NCR

NCR DECISION MATE V

MS™- DOS User's Guide

MS and MULTIPLAN are trademarks of Microsoft Corporation. CP/M is a registered trademark of Digital Research.

Third Edition, August 1984

First Edition, June 1983

It is the policy of NCR Corporation to improve products as new technology, components, software, and firmware become available. NCR Corporation, therefore, reserves the right to change specifications without prior notice.

All features, functions, and operations described herein may not be marketed by NCR in all parts of the world. In some instances, photographs are of equipment prototypes. Therefore, before using this document, consult your nearest dealer or NCR office for information that is applicable and current.

MS-DOS USER'S GUIDE

CONTENTS

1. INTRODUCTION

WHAT IS AN OPERATING SYSTEM? 1-1
HOW TO USE THIS MANUAL 1-1
Syntax Notation 1-2
Flexible/Fixed Disk Systems Considerations 1-3

2. GETTING STARTED

SYSTEM SETUP 2-1
Loading MS-DOS 2-1
Copying your Master Software Disk 2-2
 2 Flexible Disk Drives (chart) 2-3
 1 Flexible Disk Drive (chart) 2-5 4
Learning about your Drive Designations 2-7 5
Defining your Fixed Disks 2-7 5
 Sequence Summary 2-8 7
Formatting Disks 2-12 11
Defining ~~a Serial Printer . . . . 2-12~~
your processing environment

FREQUENTLY PERFORMED OPERATIONS 2-12
Entering the Date and Time 2-12
Changing the Default Drive 2-14
Backing-Up your Disks 2-14
* Using the Programmable Function Keys . . etc 2-15 14
Running Programs Automatically 2-15
Turning the System Off 2-15
System Reset 2-15

FILES 2-15 16
How MS-DOS Keeps Track of Files 2-16
The DIR (Show Directory) Command 2-16
Checking your Disks 2-17

CHAPTER REVIEW 2-18

* Escape Sequences 2-15

Copy Index

3. MORE ABOUT FILES

NAMING YOUR FILES 3-1
Wild Cards 3-2
The ? Wild Card 3-2
The * Wild Card 3-3
Examples 3-3
Illegal Filenames 3-4

COPYING YOUR FILES 3-4

PROTECTING YOUR FILES 3-6

DIRECTORIES 3-6
Filenames and Paths 3-8
Pathnames 3-9
Pathing and External Commands 3-10
Pathing and Internal Commands 3-11
Displaying your Working Directory 3-11
Creating a Directory 3-12
Changing your Working Directory 3-13
Removing a Directory 3-13

4. LEARNING ABOUT COMMANDS

GENERAL INFORMATION 4-1
Types of MS-DOS Commands 4-1
Internal Commands 4-1
External Commands 4-1
Command Options 4-2
Common Entry Conventions 4-3

BATCH PROCESSING 4-4
The AUTOEXEC.BAT File 4-6
Creating an AUTOEXEC.BAT File 4-7
Creating a .BAT File with Replaceable Parameters 4-9
Executing a .BAT File 4-10

INPUT AND OUTPUT 4-11
Redirecting your Output 4-11
Filters 4-12
Command Piping 4-12

5. MS-DOS COMMANDS

COMMAND SUMMARY 5-2
BACKUP 5-4
BREAK 5-5
CHDIR 5-6
CHKDSK 5-7
CIPHER 5-10
CLS (Clear) 5-12
CONFIG 5-13
COPY 5-17
CTTY 5-20
DATE 5-21
DEL 5-23
DIR 5-24
DISKCOPY 5-25
ECHO 5-27
EXIT 5-28
FIND 5-29
FOR 5-31
FORMAT 5-33
GOTO 5-35
IF 5-36
LOCATE 5-38
MKDIR 5-41
MORE 5-42
PATH 5-43
PAUSE 5-44
PRINT 5-45
PROMPT 5-48
RDCPM 5-50
RECOVER 5-52
REM (Remark) 5-53
REN (Rename) 5-54
RMDIR (Remove Directory) 5-56
SET 5-57
SHIFT 5-58
SORT 5-59
SYS (System) 5-61
TIME 5-63
TYPE 5-65
VER (Version) 5-66
VERIFY 5-67
VOL (Volume) 5-68

6. MS-DOS EDITING AND FUNCTION KEYS

SPECIAL EDITING KEYS 6-1
CONTROL CHARACTER FUNCTIONS 6-5

7. LINE EDITOR (EDLIN)

GENERAL INFORMATION 7-1
How to Start EDLIN 7-1
Special Editing Keys 7-3
COPY1 7-4
COPYUP 7-5
COPYALL 7-6
SKIP1 7-7
SKIPUP 7-8
VOID 7-9
INSERT 7-10
EXIT 7-12
NEWLINE 7-13

EDLIN COMMANDS 7-14
Format Conventions 7-14
Command Options 7-16

(A)PPEND 7-18
(C)OPY 7-19
(D)ELETE 7-21
EDIT 7-23
(E)ND 7-25
(I)NSERT 7-26
(L)IST 7-29
(M)OVE 7-32
(P)AGE 7-33
(Q)UIT 7-34
(R)EPLACE 7-35
(S)EARCH 7-38
(T)RANSFER 7-41
(W)RITE 7-42

ERROR MESSAGES 7-43

8. FILE COMPARE (FC) UTILITY

GENERAL INFORMATION 8-1
Limitations on Source Comparisons 8-1
File Specifications 8-1

HOW TO USE FILE COMPARE 8-2
FC Switches 8-2
Difference Reporting 8-4
Redirecting FC Output to a File 8-5
Examples 8-5

ERROR MESSAGES 8-9

9. LINKER PROGRAM (MS-LINK)

GENERAL INFORMATION 9-1
Program Overview 9-1
Definitions You'll Need To Know 9-3
Files That MS-LINK Uses 9-5
Input File Extensions 9-5
Output File Extensions 9-6
VM.TMP (Temporary) File 9-6

USING MS-LINK 9-6
Starting MS-LINK 9-6
Method 1: Prompts 9-7
Method 2: Command Line 9-8
Method 3: Response File 9-8
Command Characters 9-10
Command Prompts 9-11
MS-LINK Switches 9-13

SAMPLE MS-LINK SESSION 9-15

ERROR MESSAGES 9-17

A. DUAL-OPERATING SYSTEMS
CONSIDERATIONS A-1

B. DISK ERRORS B-1

C. HOW TO OBTAIN AND INSTALL SOFTWARE

MAKING THE RIGHT PURCHASE C-1
INSTALLING THE SOFTWARE C-2
Functional Characteristics Information C-3
MS-DOS ANSI Driver Table C-3
Terminal Function Codes C-4
Miscellaneous Information C-5
Music Codes C-6
Translation/Conversion Information C-8
Special Country Keyboard Definitions C-8
ASCII Code Chart C-9

Copy

INTRODUCTION

MS$^{\text{TM}}$-DOS is a disk operating system for the 16-bit processor of NCR DECISION MATE V. Through MS-DOS, you communicate with the computer, disk drives, and printer (if available), managing these resources to your advantage.

WHAT IS AN OPERATING SYSTEM?

An operating system is your "silent partner" when you are using the computer. It provides the interface between the hardware and both you and the other software (application packages and your own programs). An operating system can be compared to the electricity in a house: You need it for the toaster and the blender to work, but you are not always aware that it's there.

Operating systems provide varying capabilities. With MS-DOS, you can create and keep track of files, run and link programs, and access peripheral devices (for example, printers and disk drives) that are attached to your computer.

HOW TO USE THIS MANUAL

This manual describes MS-DOS and how to use it. This chapter introduces some basic MS-DOS concepts; Chapter 2 discusses how to start using MS-DOS and how to format and back up your disks.

Chapter 3 tells you about files - - what they are and how to use them. Chapters 4 through 6 introduce MS-DOS commands and Chapter 7 describes the line editor, EDLIN. Read these chapters carefully - - they contain information on protecting your data, system commands, and the MS-DOS editing commands.

Chapter 8 explains how to use the MS-DOS File Comparison utility, FC. This utility is helpful when you need to compare the contents of two source or binary files.

If you are writing programs and want to link separately-produced object modules and create relocatable modules, Chapter 9 describes a useful MS-DOS utility, MS-LINK.

Appendices to this manual include instructions if you are using MS-DOS and other operating systems on NCR DECISION MATE V, disk error messages, and special guidelines on how to run MS-DOS-compatible applications on your computer.

If you want to know more, a companion manual, the *PROGRAMMER'S MANUAL*, contains information on the technical aspects of MS-DOS. It also describes MS-DOS system architecture, additional utilities, and system calls and interrupts.

SYNTAX NOTATION

The following syntax notation is used throughout this manual in descriptions of command and statement syntax. Don't be overwhelmed by this list; after you use the commands a few times, the notation becomes quickly familiar.

[ ] square brackets
  Indicate that the enclosed entry is optional.

< > angle brackets
  Indicate that you supply the text for this entry. When the angle brackets enclose lowercase text, type in an entry defined by the text; for example, <filename>.

{ } braces
  Indicate that you have a choice between two or more entries. At least one of the entries enclosed in braces must be chosen unless the entries are also enclosed in square brackets.

. . . ellipses
  Indicate that an entry may be repeated as many times as needed or desired.

| a bar
  When used with an MS-DOS filter, the bar indicates a pipe. (This feature is fully explained in Chapter 4, Learning About Commands.)

CAPS capital letters
  Indicate portions of statements or commands that must be entered exactly as shown. Capital letters also indicate specific keys, such as <~~CR~~>.
  NEWLINE

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

FLEXIBLE/FIXED DISK SYSTEMS CONSIDERATIONS

To simplify explanations in this manual, examples are shown based on a multi-drive, flexible disk system. However, if you have a flexible/fixed disk system, you will use only the flexible disk drive to format and make copies of flexible disks. In these situations, MS-DOS always "prompts" you to change disks and waits for you to insert the new disk. You then continue processing by pressing any key.

Turn to the next chapter and learn how to start your MS-DOS system and how to format and back up your disks.

Only ok
Getting Started (2).

GETTING STARTED

SYSTEM SETUP

The MS-DOS *master disk* (or diskette), the one you received with this book, contains all the operating system software files and all commands. In this chapter, you learn how to install the software, switch processing from one disk to another, protect your master disk, and format other new disks. Finally, you'll read about files.

Before actually loading your software, you may need to know a little more about NCR DECISION MATE V and those all-important disks. Depending on your computer model, you have either a flexible disk system or a flexible/fixed disk system. The types of disks are not important to MS-DOS; the software only wants to know where to get and put information.

Regardless of which disk system you have, you always start processing from flexible disk drive A. Let's do that now by loading MS-DOS into memory.

LOADING MS-DOS

Not in new manual

Turn on your computer, insert the MS-DOS disk in drive A, and press the ↵ key. (This return key is also referred to as the <CR> key.) This is always the standard startup procedure. Depending on the size of memory, loading MS-DOS can take up to 25 seconds.

NEWLINE

Once MS-DOS is loaded, the system searches the MS-DOS disk for the COMMAND.COM file and loads it into memory. The COMMAND.COM file is a program that processes the commands you enter and then runs the appropriate programs. It is also called the *command processor.*

When the command processor is loaded, you see a copyright and software identification message on your screen. Have you read the "sign on" message, too? If so, you're right where you should be in this guide to begin using MS-DOS.

COPYING YOUR MASTER SOFTWARE DISK

You start using MS-DOS by making a backup copy of your master software disk. Actually, the software on your MS-DOS master disk begins the process automatically. You have only to help by answering simple questions or following simple directions that MS-DOS displays.

While the displays are self-explanatory, you may prefer following along with printed text. A summary of the copy procedure is shown in table form with your actions marked with a ✓ .

The first question asks how many flexible disk drives you have. Answer the question and then . . .

- Turn to page 2-3 if you have 2 flexible disk drives
- Turn to page 2-5 if you have 1 flexible disk drive

Copying the Master Diskette — Sequence Summary

2 FLEXIBLE DISK DRIVES

```
Formatting begins . . .

A> FORMAT B:

Insert new diskette for drive B:
and strike any key when ready
```

✓ **Press any key!**

```
xxxxxx bytes total disk space
xxxxxx bytes available on disk

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

✓ **Don't format any other disks now.
Press N (for no).**

```
Formatting complete
```

We're now ready to copy the master diskette in drive A to the diskette in drive B. Press any key when instructed to do so.

```
A> DISKCOPY A: B:/V

Insert source diskette into drive A:
Insert formatted target diskette into drive B:
Press any key when ready
```

✓ **Your disks are already in place.
Press any key!**

```
Copying . . .

Copying . . . Copying complete
```

cont.

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

✓ **Don’t copy any other disks now.**
Press N (for no).

Remove the master diskette from drive A and save for system protection. Take the diskette from drive B and place it in drive A.

Copying the Master Diskette — Sequence Summary

1 FLEXIBLE DISK DRIVE

```
Formatting begins . . .

A> FORMAT A:

Insert new diskette for drive A:
and strike any key when ready
```

✓ **Remove the master diskette and insert a new diskette; press any key when ready.**

```
xxxxxx bytes total disk space
xxxxxx bytes available on disk

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

✓ **Don't format any other disks now. Press N (for no).**

```
Formatting complete
```

✓ Remove the formatted diskette from drive A and again insert the master diskette into drive A.

During the following copy procedure, change diskettes as instructed until the 'copy complete' message appears.

```
A> DISKCOPY/V

Insert formatted target diskette into drive A:
Press any key when ready
```

✓ **Change diskettes; press any key when ready**

cont.

```
Copy complete

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

✓ **Don't copy any other disks now. Press N (for no).**

LEARNING ABOUT YOUR DRIVE DESIGNATIONS

When you copied your master disk, you were directed to insert or change a disk. You were instructed to do this by *drive designation.* For example, "Insert a disk into drive A."

The drive designation, which is always an alphabetic character, tells MS-DOS where to get and put information. No matter what types of disk units you have, each drive always has its own designation. Consider the following disk configuration examples:

- You have two flexible disk drives. One drive is designated (and labelled) A; the other drive is designated (and labelled) B.
- You have one flexible disk drive and one fixed disk unit. The flexible disk drive is designated (and labelled) A, but what about the fixed disk? *A fixed disk unit contains two logical disk drives.* (You can't see them, and they aren't labelled, but they're there). In this example configuration, the logical disk drives of the fixed disk unit are B and C.
- Now, assume you have two flexible disk drives and one fixed disk unit. What are the drive designations? The flexible disk drive designations are A and B; the fixed disk unit drive designations are C and D.

Drive designations are assigned by MS-DOS, which assumes you have 2 flexible disk drives. If you don't have this configuration you must describe your configuration to MS-DOS.

DEFINING YOUR FIXED DISKS

If you have fixed disks (sometimes called hard disks), you must continue setup procedures by defining your disk configuration. (If you have only a flexible disk system, you can skip this section.)

You define your disk configuration with the CONFIG routine. With self-explanatory screens, this routine is easy to use: You simply enter the name of the routine and follow the displays, carrying on a conversation with MS-DOS.

To make the procedure even simpler, however, table 1 summarizes the entire configure procedure. The left column contains the complete conversation. **What you say (enter)** is in bold type, what MS-DOS responds is in normal type, and *actions you must perform* are in italics. If you need some guidance when performing the sequence, look at the right-hand column for help.

Display/Enter/*Action*	Comments
A>	You're using a flexible/fixed disk system, but MS-DOS still "thinks" you have only flexible disks. Let's tell the software about your fixed disk with the CONFIG utility. Enter CONFIG and press ↵ .
A> **CONFIG** ↵	
- - - - - - - - - - - CONFIG UTILITY	Study this main menu screen for a minute. These functions can all be performed with CONFIG. (You'll learn about them in the "MS-DOS Commands" chapter.)
1) Modify Function Keys 2) Select Printer (Serial/Parallel) 3) Modify Retry/Restore Counter 4) Modify Serial Printer Interface	
5) Modify Disk Configuration	This function is the one you want.
6) Exit Program	
* Enter Function 5	Enter 5.
- - - - - - - - - - - CONFIG UTILITY	Another screen! You'll be using this one several times. Let's call it the *disk configuration* screen. First, modify the number of flexible disks.
Modify Disk Configuration	
1) Modify number of flexible disks	This function is the one you want.
2) Modify number of hard disks 3) Display configuration 4) Return to main menu	
* Enter Function 1	Enter 1.

Table 1 Defining Your Disk Configuration

cont.

Display/Enter/*Action*	Comments
CONFIG UTILITY	
Modify Disk Configuration Modify Number of Flexible Disks	Specify the number of flexible disks you have.
1) One Flex Disk 2) Two Flex Disks 3) Return to Main Program	This function is the one you want.
* Enter Function 1	Enter 1.
- - - - - - - - - - -	
CONFIG UTILITY	Now, modify the number of fixed disks.
Modify Disk Configuration	
1) Modify number of flexible disks 2) Modify number of hard disks 3) Display configuration 4) Return to main menu	This function is the one you want.
* Enter Function 2	Enter 2.

cont.

Display/Enter/*Action*	Comments
CONFIG UTILITY	Here you tell the software about your fixed disk.
Modify Disk Configuration Modify Number of Hard Disks	
1) No hard disk 2) One hard disk 3) Two hard disks 4) Three hard disks 5) Return to Main Program	This function is the one you want, assuming you have one fixed disk unit.
* Enter Function 2	Enter 2.
- - - - - - - - - - -	
CONFIG UTILITY Modify Disk Configuration 1) Modify number of flexible disks 2) Modify number of hard disks 3) Display configuration 4) Return to main menu	The disk configuration screen now appears. Select function 4. (If you want to see the the drive assignments, select function 3. Press ↵ to return to this screen.)
* Enter Function 4	Enter 4.

cont.

Display/Enter/*Action*	Comments
CONFIG UTILITY 1) Modify Function Keys 2) Select Printer (Serial/Parallel) 3) Modify Retry/Restore Counter 4) Modify Serial Printer Interface 5) Modify Disk Configuration	You've finished configuring.
6) Exit Program	This function is the one you want.
* Enter Function 6	Enter 6.
- - - - - - - - - - -	
1 Update O.S. disk in drive A 2 Return to main program 3 Exit CONFIG	The exit program function is important. Here, you make the changes permanent by having them written to disk.
ATTENTION: Changes to the disk configuration must be written to disk (permanent) and must be followed by a restart. Update the disk, exit CONFIG, and then turn off and on the computer when the system prompt appears.	Read the "attention" and then perform the following sequence:
* Enter Function 1	Enter 1 (You'll "hear" the changes being written to disk.)
* Enter Function 3	Enter 3 (You're leaving the CONFIG utility.)
A>	
Switch your computer off and on. Complete setup procedures as described in the following text.	Be sure to do this!

NOTE: The sequence described assumes one flexible disk drive and one fixed disk unit. Adjust your entries for your specific configuration.

FORMATTING DISKS

You've already done a lot with your DECISION MATE V and MS-DOS. You've protected your software by making a copy of the master disk and, if you have a fixed disk, defined the configuration. You also saw how to format a disk. Remember, though that *each new disk must be formatted.*

Your fixed disks, for example, are not yet formatted for MS-DOS. Because the FORMAT routine is described in detail in chapter 5, its description is not repeated here; however, a couple of comments about formatting fixed disks must be noted.

- To format one logical drive of a fixed disk unit takes approximately 20 minutes. This time is calculated based on the "standard" number of 5 certifications (read and write checks). The formula is 4 mins + (3 min. x # of certifications). You have an option of increasing or decreasing the number of certifications before formatting begins.
- Before you format a fixed disk, always check that the logical disk drive hasn't already been formatted by MS-DOS or some other operating system. Use a command like MS-DOS CHKDSK to first determine the contents of the disk.

DEFINING A SERIAL PRINTER

Are you using a printer? MS-DOS assumes it is a parallel printer. If using a serial printer, you must define it to MS-DOS with the CONFIG utility. Because printer requirements vary, refer again to chapter 5 for a full description of CONFIG.

You've now completed all setup procedures. In the next sections of this chapter, you learn some more operating procedures and about files.

FREQUENTLY PERFORMED OPERATIONS

ENTERING THE DATE AND TIME

When you load MS-DOS into memory or restart your computer, you see the date and time prompts. You should enter this information which is extremely helpful in keeping track of when you created or updated data on your disk.

When you see:

```
Enter new date: _
```

Type today's date in an mm-dd-yy format, where:

- mm is a 1- or 2-digit number from 1-12 (representing month)
- dd is a 1- or 2-digit number from 1-31 (representing the day of the month)
- yy is a 2-digit number from 80-99 (the 19 is assumed), or a 4-digit number from 1980-2099 (representing year)

Any date is acceptable in answer to the new date prompt as long as it follows the above format. Separators between the numbers can be hyphens (-) or slashes (/). For example:

```
5-1-83 or 05/01/83
```

are both acceptable answers to the "Enter new date:" prompt.

If you enter an invalid date or form of date, the system prompts you again with "Enter new date:".

After you respond to the new date prompt and enter your answer by pressing the <CR> key, you see a prompt similar to this:

```
Current time is 0.00:00.00
Enter new time: _
```

Enter the current time in the hh:mm format, where:

- hh is a 1- or 2-digit number from 0-23 (representing hours)
- mm is a 1- or 2-digit number from 0-59 (representing minutes)

MS-DOS uses this time value to keep track of when you last updated and/or created files on the system. Notice that MS-DOS uses military time; for instance, 1:30 p.m is written 13:30.

Example:

```
Current time is 0:00:00.00
Enter new time: 9:05
```

Only use the colon (:) to separate hours and minutes. If you enter an invalid number separator, MS-DOS repeats the prompt.

NOTE: If you make a mistake while typing, press the CONTROL key on your keyboard, hold it down, and then press the C key. The <CONTROL-C> function aborts your current entry. You can then re-answer the prompt or type another command. To correct a line before you press <CR>, use the <BACKSPACE> key to erase one letter at a time.

CHANGING THE DEFAULT DRIVE

The A> is the MS-DOS prompt from the command processor. It tells you that MS-DOS is ready to accept commands.

The A in the previous prompt is the *default* disk drive. This means that MS-DOS searches only the disk in drive A for any filenames you enter and writes files to that disk unless you specify a different drive. You can ask MS-DOS to search a disk in another drive by changing the drive designation or by specifying it in a command. To change the disk drive designation, enter the new drive letter followed by a colon. For example:

```
A>
A>B: <CR>    (you have typed B: in response to the prompt)
B>
```

The system prompt B> appears and drive B is now the default drive. MS-DOS searches only the disk in drive B until you specify a different default drive. To move back to drive A, simply specify A:. (Don't forget the colon.)

```
A>B:
B>A: <CR>
A>
```

BACKING-UP YOUR DISKS

You've made a backup copy of your master software disk; you should make backup copies of all your disks. If a disk becomes damaged or if files are accidentally erased, you will still have all of the information on your backup disk.

You make backup copies of flexible disks with the DISKCOPY command; you make backup copies of fixed disks with the

BACKUP command. (Both of these commands are discussed in detail in Chapter 5, MS-DOS Commands.)

USING THE PROGRAMMABLE FUNCTION KEYS

Your NCR DECISION MATE V has a row of special keys. These keys are labelled F1 through F20 and are located on the top row of the keyboard. They are special because you can define (program) them to do any function you want.

Like the automatic-program-execution feature (see next section), the programmable function keys are convenient, especially for performing an often-used or difficult function. For example, you may always want to check the contents of a disk before you access it. You could assign the directory display (DIR) command to a function key. Then, to use the command, you could simply press the key instead of typing the command through the keyboard.

Function keys are defined with the MS-DOS CONFIG utility. (See Chapter 5.)

RUNNING PROGRAMS AUTOMATICALLY

If you want to run a specific program automatically each time you start MS-DOS, you can do so with Automatic Program Execution. For example, you may want to have MS-DOS display the names of your files each time you load MS-DOS.

When you start MS-DOS, the command processor searches for a file named AUTOEXEC.BAT on the MS-DOS disk. This file is a program that MS-DOS will run each time MS-DOS is started. Chapter 4, Learning About Commands, tells you how to create an AUTOEXEC.BAT file.

TURNING THE SYSTEM OFF

There is no "logoff" command in MS-DOS. To end your terminal session, open the disk drive doors and remove the disks. Then, simply turn your terminal off in response to a default drive prompt.

FILES

A file is a collection of related information. A file on your disk can be compared to a file folder in a desk drawer. For example, one file folder might contain the names and addresses of the employees who work in the office. You might name this file the Employee Master File. A file on your disk could also contain the names and addresses of employees in the office and could also be named Employee Master File.

All programs, text, and data on your disk reside in files and each file has a unique name. You refer to files by their names. Chapter 3, More About Files, tells you how to name your files.

You create a file each time you enter and save data or text at your terminal. Files are also created when you write and name programs and save them on your disks.

HOW MS-DOS KEEPS TRACK OF YOUR FILES

The names of files are kept in directories on a disk. These directories also contain information on the size of the files, their location on the disk, and the dates that they were created and updated. The directory you are working in is called your current or *working* directory.

An additional system area is called the File Allocation Table. It keeps track of the location of your files on the disk. It also allocates the free space on your disks so that you can create new files.

These two system areas, the directories and the File Allocation Table, enable MS-DOS to recognize and organize the files on your disks. The File Allocation Table is copied to a new disk when you format it with the MS-DOS FORMAT command; also, one empty directory is created, called the *root* directory.

THE DIR (SHOW DIRECTORY) COMMAND

If you want to know what files are on your disk, you can use the DIR command. This command tells MS-DOS to display all the files in the current directory on the disk that is named. For example, if your MS-DOS disk is in drive A and you want to see the listing for the current directory on that disk, type:

