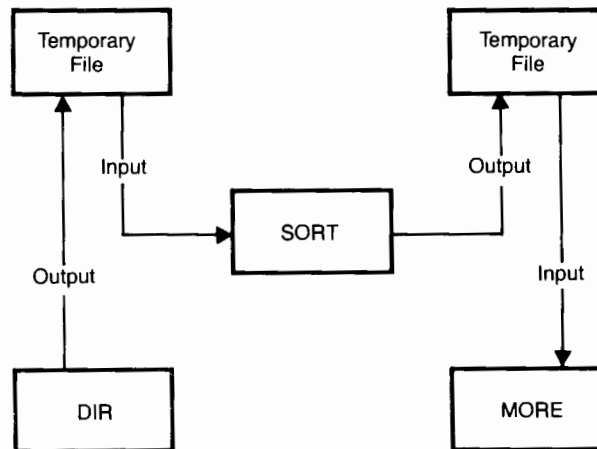


Figure 7-1. Piping and Temporary Files

Using Filters

The final topic relating to input and output is filters. A filter is any program that reads data from the standard input device, modifies or examines it, and then sends it to the standard output device.

MS-DOS provides three filters: **FIND**, **MORE**, and **SORT**.

- **FIND** searches input for a specified string of characters
- **MORE** displays output one screen at a time
- **SORT** sorts input in ascending or descending order and displays the sorted output

Filters are used extensively with piping, as we've already seen in the previous section. For additional information on each filter, refer to the chapter entitled "MS-DOS Command Descriptions".

The following command line pipes the output from the **DIR** command through the **MORE** filter. The result is the same as using the **/P** option with the **DIR** command; the directory listing pauses after each screenful of data is displayed until you press any key.

```
DIR | MORE
```

Tips on Using Redirection, Piping, and Filters

There are several points to remember when redirecting input and output, piping input and output, and using filters.

- These three capabilities won't work with every application program. To work, a program must be written to read from the standard input device and write to the standard output device. Therefore, any program that bypasses MS-DOS for character input and output (i.e., using BIOS interrupts or going to the hardware, itself) won't be able to perform these functions.
- In most cases, piping should only be used with filters. A filter is a specially written application program that uses input and output, interprets an end-of-file marker from the standard input device as a "terminate program" command, and is generally programmed to perform correctly as a filter. The use of piping with an application program that doesn't meet this criteria can lead to unpredictable or unsatisfactory results.
- The temporary files created by piping are stored in the current directory on the active drive. Therefore, to be able to use piping, the disc in the active drive can't be write-protected.
- MS-DOS allows the use of redirection, piping and filters on the same command line. The following command line pipes the output from the DIR command through the SORT filter. Then, the sorted output is redirected to the file DISCDIR.LST.

```
DIR | SORT >DISCDIR.LST
```

It is also possible to redirect output to a device. The following command line redirects the sorted directory listing to the system printer instead of the file DISCDIR.LST.

```
DIR | SORT >PRN
```

- MS-DOS sends some error messages to a logical device referred to as the Standard Error Device. Output to this device (i.e., error messages) can't be redirected. Using the previous example, if a disc read error occurs during the execution of this command line, the error message appears on the display screen instead of the system printer.



8

EDLIN

Introduction to EDLIN

EDLIN, the MS-DOS line editor, can be used to create, change, and display source files and text files. Source files are files containing program listings; text files are files containing data in legible format, such as an unformatted word processing file. Text files are used as examples in this chapter.

In addition to allowing you to create new files and save them on a disc, EDLIN allows you to:

- Update existing files and save both the original and updated versions.
- Delete, edit, insert, and display lines in a file.
- Search for, delete, or replace text within one or more lines in a file.

The files you create or edit using EDLIN are divided into lines; each line can be up to 253 characters in length. Line numbers are generated and displayed during the editing process, but are not present in the saved file.

Lines are always numbered consecutively in EDLIN files, regardless of the editing functions performed. For example, when you insert lines, all line numbers following inserted text are incremented by the number of lines inserted; when you delete lines in a file, the line numbers following the deleted text are resequenced.

How To Start EDLIN

To start EDLIN, enter:

```
EDLIN [path] <filename> [/B]
```

path	the path to the file you want to create or edit.
filename	the name of the file you want to create or edit.
/B	the option that instructs EDLIN to ignore any embedded end-of-file marker (CTRL Z) in your file and process the entire file. This is useful for editing a file which is known to contain an embedded end-of-file marker.

When you access EDLIN, you see the following display:

```
C>EDLIN EXAMPLE
New file
*
```

If EDLIN doesn't find the specified file on the active drive or the specified drive, it creates a new file using the filename you entered. EDLIN responds with the words **New file** to indicate that a new file is being created. The asterisk (*) is the EDLIN prompt.

In the example above, only a filename is specified. Therefore, after data is entered into the file and the EDLIN session is ended, the file is saved in the current directory on the disc in the active drive.

If you specify an existing file, EDLIN attempts to load it into memory. EDLIN loads lines until memory is 75% full to allow room to add lines during editing, and then displays the * prompt. If your file is too large to load, you have to save some of the edited lines to disc during the editing process in order to free memory so you can load the remaining lines for editing. The W (Write) and A (Append) commands explain the procedures for doing this. To begin entering text, enter the I (Insert) command to insert lines into EXAMPLE.

If you do not specify a filename extension when you specify a new file, one is not assigned by EDLIN. However, when you edit an existing file, EDLIN assigns the .BAK extension to your original file (the file on disc); your edited version (the file in memory) has the filename specified when you accessed EDLIN. In this way, EDLIN automatically keeps a backup copy of each file you create. The .BAK file cannot be edited. If you want to edit it, you must first rename it with a different extension, then start EDLIN.

When you have completed your editing session, you leave EDLIN and save your file to disc using the END command.

Caution



When you create or edit an EDLIN file, be certain there is enough space on the specified disc to save the file; if there is not enough space, you risk losing part of your data when you attempt to save it to disc.

Information Common To All EDLIN Commands

EDLIN commands perform editing functions on lines of text. The following information is common to all EDLIN commands, and you should be familiar with it before you create or edit an EDLIN file:

- Paths can be specified on the EDLIN command line. For example, typing:

```
EDLIN USER\USER#1\FILE1.TXT
```

allows you to edit the FILE1.TXT file in the subdirectory USER#1.

- With the exception of the Edit command, all commands consist of a single letter.
- With the exception of the E (End) and Q (Quit) commands, commands can be preceded and/or followed by parameters.
- Commands and string parameters can be entered in either upper- or lower-case letters, or a combination of both.
- Commands and parameters can be separated by delimiters for readability; however, a delimiter is only required between two adjacent line numbers. Delimiters are spaces or commas.

For example, to delete line 6, the command 6D is the same as the command 6 D.

- You can refer to line numbers relative to the current line in your EDLIN file. Use a minus sign (-) followed by a number to indicate a line preceding the current line. Use a plus sign (+) followed by a number to indicate a line following the current line. For example:

-10,+10L

displays 10 lines preceding the current line, the current line, and 10 lines following the current line. Note the use of the comma between adjacent line numbers.

- You can enter multiple commands on one line using an appropriate separator. When you enter the Edit command (<line>), subsequent commands must be separated from it by a semicolon. For all other commands, one command can follow another separated by either a comma or a space. In the case of the S (Search) and R (Replace) commands, however, <string> must be ended by **CTRL** Z instead of **Enter**.

Examples

The following command allows you to edit line 15, then list lines 10 through 20:

15;-5,+5L

In the next example, the command instructs EDLIN to search for "This string", then list the preceding five lines and the following five lines containing the matched string. If no match is found for the search string, the lines listed are those lines relative to the current line.

S This string **CTRL** Z -5,+5L

You can insert a control character (such as `CTRL C`) into text by preceding it with `CTRL V`, then typing the letter (such as C) without `CTRL`. `CTRL V` tells MS-DOS to recognize the next letter as the corresponding control character.

You can also use a control character in any of the string arguments for the S (Search) and R (Replace) commands using this method. For example:

S	<code>CTRL V Z</code>	searches for the first occurrence of <code>CTRL Z</code> in a file.
R	<code>CTRL V Z CTRL Z eof</code>	replaces all occurrences of <code>CTRL Z</code> with "eof".
S	<code>CTRL V C CTRL Z</code> bar	replaces all occurrences of <code>CTRL C</code> with "bar".
	<code>CTRL V V</code>	inserts <code>CTRL V</code> into the text.

The `CTRL Z` character is read as an end-of-file marker by EDLIN. If a `CTRL Z` appears in your file, you can use the /B switch to tell EDLIN to ignore it and show you the entire file.

The current line is the last line edited or the line currently being edited. EDLIN marks the current line with an asterisk (*). For example:

```
1: This is the first line of my new file.  
2: *
```

The MS-DOS Editing keys (explained in the appendix entitled "The MS-DOS Keyboard") function identically in EDLIN.

As with MS-DOS commands, EDLIN commands become effective only after you press **Enter**.

EDLIN Syntax

The following syntax notation is used in this chapter.

- < > Parameters enclosed by angle brackets represent data you must enter. When the brackets enclose lower-case text, for example <line>, you must supply the entry defined by the text.
- [] Information enclosed in square brackets is optional.
- CAPS Letters and words in capital letters are commands or portions of statements that must be entered exactly as shown.

Also, in the examples, the underscore (__) represents the cursor.

EDLIN Command Parameters

Some EDLIN commands allow the use of parameters. If parameters are allowed, they should be entered in the following format:

[<parameter>] <COMMAND> [<parameter>]

The most common EDLIN parameters are:

- <line> Indicates you must specify a line number. You can specify a line number in one of three ways:
- By entering a decimal integer from 1 to 65529. If you specify a number greater than the number of lines in the file, the line number is equal to the number after the last line number in the file.
 - By entering a pound sign (#) to specify the line after the last line in memory. This has the same effect as specifying a number greater than the number of lines in memory.
 - By entering a period (.) to specify the current line. The current line indicates the location of the last change to the file, but it is not necessarily the last line displayed. The current line is marked by an asterisk (*) between the line number and the first character of text in the line. For example:

10:*FIRST character of text

- <n>** Indicates you must specify the number of lines. This parameter is used only with the **W** (Write) and **A** (Append) commands, which are used only if the file to be edited is too large to fit in memory. Thus, when you use the **<n>** parameter, you enter the number of lines you want to load from or write to disc.
- <string>** Indicates you must enter one or more characters that represent text to be searched for, replaced, or deleted. This parameter is used with the **S** (Search) and **R** (Replace) commands. Each **<string>** must be terminated by **CTRL** Z or **Enter** (see the **Replace** command for details). No spaces should be left between a string and its command letter, unless you want those spaces to be part of the string.

EDLIN Commands

These EDLIN commands are described in detail on the following pages:

Command	Purpose
<line>	Edits the specified line
A	Appends lines to your file
C	Copies lines from one location to another location in your file
D	Deletes lines
E	Ends your editing session and writes your file to disc
I	Inserts lines into your file
L	Lists lines
M	Moves lines from one location to another
P	Lists your file one page (23 lines) at a time
Q	Quits editing without saving your file
R	Replaces lines in your file
S	Searches for specified strings
T	Transfers text from a file on disc to the current file
W	Writes a specified number of lines to disc

A (Append)

The Append command adds the specified number of lines from a disc file to the end of the file currently being edited (i.e., the file currently in memory).

Syntax

[<n>]A

n the number of lines you want to append from your disc file to the file currently being edited.

Operation

When you start EDLIN and specify an existing file, EDLIN attempts to load it into memory; EDLIN loads lines until memory is 75% full to allow room to add lines during editing.

If your file is too large to load, you must edit all or a portion of the lines in memory, and save the edited lines to disc in order to free memory for more lines. Then, you can load unedited lines from disc into memory using the Append command. (Refer to the Write command for information on how to save edited lines to disc.)

If you do not specify the number of lines to append, lines are appended to the file in memory until available memory is 75% full; no action is taken if available memory is already 75% full.

When the Append command has read the last line of a file into memory, the following message is displayed:

End of input file

Example

To load 23 lines from the specified disc file into memory, enter the command:

23A

C (Copy)

The Copy command copies a range of lines to a specified line number location in your file. If you use the <count> option, the Copy command can be repeated a specified number of times.

Syntax

[<fline>],[<lline>],<dline>,<count>]C

fline	the first line in the range of lines to be copied.
lline	the last line in the range of lines to be copied.
dline	the line number location (destination) where lines are to be copied; data is placed before this line. This parameter is required; you must specify a line number location when using the Copy command.
count	the number of times the specified range is to be copied. The count must be an unsigned integer.

Operation

If the <count> parameter is not specified, lines are copied once.

If either <fline> or <lline> is omitted, Copy uses the current line as the default value. This results in the current line being copied to the destination (<dline>).

Your file is renumbered automatically after the copy process is complete, and the first of the copied lines

becomes the current line. For example:

1,5,8C

copies lines 1 through 5 to line 8 and line 8 becomes the current line.

If line numbers overlap, the following error message is displayed:

Entry error

You must retype your copy command to recover. For example, the command:

3,20,15C

is in error because it instructs EDLIN to copy lines 3 through 20 and insert them before line 15.

When the copy process is complete, EDLIN displays the * prompt.

Examples

Assume the following file is in memory and ready to edit:

```
1: Creating, changing and displaying files
2: are important tasks for computers.
3: This is why EDLIN, the MS-DOS line
4: editor, is available with every
5: *computer that uses MS-DOS.
```

You can copy this entire block of text by entering:

1,5,6C

The result is:

```
1: Creating, changing and displaying files
2: are important tasks for computers.
3: This is why EDLIN, the MS-DOS line
4: editor, is available with every
5: computer that uses MS-DOS.
6:*Creating, changing and displaying files
7: are important tasks for computers.
8: This is why EDLIN, the MS-DOS line
9: editor, is available with every
10: computer that uses MS-DOS.
```

To copy lines and place them within this text, you must specify <dline> to tell EDLIN where to put them. If you want to copy lines 2 through 5 and place them before line 7, enter:

```
2,5,7C
```

The resulting file looks like this:

```
1: Creating, changing and displaying files
2: are important tasks for computers.
3: This is why EDLIN, the MS-DOS line
4: editor, is available with every
5: computer that uses MS-DOS.
6: Creating, changing and displaying files
7:*are important tasks for computers.
8: This is why EDLIN, the MS-DOS line
9: editor, is available with every
10: computer that uses MS-DOS.
11: are important tasks for computers.
12: This is why EDLIN, the MS-DOS line
13: editor, is available with every
14: computer that uses MS-DOS.
```

Notice, in both examples, that the lines have been renumbered by EDLIN.

D (Delete)

The Delete command deletes a specified range of lines in a file.

Syntax

[<fline>][,<lline>]D

fline the first line in the range of lines to be deleted.

lline the last line in the range of lines to be deleted.

Operation

EDLIN supplies default values for the first line and the last line if either one or both are omitted. If you omit the first line, as in:

,<lline>D

the delete process begins with the current line and ends with the line specified by <lline>. Note that the comma is required to indicate the missing first line parameter.

If you omit the last line parameter, as in:

<fline>D

or

<fline>,D

only the specified line is deleted.

If you omit both parameters, as in:

D

only the current line is deleted, and the line following the deleted line becomes the current line. The line following the deleted range becomes the current line even if the deleted range includes the last line in memory. The current line, and any following lines, are renumbered.

When the delete process is complete, EDLIN displays the * prompt.

Examples

Assume that the following file exists and is ready to edit:

```
1: This sample has been written
2: to demonstrate dynamic line number
3: generation. See what happens when
4: you use Delete and Insert
.
.
.
25: (the D and I commands)
26: to edit the text
27:*in your file.
```

To delete multiple lines, enter:

5,24D

The result is:

```
1: This sample file has been written
2: to demonstrate dynamic line number
3: generation. See what happens when
4: you use Delete and Insert
5: *(the D and I commands)
6: to edit the text
7: in your file.
```

To delete a single line, enter:

```
6D
```

The result is:

```
1: This sample file has been written
2: to demonstrate dynamic line number
3: generation. See what happens when
4: you use Delete and Insert
5: (the D and I commands)
6: *in your file.
```

To delete a range of lines from the following file:

```
1: This sample file has been written
2: to demonstrate dynamic line number
3: *generation. See what happens when
4: you use Delete and Insert
5: (the D and I commands)
6: in your file.
```

enter the following EDLIN command:

,6D

The result is:

```
1: This sample file has been written
2: to demonstrate dynamic line number
3:*in your file.
```

Notice that the lines are automatically renumbered.

Edit

The Edit command allows you to edit (change) one line of text.

Syntax

[<line>]

line# the line number you want to edit. It can be expressed as a decimal integer, a pound sign (#), a period (.), or **Enter**.

Operation

Any of the entries for <line> described in the section *EDLIN Command Parameters* can be used depending on the line you want to edit.

After you specify a line number and press **Enter**, EDLIN displays the line. In addition, below the line, EDLIN displays the line number. If you press **Enter** without specifying a line number, EDLIN assumes you want to edit the line after the current line,

To replace the displayed line with a new line, simply type the new line and enter it into your file by pressing **Enter**. If you want to edit the displayed line without retyping it, use any of the MS-DOS editing keys described in the appendix entitled "Using the MS-DOS Keyboard". The displayed line serves as the template until the **Enter** key is pressed.

If you decide not to edit the displayed line, position the cursor under the first character or past the last character on the line and press **Enter**. Use the cursor control keys to move the cursor.

Caution



If you position the cursor by entering characters, including spaces entered by pressing the space bar, you will lose all or part of the data on the displayed line. Be sure to use the cursor control keys to position the cursor.

Example

Assume the following file exists and is ready to edit:

```
1: This is a sample file created
2: to demonstrate what happens
3: when you edit line
4: *four.
```

To edit the fourth line, enter the line number and press **Enter**. The following display appears:

```
*4
4: *four.
4: *-
```



The underscore in this example represents the cursor on your screen.

To edit the line using the MS-DOS editing keys, enter:

Function	Edit Keys	EDLIN Response
Enter Insert mode	Ins	
Type in the data	number _	4: number _
Copy All	F3	4: number four.
Enter	Enter	*

Now, enter **L** to list the file. The result is:

```
1: This is a sample file created
2: to demonstrate what happens
3: when you edit line
4:*number four.
```

You can either continue to add lines starting with line 5, or leave the edit mode by holding down **CTRL** and then pressing **Break**.

E (End)

The End command ends the editing session, saves the edited file on disc, and returns you to the MS-DOS prompt.

Syntax

E

Operation

The End command saves the edited file on disc, renames the original input file by adding the .BAK extension, and exits EDLIN. If the input file was created during the editing session, no .BAK file is created.

The End command doesn't have any parameters. As a result, you can't tell EDLIN where to save your file (i.e., you can't specify the drive and directory). Your file is saved in the current directory of the active drive unless you specified a drive designator and path when you accessed the file via EDLIN.

Caution



Make sure your disc contains enough free space to save your entire file. If there is not enough space available, the save operation will be aborted and all or part of your file will be lost. In this case, your original file will not be renamed with a filename extension of .BAK, and the portion of data that was saved to disc, if any, will have a filename extension of .\$\$\$.

After the End command has been executed, you are returned to the MS-DOS prompt.

I (Insert)

The Insert command inserts lines of text immediately before the specified line. When you create a new file, the Insert command allows you to begin writing (inserting) lines.

Syntax

[<line>] I

line# the line number before which you want to insert lines.

Operation

When you enter a <line> followed by the Insert command, lines are inserted immediately before the line number specified. Entering the Insert command without specifying <line>, or by specifying <line> as a period (.), causes lines to be inserted immediately before the current line. Successive line numbers appear automatically each time a line is entered by pressing **Enter**.

When the insert process is complete, the line immediately following the inserted lines becomes the current line, and all line numbers following the inserted lines are incremented by the number of lines inserted.

If you specify a line number greater than the last line in your file, or if you specify the pound sign (#) as the line number, lines are inserted after the last line in memory. In this case, the last line inserted becomes the current line.

Hold down **CTRL** and then press **Break** to leave Insert mode and return to the EDLIN prompt.

Examples

Assume the following file exists and is ready to edit using the Insert command:

```
1: This sample file demonstrates
2: dynamic line number generation.
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.
```

In this example, the underscore represents the cursor. To insert text before a specific line that is not the current line, enter:

```
*4I
4:* _
```

and insert two new lines of text:

```
4:*FIRST NEW LINE OF TEXT
5:*SECOND NEW LINE OF TEXT
6*
```

and press **CTRL** **Break**. Now enter L to list the file. The result is:

```
1: This sample file demonstrates
2: dynamic line number generation.
3: See what happens when you use
4: FIRST NEW LINE OF TEXT
5: SECOND NEW LINE OF TEXT
6:*Delete and Insert
```

```
7: (the D and I commands)
8: to edit text
9: in your file.
```

If the two inserted lines had been placed at the beginning of the file, the file would look like this:

```
1: FIRST NEW LINE OF TEXT
2: SECOND NEW LINE OF TEXT
3:*This sample file demonstrates
4: dynamic line number generation.
5: See what happens when you use
6: Delete and Insert
7: (the D and I commands)
8: to edit text
9: in your file.
```

If the two inserted lines had been placed at the end of the file (#I), the file would look like this:

```
1: This sample file demonstrates
2: dynamic line number generation.
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: FIRST NEW LINE OF TEXT
9:*SECOND NEW LINE OF TEXT
```

Note the placement of the * to mark the current line in all of the above examples.

L (List)

The List command displays a specified range of lines; the current line remains unchanged.

Syntax

`[<fline>][,<lline>]L`

`fline` the first line in the range to be listed.

`lline` the last line in the range to be listed.

Operation

If you enter the List command with both the first line parameter (<fline>) and the last line parameter ((<lline>) specified, the specified range is listed. Default values are supplied by EDLIN if either one or both of the parameters are omitted.

If you omit both parameters, 23 lines are displayed -- 11 lines before the current line, the current line, and then 11 lines following the current line. If there are fewer than 11 lines before the current line, additional lines following the current line are displayed to make a total of 23 lines.

If you omit the first line parameter and specify the last line parameter, as in:

`,<lline>L`

the listing begins 11 lines before the current line and ends with the specified last line. (Note that the beginning comma is required to indicate the omitted first parameter.) If the specified last line is more than 11 lines before the current line, the listing is the same as if you had omitted

both parameters.

If you omit the last line parameter, as in:

```
<fline>L
```

23 lines are displayed, starting with the specified <fline>.

Examples

Assume the following file exists and is ready to list:

```
1: This sample file demonstrates
2: dynamic line number generation.
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 a range of lines without reference to the current line, enter:

2,5L

The result is:

```
2: dynamic line number generation.  
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, enter:

.,26L

The result is:

```
15:*The current line contains an asterisk.  
.  
      (11 lines)  
.  
26: to edit text
```

To list a range of 23 lines centered around the current line (line 15 in our example), enter:

L

The result is:

```
4: Delete and Insert
5: (the D and I commands)
.
.
.
13: The current line is listed in the middle
14: The current line remains unchanged.
15:*The current line contains an asterisk.
.
.
.
26: to edit text.
```

M (Move)

The Move command moves one line or a range of lines to the specified line.

Syntax

[<fline>],[<lline>]<,dline>M

fline	the first line in the range of lines to be moved.
lline	the last line in the range of lines to be moved.
dline	the line number location (destination) of the first line in the range of lines to be moved. This parameter is required; you must specify a line number location when using the Move command.

Operation

If the first line parameter is omitted, Move uses the current line as the default value. The same is true if the last line parameter is omitted; Move uses the current line as the default value. When the move is complete, the first line moved becomes the current line, and the other lines are renumbered according to the direction of the move. For example, the command:

`,+25,100M`

moves 26 lines of text (the current line, plus the 25 lines immediately following the current line) to line 100. At the same time, line 100 becomes the new current line.

If the line number parameters overlap, EDLIN displays the following error message:

Entry Error

To recover, retype the command using correct parameters.

Example

To move lines 20 through 30 to line 100, enter the command:

20,30,100M

P (Page)

The Page command lists a specified range of lines, one page (23 lines) at a time.

Syntax

`[<fline>][,<lline>]P`

fline the first line in the range of lines to be listed.

lline the last line in the range of lines to be listed.

Operation

The Page command pages through a range of lines or an entire file, displaying 23 lines at a time. It differs from the List command in that it changes the current line to the last line displayed.

If the first line parameter is omitted, Page uses the current line plus one as the default value. If the last line parameter is omitted, 23 lines are listed, and the new current line becomes the last line displayed.

Example

To display the first 23 lines of your file, and move the current line to the last line listed, enter:

`1P`

The current line is the last line displayed.

Q (Quit)

The Quit command ends the current editing session without saving any changes or additions.

Syntax

Q

Operation

When you enter the Quit command, EDLIN prompts you to make certain you do not want to save any changes made during the current editing session. The following message is displayed:

Abort edit (Y/N)?

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 more information about the .BAK file.)

Type "N" or any alphanumeric character except "Y", if you want to continue the editing session.

Caution



When you start EDLIN and enter an existing filename, EDLIN erases the previous copy of the file with the filename extension of .BAK to make room to save the new copy. If you respond "Y" to the Abort edit (Y/N)? message, the backup copy of your file is deleted.

R (Replace)

The Replace command replaces all occurrences of a string of text in the specified range with a specified string of text, or with blanks.

Syntax

```
[<fline>][,<lline>][ ? ]R[<string1>]  
[ CTRL Z<string2> ]
```

fline	the first line to search for <string1>.
lline	the last line to search for <string1>.
?	the option that instructs EDLIN to display an "OK?" prompt to verify the correctness of each replacement.
<string1>	the text to be replaced.
CTRL Z	this separates <string2> from <string1>.
<string2>	the text to replace <string1>.

Operation

If either one or both of the line parameters are omitted, EDLIN supplies default values. If you omit the first line parameter, the search begins with the line after the current line. If you omit the last line parameter, the search ends with the last line in memory. If you omit both line parameters, EDLIN searches from the line following the current line to the last line in memory.

The Replace command displays changed lines each time they are changed. You can specify the ? option to request the "O.K.?" prompt after each display of a modified line. This allows you to type "Y" if you want to keep the line in its modified form. Enter any other alphanumeric character (except "Y") if you don't want the modification; the line is kept in its original form. In either case, the search continues for further occurrences of the first string within the range of lines, including multiple occurrences within the same line.

EDLIN has precise syntax requirements for the Replace command. <String1> begins with the character position immediately following the R. <String1> ends when you press **CTRL** Z (if <string2> isn't omitted) or **Enter** (if <string2> is omitted). <String2> begins immediately after **CTRL** Z and ends when you press **Enter**.

If <string1> is terminated with **CTRL** Z and <string2> is omitted, <string2> is read as an "empty" string. For example:

R<string1> **CTRL** Z **Enter**

deletes all occurrences of <string1> in the specified range.

If you omit both <string1> and <string2>, EDLIN re-uses the search string (<string1>) entered with the most recent Search or Replace command, and re-uses the replacement text string (<string2>) entered in the last Replace command. Unless you are certain you know the content of your previous SEARCH and REPLACE commands, it is a good idea to supply these parameters.

When the replace process is complete, the last line changed becomes the current line.

You can place more than one command on a line containing a Replace command by terminating the replacement string with **CTRL** Z, and then beginning the next command in the following character position.

Example

Assume the following file is ready to edit using the Replace command:

```
1: This is a sample file
2: used to demonstrate dynamic line number
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 is no problem
10: because the line numbers are dynamic;
11: they will go all the way
12:*to 65529.
```

To replace the string "and" with the string "or" in lines 2 through 12, enter:

```
2,12 Rand CTRL Zor Enter
```

The result, which is not very satisfactory, follows:

```
4: Delete or Insert
5: (the D or I commors)
8: The insert commor can place new lines
```

Because of the syntax requirements of this command, EDLIN does not differentiate between individual words and strings within words. For this reason, it is a good idea to use the ? option to check and verify replacements before they are made. For example, the same command entered with the ? option:

```
2,12? Rand  Zor 
```

is executed in the following way:

```
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
*-
```

The underscore in this example represents the cursor. The resulting file is far more satisfactory. You can use the list command to check the results:


```
.
.
4: Delete or Insert
5: (The D or I commands)
.
.
.
8: The insert command can place new lines
```

S (Search)

The Search command searches a range of lines to locate a specified string of text.

Syntax

[<fline>][,<lline>][?]S[<string>]

fline	the first line in the range of lines to be searched
lline	the last line in the range of lines to be searched
?	the option that instructs EDLIN to prompt "O.K.?" when the search string is found.
string	the string to be searched for and matched; it must be terminated with  .

Operation

The string you want to have matched by the Search command must be exact as to upper-case and/or lower-case letters. If you enter a search string in all upper-case, only upper-case occurrences are matched; if you enter a search string in all lower-case, or in a combination of upper-case and lower-case, only specific occurrences are matched.

When the first line to contain the specified string is found and displayed, the search ends unless you have used the ? optional parameter. The first line found that matches the specified string becomes the current line. If no match is found, the message **Not found** is displayed.

If the ? 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 "Y" or **Enter**, the line becomes the current line and the search terminates. If you press any other alphanumeric key (except Y), the search continues until another match is found, or until all lines have been searched and the **Not found** message has been displayed.

If you omit either the first line or last line parameters, the system provides default values. If you omit the first line parameter, <fline> defaults to the line following the current line. If you omit the last line parameter, <lline> defaults to the last line in memory. If you omit both line parameters, the system searches from the line following the current line to the last line in memory.

If you do not enter a string, the Search command uses the last search string that was entered in a **Replace** or **Search** command. Unless you are certain you know the contents of the previous **Replace** and **Search** commands, it is a good idea to supply these parameters. If the specified string is not found, the search ends and the **Not found** message is displayed. The current line remains unchanged.

EDLIN has precise syntax requirements for the Search command. The search string (<string>) begins at the character position immediately following the S. The search string ends when you press **Enter**.

You can place more than one command on a line containing a Search command by terminating the Search command with **CTRL** Z, and then beginning the next command in the following character position.

Examples

Assume that you want to edit the following file using the Search command:

```
1: This is a sample file that
2: will demonstrate
3: the Search command
4: using the
5: optional ? parameter
6: and the required <string>
7:*parameter.
```

To search for the first occurrence of "and" in the file, enter:

```
1,7 Sand
```

or

```
1, Sand
```

or

```
1Sand
```

You see the following display:

```
3: the Search Text command.
*
```

The command is terminated, and line 3 is now the current line. As with the Replace command, EDLIN does not differentiate between the word "and" and the letters "and" in the word command. As a result, this may not be the "and" you are searching for. You can continue the search by simply entering the letter "S". The search continues

with the previous search string; searching begins with the line following the current line, which is the line last found. Now, in response to your Search command, you see:

```
*1,7 Sand
      3: the Search Text command
*S
      6: and required string
*
```

Another occurrence of "and" is found and the command is terminated; line 6 is now the current line.

You can also search for strings using the ? option, which causes each matching line to be displayed until the specified range is exhausted. For example, if you enter the following:

```
*1,7 ? Sand
      3: the search Text command.
O.K.? N
      6: and required string
O.K.? Y
*
```

the Search command is terminated and line 6 becomes the current line.

T (Transfer)

The Transfer command merges the contents of a file into a specific location within the file currently being edited.

Syntax

[<line>]T[<d>:][<path>]<filename>

line	the line number location where the file is to be merged with the current file.
d:	the drive that contains the file to be merged.
path	the path to the file to be merged.
filename	the name of the file to be merged with the current file.

Operation

The contents of <filename> are inserted ahead of the specified line in the file currently being edited, and all line numbers are adjusted beginning with the first line merged. If <line> is omitted, Transfer uses the current line as the default value.

The file being merged must reside on the same drive and in the same directory as the file it is being merged into (i.e., the file you are currently editing.) This holds true no matter which drive and directory you started EDLIN from.

Example

If you start EDLIN from drive C: with the following command:

```
C>EDLIN A:\USER#1\FILE1
```

you are currently editing the file called FILE1 that resides on drive A: in the subdirectory USER#1. To transfer the contents of a file called FILE2 into FILE1, FILE2 must reside on the same drive in the same directory as FILE1. Therefore, to add the contents of FILE2 ahead of the first line in FILE1, enter the command:

```
1TA:\USER#1\FILE2
```

W (Write)

The Write command writes a specified number of lines from memory to disc.

Syntax

[<n>]W

n the number of lines, starting with line 1, to be written to disc.

Operation

When you start EDLIN, lines are read into your system's memory until memory is 75% full. The remaining portion of memory is kept available to add data during editing.

If your file is too large to fit into memory, or if it becomes too large as a result of editing, you must save (write) some lines onto your disc to create more room in memory for editing, or for the remainder of your EDLIN file. The Write command allows you to save a specified number of lines from memory onto your disc so you can add more lines to memory. Use the Append command, described earlier in this chapter, to add more lines.

Lines are saved (written on your disc) beginning with line 1. The lines remaining in memory are renumbered, starting with 1, by EDLIN.

EDLIN Error Messages

When EDLIN finds an error, one of the following error messages is displayed. The cause and remedy is given for each message.

MESSAGE: Cannot edit .BAK file--rename file

Cause: You attempted to edit a file with a filename extension of .BAK. .BAK files cannot be edited; this extension is reserved for backup copies of EDLIN files.

Remedy: If you need to edit a .BAK file, either RENAME or COPY the .BAK file giving it a new extension.

MESSAGE: No room in directory for file

Cause: When you attempted to create a new EDLIN file, either the file directory was full, or you specified an illegal disc drive or an illegal filename.

Remedy: Check the filename and disc drive designation for your EDLIN command; if the command is no longer on the screen, and you have not entered a new EDLIN command, you can recover the faulty command using the **F3** (Copyall) key.

If the command line does not appear to have illegal entries, run the CHKDSK command for the specified disc. If the status report shows that the current directory is full, you can specify a path to a different directory in your EDLIN command; create a new directory using the MKDIR command, and specify the path to that directory; or change discs.

MESSAGE: Entry Error

Cause: The last command typed contained a syntax error.

Remedy: Check for the correct syntax and retype the command.

MESSAGE: Line too long

Cause: When you entered a Replace command, your replacement string extended the line length beyond the 253-character limit. EDLIN aborted your Replace command.

Remedy: Shorten the replacement line or divide it into two lines, and try the Replace command again.

MESSAGE: Disk Full--file write not completed

Cause: You entered an End command and there was not enough space on the specified disc for the entire file. EDLIN aborted the E command, exited EDLIN, and returned you MSDOS COMMANDS. Use the DIR command to see if part of the file was written to disc; it will have an extension of .\$\$\$.

Remedy: Unfortunately, there is no way to recover the edited data. Even though there is a directory entry for your file, you will not be able to access the data when you run EDLIN. You need to be certain that the default disc, or the disc specified in your EDLIN command, has sufficient free space for your file prior to beginning your editing session.

MESSAGE: Invalid drive name or file

Cause: You have specified an invalid path or drive when starting EDLIN.

Remedy: Re-enter your EDLIN command using correct path and drive designations.

MESSAGE: Filename must be specified

Cause: You did not include a filename when you started EDLIN.

Remedy: Re-enter the command; the syntax to access EDLIN is:

EDLIN <filename>

MESSAGE: Invalid Parameter

Cause: You specified a switch other than /B, which is the only valid switch available when starting EDLIN.

Remedy: Specify /B to instruct EDLIN to ignore end-of-file marks and process your entire file.

MESSAGE: Insufficient memory

Cause: There is not enough memory to run EDLIN.

Remedy: You must free some memory by writing files to another disc or by deleting files before restarting EDLIN.

MESSAGE: File not found

Cause: The path given for a Transfer command was not found.

Remedy: Enter a valid path when issuing a Transfer command. It may be necessary to return to the MS-DOS command processor and view your disc directory (the DIR command).

MESSAGE: Must specify destination number

Cause: A destination line number was not specified for a Copy or Move command.

Remedy: Re-enter the command with a destination line number.

MESSAGE: Not enough room to merge the entire file

Cause: You attempted to execute a Transfer command when there was insufficient room in memory to hold the file.

MESSAGE: File creation error

Cause: The EDLIN temporary file cannot be created.

Remedy: Either your directory is full, or your file has the same name as a subdirectory in the directory the file is to be edited or located.



9

MS-LINK

This section describes the MS-LINK utility in an abstract, theoretical way for the benefit of readers who are not already familiar with this or similar programs. Readers who want only practical examples are encouraged to refer directly to the sections entitled *Running MS-LINK* and *MS-LINK Examples*.

What MS-LINK is and What it Does

The MS-LINK utility is a program that requires a minimum of 50 Kb of memory (40 Kb for code and data, and 10 Kb for run space). It communicates with the user by displaying prompts on the screen, to which the user responds by entering the appropriate commands.

MS-LINK joins modules of 80286 object code that have been created separately by a compiler and/or assembler from the source code written by a programmer. The object-code modules must be in the form of 80286 object-code (.OBJ) files. MS-LINK transforms one or more object modules into one load module (the .EXE file).

As MS-LINK combines the object modules into an .EXE file, it resolves any external references that the object modules make to symbols that are defined in other modules. MS-LINK can then search one or more library files to find the definition of any external references that it has not been able to resolve.

MS-LINK also produces a list file that shows the external references which the program has resolved. This file also contains any error messages that were generated during MS-LINK operations.

The output (RUN) file from MS-LINK is not bound to specific memory addresses. Therefore, this file can be loaded and executed at any convenient address by your computer's operating system.

MS-LINK uses as much available memory as possible. When it has used all of the available memory, MS-LINK creates a disc file and becomes a virtual memory linker.

Figure 9-1 on the next page diagrams the relationship between MS-LINK and other elements of the system.

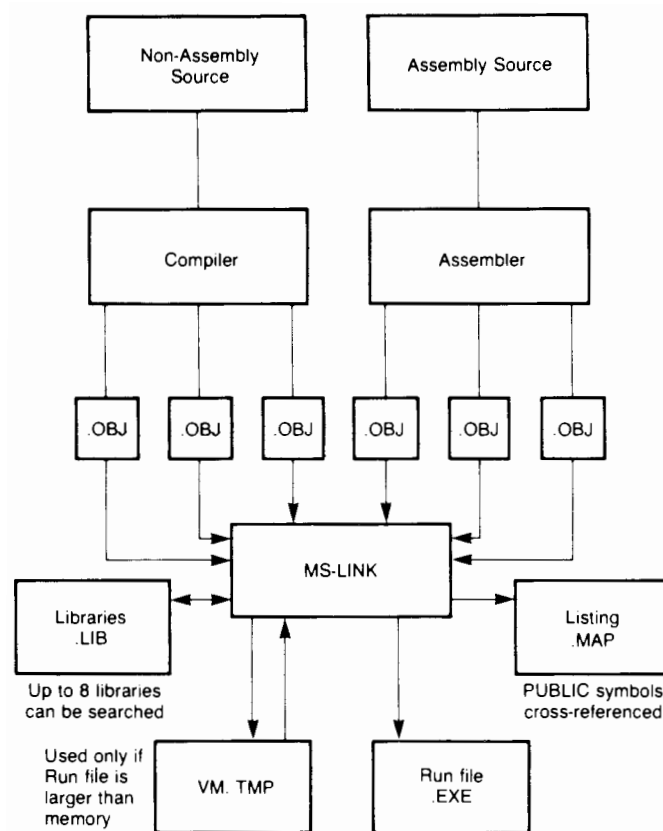


Figure 9-1. MS-LINK and Other System Elements

Definition of Terms

The terms **paragraph**, **segment**, **group** and **class** appear throughout this chapter. Before continuing, you should understand what these terms mean.

A **paragraph** is a contiguous area of memory that is 16 bytes long.

A **segment** is a contiguous area of memory that is up to 64 Kb long. A segment can be located anywhere in 80286 memory adjacent to a paragraph (16-byte) boundary. The contents of a segment are addressed by a segment-register/offset pair.

A **group** is a set of segments that fit within 64 Kb of memory: that is, a collection of segments that are addressed by a single segment register. Either the assembler, the compiler, or the user assigns the group name to the segments. In the latter case, the user assigns the group name in the assembly-language program. For high-level languages, the compiler names the group of segments.

Segments in memory are addressed via their group. Each group is addressed by one segment register. Each segment within a group is addressed by the combination of a segment register and an offset. MS-LINK checks whether the object modules in a group fit within the required 64 Kb of memory. If they don't fit, MS-LINK displays an error message saying so.

A **class** is a collection of segments. The placement of segments in a class controls the order and relative placement of segments in memory. The segments in a class need not have the same address. The user assigns the class name in an assembly-language program. For high-level languages (such as BASIC, FORTRAN, COBOL, and PASCAL), the compiler assigns the class name. The segments are placed in a class at compile time or at

assembly time.

The segments in a class are loaded into memory contiguously. MS-LINK places the segments within a class in the order in which it finds the segments in the object files. A class precedes another class in memory only if a segment in the first class precedes all of the segments in the second class during input to MS-LINK. The classes, which can be loaded across 64 Kb boundaries, are divided into groups for addressing purposes.

In summary, to use MS-LINK, you need to remember that:

- a **paragraph** is a memory area 16 bytes long
- a **segment** is a memory area up to 64 Kb long
- a **group** is a set of segments that occupies up to 64 Kb
- a **class** is a collection of contiguous segments that can be loaded into memory consecutively across 64 Kb boundaries

Figure 9-2 below illustrates the relationship between segments and groups in memory.

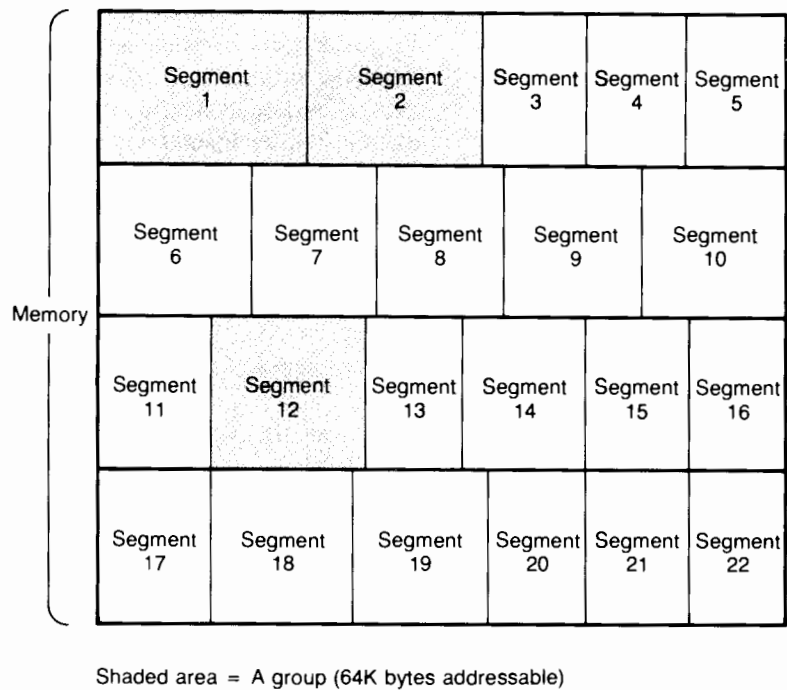


Figure 9-2. Segments and Groups in Memory

How MS-LINK Combines and Arranges Segments

In joining modules of object code, MS-LINK works with the following four types of combinations:

- Private combinations
- Public combinations
- Stack combinations
- Common combinations

These types are declared in the source module for the assembler or the compiler. (The memory-combination type available in the MACRO-86 assembler is treated as a public combination type.) MS-LINK does not automatically place memory-combination types as the highest segments. Instead, MS-LINK generally places segments in the .EXE file that it produces in the same order in which it found the segments in the object modules.

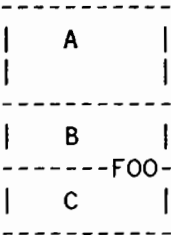
Table 9-1 summarizes how MS-LINK handles segments in private, public, and common combinations. (MS-LINK treats stack combinations very much like public combinations. Because a stack type theoretically has only one segment, combination should not be necessary.)

Table 9-1. How MS-LINK Handles Segments in Private, Public, and Common Types of Combinations

Type	Combination Procedure
Private	Segments are loaded separately, and stay separate. They can be contiguous physically but not logically, even if the segments have the same name. Each private segment has its own base address (that is, a different segment address value).
Public	Segments with the same segment and class name are loaded contiguously, and have the same single base address. The offsets range from the beginning of the first segment through the end of the last segment. (Stack and memory combination types are treated like public types, except that the stack pointer is set to the highest address of the last stack segment.)
Common	Segments with the same segment and class name are loaded overlapping one another, and have the same single base address. The length of the common area is the length of the longest segment.

As shown in Figure 9-2 (Segments and Groups in Memory), the placement of segments in a group in the assembler allows items to be addressed from a single base address for all of the segments in that group.

DS: DGROUP XXXX0H 0 -- relative offset



Any number of other segments can be placed between the segments in a group. Thus, the offset of FOO can exceed the combined size of all of the segments in the group (up to a total of 64 Kb). An operand of DGROUP:FOO returns the offset of FOO from the beginning of the first segment of DGROUP (above, Segment A).

Segments are grouped according to declared class names. MS-LINK loads all of the segments that belong to the first class name it meets, and then loads all of the segments that belong to the next class name it meets, and so on until it has loaded all of the classes.

For example, if your assembly program contains the following segments:

```
A SEGMENT 'FOO'
B SEGMENT 'BAZ'
C SEGMENT 'BAZ'
D SEGMENT 'ZOO'
E SEGMENT 'FOO'
```

they are loaded in the following order (where A, B, C, D and E are segment names; FOO, BAZ, and ZOO are class names):

```
'FOO'
  A
  E
'BAZ'
  B
  C
'ZOO'
  D
```

If you are writing an assembly language program, you can control the ordering of classes in memory by writing a dummy module and listing it first in response to the **MS-LINK Object Modules:** prompt. In this dummy module, you should declare the segments according to classes in the order in which you want to load the classes.

Caution



Do not use this method with BASIC, COBOL, FORTRAN, or PASCAL programs. Instead, let the compiler and the linker perform their tasks in the normal way.

For example, consider the following module:

```
A  SEGMENT  'CODE'
A  ENDS

B  SEGMENT  'CONST'
B  ENDS

C  SEGMENT  'DATA'
C  ENDS

D  SEGMENT  STACK  'STACK'
D  ENDS

E  SEGMENT  'MEMORY'           (9)
E  ENDS
```

In this module, be careful to declare all of the classes to be used in your program. Otherwise, you lose full control over the ordering of the classes.

You can also use this method if you want to load one or more memory-combined types as the last segments of your program. To do so, add **MEMORY** between **SEGMENT** and **'MEMORY'** in the ninth line (9) above.

It is important to understand that these segments are loaded last only because you specified this order, not because of any actions that are part of the linking or assembly operations.

Files that MS-LINK Uses

MS-LINK uses the four types of files shown in Table 9-2 below:

Table 9-2. The Types of Files Used by MS-LINK

Type	Description	Filename Extension
Input files	Object-code files	.OBJ
Output files	Run files List files	.EXE .MAP
VM.TMP file	Temporary file	.TMP
Library files	Library, created earlier	.LIB

MS-LINK works with one or more input files (which are the object modules), and produces two output files. MS-LINK can also create a virtual memory file, and can be instructed to search through from zero to eight library files.

You can specify each type of file, according to the following format:

[<d:>]<filename>

In this specification,

d: is the drive that contains the file.

filename is the name of the file (from 1 to 8 characters). The filename can include an optional filename extension from 1 to 3 characters. If the extension is included, it must be separated from the filename by a period (.).

You can also enter a path, as indicated below:

C:\BASIC\MOD1.OBJ

See the chapter entitled "Understanding MS-DOS Concepts" for more information on filenames and paths.

Input Files

If an input file (object module) specification or a library file specification does not contain a filename extension, then MS-LINK assumes the extensions listed in Table 9-3 on the next page.

Table 9-3. Object and Library Default Extensions

Type of File	Default Extension
Object	.OBJ
Library	.LIB

Output Files

MS-LINK produces the following two types of output files:

- RUN files
- LIST files

MS-LINK appends the filename extensions listed in Table 9-4 below to the output files.

Table 9-4. Default Extensions for Output Filenames

Type of File	Default Extension
RUN	.EXE (cannot be changed by the user)
LIST	.MAP (can be changed by the user)

Virtual Memory File (VM.TMP File)

MS-LINK uses available memory during linking operations. If the files that are to be joined create an output file that exceeds available memory, MS-LINK automatically creates a temporary file and names it "VM.TMP".

In such a case, MS-LINK displays the following message on the screen:

```
VM.TMP has been created
Do not change disc in drive <x:>
```

If this message appears, do not remove the disc from the indicated drive until linking operations are finished. Otherwise, MS-LINK can behave erratically, and might return the following error message:

```
Unexpected end of file on VM.TMP
```

MS-LINK writes the contents of VM.TMP to the filename that you enter in response to the "Run File:" prompt.

It is important to understand that VM.TMP is not a permanent working file. MS-LINK deletes it when the linking session ends. Therefore, do not assign the name "VM.TMP" to any file that you, the compiler, or the assembler creates.

Caution



MS-LINK deletes any existing file called VM.TMP on the disc in the default drive before it creates a new file called VM.TMP. Therefore, if you create a file called VM.TMP, and later MS-LINK needs to create a VM.TMP file, your file will be deleted. As a result, all the data in your VM.TMP file will be lost.

Running MS-LINK

To run MS-LINK, you must perform the following three tasks:

- Enter a command to load and start MS-LINK
- Enter filenames in response to the four MS-LINK prompts
- Specify the settings of the optional switches that control other MS-LINK features

Working With MS-LINK

You can run MS-LINK in any one of the following three ways:

- By entering each filename from the keyboard in response to an individual text prompt.
- By entering all of the filenames on the same line on which you enter the command that loads and starts MS-LINK.
- By entering the name of a separate Response File, containing the necessary filenames, on the same line with the command that loads and starts MS-LINK.

The syntax for each is shown in Table 9-5 on the next page.

Table 9-5. Three Ways to Run MS-LINK

Way	Syntax
Text Prompt	LINK
Command Line	LINK <filename...> [/<switch...>]
Response File	LINK @<filename>

Using the Text Prompt Method

Using this method, you load MS-LINK into memory by entering the following command:

```
LINK
```

MS-LINK then displays four text prompts on the screen, one at a time. Your responses to these prompts command MS-LINK to perform specific tasks.

These four prompts are discussed in detail in the section entitled *Command Prompts*.

The optional switches are described in more detail in the section entitled *Optional Switches*.

Using the Command Line Method

To run MS-LINK using this method, you must enter a statement with the following syntax:

```
LINK <object-list>,[<runfile>],[<listfile>]
    [<lib-list>][/<switch>...]
```

The entries that follow the word **LINK** are your responses to the four prompts. As indicated on the previous page, the entry fields for the different prompts must be separated by a comma.

In this statement,

object-list	is a list of object modules, separated by a plus-sign or a space.
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 the library modules to be searched, separated by a blank space.
/switch	is one or more optional switches, which you can specify after any of the response entries (either directly before any of the commas, or after <lib-list>, as shown above).

Note



Only object-list is required; the other responses are optional. See the section entitled *Command Prompts* for more information.

Table 9-6 below lists the default extensions that MS-LINK assumes if you do not specify filename extensions for the first four items on the command line.

Table 9-6. Default Filename Extensions for Responses

Name of Response Field	Default Filename Extension
<object-list>	.OBJ
<runfile>	.EXE
<listfile>	.MAP
<lib-list>	.LIB

To select the default extension for a response field, you need only enter a second comma, not preceded by a space, after the first comma, as shown in the following example:

```
LINK MOD1+MOD2+MOD3+MOD4/M/P,  
      ,MODLIST,COBLIB.LIB
```

This statement loads MS-LINK, and then loads the following object modules:

```
MOD1.OBJ  
MOD2.OBJ  
MOD3.OBJ  
MOD4.OBJ
```

MS-LINK then does the following things, in the order listed below:

- Links the object modules
- Searches the library file COBLIB.LIB
- Creates a list file called MODLIST.MAP
- Produces a global symbol map (as instructed by the /M switch)
- Pauses (as instructed by the /P switch)

Then, when you press any key, MS-LINK produces the default MOD1.EXE run file. MS-LINK chooses "MOD1" for the default name because "MOD1" is the name of the first object file in the list. If a semicolon terminates an incomplete command line, default values are assigned, and no prompts are given.

Using the Response File Method

To call MS-LINK using this method, enter a statement with the following syntax:

```
LINK @<filename>
```

In this statement,

@ is the character reserved by MS-LINK to indicate that the filename that follows is a response file.

filename is the name of a response file.

A response file contains answers to the MS-LINK prompts, and can also specify one or more of the switches. This

method lets you conduct the MS-LINK session without having to enter responses to the MS-LINK prompts one at a time.

However, before using this method to invoke MS-LINK, you must create the Response File; if you do not do this, MS-LINK displays the following error message:

```
"[<filename>]"
Cannot open response file.
```

When you use this method, MS-LINK displays each prompt in turn, followed by the response(s) from the response file. If the response file does not contain one or more answers to each prompt (in the form of either a filename, a command character, or **Enter**), MS-LINK displays the prompt for which a response is missing, and waits for you to enter a valid response from the keyboard. Once you enter a valid response, MS-LINK continues the linking session.

Creating a Response File


You can use any text editor, such as EDLIN, or the MS-DOS COPY command to create a response file. The file contains four text lines (one for each of the MS-LINK prompts). The responses must appear in the same order in which the command prompts appear, and each response must be separated from the previous response by a carriage return. You can use **Enter** alone on a line to pass over a prompt and use the default value. If no default has been set for that prompt, MS-LINK waits for you to enter an appropriate response from the keyboard before continuing.

The following is an example of a response file:


```
MOD1 MOD2 MOD3 MOD4
/PAUSE/MAP
MODLIST
COBLIB.LIB
```

The first line of this response file tells MS-LINK to load the following four object modules:

```
MOD1.OBJ
MOD2.OBJ
MOD3.OBJ
MOD4.OBJ
```

The second line instructs MS-LINK to pause (to let you swap discs) before generating the .EXE file which defaults to MOD1.EXE. (You may want to review the section entitled *Optional Switches* before using the /PAUSE switch. However, in brief, this switch halts linking operations just before MS-LINK writes the .EXE file. You must press the  key to resume linking operations.)