```
    DIR A:<CR>
```

MS-DOS responds with a directory listing of all the files in the current directory on your MS-DOS disk. To stop the screen to study the files, press the CONTROL key, hold it down, and then press the C key. To continue the display, press any key.

NOTE: Two MS-DOS system files, IO.SYS and MSDOS.SYS, are "hidden" files and do not appear when you issue the DIR command.

You can also get information about any file on your disk by typing DIR and a filename. For example, if you have created a file named MYFILE.TXT, the command

NEWLINE

DIR MYFILE.TXT <~~CR~~>

gives you a display of all the directory information (name of file, size of file, date last edited) for the file MYFILE.TXT.

For more information on the DIR command, refer to Chapter 5, MS-DOS Commands.

CHECKING YOUR DISKS

The MS-DOS command CHKDSK is used to check your disks for consistency and errors, much like a secretary proofreading a letter. CHKDSK analyzes the directories and the File Allocation Table on the disk that you specify. It then produces a status report of any inconsistencies, such as files which have a non-zero size in their directory but really have no data in them.

To check the disk in drive A, type:

NEWLINE

CHKDSK A: <~~CR~~>

MS-DOS displays a status report and any errors that it has found. An example of this display and more information on CHKDSK can be found in the description of the CHKDSK command in Chapter 5. You should run CHKDSK occasionally for each disk to ensure the integrity of your files.

CHAPTER REVIEW

- Always begin processing from disk drive A. Turn on your computer, insert a master disk in drive A, and press <CR>.
- The date and time messages are displayed whenever MS-DOS is read into memory. Although these messages can be bypassed, they provide important information about when data was created or updated on your disk.
- The MS-DOS master disk that came with this manual is *write protected.* You made a backup copy of the diskette and should only use the new copy for processing. (Put the original master diskette in a safe place for system protection.)
- On your MS-DOS master diskette, the software "thinks" you have two flexible disk drives and a parallel printer. To "tell" MS-DOS differently, you must use the CONFIG utility to define and write the configuration description on the new master diskette.
- Each disk drive has a unique name (drive designation), which is an alphabetic character. A fixed disk unit has two logical disk drives and, therefore, two drive designations.
- The A> is the system default prompt. It tells you which disk drive MS-DOS is using and that MS-DOS is waiting for your direction. You can change the default drive designation by entering the drive designation followed by a colon (:).
- Most entries you make on the keyboard must end by pressing the <CR> key. This function key tells MS-DOS you have completed an entry.
- Any new disk must be formatted with the FORMAT command before it can be used by MS-DOS. Because of the high storage capacity of a fixed disk, formatting it can take several minutes.
- All data on your disk is stored in files and each file name is listed in a directory. The DIR command displays the directory.
- Always make a copy of important data on your disk using the appropriate MS-DOS copy commands: DISKCOPY, COPY, or BACKUP.
- Keys F1-F20 on the top row of your computer are available for your own use. You can "program" them (with the CONFIG utility) to do anything you want.
- MS-DOS has an automatic program execution feature. Whenever you load MS-DOS, the defined program is automatically executed.
- When you have finished processing, remove the flexible disk and then turn off your computer.

MORE ABOUT FILES

In Chapter 2, you learned that directories contain the names of your files. In this chapter, you learn how to name and copy your files. You also learn more about the MS-DOS hierarchical directory structure that makes it easy for you to organize and locate your files.

NAMING YOUR FILES

The name of a typical MS-DOS file looks like this:

```
NEWFILE.EXE
```

The name of a file consists of two parts. The filename is NEWFILE and the filename extension is .EXE.

A filename can be from 1 to 8 characters long. The filename extension can be three or fewer characters. You can type any filename in small or capital letters and MS-DOS will translate these letters into uppercase characters.

In addition to the filename and the filename extension, the name of your file may include a drive designation. A drive designation tells MS-DOS to look on the disk in the designated drive to find the filename typed. For example, to find directory information about the file NEWFILE.EXE which is located on the disk in drive A (and drive A is NOT the default drive), type the following command:

```
DIR A:NEWFILE.EXE
```

Directory information about the file NEWFILE.EXE is now displayed on your screen.

If drive A is the default drive, MS-DOS will search only the disk in drive A for the filename NEWFILE and so the drive designation is not necessary. A drive designation is needed if you want to tell MS-DOS to look on the other drive to find a file.

Your filenames will probably be made up of letters and numbers, but other characters are allowed, too. Legal characters for filename extensions are the same as those for filenames. Here is a complete list of the characters you can use in filenames and extensions:

```
A - Z    0 - 9    $    &    #

%        '        (    )    -    @

\        ^        [    ]    ~    `    !
```

All of the parts of a filename comprise a file specification. The term file specification (or filespec) is used in this manual to indicate the following filename format:

```
[<drive designation:>] <filename> [<.filename extension>]
```

Remember that brackets indicate optional items. Angle brackets (< >) mean that you supply the text for the item. Note that the drive designation is not required unless you need to indicate to MS-DOS on which disk to search for a specific file. You do not have to give your filename a filename extension.

Examples of file specifications are:

```
B:MYPROG.COB
A:YOURPROG.EXT
A:NEWFILE.
TEXT
```

WILD CARDS

Two special characters (called wild cards) can be used in filenames and extensions: the asterisk (*) and the question mark (?). These special characters give you greater flexibility when using filenames in MS-DOS commands.

The ? Wild Card

A question mark (?) in a filename or filename extension indicates that any character can occupy that position. For example, the MS-DOS command

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

lists all directory entries on the default drive that have 8 characters, begin with TEST, have any next character, end with the letters RUN, and have a filename extension of .EXE. Here are some examples of files that might be listed by the previous DIR command:

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

The * Wild Card

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

```
DIR TEST*.EXE
```

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

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

The wild card designation *.* refers to all files on the disk. Note that this designation can be very powerful and destructive when used in MS-DOS commands. For example, the command DEL *.* deletes all files on the default drive, regardless of filename or extension.

Examples

To list the directory entries for all files named NEWFILE on drive A (regardless of filename extensions), simply type:

```
DIR A:NEWFILE.*
```

To list the directory entries for all files with filename extensions of .TXT (regardless of filenames) on the disk in drive B, type:

```
DIR B:????????.TXT
```

This command is useful if, for example, you have given all your text programs a filename extension of .TXT. By using the DIR commands with the wild card characters, you can obtain a listing of all your text files even if you do not remember all of their filenames.

ILLEGAL FILENAMES

MS-DOS treats some device names specially, and certain ~~3-letter~~ names are reserved for the names of these devices. The following ~~3-letter~~ names cannot be used as filenames or extensions.

AUX or COM1, COM2, COM3, COM4, COM5 RS232

Used when referring to input from or output to an ~~auxiliary device (such as a printer or disk drive).~~ communications line.

CON

Used when referring to keyboard input or to output to the terminal console (screen).

~~LST or~~ PRN or LPT1, LPT2, LPT3, LPT4, LPT5

Used when referring to the printer device.

NUL

Used when you do not want to create a particular file, but the command requires an input or output filename.

CLOCK$ used when referring to the clock driver.

Even if you add device designations or filename extensions to these filenames, they remain associated with the devices listed above. For example, A:CON.XXX always refers to the console and is not the name of a disk file.

COPYING YOUR FILES

Just as with paper files, you often need more than one copy of a disk file. The COPY command allows you to copy one or more files to another disk. You can also give the copy a different name if you specify the new name in the COPY command.

The COPY command can also make copies of files on the same disk. In this case, you must assign a different filename or you will overwrite the file. You cannot make a copy of a file on the same disk unless you specify a different filename for the new copy.

The format of the COPY command is:

```
COPY filespec [filespec]
```

For example,

```
COPY A:MYFILE.TXT  B:MYFILE.TXT
```

copies the file MYFILE.TXT on disk A to a file named MYFILE.TXT on disk B. A duplicate copy of MYFILE.TXT now exists.

Figure 3.1 illustrates how to copy files to another disk:

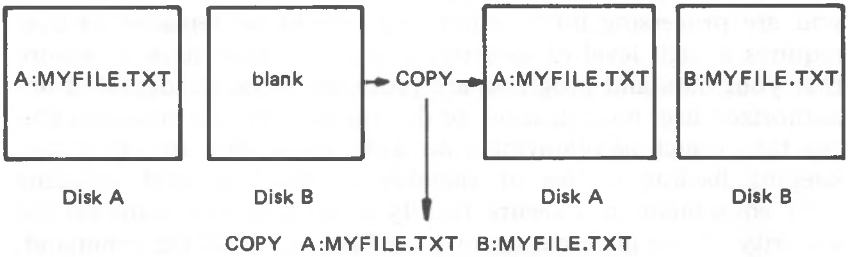


Figure 3.1 Copying files to another disk

If you want to duplicate the file named MYFILE. TXT on the same disk, type:

```
COPY A:MYFILE.TXT  A:NEWNAME.TXT
```

You now have two copies of your file on disk A, one named MYFILE.TXT and the other named NEWNAME.TXT. The following figure illustrates this example.

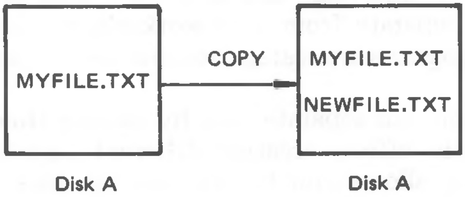


Figure 3.2 Copying files on the same disk

You can also copy all files on a disk to another disk (that is, make a backup copy) with the COPY command. Refer to Chapter 5, MS-DOS Commands, for more information on this process.

PROTECTING YOUR FILES

MS-DOS is a powerful and useful tool in processing your personal and business information. As with any information system, inadvertent errors may occur and information may be misused. If you are processing information that cannot be replaced or that requires a high level of security, you should take steps to ensure that your data and programs are protected from accidental or unauthorized use, modification, or destruction. Simple measures you can take - such as removing your disks when they are not in use, keeping backup copies of valuable information, and installing your equipment in a secure facility - can help you maintain the integrity of the information in your files. An MS-DOS command, CIPHER, can also be used to encrypt your files for total privacy. For more information on CIPHER, refer to Chapter 5, MS-DOS Commands.

DIRECTORIES

As you learned in Chapter 2, the names of your files are kept in a directory on each disk. The directory also contains information on the size of the files, their locations on the disk, and the dates that they were created and updated.

When there are multiple users on your computer, or when you are working on several different projects, the number of files in the directory can become large and unwieldy. You may want your own files kept separate from a co-worker's, or you may want to organize your programs into categories that are convenient for you.

In an office, you can separate files by putting them in different filing cabinets; in effect, creating different directories of information. MS-DOS allows you to organize the files on your disks into directories. Directories are a way of dividing your files into convenient groups of files. For example, you may want all of your accounting programs in one directory and text files in another. Any one directory can contain any reasonable number of files, and it may also contain other directories (referred to as subdirectories). This method of organizing your files is called a hierarchical directory structure.

A hierarchical directory structure can be thought of as a "tree" structure: directories are branches of the tree and files are the leaves, except that the "tree" grows downward; that is, the "root" is at the top. The root is the first level in the directory structure. It is the directory that is automatically created when you format a disk and start putting files in it. You can create additional directories and subdirectories by following the instructions in Chapter 4, Learning About Commands.

The tree or file structure grows as you create new directories for groups of files or for other people on the system. Within each new directory, files can be added, or new subdirectories can be created.

It is possible for you to "travel" around this tree; for instance, it is possible to find any file in the system by starting at the root and traveling down any of the branches to the desired file. Conversely, you can start where you are within the file system and travel towards the root.

The filenames discussed earlier in this chapter are relative to your current directory and do not apply system-wide. Thus, when you turn on your computer, you are "in" your directory. Unless you take special action when you create a file, the new file is created in the directory in which you are now working. Users can have files of the same name that are unrelated because each is in a different directory.

Figure 3.3 illustrates a typical hierarchical directory structure.

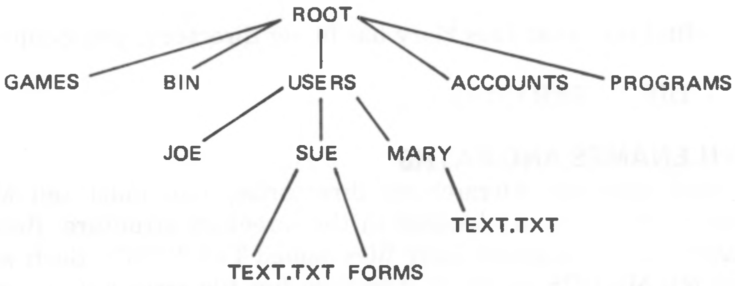


Figure 3.3 A sample hierarchical directory structure

The ROOT directory is the first level in the directory structure. You can create subdirectories from the ROOT by using the MKDIR command (refer to Chapter 5, MS-DOS Commands, for information on MKDIR). In this example, five subdirectories of ROOT have been created. These include:

- A directory of games, named GAMES
- A directory of all external commands, named BIN (refer to Chapter 4, Learning About Commands, for more information on the BIN directory)
- A USER directory containing separate subdirectories for all users of the system
- A directory containing accounting information, named ACCOUNTS
- A directory of programs, named PROGRAMS

Joe, Sue, and Mary each have their own directories which are subdirectories of the USER directory. Sue has a subdirectory under the \USER\SUE directory named FORMS. Sue and Mary have files in their directories, each named TEXT.TXT. Notice that Mary's text file is unrelated to Sue's.

This organization of files and directories is not important if you only work with files in your own directory, but, if you work with someone else or on several projects at one time, the hierarchical directory structure becomes extremely useful. For example, you could get a list of the files in Sue's FORMS directory by typing:

```
DIR \USER\SUE\FORMS
```

Note that the back slash mark (\) is used to separate directories from other directories and files.

To find out what files Mary has in her directory, you could type:

```
DIR \USER\MARY
```

FILENAMES AND PATHS

When you use hierarchical directories, you must tell MS-DOS where the files are located in the directory structure. Both Mary and Sue, for example, have files named TEXT.TXT. Each will have to tell MS-DOS in which directory her file resides if she wants to access it. This is done by giving MS-DOS a pathname to the file.

Pathnames

A simple filename is a sequence of characters that can optionally be preceded by a drive designation and followed by an extension. A pathname is a sequence of directory names followed by a simple filename, each separated from the previous one by a slash (\ ).

The syntax of pathnames is:

[<d>:] [<directory>] \ [<directory. . . >] \ [<filename>]

If a pathname begins with a slash, MS-DOS searches for the file beginning at the root (or top) of the tree; otherwise, MS-DOS begins at the user's current directory, known as the working directory, and searches downward from there. The pathname of Sue's TEXT.TXT file is \ USER \ SUE \ TEXT. TXT.

When you are in your working directory, a filename and its corresponding pathname may be used interchangeably. The following list shows some sample names:

\
Indicates the root directory.

\ PROGRAMS
Sample directory under the root directory containing program files.

\ USER \ MARY \ FORMS \ 1A
A typical full pathname. This one happens to be a file named 1A in the directory named FORMS belonging to the USER named MARY.

USER \ SUE
A relative pathname; it names the file or directory SUE in subdirectory USER of the working directory. If the working directory is the root (\ ), it names \ BIN \ SUE.

TEXT.TXT
Name of a file or directory in the working directory.

MS-DOS provides special shorthand notations for the working directory and the parent directory (one level up) of the working directory: For example in the sample of an hierarchical directory structure in this chapter, the parent of the directory JOE is USER.

. (single period)
MS-DOS uses this shorthand notation to indicate the name of the working directory in all hierarchical directory listings.

MS-DOS automatically creates this entry when a directory is made.

. . (double period)

The shorthand name of the working directory's parent directory. If you type:

```
DIR . .
```

then MS-DOS lists the files in the parent directory of your working directory. If you type:

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

then MS-DOS lists the files in the parent's PARENT directory.

Pathing and External Commands - External commands reside on disks as program files. They must be read from the disk before they execute. (For more information on external commands, refer to Chapter 4, Learning About Commands.)

When you are working with more than one directory, it is convenient to put all MS-DOS external commands into a separate directory so they do not clutter your other directories. When you issue an external command to MS-DOS, MS-DOS immediately checks your working directory to find that command. You must tell MS-DOS in which directory these external commands reside. This is done with the PATH command.

For example, if you are in a working directory named \BIN\PROG, and all MS-DOS external commands are in \BIN, you must tell MS-DOS to choose the \BIN path to find the FORMAT command. The command

```
PATH \BIN
```

tells MS-DOS to search in your working directory and the \BIN directory for all commands. You only have to specify this path once to MS-DOS during your terminal session. MS-DOS will now search in \BIN for the external commands. If you want to know what the current path is, type the word PATH and the current value of PATH will be printed.

For more information on the MS-DOS command PATH, refer to Chapter 5, MS-DOS Commands.

Pathing and Internal Commands - Internal commands are the simplest, most commonly used commands. They execute immediately because they are incorporated into the command processor. (For more information on internal commands, refer to Chapter 4, Learning About Commands.)

Some internal commands can use paths. The four commands, COPY, DIR, DEL, and TYPE, have greater flexibility when you specify a pathname after the command.

COPY <pathname pathname>
If the second pathname to COPY is a directory, all files are copied into that directory. The first pathname may only specify files in the working directory.

DEL <pathname>
If the pathname is a directory, all the files in that directory are deleted. Note: The prompt "Are you sure (Y/N)?" is displayed if you try to delete a path. Type Y to complete the command, or type N for the command to abort.

DIR <pathname>
Displays the directory for a specific path.

TYPE <pathname>
You must specify a file in a path for this command. MS-DOS will display the file on your screen in response to the TYPE pathname command.

DISPLAYING YOUR WORKING DIRECTORY

All commands are executed while you are in your working directory. You can find out the name of the directory you are in by issuing the MS-DOS command CHDIR (Change Directory) with no options. For example, if your current directory is \USER\JOE, when you type:

NEWLINE
CHDIR<~~RETURN~~>

you will see:

A: \USER\JOE

This is your current drive designation plus the working directory (\USER\JOE).

If you now want to see what is in the \USER\JOE directory, you can issue the MS-DOS command DIR. The following is an example of the display you might receive from the DIR command for a subdirectory:

```
Volume in drive A has no ID
Directory of A: \USER\JOE

.               <DIR>        5-09-83  10:09a
. .             <DIR>        5-09-83  10:09a
TEXT            <DIR>        5-09-83  10:09a
FILE1    COM           5243  5-04-83   9:30a
        4 File(s)     250518 bytes free
```

A volume ID for this disk was not assigned when the disk was formatted. Note that MS-DOS lists both files and directories in this output. As you can see, Joe has another directory in this tree structure named TEXT. The '.' indicates the working directory \USER\JOE, and '. .' is the shorthand notation for the parent directory \USER. FILE1. COM is a file in the \USER\JOE directory. All of these directories and files reside on the disk in drive A.

Because files and directories are listed together (see previous display), MS-DOS does not allow you to give a subdirectory the same name as a file in that directory. For example, if you have a path \BIN\ USER\JOE where JOE is a subdirectory, you cannot create a file in the USER directory named JOE.

CREATING A DIRECTORY

To create a subdirectory in your working directory, use the MKDIR (Make Directory) command. For example, to create a new directory named NEWDIR under your working directory, simply type:

```
MKDIR NEWDIR
```

After this command is executed by MS-DOS, a new directory exists in your tree structure under your working directory. You can also make directories anywhere in the tree structure by specifying MKDIR and then a pathname. MS-DOS automatically creates the . and . . entries in the new directory.

To put files in the new directory, use the MS-DOS Line Editor, EDLIN. Chapter 7, Line Editor (EDLIN), describes how to use EDLIN to create and save files.

CHANGING YOUR WORKING DIRECTORY

Changing from your working directory to another directory is very easy in MS-DOS. Simply issue the CHDIR (Change Directory) command and supply a pathname. For example:

```
A:CHDIR \USER
```

changes the working directory from \USER\JOE to \USER. You can specify any pathname after the command to "travel" to different branches and leaves of the directory tree. The command "CHDIR . ." will always put you in the parent directory of your working directory.

REMOVING A DIRECTORY

To delete a directory in the tree structure, use the MS-DOS RMDIR (Remove Directory) command. For example, to remove the directory NEWDIR from the working directory, type:

```
RMDIR NEWDIR
```

Note that the directory NEWDIR must be empty except for the . and . . entries before it can be removed; this will prevent you from accidentally deleting files and directories. You can remove any directory by specifying its pathname. To remove the \BIN\USER\JOE directory, make sure that it has only the . and . . entries, then type:

```
RMDIR \BIN\USER\JOE
```

To remove all the files in a directory (except for the . and . . entries), type DEL and then the pathname of the directory. For example, to delete all files in the \BIN\USER\SUE directory, type:

```
DEL \BIN\USER\SUE
```

You cannot delete the . and . . entries. They are created by MS-DOS as part of the hierarchical directory structure.

In the next chapter, you will learn about MS-DOS commands.

CHANGING YOUR WORKING DIRECTORY

Changing from your working directory to another directory is very easy in MS-DOS. Simply use the CHDIR (Change Directory) command and supply a pathname. For example:

```
A>CHDIR \USER
```

changes the working directory from \USER\JOE to \USER. You can specify any pathname after the command to "travel" to different branches and leaves of the directory tree. The command "CHDIR .." will always put you in the parent directory of your working directory.

REMOVING A DIRECTORY

To delete a directory in the tree structure, use the MS-DOS RMDIR (Remove Directory) command. For example, to remove the directory NEWDIR from the working directory, type:

```
RMDIR NEWDIR
```

Note that the directory NEWDIR must be empty except for the . and .. entries before it can be removed; this will prevent you from accidentally deleting files and directories. You can remove any directory by specifying its pathname. To remove the \BIN\USER\JOE directory, make sure that it has only the . and .. entries, then type:

```
RMDIR \BIN\USER\JOE
```

To remove all the files in a directory (except for the . and .. entries), type DEL and then the pathname of the directory. For example, to delete all files in the \BIN\USER\JOE directory, type:

```
DEL \BIN\USER\JOE
```

You cannot delete the root directory. It is always present in MS-DOS to maintain the hierarchical directory structure.

In the next chapter, you will learn [illegible]

LEARNING ABOUT COMMANDS

GENERAL INFORMATION

Commands are a way of communicating with the computer. By entering MS-DOS commands at your terminal, you can ask the system to perform useful tasks:

- Compare, copy, display, delete, and rename files
- Copy and format disks
- Execute system programs such as EDLIN, as well as your own programs
- Analyze and list directories
- Enter date, time, and remarks
- Set various printer and screen options
- Copy MS-DOS system files to another disk
- Request MS-DOS to wait for a specific period of time

TYPES OF MS-DOS COMMANDS

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

Internal Commands

Internal commands are the simplest, most commonly used commands. You cannot see these commands when you do a directory listing on your MS-DOS disk; they are part of the command processor. When you type these commands, they execute immediately. The following internal commands are described in Chapter 5:

BREAK
CHDIR (CD)
CLS
COPY
CTTY
DATE
DEL (ERASE)
DIR
ECHO
EXIT
FOR
GOTO
IF
MKDIR (MD)
PATH
PAUSE
PROMPT
REM
REN (RENAME)
RMDIR (RD)
SET
SHIFT
TIME
TYPE
VER
VERIFY
VOL

External Commands

External commands reside on disk as program files. They must be read from disk before they can execute. If the disk containing the command is not in the drive, MS-DOS will not be able to find and execute the command.

Any filename with a filename extension of .COM, .EXE or .BAT is considered an external command. For example, the program FORMAT.COM is an external command. Because all external commands reside on disk, you can create commands and add them to the system. Programs that you create with most languages (including assembly language) will be .EXE (executable files).

When you enter an external command, do not include its filename extension. The following external commands are described in Chapter 5:

ASSIGN
BACKUP
CHKDSK
CIPHER
CONFIG
DISCOMP
DISKCOPY
ESC
FIND
FORMAT

LOCATE
MODE
MORE
PRINT
RDCPM
RECOVER
RESTORE
SORT
SAVEFK
SYS
TREE

COMMAND OPTIONS

Options can be included in your MS-DOS commands to specify additional information to the system. If you do not include some options, MS-DOS provides a default value. Refer to individual command descriptions in Chapter 5 for the default values.

The following is the format of all MS-DOS commands:

```
    Command [options. . .]
```

where:

d:
    Refers to the disk drive designation.

filename
    Refers to any valid name for a disk file, including an optional filename extension. The filename option does not refer to a device or to a disk drive designation.

.ext
    Refers to an optional filename extension consisting of a period and 1-3 characters. When used, filename extensions immediately follow filenames.

filespec

Refers to an optional drive designation, a filename, and an optional three letter filename extension in the following format:

```
[<d:>] <filename> [<.ext>]
```

pathname

Refers to a pathname or filename in the following format:

```
[<directory>] \ [<directory. . .>] \ [<filename>]
```

switches

Switches are options that control MS-DOS commands. They are preceded by a forward slash (for example, /P).

arguments

Provide more information to MS-DOS commands. You usually choose between arguments; for example, ON or OFF.

COMMON ENTRY CONVENTIONS

The following information applies to all MS-DOS commands:

1. Commands are usually followed by one or more options.
2. Commands and options may be entered in uppercase of lowercase, or a combination of keys.
3. Commands and options must be separated by delimiters. Because they are easiest, you will usually use the space and comma as delimiters. For example:

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

   You can also use the semicolon (;), the equal sign (=), or the tab key as delimiters in MS-DOS commands. (In this manual, we use a space as the delimiter in commands.)
4. Do not separate a file specification with delimiters, since the colon and the period already serve as delimiters.
5. When instructions say "Press any key," you can press any alpha (A-Z) or numeric (0-9) key.
6. You must include the filename extension when referring to a file that already has a filename extension.
7. You can abort commands when they are running by pressing <CONTROL-C>.

NEWLINE

8. Commands take effect only after you have pressed the <CR> key.
9. Wild cards (global filename characters) and device names (for example, PRN or CON) are not allowed in the names of any commands.
10. When commands produce a large amount of output on the screen, the display automatically scrolls to the next screen. You can press <CONTROL-S> to suspend the display. Press any key to resume the display on the screen.
11. MS-DOS editing and function keys can be used when entering commands. Refer to Chapter 6, MS-DOS Editing and Function Keys, for a complete description of these keys.
12. The prompt from the command processor is the default drive designation plus a greater-than (>) sign; for example, A>.
13. Disk drives will be referred to as source drives and destination drives. A source drive is the drive you transfer information from; a destination drive is the drive you transfer information to.

BATCH PROCESSING

Often you may find yourself typing the same sequence of commands over and over to perform some commonly used task. With MS-DOS, you can put the command sequence into a special file, called a batch file, and execute the entire sequence simply by typing the name of the batch file. "Batches" of your commands in such files are processed as if they were typed at a terminal. Each batch file must be named with the .BAT extension, and is executed by typing the filename without its extension.

You can create a batch file by using the Line Editor (EDLIN) or by typing the COPY command. Refer to the Creating an AUTOEXEC.BAT File section later in this chapter for more information on using the COPY command to create a batch file.

Two MS-DOS commands are available for use expressly in batch files: REM and PAUSE. REM permits you to include remarks and comments in your batch files without these remarks being executed as commands. PAUSE prompts you with an optional message and permits you to either continue or abort the batch process at a given point. REM and PAUSE are described in detail in Chapter 5.

Batch processing is useful if you want to execute several MS-DOS commands with one batch command, such as when you format and check a new disk. For example, a batch file for this purpose might look like this:

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

To execute this .BAT file, simply type the filename without the .BAT extension:

```
NEWDISK
```

The result is the same as if each of the lines in the .BAT file was entered at the terminal as individual commands.

Figure 4.1 illustrates the 3 steps used to write, save, and execute an MS-DOS batch file.

The following list contains information that you should read before you execute a batch process with MS-DOS.

1. Do not enter the filename BATCH (unless the name of the file you want to execute is BATCH.BAT).
2. Enter only the filename to execute the batch file; do not enter the filename extension.
3. The commands in the file named <filename>.BAT are executed.
4. If you press <CONTROL-C> while in batch mode, this prompt appears:

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

   If you press Y, the remainder of the commands in the batch file are ignored and the system prompt appears.

   If you press N, only the current command ends and batch processing continues with the next command in the file.

5. If you remove the disk containing a batch file being executed, MS-DOS prompts you to insert it again before the next command can be read.
6. The last command in a batch file may be the name of another batch file. This allows you to call one batch file from another when the first is finished.

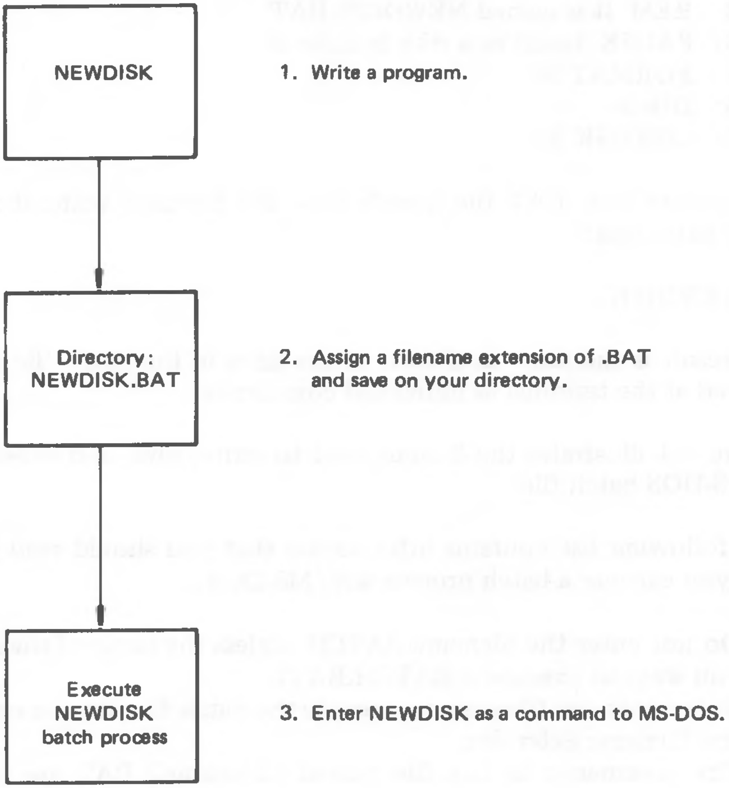

NEWDISK
1. Write a program.
Directory: NEWDISK.BAT
2. Assign a filename extension of .BAT and save on your directory.
Execute NEWDISK batch process
3. Enter NEWDISK as a command to MS-DOS.

Figure 4.1 MS-DOS batch file steps

THE AUTOEXEC.BAT FILE

As discussed in Chapter 2, an AUTOEXEC.BAT file allows you to automatically execute programs when you start MS-DOS. Automatic Program Execution is useful when you want to run a specific application package under MS-DOS, or when you want MS-DOS to execute a batch program automatically each time you start the system. You can avoid loading two separate disks to perform either of these tasks by using an AUTOEXEC.BAT file.

When you start MS-DOS, the command processor searches the MS-DOS disk for a file named AUTOEXEC.BAT. The AUTOEXEC.BAT file is a batch file that is automatically executed each time you start the system.

If MS-DOS finds the AUTOEXEC.BAT file, the file is immediately executed by the command processor and the date and time prompts are bypassed.

If MS-DOS does not find an AUTOEXEC.BAT file when you first load the MS-DOS disk, then the date and time prompts are issued. Figure 4.2 illustrates how MS-DOS uses the AUTOEXEC.BAT file.

CREATING AN AUTOEXEC.BAT FILE

To see how to create an AUTOEXEC.BAT file, assume that each time you start MS-DOS, you want to automatically load BASIC and run a program called MENU. You could create an AUTOEXEC.BAT file as follows:

1. Type:

   ```
   COPY CON: AUTOEXEC.BAT
   ```

   This statement tells MS-DOS to copy the information from the console (keyboard) into the AUTOEXEC.BAT file. Note that the AUTOEXEC.BAT file must be created in the root directory of your MS-DOS disk.
2. Now type:

   ```
   BASIC MENU
   ```

   This statement goes into the AUTOEXEC.BAT file. It tells MS-DOS to load BASIC and run the MENU program whenever MS-DOS is started.

   NEWLINE
3. Press the <CONTROL-Z> key; then press the <CR> key to put the command BASIC MENU in the AUTOEXEC.BAT file.
4. The MENU program will now run automatically whenever you start MS-DOS.

To run your own BASIC program, enter the name of your program in place of MENU in the second line of the example. You can enter any MS-DOS command or series of commands in the AUTOEXEC.BAT file.

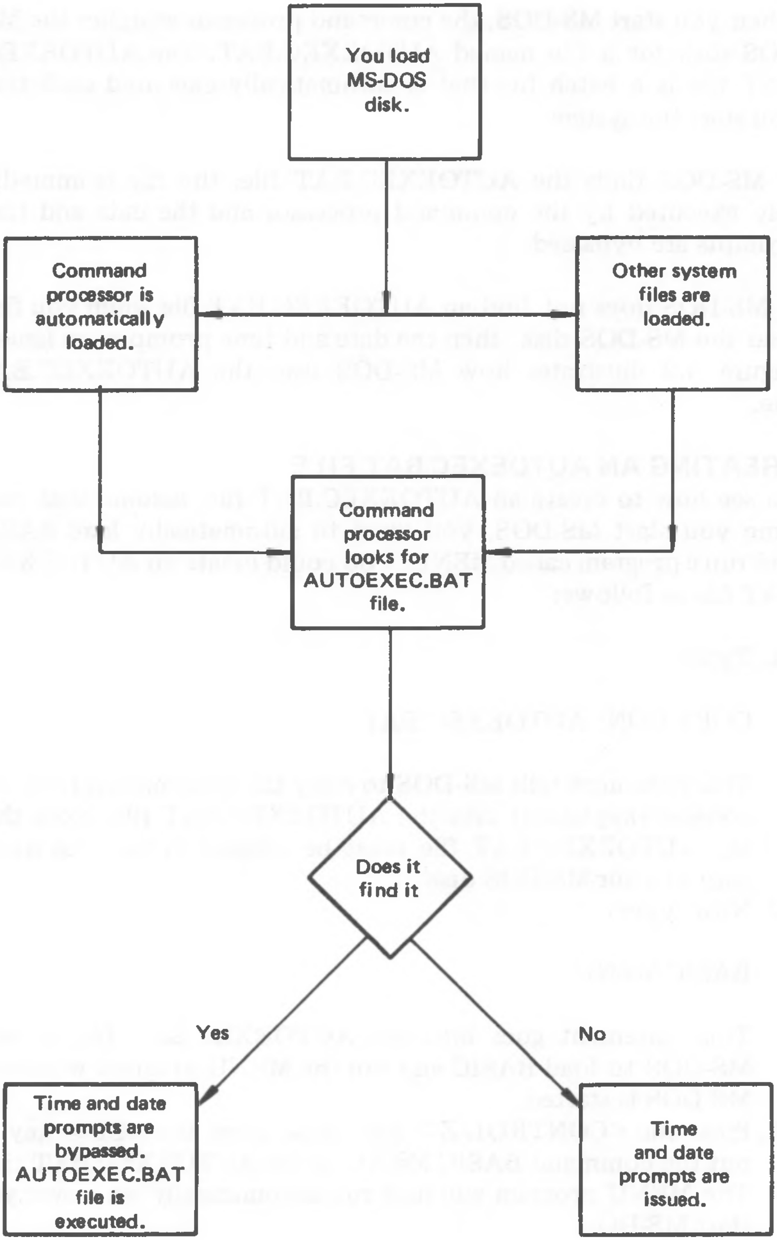

You load MS-DOS disk.
Command processor is automatically loaded.
Other system files are loaded.
Command processor looks for AUTOEXEC.BAT file.
Does it find it
Yes
No
Time and date prompts are bypassed. AUTOEXEC.BAT file is executed.
Time and date prompts are issued.

Figure 4.2 How MS-DOS uses the AUTOEXEC.BAT file

NOTE: Remember that if you use an AUTOEXEC.BAT file, MS-DOS does not prompt you for a current date and time unless you include the DATE and TIME commands in the AUTOEXEC.BAT file. You should include these two commands in your AUTOEXEC.BAT file, since MS-DOS uses this information to keep your directory current.

CREATING A .BAT FILE WITH REPLACEABLE PARAMETERS

There may be times when you want to create an application program and run it with different sets of data. This data may be stored in various MS-DOS files.

When used in MS-DOS commands, a parameter is an option that you define. With MS-DOS, you can create a batch (.BAT) file with dummy (replaceable) parameters. These parameters, named %0-%9, can be replaced by values supplied when the batch file executes.

For example, when you type the command line COPY CON MYFILE.BAT, the next lines you type are copied from the console to a file named MYFILE.BAT on the default drive:

```
A>COPY CON MYFILE.BAT
COPY  %1.MAC  %2.MAC
TYPE  %2.PRN
TYPE  %0.BAT
```

Now, press <CONTROL-Z> and then press <CR>. MS-DOS responds with this message:

```
1 File(s) copied
A>
```

The file MYFILE.BAT, which consists of three commands, now resides on the disk in the default drive.

The dummy parameters %1 and %2 are replaced sequentially by the parameters you supply when you execute the file. The dummy parameter %0 is always replaced by the drive designator, if specified, and the filename of the batch file (for example, MYFILE).

NOTE: Up to 10 dummy parameters (%0-%9) can be specified. Refer to the MS-DOS command SHIFT in Chapter 5 if you wish to specify more than 10 parameters. Also, if you use the percent sign as part of a filename within a batch file, you must type it twice. For example, to specify the file ABC%.EXE, you must type it as ABC%%.EXE in the batch file.

EXECUTING A .BAT FILE

To execute the batch file MYFILE.BAT and to specify the parameters that will replace the dummy parameters, you must enter the batch filename (without its extension) followed by the parameters you want MS-DOS to substitute for %1, %2, etc.

Remember that the file MYFILE.BAT consists of 3 lines:

```
COPY %1.MAC %2.MAC
TYPE %2.PRN
TYPE %0.BAT
```

To execute the MYFILE batch process, type:

```
MYFILE A:PROG1 B:PROG2
```

MYFILE is substituted for %0, A:PROG1 for %1, and B:PROG2 for %2.

The result is the same as if you had typed each of the commands in MYFILE with their parameters, as follows:

```
COPY A:PROG1.MAC B:PROG2.MAC
TYPE B:PROG2.PRN
TYPE MYFILE.BAT
```

The following table illustrates how MS-DOS replaces each of the above parameters:

BATCH	PARAMETER1 (%0)	PARAMETER2 (%1)	PARAMETER3 (%2)
FILENAME	(MYFILE)	(PROG1)	(PROG2)
MYFILE	MYFILE.BAT	PROG1.MAC	PROG2.MAC PROG2.PRN

Remember that the dummy parameter %0 is always replaced by the drive designator (if specified) and the filename of the batch file.

INPUT AND OUTPUT

MS-DOS always assumes that input comes from the keyboard and output goes to the terminal screen. However, the flow of command input and output can be redirected. Input can come from a file rather than a terminal keyboard, and output can go to a file or to a line printer instead of to the terminal. In addition, "pipes" can be created that allow output from one command to become the input to another. Redirection and pipes are discussed in the next sections.

REDIRECTING YOUR OUTPUT

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

```
    DIR
```

displays a directory listing of the disk in the default drive on the terminal screen. The same command can send this output to a file named MYFILES by designating the output file on the command line:

```
    DIR >MYFILES
```

If the file MYFILES does not already exist, MS-DOS creates it and stores your directory listing in it. If MYFILES already exists, MS-DOS overwrites what is in the file with the new data.

If you want to append your directory or a file to another file (instead of replacing the entire file), two greater-than signs (>>) can be used to tell MS-DOS to append the output of the command (such as directory listing) to the end of a specified file. The command

```
    DIR >>MYFILES
```

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

It is often useful to have input for a command come from a file rather than from a terminal. This is possible in MS-DOS by using a less-than sign (<) in your command. For example, the command

```
    SORT <NAMES> LIST1
```

sorts the file NAMES and sends the sorted output to a file named LIST1.

FILTERS

A filter is a command that reads your input, transforms it in some way, and then outputs it, usually to your terminal or to a file. In this way, the data is said to have been "filtered" by the program. Since filters can be put together in many different ways, a few filters can take the place of a large number of specific commands.

MS-DOS filters include CIPHER, FIND, MORE, and SORT, and perform the following functions:

CIPHER
    Encrypts/decrypts a file.
FIND
    Searches for a constant string of text in a file.
MORE
    Takes standard terminal output and displays it, one screen at a time.
SORT
    Sorts text.

You can see how these filters are used in the next section.

COMMAND PIPING

If you want to give more than one command to the system at a time, you can "pipe" commands to MS-DOS. For example, you may occasionally need to have the output of one program sent as the input to another program. A typical case would be a program that produces output in columns. It could be desirable to have this columnar output sorted.

Piping is done by separating commands with the pipe separator, which is the vertical bar symbol (|). For example, the command

```
    DIR | SORT
```

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

Piping can also be used when you want to output to a file. If you want your directory sorted and sent to a new file (for example, DIREC.FIL), you could type:

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

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

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

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

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

```
DIR | SORT | MORE
```

sorts your directory, shows it to you one screen at a time, and puts "--MORE--" at the bottom of your screen when there is more output to be seen.

You will find many uses for piping commands and filters. You will also find more information on using filters in the next chapter, MS-DOS Commands.

MS-DOS COMMANDS

This section describes each of the MS-DOS commands, arranged in alphabetical order for quick reference. Certain commands are used only if you are writing batch programs. These commands, ECHO, FOR, GOTO, IF, and SHIFT, are noted as batch processing commands in the description. The individual command descriptions are preceded by a table summarizing the complete set.

Before studying or using any of the commands, be sure to become familiar with the notations that indicate how to format a command. (The notations were explained in an earlier chapter, but are important enough to repeat.)

- Words shown in capital letters are required entries. These words are called keywords and must be entered exactly as shown. You can enter these keywords in any combination of upper/lowercase; MS-DOS converts all keywords to uppercase.
- You supply the text for any items enclosed in angle brackets (< >). For example, you should enter the name of *your* file when <filename> is shown in the format.
- Items in square brackets ([ ]) are optional. If you include optional information,.do not include the square brackets, only the information within the brackets.
- An ellipsis (. . .) indicates that you may repeat an item as many times as you want.
- You must include all punctuation where shown (with the exception of square brackets), such as commas, equal signs, question marks, colons, or slashes.

MS-DOS COMMAND SUMMARY

Name (Synonym)	Purpose	Syntax
ASSIGN	Assigns a disc drive	[d1=d2 [...]]
BACKUP	Copies fixed disk to flexible disks	BACKUP [d1:][<pathname>][<filename>[<.ext>]]
BREAK	Sets CONTROL-C check	BREAK ON BREAK OFF
CHDIR (CD)	Changes directories; prints working directory	CHDIR [pathname]
CHKDSK	Scans the directory of the default or designated drive and checks for consistency	CHKDSK [d:] <filespec> [/F] [/V]
CIPHER	Encrypts/decrypts a file	CIPHER <keyword> [<filename>]
CLS	Clears screen	CLS
CONFIG	Defines configuration information	CONFIG [d:]
COPY	Copies file(s) specified	COPY <filespec> [<filespec>] [<pathname>] [<pathname>] [/V]
CTTY	Changes console TTY	CTTY \DEV\DEV
DATE	Displays and sets date	DATE [<mm>-<dd>-<yy>]
DEL (ERASE)	Deletes file(s) specified	DEL [<filespec>] [<pathname>]
* DIR	Lists requested directory entries	DIR [<filespec>] [<pathname>] [/P] [/W]
DISKCOPY	Copies disks	DISKCOPY [d:] [d:]
ECHO	Turns batch file echo feature on/off	ECHO [ON message] ECHO [OFF message]
EXIT	Exits command and returns to lower level	EXIT
* * EXE2BIN	Converts executable files to binary format	EXE2BIN <filespec>[d:][<filename>[<.ext>]]
FIND	Searches for a constant string of text	FIND [/V /C /N] <string> [d:] [<filename . . .>]
FOR	Batch command extension	For batch processing: FOR %%<c> IN <set> DO <command> For interactive processing: FOR %<c> IN <set> DO <command>
FORMAT	Formats a disk to receive MS-DOS file • fixed disk • flexible disk	 FORMAT [d] : [/V] [/S] FORMAT [d:] [/V ~~/J /D /I /O /S]~~ /1/8/9/0/S]
GOTO	Batch command extension	GOTO <label>
IF	Batch command extension	IF <condition> <command>
* DISKCOMP	Compares to identical flexible discs	[d1:][d2:][/1][/8]
** ESC	Enables input of escape sequences through keyboard for Lear Siegler escape sequences	For ANSI escape sequences ESC [#; # ... delimiters ESC ["string";... ESC "string" ESC #

MS-DOS COMMAND SUMMARY (Cont.)

Name (Synonym)	Purpose	Syntax
? LOCATE	Converts executable files to binary format	LOCATE <filespec> [d:] [<filename> [<.ext>]]
MKDIR (MD)	Makes a directory	MKDIR <pathname>
* MORE	Displays output one screen at a time	MORE
PATH	Sets a command search path	PATH [<pathname>[;<pathname>] . . .]
PAUSE	Pauses for input in a batch file	PAUSE [<comment>]
PRINT	Background print feature	PRINT [[filespec] [/T] [/C] [/P]] . . .
PROMPT	Designates command prompt	PROMPT [<prompt–text>]
RAMDISK	Allocates RAM-disk memory (described in Appendix D)	DEVICE=RAMDISK.SYS
RECOVER	Recovers a bad disk	RECOVER <filename> RECOVER <d:>
REM	Displays a comment in a batch file	REM [<comment>]
REN (RENAME)	Renames first file as second file	REN <filespec> <filename>
RESTORE	Restores backed-up files from flexible disk to fixed disk	d1:[d2:][<pathname>][<filename>][<.ext>]][/S][/P]
RDCPM	Transfers CP/M files to an MS-DOS formatted disk	RDCPM DIR d: RDCPM d: filename [d:]
RMDIR (RD)	Removes a directory	RMDIR [d:] <pathname>
SAVEFK	Stores function definitions on disk	SAVEFK [d:][<pathname>]<filename>
SET	Sets one string value to another	SET [<string = string>]
SHIFT	Increases number of replaceable parameters in batch process	SHIFT
SORT	Sorts data alphabetically, forward or backward	[<<filespec 1>][<>filespec2>] SORT [/R] [/+n]
SYS	Transfers MS-DOS system files from drive A: to the drive specified	SYS <d>:
TIME	Displays and sets time	TIME [<hh> [:<mm>]]
TREE	Displays all directories and sub-directories	TREE[d:][/F][>PRN]
TYPE	Displays the contents of file specified	TYPE <filespec>
VER	Prints MS-DOS version number	VER
VERIFY	Verifies writes to disk	VERIFY [ON] VERIFY [OFF]
VOL	Prints volume identification number	VOL [d:]
* MODE	Defines screen parameters and serial interface characteristics	MODE CON: [Cx] [,P1][,P2][,Lx][,R] [,V] MODE COMx: <baud rate>, <parity>, <data bits>, <stop bits> [,P][,Tnnn][,R][,Annn]

5.4-5.7
ASSIGN
BACKUP

BACKUP

NAME

BACKUP

TYPE

External

PURPOSE

Copies the contents of one of the logical fixed disks in the source drive to flexible disks; also restores the fixed disk.

SYNTAX

BACKUP

COMMENTS

Before copying begins, BACKUP asks for the source and destination drive designations, the 6-character volume ID (label) to be placed on each flexible disk, and if write with verify is to be performed.

After answering the questions, insert a formatted flexible disk in the destination disk drive and press <CR> to start the copy. (The flexible disks must be formatted using no switches.) As soon as one flexible disk is filled, BACKUP prompts you to insert the next disk.

NOTE: BACKUP copies the entire contents of one logical fixed disk. If you only want to copy selected files, use the COPY command.

To restore the fixed disk, simply reverse the copy: specify the flexible disk drive as the source and the fixed disk drive as the destination. Messages are displayed if the ID you enter does not match the ID on the flexible disk or if you insert a flexible disk out of sequence.

BREAK

NAME

BREAK

TYPE

Internal

PURPOSE

Sets CONTROL-C check.

SYNTAX

```
BREAK ON
BREAK OFF
```

COMMENTS

If you are running an application program that uses CONTROL-C function keys, you will want to turn off the MS-DOS CONTROL-C function so that when you press <CONTROL-C> you affect your program and not the operating system. Specify BREAK OFF to turn off CONTROL-C and BREAK ON when you have finished running your application program and are using MS-DOS.

CHDIR

NAME

CHDIR (CHANGE DIRECTORY)

TYPE

Internal

SYNONYM

CD

PURPOSE

Changes directory to a different path; displays current (working) directory.

SYNTAX

CHDIR [pathname]

COMMENTS

If your working directory is \BIN\USER\JOE and you want to change your path to another directory (such as \BIN\USER\JOE\FORMS), type:

```
    CHDIR \BIN\USER\JOE\FORMS
```

and MS-DOS puts you in the new directory. A shorthand notation is also available with this command:

```
    CHDIR ..
```

This command always puts you in the parent directory of your working directory.

CHDIR used without a pathname displays your working directory. If your working directory is \BIN\USER\JOE on drive B, and you type CHDIR <CR>, MS-DOS displays:

```
    B: \BIN\USER\JOE
```

This command is useful if you forget the name of your working directory.

CHKDSK

NAME

CHKDSK (CHECK DISK)

TYPE

External

PURPOSE

Scans the directory of the specified disk drive and checks it for consistency.

SYNTAX

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

COMMENTS

CHKDSK should be run occasionally on each disk to check for errors in the directory. If any errors are found, CHKDSK displays error messages, if any, and then a status report similar to the one below.

```
    160256   bytes total disk space
      8192   bytes in 2 hidden files
       512   bytes in 2 directories
     30720   bytes in 8 user files
    121344   bytes available on disk

     65536   bytes total memory
     53152   bytes free
```

CHKDSK does not correct the errors found in your directory unless you specify the /F (fix) switch. Typing /V causes CHKDSK to display messages while it is running.

You can redirect the output from CHKDSK to a file. Simply type:

```
    CHKDSK A:>filename
```

The errors are sent to the filename specified. Do not use the /F switch if you redirect CHKDSK output.

This page
omitted from new
manual?

CHKDSK
5.10 - 5.11

The following errors are corrected automatically if you specify the /F switch:

Invalid drive specification

Invalid parameter

Invalid sub-directory entry

Cannot CHDIR to <filename>
Tree past this point not processed

First cluster number is invalid
entry truncated

Allocation error, size adjusted

Has invalid cluster, file truncated

Disk error reading FAT

Disk error writing FAT

<filename> contains
non-contiguous blocks

All specified file(s) are contiguous

You must correct the following errors returned by CHKDSK, even if you specified the /F switch:

Incorrect DOS version

> You cannot run CHKDSK on versions of MS-DOS that are not 2.0 or higher.

Insufficient memory
Processing cannot continue

> There is not enough memory in your machine to process CHKDSK for this disk. You must obtain more memory to run CHKDSK.

Errors found, F parameter not specified
Corrections will not be written to disk

> You must specify the /F switch if you want the errors corrected by CHKDSK.

Invalid current directory
Processing cannot continue

> Restart the system and rerun CHKDSK.

Cannot CHDIR to root
Processing cannot continue

> The disk you are checking is bad. Try restarting MS-DOS and RECOVER the disk.

<filename> is cross linked on cluster

> Make a copy of the file you want to keep, and then delete both files that are cross linked.

X lost clusters found in y chains
Convert lost chains to file (Y/N)?

> If you respond Y to this prompt, CHKDSK creates a directory entry and a file for you to resolve this problem (files created by CHKDSK are named FILEnnnnnnnn).
>
> CHKDSK then displays:
>
> X bytes disk space freed
>
> If you respond N to this prompt and have not specified the /F switch, CHKDSK frees the clusters and displays:
>
> X bytes disk space would be freed

Probable non-DOS disk
Continue (Y/N)?

> The disk you are using is a non-DOS disk. You must indicate whether or not you want CHKDSK to continue processing.

Insufficient room in root directory
Erase files in root and repeat CHKDSK

> CHKDSK cannot process until you delete files in the root directory.

Unrecoverable error in directory
Convert directory to file (Y/N) ?

> 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.

CIPHER

NAME

CIPHER

TYPE

External

PURPOSE

Encrypts and decrypts files based on a specified keyword.

SYNTAX

```
CIPHER <keyword> [<filename>]
```

COMMENTS

Use this command when you want to encrypt a file for security purposes. The CIPHER command uses a keyword that must be provided when encrypting the file. To encrypt the file NSA.CIA using the keyword "SECRET," enter:

```
    CIPHER SECRET NSA.CIA
```

This displays the encrypted file (NSA.CIA) on your screen. If you want the encrypted file sent to another file, enter:

```
    CIPHER <NSA.CIA  >MYSTERY.NEW
```

where MYSTERY.NEW is the name of the file where you are storing the encrypted file. You may delete the original file NSA. CIA.

To decrypt the encrypted file MYSTERY.NEW, simply reverse the process:

```
    CIPHER SECRET  <MYSTERY.NEW
```

This command decrypts the file MYSTERY.NEW and displays it on your screen. If you want the decrypted file sent to another file, called NOSECRET.XXX, type:

```
    CIPHER SECRET  <MYSTERY.NEW >NOSECRET.XXX
```

NOTE: You must supply the same keyword that you encrypted the file with when you decrypt the file, or the CIPHER command will not work.

This page missing from
new manual

CLS

NAME

CLS

TYPE

Internal

PURPOSE

Clears the terminal screen.

SYNTAX

CLS

COMMENTS

The CLS command causes MS-DOS to send the ANSI escape sequence ESC[2J (that clears the screen) to the console.

5.14 – 5.20
Config

CONFIG

NAME

CONFIG

TYPE

External

PURPOSE

Defines and modifies (temporarily or permanently) configuration information to MS-DOS.

SYNTAX

CONFIG

COMMENTS

Use this command to define your processing environment to MS-DOS: the type of printer and disks, any programmable function keys, and the number of retries to be performed on disk read and writes.

MS-DOS is initially set up with specific parameters. The following table shows these parameters and the changes that you can make with CONFIG.

	Initial Definition	With CONFIG
Programmable Function Keys	none	up to 20
Printer	parallel	serial
Serial Printer Interface:		
— Stop Bits	1	1 1/2 or 2
— Parity	even	disabled or odd
— Character length	7 bits	5, 6, or 8
— Baud rate	9600	50-19200
Disk	2 flexible	1 flexible, 1 fixed
Retry/Restore Counters:		
— Flexible	5, 5	1-9, 1-9
— Fixed	5, 5	1-9, 1-9

CONFIG is made up of a series of lead-through screens. To begin, type CONFIG and you see the main function screen.

```
CONFIG
      . . .
                    CONFIG UTILITY

1)  Modify Function Keys
2)  Select Printer (Serial/Parallel)
3)  Modify Retry/Restore Counter
4)  Modify Serial Printer Interface
5)  Modify Disk Configuration
6)  Exit Program

*   Enter function
```

After you select the function, further screens guide you in defining your configuration. Although the screens are self-explanatory, some usage conventions should be noted.

None of the 20 programmable function keys are predefined. A single definition can be approximately 255 characters long (the exact length depends on the characters used). The definition may specify any function, but cannot include another function key (no characters in the range of 80-FF are accepted).

Function key definitions are placed in a table that can hold up to 492 characters. If more are entered a message is displayed. When you continue processing, the input definition of the key that caused the overflow is deleted, but any original contents is not.

The function keys are a convenience feature. An often-used function, for example, can be assigned to a function key and then initiated simply by pressing the key.

Assume you usually begin processing by displaying the directory on your system disk. You could assign the command DIR A: (plus the <CR> return function) to function key 1 with CONFIG. Request function 1 from the main screen, function 2 from the function key screen, and then press F1 in response to the Enter Function Key message. You will see:

```
    Function 01:
```

Now, enter the command, including the <CR> function.

```
    Function 01: DIR A:<CR>
```

NOTE: The control character keys with hexadecimal values from 00 through 1F are displayed between the symbols < >.

Check your definition and press the function key. (The definition always begins and ends by pressing the function key.) The key is now programmed to do a directory display. If you want to check the assignment, request the 'display definition' function.

As already discussed in Chapter 2, CONFIG must be used to define the disk system unless two flexible disks are being used. Modifications to the disk configuration parameters must be specified as "permanent," written to the operating system disk, and must be followed by a system restart. (A restart simply means turning off and on the computer.) The restart initializes the disk drives.

The Exit Program function may be used after each configuration function is performed or after all functions are completed. When requested, Exit Program displays three options.

```
    1)  Update O.S. disk in drive A
    2)  Return to main program
    3)  Exit CONFIG
```

ATTENTION: Changes to the disk configuration must be written to disk (permanent) and must be followed by a restart. Update the disk, exit CONFIG, and then turn off and on the computer when the system prompt appears.

```
    *   Enter function
```

Function 1 is used to have the new configuration parameters written to disk. If the modifications are only temporary (for a specific run, for example), use function 3; the changes are only made in memory.

If you are making permanent changes and want to also update all other copies of your operating system disk, just insert another disk in drive A and request function 1. You can repeat this procedure and update all MS-DOS operating system disks.

You seldom have errors when using CONFIG, but you may – if your operating system disk is nearing its capacity. During the last phase of a "modify disk configuration" function, CONFIG writes the changes to disk in a file called CONFIG.SYS. If the directory or the file area itself is full, you will see either of two messages:

```
DIRECTORY FULL
DELETE A FILE FROM YOUR O.S. DISK; THEN REPEAT
THIS FUNCTION.
```

or

```
DISK FULL
DELETE OR SHORTEN A FILE FROM YOUR O.S. DISK;
THEN REPEAT THIS FUNCTION.
```

To correct the problem, simply delete or shorten a non-essential file (a scratch or backup file, perhaps); then, request CONFIG again, go directly to the Exit Program function, and specify function 1. The configuration changes are written to disk.

COPY

NAME

COPY

TYPE

Internal

PURPOSE

Copies one or more files to another disk. If you prefer, you can give the copies different names. This command can also copy files on the same disk.

SYNTAX

```
COPY <filespec> [<filespec>] [<pathname>] [<pathname>] [/V]
```

COMMENTS

Before using this command, be sure the destination disk contains sufficient space for the copy.

If the second filespec option is not given, the copy is to the default drive and has the same name as the original file (first filespec option). If the first filespec is on the default drive and the second filespec is not specified, the COPY is aborted (copying files to themselves is not allowed) and MS-DOS returns the error message:

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

NOTE: You cannot copy a file on flexible disk to another flexible disk using a single flexible disk drive. To copy selected files, you must first copy the files to the fixed disk and then to another flexible disk.

The second option may take three forms:

1. If the second option is a drive designation (d:) only, the original file is copied with the original filename, to the designated drive.

2. If the second option is a filename only, the original file is copied to a file on the default drive with the filename specified.
3. If the second option is a full filespec, the original file is copied to a file on the default drive with the filename specified.

The /V switch causes MS-DOS to verify that the sectors written on the destination disk are recorded properly. Although there are rarely recording errors when you run COPY, you can verify that critical data has been correctly recorded. This option causes the COPY command to run more slowly because MS-DOS must check each entry recorded on the disk.

The COPY command also allows file concatenation (joining) while copying. Concatenation is accomplished by simply listing any number of files as options to COPY, separated by "+."

For example,

```
COPY A.XYZ + B.COM + B:C.TXT BIGFILE.CRP
```

This command concatenates files named A.XYZ, B.COM, and B:C.TXT and places them in the file on the default drive called BIGFILE.CRP.

To combine several files using wild cards into one file, you could type:

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

This command would take all files with a filename extension of .LST and combine them into a file named COMBIN.PRN.

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

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

5-23 - 5.24

The following COPY command combines all files matching *.LST, then all files matching *.REF, into one file named COMBIN.PRN:

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

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

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

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

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

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

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

Addition in new manual

CTTY

NAME

CTTY

TYPE

Internal

PURPOSE

Allows you to change the device from which you issue commands (TTY represents the console).

SYNTAX

```
CTTY \DEV\DEV
```

COMMENTS

DEV stands for "device", which is the device from which you are giving commands to MS-DOS. This command is useful if you want to change the device on which you are working. The command

```
    CTTY \DEV\AUX
```

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

```
    CTTY \DEV\CON
```

moves I/O back to the original device (here, the console). Refer to "Illegal Filenames" in Chapter 3 for a list of valid device names to use with the CTTY command.

DATE

NAME

DATE

TYPE

Internal

PURPOSE

Enter or change the date known to the system. This date is recorded in the directory for any files you create or alter.

You can change the date from your terminal or from a batch file. (MS-DOS does not display a prompt for the date if you use an AUTOEXEC.BAT file, so you may want to include a DATE command in that file.)

SYNTAX

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

COMMENTS

If you type DATE, DATE responds with the message:

```
Current date is <mm>-<dd>-<yy>
Enter new date:_
```

Press <~~CR~~ NEWLINE> if you do not want to change the date shown.

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

```
DATE 5-~~9~~7-~~83~~84
```

In this case, you do not have to answer the "Enter new date:" prompt.

The new date must be entered using numerals only; letters are not permitted. The allowed options are:

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

The date, month, and year entries may be separated by hyphens (-) or slashes (/).

If the options or separators are not valid, DATE displays the message:

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

DATE then waits for you to enter a valid date.

DEL

NAME

DEL (DELETE)

TYPE

Internal

SYNONYM

ERASE

PURPOSE

Deletes all files with the designated filespec.

SYNTAX

DEL [<filespec>] [<pathname>]

COMMENTS

If the filespec is *.*, the prompt "Are you sure?" appears. If a Y or y is typed as a response, then all files are deleted as requested. You can also type ERASE for the DELETE command.

DIR

NAME

DIR (DIRECTORY)

TYPE

Internal

SYNTAX

DIR [<filespec>] [<pathname>] [/P] [/W]

PURPOSE

Lists the files in a directory.

COMMENTS

If you just type DIR, all directory entries on the default drive are listed. If only the drive specification is given (DIR d:), all entries on the disk in the specified drive are listed. If only a filename is entered with no extension (DIR filename), then all files with the designated filename on the disk in the default drive are listed. If you designate a file specification (for example, DIR d:filename.ext), all files with the filename specified on the disk in the drive specified are listed. In all cases, files are listed with their size in bytes and with the time and date of their last modification.

The wild card characters ? and * (question mark and asterisk) may be used in the filename option. As examples, the following table shows equivalent command designations.

COMMAND	EQUIVALENT
DIR	DIR *.*
DIR FILENAME	DIR FILENAME.*
DIR .EXT	DIR *.EXT
DIR .	DIR *.

Two switches may be specified with DIR. The /P switch selects Page Mode. With /P, display of the directory pauses after the screen is filled. To resume display of output, press any key.

The /W switch selects Wide Display. With /W, only filenames are displayed, without other file information. Files are displayed five per line.

5.30
Diskcomp

DISKCOPY

NAME

DISKCOPY

TYPE

External

PURPOSE

Copies the contents of the disk in the source drive to the disk in the destination drive.

SYNTAX

```
DISKCOPY [d:] [d:]
```

COMMENTS

The first option you specify is the source drive; the second option is the destination drive.

The disk in the destination drive must be formatted (by the same operating system and in the same format as the source disk) before using DISKCOPY.

With DISKCOPY you can copy a flexible disc to another flexible disc, or a fixed disc to another fixed disk.

You can specify the same drives or you may specify different drives. If the drives designated are the same, a single-drive copy operation is performed. You are prompted to insert the disks at the appropriate times. DISKCOPY waits for you to press any key before continuing.

After copying, DISKCOPY prompts:

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

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

To end the COPY, press N.

Before using DISKCOPY, also consider the following command characteristics:

- If you omit both options, a single-drive copy operation is performed on the default drive.
- If you omit the second option, the default drive is used as the destination drive.
- Both disks must have the same number of physical sectors and those sectors must be the same size.
- Disks that have had a lot of file creation and deletion activity become fragmented, because disk space is not allocated sequentially. The first free sector found is the next sector allocated, regardless of its location on the disk.

  A fragmented disk can cause poor performance due to delays involved in finding, reading, or writing a file. If this is the case, you must use the COPY command, instead of DISKCOPY, to copy your disk and eliminate the fragmentation.

  For example:

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

  copies all files from the disk in drive A to the disk in drive B.
- DISKCOPY automatically determines the number of sides to copy, based on the source drive and disk.
- If disk errors are encountered during a DISKCOPY, MS-DOS displays:

  ```
  DISK error while reading drive A
  Abort, Ignore, Retry?
  ```

Refer to Appendix B, Disk Errors, for information on this error message.

ECHO

NAME

ECHO

TYPE

Internal; Batch processing

PURPOSE

Turns batch echo feature on and off.

SYNTAX

ECHO [ON message]
ECHO [OFF message]

COMMENTS

Normally, commands in a batch file are displayed ("echoed") on the console when they are seen by the command processor. ECHO OFF turns off this feature. ECHO ON turns the echo back on.

If ON or OFF are not specified, the current setting is displayed.

5.34 - 5.38

esc EXE2BIN

EXIT

NAME

EXIT

TYPE

Internal

PURPOSE

Exits the program COMMAND.COM (the command processor) and returns to a previous level, if one exists.

SYNTAX

```
EXIT
```

COMMENTS

This command can be used when you are running an application program and want to start the MS-DOS command processor, then return to your program. For example, to look at a directory on drive B while running an application program, you must start the command processor by typing COMMAND in response to the default drive prompt:

```
    A>COMMAND
```

You can now type the DIR command and MS-DOS displays the directory for the default disk. When you type EXIT, you return to the previous level (your application program).

FIND

NAME

FIND

TYPE

External

PURPOSE

Searches for a specific string of text in a file or files.

SYNTAX

FIND [/V /C /N] <string> [d:] [<filename. . .>]

Filenames with the wild card characters "?" and "*" cannot be used.

COMMENTS

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

If no files are specified, FIND takes the input on the screen and displays all lines that contain the specified string.

These switches can be used with FIND:

/V
: This switch causes FIND to display all lines not containing the specified string.

/C
: This switch causes FIND to display only the count of lines that contained a match in each of the files.

/N
: This switch causes each line to be preceded by its relative line number in the file.

The string should be enclosed in quotes. For example,

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

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

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

causes MS-DOS to display all names of the files on the disk in drive B that do not contain the string DAT. Type double quotes around a string that already has quotes in it.

When an error is detected, FIND responds with one of the following error messages:

Incorrect DOS version
: FIND only runs on versions of MS-DOS that are 2.0 or higher.

FIND: Invalid number of parameters
: You did not specify a string when issuing the FIND command.

FIND: Syntax error
: You typed an illegal string when issuing the FIND command.

FIND: File not found <filename>
: The filename you have specified does not exist or FIND cannot find it.

FIND: Read error in <filename>
: An error occurred when FIND tried to read the file specified in the command.

FIND: Invalid parameter <option-name>
: You specified an option that does not exist.

FOR

NAME

FOR

TYPE

Internal; Batch processing

PURPOSE

Command extension used in batch and interactive file processing.

SYNTAX

- For batch processing:

```
FOR %%<c> IN <set> DO <command>
```

- For interactive processing:

```
FOR %<c> IN <set> DO <command>
```

COMMENTS

<c> can be any character except 0, 1, 2, 3, . . . , 9 to avoid confusion with the %0-%9 batch parameters.

```
<set> is (<item>. . .)
```

The %%<c> variable is set sequentially to each member of <set>, and then <command> is evaluated. If a member of <set> is an expression involving * and/or ?, then the variable is set to each matching pattern from disk. In this case, only one such <item> may be in the set, and any <item> besides the first is ignored.

NOTE: The words IN, FOR, and DO must be in uppercase.

Consider these examples:

```
FOR %%f IN ( *.ASM ) DO MASM %%f;
FOR %%f IN (FOO BAR BLECH) DO REM %%f
```

The '%%' is needed so that after batch parameter (%0-%9) processing is done, there is one '%' left. If only '%f' were there, the batch parameter processor would see the '%', look at 'f', decide that '%f' was an error (bad parameter reference) and throw out the '%f', so that the command FOR would never see it. If the FOR is not in a batch file, then only one '%' should be used.

5.44 – 5.45
FORMAT

FORMAT

NAME

FORMAT

TYPE

External

PURPOSE

Formats the disk in the specified drive to accept MS-DOS files.

SYNTAX

```
FORMAT [d:] [/V /J /D /I /O /S]    (flexible disks)
FORMAT [d:] [/V]                   (fixed disks)
```

COMMENTS

This command formats the disk and initializes the directory and file allocation tables. If no drive is specified, the disk in the default drive is formatted. All disks are formatted at double density, double sided and either at 9 sectors per track for a flexible disk or 17 sectors per track for a fixed disk.

When formatting a fixed disk, FORMAT displays a message asking for the number of certifications (read-after-write checks on each track). The default value is 5; increasing the number will significantly increase the time for formatting.

Six switches control options that can be requested when formatting a flexible disk; only the /V switch is valid when formatting a fixed disk. Of the switches, two are used more frequently than the others.

/V

Causes FORMAT to pause in the formatting process and display a message asking for a volume label (useful in disk identification).

/S

Causes FORMAT to copy the operating system files from the disk in the default drive to the newly formatted disk. The files are copied in the following order: IO.SYS, MSDOS.SYS, COMMAND.COM. (Other files can then be selectively copied with the COPY command.) When used with other switches, /S must be entered last.

NOTE: To copy an entire system disk, format with *no switches* and then use DISKCOPY. DISKCOPY produces a "mirror image" and writes over any label.

The next group of switches allows some other format to be created on a flexible disk. (These formats may be needed if you want to copy and use information from a non-NCR format disk.) If no switch is specified, the default format is assumed: 9 sectors per track; double sided, double density (360 KB disk capacity).

/J

Formats at 9 sectors per track; single sided, double density (180KB disk capacity).

/D

Formats at 8 sectors per track; double sided, double density (320KB disk capacity).

/I

Formats at 8 sectors per track; single sided, double density (160KB disk capacity).

The /O switch generates an E5 character in the first position of an empty (available) directory entry. Use this switch only if you need to maintain compatibility with older versions of MS-DOS; the current standard entry to indicate empty directory entries is 00.

GOTO

NAME

GOTO

TYPE

Internal; Batch processing

PURPOSE

Command extension used in batch file processing.

SYNTAX

GOTO <label>

COMMENTS

GOTO causes commands to be taken from the batch file beginning with the line after the <label> definition. If no label has been defined, the current batch file terminates.

For example:

```
    :foo
    REM looping . . .
    GOTO foo
```

produces an infinite sequence of messages: REM looping . . . .

Starting a line in a batch file with ':' causes the line to be ignored by batch processing. The characters following GOTO define a label, but this procedure may also be used to put in comment lines.

IF

NAME

IF

TYPE

Internal; Batch processing

PURPOSE

Command extension used in batch file processing.

SYNTAX

IF <condition> <command>

COMMENTS

The parameter <condition> is one of the following:

ERRORLEVEL <number>

> True if and only if the previous program executed by COMMAND had an exit code of <number> or higher.

<string1> == <string2>

> True if and only if <string1> and <string2> are identical after parameter substitution. Strings may not have embedded separators.

EXIST <filename>

> True if and only if <filename> exists.

NOT <condition>

> True if and only if <condition> is false.

The IF statement allows conditional execution of commands. When the <condition> is true, then the <command> is executed. Otherwise, the <command> is ignored.

NOTE: The words ERRORLEVEL, EXIST, and NOT must be uppercase.

Consider the following examples:

```
IF NOT EXIST \TMP\FOO ECHO Can't find file

IF NOT ERRORLEVEL 3 LINK $1, , ;
```

LOCATE

(not in new manual!)

NAME

LOCATE

TYPE

External

PURPOSE

Converts .EXE (executable) files to binary format. This results in a saving of disk space and faster program loading.

SYNTAX

```
LOCATE <filespec> [d:] [<filename>[<.ext>] ]
```

COMMENTS

This command is useful only if you want to convert .EXE files to binary format. The file named by filespec is the input file. If no extension is specified, it defaults to .EXE. The input file is converted to .COM file format (memory image of the program) and placed in the output file. If you do not specify a drive, the drive of the input file is used. If you do not specify an output filename, the input filename is used. If you do not specify a filename extension in the output filename, the new file is given an extension of .BIN.

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

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

1. If CS:IP is not specified in the .EXE file, a pure binary conversion is assumed. If segment fixups are necessary (that is, the program contains instructions requiring segment relocation), you are prompted for the fixup value. This value is the absolute segment at which the program is to be loaded. The resulting program is usable only when loaded at the absolute

memory address specified by a user application. The command processor will not be capable of properly loading the program.

2. If CS:IP is specified as 0000:100H, it is assumed that the file is to be run as a .COM file with the location pointer set at 100H by the assembler statement ORG; the first 100H bytes of the file are deleted. No segment fixups are allowed, as .COM files must be segment relocatable; that is, they must assume the entry conditions explained in the *PROGRAMMER'S MANUAL.* Once the conversion is complete, you may rename the resulting file with a .COM extension. Then the command processor is able to load and execute the program in the same way as the .COM programs supplied on your MS-DOS disk.

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
```

This message is also displayed if the file is not a valid executable file.

If LOCATE finds an error, one or more of the following error messages is displayed:

File not found
: The file is not on the disk specified.

Insufficient memory
: There is not enough memory to run LOCATE.

File creation error
: LOCATE cannot create the output file. Run CHKDSK to determine if the directory is full, or if some other condition caused the error.

Insufficient disk space
: There is not enough disk space to create a new file.

Fixups needed - base segment (hex):
: The source (.EXE) file contained information indicating that a load segment is required for the file. Specify the absolute segment address at which the finished module is to be located.

File cannot be converted

The input file is not in the correct format.

WARNING - Read error on .EXE file.
Amount read less than size in header

This is a warning message only.

MKDIR

NAME

MKDIR (MAKE DIRECTORY)

TYPE

Internal

SYNONYM

MD

PURPOSE

Makes a new directory.

SYNTAX

MKDIR < pathname>

COMMENTS

This command is used to create a hierarchical directory structure. When you are in your root directory, you can create subdirectories by using the MKDIR command. The command

```
    MKDIR \USER
```

creates a subdirectory \USER in your root directory. To create a directory named JOE under \USER, type:

```
    MKDIR \USER\JOE
```

5.50 - 5.53
MODE

MORE

NAME

MORE

TYPE

External

PURPOSE

Sends output to console one screen at a time.

SYNTAX

MORE

COMMENTS

MORE is a filter that reads from standard input (such as a command from your terminal) and displays one screen of information at a time. The MORE command then pauses and displays the --MORE-- message at the bottom of your screen.

Pressing the <~~CR~~ NEWLINE> key displays another screen of information. This process continues until all the input data has been read.

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

```
    TYPE MYFILES.COM | MORE
```

MS-DOS displays the file MYFILES.COM (on the default drive) one screen at a time.

PATH

NAME

PATH

TYPE

Internal

PURPOSE

Sets a command path.

SYNTAX

PATH [<pathname>[;<pathname>] . . .]

COMMENTS

This command allows you to tell MS-DOS which directories ~~to~~ should be searched for external commands after MS-DOS searches your working directory. The default value is ~~\BIN, where \BIN is the name of the directory in which all MS-DOS external commands reside.~~ no path.

To tell MS-DOS to search your \BIN\USER\JOE directory for external commands ~~(in addition to a search of the \BIN directory),~~ type:

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

MS-DOS now also searches the \BIN\USER\JOE directory for external commands until you set another path or shut down MS-DOS.

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

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

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

The command PATH with no options prints the current path. If you specify PATH;, MS-DOS sets the NUL path, meaning that only the working directory is searched for external commands.

PAUSE

NAME

PAUSE

TYPE

Internal

PURPOSE

Suspends execution of the batch file.

SYNTAX

PAUSE [<comment>]

COMMENTS

During the execution of a batch file, you may need to change disks or perform some other action. PAUSE suspends execution until you press any key, except <CONTROL-C>.

When the command processor encounters PAUSE, it prints:

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

If you press <CONTROL-C>, another prompt will be displayed:

```
    Abort batch job (Y/N)?
```

If you type Y in response to this prompt, execution of the remainder of the batch command file is aborted and control is returned to the operating system command level. Therefore, PAUSE is used to break a batch file into pieces, allowing you to end the batch command file at an intermediate point.

The comment is optional and is entered on the same line as PAUSE. You may want to prompt the user of the batch file with some meaningful message when the batch file pauses. For example, you may want to change disks in one of the drives. An optional prompt message may be given in such cases. The comment prompt is displayed before the "Strike a key" message.

PRINT

NAME

PRINT

TYPE

External

PURPOSE

Prints a text file on a line printer while you are processing other MS-DOS commands (usually called "background printing").

SYNTAX

PRINT [[filespec] [/T] [/C] [/P]] . . .

COMMENTS

You use the PRINT command only if you have a line printer attached to your computer. The following switches are provided with this command:

/T - TERMINATE
  This switch deletes all files in the print queue (those waiting to be printed). A message to this effect is printed.

/C - CANCEL
  This switch turns on cancel mode. The preceding filespec and all following filespecs are suspended in the print queue until you type a /P switch.

/P - PRINT
  This switch turns on print mode. The preceding filespec and all following filespecs are added to the print queue until you issue a /C switch.

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

Consider the following examples:

```
    PRINT /T
```

empties the print queue.

```
PRINT /T *.ASM
```

empties the print queue and queues all .ASM files on the default drive.

```
PRINT A:TEMP1.TST/C A:TEMP2.TST A:TEMP3.TST
```

removes the three files indicated from the print queue.

```
PRINT TEMP1.TST /C TEMP2.TST /P TEMP3.TST
```

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

If an error is detected, PRINT displays one of the following error messages:

Name of list device [PRN:]

This prompt appears when PRINT is run the first time. Any current device may be specified and that device then becomes the PRINT output device. As indicated in the [ ], simply pressing <~~CR~~ NEWLINE> results in the device PRN being used.

The device name entered is valid for all subsequent "PRINT" commands, To alter it, the system must be reloaded.

List output is not assigned to a device

This message is displayed if the "Name of list device" specified to the preceding prompt is invalid. Subsequent attempts return the same message until a valid device is specified.

PRINT queue is full

There is room for 10 files in the queue. If you attempt to put more than 10 files in the queue, this message appears on the console.

PRINT queue is empty

There are no files in the print queue.

No files match d:XXXXXXXX.XXX

A filespec was given for files to add to the queue, but no files match a specification. (If there are no files in the queue to match a cancelled filespec, no error message appears.)

Drive not ready

If this message occurs when PRINT attempts a disk access, PRINT keeps trying until the drive is ready. Any other error causes the current file to be cancelled. In such a case, an error message is output to your printer.

All files cancelled

If the /T (TERMINATE) switch is issued, the message "All files cancelled by operator" is output on your printer. If the current file being printed is cancelled by a /C, the message "File cancelled by operator" is printed.

PROMPT

NAME

PROMPT

TYPE

Internal

PURPOSE

Changes the MS-DOS command prompt.

SYNTAX

PROMPT [<prompt-text>]

COMMENTS

This command allows you to change the MS-DOS system prompt. If no text is typed, the prompt is set to the default prompt, which is the default drive designation. You can set the prompt to a special prompt by using the characters indicated below.

The following characters can be used in the prompt command to specify special prompts. They must all be preceded by a dollar sign ($) in the prompt command:

Specify This Character	To Get This Prompt:
$	The '$' character
t	The current time
d	The current date
p	The current directory of the default drive
v	The version number
n	The default drive
g	The '>' character
l	The '<' character
b	The '|' character
_	A CR LF sequence
s	A space
h	A backspace
e	ASCII code X'1B' (escape)

Consider the following example:

```
PROMPT $n:
```

Sets the prompt to the default drive followed by a colon.

You can also use escape sequences in your prompts. For example:

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

Sets the prompts in inverse video mode and returns to video mode for other text.

RDCPM

NAME

RDCPM (READ CPM)

TYPE

External

PURPOSE

Transfers NCR CP/M® files to an MS-DOS formatted disk.

SYNTAX

```
RDCPM DIR d:                (displays directory on CP/M disk)
RDCPM d: filename [d:]      (transfers CP/M file to MS-DOS disk)
```

COMMENTS

RDCPM reads the file from an NCR CP/M formatted disk and transfers it to an MS-DOS formatted disk. Once transferred, the file is an MS-DOS file.

The DIR variation of RDCPM displays the directory of the CP/M disk, so you can see the names of the files. The drive designation of the CP/M disk must be specified.

To transfer the file, you must specify the drive designation of the CP/M disk and the filename. The wild-card characters (* and ?) may be used to transfer several files. (See Chapter 3, More about Files, for a description of naming files with wild cards.) The destination drive designation is optional; and, if not specified, the disk in the default drive is assumed to be the destination disk.

Consider the following examples:

```
A>RDCPM C: MYFILE.TXT B:
A>RDCPM C: MYFILE.TXT
```

The first command transfers MYFILE.TXT from the disk in drive C to the disk in drive B; the second command transfers the same file to the disk in drive A, the default drive.