In response to the third line, MS-LINK assigns the name MODLIST.MAP to the output files.

Last, as instructed in the fourth line, MS-LINK searches the library file COBLIB.LIB.

You can use the command characters "+" and ";" and the switches in a response file the same way you use them in entries from the keyboard. See the sections on *Command Characters* and *Optional Switches* for more information. The following is an example of using command characters in a response file:

```
MOD1+MOD2+MOD3+MOD4
/PAUSE/MAP
MODLIST;
```

Command Prompts

MS-LINK executes the commands that you enter in response to the following four text prompts:

- Object Modules [.OBJ]:
- Run File [First-Object-filename.EXE]
- List File [NUL.MAP]
- Libraries [.LIB]

After you enter a response to one prompt, the next prompt appears. After you enter a response to the last prompt, MS-LINK automatically starts linking the object files. At this point, you do not need to enter any further commands.

Once the link session has ended, MS-LINK returns control to MS-DOS. The re-appearance of the MS-DOS prompt (for example, C>) on the screen indicates that the MS-LINK session was successful. If the link session did not end successfully, the appropriate MS-LINK error message appears on the screen.

MS-LINK prompts you for the names of the object, run, and list files, and for the name of the library file(s). If a prompt has a default response, that default response appears in brackets after the prompt. (The "Object Modules:" prompt is followed only by a default filename-extension response because this prompt has no default filename response; instead, it requires that you enter a filename.)

Table 9-7 below lists the four prompts and the corresponding actions that you can take.

Table 9-7. MS-LINK Prompts and User Responses

Prompt	User Response
Object Modules [.OBJ]:	Enter the name(s) of .OBJ files(s) to be linked. There isn't a default response; you must enter a response.
Run File [First-Object-file.EXE]:	Enter the filename for the executable object code. The default response is: First-Object-file.EXE.
File [NUL.MAP]:	Enter the filename for the output listing. The default response is: NUL.MAP. If you use the default response, no list file is created.
Libraries [.LIB]:	Enter the names of the files to be searched. There isn't a default response; if you don't enter any filenames, no files are searched.

At the end of each response line, you can specify one or more optional switches. Each switch must be preceded by a slash (/). Switches are discussed in detail in the section on *Optional Switches*.

Object Modules [.OBJ]

You must enter a list of the object modules to be linked. MS-LINK assumes that the filename extension is **.OBJ** unless you specify a different extension in response to this prompt.

Modules must be separated by a plus-sign (+) or a space.

Recall that MS-LINK loads segments into classes in the order in which it meets them; this condition is useful when you are deciding on the order in which to enter the object modules.

Run File [First-Object-filename.EXE]:

MS-LINK assigns the specified filename to the run (executable) file it creates. This file will contain the results of the linking session. MS-LINK assigns the filename extension ".EXE" to all run files; it ignores any other extension that you specify.

If you do not enter a response to the **Run File:** prompt, MS-LINK takes the first filename entered in response to the **Object Modules:** prompt and uses it as the name of the run file. MS-LINK assumes that you want the .EXE file placed in the current directory, even if your response to the **Object Modules:** prompt specifies another path or drive.

List File [NUL.MAP]:

Your response to this prompt instructs MS-LINK to generate a list file that contains an entry for each segment in the input (object) modules. Each of these entries also shows the offset (addressing) in the run file.

A typical list file might look like this:

Start	Stop	Length	Name	Class
00000H	0002CH	002DH	CODE	CODE
0002DH	00D74H	0D48H	XCODE	MORECODE
00D75H	00DA2H	002EH	DSEG	DATA
00DBOH	00FAFH	0200H	STACK	STACK
00fBOH	010E5H	0136H	VECTOR	VECTORS

Origin	Group
00D7:0	DGROUP
0000:0	PGROUP

If you do not enter a response to the *List File:* prompt, MS-LINK uses the default response of NUL.MAP and doesn't generate a list file.

Libraries [.LIB]:

A valid response to this prompt is from one to eight library filenames. Press to indicate that you do not want MS-LINK to search any library files.

The library files must have been created beforehand by a library utility such as the MS-DOS LIB utility documented in the *Vectra MS-DOS Macro Assembler* manual, or created as part of a pre-packaged library, such as the library available with a high-level language. MS-LINK assumes that each library file has the filename extension ".LIB".



Library filenames must be separated by a blank space or by a plus sign (+).

MS-LINK searches the library files, in the order in which the files were entered, to resolve external references. When it finds the module that defines the external symbol, MS-LINK processes that module as though it were another object module.

If MS-LINK cannot find the library file on the discs that are currently in the disc drives, the following message appears on the screen:

```
Cannot find library <library-name>  
Enter new drive letter:
```

In response, you need only type the letter designating the desired drive (for example, "C"), without entering a colon after the letter.

To find references, MS-LINK uses a method known as a "dictionary- indexed library search." Under this approach, MS-LINK finds definitions for external references via access to an index, instead of searching for each reference by scanning a file sequentially from beginning to end. This method saves time during a linking session that requires a library search.

When using object modules generated by a compiler, you should know that the compiler may assume the existence of certain default library files. For example, MS-COBOL (Version 1.10) assumes that the library files COBOL1.LIB and COBOL2.LIB exist. In this case, if you are using MS-LINK to join COBOL object (.OBJ) modules, you should enter the names of these library files.

Command Characters

MS-LINK provides the following three command characters:

- The plus-sign (+)
- The semicolon (;)
-

The Plus-Sign

You can use a plus-sign (+) to separate entries and to enter a series of items in response to the **Object Modules:** prompt and the **Libraries:** prompt. (You can also use a blank space, or a mixture of blank spaces and plus-signs, to separate object modules on the same line.)

To enter a series of responses that occupy more than one line (80 characters, less the number of characters occupied by the prompt), enter a plus-sign followed by at the end of the line. If the Plus-sign/ is the last entry in response to the **Object Modules:** or **Libraries:** prompt, MS-LINK displays the **Object Modules:** or **Libraries:** prompt again, and you can continue to enter responses.

After you enter the names of all of the modules to be linked, be sure to end the response line with a module name and -- NOT with a Plus-sign and .

For example, to use spaces and plus-signs to enter a set of modules called MOD1, MOD2, MOD3, MOD4, and MOD5, you can enter the following responses in response to the **Object Modules:** prompt:

```
Object Modules [.OBJ]: MOD1 MOD2+  
Object Modules [.OBJ]: MOD3 MOD4+  
Object Modules [.OBJ]: MOD5 
```

The Semicolon

After you respond to the **Object Modules:** prompt, you can use a single semicolon (;) followed immediately by to select the default responses for the remaining prompts. Entering this command saves time and avoids the need to enter a series of keystrokes.

However, once you enter a semicolon, you cannot enter a response to any of the remaining prompts in the current MS-LINK session. Therefore, do not use the semicolon to skip over one or more prompts on your way to a prompt that you do want to respond to. Instead, use one or more keystrokes to do so.

For example, consider the prompts and responses shown below:

```
Object Modules [.OBJ]: MOD1 MOD2  
Run Module [MOD1.EXE]: ;
```

Because a semicolon appears directly before the in the response to the **Run Module** prompt, none of the subsequent prompts appear, and MS-LINK assumes the default responses to them (including NUL.MAP for the listing file).

CONTROL C

To end an MS-LINK session at any time, press **CTRL** and **C**, simultaneously. If you enter an incorrect response (such as an undesired or misspelled filename), you must end the MS-LINK session by entering **CTRL** **C**, and then begin a new MS-LINK session. **CTRL** **Break** is equivalent to **CTRL** **C**.

However, if you have typed an erroneous response but have not entered it (that is, if the cursor is still on the line that contains the mistake), you can press the **←** key to delete each erroneous character, and then type a correct response.

Optional Switches

A table of optional switch functions appears below.

Table 9-8. Optional Switch Functions

Switch	Function
/DSALLOCATE	Loads data at the high end of the data segment. This is required for PASCAL and FORTRAN programs.
/HIGH	Places the run file as high as possible in memory. This must not be used with PASCAL or FORTRAN programs.

Table 9-8. Optional Switch Functions (Cont.)

Switch	Function
/LINENUMBERS	Includes line numbers in the list file.
/MAP	Lists all global symbols and their definitions in the .MAP file.
/PAUSE	Pauses the MS-LINK session. This causes the system to wait until the user presses the Enter key.
/STACK:<number>	Sets a fixed stack size in the run file.

You must specify optional switches at the end of a prompt response, regardless of the method used to invoke MS-LINK. You can group the switch specifications at the end of any one response, or you can place them at the end of several responses. If you include more than one switch at the end of one response, each switch must be preceded by a slash (/).

For example:

Object Modules [.OBJ]:MOD1/DS/HI/STACK:512

The switch specifications can be spelled out in whole, or can be truncated. However, a truncation must consist of a sequential subset of the letters in the specification, starting with the first letter. No letter(s) can be skipped or transposed. The examples listed below show

valid and invalid truncations for a switch specification; the /DSALLOCATE switch has been used as a sample:

Valid	Invalid
/D	/DSL
/DS	/DAL
/DSA	/DL
/DSALLO	/ALLOC
/DSALLOCA	/DSALLOCT

The /DSALLOCATE Switch

This switch instructs MS-LINK to load all data in the group (DGROUP) at the high end of the group. (Otherwise, MS-LINK loads all data at the low end of the group.)

At runtime, the DS pointer is set to the lowest possible address, thus letting the entire data segment be used. If the /DSALLOCATE switch is used with the /HIGH switch, the user application can dynamically allocate any available memory located below the area specifically allocated within DGROUP. However, this dynamically allocated memory can still be addressed by the same DS pointer. Certain compilers need this type of dynamic allocation.

Note

Your application program can dynamically allocate up to 64 Kb, less the amount allocated within DGROUP.

The /HIGH Switch

The /HIGH switch instructs MS-LINK to place the image of the run file as high as possible in memory. Otherwise, MS-LINK places the image of the run file as low as possible in memory. This action can cause the program to be written over the transient portion of PAM or the COMMAND.COM file. In such a case, MS-DOS displays one of the following error messages:

Insert COMMAND.COM in default drive
and strike any key when ready

or

Insert System Disc in A: and
strike any key to continue.

Note

Do not use the /HIGH switch with PASCAL or FORTRAN programs. They will not run.

The /LINENUMBERS Switch

The /LINENUMBERS switch instructs MS-LINK to include in the listing file the line numbers and addresses of the source statements in the input modules. (Otherwise, MS-LINK omits line numbers from the listing file.)

Not all compilers produce object modules that contain line-number information. If the object module doesn't contain line-number information, then MS-LINK cannot include it.

The /MAP Switch

The /MAP switch instructs MS-LINK to list all public (global) symbols defined in the input modules. If you do not specify /MAP, MS-LINK lists only errors (including undefined global symbols), symbols, classes, and groups.

The MAP switch produces two lists: one in alphabetical order and one in value order. The lists state the value of each symbol, and also give its segment:offset location in the run file.

After MS-LINK locates and searches the specified library file(s) (if told to do so), it produces a map listing the segments in the order in which they appear in the run (.EXE) output file. A typical linker map appears in Figure 9-3 on the next page:

Figure 9-3. Typical MS-LINK Linker Map

Start	Stop	Length	Name
00000H	009ECH	09EDH	CODE
009F0H	01166H	0777H	SYSINITSEG

Address	Publics by Name
009F:001	BUFFERS
009F:000	CURRENT__DOS__LOCATION
009F:001	DEFAULT__DRIVE
009F:000	DEVICE__LIST
009F:001	FINAL__DOS__LOCATION
009F:000	MEMORY__SIZE
009F:000	SYSINIT

Address	Publics by Value
009F:000	SYSINIT
009F:000	CURRENT__DOS__LOCATION
009F:000	FINAL__DOS__LOCATION
009F:000	DEVICE__LIST
009F:000	MEMORY__SIZE
009F:001	DEFAULT__DRIVE
009F:001	BUFFERS

In the first part of this example, the "Start" and "Stop" columns do not contain the absolute addresses where these segments are loaded. Instead, they contain the 20-bit hex address of each segment relative to the beginning of the load module. (This point is known as "location zero.")

In the second part of this example, MS-LINK lists the public symbols: first alphabetically by name, and then by value.

The /PAUSE Switch

The /PAUSE switch causes MS-LINK to pause the linking operations until you press the **Enter** key. (Otherwise, MS-LINK runs through the linking session non-stop from beginning to end.) The pause lets you change discs before MS-LINK outputs the run (.EXE) file.

When MS-LINK encounters the /PAUSE switch, it displays the following message:

About to generate .EXE file
Change disc and press Enter

However, do not remove a disc that is to receive the list file, or a disc that contains the VM.TMP file (that is, if MS-LINK created a VM.TMP file).

The /STACK:<number> Switch

In this specification, <number> represents any positive integral numerical value (expressed in decimal notation) up to 65,536 bytes. If you specify a value from 1 through 511, MS-LINK uses 512. If you do not specify the /STACK switch, MS-LINK looks for the stack size

information in the .OBJ files.

All compilers and assemblers should provide information in the object modules that lets MS-LINK compute the required stack size. At least one object module must contain a stack-allocation statement. Otherwise, MS-LINK displays the following error message:

Warning : No Stack Statement

Error Messages

If an error occurs during an MS-LINK linking session, the linker reports the error and attempts to continue the linking session. Therefore, after you determine the cause and correct the erroneous condition, you must restart MS-LINK and conduct the linking session again.

The MS-LINK error messages are listed below, along with explanations and solutions or suggested actions.

MESSAGE: ATTEMPT TO ACCESS DATA OUTSIDE
 OF SEGMENT BOUNDS, POSSIBLY BAD
 OBJECT MODULE

Cause: An object file is deficient.

Remedy: Recompile the object file and try
 again.

MESSAGE: BAD NUMERIC PARAMETER

Cause: A numerical value is not expressed in
 digits.

Remedy: Express the numerical value in digits.

MESSAGE: CANNOT OPEN TEMPORARY FILE

Cause: MS-LINK cannot create the VM.TMP
 virtual memory file because the disc
 directory is full.

Remedy: Insert a new disc. However, do not
 change the disc that is to receive the
 LIST.MAP file; in this case, restart
 LINK.

MESSAGE: ERROR: DUP RECORD TOO COMPLEX

Cause: The DUP record in an assembly-language module is too complex.

Remedy: Simplify the DUP record in the assembly language program.

MESSAGE: ERROR: FIXUP OFFSET EXCEEDS FIELD WIDTH

Cause: An assembly-language instruction refers to an address with a short instruction instead of a long module.

Remedy: Edit the assembly-language source and re-assemble the instruction.

MESSAGE: INPUT FILE READ ERROR

Cause: An object file is deficient.

Remedy: Recompile the object file and try again.

MESSAGE: INVALID OBJECT MODULE

Cause: One or more object modules are improperly formed or incomplete (for example, this happens when assembly is stopped in mid-process).

Remedy: Recompile the object module(s).

MESSAGE: SYMBOL DEFINED MORE THAN ONCE

Cause: MS-LINK found two or more modules that define a single symbol name.

Remedy: Remove the definition or change the name in one of the conflicting

modules. You should be aware that some languages use only the first eight characters in a symbol name.

MESSAGE:

PROGRAM SIZE OR NUMBER OF SEGMENTS EXCEEDS CAPACITY OF LINKER

Cause:

The aggregate size of all of the files, once linked, cannot exceed 384 Kb; the total number of segments cannot exceed 255.

Remedy:

Review the size of the files and the number of segments; reduce the aggregate size by combining segments.

MESSAGE:

REQUESTED STACK SIZE EXCEEDS 64K

Cause:

A stack size greater than 64 Kb was requested.

Remedy:

Use the /STACK switch to specify a stack size that is less than or equal to 64 Kb.

MESSAGE:

SEGMENT SIZE EXCEEDS 64K

Cause:

64 Kb is the upper limit of the addressing system.

Remedy:

Reduce the segment size by splitting it into two or more segments.

MESSAGE:

SYMBOL TABLE CAPACITY EXCEEDED

Cause:

Many long names were entered (occupying approximately 25 Kb).

Remedy:

Review the number and length of the



names entered in the symbol table and shorten the names.

MESSAGE: TOO MANY EXTERNAL SYMBOLS IN ONE MODULE

Cause: The upper limit on external symbols per module is 256.

Remedy: Reduce the number of external symbols per module by eliminating names or by splitting one or more modules.

MESSAGE: TOO MANY GROUPS

Cause: The upper limit on groups is 10.

Remedy: Declare fewer groups.

MESSAGE: TOO MANY LIBRARIES SPECIFIED

Cause: The upper limit on library files is 8.

Remedy: Reduce the number of library files specified in response to the "Libraries" prompt. Use the MS-DOS LIB Utility documented in the *Vectra MS-DOS Macro Assembler* manual to combine libraries and/or remove unnecessary routines.

MESSAGE: TOO MANY PUBLIC SYMBOLS

Cause: The upper limit on public symbols is 1024.

Remedy: Declare fewer public symbols.

MESSAGE: TOO MANY SEGMENTS OR CLASSES

Cause: The upper limit on segments and classes (considered together) is 256.

Remedy:	Declare fewer segments or classes.
MESSAGE:	UNRESOLVED EXTERNALS: <list>
Cause:	For the external symbols listed, MS-LINK found no defining module among the modules or library files specified.
Remedy:	Include the necessary definition module among the specified modules or library files by placing filename after the Object Modules or Libraries prompt.
MESSAGE:	VM READ ERROR
Cause:	A disc problem, not caused by MS-LINK.
Remedy:	Check your system.
MESSAGE:	WARNING : NO STACK SEGMENT
Cause:	None of the object modules specified contains a statement allocating stack space.
Remedy:	Review the object modules; at least one object module should contain a stack-allocation statement.
MESSAGE:	WARNING : SEGMENT OF ABSOLUTE OR UNKNOWN TYPE
Cause:	An object module is bad, or an improper module was loaded for linking (i.e., a module MS-LINK cannot handle, such as an absolute object module).
Remedy:	Verify the type of module and/or segment loaded.

MESSAGE:	WRITE ERROR IN TMP FILE
Cause:	No more disc space remains into which the VM.TMP file can expand.
Remedy:	Free some disc space by moving or deleting files.
MESSAGE:	WRITE ERROR ON RUN FILE
Cause:	The remaining disc space is too small for the run file.
Remedy:	Free some disc space by moving or deleting files.

MS-LINK Examples

The following examples demonstrate the use of the MS-LINK Utility.

Example 1:

This example demonstrates the linking of object modules from several discs and several directories, and the use of the "+" command character to enter a series of responses to the Object Modules prompt. Note, too, the use of the semicolon (;) to select default filenames for the Run File and List File, and to pass over the Library search.

```
B>link
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
Object Modules [.OBJ]: MOD1+MOD2+MOD3+  
Object Modules [.OBJ]: E:\BASIC\MOD4+  
Object Modules [.OBJ]: C:MOD5  
Run File [MOD1.EXE]: ;
```

```
B>
```

Example 2:

In this example, the user supplies RUNFILE as the name of the run file. Otherwise, the Linker would use the name of the first Object Module and name the run file MOD1.EXE. Note the use of the semicolon to skip over the remaining prompts.

```
B>LINK
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
Object Modules [.OBJ]: MOD1+MOD2  
Run File [MOD1.EXE]: RUNFILE;
```

```
B>
```


Example 3:

In this third example, the user specifies the name of a list file, then uses the MS-DOS TYPE command to view its contents. Note the use of the semicolon to skip over the remaining prompts.

```
B>LINK
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
Object Modules [.OBJ]: MOD1 MOD2 MOD3  
Run File [MOD1.EXE]:  
List File [NUL.MAP]: MOD1;
```

```
B>TYPE MOD1.MAP
```

Start	Stop	Length	Name	Class
00000H	0002CH	002DH	CODE	CODE
0002DH	00D74H	0D48H	XCODE	MORE_CODE
00D75H	00DA2H	002EH	DSEG	DATA
00DB0H	00FAFH	0200H	STACK	STACK
00FB0H	010E5H	0136H	VECTOR	VECTORS

Origin	Group
00D7:0	DGROUP
0000:0	PGROUP

```
B>
```

Example 4:

Here, the user specifies one object file, skips over the run file and list file prompts causing default file names to be used, and specifies a Library Search by entering a filename in response to the Libraries prompt.

```
B>LINK
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
Object Modules [.OBJ]: MOD1  
Run File [MOD1.EXE]:  
List File [NUL.MAP]:  
Libraries [.LIB]: LIBRARY1
```

```
B>
```

Example 5:

Example 5 is the same as Example 1, except that the user places all entries specifying object modules on the command line (the Command Line Method), separated by the "+" command character.

```
B>LINK MOD1+MOD2+MOD3+E:\BASIC\MOD4+C:MOD5;
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
B>
```

Example 6:

This example achieves the same result as Example 3, except that the user places all instructions on the command line rather than responding to individual prompts:

```
B>LINK MOD1 MOD2 MOD3,,MOD1;
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
B>TYPE MOD1.MAP
```

Start	Stop	Length	Name	Class
00000H	0002CH	002DH	CODE	CODE
0002DH	00D74H	0D48H	XCODE	MORE_CODE
00D75H	00DA2H	002EH	DSEG	DATA
00DB0H	00FAFH	0200H	STACK	STACK
00FB0H	010E5H	0136H	VECTOR	VECTORS

Origin	Group
00D7:0	DGROUP
0000:0	PGROUP

```
B>
```

Note the following:

- Both "+" and space are used as separators on the command line
- A comma followed by a second comma is used to skip the run file prompt.

- The semicolon is used to skip the Libraries prompt so that no library search is made.
- Finally, the user uses the MS-DOS TYPE command to view the contents of the list file MOD1.MAP.

Example 7:

In Example 7, the user creates a response file called LINKOBS, and uses the MS-DOS TYPE command to view its contents. The command to call and run MS-LINK is followed by:

```
LINK @LINKOBS
```

enabling the response file to respond to the four prompts.

```
B>TYPE LINKOBS  
MOD1+MOD2+MOD3+  
E:\BASIC\MOD4+  
C:MOD5  
;
```

```
B>LINK @LINKOBS
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
Object Modules [.OBJ]: MOD1+MOD2+MOD3+  
Object Modules [.OBJ]: E:\BASIC\MOD4+  
Object Modules [.OBJ]: C:MOD5  
Run File [MOD1.EXE];
```

```
B>
```

If the Linking session is not successful, error messages are displayed before control is returned to MS-DOS.

Example 8:

In this example, the response file method is again used. The Linker is unable to locate the object module LONGFILENAME.OBJ and prompts the user to change discs; the user terminates the session with the **CTRL C** command character.

```
B>TYPE INVALID
LONGFILENAME
LONG
```

```
B>LINK @INVALID
```

```
Microsoft 8086 Object Linker
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
Object Modules [.OBJ]: LONGFILENAME
Run File [LONGFILENAME.EXE]: LONG
List File [NUL.MAP]:
Libraries [.LIB]:
Cannot find file LONGFILENAME.OBJ
change diskette <hit ENTER> ^C
```

```
B>
```

Example 9:

In this example, the user fails to specify a library file and the Link session pauses. The user enters the library file so the session can continue.

```
B>TYPE SAMPLE
MOD1 MOD2 MOD3
MOD1PROG
MOD1MAP
```

```
B>LINK @SAMPLE
```

```
Microsoft 8086 Object Linker
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
Object Modules [.OBJ]: MOD1 MOD2 MOD3
Run File [MOD1.EXE]: MOD1PROG
List File [NUL.MAP]: MOD1MAP
Libraries [.LIB]: library
```

```
B>
```


Example 10:

The following lines of code illustrate the use and non-use of the /DS switch. In the first part of the sample code, the /DS switch is used causing data to be placed in high memory. In the second part of the sample code, the switch is not used, and data is placed in low memory.

B>LINK

Microsoft 8086 Object Linker
Version 3.00 (C) Copyright Microsoft Corp 19XX

Object Modules [.OBJ]: SAMPLE/DS
Run File [SAMPLE.EXE]:
List File [NUL.MAP]: CON;

Start	Stop	Length	Name	Class
00000H	00012H	0013H	YCODE	CODE
00020H	0003AH	001BH	CODE	CODE
0003BH	00051H	0017H	DSEG	DATA
00052H	00D99H	0D48H	XCODE	MORE_CODE
00DA0H	00ED5H	0136H	VECTOR	VECTORS
00EE0H	010DFH	0200H	STACK_SE	STACK

Origin	Group
F0EE:0	DGROUP
0002:0	PGROUP

Program entry point at 0002:0000

B>LINK

Microsoft 8086 Object Linker
Version 3.00 (C) Copyright Microsoft Corp 19XX

Object Modules [.OBJ]: SAMPLE

Run File [SAMPLE.EXE]:

List File [NUL.MAP]: CON;

Start	Stop	Length	Name	Class
00000H	00012H	0013H	YCODE	CODE
00020H	0003AH	001BH	CODE	CODE
0003BH	00051H	0017H	DSEG	DATA
00052H	00D99H	0D48H	XCODE	MORE_CODE
00DA0H	00ED5H	0136H	VECTOR	VECTORS
00EE0H	010DFH	0200H	STACK_SEG	STACK

Origin Group

0003:0 DGROUP

0002:0 PGROUP

Program entry point at 0002:0000

B>

Example 11:

Because the /HIGH switch is used in this example, COMMAND.COM is overwritten in memory. The Operating System notifies the user with an error message instructing him to insert the disc with the COMMAND.COM file and start again. In the second try, the user omits the /HIGH switch and the Linking session is successful.

```
B>LINK SAMPLE,A:SAMPLE/HIGH;
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
B>A:
```

```
A>SAMPLE  
Simple linker example
```

```
Insert COMMAND.COM in default drive and strike  
ready
```

```
A>B:
```

```
B>LINK SAMPLE,A:SAMPLE;
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
B>A:
```

```
A>SAMPLE  
Simple linker example
```

```
A>B:
```

Example 12:

This example illustrates the use and non-use of the /MAP switch. In the first part of the sample code, the /MAP switch is used; in the second part of the sample, it is not. Note, too, that list output is directed to the console.

```
B>LINK SAMPLE,,CON/MAP;
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

Start	Stop	Length	Name	Class
00000H	00012H	0013H	YCODE	CODE
00020H	0003AH	001BH	CODE	CODE
0003BH	00051H	0017H	DSEG	DATA
00052H	00D99H	0D48H	XCODE	MORE_CODE
00DA0H	00ED5H	0136H	VECTOR	VECTORS
00EE0H	010DFH	0200H	STACK_SEG	STACK

Origin	Group
0003:0	DGROUP
0002:0	PGROUP

Address	Publics by Value
0000:0000	PRINT_SETUP
0002:0007	PRINT_IT
0005:0002	CALLO
0005:089A	CALL1
0003:0D70	ARRAY0
0003:0E38	DIRECTION

Program entry point at 0002:0000

B>LINK SAMPLE,,CON;

Microsoft 8086 Object Linker

Version 3.00 (C) Copyright Microsoft Corp 19XX

Start	Stop	Length	Name	Class
00000H	00012H	0013H	YCODE	CODE
00020H	0003AH	001BH	CODE	CODE
0003BH	00051H	0017H	DSEG	DATA
00052H	00D99H	0D48H	XCODE	MORE_CODE
00DA0H	00ED5H	0136H	VECTOR	VECTORS
00EE0H	010DFH	0200H	STACK_SEG	STACK

Origin	Group
0003:0	DGROUP
0002:0	PGROUP

Address	Publics by Value
0000:0000	PRINT_SETUP
0002:0007	PRINT_IT
0005:0002	CALL0
0005:089A	CALL1
0003:0D70	ARRAY0
0003:0E38	DIRECTION

Program entry point at 0002:0000

Example 13:

This example demonstrates the use of the /PAUSE Switch.

```
B>LINK SAMPLE/P;
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

```
About to generate .EXE file  
Change diskette and press Enter
```

```
B>
```

Example 14:

In this final example, the use of the /STACK switch is shown. In the first part of the sample code, the Stack size is specified as 1024 bytes; in the second part of the sample code, the Stack size defaults to 512 bytes.

```
B>LINK SAMPLE/STACK:1024,,CON;
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

Start	Stop	Length	Name	Class
00000H	00012H	0013H	YCODE	CODE
00020H	0003AH	001BH	CODE	CODE
0003BH	00051H	0017H	DSEG	DATA
00052H	00D99H	0D48H	XCODE	MORE_CODE
00DA0H	00ED5H	0136H	VECTOR	VECTORS
00EE0H	012DFH	0400H	STACK_SEG	STACK

Origin	Group
0003:0	DGROUP
0002:0	PGROUP

Program entry point at 0002:0000

```
B>LINK SAMPLE,,CON;
```

```
Microsoft 8086 Object Linker  
Version 3.00 (C) Copyright Microsoft Corp 19XX
```

Start	Stop	Length	Name	Class
00000H	00012H	0013H	YCODE	CODE
00020H	0003AH	001BH	CODE	CODE
0003BH	00051H	0017H	DSEG	DATA
00052H	00D99H	0D48H	XCODE	MORE_CODE
00DA0H	00ED5H	0136H	VECTOR	VECTORS
00EE0H	010DFH	0200H	STACK_SEG	STACK

Origin	Group
0003:0	DGROUP
0002:0	PGROUP

Program entry point at 0002:0000

B>



10

DEBUG

Introduction

The DEBUG program provides an environment to test binary and executable object files. Whereas EDLIN is available to alter source files, DEBUG can be used for executable files.

DEBUG is an advanced program; it deals with topics such as 8086 family architecture, assembly language programming, and other topics which are more difficult than other MS-DOS concepts. We assume the reader has some familiarity with these related topics.

DEBUG eliminates the need to reassemble a program to see if a problem has been fixed by a minor change to the code. It allows you to alter the contents of a file or of a CPU register, and then to immediately execute your program to test your changes.

DEBUG Commands

Each DEBUG command consists of a single letter followed by one or more parameters. If you enter a DEBUG command incorrectly, DEBUG prompts you by reprinting the command line and pointing to the error with the up-arrow character followed by the word "error." For example:

```
dcx:100 cs:110
      ^Error
```

All DEBUG commands can be aborted at any time by pressing **CTRL Break**. **CTRL S** suspends the display of data so that you can read it before it scrolls off the display. You can restart the display by pressing any alphabetic or numeric key.

Any combination of upper- and lower-case letters can be used in DEBUG commands and their parameters. The DEBUG commands are summarized in the table that follows, and are described in detail later in this chapter.

Table 10-1. DEBUG Command Summary

DEBUG Command	Function
A [<address>]	Assemble
C [<range>]	Compare
D [<range>]	Dump
E <address> [<list>]	Enter
F <range> <list>	Fill
G [=<addr>][<addr>[<addr>...]]	Go
H <value> <value>	Hex
I <value>	Input
L [<addr>[<drive><rec><rec>]]	Load
M <range> <address>	Move
N <filename> [<arglist>]	Name
O <value> <byte>	Output

P [= <address>] [<value>]	Procedure
Q	Quit
R [<register-name>]	Register
S <range> <list>	Search
T [= <address>] [<value>]	Trace
U [<range>]	Unassemble
W [<address> [<drive> <rec> <rec>]]	Write

DEBUG Command Parameters

All DEBUG commands, except the Quit command, accept parameters. Parameters can be separated by either a space or comma delimiter, but a delimiter is only required between two consecutive hexadecimal values. Thus, the following commands are equivalent:

```

dcs:100 110
d cs:100 110
d,cs:100,110

```

Parameter definitions are the same for each DEBUG command that uses them. The DEBUG parameters and their definitions are shown below:

Parameter	Definition
<drive>	A one-digit hexadecimal value that indicates which drive a file is loaded from or written to. These values

designate the drives as follows: 0 = A:, 1 = B:, 2 = C:, ...

<byte>	A two-digit hexadecimal value to be placed in or read from an address or register.
<rec>	A 1- to 3-digit hexadecimal value that indicates the logical record number on the disc and the number of disc sectors to be written or loaded. Logical records correspond to sectors.
<value>	Up to four hexadecimal digits that specify a port number; or the number of times a command should be repeated.
<address>	A two-part designation consisting of either an alphabetic segment register designation or a four-digit segment address plus an offset value. The segment designation or segment address can be omitted, in which case the default segment is used. DS is the default segment for all commands except Go, Load, Trace, Unassemble, and Write. The default segment for these commands 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. See the next section entitled *Memory Addressing on Your Computer* for more information on the address parameter.

<range> Two addresses, as in <address><address>; or one address followed by "L" and a value, as in <address> L <value>. The <value> specifies the number of lines the command should operate on; L80 is assumed. This second syntax for <range> cannot be used if another hex value follows the range because the hex value is interpreted as the second address of the range. For example:

```
CS:100 110
CS:100 L 10
CS:100
```

are all legal specifications for <range>. The following is illegal:

```
CS:100 CS:110
      ^Error
```

The maximum value for <range> is 10000 hex. To specify a value of 10000 hex within four digits, type 0000 (or 0).

<list> A series of byte or string values. <list> must be the last parameter on a command line. For example:

```
fcs:100 42 45 52 54 41
```

<string> Any number of characters delimited by either single or double quotation

marks. Quotation marks within a delimited string must be doubled. For example:

```
'This is a "string" is legal'  
'This is a ''string'' is legal"  
"This is a ""string"" is legal"
```

```
'This 'string' is not legal'  
"This "string" is not legal"
```

Note that double quote marks are not necessary in the following strings:

```
"This ''string'' is not needed"  
'This ""string"" is not needed"
```

The ASCII values of the characters in the string are used as a <list> of byte values.

Memory Addressing on Your Computer

Before you run DEBUG, it helps to understand how addresses are stored in the system's memory.

Every byte of RAM has a unique address which can be stored in up to four bytes (which equals 2 words). This format allows 65,536 unique addresses. To increase the number of possible addresses in RAM, the 8086 family of processors uses an algorithm in which the first word is multiplied by 10 (Hex) and added to the second word. This allows 1,048,576 unique addresses.

To accommodate the increased size of RAM and improve its management, the 80286 processor has four special registers. These registers hold the first word so that programmers can, in most cases, deal only with the second word called the "offset."

Each of these special registers holds the first address of a "segment" of memory that contains up to 65,536 bytes of RAM. For this reason, these registers are called "segment registers". Each register has a special purpose as indicated by its name: CS (code segment), DS (data segment), SS (stack segment), and ES (extra segment).

If you load a program (code) into memory, the code segment register contains the first word of the address. For example, if the address in the code segment register is 20A2, you can view the first portion of your program by entering either of the following commands:

```
dcx:100      (offset)
d20A2:100    (address + offset)
```

In the above examples,

d	represents the DEBUG DUMP command (explained in more detail later in this chapter)
cs	refers to the code segment register
20A2H	is the address of the code segment register, and
100H	is the offset.

In some instances, you do not need to use either the segment name or its address and offset in conjunction with

a specific DEBUG command. For example, the Load command automatically loads at a default address; and when you enter the Dump command without an address specification, you see the first portion of your program displayed on your screen just as if you had entered `dcs:100`. See individual command descriptions for more details.

Using the actual address rather than naming the register (i.e., 20A2 rather than cs) can cause problems. For example, if you load to an address instead of using the default, you can overwrite the operating system, or anything else in memory, while using DEBUG. For this reason, it is a good idea to specify a register or use the default.

Caution



Most DEBUG commands, when used incorrectly, can overwrite critical areas of memory and "crash" your system. If you do make such an error, you can usually recover by performing one of the following steps:

1. Perform a **CTRL** **ALT** **DEL** reset. If this doesn't work, go to step 2.
 2. Turn your system off, then on, and reload the operating system.
-

WE STRONGLY RECOMMEND THAT YOU READ EACH COMMAND DESCRIPTION CAREFULLY BEFORE USING IT.

How to Start DEBUG

The syntax to run DEBUG is:

DEBUG

or

DEBUG [<filename> [<arglist>]]

When you enter the DEBUG command without parameters, DEBUG responds with:

-

Because no filename has been specified, current memory, disc sectors, and disc files can be worked on using any of the DEBUG commands.

Caution



When DEBUG is started, it sets up a program header at offset 0 in the program's work area. You can overwrite this header without consequences if no <filename> is given to DEBUG. However, if you are debugging a .COM or .EXE file, *do not* tamper with the program header below address 5CH (the H stands for Hexadecimal). If you do, DEBUG may not be able to terminate correctly when you quit. Do not attempt to restart a program after the message:

Program terminated normally

is displayed. You must reload the program using the Name and Load commands for it to run properly.

You can also start DEBUG using a command line, as in:

DEBUG [[d:][path]filename[<arglist>]]

When you use this method, the file you specify is loaded into memory. MS-DOS determines the lowest segment after the end of its resident portion and loads a Program Segment Prefix (PSP) at this address. Refer to the *MS-DOS 3.2 Programmer's Reference Manual* for details about the PSP. The PSP is 100H bytes long. The program file is loaded in after the PSP.

If the file has an extension of either .EXE or .HEX, the DS and ES segment registers point to the PSP, and the CS, SS, SP, and IP registers are set to the value specified in the .EXE or .HEX file. .HEX files are assumed to be in Intel .HEX format and are converted into executable format as they are loaded.

For all other files, all segment registers point to the PSP, the IP register is set to 100H, and the SP register is set to FFFEh.

DEBUG also places the number of bytes in the program as a doubleword value in BX: CX. The Least Significant word is placed in CX, the Most Significant word in BX.

The initial values for the flags are:

NV UP EI PL NZ NA PO NC

An <arglist> can be specified only if <filename> is present. The <arglist> is a list of filename parameters and switches that are to be passed to <filename>. Thus, when <filename> is loaded into memory, it is loaded as if it had been started with the command:

A><filename> <arglist>

Here, <filename> is the file to be debugged, and <arglist> is the command line to be passed to the program when it is executed.

DEBUG Command Descriptions

All debug commands are described in detail on the pages that follow. Each DEBUG command consists of a single letter followed by one or more parameters. DEBUG commands can be edited using the Editing Keys described in the appendix entitled "Using the MS-DOS Keyboard".

We strongly recommend that you read the information on the DEBUG commands before you attempt to use them. Many commands contain important notes and cautions.

A (Assemble)

The Assemble command assembles 8086, 8087, and 8088 mnemonics directly into memory.

Syntax

A [<address>]

Operation

All numeric values are hexadecimal and must be entered as 1-4 characters. Prefix mnemonics (such as **JMP** for Jump) must be specified in front of the opcode to which they refer; or they can 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 automatically assembles short, near, or far jumps and calls, depending on the byte displacement to the destination address. These can be overridden with the **NEAR** or **FAR** prefix. For example:

```
0100:0500 JMP 502          ; a 2-byte short jump
0100:0502 JMP NEAR 505     ; a 3-byte near jump
0100:0505 JMP FAR 50A      ; a 5-byte far jump
```

The **NEAR** prefix can be abbreviated to **NE**, but the **FAR** prefix cannot be abbreviated.

DEBUG cannot tell whether some operands refer to a word memory location or a byte memory location. In such a case, the data type must be explicitly stated with one of

the prefixes "WORD PTR" or "BYTE PTR." Acceptable abbreviations are "WO" and "BY", respectively. For example:

```
NEG    BYTE PTR [128]
DEC    WO [SI]
```

Nor can DEBUG tell whether an operand refers to a memory location or to an immediate operand. DEBUG uses the common convention that an operand enclosed in square brackets refers to a memory location. For example:

```
MOV     AX,21      ;Load AX with 21H
MOV     AX,[21]    ;Load AX with contents
                        ;of location DS:21H
```

Two popular pseudo-instructions are available with Assemble. The DB opcode assembles byte values directly into memory; the DW opcode assembles word values directly into memory. For example:

```
DB      1,2,3,4,"THIS IS AN EXAMPLE"
DB      'THIS IS A QUOTE:'
DB      "THIS IS A QUOTE:"

DW      1000,2000,3000,"BACH"
```

Assemble supports all forms of register indirect commands. For example:

```
ADD     BX,34[BP+2].[SI-1]
POP     [BP+DI]
PUSH    [SI]
```

All opcode synonyms are also supported. For example:

LOOPZ	100
LOOPE	100

JA	200
JNBE	200

Press **Enter** without typing an opcode to terminate the Assemble command. The Assemble command can be terminated at any byte position.

C (Compare)

The Compare command compares the portion of memory specified by <range> to a portion of the same size beginning at a specified address.

Syntax

<range><address>

Operation

If <range> and <address> specify identical areas of memory, no comparison is made and you are returned to the DEBUG command prompt.

If <range> and <address> are different, the non-matching bytes are displayed in the following format:

<address1> <byte1> <byte2> <address2>

Examples

To compare the block of memory from 100 to 1FFH with the block of memory from 300 to 3FFH, enter:

C100,1FF 300

or

C100L100 300

The two commands have the same effect.

D (Dump)

The Dump command displays the contents of the specified region of memory.

Syntax

D [<range>]

Operation

If you enter the D command without parameters, 128 bytes are displayed at the first address after the address displayed by the previous Dump command. If a <range> of addresses is specified, the contents of the range are displayed.

The dump is displayed in two portions: a hexadecimal dump with each byte shown as a hexadecimal value, and an ASCII dump with bytes shown as their corresponding ASCII characters. Nonprinting characters are denoted with 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 enter the command:

```
dcx:0100 L 20
```

or

```
cs:0100 011F
```

DEBUG displays the dump in the following format:

```
207E:0100  48 50 20 56 65 63 74 72-61 20 54 6F 75 63 68 20  HP.....
207E:0110  53 63 72 65 65 6E 00 00-00 00 00 00 00 00 00  Computer...
```

If you type the following command:

D

the display is formatted as described above. Each line begins with an address, incremented by 16 from the address on the previous line. Each subsequent D command entered without parameters displays the bytes immediately following those last displayed.

If you type the command:

DCS:100 L 20

the display is formatted as described above, but 20H bytes are displayed. If you then type the command:

DCS:100 115

the display is still formatted in the same way but only the bytes in the range of lines from 100H to 115H in the CS segment are displayed.

E (Enter)

The Enter command enters byte values into memory at the specified address.

Syntax

E <address> [<list>]

Operation

If <address> is specified without the optional list, DEBUG displays the address and its contents, then waits for your input. At this point, DEBUG expects you to perform one of the following actions:

1. Replace a current byte value with a new value by entering the new value after the current value. If the value you enter is not a legal hexadecimal value, or if more than two digits are typed, the illegal extra character(s) are not echoed.
2. Use the space bar to advance to the next byte; to change its value, simply type the new value after the current value. If you space beyond an 8-byte boundary, DEBUG starts a new display line with the address displayed at the beginning.
3. Type a hyphen (-) to return to the preceding byte. If you decide to change a byte prior to the current position, the hyphen returns the current position to the previous byte, and a new line is started with the address and its byte value displayed.
4. Press **Enter** to terminate the Enter command. You can press **Enter** to terminate the command at any byte position.

If the optional list of values is entered with the Enter

command, byte values are replaced automatically with new values. If an error occurs, no byte values are replaced.

Examples

If you enter the following DEBUG command:

```
ECS:100
```

DEBUG might respond by displaying:

```
04Ba:0100 EB._
```

(The underscore represents the cursor.) To change this value to 41, type 41 as shown below:

```
04BA:0100 EB.41_
```

To step through the subsequent bytes, press the SPACE bar to see:

```
04BA:0100 EB.41 10. 00. BC._
```

To change BCH to 42H, enter:

```
04BA:0100 EB.41 10. 00.  
BC.42_
```

Now, to change 10H to 6FH, type the hyphen as many times as needed to return the cursor to byte 0101H (value 10H), then replace 10H with 6FH:

```
04BA:0100 EB.41 10. 00. BC.42  
04BA:0102 00.-  
04BA:0101 10.6F-
```

Press **Enter** to terminate the Enter command and return to the DEBUG command prompt.

F (Fill)

The Fill command fills the addresses in the specified range with the values specified in the list.

Syntax

F <range><list>

Operation

If the specified range contains more bytes than the number of values in the specified list, the list is used repeatedly until all bytes in the range are filled.

If the list contains more values than the number of bytes in the range, the extra values in the list are ignored. If any of the memory in the range is not valid (bad or non-existent), you receive an error message. If a segment value is not specified in the range, DS is assumed.

Examples

If you enter the following DEBUG command:

```
F04BA:100 L 100 42 45 52 54 41
```

DEBUG fills memory locations 04BA:100 through 04BA:1FF with the bytes specified. The five values are repeated until all 100H bytes are filled.



G (Go)

The Go command executes the program currently in memory.

Syntax

`G [=<address>] [<address> [<address>...]]`

Operation

If the Go command is entered without parameters, the program in memory is executed as if it had been executed from the MS- DOS prompt.

If the `=<address>` parameter is set, execution begins at the specified address. The equal sign (=) is required so that DEBUG can distinguish the start address from the breakpoint address(es).

When additional addresses are set, program execution stops at the first of those addresses encountered, regardless of its position in the address list. When program execution reaches a breakpoint, the registers, flags, and decoded instructions 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 can be set. Breakpoints can be set only at addresses containing the first byte of an 80286 opcode. If more than ten breakpoints are set, DEBUG returns the BP error message.

The user stack pointer must be valid and have six bytes available for the Go command. The Go command uses an IRET instruction to cause a jump to the program under test. The user stack pointer is set, and the user flags, Code Segment register, and Instruction Pointer are pushed on the user stack.

Thus, if the user stack is not valid or is too small, the operating system may crash. An interrupt code (OCCH) is placed at the specified breakpoint address(es).

When an instruction with the breakpoint code is encountered, all breakpoint addresses are restored to their original instructions. If execution is not halted at one of the breakpoints, the interrupt codes are not replaced with the original instructions.

Examples

Example 1

If you enter the following command:

```
GCS:7550
```

The program currently in memory executes up to the address 7550 in the CS segment. DEBUG then displays registers and flags, and terminates the Go command.

If you re-enter the Go command after a breakpoint has been encountered, the program in memory restarts at the location of the breakpoint.

When you execute all or part of a program using Go, the registers and memory may change. To run the program again, reload the program using the Load command to reinitialize the registers and memory.

Example 2

If the IP is set to 100H, the command:

```
G 200
```

starts program execution at IP and stops at 200H.

If you enter:

G=100

execution starts at 100H and continues to the end of the program.

If you enter:

G=100 200H

execution starts at 100H and continues to 200H.

H (Hex)

The **Hex** command performs hexadecimal arithmetic on the two parameters specified.

Syntax

H <value><value>

Operation

This command is useful in calculating offsets from current or known addresses. **DEBUG** first adds the two specified values, then subtracts the second value from the first value. The results of these operations are displayed on one line; first the sum, then the difference.

Examples

If you enter the following **DEBUG** command:

H19F 10A

DEBUG first sums the two values, then subtracts 10AH from 19FH. The following result is displayed:

02A9 0095

I (Input)

Purpose

The Input command inputs and displays one byte from the port specified by the value parameter.

Syntax

I <value>

Operation

A 16-bit port address can be specified as the <value>.

Examples

If you enter the following command:

I2F8

and the byte at the port is 42H, DEBUG inputs the byte and displays the following value:

42

Because programmers can use MS-DOS function calls inside their programs, you should not find it necessary to use the Input command.

L (Load)

The Load command loads a disc file or absolute disc sectors into memory.

Syntax

L [<address> [<drive><rec><rec>]]

Operation

To debug a disc file rather than absolute disc sectors, the file to be loaded must have been named either when DEBUG was started, or using the DEBUG Name command. Both the command to start DEBUG and the Name command create a non-extended File Control Block at CS:5C. BX:CX is the number of bytes read.

If you enter the Load command without any parameters, DEBUG loads the named file into memory beginning at address CS:100 and sets BX:CX to the number of bytes loaded.

If the Load command is entered with just an <address> parameter, loading begins at the memory address specified.

If L is entered with all the parameters, absolute disc sectors are loaded rather than a file. The specified records are taken from the specified drive and DEBUG begins loading with the first record specified, and continues until the number of sectors specified in the second record have been loaded.

Examples

If you enter the following commands:

```
B>DEBUG  
-NFILE.COM
```

NFILE.COM is the DEBUG command to Name the file to be loaded into memory. Now, to load FILE.COM, enter:

```
L
```

DEBUG loads FILE.COM and displays the DEBUG prompt. Assuming that you want to load only portions of a file or certain records from disc, you can enter:

```
L04BA:100 2 0F 6D
```

DEBUG then loads 109 (6D hex) records from drive 2 (C:) beginning with logical record number 15 (0FH) into memory starting at address 04BA:0100. When the records have been loaded, DEBUG again returns the "-" prompt.

If the file to be loaded has an .EXE extension, it is relocated to the load address specified in the header of the .EXE file; the address parameter is always ignored for .EXE files. The header itself is stripped from the .EXE file before it is loaded into memory; thus, the size of an .EXE file on disc differs from its size in memory.

If the file named with the Name command or specified when DEBUG is started is a .HEX file, then entering the L command without parameters causes DEBUG to load the file beginning at the address specified in the .HEX file. If the L command includes the address parameter, DEBUG adds the specified address to the address found in the .HEX file to determine the start address for loading the file. Registers BX:CX contain the doubleword count of the number of bytes in the program. The Least Significant word is stored in CX, the Most Significant word in BX.

M (Move)

The Move command moves the block of memory in the specified range to the location beginning at the specified address.

Syntax

M <range><address>

Operation

If you instruct DEBUG to make an "overlapping" move (that is, one in which part of the block overlaps some of the current address), the move is performed without loss of data because 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, then to work toward the highest.

The sequence for moves from lower addresses to higher addresses is to move the data beginning at the block's highest address and to work toward the lowest.

Examples

If you enter the following command:

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. To see the results of the Move, use the Dump command and specify the address specified for the Move command. DEBUG assumes the segment of all addresses is DS unless a segment is specified.

N (Name)

The Name command sets a filename for a later Load or Write command, and/or sets filename parameters.

Syntax

N <filename> [<filename> ...]

Operation

If you start DEBUG without naming a file to be debugged, you must use the Name command before a file can be loaded. The Name command also accepts parameters that are used by the file being debugged.

Assume you have entered the following commands:

```
B>DEBUG
-NFILE1.EXE
-L
-G
```

The Name command performs or enables the following steps:

1. Name sets FILE1.EXE as the filename to be used in any later Load or Write commands during the current DEBUG session.
2. Name also sets FILE1.EXE as the first parameter used by any program that is later debugged during the current DEBUG session.
3. The Load command loads the named FILE1.EXE into memory.

4. The Go command causes FILE1.EXE to be executed with FILE1.EXE as the single filename parameter; that is, FILE1.EXE is executed as if:

```
FILE1 FILE1.EXE
```

had been typed at the MS-DOS prompt.

A more useful group of commands might look like this:

```
B>DEBUG  
-NFILE1.EXE  
-L  
-NFILE2.DAT FILE3.DAT  
-G
```

In this series of commands, Name sets FILE1.EXE as the filename to be used by the subsequent Load command. The Load command loads FILE1.EXE into memory, and 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 the following command had been entered:

```
B>FILE1 FILE2.DAT FILE3.DAT
```

Note



If a Write command is executed at this point, the file being debugged (FILE1.EXE) is saved with the filename FILE2.DAT. To avoid this, always execute a Name command before either Load or Write.

Four regions of memory 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  Unparsed command line
```

A File Control Block (FCB) is set up at CS:5C for the first filename parameter given to the Name command. If a second parameter is entered, then a FCB is set up for it beginning at CS:6C.

The number of characters in the Name command, exclusive of the "N" itself, is given at location CS:80. The actual stream of characters in the Name command, again exclusive of the letter "N," begins at CS:81. Note that this stream of characters can contain switches and delimiters that are legal in any command typed at the MS-DOS command level.

Examples

A typical use of the Name command follows:

```
B>DEBUG PROG.COM
-NPARAM1 PARAM2/C
-G
-
```

In this instance, the Go command executes the file in memory as if the following command line had been typed from MS-DOS:

```
B>PROG PARAM1 PARAM2/C
```

Testing and debugging, therefore, reflect a normal runtime environment for PROG.COM.

O (Output)

The Output command sends the specified byte to the specified output port.

Syntax

O <value> <byte>

Operation

A 16-bit port address is allowed in the output command. If you enter:

O2F8 4F

DEBUG outputs the byte value 4F to output port 2F8.

Because programmers can use MS-DOS function calls inside their programs, you should not find it necessary to use the Output command.

P (Procedure)

The Procedure command executes one instruction and displays the contents of all registers and flags and the decoded instruction.

Syntax

P [=<address>] [<value>]

Operation

The Procedure command operates exactly like the Trace command with the exception that when a subroutine is encountered during a trace operation, it is executed and the trace is resumed at the first line following the subroutine.

This command is most useful to skip over either a tested or well known procedure in order to see the returned values of the next instruction in the current procedure. One such example might be to execute MS-DOS function calls by tracing to INT 21H, (the MS-DOS interrupt for an MS-DOS Function call), then to type P and execute the function. DEBUG returns at the next instruction in your sequence.

See the section on the Trace command later in this chapter for details on the operation of this command.

Q (Quit)

The Quit command terminates the **DEBUG** utility and returns the user to the MS-DOS prompt.

Syntax

Q

Operation

The Quit command uses no parameters. It terminates **DEBUG** without saving the file currently in memory. You are returned to the MS-DOS command level.

Examples

To end the current debugging session, enter:

Q

DEBUG terminates, and control is returned to MS-DOS.

R (Register)

The Register command displays the contents of one or more CPU registers.

Syntax

R [<register-name>]

Operation

When DEBUG is started, the segment registers are set to the bottom of free memory, the Instruction Pointer is set to 0100H, all flags are cleared, and the remaining registers are set to zero.

If no <register-name> parameter is used, the Register command dumps the register save area and displays the contents of all registers and flags.

If a <register-name> is entered, the 16-bit value of that register is displayed in hexadecimal, followed by a colon (:). You can then either enter a <value> to change the register, or you can press if no change is needed.

Valid register-names are:

AX	BP	SS	
BX	SI	CS	
CX	DI	IP	(IP and PC both refer
DX	DS	PC	to the Instruction
SP	ES	F	Pointer.)