NOTE: The source and destination drive designations must be different. Therefore, if you have a single flexible disk drive and want to transfer a CP/M file from a flexible disk, you must first copy the file (with the CP/M operating system) to another disk drive. Then, use the RDCPM command.

If you have two flexible disc drives, the destination disc must contain RDCPM, and RDCPM must be started from there.

The following messages may be displayed; most are self-explanatory.

Hard disk error on CP/M drive
The disk specified by the source drive designation may not be a CP/M disk.

Source and destination drives must not be the same
The CP/M and MS-DOS disks must be on different drives.

Drive not available for CP/M reading
MS-DOS cannot access the specified drive. This error occurs if your MS-DOS disk configuration is not correct. For example, you specified

```
RDCPM C: YOURFILE B:
```

but the system is configured for a 2-flexible disk system with drives A and B. (No fixed disk was defined.) If the configuration definition is the cause of the error, use the CONFIG utility to modify the definition and then run RDCPM again.

Insufficient disk space
The MS-DOS destination disk does not have enough space for the CP/M file(s).

No room in directory to create file
The directory on the MS-DOS disk has no space to create an entry for the CP/M file(s).

Source file name missing
The specified file is not on the NCR CP/M disk. Check that the filename was entered correctly.

Source file not found
The specified file is not on the NCR CP/M disk. Check that the filename was entered correctly.

placed on p.64 in new manual.

File transfer complete
The specified file(s) was successfully transferred.

RECOVER

NAME

RECOVER

TYPE

External

PURPOSE

Recovers a file or an entire disk containing bad sectors.

SYNTAX

```
RECOVER <filename>
RECOVER <d:>
```

COMMENTS

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

To recover a particular file, type:

```
    RECOVER <filename>
```

This causes MS-DOS to read the file sector by sector and to skip the bad sector(s). When MS-DOS finds the bad sector(s), the sector(s) are marked and MS-DOS no longer allocates your data to that sector.

To recover a disk, type:

```
    RECOVER <d:>
```

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

If there is not enough room in the root directory, RECOVER prints a message and stores information about the extra files in the File Allocation Table. You can run RECOVER again to regain these files when there is more room in the root directory.

REM

NAME

REM (REMARK)

TYPE

Internal

PURPOSE

Displays remarks that are on the same line as the REM command in a batch file during execution of that batch file.

SYNTAX

REM [comment]

COMMENTS

The only separators allowed in the comment are the space, tab, and comma. Consider the following example:

```
1:  REM  This file checks new disks
2:  REM It is named NEWDISK.BAT
3:  PAUSE Insert new disk in drive B:
4:  FORMAT B:/S
5:  DIR B:
6:  CHKDSK B:
```

REN

NAME

REN (RENAME)

TYPE

Internal

SYNONYM

RENAME

PURPOSE

Changes the name of the first option (filespec) to the second option (filename).

SYNTAX

REN <filespec> <filename>

COMMENTS

The first option (filespec) must be given a drive designation if the disk resides in a drive other than the default drive. Any drive designation for the second option (filename) is ignored. The file remains on the disk where it currently resides.

The wild card characters may be used in either option. All files matching the first filespec are renamed. If wild card characters appear in the second filename, corresponding character positions are not changed.

For example, the following command changes the names of all files with the .LST extension to similar names with the .PRN extension:

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

In the next example, REN renames the file ABODE on drive B to ADOBE:

```
REN B:ABODE ?D?B?
```

The file remains on drive B.

An attempt to rename a filespec to a name already present in the directory results in the error message "File not found."

Duplicate filename or

67 also.

5.68-5.69
RESTORE

RMDIR

NAME

RMDIR (REMOVE DIRECTORY)

TYPE

Internal

SYNONYM

RD

PURPOSE

Removes a directory from a hierarchical directory structure.

SYNTAX

RMDIR [d:] <pathname>

COMMENTS

This command removes a directory *that is empty* except for the . and . . shorthand symbols.

To remove the \BIN\USER\JOE directory, first issue a DIR command for that path to ensure that the directory does not contain any important files that you do not want deleted. Then type:

```
    RMDIR \BIN\USER\JOE
```

The directory is deleted from the directory structure.

5.71
SAVEFX
~~SET~~

SET

NAME

SET

TYPE

Internal

PURPOSE

Sets one string value equivalent to another string for use in later programs.

SYNTAX

SET [<string = string>]

COMMENTS

This command is meaningful only if you want to set values that will be used by programs you have written. An application program can check all values that have been set with the SET command by issuing SET with no options. For example, SET TTY = VT52 sets your TTY value to VT52 until you change it with another SET command.

The SET command can also be used in batch processing. In this way, you can define your replaceable parameters with names instead of numbers. If your batch file contains the statement "LINK %FILE%", you can set the name that MS-DOS will use for that variable with the SET command. The command SET FILE = DOMORE replaces the %FILE% parameter with the filename DOMORE. Therefore, you do not need to edit each batch file to change the replaceable parameter names. Note that when you use text (instead of numbers) as replaceable parameters, the name must be ended by a percent sign.

SHIFT

NAME

SHIFT

TYPE

Internal; Batch processing

PURPOSE

Allows access to more than 10 replaceable parameters in batch file processing.

SYNTAX

SHIFT

COMMENTS

Usually, command files are limited to handling 10 parameters, %0 through %9. To allow access to more than ten parameters, use SHIFT to change the command line parameters. For example, if

%0 = “foo”
%1 = “bar”
%2 = “name”
%3 . . .%9 are empty

then a SHIFT results in the following:

%0 = “bar”
%1 = “name”
%2 . . .%9 are empty

If there are more than 10 parameters given on a command line, those that appear after the 10th (%9) are shifted one at a time into %9 by successive shifts.

5.74-5.75
Sort

SORT

NAME

SORT

TYPE

External

PURPOSE

SORT reads input from your terminal, sorts the data, then writes it to your terminal screen or files.

SYNTAX

```
SORT [/R]  [/+n]
```

COMMENTS

SORT can be used, for example, to alphabetize a file by a certain column. There are two switches that allow you to select options:

/R
: Reverses the sort; that is, sorts from Z to A.

/+n
: Sorts starting with column n where n is some number. If you do not specify this switch, SORT begins sorting from column 1.

Consider the following examples. In the first one, the command reads the file UNSORT.TXT, reverses the sort, and then writes the output to a file named SORT.TXT:

```
SORT /R <UNSORT.TXT >SORT.TXT
```

The next command pipes the output of the directory command to the SORT filter. The SORT filter sorts the directory listing starting with column 14 (this is the column in the directory listing that contains the file size), then sends the output to the console. Thus, the result of this command is a directory sorted by file size:

```
DIR | SORT /+14
```

The command

```
DIR | SORT /+14 | MORE
```

does the same thing as the command in the previous example, except that the MORE filter gives you a chance to read the sorted directory one screen at a time.

SYS

NAME

SYS (SYSTEM)

TYPE

External

PURPOSE

Transfers the MS-DOS system files from the disk in the default drive to the disk in the drive specified by d:.

SYNTAX

SYS <d>:

COMMENTS

SYS is normally used to update the system or to place the system on a formatted disk that contains no files. An entry for d: is required.

If IO.SYS and MSDOS.SYS are on the destination disk, they must take up the same amount of space on the disk as the new system will need. This means that you cannot transfer system files from an MS-DOS 2.0 disk to an MS-DOS 1.1 disk. You must reformat the MS-DOS 1.1 disk with the MS-DOS FORMAT command before the SYS command will work.

The destination disk must be completely blank or already have the system files IO.SYS and MSDOS.SYS.

The transferred files are copied in the following order:

IO.SYS
MSDOS.SYS

IO.SYS and MSDOS.SYS are both hidden files that do not appear when the DIR command is executed. COMMAND.COM (the command processor) is not transferred. You must use the COPY command to transfer COMMAND.COM.

If SYS detects an error, one of the following messages will be displayed:

No room for system on destination disk

> There is not enough room on the destination disk for the IO.SYS and MSDOS.SYS files.

Incompatible system size

> The system files IO.SYS and MSDOS.SYS do not take up the same amount of space on the destination disk as the new system will need.

TIME

NAME

TIME

TYPE

Internal

PURPOSE

Displays and sets the time.

SYNTAX

```
TIME [<hh>[:<mm>] ]
```

COMMENTS

If the TIME command is entered without any arguments, the following message is displayed:

```
    Current time is <hh>:<mm>:<ss>.<cc>
    Enter new time: _
```

Press the <~~CR~~ NEWLINE> key if you do not want to change the time shown. A new time may be given as an option to the TIME command as in:

```
    TIME 8:20
```

The new time must be entered using numerals only; letters are not allowed. The allowed options are:

```
    <hh>  = 00-24
    <mm> = 00-59
```

The hour and minute entries must be separated by colons. You do not have to type the <ss> (seconds) or <cc> (hundredths of seconds) options.

MS-DOS uses the time entered as the new time if the options and separators are valid. If the options or separators are not valid, MS-DOS displays the message:

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

MS-DOS then waits for you to type a valid time.

5.90
TREE

TYPE

NAME

TYPE

TYPE

Internal

PURPOSE

Displays the contents of the file on the console screen.

SYNTAX

TYPE <filespec>

COMMENTS

Use this command to examine a file without modifying it. (Use DIR to find the name of a file and EDLIN to alter the contents of a file.) The only formatting performed by TYPE is that tabs are expanded to spaces consistent with tab stops every eighth column. Note that a display of binary files causes control characters (such as CONTROL-Z) to be sent to your computer, including bells, form feeds, and escape sequences.

VER

NAME

VER

TYPE

Internal

PURPOSE

Prints MS-DOS version number.

SYNTAX

VER

COMMENTS

If you want to know what version of MS-DOS you are using, type VER. The version number is displayed on your screen.

VERIFY

NAME

VERIFY

TYPE

Internal

PURPOSE

Turns the verify switch on or off when writing to disk.

SYNTAX

VERIFY [ON]
VERIFY [OFF]

COMMENTS

This command has the same purpose as the /V switch in the COPY command. If you want to verify that all files are written correctly to disk, you can use the VERIFY command to tell MS-DOS to verify that your files are intact (no bad sectors, for example). MS-DOS performs a VERIFY each time you write data to a disk. You receive an error message only if MS-DOS was unable to successfully write your data to disk.

VERIFY ON remains in effect until you change it in a program (by a SET VERIFY system call), or until you issue a VERIFY OFF command to MS-DOS.

If you want to know what the current setting of VERIFY is, type VERIFY with no options.

VOL

NAME

VOL (VOLUME)

TYPE

Internal

PURPOSE

Displays disk volume number, if it exists.

SYNTAX

```
VOL [d:]
```

COMMENTS

This command prints the volume ID of the disk in drive d:. If no drive is specified, MS-DOS prints the volume ID of the disk in the default drive.

MS-DOS EDITING AND FUNCTION KEYS

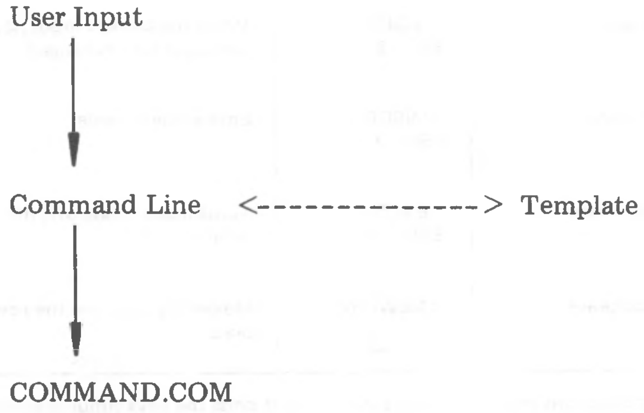
SPECIAL EDITING KEYS

The special editing keys deserve particular emphasis because they depart from the way in which most operating systems handle command input. You do not have to type the same sequences of keys repeatedly, because the last command line is automatically placed in a special storage area called a template.

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

- A command line can be instantly repeated by pressing one (or two) key(s).
- If you make a mistake in the command line, you can edit it and retry without having to retype the entire command line.
- A command line that is similar to a preceding command line can be edited and executed with a minimum of typing by pressing a special editing key.

The relationship between the command line and the template is shown in Figure 6.1.



User Input

Command Line <--------------> Template

COMMAND.COM

Figure 6.1 Command line and template

You type a command to MS-DOS on the command line. When you press the NEWLINE <CR> key, the command is automatically sent to the command processor (COMMAND.COM) for execution. At the same time, a copy of this command is sent to the template. You can now recall the command or modify it with MS-DOS special editing keys.

Table 6.2 contains a complete list of the special editing keys. Each of these keys is more fully described in Chapter 7, Line Editor (EDLIN), where they can be used to edit your text files.

Function	Key(s) *	Description
Copy one character	<COPY1> ESC S or F1	Copies one character from the template to the command line.
Copy up to character	<COPYUP> ESC T or F1 (sic)	Copies characters up to the character specified in the template and puts these characters on the command line.
Copy template	<COPYALL> ESC U or F3	Copies all remaining characters in the template to the command line.
Skip one character Not in new manual	<SKIP1> ESC V	Skips over (does not copy) a character in the template.
Skip up to character	<SKIPUP> ESC W or F4	Skips over (does not copy) the characters in the template up to the character specified.
Kill line ~~Quit input~~	KILL <~~VOID~~> ESC ~~E~~ J or F5	Voids ~~the current input~~ line on template; ~~leaves the template unchanged.~~ current input sent to template.
Insert mode	<INSERT> ESC P or F7 (sic)	Enters insert mode, or turns it off.
Replace mode	<EXIT> ESC Q or F7	Enters insert mode or turns it off; ~~Turns insert mode off~~; this is the default mode. ESC P and ESC Q can be used interchangeably; they are toggle switches.
Quit input ~~New template~~	VOID <~~NEWLINE~~> F9 or ✓ ESC E	~~Makes the new line the new template.~~ Voids the current input. leaves the template unchanged.

~~* Most functions require a 2-key entry. Do not press the keys simultaneously.~~

Table 6.2 Special editing functions

~~Notice in the table that~~ A an editing function is usually initiated with a function key, but you can also press the corresp. combination of keys. In this case, ~~(except for <NEWLINE>) is initiated with two keys.~~ p Press the ESC key first and then the ~~editing~~ second key. Do not press the keys simultaneously.

Consider the following examples. In the examples, the name of the function key is used, not the actual key.

If you type the following command

```
DIR PROG.COM
```

MS-DOS displays information about the file PROG.COM on your screen. The command line is also saved in the template. To repeat the command, just use the editing keys: <COPYALL> and ~~<CR>~~. NEWLINE

The repeated command is displayed on the screen as you type:

```
<COPYALL>DIR PROG.COM~~<CR>~~ NEWLINE
```

Notice that pressing the <COPYALL> key causes the contents of the template to be copied to the command line; pressing ~~<CR>~~ NEWLINE causes the command line to be sent to the command processor for execution.

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

```
<COPY~~ALL~~ UP>C
```

Typing <COPY~~ALL~~ UP>C copies all characters from the template to the command line, up to but not including "C". MS-DOS displays:

```
DIR PROG._
```

Note that the underline is your cursor. Now type:

```
ASM
```

The result is:

```
DIR PROG.ASM_
```

The command line "DIR PROG.ASM" is now in the template and

ready to be sent to the command processor for execution. To do this, press <~~CR~~>.
NEWLINE

Now assume that you want to execute the following command:

    TYPE PROG.ASM

To do this, type:

NEWLINE

    TYPE<INSERT> <COPYALL><~~RETURN~~>

Notice that when you are typing, the characters are entered directly into the command line and overwrite corresponding characters in the template. This automatic replacement feature is turned off when you press the insert key. Thus, the characters "TYPE" replace the characters "DIR" in the template. To insert a space between "TYPE" and "PROG.ASM", you pressed <INSERT> and then the space bar. Finally, to copy the rest of the template to the command line, you pressed <COPYALL> and then <~~CR~~>. NEWLINE The command TYPE PROG.ASM has been processed by MS-DOS, and the template becomes "TYPE PROG.ASM".

If you had misspelled "TYPE" as "BYTE", a command error would have occurred. Still, instead of throwing away the whole command, you could save the misspelled line before you press <~~CR~~> by creating a new template with the <~~NEWLINE~~> key:
NEWLINE KILL

KILL

    BYTE PROG.ASM<~~NEWLINE~~>

You could then edit this erroneous command by typing:

    T<COPY1>P<COPYALL>

The <COPY1> key copies a single character from the template to the command line. The resulting command line is then the command that you want:

    TYPE PROG.ASM

As an alternative, you can use the same template containing BYTE PROG.ASM and then use the <SKIP1> and <INSERT> keys to achieve the same result:

    <SKIP1><SKIP1><COPY1><INSERT>YP<COPYALL>

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

<SKIP1>	–	Skips over 1st template character
<SKIP1>	–	Skips over 2nd template character
<COPY1>	T	Copies 3rd template character
<INSERT>YP	TYP	Inserts two characters
<COPYALL>	TYPE PROG.ASM	Copies rest of template

Notice that <SKIP1> does not affect the command line. It affects the template by deleting the first character. Similarly, <SKIPUP> deletes characters in the template, up to but not including a given character.

These special editing keys can add to your effectiveness at the keyboard. The next section describes control character functions that can also help when you are typing commands.

CONTROL CHARACTER FUNCTIONS

A control character function is a function that affects the command line. You have already learned about <CONTROL-C> and <CONTROL-S>. Other control character functions are summarized in the following table.

Remember that when you type a control character, such as <CONTROL-C>, you must hold down the control key and then press the "C" key.

Add 6.6 - 6.8

PROGRAMMING FUNCTION KEYS.

Control Character	Function
<CONTROL-C>	Aborts current command.
<CONTROL-H>	Removes last character from command line, and erases character from terminal screen (same as Backspace key).
<CONTROL-J>	Inserts physical end-of-line, but does not empty command line. Use the <LINE FEED> key to extend the current logical line beyond the physical limits of one terminal screen.
<CONTROL-P> or <CONTROL-N>	Echoes terminal output to the line printer. Press this key again to cancel echoing.
<CONTROL-S>	Suspends display of output to terminal screen. Press any key to resume.
<CONTROL-X>	Cancels the current line; empties the command line; and then outputs a back slash (\), carriage return, and line feed. The template used by the special editing commands is not affected.

Table 6.3 Control character functions

Note. If you press CONTROL-P during a printout and continue keyboard entry, the printout will be a mixture of the keyboard entries and information already on the print spool. Be sure your printout is completed before using the CONTROL-P character.

LINE EDITOR (EDLIN)

GENERAL INFORMATION

In this chapter, you learn how to use the Line Editor (EDLIN). You can use EDLIN to create, change, and display files, whether they are source program or text files. Specifically, you can use EDLIN to perform the following functions:

- Create new source files and save them.
- Update existing files and save both the updated and original files.
- Delete, edit, insert, and display lines.
- Search for, delete, or replace text within one or more lines.

The text in files created or edited by EDLIN is divided into lines, each up to 253 characters long. Line numbers are generated and displayed by EDLIN during the editing process, but are not actually present in the saved file.

When you insert lines, all line numbers following the inserted text advance automatically by the number of lines being inserted. When you delete lines in a file, all line numbers following the deleted text decrease automatically by the number of lines deleted. As a result, lines are always numbered consecutively in your file.

HOW TO START EDLIN

To start EDLIN, type:

```
EDLIN <filespec>
```

If you are creating a new file, the <filespec> should be the name of the file you wish to create. If EDLIN does not find this file on a drive, EDLIN creates a new file with the name you specify. The following message and prompt are displayed:

```
New file
*
 _
```

Notice that the prompt for EDLIN is an asterisk (*).

You can now type lines of text into your new file. To begin entering text, you must enter an I (Insert) command to insert lines. The I command is discussed later in this chapter.

If you want to edit an existing file, <filespec> should be the name of the file you want to edit. When EDLIN finds the file you specify on the designated or default drive, the file is loaded into memory. If the entire file can be loaded, EDLIN displays the following message on your screen:

```
End of input file
*
```

You can then edit the file using EDLIN editing commands.

If the file is too large to be loaded into memory, EDLIN loads lines until memory is 3/4 full, and then displays the * prompt. You can then edit the portion of the file that is in memory.

To edit the remainder of the file, you must save some of the edited lines on disk to free memory; then EDLIN can load the unedited lines from disk into memory. Refer to the Write and Append commands in this chapter for the procedure.

When you complete the editing session, you can save the original and the updated (new) files by using the End command. The End command is discussed in this chapter in the section EDLIN Commands. The original file is renamed with an extension of .BAK, and the new file has the filename and extension you specify in the EDLIN command. The original .BAK file is not erased until the end of the editing session, or until disk space is needed by the editor (EDLIN).

Do not try to edit a file with a filename extension of .BAK because EDLIN assumes that any .BAK file is a backup file. If you find it necessary to edit such a file, rename the file with another extension (using the MS-DOS RENAME command discussed in Chapter 5); then start EDLIN and specify the new <filespec>.

SPECIAL EDITING KEYS

The special editing keys and template discussed in Chapter 6 can be used to edit your text files. These keys are discussed in detail in this section.

Table 7.1 summarizes the commands, codes, and functions. Descriptions of the special editing keys follow the table.

Function	Key(s) *	Description
Copy one character	<COPY1> F1 or ESC S	Copies one character from the template to the ~~new line.~~ command line.
Copy up to character	<COPYUP> F1 or ESC T	Copies all characters from upto the character specified in the template ~~to the new line, up to the character specified.~~ and puts these characters into the command line.
Copy template	<COPYALL> F3 or ESC U	Copies all remaining characters in the template to the ~~screen.~~ command line.
Skip one character	<SKIP1> F8 or ESC V	Does not copy (skips over) a character. in the template
Skip up to character	<SKIPUP> F4 or ESC W	Skips over (does not copy) ~~Does not copy (skips over)~~ the characters in the template, up to the character specified.
Quit input	<VOID> F9 or ESC E	Voids the current input; leaves the template unchanged.
Insert mode	<INSERT> F7 or ESC P	Enters insert mode. or turns it off.
Replace mode	<EXIT> F7 or ESC Q	Enters insert mode or turns it off; ~~Turns insert mode off~~; this is the default. *
Kill Line	<KILL> F5 or ESC J	Voids line on template; current input is sent to template.
New template	<NEWLINE> ↲ ESC J	Makes the new line the new template.
* Most functions require a 2-key entry. Do not press the keys simultaneously.		

Table 7.1 Special editing keys

* (ESC P and ESC Q can be used interchangeably, they are toggle switches).

<COPY1>

KEY

F1 or ESC S

PURPOSE

Copies one character from the template to the command line.

COMMENTS

Pressing the <COPY1> key copies one character from the template to the command line. When the <COPY1> key is pressed, one character is inserted in the command line and insert mode is automatically turned off.

EXAMPLE

Assume that the screen shows:

```
1:*This is a sample file.
1:* _
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the <COPY1> key copies the first character (T) to the second of the two lines displayed:

```
        1:*This is a sample file
<COPY1> 1:*T_
```

Each time the <COPY1> key is pressed, one more character appears:

```
<COPY1> 1:*Th_
<COPY1> 1:*Thi_
<COPY1> 1:*This_
```

<COPYUP>

KEY

F2 or ESC T

PURPOSE

Copies multiple characters up to a given character.

COMMENTS

Pressing the <COPYUP> key copies all characters up to a given character from the template to the command line. The given character is the next character typed after <COPYUP>; it is not copied or displayed on the screen. Pressing the <COPYUP> key causes the cursor to move to the single character that is specified in the command. If the template does not contain the specified character, nothing is copied. Pressing <COPYUP> also automatically turns off insert mode.

EXAMPLE

Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the <COPYUP> key copies all characters up to the character specified immediately after the <COPYUP> key.

```
             1:*This is a sample file
<COPYUP>p    1:*This is a sam_
```

<COPYALL>

KEY

F3 ✓

ESC U

PURPOSE

Copies template to command line.

COMMENTS

Pressing the <COPYALL> key copies all remaining characters from the template to the command line. Regardless of the cursor position at the time the <COPYALL> key is pressed, the rest of the line appears, and the cursor is positioned after the last character on the line.

EXAMPLE

Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the <COPYALL> key copies all characters from the template (shown in the upper line displayed) to the line with the cursor (the lower line displayed):

```
          1:*This is a sample file   (template)
<COPYALL> 1:*This is a sample file._ (command line)
```

Also, insert mode is automatically turned off.

<SKIP1>

KEY

F8 or ESC V

PURPOSE

Skips over one character in the template.

COMMENTS

Pressing the <SKIP1> key skips over one character in the template. Each time you press the <SKIP1> key, one character is not copied from the template. The action of the <SKIP1> key is similar to the <COPY1> key, except that <SKIP1> skips a character in the template rather than copying it to the command line.

EXAMPLE

Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the <SKIP1> key skips over the first character ("T").

```
        1:*This is a sample file
<SKIP1> 1:*_
```

The cursor position does not change and only the template is affected. To see how much of the line has been skipped over, press the <COPYALL> key, which moves the cursor beyond the last character of the line.

```
          1:*This is a sample file.
  <SKIP1> 1:*_
<COPYALL> 1:*his is a sample file._
```

<SKIPUP>

KEY

F4 or ESC W

PURPOSE

Skips multiple characters in the template up to the specified character.

COMMENTS

Pressing the <SKIPUP> key skips over all characters up to a given character in the template. This character is not copied and is not shown on the screen. If the template does not contain the specified character, nothing is skipped over. The action of the<SKIPUP> key is similar to the<COPYUP> key, except that<SKIPUP> skips over characters in the template rather than copying them to the command line.

EXAMPLE

Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the <SKIPUP> key skips over all the characters in the template up to the character pressed after the<SKIPUP> key:

```
            1:*This is a sample file
<SKIPUP>p   1:*_
```

The cursor position does not change. To see how much of the line has been skipped over, press the <COPYALL> key to copy the template. This moves the cursor beyond the last character of the line:

```
             1:*This is a sample file:
 <SKIPUP>p   1:*_
<COPYALL>    1:*ple file._
```

<VOID>

KEY

F9 or ESC E

PURPOSE

Quits input and empties the command line.

COMMENTS

Pressing the <VOID> key empties the command line, but it leaves the template unchanged. <VOID> also prints a back slash (\ ), carriage return, and line feed, and turns insert mode off. The cursor (indicated by the underline) is positioned at the beginning of the line. Pressing the <COPYALL> key copies the template to the command line and the command line appears as it was before <VOID> was pressed.

EXAMPLE

Assume that the screen shows:

```
    1:*This is a sample file.
    1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you want to replace the line with "Sample File":

```
    1:*This is a sample file.
    1:*Sample File_
```

To cancel the line you just entered (Sample File), and to keep "This is a sample file.", press <VOID>. Notice that a backslash appears on the Sample File line to tell you it is cancelled.

```
           1:*This is a sample file.
   <VOID>  1:*Sample File\
           1:_
```

Press <CR> to keep the original line, or to perform any other editing functions. If <COPYALL> is pressed, the original template is copied to the command line:

```
   <COPYALL> 1: This is a sample file._
```

<INSERT>

KEY

F7 or ESC P

PURPOSE

Enters insert mode.