Any other entry for register-name results in a BR error message (see the section entitled *DEBUG Error Messages* for an explanation).

If F is used 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 the following table with their codes for SET and CLEAR:

FLAG NAME	SET	CLEAR
Overflow	OV	NV
Direction	DN Decrement	UP Increment
Interrupt	EI Enabled	DI Disabled
Sign	NG Negative	PL Plus
Zero	ZR	NZ
Auxiliary	AC	NA Carry
Parity	PE Even	PO Odd
Carry	CY	NC

When you type the command "RF", the flags are displayed in a row at the beginning of a line in the order shown above. At the end of the list of flags, DEBUG displays a hyphen (-). You can enter new flag values as alphabetic pairs in any order. You do not have to leave spaces between the flag entries.

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 the point of the error and following are not changed.

To exit the R command, press **Enter**. Flags for which new values were not entered remain unchanged.

Examples

If you enter:

R

DEBUG displays all registers, flags, and the decoded instruction for the current location. If the location is CS:11A, then the display looks similar to the following:

```
AX=0E00 BX=00FF CX=0007 DX=01FF SP=039D BP=0000 SI=005C DI=0000
DS=04BA ES=04BA SS=04BA CS=04BA IP=011A NV UP DI NG NZ AC PE NC
04BA:011A CD21          INT      21
```

If you enter:

RF

DEBUG displays the flags:

```
NV UP DI NG NZ AC PE NC - _
```

Now, if you enter a valid flag designation, in any order, with or without spaces, as in:

```
NV UP DI NG NZ AC PE NC - PLEICY Enter
```

DEBUG responds only with the DEBUG prompt. Enter the **R** or **RF** command to see the following display of the changes:

```
RF
NV UP EI PL NZ AC PE CY - _
```

Press to leave the flags this way, or specify new flag values.

S (Search)

The Search command searches the specified range for the specified list of bytes.

Syntax

S <range><list>

Operation

The <list> can contain one or more bytes, each separated by a space or comma delimiter. If the <list> contains more than one byte, only the first address of the byte string is returned. Otherwise, all addresses of the byte in the <range> are displayed. If no match is found, no address is displayed.

Examples

If you enter:

SCS:100 110 41

DEBUG displays a response similar to the following:

04BA:0104

04BA:010D

-

T (Trace)

The Trace command executes one instruction and displays the contents of all registers and flags and the decoded instruction.

Syntax

T [=<address>] [<value>]

Operation

If the optional =<address> parameter is entered, tracing occurs at the specified address. The optional value causes DEBUG to execute and trace the number of steps specified by <value>.

Because the Trace command uses the hardware trace mode of the 80286 microprocessor, you can also trace instructions stored in Read Only Memory.

Examples

When you type:

T

DEBUG displays the state of the registers and flags after the last instruction is executed, and decodes the next instruction to be executed. Assuming that the current position is 04BA:011A, DEBUG might return this information:

```
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 enter:

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 continues to scroll until the last instruction is executed. When scrolling stops, you see the register and flag values for the last few instructions performed.

Remember that **CTRL** **S** suspends the display at any point to allow you time to study the registers and flags for any instruction. Resume scrolling by pressing any alphabetic or numeric character.

Tracing interrupts is possible, as is tracing INs and OUTs; however, most users don't want to watch each instruction of an MS-DOS function call being executed. For this reason, it is recommended that you use the **P** (Procedure) command at INT 21H, the MS-DOS interrupt for an MS-DOS function call, and at CALL instructions.



U (Unassemble)

The Unassemble command disassembles bytes and displays their corresponding source statements with their addresses and byte values.

Syntax

U [<range>]

Operation

If you enter the U command without parameters, 20 hexadecimal bytes are disassembled at the first address after the address 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.

The display of disassembled code has the same appearance as a listing for an assembled file.

Examples

If you enter:

U04Ba:100 L10

DEBUG disassembles 16 bytes beginning at address
04BA:0100:

04BA:0100	206472	AND	[SI+72],AH
04BA:0103	69	DB	69
04BA:0104	7665	JBE	016B
04BA:0106	207370	AND	[BP+DI+70],DH
04BA:0109	65	DB	65
04BA:010A	63	DB	63
04BA:010B	69	DB	69
04BA:010C	66	DB	66
04BA:010D	69	DB	69
04BA:010E	63	DB	63
04BA:010F	61	DB	61

If you enter:

U04ba:0100 0108

you see the following display:

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.

W (Write)

The Write command writes the file being debugged to a disc file.

Syntax

W[<address>[<drive><rec><rec>]]

Operation

If you enter **W** without 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 entered with just an <address>, then the file is written beginning at that address. Again, **BX:CX** must be set before you use the Write command.

Note that if a file is loaded and modified, the name, length, and starting address are all set correctly to save the modified file, so long as the length has not changed. The file must have been named either when **DEBUG** was called, or with the **N (Name)** command, described earlier in this section. Both the **DEBUG** call and the **Name** command format a filename properly in the normal format of a File Control Block at **CS:5C**.

If the W command is entered with all the parameters, the write begins from the memory address specified; the file is written to the specified <drive>. DEBUG writes the file beginning with the logical record number specified by the first <rec>; and continues to write the file until the number of sectors specified in the second <rec> have been written.

Caution



Writing to absolute sectors is **EXTREMELY** dangerous because the process bypasses the file handler.

Debug Error Messages

Error Code	Definition
BF	<p>Bad flag</p> <p>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.</p>
BP	<p>Too many breakpoints</p> <p>You specified more than ten breakpoints as parameters to the G (Go) command. Retype the Go command with ten or fewer breakpoints.</p>
BR	<p>Bad register</p> <p>You typed the R command with an invalid register name. See the Register command for the list of valid register names.</p>
DF	<p>Double flag</p> <p>You typed two values for one flag. You can specify a flag value only once per RF command.</p>



A

MS-DOS Message Directory

MS-DOS messages are issued when MS-DOS or an MS-DOS command encounters an error, needs to inform the user of system status, or needs to prompt the user for an action.

MS-DOS messages are generated by an MS-DOS command, or by the resident portion of MS-DOS. Messages generated by the resident portion of MS-DOS occur only when an error condition is encountered on a disc or device. These disc and device error messages are listed in the next section. The messages generated by the various MS-DOS commands consist of error messages, informational messages, and prompts. They are listed following the disc and device error messages.

Disc and Device Errors

If a disc or device error occurs at any time during a command or application program, MS-DOS displays an error message in the following format:

<type><action> drive <x>
Abort,Retry,Ignore:

In this message, <type> is one of the following error conditions:

- Write protect error
- Bad unit error
- Not ready error
- Bad command error
- Data error
- Bad call format error
- Seek error
- Non-DOS disk error
- Sector not found error
- No paper error
- Write fault error
- Read fault error
- General failure
- Sharing violation
- Lock violation
- Invalid disk change
- FCB unavailable

The <action> is either **READING** or **WRITING**. The drive designator (drive <x>) indicates the drive on which the error occurred. For example, if you attempt to write to the disc in drive A: and the disc has a write protect tab on it, the following error message is displayed:

Write protect error writing drive A
Abort, Retry, Ignore?

MS-DOS waits for you to respond in one of the following ways:

A (Abort)	End the program requesting the disc read or write.
R (Retry)	Repeat the operation. Use this response if you corrected the error (e.g., if you removed the write protect tab in the above example).
I (Ignore)	Ignore the error condition and proceed with the program.

In addition to the Abort,Retry,Ignore messages, you might encounter the following error message:

FILE ALLOCATION TABLE BAD FOR DRIVE x

It indicates that one of the file allocation tables (FATs) has an error in it. If MS-DOS displays this error message, run CHKDSK with the /F option. If the error persists after you run CHKDSK, the disc is unusable and must be formatted prior to further use.

MS-DOS Command Messages

MESSAGE:	Abort Edit (Y/N)?
Command:	EDLIN
Cause:	This message is displayed when you enter the Q (Quit) command in EDLIN. The Quit command exits the editing session without saving any editing changes.
Remedy:	Enter "Y" if you want to exit without saving the editing changes to disc, or "N" if you want to cancel the command and continue the editing session. Use the "E" command to exit EDLIN and save editing changes.
MESSAGE:	Abort, Retry, Ignore?
Command:	MS-DOS
Cause:	A disc or disc device error has been detected by MS- DOS during a program or command's execution.
Remedy:	Enter "A" if you want to abort the program, "R" if you MS-DOS to attempt the operation again, or "I" if you want MS-DOS to proceed with the program or command.
MESSAGE:	ABORTED - Assigned Port # missing or invalid

Command:	MODE
Cause:	This error message is displayed when you are re-directing printer output with the MODE command and the serial port specified is not present on the system or the port specified is not COM1:, COM2:, COM3:, or COM4:.
Remedy:	Re-enter command line.
MESSAGE:	ABORTED - Bad device parameter keyword
Command:	MODE
Cause:	This error message is displayed when a device other than LPTn: or COMn: is specified on the command. MODE only affects the parallel ports, serial ports, and screen.
Remedy:	Re-enter command line.
MESSAGE:	ABORTED - Illegal BUSY value
Command:	MODE
Cause:	This error message is displayed when something other than "P" is specified to initialize a port as the printer device.
Remedy:	Re-enter command line.
MESSAGE:	ABORTED - Illegal STOPBITS value
Command:	MODE
Cause:	This error message is displayed when a value other than 1 or 2 is specified as the number of stop bits.

Remedy:	Re-enter command line.
MESSAGE:	ABORTED - Illegal DATABITS value
Command:	MODE
Cause:	This error message is displayed when a value other than 7 or 8 is specified as the number of data bits.
Remedy:	Re-enter command line.
MESSAGE:	ABORTED - Invalid Switch
Command:	MODE
Cause:	This error message is displayed when an invalid option is specified on the command line.
Remedy:	Re-enter command line.
MESSAGE:	Access denied
Command:	REPLACE, XCOPY
Cause:	You tried to overwrite a read-only file.
Remedy:	Change the status of the read-only file using the ATTRIB command and then re-enter the command.
MESSAGE:	All files cancelled by operator
Command:	PRINT
Cause:	This is an informational message displayed by PRINT when the /T (terminate queue) option is specified.

Remedy:	No action required.
MESSAGE:	All specified files are contiguous
Command:	CHKDSK
Cause:	This is an informational message displayed by CHKDSK. It indicates that all files are allocated contiguously on the disc and there is no fragmentation.
Remedy:	No action required.
MESSAGE:	Allocation error in file, size adjusted
Command:	CHKDSK
Cause:	This message is displayed by CHKDSK when the size of the file indicated in the directory is not consistent with the amount of data actually allocated to the file. The name of the file in error precedes this message.
Remedy:	<p>If the /F option was specified for CHKDSK, the file is truncated at the end of the last valid cluster.</p> <p>If the /F option was not specified, this is an informational message only, and no corrective action has been taken by CHKDSK. CHKDSK must be re-run with the /F option specified to correct the error.</p>
MESSAGE:	Are you sure (Y/N)?
Command:	DEL, ERASE, FORMAT
Cause:	This cautionary message is displayed if you instruct MS-DOS to delete all of the

files in a directory. This message is also displayed by **FORMAT** if you instruct it to format Drive C: or D: (the hard disc).

Remedy: Enter "Y" if you want to proceed or "N" if not.

MESSAGE: Bad command or file name

Command: MS-DOS

Cause: This message is displayed when MS-DOS is unable to find the program or command just entered.

Remedy: Re-enter the command with the correct spelling of the program or command, move a copy of the program or command file into an accessible directory, change the current directory to one that contains the file, or issue a **PATH** command to tell MS-DOS where to look for the command.

MESSAGE: Bad file

Command: FC

Cause: The message is displayed when FC finds a defect in one of the files specified.

Remedy: Run **CHKDSK** to verify the integrity of your disc.

MESSAGE: Bad or missing <filename>

Command: MS-DOS

Cause: This message is displayed when MS-DOS can't find the file containing the device driver (when it's specified in the

CONFIG.SYS file) in the root directory of the boot disc, or an error is detected in the driver as it's being loaded. The name of driver <filename> is displayed in the error message. In either case, the device driver is not incorporated into MS-DOS.

Remedy: Check the spelling of the filename in CONFIG.SYS or copy the file containing the device driver into the root directory of the boot disc.

MESSAGE: Bad or missing Command Interpreter

Command: MS-DOS

Cause: This message is displayed when MS-DOS can't find the COMMAND.COM file (or the file containing the alternate command processor specified in the CONFIG.SYS file using the SHELL command) in the root directory of the boot disc or if an error is encountered as the file is loaded.

Remedy: Copy the file containing the command processor into the root directory of the boot disc or check the spelling of the filename in the CONFIG.SYS file.

MESSAGE: BREAK is off (or on)

Command: MS-DOS

Cause: This is an informational message displayed by MS-DOS to inform you of the current setting of the extended **CTRL** **Break** and **CTRL** **C** checking.

Remedy: No action required.

MESSAGE:	Buffer size adjusted
Command:	VDISK
Cause:	This informational message is displayed when it is necessary to adjust the buffer size specified in the DEVICE=VDISK.SYS command in the CONFIG.SYS file.
Remedy:	Change buffer size if necessary.
MESSAGE:	Cannot CHDIR to <path> - tree past this point not processed
Command:	CHKDSK
Cause:	This message is displayed by CHKDSK if it is unable to reach the specified subdirectory. All subdirectories beneath this directory are not verified.
Remedy:	No action required.
MESSAGE:	Cannot CHDIR to root. Processing cannot continue
Command:	CHKDSK
Cause:	This message is displayed by CHKDSK if it is unable to return to the root directory while it is verifying the tree directory structure. CHKDSK is unable to continue checking the subdirectories in the root.
Remedy:	No action required.
MESSAGE:	Cannot CHKDSK a Network drive
Command:	CHKDSK

Cause: This message is displayed by CHKDSK if a drive is specified that is redirected over a network. CHKDSK can only operate on local disc drives.

Remedy: Do not specify a disc drive that is not local.

MESSAGE: Cannot COPY to (or from) a reserved device

Command: XCOPY

Cause: You can't use XCOPY to copy to or from a device (e.g., CON or PRN).

Remedy: Re-enter the command copying to or from a file instead of a device.

MESSAGE: Cannot do binary reads from a device

Command: COPY

Cause: This message is displayed by COPY when an attempt is made to copy a file from a device and the /B option is specified. This mode is invalid due to the fact that COPY needs to be able to detect the end-of-file marker from the device.

Remedy: Re-enter the command and omit the /B option or specify the /A option after the device name on the command line.

MESSAGE: Cannot edit .BAK file -- rename file

Command: EDLIN

Cause: This error message is displayed by EDLIN when the file you are trying to edit has an extension of .BAK.

Remedy: Rename the file with a different extension, or copy the file to a new file with an extension other than .BAK.

MESSAGE: Cannot find or execute d:FORMAT.COM

Command: SELECT

Cause: This message is displayed by SELECT when it is unable to execute the FORMAT command. Either the FORMAT command is missing or you do not have enough memory to load it.

Remedy: Make sure the FORMAT command is on the disc in the source drive. Also, make sure the source drive is the active drive (if the source disc is in drive A:, make sure the MS-DOS prompt is A>). If the FORMAT command is in a subdirectory, use the CHDIR command to make that subdirectory the current directory before you run SELECT.

MESSAGE: Cannot find or execute d:XCOPY.EXE

Command: SELECT

Cause: This message is displayed by SELECT when it is unable to execute the XCOPY command. Either the XCOPY command is missing or you do not have enough memory to load it.

Remedy: Make sure the XCOPY command is on the disc in the source drive. Also, make sure the source drive is the active drive (if the source disc is in drive A:, make sure the MS-DOS prompt is A>). If the XCOPY command is in a subdirectory, use the CHDIR command to make that

subdirectory the current directory before you run SELECT.

MESSAGE: Cannot format an Assigned drive

Command: FORMAT

Cause: This error message is displayed by **FORMAT** when a drive with an active assignment is specified as the target drive.

Remedy: Use the **ASSIGN** command to clear the assignment and re- enter the **FORMAT** command.

MESSAGE: Cannot format a Network drive

Command: FORMAT

Cause: This error message is displayed by **FORMAT** when a drive is specified that is redirected over a network.

Remedy: Do not specify a drive that is not local.

MESSAGE: Cannot open <filename>

Command: PRINT

Cause: This error message is displayed by **PRINT** when the specified file cannot be found in an accessible directory.

Remedy: Make sure that the filename is correctly entered on the command line, or move the file into an accessible directory.

MESSAGE: Cannot perform a cyclic copy

Command: XCOPY

Cause:	This message is displayed by XCOPY when you specify the /S switch and a target that is a subdirectory of the source.
Remedy:	Do not specify a target that is a subdirectory of the source when using the /S switch.
MESSAGE:	Cannot recover . entry, processing continued
Command:	CHKDSK
Cause:	This error message is displayed by CHKDSK when the active directory (indicated by the single period) is defective.
Remedy:	No action required.
MESSAGE:	Cannot recover a Network drive
Command:	RECOVER
Cause:	This error message is displayed by RECOVER when a drive is specified that is redirected over a network.
Remedy:	Do not specify a drive that is not local.
MESSAGE:	Cannot recover .. entry
Command:	CHKDSK
Cause:	This error message is displayed by CHKDSK when the parent directory (indicated by the two periods) is defective.
Remedy:	No action required.

MESSAGE: Compare error on side xx, track yy

Command: DISKCOMP

Cause: There are one or more locations on the indicated side and track of the two discs that contain differing information.

Remedy: No action required.

MESSAGE: Content of destination lost before copy

Command: COPY

Cause: This error message is displayed by COPY when the same file is specified as both the source and target drive. For example:

COPY FILE1+FILE2 FILE1

FILE1 is overwritten before it can be copied.

Remedy: Copy to a temporary file, delete the duplicate file, and rename the temporary file to the desired filename.

MESSAGE: Convert lost chains to files (Y/N)?

Command: CHKDSK

Cause: This error message is displayed by CHKDSK when it detects lost clusters or chains (multiple clusters).

Remedy: If you answer "Y", each chain is converted to a file with the filename FILEnnnn.CHK, where nnnn is a sequential number starting with 0000. If you answer "N", CHKDSK frees the lost

chains, and their space can be reallocated by MS-DOS. These actions only occur if the /F option is specified. If not, CHKDSK asks the question, but no action is taken, regardless of the answer.

MESSAGE: Copy Another (Y/N)?

Command: DISKCOPY

Cause: This message is displayed by DISKCOPY after it completes copying a disc.

Remedy: Answer "Y" if you want to copy another disc, "N" if not.

MESSAGE: Copy complete

Command: DISKCOPY

Cause: This is an informational message displayed by DISKCOPY after it completes copying a disc.

Remedy: No action required.

MESSAGE: Copy not completed

Command: DISKCOPY

Cause: This error message is displayed by DISKCOPY when it can't copy the entire disc. This error may be due to a defect on either the source or target disc.

Remedy: If the error is on the target disc, use a new disc which has no defects. If the error is on the source disc, use COPY to copy the files to another disc.

MESSAGE: Copying...

Command: DISKCOPY

Cause: This informational message is displayed by DISKCOPY to indicate that it is copy-ing a disc.

Remedy: No action required.

MESSAGE: Corrections will not be written to disk

Command: CHKDSK

Cause: This error message is displayed by CHKDSK when an error is detected on the disc and the /F option isn't specified.

Remedy: Run CHKDSK again with the /F option specified.

MESSAGE: Data left in <filename>

Command: FC

Cause: This informational message is displayed by FC if there is data left in one of the files being compared after the end of the other file is reached.

Remedy: No action required.

MESSAGE: Directory entries adjusted

Command: VDISK

Cause: This informational message is displayed when it is necessary to adjust the number of directory entries specified in the DEVICE=VDISK.SYS command in the CONFIG.SYS file.

Remedy:	Change the number of directory entries if necessary.
MESSAGE:	Directory is joined
Command:	CHKDSK
Cause:	This error message is displayed by CHKDSK if a directory on the disc being checked is joined.
Remedy:	Unjoin the directory and run CHKDSK again.
MESSAGE:	Directory not empty
Command:	JOIN
Cause:	This error message is issued by JOIN if an attempt is made to join to a directory which is not empty.
Remedy:	Join to a new directory which is empty, or empty the directory and rerun JOIN.
MESSAGE:	Directory is totally empty, no . or ..
Command:	CHKDSK
Cause:	This error message is displayed by CHKDSK if a subdirectory does not contain the parent or current directory entries.
Remedy:	Delete the subdirectory with RMDIR and recreate it using MKDIR.
MESSAGE:	Disk error reading FAT <x:>
Command:	CHKDSK



Cause: This error message is displayed by CHKDSK when one of the two File Allocation Tables for drive x: has a defective sector in it. MS-DOS automatically uses the good FAT.

Remedy: Copy the files to another disc and reformat the disc. If the sectors for the FAT are still defective, discard the disc.

MESSAGE: Disk error writing FAT <x:>

Command: CHKDSK

Cause: This error message is displayed by CHKDSK when one of the two File Allocation Tables for drive x: has a defective sector in it. MS-DOS automatically uses the good FAT.

Remedy: Copy the files to another disc and reformat the disc. If the sectors for the FAT are still defective, discard the disc.

MESSAGE: Disk full. Edits lost

Command: EDLIN

Cause: This error message is displayed by EDLIN when it is unable to save your file due to a lack of disc space.

Remedy: Delete enough files to make room for your document and re-enter it with EDLIN.

MESSAGE: Disk full--write not completed

Command: EDLIN

Cause: This error message is displayed by EDLIN

when it is unable to save your file due to a lack of disc space.

Remedy: Delete enough files to make room for your document and re-enter it with EDLIN.

MESSAGE: Disk unsuitable for system drive

Command: FORMAT

Cause: This error message is displayed by FORMAT if it detects a bad track on the disc where the system files should reside.

Remedy: This disc should only be used for data. Use another disc if you want to make a system disc.

MESSAGE: Diskette/Drive not compatible

Command: DISKCOMP

Cause: The discs being compared are of differing densities.

Remedy: Only compare discs of similar densities.

MESSAGE: Diskettes compare ok

Command: DISKCOMP

Cause: This informational message indicates the two discs contain identical information.

Remedy: No action required.

MESSAGE: Disks must be the same size.

Command: DISKCOPY

Cause: This error message is displayed if you attempt to use DISKCOPY with two discs with different formats and capacities.

Remedy: Use discs with the same format or use the COPY command to transfer files and SYS to transfer the system files (if desired).

MESSAGE: Divide overflow

Command: MS-DOS

Cause: This error message is displayed by MS-DOS if a program attempts to divide by 0 or a logic error causes an internal malfunction.

Remedy: Correct the error in the program. If the program was purchased, contact your dealer.

MESSAGE: [.][..] Does not exist

Command: CHKDSK

Cause: This error message is displayed by CHKDSK if either the . or .. entry in a subdirectory is invalid.

MESSAGE: Drive letter must be specified

Command: FORMAT

Cause: You are trying to format a disc without specifying the drive that contains the disc. This prevents you from accidentally formatting the disc in the active drive.

Remedy: Re-enter the command with a drive specified.

MESSAGE:	Drive types or diskette types not compatible
Command:	DISKCOPY
Cause:	This error message is displayed when you try to copy discs of differing densities.
Remedy:	Only copy discs of similar densities.
MESSAGE:	Drive x: not ready.
Command:	DISKCOMP, DISKCOPY
Cause:	One of the disc drives is not ready.
Remedy:	If there is a disc in the drive, open the door, reinsert the disc, and reclose the door.
MESSAGE:	Duplicate file name or File not found
Command:	RENAME
Cause:	Either the new filename already exists or the file being renamed doesn't exist.
Remedy:	Make sure the new filename doesn't already exist or make sure the file being renamed does exist.
MESSAGE:	End of input file
Command:	EDLIN
Cause:	This informational message is displayed by EDLIN when the entire file has been read into memory. If the file has been read in sections (using the Append command) this message is displayed when the last section is read.

Remedy:	No action required.
MESSAGE:	EOF mark not found
Command:	COMP
Cause:	This informational message is displayed by COMP when one or both of the files being compared do not contain end-of-file markers.
Remedy:	No action required.
MESSAGE:	Enter current Volume Label for drive <x:>
Command:	FORMAT
Cause:	FORMAT asks you to enter the current volume label for verification before it formats the hard disc in the specified drive.
Remedy:	Enter the current volume label for the hard disc. If you can't remember the current volume label, use the DIR command to display it and then re-enter the FORMAT command.
MESSAGE:	Entry error
Command:	EDLIN
Cause:	This error message is displayed by EDLIN when it detects a syntax error in a command.
Remedy:	Retype the command with the correct syntax.
MESSAGE:	Entry has a bad (attribute or link or size)

Command: CHKDSK

Cause: This error message is displayed by CHKDSK when an error condition is detected in one of the subdirectory entries. The message is preceded by one or two periods to indicate which entry.

Remedy: None. CHKDSK attempts to correct the error if the /F option is specified.

MESSAGE: Error in .EXE file

Command: MS-DOS

Cause: This error message is displayed by MS-DOS when the .EXE program it is attempting to load and execute has an invalid internal format.

Remedy: Relink program or make a new copy to execute. If you are using a purchased program and the error persists, contact your dealer.

MESSAGE: Error while copying files

Command: SELECT

Cause: This message is displayed by SELECT when it runs into an error while executing the XCOPY command. The target disc may be out of room or you may have specified an incorrect path.

Remedy: If the target disc doesn't have enough space for the files, rerun the SELECT command on a larger capacity blank disc (in a compatible drive). Also, make sure that you don't specify a path that incorporates the same name as one of the system files you are copying over.

MESSAGE: Error while copying KEYBxx.COM file

Command: SELECT

Cause: The target disc may be out of room, write protected, or have some other problem indicated by the message.

Remedy: Correct the problem according to the message displayed (if the target disc is out of room, use a higher capacity disc in a compatible drive), then re-enter the command.

MESSAGE: Error while formatting target disk

Command: SELECT

Cause: SELECT received an error while formatting the target disc. (The nature of the error should be listed in the error message.)

Remedy: Correct the problem listed in the error message and re-enter the command.

MESSAGE: Error while making config.sys and autoexec.bat

Command: SELECT

Cause: There probably is not enough room on the target disc.

Remedy: Try a larger capacity disc in a compatible drive. Then re-enter the command.

MESSAGE: Error writing to device

Command: MS-DOS

Cause: Too much data is being sent to a device. MS-DOS is unable to write the data to the specified device.

Remedy: Send less data to the device.

MESSAGE: Errors found, F parameter not specified
Corrections will not be written to disk

Command: CHKDSK

Cause: This error message is displayed by CHKDSK when it detects an error and the /F option is not specified. Any corrective action taken by CHKDSK is not written to disc.

Remedy: Rerun CHKDSK with the /F option specified in order to correct the errors.

MESSAGE: Errors on list device indicate that it may be off-line. Please check it.

Command: PRINT

Cause: This message is displayed by PRINT if a printer time-out error is detected. This error can also be displayed during a long printer operation such as a form feed. This message is displayed only when the PRINT command is executed.

Remedy: Turn the printer on or return it to an online status if appropriate. If the error occurs during a long printer operation, ignore the message.

MESSAGE: EXEC failure

Command: MS-DOS

Cause: This error message is displayed by MS-DOS when either an error is found while reading a command or the Files command in the CONFIG.SYS file is set too low.

Remedy: Check the integrity of the disc with the command file or increase the FILES value and restart MS-DOS.

MESSAGE: File allocation table bad

Command: MS-DOS

Cause: This error message is displayed by MS-DOS when a disc may be defective.

Remedy: Run CHKDSK to check the disc.

MESSAGE: File allocation table bad drive <x:>

Command: CHKDSK, PRINT

Cause: This error message is displayed by CHKDSK and PRINT when a disc may be defective.

Remedy: Run CHKDSK to check disc.

MESSAGE: File cannot be copied onto itself

Command: MS-DOS

Cause: This error message is displayed by MS-DOS when the source file has the same name as the target file. For example:

COPY FILE1 FILE1

Remedy: Specify a different name for the target file.

MESSAGE:	File cannot be converted
Command:	EXE2BIN
Cause:	This error message is displayed by EXE2BIN when the input file is not in the correct format.
MESSAGE:	File creation error
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS if you try to add a new file 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.
Remedy:	Run CHKDSK on the disc to determine the cause.
MESSAGE:	File is READ-ONLY
Command:	EDLIN
Cause:	This error message is displayed by EDLIN if you try to change a file that is designated as read-only.
MESSAGE:	<filename> cancelled by operator
Command:	PRINT
Cause:	This message is printed on the printer when you specify the /T switch in the PRINT command.
MESSAGE:	<filename> contains non-contiguous blocks
Command:	CHKDSK

Cause: This error message is displayed by CHKDSK when the file specified is not allocated contiguously on the disc.

Remedy: If you specify the /F switch, CHKDSK fixes this error.

MESSAGE: <filename> file not found

Command: PRINT

Cause: This error message is displayed by PRINT if discs are switched while a file is queued up, but before it starts to print.

Remedy: Re-enter the PRINT command for that file.

MESSAGE: <filename> is cross linked on cluster

Command: CHKDSK

Cause: This error message is displayed by CHKDSK when two or more files are cross linked.

Remedy: Make a copy of the file you want to keep, and then delete the files that are cross linked.

MESSAGE: <filename> is currently being printed

Command: PRINT

Cause: This error message is displayed by PRINT when the file specified is being printed.

Remedy: No action required.

MESSAGE: <filename> is in queue

Command:	PRINT
Cause:	This error message is displayed by PRINT when the file specified is waiting to be printed.
Remedy:	No action required.
MESSAGE:	Filenames must be specified
Command:	EDLIN
Cause:	This error message is displayed by EDLIN if you don't specify a file at the time you start EDLIN .
Remedy:	Specify a file to edit on the command line to start EDLIN .
MESSAGE:	File not found
Command:	EDLIN, EXE2BIN, FC, FIND, MS-DOS, RECOVER, REN
Cause:	This error message is displayed when MS-DOS cannot find the file that you specified.
Remedy:	Make sure that the path is accurate and the file exists in the directory you specified.
MESSAGE:	to (or from) File not found
Command:	XCOPY
Cause:	This message is displayed when MS-DOS cannot find the file that you specified.
Remedy:	Make sure that the path is correct and

the file exists in the directory you specified.

MESSAGE: Files are different

Command: FC

Cause: This error message is displayed by FC after it detects a number of differences and gives up attempting to compare the files.

Remedy: Edit the files to remove the reported differences and try again.

MESSAGE: Files are different sizes

Command: COMP

Cause: This informational message is displayed by COMP when the two files are different sizes. The compare operation is terminated.

Remedy: No action required.

MESSAGE: Find: File not found <filename>

Command: FIND

Cause: This error message is displayed by FIND when you specify a file that doesn't exist.

Remedy: Make sure you type the filename correctly.

MESSAGE: Find: Invalid number of parameters

Command: FIND

Cause: This error message is displayed by FIND if you specify too many or too few options on the command line.

MESSAGE:	Find: Invalid Parameter
Command:	FIND
Cause:	This error message is displayed by FIND if one of the options you specified is wrong.
Remedy:	Re-enter the command correctly.
MESSAGE:	Find: Read error in <filename>
Command:	FIND
Cause:	This error message is displayed by FIND if the program can't read the specified file.
Remedy:	Run CHKDSK to ensure the integrity of the disc and try again.
MESSAGE:	Find: Syntax error
Command:	FIND
Cause:	This error is displayed by FIND if the command is typed incorrectly.
Remedy:	Re-enter the command correctly.
MESSAGE:	First cluster number is invalid, entry truncated
Command:	CHKDSK
Cause:	This error message is displayed by CHKDSK if the file directory entry contains an invalid pointer to the data area. If you specified the /F option, the file is truncated to a zero length file.

Remedy: No action required.

MESSAGE: Fixups needed - base segment <hex:>

Command: EXE2BIN

Cause: This error message is displayed by EXE2BIN if the source (.EXE) file contains information indicating that a load segment is required for the file.

Remedy: Specify the absolute segment address at which the finished module is to be located.

MESSAGE: For cannot be nested

Command: BATCH PROCESSOR

Cause: This error message is displayed by MS-DOS when FOR commands are nested in a batch file.

Remedy: Edit the batch file to remove the nested FOR commands.

MESSAGE: Format failure

Command: FORMAT

Cause: This error message is displayed when MS-DOS can't format a disc. The message is usually displayed with an explanation as to why MS-DOS can't format the disc.

Remedy: Restart the FORMAT command.

MESSAGE: Formatting While Copying

Command: DISKCOPY

Cause:	This informational message is displayed when the target disc is not recognized as a formatted disc. DISKCOPY formats it while it copies from the source disc.
Remedy:	No action required.
MESSAGE:	Has invalid cluster, file truncated
Command:	SYS
Cause:	This error message is displayed by SYS when the system files IBMBIO.COM and IBMDOS.COM occupy more space on the source disc than is available on the target disc.
Remedy:	No action required.
MESSAGE:	Incorrect DOS version
Command:	MS-DOS
Cause:	This error message is displayed because many version 3.20 commands and utilities won't run on older versions of MS-DOS . Most commands and utilities only run under the exact version of MS-DOS for which they are configured.
MESSAGE:	Incorrect number of parameters
Command:	JOIN, SUBST
Cause:	This error message is displayed by JOIN or SUBST when you specify too many or too few options on the command line.
Remedy:	Re-enter the command correctly.

MESSAGE:	Incorrect number of parameters
Command:	SELECT
Cause:	You entered too few or too many parameters. (The valid number of parameters is from 2 to 4).
Remedy:	Re-enter the command with the correct number of parameters.
MESSAGE:	Incorrect parameter
Command:	ASSIGN, SELECT, SHARE
Cause:	This error message is displayed by ASSIGN, SELECT or SHARE when one of the options you specified, or the syntax you used, was wrong.
Remedy:	Re-enter the command correctly.
MESSAGE:	Insert diskette for drive <x:> and strike any key when ready
Command:	MS-DOS
Cause:	This error message is displayed when MS-DOS is copying and formatting.
Remedy:	Insert a disc in the appropriate drive and press any character or number key to begin processing.
MESSAGE:	Insert diskette with batch file and press any key when ready
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS when the disc containing the batch file you specified is not in the drive you originally specified.

Remedy:	Reinsert the disc that contains the batch file in the appropriate drive.
MESSAGE:	Insert diskette with COMMAND.COM in drive x and strike a key when ready
Command:	DISKCOMP, DISKCOPY
Cause:	MS-DOS is trying to load the command processor COMMAND.COM, but it isn't in the drive from which you started your system.
Remedy:	Insert the MS-DOS disc in the indicated drive and press any key.
MESSAGE:	Insert DOS diskette into drive <x:> and strike any key when ready
Command:	FORMAT
Cause:	This error message is displayed by FORMAT when you specify the /S option and the disc in the default drive does not contain the MS-DOS system files.
Remedy:	Insert the disc the contains the IBMBIO.COM and IBMDOS.COM files in the specified drive.
MESSAGE:	Insert new diskette for drive <x:> and strike any key when ready
Command:	FORMAT
Cause:	This error message is displayed by FORMAT when you need to insert a blank disc into the appropriate drive.

Remedy:	Press any character or number key to begin formatting. If there is any data on the disc, it is destroyed by the format process.
MESSAGE:	Insert system diskette in drive <x:> and strike any key when ready.
Command:	SYS
Cause:	This error message is displayed when SYS needs a disc from which to read the IBMBIO.COM and IBMDOS.COM files into memory.
Remedy:	Press any character or number key to start the system copy process.
MESSAGE:	Insert target diskette into drive <x:>
Command:	DISKCOPY
Cause:	This error message is displayed if you are running DISKCOPY and your source and target drives are the same.
Remedy:	Reinsert the target disc into the specified drive.
MESSAGE:	Insufficient disk space
Command:	EXE2BIN, MS-DOS, REPLACE, SORT, XCOPY
Cause:	This error message is displayed when the disc is full. It does not contain enough room to perform the specified operation.
Remedy:	Delete non-essential files from the disc and try again, or, in the case of XCOPY, use a higher capacity disc in a compatible drive.

MESSAGE:	Insufficient memory
Command:	REPLACE, XCOPY
Cause:	This message is displayed when there is not enough computer base memory to run the program.
Remedy:	REPLACE or XCOPY fewer files at a time, or one-at-a-time.
MESSAGE:	Insufficient memory for system transfer Processing cannot continue
Command:	FORMAT
Cause:	This error message is displayed by FORMAT when the memory configuration is insufficient to transfer the MS-DOS system files IBMBIO.COM and IBMDOS.COM (the /S option).
Remedy:	Reboot the system with fewer device drivers and virtual discs, or more memory.
MESSAGE:	Insufficient room in root directory. Erase files in root and repeat CHKDSK
Command:	CHKDSK
Cause:	This error message is displayed by CHKDSK. CHKDSK always recovers lost files into the root directory. In this case, your root directory is full.
Remedy:	Delete some files in your root directory to make room for the lost files.
MESSAGE:	Intermediate file error during pipe



Command: MS-DOS

Cause: This error message is displayed by MS-DOS when the piping process makes use of temporary files on the disc which are automatically deleted after the piping process is complete. An error has occurred in one of these files.

Remedy: Check the integrity of your disc with CHKDSK. Then, try again with a new disc.

MESSAGE: Internal error

Command: FC

Cause: This error message is displayed by FC. It indicates an internal logic error in the FC utility.

Remedy: Use COMP to compare your files.

MESSAGE: Internal Stack Failure

Command: MS-DOS

Cause: The available stack resources have been exceeded due to a rapid succession of hardware interrupts.

Remedy: Increase the stack resources by including the STACKS command in your CONFIG.SYS file.

MESSAGE: Invalid baud rate specified

Command: MODE

Cause: This error message is displayed by MODE if a baud rate other than 110, 150, 300,

600, 1200, 2400, 4800, 9600, or 19200 is specified.

Remedy: Re-enter the command with a valid baud rate.

MESSAGE: Invalid characters in volume label

Command: FORMAT

Cause: This error message is displayed by FORMAT when the volume label contains too many characters or invalid characters.

Remedy: Use the LABEL command to ensure the volume label contains no more than 11 alphanumeric characters.

MESSAGE: Invalid COMMAND.COM. Insert COMMAND.COM disk in default drive and strike any key when ready

Command: MS-DOS

Cause: This error message is displayed by MS-DOS when the program you just ran used up almost all of the memory. MS-DOS must now reload the COMMAND.COM file from disc. However, MS-DOS cannot find COMMAND.COM on the disc or the copy found is invalid.

Remedy: Insert a disc into the default drive which contains a copy of COMMAND.COM similar to the version on the disc with which you started MS-DOS.

MESSAGE: Invalid country code

Command:	COUNTRY, SELECT
Cause:	This error message is displayed when you specify a country code that is not on the list of valid country codes for this version of MS-DOS.
Remedy:	See the COUNTRY or SELECT command for a list of valid country codes.
MESSAGE:	Invalid date
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS when an invalid date is specified in response to the date prompt when starting MS-DOS.
Remedy:	Retype the correct date.
MESSAGE:	Invalid device
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS when the device specified is not CON, NUL, AUX, or PRN.
Remedy:	Use a valid device name.
MESSAGE:	Invalid directory
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS when the directory specified either does not exist or is invalid.
Remedy:	Check to see that you entered the directory name correctly.

MESSAGE:	Invalid drive in search path
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS when an invalid drive is detected in the PATH string.
Remedy:	Issue a new PATH command.
MESSAGE:	Invalid drive or filename
Command:	EDLIN, RECOVER
Cause:	This error message is displayed by EDLIN or RECOVER when a valid drive or a valid filename needs to be specified.
Remedy:	Specify a valid drive or filename.
MESSAGE:	Invalid drive specification
Command:	CHKDSK, FORMAT, REPLACE, SYS, XCOPY
Cause:	This error message is displayed when a valid drive needs to be specified.
Remedy:	Specify a valid drive.
MESSAGE:	Invalid drive specification. Specified drive does not exist, or is non-removable.
Command:	DISKCOPY
Cause:	This error message is displayed when a drive designator is specified for a drive that doesn't exist or is non-removable.

Remedy:	Re-enter the command with a drive that exists and is removable.
MESSAGE:	Invalid keyboard code
Command:	SELECT
Cause:	You specified an invalid keyboard code.
Remedy:	Re-enter the command with a valid keyboard code. Refer to the SELECT command description in this manual for a list of valid keyboard codes.
MESSAGE:	Invalid number of parameters
Command:	FC, FIND, RECOVER, XCOPY
Cause:	This error message is displayed when the wrong number of options are specified on the command line.
Remedy:	Re-enter the command with the correct number of options.
MESSAGE:	Invalid parameter
Command:	CHKDSK, EDLIN, FC, FIND, FORMAT, PRINT, REPLACE, SELECT, XCOPY
Cause:	This error message is displayed when one of the options specified is wrong.
Remedy:	Check the correct syntax and re-enter the command.
MESSAGE:	Invalid parameter. Do not specify filename(s). Command Format: DISKCOMP d: d: [/1][/8]
Command:	DISKCOMP

Cause:	This error message is displayed when one or more filenames are specified. DISKCOMP only accepts drive designators.
Remedy:	Re-enter the command with the correct parameters.
MESSAGE:	Invalid parameter. Do not specify filename(s). Command Format: DISKCOPY d: d: [/1]
Command:	DISKCOPY
Cause:	This error message is displayed when one or more filenames are specified. DISKCOPY only accepts drive designators.
Remedy:	Re-enter the command with the correct parameters.
MESSAGE:	Invalid parameter. The last two parameters must be country code and keyboard code.
Command:	SELECT
Cause:	You entered parameters out of order.
Remedy:	Re-enter the command using the correct syntax and order.
MESSAGE:	Invalid path
Command:	XCOPY
Cause:	You specified a source path that lists a subdirectory that doesn't exist.

Remedy:	Specify a source path that lists a sub-directory that does exist.
MESSAGE:	Invalid path or file name
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS when a valid path or filename needs to be specified for the COPY command.
Remedy:	Re-enter the command with a valid path or filename.
MESSAGE:	Invalid path or path too long.
Command:	SELECT
Cause:	You specified a path which contains invalid characters or is longer than 64 characters.
Remedy:	Re-enter the command using only valid characters (see section in this manual on <i>Invalid Characters</i>) and a shorter path.
MESSAGE:	Invalid path, not directory, or directory not empty
Command:	RMDIR
Cause:	This error message is displayed by RMDIR when you are unable to remove the directory requested for one of the specified reasons.
Remedy:	Make sure the directory actually exists and is empty. Then, re-enter the command.

MESSAGE: Invalid source drive, must be A or B

Command: SELECT

Cause: You specified an invalid source drive. Drives A: and B: are the only valid source drives for this command.

Remedy: Re-enter the command with a valid source drive.

MESSAGE: Invalid sub-directory entry

Command: CHKDSK

Cause: This error message is displayed by CHKDSK when the subdirectory specified either does not exist or is invalid.

Remedy: Make sure you typed the subdirectory name correctly.

MESSAGE: Invalid switch character

Command: VDISK

Cause: This error message is displayed if the character following the "/" on the command line isn't "E". VDISK attempts to install the virtual disc in low memory.

Remedy: Re-enter the command in the CONFIG.SYS file.

MESSAGE: Invalid target drive

Command: SELECT

Cause: You specified an invalid or non-existent target drive.

Remedy: Re-enter the command using a valid drive designator (A:, B:, C:, etc.).

MESSAGE: Invalid time

Command: MS-DOS

Cause: This error message is displayed when you specify an invalid time in response to the time prompt when starting MS-DOS.

Remedy: Enter a valid time in the correct (24-hour) format.

MESSAGE: Invalid working directory

Command: MS-DOS

Cause: This error message is displayed by MS-DOS when your disc is bad.

Remedy: Replace the disc or make another copy from your backup system disc.

MESSAGE: Is KEYBxx.COM available on another disk? (Y/N)

Command: SELECT

Cause: You specified a non-U.S. keyboard and the driver for that keyboard is NOT on the disc in the source drive.

Remedy: If you have the driver on another disc, enter "Y" and wait until you are prompted to insert the disc. If you don't have the driver, enter "N" and the program will abort. Refer to the SELECT command description in this manual for a list of the available keyboard codes.

MESSAGE:	Label not found
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS when there is a GOTO command to a nonexistent label in a batch file.
Remedy:	Edit the batch file to add the label or remove the reference to the label.
MESSAGE:	Line too long
Command:	EDLIN
Cause:	This error message is displayed by EDLIN during a REPLACE command. The string given as the replacement causes the line to exceed 253 characters in length.
Remedy:	Divide the long line into two lines and retry the REPLACE command.
MESSAGE:	output is not assigned to a device
Command:	PRINT
Cause:	This error message is displayed by 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 device that doesn't exist.
Remedy:	Specify a valid device name.
MESSAGE:	Memory allocation error
Command:	MS-DOS

Cause:	A program has written into "free" memory, which prevents MS-DOS from allocating that memory. This error is usually fatal.
Remedy:	When this error message is displayed, restart MS-DOS. If this error persists, make a new copy of the MS-DOS disc from your backup copy of the system disc. If this error occurs consistently with purchased software, contact your dealer.
MESSAGE:	--More--
Command:	MORE
Cause:	The program that piped its output to MORE has produced more than one screen of data for MORE to print.
Remedy:	Press the spacebar to view more of the file or directory.
MESSAGE:	MORE: Incorrect DOS version
Command:	MORE
Cause:	This error message is displayed because MORE won't run on versions of MS-DOS prior to 3.20.
Remedy:	Use the correct versions of MORE and MS-DOS.
MESSAGE:	Must specify destination line number
Command:	MORE
Cause:	This error message is displayed by MORE when you need to specify a destination

	line number when copying and inserting lines with EDLIN.
Remedy:	Re-enter the command with the destination line.
MESSAGE:	Must specify ON or OFF
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS when the command requires either an ON or OFF argument.
Remedy:	Re-enter the command with ON or OFF.
MESSAGE:	Name of list device (PRN):
Command:	PRINT
Cause:	This prompt appears the first time PRINT is run. Any valid device can be specified; that device then becomes the PRINT output device.
Remedy:	Specify the device you want to use to print.
MESSAGE:	New file
Command:	EDLIN
Cause:	This message is displayed if EDLIN can't find an existing file with the name you specified.
Remedy:	If you want to create a new file, ignore this message. If you don't want to create a new file, make sure you correctly type the name of the file you want to edit.

MESSAGE: No Drive Specified

Command: DEVICE

Cause: This message is displayed if you enter the command `DEVICE=DRIVER.SYS` and you don't specify a physical drive number.

Remedy: Re-enter the command and specify a physical drive number.

MESSAGE: No files found <filename>

Command: REPLACE

Cause: This message is displayed by `REPLACE` if it can't find matching source or target files.

Remedy: Re-enter the command with the correct source or target filenames.

MESSAGE: No files match d:xxxxxxx.xxx

Command: PRINT

Cause: This error message is displayed by `PRINT` when you want to add a file to the queue, but no filename matches the specification.

Remedy: Re-enter the command with a valid filename.

MESSAGE: No free file handles. Cannot start COMMAND.COM, exiting

Command: MS-DOS

Cause: `COMMAND.COM` is unable to open a file because there aren't enough file handles available on the system.

Remedy:	Increase the value of the FILES command in the CONFIG.SYS file. Then, restart MS-DOS .
MESSAGE:	No path
Command:	MS-DOS
Cause:	This message is displayed by MS-DOS when you enter the PATH command without parameters to determine the current search path and there isn't a current search path.
Remedy:	No action required.
MESSAGE:	No room for system on destination disk
Command:	SYS
Cause:	This error message is displayed by SYS when there is not enough room for the system files on the target disc.
Remedy:	Delete some files to make room for the system files or use another disc. You may need to reformat the disc to put the system files on it.
MESSAGE:	No room in directory for file
Command:	EDLIN
Cause:	This error message is displayed by EDLIN when you try to save a file in the root directory and it is full. Subdirectories are not limited in size as is the root directory.
Remedy:	Delete extraneous files from the root directory, or specify a filename in a

subdirectory and re-enter your text into EDLIN.

MESSAGE: Not enough memory

Command: JOIN, SHARE, SUBST

Cause: This error message is displayed by JOIN, SHARE or SUBST when there is not enough memory for MS-DOS to run the command.

Remedy: Reduce the number of installed drives and virtual discs, or acquire more memory for your system.

MESSAGE: Not enough room to merge the entire file

Command: EDLIN

Cause: This error message is displayed by EDLIN if there is not enough room in memory to hold the file during a Transfer command.

Remedy: Free some memory by writing some files to disc. Then, re-enter the Transfer command.

MESSAGE: Not found

Command: EDLIN

Cause: This error message is displayed by EDLIN if you enter a Search or Replace command that is unable to find a further occurrence of the specified Search or Replace string.

Remedy: No action required.

MESSAGE:	O.K.?
Command:	EDLIN
Cause:	This prompt occurs during Search and Replace command processing.
Remedy:	Press any key except [V] or [Enter] to continue the Search or Replace process.
MESSAGE:	Out of environment space
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS when there isn't enough room in the program environment to accept more data.
Remedy:	Remove unnecessary strings from the environment. Then, re-enter the command.
MESSAGE:	Parameters not compatible
Command:	REPLACE
Cause:	This message is displayed when you enter invalid or incorrect parameters.
Remedy:	Re-enter the command using the correct syntax and valid parameters.
MESSAGE:	Path not found
Command:	CHKDSK, REPLACE, XCOPY
Cause:	This error message is displayed when you specify an invalid path.
Remedy:	Make sure you enter the path correctly and the path exists.



MESSAGE: Path too long

Command: REPLACE, XCOPY

Cause: The specified path is too long.

Remedy: Change subdirectories to REPLACE or XCOPY files in lower level subdirectories.

MESSAGE: Port not installed

Command: MODE

Cause: This error message is displayed if the port specified (LPTn: or COMn:) is not present on the system.

Remedy: Specify a port that exists on the system and re-enter command line.

MESSAGE: Press any key to begin formatting <x:>

Command: FORMAT

Cause: This prompt is issued before you format a disc.

Remedy: Press any character or number to begin the format process. To end this command, press **CTRL** **C**.

MESSAGE: Press any key to begin recovery of the <xxx> file(s) on drive <x:>

Command: RECOVER

Cause: This prompt is issued before you recover a disc or file.

Remedy: Press any character or number to begin

the recover process. To end this command, press **CTRL** **C**.

MESSAGE: PRINT queue is empty

Command: PRINT

Cause: This error message is displayed by PRINT when there aren't any files waiting to be printed.

Remedy: No action required.

MESSAGE: PRINT queue is full

Command: PRINT

Cause: This error message is displayed by PRINT when there is no room in the list of files waiting to be printed.

Remedy: Wait until some of the waiting files are printed, or remove them from the queue. Re-enter the PRINT command.

MESSAGE: Probable non-DOS disk. Continue (Y/N)?

Command: CHKDSK

Cause: This error message is displayed by CHKDSK when the disc you are using is not recognized by this version of MS-DOS. Either the disc was created by another system with a format that is not supported on this version of MS-DOS or it is not an MS-DOS disc.

Remedy: If the /F option was used, re-enter the command without it. The possible corrections are displayed. Then,

RECOVER the disc with the /F option or reFORMAT it.

MESSAGE: Processing cannot continue

Command: CHKDSK

Cause: This error message is displayed by CHKDSK when there is not enough memory in your system to process CHKDSK for this disc.

Remedy: Remove some drivers or virtual discs, or obtain more memory for your system.

MESSAGE: Program too big to fit in memory

Command: MS-DOS

Cause: This error message is displayed by MS-DOS when you must acquire more memory to run your application. It is possible that some programs you have run are still using some memory.

Remedy: You can try to restart MS-DOS; however, if you still receive this message, you must acquire more memory.

MESSAGE: Read error in <filename>

Command: FC, FIND

Cause: This error message is displayed by FC or FIND when MS-DOS cannot read the file.

Remedy: Run CHKDSK to ensure the integrity of the files. RECOVER them, if necessary.

MESSAGE: Resident part of PRINT installed

Command:	PRINT
Cause:	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.
Remedy:	No action required.
MESSAGE:	Reinsert diskette for drive x
Command:	FORMAT
Cause:	This message is displayed by FORMAT when you need to re-insert the disc being formatted in the indicated drive.
Remedy:	Insert the disc to be formatted in the specified drive.
MESSAGE:	Sector size adjusted
Command:	VDISK
Cause:	This informational message is displayed when it is necessary to adjust the sector size specified in the DEVICE=VDISK.SYS command in the CONFIG.SYS file.
Remedy:	Change sector size, if necessary.
MESSAGE:	Sector size too large in file <filename>
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS when 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.

Remedy: No action required.

MESSAGE: SHARE already installed

Command: SHARE

Cause: This error message is displayed if you try to install SHARE more than once.

Remedy: No action required.

MESSAGE: Sort: Incorrect DOS version

Command: SORT

Cause: This error message is displayed if you try to run SORT on any version of MS-DOS other than 3.20.

Remedy: Use the correct versions of SORT and MS-DOS.

MESSAGE: SORT: Insufficient disk space

Command: SORT

Cause: This error message is displayed by SORT when the disc is full.

Remedy: Remove some extraneous files from the disc and try again.

MESSAGE: SORT: Insufficient memory

Command: SORT

Cause: This error message is displayed when

there is not enough memory to run the
SORT command.

Remedy: Reduce the amount of data to be sorted
and try again.

MESSAGE: Source and target diskettes are not the
same format

Command: DISKCOPY

Cause: This error message is displayed by
DISKCOPY when the disc types differ.
Example: you cannot copy from a single-
sided disc to a double- sided disc.

Remedy: Reformat the target disc so that it is the
same type as the source disc.

MESSAGE: SOURCE diskette bad or incompatible

Command: DISKCOMP

Cause: One of the discs is unformatted or in-
compatible with the drive.

Remedy: Use a properly formatted disc.

MESSAGE: Source drive must be different from tar-
get drive. If no source drive is specified,
it defaults to A:

Command: SELECT

Cause: You specified the same drive designator
for both the source and target drives.
The source and target drives must be
different.

Remedy: Re-enter the command using the correct
syntax (only drive A: or drive B: can be

the source drive). Note that if you do not enter a source drive, **SELECT** automatically assumes the source drive is drive A:. Therefore, the command line **SELECT A: 1 US** would result in an error.

MESSAGE: Source path required

Command: **REPLACE**

Cause: This message is displayed by **REPLACE** when the correct source path isn't specified.

Remedy: Re-enter the command using the correct source path.

MESSAGE: Specified MS-DOS search directory bad

Command: **MS-DOS**

Cause: This error message is displayed by **MS-DOS** because the **SHELL** command in the **CONFIG.SYS** file is incorrect. The place that you told **MS-DOS** to find **COMMAND.COM** does not exist or **COMMAND.COM** is not in that place.

Remedy: Ensure that the **SHELL** command is started from the root directory.

MESSAGE: Strike a key when ready...

Command: **MS-DOS**

Cause: This prompt occurs during command processing and is always accompanied by another message. This message is also displayed if you insert a **PAUSE** command in a batch file. Usually, you are asked to

insert discs into appropriate drives before this prompt.

Remedy: Perform the specified task, then press any alphanumeric key.

MESSAGE: Syntax error

Command: FIND

Cause: You have typed a command line that FIND cannot interpret.

Remedy: Check to make sure you typed the command line correctly.

MESSAGE: System transferred

Command: FORMAT, SYS

Cause: This error message is displayed by FORMAT and SYS when the system files IBMDOS.COM and IBMBIO.COM are transferred during FORMAT and SYS command processing.

Remedy: No action required.

MESSAGE: Target diskette is write protected
Correct, then strike any key

Command: DISKCOPY

Cause: This error message is displayed when the target disc has a write protect tab covering the write-enable notch.

Remedy: Remove the tab and reinsert the disc.

MESSAGE: Target Diskette may be unusable

Command:	DISKCOPY
Cause:	This error message is displayed because of read, write, or verify errors in the copying process. The target disc may be incomplete.
Remedy:	Compare the two discs with DISKCOMP . Try again. A new target disc may be necessary.
MESSAGE:	Terminate batch job (Y/N)?
Command:	MS-DOS
Cause:	This error message is displayed by MS-DOS if you press CTRL C while in batch mode. MS-DOS asks you whether or not you want to end batch processing.
Remedy:	Press "Y" to end processing. Press "N" to continue batch processing.
MESSAGE:	The file is not in the root directory of this disk.
Command:	SELECT
Cause:	This message is displayed by SELECT when it can't find the keyboard driver in the root directory of the source disc.
Remedy:	Make sure the keyboard driver you want is in the root directory of the source disc by using the DIR command. If the driver you want is in a subdirectory, copy it to the root directory and re-enter the command.
MESSAGE:	Too many files open

Command:	EDLIN
Cause:	This error message is displayed by EDLIN when MS-DOS cannot open the file to edit on the .BAK file due to lack of system file handles.
Remedy:	Increase the value of the FILES command in the CONFIG.SYS file.
MESSAGE:	Too many open files
Command:	XCOPY
Cause:	This error message is displayed when MS-DOS cannot open the XCOPY command file due to a lack of system file handles.
Remedy:	Increase the value of the FILES command in the CONFIG.SYS file.
MESSAGE:	Track 0 bad - disk unusable
Command:	FORMAT
Cause:	The FORMAT command can accommodate defective sectors on the disc <i>except</i> for those near the beginning.
Remedy:	Try formatting another disc.
MESSAGE:	Unable to create a directory
Command:	MS-DOS, MKDIR
Cause:	This error message is displayed when MS-DOS cannot create the directory you specified.
Remedy:	Check to see that there is not a name

conflict (you may have a file by the same name). Or, the disc may be full.

MESSAGE: Unable to create directory

Command: XCOPY

Cause: This message is displayed by XCOPY when MS-DOS can't create the directory you specified.

Remedy: Check for a filename conflict (you may have a file with the same name you want to use for your directory). Or, the disc may be full, in which case you can either delete files no longer needed or use a higher capacity disc in a compatible drive.

MESSAGE: Unexpected DOS Error <NNN>

Command: REPLACE

Cause: An unexpected error occurred. <NNN> is the MS-DOS error number.

MESSAGE: Unrecognized command in CONFIG.SYS

Command: MS-DOS

Cause: This error message is displayed by MS-DOS when there is an invalid command in your CONFIG.SYS file.

Remedy: Refer to the chapter entitled "System Configuration" for a list of valid commands.

MESSAGE: Unrecoverable error in Directory Convert directory to file (Y/N)?

Command:	CHKDSK
Cause:	This error message is displayed by CHKDSK when there is an unrecoverable error in a directory.
Remedy:	If you respond "Y" to this prompt, CHKDSK converts the bad directory into a file. You can then fix the directory yourself or delete it.
MESSAGE:	Unrecoverable Read Error on Drive x: Side y, Track z
Command:	DISKCOMP, DISKCOPY
Cause:	This error message is displayed after several unsuccessful attempts are made to read the data from the specified track and sector on the disc in the indicated drive.
Remedy:	The disc is bad. RECOVER as many files as possible, copy the files to another disc, and try reformatting.
MESSAGE:	Unrecoverable Write Error on Drive x: Side y, Track z
Command:	DISKCOPY
Cause:	This error message is displayed after several unsuccessful attempts are made to write data to the specified track and sector on the disc in the indicated drive.
Remedy:	The disc is bad. RECOVER as many files as possible, copy the files to another disc, and try reformatting.
MESSAGE:	VDISK not installed - buffer too small

Command:	VDISK
Cause:	This error message is displayed when the virtual disc drive cannot be installed due to an incorrect buffer size.
Remedy:	Change the buffer size, if necessary.
MESSAGE:	VDISK not installed - insufficient memory
Command:	VDISK
Cause:	This error message is displayed when the virtual disc drive cannot be installed due to insufficient memory. If less than 64K of system memory would be left after the virtual disc is installed, this message is displayed.
Remedy:	Change the buffer size, if necessary.
MESSAGE:	VDISK not installed - no extended memory
Command:	VDISK
Cause:	This error message is displayed when the /E option is specified, but the system does not have extended memory, or the amount of available extended memory is insufficient to contain the virtual disc even after adjusting the parameters.
Remedy:	Ensure that you have enough extended memory to run VDISK, or reduce the size of the VDISK.
MESSAGE:	Volume label (11 characters, ENTER for none)?

Command:	FORMAT
Cause:	This message is displayed when you specify the /V option in the FORMAT command.
Remedy:	Specify a volume label or press Enter to indicate that you do not want a volume label for the disc.
MESSAGE:	WARNING, ALL DATA ON NON-REMOVABLE DISK DRIVE X WILL BE LOST! Proceed with Format (Y/N)?
Command:	FORMAT
Cause:	This error message is displayed by FORMAT when there is data on the hard disc that you are trying to format.
Remedy:	If you want to erase the data and format the disc, press "Y" (Yes). If you do not want the files on your hard disc erased, press "N" (No), copy the files to a flexible disc, and then repeat the FORMAT command.
MESSAGE:	Warning - directory full
Command:	RECOVER
Cause:	This error message is displayed by RECOVER when the root directory is too full for RECOVER processing.
Remedy:	Delete some files in the root directory to free space.
MESSAGE:	Warning: Read error in EXE file

Command: EXE2BIN

Cause: This error message is displayed by EXE2BIN when the amount read is less than the size of the header. This is a warning message only.

Remedy: No action required.

MESSAGE: <xxxx> of <xxxx> Bytes recovered

Command: RECOVER

Cause: This error message is displayed by RECOVER. It tells you how many bytes MS-DOS was able to recover of the disc or file.

MESSAGE: 10 Mismatches - ending compare

Command: COMP

Cause: This informational message is displayed by COMP when 10 mismatches are found. The compare operation is terminated.



B

The MS-DOS Keyboard

The MS-DOS keyboard has many features which are designed to make your computer easier and faster to use. These features allow you to edit an MS-DOS command line and perform other useful functions. This chapter explains how to use these features.

Editing a Command Line

MS-DOS provides a set of editing keys which can assist in entering command lines. The MS-DOS editing keys are

, , , , , , and .

The heart of this editing capability is the Template and the Template Pointer. Each time a command line is entered, MS-DOS stores a copy of it in the template, and sets the template pointer to point to the first character in the template. Characters can then be easily copied from the template pointer to create a new command line. For example, when the command line:

```
C>COPY *.TXT B:
```

is entered, the template contains:

```
COPY *.TXT B:
```

The template can now be edited to create the next command line. The editing keys and their respective functions are listed on the following pages.

F1 or
→ Copies the character pointed to by the template pointer to the command line each time it is pressed. The template pointer is then advanced one character in the template. For example, if the template contains the command COPY *.TXT B: and the current command line is:

A>

after **F1** is pressed twice, the command line is:

A>CO

F2 Copies characters from the template to the current command line, starting at the template pointer, until a specified character is found in the template. For example, if the template contains the command COPY *.TXT B: and you press:

F2 *

COPY is added to the current command line.

F3 Moves the contents of the template to the current command line, starting at the template pointer, until the last character in the template is reached. For example, if the template contains the command COPY *.TXT B: and the current command line is:

C>

after **F3** is pressed, the command line is:

C>COPY *.TXT B:

F4

Moves the template pointer forward so that it points to the next occurrence of the specified character. If the character isn't found, the template pointer isn't moved. This makes it easy to skip over large sections of the template. For example, if the template contains the command `COPY *.TXT B:` and you press:

F4 *

the template pointer points to the *.

F5

Moves the contents of the current command line to the template. The previous contents of the template are lost. For example, if the template contains the command `COPY *.TXT B:` and the current command line is:



`A>FORMAT A:`

after **F5** is pressed, the template contains the command `FORMAT A:`.

Ins

Normally, the template pointer is advanced when alphanumeric keys are typed on the command line. After **Ins** is pressed, subsequent alphanumeric keys do not cause the template pointer to be advanced until **Ins** is pressed again. When used with the **P** key, the **Ins** key allows characters to be "Inserted" into the template.



Each time the  key is pressed, the character pointed to by the template pointer is deleted from the template. The character is not displayed on the screen, and the current command line is unaffected. For example, if the current template contains the command `COPY *.TXT B:` and the template pointer points to the `*`, after the  key is pressed, the template contains:


`COPY .TXT B:`



or 

Moves the cursor and the template pointer one character to the left. The character on the screen is deleted, but the contents of the template is unchanged.



The  key cancels the current command line. The contents of the template is unchanged.

Other Useful Functions

This section discusses the other useful functions you can perform with the MS-DOS keyboard. These functions include:

- Canceling a Command
- Closing a File
- Logging the Screen
- Pausing the Screen
- Printing the Screen

Canceling a Command

The **CTRL** **C** (or **CTRL** **Break**) key combination cancels a command before it finishes executing. In addition, this key combination can be used like the **ESC** key to cancel current input lines before **Enter** is pressed.

Each time a character is input or output using the standard input/output, standard printer, or standard auxiliary device, MS-DOS checks to see if **CTRL** **C** (or **CTRL** **Break**) has been pressed. If it has, MS-DOS cancels the command. However, many commands are "disc intensive" and do not input and output characters frequently. Thus, it may be impossible to quickly terminate one of these commands. The **BREAK** command is used to extend checking for **CTRL** **C** (and **CTRL** **Break**) to *all* MS-DOS operations. See the **BREAK** command in the chapter entitled "MS-DOS Command Descriptions" for additional information.

Closing a File

The **CTRL** **Z** key combination creates an "end-of-file marker" in a file. Pressing **F6** is the same as pressing **CTRL** **Z**.

Logging the Screen

The **CTRL** **P** key combination turns on "Screen Logging". When Screen Logging is on, each character is sent to the printer at the same time it appears on the display screen. Pressing **CTRL** **P** again turns Screen Logging off.

On some computers, pressing **CTRL** **Print Screen** may also turn on Screen Logging.

Note



This function only works for characters output to the standard input/output device. It won't work for characters output to the screen through BIOS calls or by directly accessing the video hardware.

Pausing the Screen

The **CTRL** **S** key combination temporarily stops all processing by MS-DOS. This allows you time to read data that might otherwise scroll by too quickly (for example, during a long directory listing). Pressing any key resumes processing by MS-DOS.

On some computers, pressing **CTRL** **Num lock** or **Pause** may also temporarily stop all processing by MS-DOS.

Printing the Screen

The **Shift** **Print Screen** key combination takes a "snapshot" of the screen and sends it to the printer. This is also referred to as a "screen dump". If an MS-DOS command is executing, it is temporarily suspended while the current contents of the screen are printed.

On some computers, simply pressing **Print Screen** takes a snapshot of the screen and sends it to the printer.

Output for the printer is sent to parallel port 1 unless a **MODE** command has been issued to redirect that output. See the **MODE** command in the chapter entitled "MS-DOS Command Descriptions" for additional information.

Note



MS-DOS can print the screen any time the screen is in alphanumeric mode. However, MS-DOS cannot print the screen if it is in graphics mode. See the **GRAPHICS** command in the chapter entitled "MS-DOS Command Descriptions" for more information on how to print a screen in graphics mode.

Unused Keys

The following keys are not used by MS-DOS. They may, however, be used by application programs that you run.

F8 - F10

Scroll lock

Sys req

Home



Page Up

End



Page Down

C

Extended Screen and Keyboard Control (ANSI.SYS)

This appendix explains the use and features of ANSI.SYS. This device driver provides support for ANSI standard Terminal Escape sequences. In order to utilize these features, the extended screen and keyboard control device driver ANSI.SYS must be installed. To install this driver, insert the command line:

```
DEVICE=[<d>:] [<path>]ANSI.SYS
```

in the CONFIG.SYS file. Additional information regarding the installation of device drivers is contained in the chapter entitled "System Configuration".


Extended screen and keyboard control codes are valid for programs using standard input, standard output, and standard error message devices. These devices can be accessed through MS-DOS system calls 01H, 02H, 06H, 07H, 09H, 0AH, and 40H.



Control Sequence Syntax

All control sequences use the syntax shown below. Note that this syntax *is not* the same as the syntax used for MS-DOS commands.

ESC [<parameters> <command identifier>]

ESC This is the one byte ASCII escape code. All control sequences begin with this character. ESC is 1BH or 27 decimal. To generate the escape code, (1) enter the MS-DOS **PROMPT** command followed by **\$e** or (2) use an application program or a programming language. **DO NOT** type the letters ESC or press .

[This is the second character in the sequence.

<parameters> These are optional parameters. Parameters can consist of either alpha character strings or numeric values. Numeric values are indicated with the pound sign (#) character. If a parameter is omitted or a value of 0 is entered, the default value of the parameter is assumed.

<command
identifier> This is a single alphabetic character that identifies the control sequence.

Control Sequences

The control sequences recognized by ANSI.SYS are listed on the following pages. The screen control sequences are listed first, followed by the keyboard sequences.

Cursor Position

This sequence moves the current cursor position to the specified row and column. The row is specified by the first number; the column by the second. If the row and/or column number is omitted, the default value of 1 is substituted. If both numbers are omitted, the cursor is moved to the HOME position (Row 1, Column 1).

Syntax

ESC[#;#H

Example

The following sequence moves the cursor to Row 10, Column 5:

ESC[10;5H

Cursor Up

This sequence moves the cursor up a specified number of rows. If the cursor is already in the top row, this sequence is ignored by ANSI.SYS. The default number of rows to move is 1.

Syntax

ESC[#A

Example

The following sequence moves the cursor up 5 rows:

ESC[5A

Cursor Down

This sequence moves the cursor down a specified number of rows. If the cursor is already on the bottom row, this sequence is ignored by ANSI.SYS. The default number of rows to move is 1.

Syntax

ESC[#B

Example

The following sequence moves the cursor down 10 rows.

ESC[10B

Cursor Forward

This sequence moves the cursor forward (to the right) a specified number of columns. If the cursor is in the right-most column, this sequence is ignored by ANSI.SYS. The default number of columns to move is 1.

Syntax

ESC[#C

Example

The following sequence moves the cursor forward 8 columns:

ESC[8C

Cursor Backward

This sequence moves the cursor backward (to the left) a specified number of columns. If the cursor is in the left-most column, this sequence is ignored by ANSI.SYS. The default number of columns to move is 1.

Syntax

ESC[#D

Example

The following sequence moves the cursor backward 2 columns:

ESC[2D

Horizontal and Vertical Position

This sequence moves the current cursor position to the specified row and column. The row is specified by the first number and the column by the second. If the row and/or column number is omitted, the default value of 1 is substituted. If both numbers are omitted, the cursor is moved to the HOME position (Row 1, Column 1).

Syntax

ESC[#;#f

Example

The following sequence moves the cursor to Row 10, Column 5:

ESC[10;5f

Cursor Position Report

This sequence is returned in response to an inquiry. The sequence is returned through the standard input device. The first parameter specifies the current row and the second parameter specifies the current column.

Syntax

ESC[#;#R

Example

The following sequence reports the current cursor row position as 10 and column position as 5:

ESC[10;5R

Device Status Request

This sequence requests a Cursor Position Report from the console driver. The request is returned in the format shown for the Cursor Position Report on the previous page.

Syntax

ESC[6n

Example

The following sequence requests the current device status:

ESC[6n

After issuing this sequence, a Cursor Position Report can be read from the standard input device.

Save Cursor Position

This sequence saves the current cursor position so that it can be recalled later by the Restore Cursor Position sequence.

Syntax

ESC[s

Example

The following sequence saves the current cursor position:

ESC[s

Restore Cursor Position

This sequence restores the cursor position to the location it had when the console driver last received the Save Cursor Position sequence.

Syntax

ESC[u

Example

The following sequence restores the cursor position:

ESC[u

Erase in Display

This sequence clears the entire screen and homes the cursor.

Syntax

ESC[2J

Example

The following sequence clears the screen:

ESC[2J

Erase in Line

This sequence erases part of a line (or row), from the character under the cursor to the end of the line.

Syntax

ESC[K

Example

The following sequence erases to the end of the current line:

ESC[K

Set Graphics Rendition (SGR)

This sequence sets display attributes for the character at the current cursor position. The attributes apply to all characters following the current cursor position until the next SGR is encountered. Each character can have more than one display attribute. The following table lists the numbers assigned to each of the possible attributes.

Parameter	Graphics Attribute
0	All attributes off (White foreground, Black background)
1	Bold On (High Intensity)
4	Underscore On (monochrome display adapter only)
5	Blink On
7	Reverse Video
8	Cancel On (invisible)
30	Black foreground
31	Red foreground
32	Green foreground
33	Yellow foreground
34	Blue foreground
35	Magenta foreground
36	Cyan foreground

37	White foreground
40	Black background
41	Red background
42	Green background
43	Yellow background
44	Blue background
45	Magenta background
46	Cyan background
47	White background

Syntax

ESC[#;...;#m

Example

The following sequence sets the SGR at the current cursor position to Green foreground on a Black background:

ESC[32;40m

Set Mode

This sequence invokes the screen width or type specified by the parameter.

Parameter	Screen Width or Type
0	40X25 Black & White
1	40X25 Color or Gray Scale
2	80X25 Black & White
3	80X25 Color or Gray Scale
4	320X200 Color or Gray Scale
5	320X200 Black & White
6	640X200 Black & White
7	Wrap at end of line. Typing past end-of-line results in new line.

Syntax

ESC[=#h

or

ESC[=h

or

ESC[=0h

or

ESC[?7h

Example

The following sequence sets the screen to the 80X25 Black & White mode:

ESC[=2h

Reset Mode

This sequence is identical to Set Mode except that parameter 7 resets wrap at end-of-line mode. (Characters past end-of-line are thrown away.)

Examples

ESC[#1

or

ESC[=1

or

ESC[=01

or

ESC[?71

Keyboard Key Reassignment

The first ASCII code in the control sequence defines which code is being mapped. The remaining numbers define the sequence of ASCII codes generated when this key is intercepted. Note that there is one exception: If the first code in the sequence is zero, then the first and second code make up an extended ASCII definition.

Syntax

ESC[#;#;...#p

or

ESC["string"p

or

ESC[#;"string";#;#;"string";#p

or any other combination of strings and decimal numbers.

Examples

1. Reassign the Q and q keys to the A and a keys (and vice versa):

ESC[65;81p	A becomes Q
------------	-------------

ESC[97;113p	a becomes q
-------------	-------------

ESC[81;65p	Q becomes A
------------	-------------

ESC[113;97p	q becomes a
-------------	-------------

2. Reassign the **(F10)** key to a **DIR** command followed by a carriage return:

```
ESC[0;68;"DIR";13p
```

The 0;68 is the extended ASCII code for the **(F10)** key; 13 decimal is a carriage return.

D

Disc and Disc Drive Compatibility

This appendix describes the compatibility between flexible discs and disc drives. The following table indicates which types of flexible discs are compatible with each of the types of flexible disc drives. Only the combinations listed in this table are allowed.

Disc Drive Type	Disc Type
5.25-inch 160 or 180 Kb drive	5.25-inch single-sided (160/180 Kb) discs
5.25-inch 320 or 360 Kb drive	5.25-inch single-sided (160/180 Kb) or double-sided (320/360 Kb) discs
3.5-inch 720 Kb drive	3.5-inch double-sided (720 Kb) discs
5.25-inch 1.2 Mb drive	5.25-inch single-sided (160/180 Kb), double-sided (320/360 Kb), or high capacity (1.2 Mb) discs
3.5-inch 1.44 Mb drive	3.5-inch double-sided (720 Kb) or high density (1.44 Mb) discs

Note










To format a 160, 180, 320, or 360 Kb disc in a 1.2 Mb drive, specify the /4 option of the MS-DOS FORMAT command. To format a 720 Kb disc in a 1.44 Mb drive, specify the /T:80 and /N:9 options of the FORMAT command.



E

Character Codes and Keystrokes

Keystroke	Note	ASCII Value		Character Set		Color	
		Hex	Dec	STD	HP	Background	Foreground
Ctrl 2		00	0	Blank (Null)	N U	Black	Black
Ctrl A		01	1	☺	S T	Black	Blue
Ctrl B		02	2		S H	Black	Green
Ctrl C		03	3	♥	E H	Black	Cyan
Ctrl D		04	4	♦	E T	Black	Red
Ctrl E		05	5	♣	E Q	Black	Magenta
Ctrl F		06	6	♠	A K	Black	Brown
Ctrl G		07	7	•	⏏	Black	Lt. Gray
Ctrl H Backspace, Shift Backspace		08	8	•	B S	Black	Dk. Gray
Ctrl I		09	9	○	H T	Black	Lt. Blue
Ctrl J Ctrl		0A	10	○	L F	Black	Lt. Green
Ctrl K		0B	11	♂	V T	Black	Lt. Green
Ctrl L		0C	12	♀	F F	Black	Lt. Red
Ctrl M Shift		0D	13	♪	C R	Black	Lt. Magenta

Keystroke	Note	ASCII Value		Character Set		Color	
		Hex	Dec	STD	HP	Background	Foreground
Ctrl N		0E	14		S O	Black	Yellow
Ctrl O		0F	15		S I	Black	White
Ctrl P		10	16		D L	Blue	Black
Ctrl Q		11	17		D 1	Blue	Blue
Ctrl R		12	18		D 2	Blue	Green
Ctrl S		13	19	!!	D 3	Blue	Cyan
Ctrl T		14	20		D 4	Blue	Red
Ctrl U		15	21	§	N K	Blue	Magenta
Ctrl V		16	22	■	S Y	Blue	Brown
Ctrl W		17	23		E B	Blue	Lt. Gray
Ctrl X		18	24	↑	C N	Blue	Dk. Gray
Ctrl Y		19	25	↓	E M	Blue	Lt. Blue
Ctrl Z		1A	26	→	S B	Blue	Lt. Green
Ctrl [, Esc, Shift Esc, Ctrl Esc		1B	27	←	E C	Blue	Lt. Cyan
Ctrl \		1C	28		F S	Blue	Lt. Red
Ctrl]		1D	29		G S	Blue	Lt. Magenta
Ctrl 6		1E	30	▲	R S	Blue	Yellow
Ctrl -		1F	31	▼	V S	Blue	White

Keystroke	Note	ASCII Value		Character Set		Color	
		Hex	Dec	STD	HP	Background	Foreground
Space Bar, Shift, Space, Ctrl Space, or Alt Space		20	32	Blank Space		Green	Black
!	*	21	33	!	!	Green	Blue
"	*	22	34	"	"	Green	Green
#	*	23	35	#	#	Green	Cyan
\$	*	24	36	\$	\$	Green	Red
%	*	25	37	%	%	Green	Magenta
&	*	26	38	&	&	Green	Brown
'		27	39	'	'	Green	Lt. Gray
(*	28	40	((Green	Dk. Gray
)	*	29	41))	Green	Lt. Blue
*	&,*	2A	42	*	*	Green	Lt. Green
+	*	2B	43	+	+	Green	Lt. Cyan
,		2C	44	,	,	Green	Lt. Red
-		2D	45	-	-	Green	Lt. Magenta
.	\$	2E	46	.	.	Green	Yellow
/		2F	47	/	/	Green	White
0	+	30	48	0	0	Cyan	Black
1	+	31	49	1	1	Cyan	Blue
2	+	32	50	2	2	Cyan	Green
3	+	33	51	3	3	Cyan	Cyan
4	+	34	52	4	4	Cyan	Red
5	+	35	53	5	5	Cyan	Magenta
6	+	36	54	6	6	Cyan	Brown
7	+	37	55	7	7	Cyan	Lt. Gray
8	+	38	56	8	8	Cyan	Dk. Gray
9	+	39	57	9	9	Cyan	Lt. Blue

Keystroke	Note	ASCII Value		Character Set		Color	
		Hex	Dec	STD	HP	Background	Foreground
:	*	3A	58	:	:	Cyan	Lt. Green
;		3B	59	;	;	Cyan	Lt. Cyan
<	*	3C	60	<	<	Cyan	Lt. Red
=		3D	61	=	=	Cyan	Lt. Magenta
>	*	3E	62	>	>	Cyan	Yellow
?	*	3F	63	?	?	Cyan	White
@	*	40	64	@	@	Red	Black
A	#	41	65	A	A	Red	Blue
B	#	42	66	B	B	Red	Green
C	#	43	67	C	C	Red	Cyan
D	#	44	68	D	D	Red	Red
E	#	45	69	E	E	Red	Magenta
F	#	46	70	F	F	Red	Brown
G	#	47	71	G	G	Red	Lt. Gray
H	#	48	72	H	H	Red	Dk. Gray
I	#	49	73	I	I	Red	Lt. Blue
J	#	4A	74	J	J	Red	Lt. Green
K	#	4B	75	K	K	Red	Lt. Cyan
L	#	4C	76	L	L	Red	Lt. Red
M	#	4D	77	M	M	Red	Lt. Magenta
N	#	4E	78	N	N	Red	Yellow
O	#	4F	79	O	O	Red	White
P	#	50	80	P	P	Magenta	Black
Q	#	51	81	Q	Q	Magenta	Blue
R	#	52	82	R	R	Magenta	Green
S	#	53	83	S	S	Magenta	Cyan
T	#	54	84	T	T	Magenta	Red
U	#	55	85	U	U	Magenta	Magenta
V	#	56	86	V	V	Magenta	Brown
W	#	57	87	W	W	Magenta	Lt. Gray































E-4 Character Codes and Keystrokes




Keystroke	Note	ASCII Value		Character Set		Color	
		Hex	Dec	STD	HP	Background	Foreground
X	#	58	88	x	x	Magenta	Dk. Gray
Y	#	59	89	y	y	Magenta	Lt. Blue
Z	#	5A	90	z	z	Magenta	Lt. Green
[5B	91	[[Magenta	Lt. Cyan
\		5C	92	\	\	Magenta	Lt. Red
]		5D	93]]	Magenta	Lt. Magenta
^	*	5E	94	^	^	Magenta	Yellow
_	*	5F	95	_	_	Magenta	White
'		60	96	'	'	Yellow	Black
a	-	61	97	a	a	Yellow	Blue
b	-	62	98	b	b	Yellow	Green
c	-	63	99	c	c	Yellow	Cyan
d	-	64	100	d	d	Yellow	Red
e	-	65	101	e	e	Yellow	Magenta
f	-	66	102	f	f	Yellow	Brown
g	-	67	103	g	g	Yellow	Lt. Gray
h	-	68	104	h	h	Yellow	Dk. Gray
i	-	69	105	i	i	Yellow	Lt. Blue
j	-	6A	106	j	j	Yellow	Lt. Green
k	-	6B	107	k	k	Yellow	Lt. Cyan
l	-	6C	108	l	l	Yellow	Lt. Red
m	-	6D	109	m	m	Yellow	Lt. Magenta
n	-	6E	110	n	n	Yellow	Yellow
o	-	6F	111	o	o	Yellow	White
p	-	70	112	p	p	White	Black
q	-	71	113	q	q	White	Blue
r	-	72	114	r	r	White	Green
s	-	73	115	s	s	White	Cyan
t	-	74	116	t	t	White	Red
u	-	75	117	u	u	White	Magenta

Keystroke	Note	ASCII Value		Character Set		Color	
		Hex	Dec	STD	HP	Background	Foreground
v	-	76	118	v	v	White	Brown
w	-	77	119	w	w	White	Lt. Gray
x	-	78	120	x	x	White	Dk. Gray
y	-	79	121	y	y	White	Lt. Blue
z	-	7A	122	z	z	White	Lt. Green
{	*	7B	123	{	{	White	Lt. Cyan
	*	7C	124			White	Lt. Red
}	*	7D	125	}	}	White	Lt. Magenta
~	*	7E	126	~	~	White	Yellow
Ctrl ←		7F	127	Δ	■	White	White
Alt 128	%"	80	128	Ç	HP Reserved	Black	Black
Alt 129	%"	81	129	ü	HP Reserved	Black	Blue
Alt 130	%"	82	130	é	HP Reserved	Black	Green
Alt 131	%"	83	131	à	HP Reserved	Black	Cyan
Alt 132	%"	84	132	ä	HP Reserved	Black	Red
Alt 133	%"	85	133	à	HP Reserved	Black	Magenta
Alt 134	%"	86	134	ä	HP Reserved	Black	Brown
Alt 135	%"	87	135	ç	HP Reserved	Black	Lt. Gray
Alt 136	%"	88	136	ê	HP Reserved	Black	Dk. Gray
Alt 137	%"	89	137	ë	HP Reserved	Black	Lt. Blue
Alt 138	%"	8A	138	è	HP Reserved	Black	Lt. Green
Alt 139	%"	8B	139	ï	HP Reserved	Black	Lt. Green
Alt 140	%"	8C	140	î	HP Reserved	Black	Lt. Red

Keystroke	Note	ASCII Value		Character Set		Color	
		Hex	Dec	STD	HP	Background	Foreground
Alt 141	%"	8D	141	ì	HP Reserved	Black	Lt. Magenta
Alt 142	%"	8E	142	Ä	HP Reserved	Black	Yellow
Alt 143	%"	8F	143	Å	HP Reserved	Black	White
Alt 144	%"	90	144	É	HP Reserved	Blue	Black
Alt 145	%"	91	145	æ	HP Reserved	Blue	Blue
Alt 146	%"	92	146	Æ	HP Reserved	Blue	Green
Alt 147	%"	93	147	ô	HP Reserved	Blue	Cyan
Alt 148	%"	94	148	ö	HP Reserved	Blue	Red
Alt 149	%"	95	149	ò	HP Reserved	Blue	Magenta
Alt 150	%"	96	150	û	HP Reserved	Blue	Brown
Alt 151	%"	97	151	ù	HP Reserved	Blue	Lt. Gray
Alt 152	%"	98	152	ÿ	HP Reserved	Blue	Dk. Gray
Alt 153	%"	99	153	ö	HP Reserved	Blue	Lt. Blue
Alt 154	%"	9A	154	ü	HP Reserved	Blue	Lt. Green
Alt 155	%"	9B	155	¢	HP Reserved	Blue	Lt. Cyan
Alt 156	%"	9C	156	£	HP Reserved	Blue	Lt. Red
Alt 157	%"	9D	157	¥	HP Reserved	Blue	Lt. Magenta
Alt 158	%"	9E	158	Pt	HP Reserved	Blue	Yellow
Alt 159	%"	9F	159	§	HP Reserved	Blue	White
Alt 160	%	A0	160	à		Green	Black

Keystroke	Note	ASCII Value		Character Set		Color	
		Hex	Dec	STD	HP	Background	Foreground
Alt 161	%	A1	161	í	À	Green	Blue
Alt 162	%	A2	162	ó	Á	Green	Green
Alt 163	%	A3	163	ú	Ê	Green	Cyan
Alt 164	%	A4	164	ñ	Ë	Green	Red
Alt 165	%	A5	165	Ñ	Ê	Green	Magenta
Alt 166	%	A6	166	ø	Ï	Green	Brown
Alt 167	%	A7	167	ø	Ï	Green	Lt. Gray
Alt 168	%	A8	168	¿	·	Green	Dk. Gray
Alt 169	%	A9	169		·	Green	Lt. Blue
Alt 170	%	AA	170	¬	·	Green	Lt. Green
Alt 171	%	AB	171	½	·	Green	Lt. Cyan
Alt 172	%	AC	172	¼	·	Green	Lt. Red
Alt 173	%	AD	173	¡	Ù	Green	Lt. Magenta
Alt 174	%	AE	174	<<	Ù	Green	Yellow
Alt 175	%	AF	175	>>	£	Green	White
Alt 176	%	B0	176	⋮	—	Cyan	Black
Alt 177	%	B1	177	⋮	Ÿ	Cyan	Blue
Alt 178	%	B2	178	⋮	Ÿ	Cyan	Green
Alt 179	%	B3	179	⋮	°	Cyan	Cyan
Alt 180	%	B4	180	⋮	Ç	Cyan	Red
Alt 181	%	B5	181	⋮	ç	Cyan	Magenta
Alt 182	%	B6	182	⋮	Ñ	Cyan	Brown
Alt 183	%	B7	183	⋮	ñ	Cyan	Lt. Gray
Alt 184	%	B8	184	⋮	¿	Cyan	Dk. Gray
Alt 185	%	B9	185	⋮	¿	Cyan	Lt. Blue
Alt 186	%	BA	186	⋮	*	Cyan	Lt. Green
Alt 187	%	BB	187	⋮	£	Cyan	Lt. Cyan
Alt 188	%	BC	188	⋮	¥	Cyan	Lt. Red
Alt 189	%	BD	189	⋮	§	Cyan	Lt. Magenta
Alt 190	%	BE	190	⋮	ƒ	Cyan	Yellow

Keystroke	ASCII Value			Character Set		Color	
	Note	Hex	Dec	STD	HP	Background	Foreground
Alt 191	%	BF	191		€	Cyan	White
Alt 192	%	C0	192		à	Red	Black
Alt 193	%	C1	193		é	Red	Blue
Alt 194	%	C2	194		ò	Red	Green
Alt 195	%	C3	195		ú	Red	Cyan
Alt 196	%	C4	196		á	Red	Red
Alt 197	%	C5	197		ê	Red	Magenta
Alt 198	%	C6	198		ó	Red	Brown
Alt 199	%	C7	199		û	Red	Lt. Gray
Alt 200	%	C8	200		ä	Red	Dk. Gray
Alt 201	%	C9	201		ë	Red	Lt. Blue
Alt 202	%	CA	202		ö	Red	Lt. Green
Alt 203	%	CB	203		ü	Red	Lt. Cyan
Alt 204	%	CC	204		å	Red	Lt. Red
Alt 205	%	CD	205		è	Red	Lt. Magenta
Alt 206	%	CE	206		õ	Red	Yellow
Alt 207	%	CF	207		ù	Red	White
Alt 208	%	D0	208		À	Magenta	Black
Alt 209	%	D1	209		Í	Magenta	Blue
Alt 210	%	D2	210		Ó	Magenta	Green
Alt 211	%	D3	211		Æ	Magenta	Cyan
Alt 212	%	D4	212		å	Magenta	Red
Alt 213	%	D5	213		í	Magenta	Magenta
Alt 214	%	D6	214		Ó	Magenta	Brown
Alt 215	%	D7	215		æ	Magenta	Lt. Gray
Alt 216	%	D8	216		À	Magenta	Dk. Gray
Alt 217	%	D9	217		ï	Magenta	Lt. Blue
Alt 218	%	DA	218		ö	Magenta	Lt. Green
Alt 219	%	DB	219		Û	Magenta	Lt. Cyan
Alt 220	%	DC	220		É	Magenta	Lt. Red

Keystroke	Note	ASCII Value		Character Set		Color	
		Hex	Dec	STD	HP	Background	Foreground
Alt 221	%	DD	221		ı	Magenta	Lt. Magenta
Alt 222	%	DE	222		β	Magenta	Yellow
Alt 223	%	DF	223		δ	Magenta	White
Alt 224	%	E0	224	α	Α	Yellow	Black
Alt 225	%	E1	225	β	Β	Yellow	Blue
Alt 226	%	E2	226	γ	Γ	Yellow	Green
Alt 227	%	E3	227	π	Δ	Yellow	Cyan
Alt 228	%	E4	228	Σ	δ	Yellow	Red
Alt 229	%	E5	229	δ	ı	Yellow	Magenta
Alt 230	%	E6	230	μ	İ	Yellow	Brown
Alt 231	%	E7	231	τ	Ó	Yellow	Lt. Gray
Alt 232	%	E8	232	Φ	ò	Yellow	Dk. Gray
Alt 233	%	E9	233	θ	Õ	Yellow	Lt. Blue
Alt 234	%	EA	234	Ω	ò	Yellow	Lt. Green
Alt 235	%	EB	235	δ	˘	Yellow	Lt. Cyan
Alt 236	%	EC	236	∞	˘	Yellow	Lt. Red
Alt 237	%	ED	237	φ	Ú	Yellow	Lt. Magenta
Alt 238	%	EE	238	ε	Ÿ	Yellow	Yellow
Alt 239	%	EF	239	∩	γ	Yellow	White
Alt 240	%	F0	240	≡	þ	White	Black
Alt 241	%	F1	241	±	þ	White	Blue
Alt 242	%	F2	242	≥	.	White	Green
Alt 243	%	F3	243	≤	μ	White	Cyan
Alt 244	%	F4	244	∫	¶	White	Red
Alt 245	%	F5	245	∫	¾	White	Magenta
Alt 246	%	F6	246	÷	–	White	Brown
Alt 247	%	F7	247	≈	¼	White	Lt. Gray
Alt 248	%	F8	248	○	½	White	Dk. Gray
Alt 249	%	F9	249	●	¾	White	Lt. Blue

E-10 Character Codes and Keystrokes

Keystroke	ASCII Value			Character Set		Color	
	Note	Hex	Dec	STD	HP	Background	Foreground
Alt 250	%	FA	250	●	ø	White	Lt. Green
Alt 251	%	FB	251	√	<<	White	Lt. Cyan
Alt 252	%	FC	252	η	■	White	Lt. Red
Alt 253	%	FD	253	²	>>	White	Lt. Magenta
Alt 254	%	FE	254	■	±	White	Yellow
Alt 255	%	FF	255	BLANK		White	White



F

The SORT Command Collating Sequences

ASCII Value	ASCII Weight	ASCII Value	ASCII Weight
128	67	153	79
129	85	154	85
130	69	155	36
131	65	156	36
132	65	157	36
133	65	158	36
134	65	159	36
135	67	160	65
136	69	161	73
137	69	162	79
138	69	163	85
139	73	164	78
140	73	165	78
141	73	166	166
142	65	167	167
143	65	168	63
144	69	169	169
145	65	170	170
146	65	171	171
147	79	172	172
148	79	173	33
149	79	174	34
150	85	175	34
151	85	225	83
152	89		

INDEX

A

Active drive 2-15
ANSI.SYS 6-10, C-1
Append command 8-11
Application program support 1-1
Archive file attribute 2-12
Assemble command 10-12
ASSIGN command 3-7, 4-7
ATTRIB command 3-6, 4-9
AUTOEXEC.BAT file 5-6
AUX 2-8

B

BACKUP command 3-6, 4-12
.BAK 2-9
Batch commands 5-10
 ECHO 5-11
 FOR 5-14
 GOTO 5-16
 IF 5-18
 PAUSE 5-20
 REM 5-22
 SHIFT 5-23
Batch files 5-1
Batch processing 5-1
 AUTOEXEC.BAT 5-6
 Batch files 5-1
 Chaining batch files 5-5
 Creating a batch file 5-2
 Executing a batch file 5-3
 Replaceable parameters, 5-6

Using %0 - %9	5-7
Using the SET command	5-9
Valid batch filenames	5-2
BREAK command	3-9, 4-17, 6-3
BUFFERS command	6-5

C

Canceling a command	B-5
Character codes	E-1
CHDIR command	3-7, 4-19
CHKDSK command	3-8, 4-21
Class	9-4
CLOCK\$	2-8
Closing a file	B-6
CLS command	3-9, 4-25
Collating sequences, SORT command	F-1
COM	3-1
Command characters, MS-LINK	9-28
COMMAND command	3-10, 4-26
COMMAND.COM	3-1
Command line method	9-17
Command line template	B-1
Command prompts, MS-LINK	9-23
Command syntax	4-2
Common combinations	9-7
COMP command	3-6, 4-30
Compare command	10-15
CON	2-8, 4-38, 7-1
CONFIG.SYS file	6-1
Configuration commands	6-2
Control sequence syntax	C-2
Control sequences	C-3
Cursor backward	C-8
Cursor down	C-6
Cursor forward	C-7
Cursor position	C-4
Cursor position report	C-10
Cursor up	C-5

Device status request	C-11
Erase in display	C-14
Erase in line	C-15
Horizontal and vertical position	C-9
Keyboard key reassignment	C-21
Reset mode	C-20
Restore cursor position	C-13
Save cursor position	C-12
Set graphics rendition (SGR)	C-16
Set mode	C-18
Syntax	C-2
Copy command	8-13
COPY command	3-6, 4-34
COUNTRY command	6-7
CTRL C	4-17, 6-3
CTRL C command character	9-28, 9-30
CTTY command	3-9, 4-41
Current directory	2-17
Cursor backward	C-8
Cursor down	C-6
Cursor forward	C-7
Cursor position	C-4
Cursor position report	C-10
Cursor up	C-5

D

Data files	2-2
DATE command	3-10, 4-43
DEBUG	10-1
Command descriptions	10-11
Command parameters	10-3
Error messages	10-46
How to start DEBUG	10-9
Introduction	10-1
Memory addressing	10-6
DEBUG commands,	
(A) Assemble command	10-12
(C) Compare command	10-15

(D) Dump command	10-16
(E) Enter command	10-18
(F) Fill command.....	10-20
(G) Go command.....	10-21
(H) Hex command.....	10-24
(I) Input command	10-25
(L) Load command	10-26
(M) Move command.....	10-28
(N) Name command.....	10-29
(O) Output command	10-32
(P) Procedure command.....	10-33
(Q) Quit command	10-34
(R) Register command	10-35
(S) Search command.....	10-39
(T) Trace command	10-40
(U) Unassemble command.....	10-42
(W) Write command	10-44
DEL command.....	3-6, 4-46
Delete command.....	8-16
DEVICE command	6-9
Device drivers	6-27
Device status request.....	C-11
DIR command	3-6, 4-49
Directories,	
Definition.....	2-12
Entries	2-13
Hierarchical structure	2-12
Root directory	2-13
Subdirectories.....	2-13
Disc, file and system management	1-1
Disc management.....	1-1
Discs,	
Compatibility	D-1
Formatting.....	4-88
Virtual	6-14
Volume label.....	4-183
Discs and disc drive compatability	D-1
DISKCOMP command	3-8, 4-54
DISKCOPY command	3-8, 4-57
Drive designator	3-3
DRIVER.SYS	6-11

/DSALLOCATE switch	9-32
Dump command	10-16

E

ECHO command	5-11
Edit command	8-20
Editing a command line	B-1
EDLIN	8-1
Command parameters	8-8
Error messages	8-46
How to start EDLIN	8-2
Information common to all EDLIN commands	8-4
Introduction	8-1
Syntax	8-7
EDLIN commands	8-10
A (Append)	8-11
C (Copy)	8-13
D (Delete)	8-16
Edit	8-20
E (End)	8-23
I (Insert)	8-24
L (List)	8-27
M (Move)	8-31
P (Page)	8-33
Q (Quit)	8-34
R (Replace)	8-35
S (Search)	8-39
T (Transfer)	8-43
W (Write)	8-45
EDLIN error messages	8-46
End command	8-23
End-of-file marker	B-6
Enter command	10-18
Environment, MS-DOS	4-156
ERASE command	3-6, 4-60
Erase in display	C-14
Erase in line	C-15
Error Messages,	

DEBUG	10-46
EDLIN	8-46
MS-DOS	A-1
MS-LINK	9-38
.EXE	3-1
EXE2BIN command	3-6, 4-63
EXIT command	3-10, 4-67
Extended screen and keyboard control	C-1
External MS-DOS commands	3-1

F

FC command	3-6 4-68
FCBS command	6-18
FDISK command	3-8, 4-75
File,	
Attributes	2-11
Clusters	2-15
Definition	2-1
File allocation tables (FATs)	2-15
File control blocks (FCBs)	6-18
Files, directories, and paths	2-17
File handles	6-20
File management	1-1
Filenames,	
Naming a file	2-3
Reserved filenames and extensions	2-7
Valid filenames and extensions	2-4
Wildcard characters	2-10
FILES command	6-20
Fill command	10-20
Filters	7-7
FIND command	3-6, 4-83, 7-7
FOR command	5-14
FOR150 command	3-8, 4-86
FORMAT command	3-8, 4-88

G	Go command	10-21
	GOTO command.....	5-16
	GRAFTABL command	3-9, 4-94
	GRAPHICS command	3-9, 4-96
	Group.....	9-4

H	Hex command	10-24
	Hidden file attribute	2-11
	Hierarchical directory structure.....	2-12
	/HIGH Switch.....	9-33

I	IBMBIO.COM.....	2-11
	IBMDOS.COM.....	2-11
	IF command	5-18
	Input command.....	10-25
	Input files.....	9-12, 9-13
	Insert command.....	8-24
	Installable device drivers.....	6-28
	Internal MS-DOS commands.....	3-1

J	JOIN command	3-8, 4-99
----------	--------------------	-----------

K	Keyboard, MS-DOS.....	B-1
	KEYBxx command.....	3-9, 4-103
	Keys, MS-DOS editing	B-1

L	LABEL command.....	3-8, 4-106
	LASTDRIVE.....	6-22
	Library files	9-12, 9-26
	Line editor (EDLIN)	8-1
	/LINENUMBERS switch.....	9-34
	List command.....	8-27
	List files	9-12, 9-14, 9-26
	Load command	10-26
	Logging the screen.....	B-6

M

/MAP switch	9-34
Memory addressing	10-6
MKDIR command	3-8, 4-109
MODE command	3-9, 4-112
MORE command	3-6, 4-120, 7-7
Move command.....	8-31, 10-28
MS-DOS	1-1
Command basics	3-1
Command descriptions.....	4-1
Concepts	2-1
Constructing a command line	3-3
Definition.....	1-1
Directory structure	2-14
Editing keys	B-2
Environment.....	4-156
Error messages.....	A-1
Executing a command line	3-5

Filters.....	7-7
Introduction to MS-DOS.....	3-1
Keyboard	B-1
Message directory.....	A-1
Partition	4-75
Paths.....	2-17
Prompt.....	3-2
System date and time.....	4-43, 4-172
MS-DOS command basics	3-1
MS-DOS command line	3-2
Constructing a command line	3-3
Executing a command line	3-5
Prompt.....	3-2
MS-DOS commands	3-6
ASSIGN.....	3-7, 4-7
ATTRIB.....	3-6, 4-9
BACKUP.....	3-6, 4-12
BREAK	3-9, 4-17
CHDIR.....	3-7, 4-19
CHKDSK	3-8, 4-21
CLS	3-9, 4-25
COMMAND	3-10, 4-26
COMP.....	3-6, 4-30
COPY.....	3-6, 4-34
CTTY	3-9, 4-41
DATE.....	3-10, 4-43
DEL.....	3-6, 4-46
DIR	3-6, 4-49
DISKCOMP	3-8, 4-54
DISKCOPY	3-8, 4-57
ERASE.....	3-6, 4-60
EXE2BIN	3-6, 4-63
EXIT.....	3-10, 4-67
FC	3-6, 4-68
FDISK	3-8, 4-75
FIND.....	3-6, 4-83, 7-7
FOR150.....	3-8, 4-86
FORMAT	3-8, 4-88
GRAFTABL	3-9, 4-94
GRAPHICS	3-9, 4-96
JOIN	3-8, 4-99

KEYBxx.....	3-9, 4-103
LABEL.....	3-8, 4-106
MKDIR.....	3-8, 4-109
MODE.....	3-9, 4-112
MORE.....	3-6, 4-120, 7-7
PATH.....	3-7, 4-122
PRINT.....	3-9, 4-125
PROMPT.....	3-10, 4-131
RECOVER.....	3-7, 4-135
RENAME.....	3-7, 4-139
REPLACE.....	3-7, 4-141
RESTORE.....	3-7, 4-146
RMDIR.....	3-8, 4-149
SELECT.....	3-8, 4-151
SET.....	3-10, 4-156, 5-9
SHARE.....	3-7, 4-159
SORT.....	3-7, 4-162, 7-7
SUBST.....	3-8, 4-165
SYS.....	3-8, 4-170
TIME.....	3-10, 4-172
TREE.....	3-7, 4-175
TYPE.....	3-7, 4-178
VER.....	3-10, 4-180
VERIFY.....	3-10, 4-181
VOL.....	3-8, 4-183
XCOPY.....	3-7, 4-185
MS-DOS line editor.....	8-1
MS-DOS syntax.....	4-2
MS-LINK,	
Command characters.....	9-28
Command prompts.....	9-23
Creating a response file.....	9-21
Definition of terms.....	9-4
Error messages.....	9-38
Examples.....	9-44
Files that MS-LINK uses.....	9-12
How MS-LINK combines & arranges segments....	9-7
Running MS-LINK.....	9-16
What MS-LINK is and what it does.....	9-1
MS-LINK Switches.....	9-30
/DALLOCATE.....	9-32

/HIGH	9-33
/LINENUMBERS	9-34
/MAP	9-34
/PAUSE	9-36
/STACK	9-36

N

Name command	10-29
Naming a file.....	2-3
Invalid characters.....	2-6
MS-DOS reserved extensions	2-9
MS-DOS reserved filenames.....	2-8
Valid characters	2-5
Wildcard characters	2-10
NUL	2-9
NUL.MAP	9-24, 9-26

O

Object files	9-12
Object modules	9-25
Options.....	3-3
Organizing files.....	2-12
Output command	10-32
Output files.....	9-12, 9-14

P

Page command.....	8-33
PAM.CIF.....	2-11
Paragraph.....	9-4
Parallel printer mode.....	4-114
Parameters.....	3-3
Partition, MS-DOS.....	4-75
Path.....	2-17, 3-3
PATH command.....	3-7, 4-122
PAUSE command.....	5-20
/PAUSE switch.....	9-36
Pausing the screen.....	B-6
Peripherals.....	6-1
Piping Input/Output.....	7-5
Plus-Sign command character.....	9-28
PRINT command.....	3-9, 4-125
Printing the screen.....	B-7
Private combinations.....	9-7
Procedure command.....	10-33
PRN.....	2-9
Program files.....	2-2
PROMPT command.....	3-10, 4-131
Public combinations.....	9-7

Q

Quit command.....	8-34, 10-34
-------------------	-------------

R

Read-only file attribute.....	2-11
RECOVER command.....	3-7, 4-135
Redirecting input/output.....	7-2
Redirecting input.....	7-3
Redirecting output.....	7-4
Redirecting parallel printer output.....	4-114
Redirection.....	7-2
Register command.....	10-35
REM command.....	5-22
RENAME command.....	3-7, 4-139
Replace command.....	8-35
REPLACE command.....	3-7, 4-141
Replaceable parameters and batch files.....	5-6
Reset mode.....	C-20
Response file.....	9-20
Response file method.....	9-20
RESTORE command.....	3-7, 4-146
Restore cursor position.....	C-13
RMDIR command.....	3-8, 4-149
Root Directory.....	2-13
Run files.....	9-12, 9-14, 9-25

S

Save cursor position.....	C-12
Screen dump.....	B-7
Search command.....	8-39, 10-39
Segment.....	9-4
Segments and groups in memory.....	9-6
SELECT command.....	3-8, 4-151
Semicolon command character.....	9-28, 9-29
Serial communication mode.....	4-113
SET command.....	3-10, 4-156, 5-9
Set graphics rendition (SGR).....	C-16
Set mode.....	C-18
SHARE command.....	3-7, 4-159
SHELL command.....	6-23

SHIFT command	5-23
SORT command	3-7, 4-162, 7-7
SORT command collating sequences	F-1
Stack combinations	9-7
/STACK switch	9-36
STACKS command	6-25
Standard input/output devices	7-1
Subdirectories	2-13
SUBST command	3-8, 4-165
Switches, MS-LINK	9-30
SYS command	3-8, 4-170
System configuration commands,	
BREAK	6-3
BUFFERS	6-5
COUNTRY	6-7
DEVICE	6-9
FCBS	6-18
FILES	6-20
LASTDRIVE	6-22
SHELL	6-23
STACKS	6-25
System date and time	4-43, 4-172
System file attribute	2-12
System management	1-1

T

Template, MS-DOS	B-1
Text prompt method	9-17
TIME command	3-10, 4-172
Trace command	10-40
Transfer command	8-43
TREE command	3-7, 4-175
TYPE command	3-7, 4-178

U	Unassemble command	10-42
	Understanding files	2-1

V	VDISK.SYS	6-14
	VER command	3-10, 4-180
	VERIFY command	3-10, 4-181
	Video display mode	4-112
	Virtual memory file	9-15
	VOL command	3-8, 4-183
	Volume label	4-183
	VM.TMP file	9-12, 9-15

W	Wildcard characters	2-10
	Write command	8-45, 10-44

X	XCOPY command	3-7, 4-185
----------	---------------------	------------