COMMENTS

Pressing the <INSERT> key causes EDLIN to enter insert mode. The current cursor position in the template is not changed. The cursor does move as each character is inserted. However, when you are finished inserting characters, the cursor is positioned at the same character as it was before the insertion began. Thus, characters are inserted in front of the character to which the cursor points.

EXAMPLE

Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you press the <COPYUP> and "f" keys:

```
              1:*This is a sample file
<COPYUP>f     1:*This is a sample _
```

Now press the <INSERT> key and insert the characters "edit" and a space:

```
                 1:*This is a sample file.
<COPYUP>f        1:*This is a sample _
<COPYUP>edit     1:*This is a sample edit _
```

If you now press the <COPYALL> key, the rest of the template is copied to the line:

```
             1:*This is a sample edit
<COPYALL>    1:*This is a sample edit file._
```

If you press the <~~CR~~ NEWLINE> key, the remainder of the template is truncated, and the command line ends at the end of the insert:

<INSERT>edit <~~CR~~ NEWLINE> 1:*This is a sample edit _

<EXIT>

KEY

F7 or ESC Q

PURPOSE

Enters replace mode.

COMMENTS

Pressing the <EXIT> key causes EDLIN to exit insert mode and to enter replace mode. All the characters you type overstrike and replace characters in the template. When you start to edit a line, replace mode is in effect. If the ~~<CR>~~ key is pressed, the remainder of the template is deleted. NEWLINE

EXAMPLE

Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you then press <COPYUP>m, <INSERT>lary, <EXIT> tax, and then <COPYALL>:

```
                   1:*This is a sample file.
<COPYUP>m          1:*This is a sa_
<INSERT>lary       1:*This is a salary_
<EXIT> tax         1:*This is a salary tax_
<COPYALL>          1:*This is a salary tax file._
```

Notice that you inserted "lary" and replaced "mple" with "tax." If you type characters that extend beyond the length of the template, the remaining characters in the template are automatically appended when you press <COPYALL>.

<NEWLINE>

KEY

F5 or ESC J

PURPOSE

Creates a new template.

COMMENTS KILL

Pressing the <~~NEWLINE~~> key copies the current command line to the template. The contents of the old template are deleted. Pressing KILL <~~NEWLINE~~> outputs an @ ("at sign" character), a carriage return, and a line feed. The command line is also emptied and insert mode is turned off.

NOTE: KILL <~~NEWLINE~~> performs the same function as the <VOID> key, except that the template is changed and an @ ("at sign" character) is printed instead of a \ (backslash).

EXAMPLE

Assume that the screen shows:

```
1:*This is a sample file.
1:*_
```

At the beginning of the editing session, the cursor (indicated by the underline) is positioned at the beginning of the line. Assume that you enter <COPYUP>m, <INSERT>lary, <EXIT> tax, and then <COPYALL>:

```
                       1:*This is a sample file.
<COPYUP>m              1:*This is a sa_
<INSERT>lary           1:*This is a salary_
<EXIT> tax             1:*This is a salary tax_
<COPYALL>              1:*This is a salary tax file._
```

At this point, assume that you want this line to be the new template; press the <~~NEWLINE~~> KILL key:

```
KILL
<~~NEWLINE~~>1:*This is a salary tax file. @
```

The @ indicates that this new line is now the new template. Additional editing can be done using the new template.

EDLIN COMMANDS

This section describes the individual EDLIN commands that perform editing functions on lines of text. Before using an EDLIN command, read the conventions and options that apply to all commands.

FORMAT CONVENTIONS

1. Pathnames are acceptable as options to commands. For example, typing EDLIN\BIN\USER\JOE\TEXT.TXT allows you to edit the TEXT.TXT file in the subdirectory JOE.
2. You can reference line numbers relative to the current line (the line with the asterisk). Use a minus sign with a number to indicate lines before the current line. Use a plus sign with a number to indicate lines after the current line.

   Example:

   ```
   -10, +10L
   ```

   This command lists 10 lines before the current line, the current line, and 10 lines after the current line.
3. Multiple commands may be issued on one command line. When you issue a command to edit a single line using a line number (<line>), a semicolon must separate commands on the line. Otherwise, one command may follow another without any special separators. In the case of a Search or Replace command, the <string> may be ended by a <CONTROL-Z> instead of a <~~CR~~>.
   NEWLINE

   Examples:

   ```
   15;-5,+5L
   ```

   The command line in the next example searches for "This string" and then displays 5 lines before and 5 lines after the line containing the matched string. If the search fails, then the displayed lines are those line numbers relative to the current line.

   ```
   SThis string<CONTROL-Z>-5,+L
   ```

4. You can type EDLIN commands with or without a space between the line number and command. For example, to delete line 6, the command 6D is the same as 6 D.
5. It is possible to insert a control character (such as CONTROL-C) into text by using the quote character CONTROL-V before it while in insert mode. CONTROL-V tells MS-DOS to recognize the next capital letter typed as a control character. It is also possible to use a control character in any of the string arguments of Search or Replace by using the special quote character. For example:

   S<CONTROL-V>Z
   finds the first occurrence
   of CONTROL-Z in a file

   R<CONTROL-V>Z<CONTROL-Z>foo
   replaces all occurrences
   of CONTROL-Z in a file with foo

   S<CONTROL-V>C<CONTROL-Z>bar
   replaces all occurrences
   of CONTROL-C with bar

   It is possible to insert CONTROL-V into the text by typing CONTROL-V-V.
6. The CONTROL-Z character ordinarily tells EDLIN, "This is the end of the file." If you have CONTROL-Z characters elsewhere in your file, you must tell EDLIN that these other control characters do not mean "End of File." Use the /B switch to tell EDLIN to ignore any CONTROL-Z characters in the file and to show you the entire file.

The EDLIN commands are summarized in the following table. They are also described in further detail following the description of command options.

Command	Purpose
<line>	Edits line no.
A	Appends lines
C	Copies lines
D	Deletes lines
E	Ends editing
I	Inserts lines
L	Lists text
M	Moves lines
P	Pages text
Q	Quits editing
R	Replaces lines
S	Searches text
T	Transfers text
W	Writes lines

Table 7.2 EDLIN commands

COMMAND OPTIONS

Several EDLIN commands accept one or more options. The effect of a command option varies, depending on with which command it is used. The following list describes each option.

<line>

<line> indicates a line number that you type. Line numbers must be separated by a comma or a space from other line numbers, other options, and from the command.

<line> may be specified in one of three ways:

- Number (n). Any number less than 65534 - - If a number larger than the largest existing line number is specified, then <line> means the line after the last line number.
- Period (.) - - If a period is specified for <line>, then <line> means the current line number. The current line is the last line edited, and is not necessarily the last line displayed. The current line is marked on your screen by an asterisk (*) between the line number and the first character.
- Pound (#) - - The pound sign indicates the line after the last line number. If you specify # for <line>, this has the same effect as specifying a number larger than the last line number.

NEWLINE

<CR>

A ~~carriage return~~ <NEWLINE> entered without any of the <line> specifiers directs EDLIN to use a default value appropriate to the command.

?

The question mark option directs EDLIN to ask you if the correct string has been found. The question mark is used only with the Replace and Search commands. Before continuing, EDLIN waits for either a Y or <~~CR~~> NEWLINE for a yes response, or for any other key for a no response.

<string>

<string> represents text to be found, to be replaced, or to replace other text. The <string> option is used only with the Search and Replace commands. Each <string> must be ended by a <CONTROL-Z> or a <~~CR~~> NEWLINE (see the Replace command for details). Do not leave spaces between strings or between a string and its command letter, unless you want those spaces to be part of the string.

(A)PPEND

NAME

Append

PURPOSE

Adds the specified number of lines from disk to the file being edited in memory. The lines are added at the end of lines that are currently in memory.

SYNTAX

[<n>] A

COMMENTS

This command is meaningful only if the file being edited is too large to fit into memory. As many lines as possible are read into memory for editing when you start EDLIN.

To edit the remainder of the file that will not fit into memory, lines that have already been edited must be written to disk. Then you can load unedited lines from disk into memory with the Append command. (Refer to the Write command in this chapter for information on how to write edited lines to disk.)

If you do not specify the number of lines to append, lines are appended to memory until available memory is 3/4 full. No action is taken if available memory is already 3/4 full.

The message "End of input file" is displayed when the Append command has read the last line of the file into memory.

(C)OPY

NAME

Copy

PURPOSE

Copies a range of lines to a specified line number. The lines can be copied as many times as you want by using the =countŎ option.

SYNTAX

```
[<line>] , [<line>] ,<line> , [<count>] C
```

COMMENTS

If you do not specify a number in <count>, EDLIN copies the lines one time. If the first or the second <line> is omitted, the default is the current line. The file is renumbered automatically after the copy.

The line numbers must not overlap or you will get an "Entry error" message. For example, 3,20,15C would result in an error message.

EXAMPLES

Assume that the following file exists and is ready to edit:

```
1:  This is a sample file
2:  used to show copying lines.
3:  See what happens when you use
4:  the Copy command
5:  (the C command)
6:  to copy text in your file.
```

You can copy this entire block of text by issuing the following command:

```
1,6,7C
```

The result is:

```
1:  This is a sample file
2:  used to show copying lines.
```

```
 3:  See what happens when you use
 4:  the Copy command
 5:  (the C command)
 6:  to copy text in your file.
 7:  This is a sample file
 8:  used to show copying lines.
 9:  See what happens when you use
10:  the Copy command
11:  (the C command)
12:  to copy text in your file.
```

If you want to place the text within other text, the third <line> option should specify the line before which you want the copied text to appear. For example, assume that you want to copy lines and insert them within the following file:

```
 1:  This is a sample file
 2:  used to show copying lines.
 3:  See what happens when you use
 4:  the Copy command
 5:  (the C command)
 6:  to copy text in your file.
 7:  You can also use COPY
 8:  to copy lines of text
 9:  to the middle of your file.
10:  End of sample file.
```

The command 3,6,9C results in the following file:

```
 1:  This is a sample file
 2:  used to show copying lines.
 3:  See what happens when you use
 4:  the Copy command
 5:  (the C command)
 6:  to copy text in your file.
 7:  You can also use COPY
 8:  to copy lines of text
 9:  to the middle of your file.
10:  See what happens when you use
11:  the Copy command
12:  (the C command)
13:  to copy text in your file.
14:  End of sample file.
```

(D)ELETE

NAME

Delete

PURPOSE

Deletes a specified range of lines in a file.

SYNTAX

```
[<line>] [,<line>] D
```

COMMENTS

If the first <line> is omitted, that option will default to the current line (the line with the asterisk next to the line number). If the second <line> is omitted, then just the first <line> will be deleted. When lines have been deleted, the line immediately after the deleted section becomes the current line and has the same line number as the first deleted <line> had before the deletion occurred.

EXAMPLES

Assume that the following file exists and is ready to edit:

```
 1:  This is a sample file
 2:  used to show dynamic line numbers.
 3:  See what happens when you use
 4:  Delete and Insert
  .
  .
  .
25:  (the D and I commands)
26:  to edit the text
27:* in your file.
```

To delete multiple lines, type <line>,<line>D:

```
5,24D
```

The result is:

```
1:  This is a sample file
2:  used to show dynamic line numbers.
```

```
3:  See what happens when you use
4:  Delete and Insert
5:  (the D and I commands)
6:  to edit text
7:* in your file.
```

To delete a single line, type:

```
6D
```

The result is:

```
1:  This is a sample file
2:  used to show dynamic line numbers.
3:  See what happens when you use
4:  Delete and Insert
5:  (the D and I commands)
6:* in your file.
```

Next, delete a range of lines from the following file:

```
1:  This is a sample file
2:  used to show dynamic line numbers.
3:* See what happens when you use
4:  Delete and Insert
5:  (the D and I commands)
6:  to edit text
7:  in your file.
```

To delete a range of lines beginning with the current line, type:

```
,6D
```

The result is:

```
1:  This is a sample file
2:  used to show dynamic line numbers.
3:* in your file.
```

Notice that the lines are automatically renumbered.

<line> EDIT

NAME

Edit

PURPOSE

Edits line of text.

SYNTAX

[<line>]

COMMENTS

When a line number is typed, EDLIN displays the line number and text; then, on the line below, EDLIN reprints the line number. The line is now ready for editing. You may use any of the EDLIN editing commands to edit the line. The existing text of the line serves as the template until the <~~CR~~ NEWLINE> key is pressed.

If no line number is typed (that is, if only the <~~CR~~ NEWLINE> key is pressed), the line after the current line (marked with an asterisk) is edited. If no changes to the current line are needed and the cursor is at the beginning or end of the line, press the <~~CR~~ NEWLINE> key to accept the line as is.

CAUTION

> If the <~~CR~~ NEWLINE> key is pressed while the cursor is in the middle of the line, the remainder of the line is deleted.

EXAMPLE

Assume that the following file exists and is ready to edit:

```
1:  This is a sample file.
2:  used to show
3:  the editing of line
4:* four.
```

To edit line 4, type:

```
4
```

The contents of the line are displayed with a cursor below the line:

```
4:* four.
4:*_
```

Now, using the <COPYALL> special editing key, type:

```
<INSERT>number          4: number_
<COPYALL><CR>           4: number four.
   NEWLINE              5:*_
```

(E)ND

NAME

End

PURPOSE

Ends the editing session.

SYNTAX

E

COMMENTS

This command saves the edited file on disk, renames the original input file <filename>.BAK, and then exits EDLIN. If the file is created during the editing session, no .BAK file is created.

The E command takes no options. Therefore, you cannot tell EDLIN on which drive to save the file. The drive you want to save the file on must be selected when the editing session is started. If the drive is not selected when EDLIN is started, the file is saved on the disk in the default drive. It is still possible to COPY the file to a different drive using the MS-DOS COPY command.

You must be sure that the disk contains enough free space for the entire file. If the disk does not contain enough free space, the write is aborted and the edited file is lost, although part of the file might be written out to the disk.

EXAMPLE

NEWLINE

```
E<~~CR~~>
```

After execution of the E command, the MS-DOS default drive prompt (for example, A>) is displayed.

(I)NSERT

NAME

Insert

PURPOSE

Inserts text immediately before the specified <line>.

SYNTAX

[<line>] I

COMMENTS

If you are creating a new file, the I command must be given before text can be typed (inserted). Text begins with line number 1. Successive line numbers appear automatically each time <CR> is pressed.

EDLIN remains in insert mode until <CONTROL-C> is typed. When the insert is completed and insert mode has been exited, the line immediately following the inserted lines becomes the current line. All line numbers following the inserted section are incremented by the number of lines inserted.

If <line> is not specified, the default is the current line number and the lines are inserted immediately before the current line. If <line> is any number larger than the last line number, or if a pound sign (#) is specified as <line>, the inserted lines are appended to the end of the file. In this case, the last line inserted becomes the current line.

EXAMPLES

Assume that the following file exists and is ready to edit:

```
    1:  This is a sample file
    2:  used to show dynamic line numbers.
    3:  See what happens when you use
    4:  Delete and Insert
    5:  (the D and I commands)
    6:  to edit text
    7:* in your file.
```

To insert text before a specific line that is not the current line, type <line>I:

```
    7I
```

The result is:

```
    7:_
```

Now, type the new text for line 7:

```
    7:  and renumber lines
```

Then to end the insertion, press <CONTROL-Z> on the next line:

```
    8:  <CONTROL-Z>
```

Now type L to list the file. The result is:

```
    1:  This is a sample file
    2:  used to show dynamic line numbers.
    3:  See what happens when you use
    4:  Delete and Insert
    5:  (the D and I commands)
    6:  to edit text
    7:  and renumber lines
    8:* in your file.
```

To insert lines immediately before the current line, type:

```
    I
```

The result is:

```
    8:  _
```

Now, insert the following text and terminate with a <CONTROL-Z> on the next line:

```
    8:  so they are consecutive
    9:  <CONTROL-Z>
```

Now to list the file and see the result, type L:

The result is:

```
 1:  This is a sample file
 2:  used to show dynamic line numbers.
 3:  See what happens when you use
 4:  Delete and Insert
 5:  (the D and I commands)
 6:  to edit text
 7:  and renumber lines
 8:  so they are consecutive
 9:* in your file.
```

To append new lines to the end of the file, type:

```
 10I
```

This produces the following:

```
 10:_
```

Now, type the following new lines:

```
10:  The insert command can place new lines
11:  in the file; there's no problem
12:  because the line numbers are dynamic;
13:  they'll go all the way to 65533.
```

End the insertion by pressing <CONTROL-Z> on line 14. The new lines appear at the end of all previous lines in the file. Now type the list command, L:

The result is:

```
 1:  This is a sample file
 2:  used to show dynamic line numbers.
 3:  See what happens when you use
 4:  Delete and Insert
 5:  (the D and I commands)
 6:  to edit text
 7:  and renumber lines
 8:  so they are consecutive
 9:  in your file.
10:  The insert command can place new lines
11:  in the file; there's no problem
12:  because the line numbers are dynamic;
13:  they'll go all the way to 65533.
```

(L)IST

NAME

List

PURPOSE

Lists a range of lines, including the two lines specified.

SYNTAX

```
[<line>] [,<line>]L
```

COMMENTS

Default values are provided if either one or both of the options are omitted. If you omit the first option, as in:

```
,<line>L
```

the display starts 11 lines before the current line and ends with the specified <line>. The beginning comma is required to indicate the omitted first option.

NOTE: If the specified <line> is more than 11 lines before the current line, the display is the same as if you omitted both options.

If you omit the second option, as in

```
<line>L
```

23 lines are displayed, starting with the specified <line>.

If you omit both parameters, as in

```
L
```

23 lines are displayed: the 11 lines before the current line, the current line, and the 11 lines after the current line. If there are less than 11 lines before the current line, more than 11 lines after the current line are displayed to make a total of 23 lines.

EXAMPLES

Assume that the following file exists and is ready to edit:

```
     1:  This is a sample file
     2:  used to show dynamic line numbers
     3:  See what happens when you use
     4:  Delete and Insert
     5:  (the D and I commands)
     .
     .
     .
    15:* The current line contains an asterisk.
     .
     .
     .
    26:  to edit text
    27:  in your file.
```

To list a range of lines without reference to the current line, type <line> ,<line>L:

```
2,5L
```

The result is:

```
 2:  used to show dynamic line numbers.
 3:  See what happens when you use
 4:  Delete and Insert
 5:  (the D and I commands)
```

To list a range of lines beginning with the current line, type, <line> L:

```
,26L
```

The result is:

```
15:* The current line contains an asterisk.
  .
  .
  .
26:  to edit text
```

To list a range of 23 lines centered around the current line, type only L:

```
L
```

The result is:

```
 4:  Delete and Insert
 5:  (the D and I commands)
   .
   .
   .
13:  The current line is listed in the middle of the range.
14:  The current line remains unchanged by the L command.
15:* The current line contains an asterisk.
   .
   .
   .
26:  to edit text.
```

(M)OVE

NAME

Move

PURPOSE

Moves a range of text to the line specified.

SYNTAX

```
[<line>],[<line>],<line>M
```

COMMENTS

Use the Move command to move a block of text (from the first <line> to the second <line> to another location in the file. The lines are renumbered according to the direction of the move. For example,

```
,+25,100M
```

moves the text from the current line plus 25 lines to line 100. If the line numbers overlap, EDLIN displays an "Entry error" message.

To move lines 20-30 to line 100, type:

```
20,30,100M
```

(P)AGE

NAME

Page

PURPOSE

Pages through a file 23 lines at a time.

SYNTAX

[<line>] [,<line>]P

COMMENTS

If the first <line> is omitted, that number defaults to the current line plus one. If the second <line> is omitted, 23 lines are listed. The new current line becomes the last line displayed and is marked with an asterisk.

(Q)UIT

NAME

Quit

PURPOSE

Quits the editing session, does not save any editing changes, and exits to the MS-DOS operating system.

SYNTAX

Q

COMMENTS

EDLIN prompts you to make sure you don't want to save the changes.

Type Y if you want to quit the editing session. No editing changes are saved and no .BAK file is created. Refer to the End command in this chapter for information about the .BAK file.

Type N or any other character if you want to continue the editing session.

NOTE: When started, EDLIN erases any previous copy of the file with an extension of .BAK to make room to save the new copy. If you reply Y to the "Abort edit (Y/N)?" message, your previous backup copy no longer exists.

EXAMPLE

```
    Q
    Abort edit (Y/N)?Y<CR>
A>_
```

(R)EPLACE

NAME

Replace

PURPOSE

Replaces all occurrences of a string of text in the specified range with a different string of text or blanks.

SYNTAX

```
[<line>] [,<line>] [?] R<string1><CONTROL-Z><string2>
```

COMMENTS

As each occurrence of <string1> is found, it is replaced by <string2>. Each line in which a replacement occurs is displayed. If a line contains two or more replacements of <string1> with <string2>, then the line is displayed once for each occurrence. When all occurrences of <string1> in the specified range are replaced by <string2>, the R command terminates and the asterisk prompt reappears.

If a second string is to be given as a replacement, then <string1> must be separated from <string2> with a <CONTROL-Z>. <String2> must also be ended with a <CONTROL-Z> <CR> combination or with a simple <~~CR~~>.
NEWLINE

If <string1> is omitted, then Replace takes the old <string1> as its value. If there is no old <string1> (that is, this is the first replace done), then the replacement process is terminated immediately. If <string2> is omitted, then <string1> may be ended with a <CR>. If the first <line> is omitted in the range argument (as in, <line>) then the first <line> defaults to the line after the current line. If the second <line> is omitted (as in <line> or <line>,), the second <line> defaults to # . Remember that # indicates the line after the last line of the file.

If <string1> is ended with a <CONTROL-Z> and there is no <string2>, <string2> is taken as an empty string and becomes the new replace string. For example,

```
R<string2><CONTROL-Z><CR>
```

deletes occurrences of <string1>, but

    R<string1><CR>   and
    R<CR>

replaces <string1> by the old <string2> and the old <string1> with the old <string2>, respectively. Note that "old" here refers to a previous string specified either in a Search or a Replace command.

If the question mark (?) option is given, the Replace command stops at each line with a string that matches <string1>, displays the line with <string2> in place, and then displays the prompt "O.K.?." If you press Y or the <~~CR~~ NEWLINE> key, then <string2> replaces <string1>, and the next occurrence of <string1> is found. Again, the "O.K.?" prompt is displayed. This process continues until the end of the range or until the end of the file. After the last occur-rence of <string1> is found, EDLIN displays the asterisk prompt.

If you press any key besides Y or <~~CR~~ NEWLINE> after the "O.K.?" prompt, the <string1> is left as it was in the line, and Replace goes to the next occurrence of <string1>. If <string1> occurs more than once in a line, each occurrence of <string1> is replaced individually, and the "O.K.?" prompt is displayed after each replacement. In this way, only the desired <string1> is replaced, and you can prevent unwanted substitutions.

EXAMPLES

Assume that the following file exists and is ready for editing:

```
 1: This is a sample file
 2: used to show dynamic line numbers.
 3: See what happens when you use
 4: Delete and Insert
 5: (the D and I commands)
 6: to edit text
 7: in your file.
 8: The insert command can place new lines
 9: in the file; there's no problem
10: because the line numbers are dynamic;
11: they'll go all the way to 65533.
```

To replace all occurrences of <string1> with <string2> in a specified range, type:

2,12 Rand<CONTROL-Z> or <~~CR~~ NEWLINE>

The result is:

```
4: Delete or Insert
5: (the D or I commors)
8: The insert commor can place new lines
```

Note that in the replacements, some unwanted substitutions have occurred. To avoid these and to confirm each replacement, the same original file can be used with a slightly different command.

In the next example, to replace only certain occurrences of the first <string> with the second <string>, type:

2? Rand <CONTROL-Z> or <~~CR~~ NEWLINE>

The result is:

```
4: Delete or Insert
O.K.? Y
5: (the D or I commands)
O.K.? Y
5: (the D or I commors)
O.K.? N
8: The insert commor can place new lines
O.K.? N
*_
```

Now, type the List command (L) to see the result of all these changes:

```
.
.
4: Delete or Insert
5: (The D or I commands)
.
8: The insert command can place new lines
.
.
```

(S)EARCH

NAME

Search

PURPOSE

Searches the specified range of lines for a specified string of text.

SYNTAX

NEWLINE

[<line>] [,<line>] [?]S<string><~~CR~~>

COMMENTS

The <string> must be ended with a <~~CR~~> NEWLINE. The first line that matches <string> is displayed and becomes the current line. If the question mark option is not specified, the Search command terminates when a match is found. If no line contains a match for <string>, the message "Not found" is displayed.

If the question mark option (?) is included in the command, EDLIN displays the first line with a matching string; it then prompts you with the message "O.K.?". If you press either the Y or <~~CR~~> NEWLINE key, the line becomes the current line and the search terminates. If you press any other key, the search continues until another match is found, or until all lines are searched (and the "Not found" message is displayed).

If the first <line> is omitted (as in ,<line>S <string>), the first <line> defaults to the line after the current line. If the second <line> is omitted (as in <line> S <string> or <line>, S <string>), the second <line> defaults to # (line after last line of file), which is the same as <line>, # S<string>. If <string> is omitted, Search takes the old string if there is one. (Note that "old" here refers to a string specified in a previous Search or Replace command.) If there is not an old string (that is, no previous search or replace has been done), the command terminates immediately.

EXAMPLES

Assume that the following file exists and is ready for editing:

```
    1:  This is a sample file
    2:  used to show dynamic line numbers.
    3:  See what happens when you use
```

```
 4:  Delete and Insert
 5:  (the D and I commands)
 6:  to edit text
 7:  in your file.
 8:  The insert command can place new lines
 9:  in the file; there's no problem
10:  because the line numbers are dynamic;
11:* they'll go all the way to 65533.
```

To search for the first occurrence of the string "and," type

2,12 Sand<~~CR~~ NEWLINE>

The following line is displayed:

```
4:  Delete and Insert
```

to get the "and" in line 5, modify the search command by typing:

<SKIP1><COPYALL>,12 Sand<~~CR~~ NEWLINE>

The search then continues from the line after the current line (line 4), since no first line was given. The result is:

```
5:  (the D and I commands)
```

To search through several occurrences of a string until the correct string is found, type:

```
1, ? Sand
```

The result is:

```
4:  Delete and Insert
O.K.?_
```

If you press any key (except Y or <~~CR~~ NEWLINE>), the search continues, so type N here:

```
O.K.? N
```

Continue:

```
5: (the D and I commands)
O.K.?_
```

Now press Y to terminate the search:

```
O.K.? Y
*_
```

To search for string XYZ without the verification (O.K.?), type:

```
SXYZ
```

EDLIN reports a match and continues to search for the same string when you issue the S command:

```
S
```

EDLIN reports another match.

```
S
```

EDLIN reports the string is not found.

Note that <string> defaults to any string specified by a previous Replace or Search command.

(T)RANSFER

NAME

Transfer

PURPOSE

Inserts (merges) the contents of <filename> into the file currently being edited at <line>. If <line> is omitted, then the current line is used.

SYNTAX

[<line>] T <filename>

COMMENTS

This command is useful if you want to put the contents of a file into another file or into the text you are typing. The transferred text is inserted at the line number specified by <line> and the lines are renumbered.

(W)RITE

NAME

Write

PURPOSE

Writes a specified number of lines to disk from the lines that are being edited in memory. Lines are written to disk beginning with line number 1.

SYNTAX

[<n>] W

COMMENTS

This command is meaningful only if the file you are editing is too large to fit into memory. When you start EDLIN, EDLIN reads lines into memory until memory is 3/4 full.

To edit the remainder of your file, you must write edited lines in memory to disk. Then you can load additional lines from disk into memory by using the Append command.

NOTE: If you do not specify the number of lines, lines are written until memory is 3/4 full. No action is taken if available memory is already more than 3/4 full. All lines are renumbered, so that the first remaining line becomes line number 1.

ERROR MESSAGES

When EDLIN finds an error, one of the following error messages is displayed:

```
Cannot edit .BAK file- -rename file
```

Explanation

You attempted to edit a file with a filename extension of .BAK. .BAK files cannot be edited because this extension is reserved for backup copies.

Action

If you need the .BAK file for editing purposes, you must either RENAME the file with a different extension, or COPY the .BAK file and give it a different filename extension.

```
No room in directory for file
```

Explanation

When you attempted to create a new file, either the file directory was full or you specified an illegal disk drive or an illegal filename.

Action

Check the command line that started EDLIN for illegal filename and illegal disk drive entries. If the command is no longer on the screen and if you have not yet typed a new command, the EDLIN start command can be recovered by pressing the <COPYALL> key.

If this command line contains no illegal entries, run the CHKDSK program for the specified disk drive. If the status report shows that the disk directory is full, remove the disk. Insert and format a new disk.

```
Entry Error
```

Explanation

The last command typed contained a syntax error.

Action

Retype the command with the correct syntax and press <~~C~~R>.

NEWLINE

Line too long

Explanation

During a Replace command, the string given as the replacement caused the line to expand beyond the limit of 253 characters. EDLIN aborted the Replace command.

Action

Divide the long line into two lines; then try the Replace command twice.

Disk Full- -file write not completed

Explanation

You gave the End command, but the disk did not contain enough free space for the whole file. EDLIN aborted the E command and returned you to the operating system. Some of the file may have been written to the disk.

Action

Only a portion (if any) of the file has been saved. You should probably delete that portion of the file and restart the editing session. The file is not available after this error. Always be sure that the disk has sufficient free space for the file to be written to disk before you begin your editing session.

Incorrect DOS version

Explanation

You attempted to run EDLIN under a version of MS-DOS that was not 2.0 or higher.

Action

You must make sure that the version of MS-DOS that you are using is 2.0 or higher.

Invalid drive name or file

Explanation

You have not specified a valid drive or filename when starting EDLIN.

Action

Specify the correct drive or filename.

Filename must be specified

Explanation

You did not specify a filename when you started EDLIN.

Action

Specify a filename.

Invalid parameter

Explanation

You specified a switch other than /B when starting EDLIN.

Action

Specify the /B switch when you start EDLIN.

Insufficient memory

Explanation

There is not enough memory to run EDLIN.

Action

You must free some memory by writing files to disk or by deleting files before restarting EDLIN.

File not found

Explanation

The filename specified during a Transfer command was not found.

Action

Specify a valid filename when issuing a Transfer command.

Must specify destination number

Explanation

A destination line number was not specified for a Copy or Move command.

Action

Reissue the command with a destination line number.

Not enough room to merge the entire file

Explanation

There was not enough room in memory to hold the file during a Transfer command.

Action

You must free some memory by writing some files to disk or by deleting some files before you can transfer this file.

FILE COMPARE (FC) UTILITY

GENERAL INFORMATION

It is sometimes useful to compare files on your disk. If you have copied a file and later want to compare copies to see which one is current, you can use the MS-DOS File Compare (FC) Utility.

The File Compare Utility compares the contents of two files. The difference between the two files can be output to the console or to a third file. The files being compared may be either source files (files containing source statements of a programming language), or binary files (files output by the assembler, the MS-LINK Linker utility, or by a high-level language compiler).

The comparisons are made in one of two ways: on a line-by-line or a byte-by-byte basis. The line-by-line comparison isolates blocks of lines that are different between the two files and prints those blocks of lines. The byte-by-byte comparison displays the bytes that are different between the two files.

LIMITATIONS ON SOURCE COMPARISONS

FC uses a large amount of memory as buffer (storage) space to hold the source files. If the source files are larger than available memory, FC compares what can be loaded into the buffer space. If no lines match in the portions of the files in the buffer space, FC displays only the message:

```
FILES ARE DIFFERENT
```

For binary files larger than available memory, FC compares both files completely, overlaying the portion in memory with the next portion from disk. All differences are output in the same manner as those files that fit completely in memory.

FILE SPECIFICATIONS

All file specifications use the following syntax:

```
[d:] <filename> [<.ext>]
```

d: is the letter designating a disk drive. If the drive designation is omitted, FC defaults to the operating system's (current) default drive.

filename is a 1- to 8-character name of the file.

.ext is a 1- to 3-character extension to the filename.

HOW TO USE FILE COMPARE

The syntax of FC is as follows:

```
    FC [/# /B /W /C] <filename1> <filename2>
```

FC matches the first file (filename1) against the second (filename2) and reports any differences between them. Both filenames can be pathnames. For example,

```
    FC B:\FOO\BAR\FILE1.TXT \BAR\FILE2.TXT
```

FC takes FILE1.TXT in the \FOO\BAR directory of disk B and compares it with FILE2.TXT in the \BAR directory. Since no drive is specified for filename2, FC assumes that the \BAR directory is on the disk in the default drive.

FC SWITCHES

There are four switches that you can use with the File Compare Utility:

/B

Forces a binary comparison of both files. The two files are compared byte-to-byte, with no attempt to re-synchronize after a mismatch. The mismatches are printed as follows:

```
--ADDRS----F1----F2-
xxxxxxxx    yy    zz
```

(where xxxxxxxx is the relative address of the pair of bytes from the beginning of the file). Addresses start at 00000000; yy and zz are the mismatched bytes from file1 and file2, respectively. If one of the files contains less data than the other, then a message is printed out. For example, if file1 ends before file2, then FC displays:

Data left in F2

/#,

stands for a number from 1 to 9. This switch specifies the number of lines required to match for the files to be considered as matching again after a difference is found. If this switch is not specified, it defaults to 3. This switch is used only in source comparisons.

/W

Causes FC to compress "whites" (tabs and spaces) during the comparison. Thus, multiple contiguous whites in any line are considered as a single white space. Note that although FC compresses whites, it does not ignore them. The two exceptions are beginning and ending whites in a line, which are ignored. For example (note that an underscore represents a white)

__More_data_to_be_found__

matches with

More_data_to_be_found

and with

____More___data_to_be___found____

but does not match with

___Moredata_to_be_found

This switch is used only in source comparisons.

/C

Causes the matching process to ignore the case of letters. All letters in the files are considered uppercase letters. For example,

Much_MORE_data_IS_NOT_FOUND

matches

much_more_data_is_not_found

If both the /W and /C options are specified, then FC compresses whites and ignores case. For example,

__DATA_was_found__

will match:

data_was_found

This switch is used only in source comparisons.

DIFFERENCE REPORTING

The File Compare Utility reports the differences between the two files you specify by displaying the first filename, followed by the lines that differ between the files, followed by the first line to match in both files. FC then displays the name of the second file followed by the lines that are different, followed by the first line that matches. The default for the number of lines to match between the files is 3. (If you want to change this default, specify the number of lines with the /# switch.) For example,

```
              . . .
              . . .


  ----------<filename1>
  <difference>
  <1st line to match file 2 in file1>

  ----------<filename2>
  <difference>
  <1st line to match file1 in file2>

  -------------------------------------

              . . .
              . . .
```

FC continues to list each difference.

If there are too many differences (involving too many lines), the program simply reports that the files are different and stops.

If no matches are found after the first difference is found, FC displays:

```
*** Files are different ***
```

and returns to the MS-DOS default drive prompt (for example, A>).

REDIRECTING FC OUTPUT TO A FILE

The differences and matches between the two files you specify are displayed on your screen unless you redirect the output to a file. This is accomplished in the same way as MS-DOS command redirection (refer to Chapter 4, Learning About Commands).

To compare File1 and File2 and then send the FC output to DIFFER.TXT, type:

```
FC File1 File2 > DIFFER.TXT
```

The differences and matches between File1 and File2 are put into DIFFER.TXT on the default drive.

EXAMPLES

Example 1 - - Compare (No Switches)

Assume these two ASCII files are on disk:

```
ALPHA.ASM        BETA.ASM

FILE A           FILE B
-------------------------------
A                A
B                B
C                C
D                G
E                H
F                I
G                J
H                1
I                2
M                P
N                Q
O                R
P                S
Q                T
R                U
S                V
```

```
    T                   4
    U                   5
    V                   W
    W                   X
    X                   Y
    Y                   Z
    Z
```

To compare the two files and display the differences on the terminal screen, type:

```
    FC ALPHA.ASM BETA.ASM
```

FC compares ALPHA.ASM with BETA.ASM and displays the differences on the terminal screen. All other defaults remain intact. (The defaults are: do not use tabs, spaces, or comments for matches, and do a source comparison on the two files.)

The output appears as follows on the terminal screen (the Notes do not appear):

```
-----------ALPHA.ASM
D                                    NOTE: ALPHA file
E                                    contains defg,
F                                    BETA contains g.
G

-----------BETA.ASM
G

------------------------------

-----------ALPHA.ASM
M                                    NOTE: ALPHA file
N                                    contains mno where
O                                    BETA contains j12.
P
```

```
-----------BETA.ASM
J
1
2
P

-----------------------------

-----------ALPHA.ASM
W                         NOTE: ALPHA file
                          contains w where
-----------BETA.ASM       BETA contains 45w.
4
5
W
```

Example 2 - - Compare with Number Switch

You can print the differences on the line printer using the same two source files. In this example, four successive lines must be the same to constitute a match.

Type:

```
FC /4 ALPHA.ASM BETA.ASM > PRN
```

The following output appears on the line printer:

```
----------ALPHA.ASM
D
E
F
G
H
I
M
N                         NOTE: p is the 1st of
O                         a string of 4 matches.
P
```

```
----------BETA.ASM
G
H
I
J
1
2
P

-----------------------------

----------ALPHA.ASM
W

----------BETA.ASM
4
5
W
```

NOTE: w is the 1st of a string of 4 matches.

Example 3 - - Compare with Binary Switch

This example forces a binary comparison and then displays the differences on the terminal screen using the same two source files as were used in the previous examples.

Type:

```
FC /B ALPHA.ASM BETA.ASM
```

The /B switch in this example forces binary comparison. This switch and any others must be typed before the filenames in the FC command line. The following display appears:

```
--ADDRS----F1---F2--
 00000009    44   47
 0000000C    45   48
 0000000F    46   49
 00000012    47   4A
 00000015    48   31
 00000018    49   32
 0000001B    4D   50
 0000001E    4E   51
 00000021    4F   52
 00000024    50   53
 00000027    51   54
```

```
0000002A     52   55
0000002D     53   56
00000030     54   34
00000033     55   35
00000036     56   57
00000039     57   58
0000003C     58   59
0000003F     59   5A
00000042     5A   1A
```

ERROR MESSAGES

When the File Compare Utility detects an error, one or more of the following error messages are displayed:

Incorrect DOS version
    You are running FC under a version of MS-DOS that is not 2.0 or higher.

Invalid parameter:<option>
    One of the switches that you have specified is invalid.

File not found:<filename>
    FC could not find the filename you specified.

Read error in:<filename>
    FC could not read the entire file.

Invalid number of parameters
    You have specified the wrong number of options on the FC command line.

017-0031781

IMPORTANT INFORMATION
FOR MS-DOS USERS WITH 256KB MEMORY

Do not use the /HIGH switch when linking a program with MS-LINK on systems with 256 KB memory. (The switch can never be used with PASCAL and FORTRAN programs, regardless of memory size.) While MS-LINK performs successfully, the computer "goes down" when MS-DOS attempts to load the .EXE program into memory.

If using interpretive BASIC, you can store a subroutine in high memory, but you should only use this technique if you are familiar with debugger:

1. Debug subroutine
2. Move subroutine to higher memory
3. Record register values (segment address of subroutine)
4. Load BASIC
5. Load BASIC program
6. Change DEF SEG to refer to subroutine
7. Save subroutine (BSAVE)

Finally, use the following sequence in programs that compute the size of the user area. This sequence ensures a successful program load.

1. Load the application program.
2. Save segment address of program (in CS).
3. Issue the System Call, "Modify allocated memory block":
   - move CS to ES
   - move length of program to BX
   - move 4AH to AH
   - INT 21H

4. Issue the System Call "Allocate memory":
   - move FFFFH to BX
   - move 48H to AH
   - INT 21H

5. Multiply the contents of BX by 16 to determine the size of the user area.

(For detailed information see the MS-DOS Programmer's Manual, "System Calls" chapter.)

LINKER PROGRAM (MS-LINK)

GENERAL INFORMATION

In this chapter you learn about the Linker program, called MS-LINK. Read the entire chapter before you use MS-LINK. ~~Also, for more detailed information, refer to the MS-LINK chapter in the *PROGRAMMER'S MANUAL*.~~

NOTE: If you are not going to compile and link programs, you do not need to read this chapter.

MS-LINK is a program that performs the following functions:

- Combines separately produced object modules into one relocatable load module - - a program you can run.
- Searches library files for definitions of unresolved external references.
- Resolves external cross-references.
- Produces a listing that shows both the resolution of external references and error messages.

PROGRAM OVERVIEW

When you write a program, you write it in source code. This source code is passed through a compiler which produces object modules. The object modules must be passed through the link process to produce machine language that the computer can understand directly. This machine language is in the form required for running programs.

You may wish to link (combine) several programs and run them together. Each of your programs may refer to a symbol that is defined in another object module. This reference is called an external reference.

MS-LINK combines several object modules into one relocatable load module, or Run file (called an .EXE or Executable file). As it combines modules, MS-LINK makes sure that all external references between object modules are defined. LINK can search

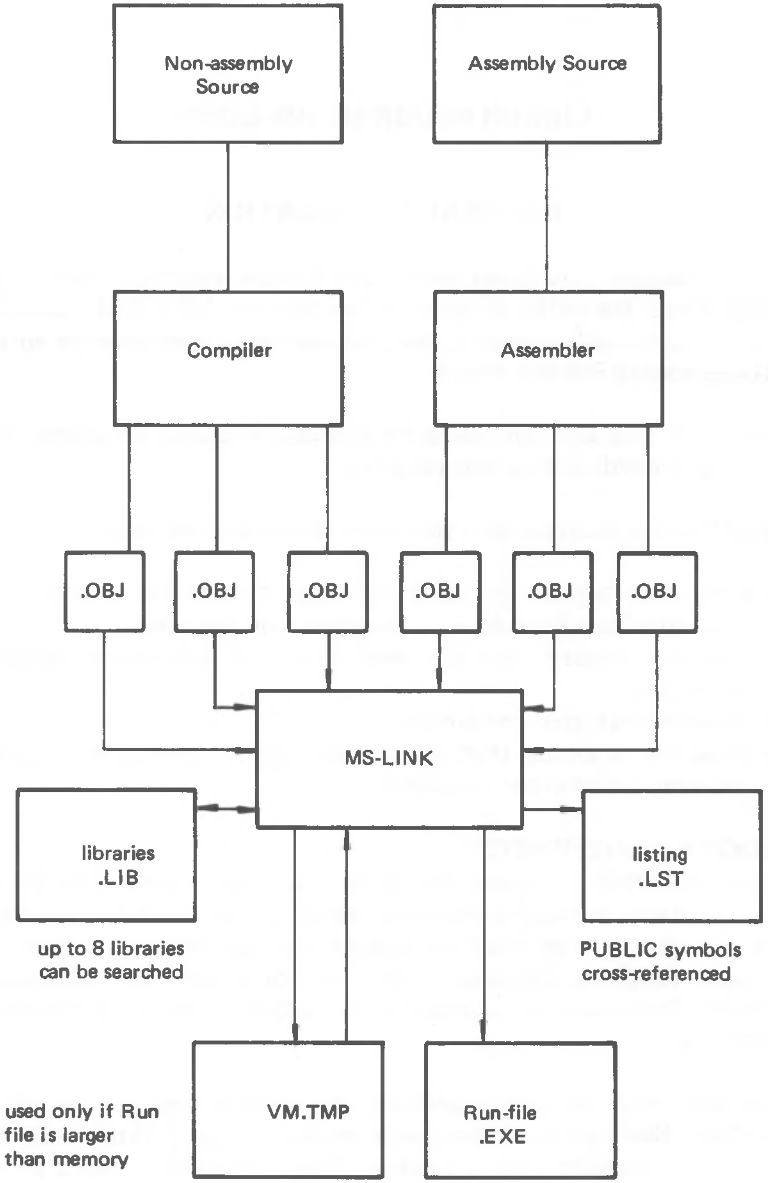


Figure 9.1 The MS-LINK Operation

several library files for definitions of any external references that are not defined in the object modules.

MS-LINK also produces a List file that shows external references resolved, and it also displays any error messages.

MS-LINK uses available memory as much as possible. When available memory is exhausted, MS-LINK creates a temporary disk file named VM.TMP.

Figure 9.1 illustrates the various parts of the MS-LINK operation.

DEFINITIONS YOU'LL NEED TO KNOW

Some of the terms used in this chapter are explained below to help you understand how MS-LINK works. Generally, if you are linking object modules compiled from BASIC, Pascal, or a high-level language, you do not need to know these terms. If you are writing and compiling programs in assembly language, however, you need to understand MS-LINK and the definitions described in this section.

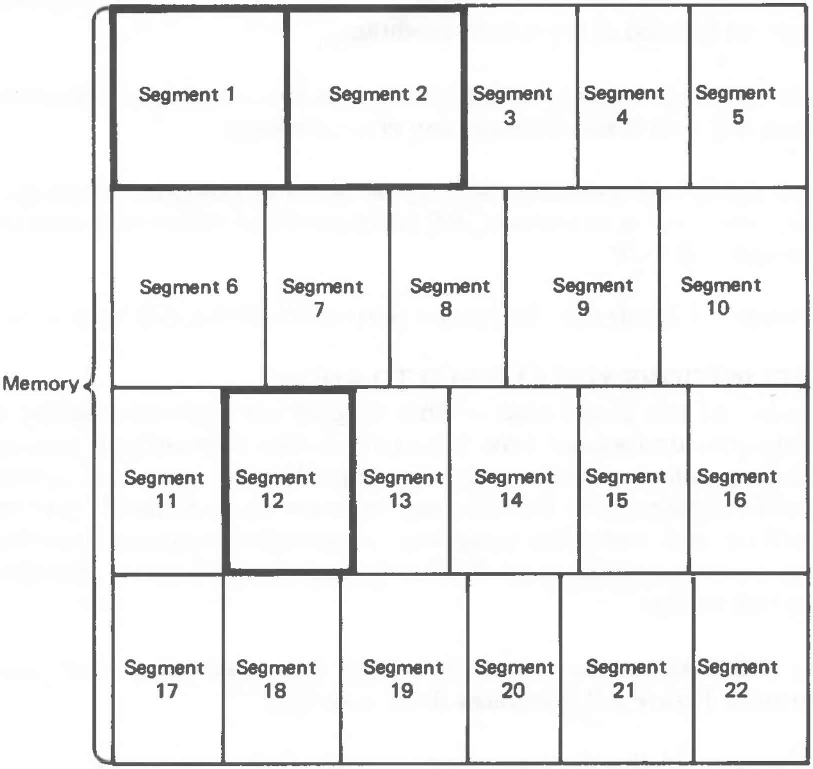
In MS-DOS, memory can be divided into segments, classes, and groups. Figure 9.2 illustrates these concepts.

Assume that the three segments have the following names.

	Segment Name	Segment Class Name
Segment 1	PROG.1	CODE
Segment 2	PROG.2	CODE
Segment 3	PROG.3	DATA

Note that segments 1, 2, and 12 have different segment names but may or may not have the same segment class name. Segments 1, 2, and 12 form a group with a group address of the lowest address of segment 1 (that is, the lowest address in memory).

Each segment has a segment name and a class name. MS-LINK loads all segments into memory by class name from the first segment encountered to the last. All segments assigned to the same class are loaded into memory contiguously.



Highlighted area = a group (64K bytes addressable)

Figure 9.2 How memory is divided

During processing, MS-LINK references segments by their addresses in memory (where they are located). MS-LINK does this by finding groups of segments.

A group is a collection of segments that fit within a 64K byte area of memory. The segments do not need to be contiguous to form a group (see illustration). The address of any group is the lowest address of the segments in that group. At link time, MS-LINK analyzes the groups, then references the segments by the address in memory of that group. A program may consist of one or more groups.

If you are writing in assembly language, you may assign the group and class names in your program. In high-level languages (BASIC, COBOL, FORTRAN, Pascal), the naming is done automatically by the compiler.

FILES THAT MS-LINK USES

MS-LINK works with one or more input files, produces two output files, may create a temporary disk file, and may be directed to search up to eight library files.

For each type of file, you may give a 3-part file specification. The format for MS-LINK file specifications is the same as that of a disk file:

[d:]<filename>[<.ext>]

- d: is the drive designation. Permissible drive designations for MS-LINK are A: through O:. The colon is always required as part of the drive designation.
- filename is any legal filename of one to eight characters.
- .ext is a 1- to 3-character extension to the filename. The period is always required as part of the extension.

Input File Extensions

If no filename extensions are given in the input (object) file specifications, MS-LINK recognizes the following extensions by default:

.OBJ	Object
.LIB	Library

Output File Extensions

MS-LINK appends the following default extensions to the output (Run and List) files:

.EXE	Run (may not be overridden)
.MAP	List (may be overridden)

VM.TMP (Temporary) File

MS-LINK uses available memory for the link session. If the files to be linked create an output file that exceeds available memory, MS-LINK creates a temporary file, names it VM.TMP, and puts it on the disk in the default drive. If MS-LINK creates VM.TMP, it displays the message:

```
VM.TMP has been created.
Do not change disk in drive,<d:>
```

Once this message has been displayed, you must not remove the disk from the default drive until the link session ends. If the disk is removed, the operation of MS-LINK will be unpredictable, and MS-LINK might display the error message:

```
Unexpected end of file on VM.TMP
```

The contents of VM.TMP are written to the file named following the "Run File:" prompt. VM.TMP is a working file only and is deleted at the end of the linking session.

CAUTION

Do not use VM.TMP as a filename for any file. If you have a file named VM.TMP on the default drive and MS-LINK requires the VM.TMP file, MS-LINK deletes the VM.TMP already on disk and creates a new VM.TMP. Thus, the contents of the previous VM.TMP file will be lost.

USING MS-LINK

STARTING MS-LINK

MS-LINK requires two types of input: a command to start MS-LINK and responses to command prompts. In addition, six switches control MS-LINK features. Usually, you type all the commands to MS-LINK on the terminal keyboard. As an option, answers to the command prompts and any switches may be contained in a response file. Command characters can be used to assist you while giving commands to MS-LINK.

You may start MS-LINK in any of three ways. The first method is to type the commands in response to individual prompts. In the second method, you type all commands on the line used to start MS-LINK. To start MS-LINK by the third method, you must create a response file that contains all the necessary commands and tell MS-LINK where that file is when you start MS-LINK.

Method 1	LINK
Method 2	LINK <filenames> [/switches]
Method 3	LINK @<filespec>

Summary of methods to start MS-LINK

Method 1: Prompts

To start MS-LINK with method 1, type:

```
LINK
```

MS-LINK is loaded into memory, and then MS-LINK displays four text prompts that appear one at a time. You answer the prompts to tell MS-LINK to perform specific tasks.

At the end of each line, you may type one or more switches, preceded by the switch character (in this case, a forward slash).

The command prompts are summarized in the following table and are described in more detail in "Command Prompts."

PROMPT	RESPONSES
Object Modules [.OBJ]:	List .OBJ files to be linked. They must be separated by spaces or plus signs (+). If a plus sign is the last character typed, the prompt reappears. There is no default; a response is required.
Run File [Object-file.EXE]:	Give filename for executable object code. The default is first-object-filename.EXE. (You cannot change the output extension.)

List File [Run-file.MAP]:	Give filename for listing. The default is RUN filename.
Libraries [ ]:	List filenames to be searched, separated by spaces or plus signs (+). If a plus sign is the last character typed, the prompt reappears. The default is no search. (Extensions will be changed to .LIB.)

Method 2: Command Line

To start MS-LINK using method 2, type all commands on one line. The entries following LINK are responses to the command prompts. The entry fields for the different prompts must be separated by commas. Use the following syntax:

```
LINK <object-list>,<runfile>,<listfile>,<lib-list>[/switch. . . ]
```

- object-list is a list of object modules, separated by plus signs.
- runfile is the name of the file to receive the executable output.
- listfile is the name of the file to receive the listing.
- lib-list is a list of library modules to be searched.
- /switch refers to optional switches, which may be placed following any of the response entries (just before any of the commas or after the <lib-list>, as shown).

To select the default for a field, simply type a second comma with no spaces between the two commas.

```
LINK
FUN+TEXT+TABLE+CARE/P/M, ,FUNLIST,COBLIB.LIB
```

This command causes MS-LINK to be loaded, followed by the object modules FUN.OBJ, TEXT.OBJ, TABLE.OBJ, and CARE.OBJ. MS-LINK then pauses (as a result of using the /P switch). MS-LINK links the object modules when you press any key, and produces a global symbol map (the /M switch); defaults to FUN.EXE Run file; creates a List file named FUNLIST.MAP; and searches the Library file COBLIB.LIB.

Method 3: Response File

To start MS-LINK with method 3, type:

```
LINK @<filespec>
```

filespec is the name of a response file. A response file contains answers to the MS-LINK prompts (shown in method 1) and may also contain any of the switches. When naming a response file, the use of filename extensions is optional. Method 3 permits the command that starts MS-LINK to be entered from the keyboard or within a batch file without requiring you to take any further action.

To use this option, you must create a response file containing several lines of text, each of which is the response to an MS-LINK prompt. The responses must be in the same order as the MS-LINK prompts discussed in method 1. If desired, a long response to the "Object Modules:" or "Libraries:" prompt may be typed on several lines by using a plus sign (+) to continue the same response onto the next line.

Use switches and command characters in the response file the same way as they are used for responses typed on the terminal keyboard.

When the MS-LINK session begins, each prompt is displayed in order with the responses from the response file. If the response file does not contain answers for all the prompts (in the form of filenames, the semicolon command character, or carriage returns), MS-LINK displays the prompt that does not have a response and then waits for you to type a legal response. When a legal response is typed, MS-LINK continues the link session.

Consider the following example:

```
FUN TEXT TABLE CARE
/PAUSE/MAP
FUNLIST
COBLIB.LIB
```

This response file tells MS-LINK to load the four object modules named FUN, TEXT, TABLE, and CARE. MS-LINK pauses before producing a public symbol map to permit you to swap disks (see discussion under /PAUSE in the Switches section before using this feature). When you press any key, the output files are named FUN.EXE and FUNLIST.MAP. MS-LINK searches the library file COBLIB.LIB and uses the default setting for the switches.

COMMAND CHARACTERS

MS-LINK provides three command characters.

Plus sign

Use the plus sign (+) to separate entries and to extend the current line in response to the "Object Modules:" and "Libraries:" prompts. (A space may be used to separate object modules.) To type a large number of responses (each may be very long), type a plus sign and a <CR> at the end of the line to extend it. If the plus sign and <CR> is the last entry following these two prompts, MS-LINK prompts you for more module names. When the "Object Modules:" or "Libraries:" prompt appears again, continue to type responses. When all the modules to be linked and libraries to be searched are listed be sure the response line ends with a module name and a <CR> and not a plus sign and <CR>.

Example:

```
Object Modules [.OBJ] : FUN TEXT
TABLE CARE+<CR>
Object Modules [.OBJ] :
FOO+FLIPFLOP+JUNQUE+<CR>
Object Modules [.OBJ] :
CORSAIR<CR>
```

Semicolon

To select default responses to the remaining prompts, use a single semicolon (;) followed immediately by ~~a carriage return~~ pressing the <NEWLINE> key at any time after the first prompt (Run File:). This feature saves time and overrides the need to press a series of <~~CR~~ NEWLINE> keys.

NOTE: Once the semicolon has been typed and entered (by pressing the <~~CR~~ NEWLINE> key), you can no longer respond to any of the prompts for that link session. Therefore, do not use the semicolon to skip some prompts. To skip prompts, use the <~~CR~~ NEWLINE> key.

Example:

```
Object Modules
[.OBJ] : FUN TEXT TABLE CARE<~~CR~~ NEWLINE>
Run Module [FUN.EXT] : ;<~~CR~~ NEWLINE>
```

No other prompts appear, and MS-LINK uses the default values (including FUN.MAP for the List file).

<CONTROL-C>

Use the <CONTROL-C> key to abort the link session at any time. If you type an erroneous response, such as the wrong filename or an incorrectly spelled filename, you must press <CONTROL-C> to exit MS-LINK then restart MS-LINK. If you typed the error but did not press the <~~CR~~ NEWLINE> key, you may delete the erroneous characters with the backspace key, but for that line only.

COMMAND PROMPTS

MS-LINK asks you for responses to four text prompts. When you type a response to a prompt and press <~~CR~~ RETURN>, the next prompt appears. When the last prompt is answered, MS-LINK begins linking automatically without further command. When the link session is finished, MS-LINK exits to the operating system. When the operating system prompt appears, MS-LINK has finished successfully. If the link session is unsuccessful, MS-LINK displays the appropriate error message.

MS-LINK prompts the user for the names of Object, Run, and List files, and for Libraries. The prompts are listed in order of appearance. The default response is shown in square brackets ([ ]) following the prompt, for prompts which can default to preset responses. The "Object Modules:" prompt, however, has no preset filename response and requires you to type a filename.

Object Modules [.OBJ] :

Type a list of the object modules to be linked. MS-LINK assumes by default that the filename extension is .OBJ. If an object module has any other filename extension, the extension must be given. Otherwise, the extension may be omitted.

Modules must be separated by plus signs (+).

Remember that MS-LINK loads segments into classes in the order encountered. You can use this information to set the order in which the object modules are read by MS-LINK.

Run File [First-Object-filename.EXE] :

Typing a filename creates a file for storing the Run (executable) file that results from the link session. All Run files receive the filename extension .EXE, even if you specify an extension other than .EXE.

If no response is typed to the "Run File:" prompt, MS-LINK uses the first filename typed in response to the "Object Modules:" prompt as the RUN filename.

Example:

```
Run File [FUN.EXE] : B:PAYROLL/P
```

This response directs MS-LINK to create the Run file PAYROLL.EXE on drive B:. Also, MS-LINK pauses, which allows you to insert a new disk to receive the Run file.

List File [Run-Filename.MAP] :

The List file contains an entry for each segment in the input (object) modules. Each entry shows the addressing in the Run file.

The default response is the Run filename with the default filename extension .MAP.

Libraries [ ] :

The valid responses are up to eight library filenames or simply a carriage return. (A carriage return means no library search.) Library files must have been created by a library utility. MS-LINK assumes by default that the filename extension is .LIB for library files.

Library filenames must be separated by spaces or plus signs (+).

MS-LINK searches library files in the order listed to resolve external references. When it finds the module that defines the external symbol, MS-LINK processes that module as another object module.

If MS-LINK cannot find a library file on the disks in the disk drives, it displays the message:

```
Cannot find library <library-name>
Type new drive letter:
```

Press the letter for the drive designation (for example, B).

MS-LINK SWITCHES

The six MS-LINK switches control various MS-LINK functions. Switches must be typed at the end of a prompt response, regardless of the method used to start MS-LINK. Switches may be grouped at the end of any response, or may be scattered at the end of several. If more than one switch is typed at the end of one response, each switch must be preceded by a forward slash (/).

All switches may be abbreviated. The only restriction is that an abbreviation must be sequential from the first letter through the last typed; no gaps or transpositions are allowed. For example, examine the following lists of valid and invalid abbreviations.

Legal	Illegal
/D	/DSL
/DS	/DAL
/DSA	/DLC
/DSALLOCA	/DSALLOCT

/DSALLOCATE

Using the /DSALLOCATE switch tells MS-LINK to load all data at the high end of the Data Segment. Otherwise, MS-LINK loads all data at the low end of the Data Segment. At runtime, the DS pointer is set to the lowest possible address to allow the entire DS segment to be used. Use of the /DSALLOCATE switch in combination with the default load low (that is, the /HIGH switch is not used) permits the user application to dynamically allocate any available memory below the area specifically allocated within DGroup, yet to remain addressable by the same DS pointer. This dynamic allocation is needed for Pascal and FORTRAN programs.

NOTE: Your application program may dynamically allocate up to 64K bytes (or the actual amount of memory available) less the amount allocated within DGroup.

/HIGH

Use of the /HIGH switch causes MS-LINK to place the Run file as high as possible in memory. Otherwise, MS-LINK places the Run file as low as possible.

CAUTION

Do not use the /HIGH switch with Pascal or FORTRAN programs.

/LINENUMBERS

The /LINENUMBERS switch tells MS-LINK to include in the List file the line numbers and addresses of the source statements in the input modules. Otherwise, line numbers are not included in the List file.

NOTE: Not all compilers produce object modules that contain line number information. In these cases, of course, MS-LINK cannot include line numbers.

/MAP

/MAP directs MS-LINK to list all public (global) symbols defined in the input modules. If /MAP is not given, MS-LINK lists only errors (including undefined globals).

The symbols are listed alphabetically. For each symbol, MS-LINK lists its value and its segment:offset location in the Run file. The symbols are listed at the end of the List file.

/PAUSE

The /PAUSE switch causes MS-LINK to pause in the link session when the switch is encountered. Normally, MS-LINK performs the linking session from beginning to end without stopping. This switch allows the user to swap the disks before MS-LINK outputs the Run (.EXE) file.

When MS-LINK encounters the /PAUSE switch, it displays the message:

```
About to generate .EXE file
Change disks<hit any key>
```

MS-LINK resumes processing when the user presses any key.

CAUTION

Do not remove the disk that receives the List file, or the disk used for the VM.TMP file, if one has been created.

/STACK:<number>

The number entry represents any positive numeric value (in hexadecimal radix) up to 65536 bytes. If a value from 1 to 511 is typed, MS-LINK uses 512. If the /STACK switch is not used for a link session, MS-LINK calculates the necessary stack size automatically.

All compilers and assemblers should provide information in the object modules that allow the linker to compute the required stack size.

At least one object (input) module must contain a stack allocation statement. If not, MS-LINK displays the following error message:

```
WARNING: NO STACK STATEMENT
```

SAMPLE MS-LINK SESSION

This sample shows you the type of information that is displayed during an MS-LINK session.

In response to the MS-DOS prompt, type:

```
LINK
```

The system displays the following messages and prompts (your answers are underlined):

```
Microsoft Object Linker V.2.00
(C) Copyright 1982 by Microsoft Inc.

Object Modules [.OBJ] : NCRIO SYSINIT
Run File [NCRIO.EXE] :
List File [NUL.MAP] : NCRIO /MAP
Libraries [.LIB] : ;
```

Consider how your answers direct MS-LINK and how others affect the output:

- By specifying /MAP, you get both an alphabetic listing and a chronological listing of public symbols.

- By responding PRN to the "List File:" prompt, you can redirect your output to the printer.
- By specifying the /LINE switch, MS-LINK gives you a listing of all line numbers for all modules. (Note that the /LINE switch can generate a large volume of output.)
- By pressing <~~CR~~ NEWLINE> in response to the "Libraries:" prompt, an automatic library search is performed.

Once MS-LINK locates all libraries, the linker map displays a list of segments in the order of their appearance within the load module. The list might look like this:

```
Start       Stop        Length      Name
00000H      009ECH      09EDH       CODE
009F0H      01166H      0777H       SYSINITSEG
```

The information in the Start and Stop columns shows the 20-bit hex address of each segment relative to location zero. Location zero is the beginning of the load module.

The addresses displayed are not the absolute addresses where these segments are loaded. Consult the *PROGRAMMER'S MANUAL* for information on how to determine where relative zero is actually located, and also on how to determine the absolute address of a segment.

Because the /MAP switch was used, MS-LINK displays the public symbols by name and value. For example:

```
ADDRESS             PUBLICS_BY_NAME
009F:0012           BUFFERS
009F:0005           CURRENT_DOS_LOCATION
009F:0011           DEFAULT_DRIVE
009F:000B           DEVICE_LIST
009F:0013           FILES
009F:0009           FINAL_DOS_LOCATION
009F:000F           MEMORY_SIZE
009F:0000           SYSINIT

ADDRESS             PUBLICS BY VALUE
009F:0000           SYSINIT
009F:0005           CURRENT_DOS_LOCATION
009F:0009           FINAL_DOS_LOCATION
```

```
009F:000B          DEVICE_LIST
009F:000F          MEMORY_SIZE
009F:0011          DEFAULT_DRIVE
009F:0012          BUFFERS
009F:0013          FILES
```

ERROR MESSAGES

All errors cause the link session to abort. After you find the cause of the error and correct it, rerun MS-LINK. The following error messages are displayed by MS-LINK; they are mostly self-explanatory.

ATTEMPT TO ACCESS DATA OUTSIDE OF SEGMENT BOUNDS, POSSIBLY BAD OBJECT MODULE

There is probably a bad object file.

BAD NUMERIC PARAMETER

Numeric value is not in digits.

CANNOT OPEN TEMPORARY FILE

MS-LINK is unable to create the file VM.TMP because the disk directory is full. Insert a new disk. Do not remove the disk that will receive the List.MAP file.

~~ERROR~~: DUP RECORD TOO COMPLEX

DUP record in assembly language module is too complex. Simplify DUP record in assembly language program.

~~ERROR:~~ FIXUP OFFSET EXCEEDS FIELD WIDTH

An assembly language instruction refers to an address with a short instruction instead of a long instruction. Edit assembly language source and reassemble.

INPUT FILE READ ERROR

There is probably a bad object file.

INVALID OBJECT MODULE

An object module(s) is incorrectly formed or incomplete (as when assembly is stopped in the middle).

SYMBOL DEFINED MORE THAN ONCE

MS-LINK found two or more modules that define a single symbol name.

PROGRAM SIZE ~~OR NUMBER OF SEGMENTS EXCEEDS~~ CAPACITY OF LINK~~ER~~

The total size may not exceed 384K bytes and the number of segments may not exceed 255.

REQUESTED STACK SIZE EXCEEDS 64K

Specify a size greater than or equal to 64K bytes with the /STACK switch.

SEGMENT SIZE EXCEEDS 64K

64K bytes is the addressing system limit.

SYMBOL TABLE CAPACITY EXCEEDED

Very many and/or very long names were typed, exceeding the limit of approximately 25K bytes.

TOO MANY EXTERNAL SYMBOLS IN ONE MODULE

The limit is 256 external symbols per module.

TOO MANY GROUPS

The limit is 10 groups.

TOO MANY LIBRARIES SPECIFIED

The limit is 8 libraries.

TOO MANY PUBLIC SYMBOLS

The limit is 1024 public symbols.

TOO MANY SEGMENTS OR CLASSES

The limit is 256 (segments and classes taken together).

UNRESOLVED EXTERNALS: <list>

The external symbols listed have no defining module among the modules or library files specified.

VM READ ERROR

This is a disk error; it is not caused by MS-LINK.

WARNING: NO STACK SEGMENT

None of the object modules specified contains a statement allocating stack space, but the user typed the /STACK switch.

WARNING: SEGMENT OF ABSOLUTE OR UNKNOWN TYPE

There is a bad object module or an attempt has been made to link modules that MS-LINK cannot handle (e.g., an absolute module).

~~WRITE ERROR IN TMP FILE~~ OUT OF SPACE ON VM.TMP

No more disk space remains to expand VM.TMP file.

~~WRITE ERROR ON RUN FILE~~ OUT OF SPACE ON RUN FILE

Usually, there is not enough disk space for the Run file.

10.1 → 10.38

MESSAGES.

DUAL-OPERATING SYSTEMS CONSIDERATIONS

You may use your NCR DECISION MATE V with more than one operating system. The dual-processor model, for example, provides processing capabilities of both 8- and 16-bit based applications. If you are planning to use MS-DOS and another operating system, you are responsible for protecting your data files and other disk software. Generally, data protection is simply a matter of keeping the disks properly labelled so that you always process using compatible ones. You must also be sure that only compatible software and data files are on the same disk.

This procedure of 'separation' is also valid for a fixed disk, where one of the logical units may be formatted for use by MS-DOS while the other is reserved for another operating system. When the fixed disk is shared, you should put a label on the unit clearly showing which logical disk is for which operating system.

Properly used, dual-operating sytems significantly increase your processing capabilities. Just remember the guidelines:

- Keep compatible operating system software, application software, and data on the same disks.
- Clearly label all disks, especially the fixed disk where the logical units are not visible.
- Never use a command that could destroy the contents of a disk without first finding out what's on the disk. If you are not sure about the contents, use a command that identifies the contents (MS-DOS CHKDSK for example).
- Finally, be sure you always have backup copies of all important software and data.

DISK ERRORS

If a disk error occurs at any time during a command or program, MS-DOS retries the operation three times. If the operation cannot be completed successfully, MS-DOS returns an error message in the following format:

```
<yyy> ERROR WHILE <I/O action> ON DRIVE x
Abort, Ignore, Retry:_
```

In this message, <yyy> may be one of the following:

- WRITE PROTECT
- NOT READY
- SEEK
- DATA
- SECTOR NOT FOUND
- WRITE FAULT
- DISK

The <I/O-action> may be either of the following:

- READING
- WRITING

The drive <x> indicates the drive in which the error occurred.

MS-DOS waits for you to enter one of the following responses:

A (Abort)
 Terminate the program requesting the disk read or write.

I (Ignore)
 Ignore the bad sector and pretend the error did not occur.

R (Retry)
 Repeat the operation. Use this response when the error is corrected (such as with NOT READY or WRITE PROTECT errors).

Usually, you will want to attempt recovery by entering responses in this order:

R (to try again)
A (to terminate program and try a new disk)

One other error message might be related to faulty disk read or write:

FILE ALLOCATION TABLE BAD FOR DRIVE x

This message means that the copy in memory of one of the allocation tables has pointers to nonexistent blocks. Possibly the disk was incorrectly formatted or not formatted before use. If this error persists, the disk is currently unusable and must be formatted prior to use.

ATTENTION

Be careful when the message

Non-DOS disc error writing drive x
Abort, Retry, Ignore?

is returned. If R were pressed, MS-DOS would consider the non-DOS disk to be double sided, with 8 sectors per track. On the one hand this would probably cause the disc to be destroyed by a write command. On the other hand, pressing R enables you to access at least the first nine sectors of a non-DOS disk with the DEBUG utility

The following error messages may be displayed if an error occurs while the system is being loaded from fixed disk:

Non-System disk..
Insert system diskette and strike any key

This message is returned if the operating system files are not present on the disk in drive x.

Hard disk read error...
Insert system diskette and strike any key

This message means that the firmware could not read a valid boot record from the disk in drive x, even though the operating system files may be present on the disk.

HOW TO OBTAIN AND INSTALL SOFTWARE

MAKING THE RIGHT PURCHASE

Your NCR DECISION MATE V with MS-DOS is highly flexible, accommodating nearly any application and language software as long as it is MS-DOS compatible. Think of the software you can use on your computer in three categories:

- Software that is distributed by NCR.
- Software that has been run by NCR on the DECISION MATE V and is available to any MS-DOS user.
- Software that has *not* been tested by NCR, but is available to any MS-DOS user.

Those packages in the first group can be obtained directly from NCR by contacting your NCR representative or licensed dealer. The software in the other categories can be purchased from an NCR licensed dealer or any reputable software house. You will want to be sure, however, that the software will run on your computer. Knowing what questions to ask and working with a knowledgeable dealer will help you to make the right decision.

1. Ask if the software is on an NCR MS-DOS format disk.
2. If the software is not available on an NCR MS-DOS format disk, ask if it is available on an IBM-PC format disk. (A 5 1/4-inch disk with either 160/180/320/360KB capacity.) Any of these disks can be used on your computer.
3. Most application packages and some language software require an installation program to interface with the specific hardware. Ask if the software is already installed to be used on an NCR DECISION MATE V, or if the package includes an installation (or install) program so that you can tailor the software yourself. If the software is installed for an MS-DOS ANSI device driver or a Lear Siegler terminal, it can be installed simply on your computer (see next section).

INSTALLING THE SOFTWARE

To fully use every feature on NCR DECISION MATE V, each application must be tailored to, or installed on, your computer. The installation procedure is different for each application, so every "off the shelf" package, such as MULTIPLAN™, contains its own program for accomplishing this task. The documentation that accompanies the application disk you purchase describes the way this program is run. We recommend that before you customize your application, you make a copy of the purchased disk and use the copy as your work disk. (See the FORMAT and DISKCOPY commands in chapter 5.) Save the original disk in a safe place and use it only to make copies.

To begin, most installation programs display a list of computer terminals. If this list includes NCR DECISION MATE V, select it; otherwise, choose an MS-DOS ANSI device driver. the Lear Siegler ADM-31, or the Lear Siegler ADM-3A terminal. Refer to table 1 for the functions that are active if you install your application as for an MS-DOS ANSI device driver, or table 2 if you install your application as on a Lear Siegler terminal.

If none of the above terminals are listed, you must describe the terminal characteristics of NCR DECISION MATE V to the application's installation program. Use the tables as a guide if you must enter these codes to install an application package. (All of these control characters/sequences are for use by applications, not for entry from the keyboard.)

Other tables include information you may need, depending on the particular application. Table 3 contains miscellaneous information and table 4 gives notes, frequencies, and cycles for programming music applications. For further reference, conversion and translation tables are included at the end of this section.

FUNCTIONAL CHARACTERISTICS INFORMATION

The following functions must be preceded by an ESCAPE <ESC> character (hex value 1B) and a left bracket <[> character (hex value 5B).

MS-DOS ANSI DRIVER TABLE (TABLE 1)

Function	ASCII string	hex string
Cursor Position	(row#);(col#)H or (row#);(col#)f	(row#)3B(col#)48 (row#)3B(col#)66
Cursor Left	(# of cols)D	(# of cols)44
Cursor Right	(# of cols)C	(# of cols)43
Cursor Down	(# of rows)B	(# of rows)42
Cursor Up	(# of rows)A	(# of rows)41
Device Status Report	6n	366E
Cursor Position Report (returned after 6n function)	(row#);(col#)R	(row#)3B(col#)52
Save Cursor Position	s	73
Restore Cursor Position	u	75
Erase Screen	2J	324A
Erase to End of Line	K	4B
Graphic Attributes:		
All Attributes Off	0m	306D
Half Intensity Off	1m	316D
Blink On	5m	356D
Reverse Video On	7m	376D
Half Intensity On	8m	386D
Define Function Key	0;(fk#);"string"p	303B(fkH)3B22string2270
Disable Function Key Ext.	0;0p	303B3070
Enable Function Key Ext.	0;99p	303B393970

The information in parentheses represents data you supply. You do not enter the parentheses.

Not in new manual

For example, to assign CHKDSK <~~CR~~> NEWLINE to function key 18, enter

<ESC>[0;18;"CHKDSK";13p

where 13 stands for <CR>, hex value 0D.
NEWLINE

TERMINAL FUNCTION CODES (TABLE 2)

Function	ASCII string	hex string	Terminal*
Position Cursor	<ESC> =	1B3D	A,B
Row + Offset	(row# +32)	(row#+20hex)	A,B
Column + Offset	(col#+32)	(col#+20hex)	A,B
Cursor Left	^H	08	A,B
Cursor Right	^L	0C	A,B
Cursor Down	^J	0A	A,B
Cursor Up	^K	0B	A,B
Clear Screen & Cursor Home	^Z	1A	A,B
Clear to End of Line	^W	17	
Carriage Return	^M	0D	A,B
Escape	<ESC>	1B	A,B
Bell	^G	07	A,B
Home Cursor	^~	1E	
The following functions must be preceded by an ESCAPE <ESC> character, hex value 1B.			
Clear to End of Line	T or t	54 or 74	A
Clear to End of Screen	Y or y	59 or 79	A
Clear Screen and Cusor Home	: or *	3A or 2A	
Half Intensity On	)	29	A
Half Intensity Off	(	28	A
Reverse Video On	G4	4734	A
Blinking On	G2	4732	A

The information in parentheses represents data you supply. You do not enter the parentheses.

* A = Lear Siegler ADM-31; B = Lear Siegler ADM-3A

cont.

TERMINAL FUNCTION CODES (TABLE 2) . . . cont.

Function	ASCII string	hex string	Terminal*
Rev. Video & Blinking Off	G0	4730	A
Insert Line	E	45	A
Insert Character	Q	51	A
Delete Line	R	52	A
Delete Character	W	57	A
Play Music **	M	4D	

The information in parentheses represents data you supply. You do not enter the parentheses.

* A = Lear Siegler ADM-31; B = Lear Siegler ADM-3A

** Music can be programmed on the NCR DECISION MATE V. On receiving the string <ESC> M, the CRT driver accepts the next two numbers as frequency and tone length, respectively. Refer to Table 4 at the end of this section for the corresponding note, frequency, and number of cycles.

MISCELLANEOUS INFORMATION – Table 3

	ANSI	Lear. Sigler ADM 31/A
Number of ~~rows~~ lines*	24 (1-24) 5 5	25 (0-24)
Number of columns	80 (1-80)	80 (0-79)
Cursor origin	1/1	0/0
Input/Output technique	MS-DOS calls and commands (e.g. TYPE)	– ditto –
Cursor on/off	not active	not active
Keyboard click on/off	not active	not active

* Line 25 does not scroll if it is used for steady displays.

MUSIC CODES — Table 4

Note	Frequency			Cycles
	dec.	hex	key	
Pause	32	20	Space	—
A	33	21	!	110
A#	34	22	"	116.5
B	35	23	#	123.5
C	36	24	$	131
C#	37	25	%	138.6
D	38	26	&	146.8
D#	39	27	'	155.8
E	40	28	(	164.8
F	41	29	)	174.6
F#	42	2A	*	185
G	43	2B	+	196
G#	44	2C	,	208
A	45	2D	—	220
A#	46	2E	.	233
B	47	2F	/	246.9
C (Middle C)	48	30	0	261.6
C#	49	31	1	277.4
D	50	32	2	293.7
D#	51	33	3	311
E	52	34	4	329.6
F	53	35	5	349.2
F#	54	36	6	370
G	55	37	7	392
G#	56	38	8	415
A	57	39	9	440
A#	58	3A	:	465
B	59	3B	;	493.9

cont.

MUSIC CODES — Table 4 (cont.)

Note	Frequency			Cycles
	dec.	hex	key	
C	60	3C	<	523.2
C#	61	3D	=	553
D	62	3E	>	587.3
D#	63	3F	?	622
E	64	40	@	659.3
F	65	41	A	698.5
F#	66	42	B	740
G	67	43	C	784
G#	68	44	D	830
A	69	45	E	880
A#	70	46	F	932
B	71	47	G	987.8
C	72	48	H	1046.5
C#	73	49	1	1108.7
D	74	4A	J	1174.7

C-8

GRAPHIC ATTRIBUTES.

TRANSLATION/CONVERSION INFORMATION

Because of the special language requirements of different countries, examples in this manual of characters, either input or displayed, may not match the characters on the keyboard. The following table shows substitute characters which produce the same hexadecimal code and satisfy the requirements of the operating system. Also, for users who need to refer to the complete hexadecimal chart, the ASCII code chart with the USASI code set is given after the keyboard table.

Country	Hex Codes and Corresponding Characters											
	23	24	40	5B	5C	5D	5E	60	7B	7C	7D	7E
US-English	#	$	@	[	\	]	^	`	{	|	}	~
UK-English	£	$	@	[	\	]	↑	`	{	|	}	~
French	£	$	à	°	ç	§	^	`	é	ù	è	¨
German	#	$	§	Ä	Ö	Ü	^	`	ä	ö	ü	ß
Swedish/Finnish	#	¤	@	Ä	Ö	Å	^	`	ä	ö	å	¨
Danish/Norwegian	£	$	@	Æ	Ø	Å	^	`	æ	ø	å	¨
Spanish	£	$	@	¡	Ñ	¿	^	`	{	ñ	}	~
Italian	£	$	§	°	ç	é	^	ù	à	ò	è	ì
Swiss	£	$	ç	à	é	è	^	`	ä	ö	ü	¨
Canadian	#	$	@	[	\	]	^	`	£	|	¢	¨
Canadian (bilingual)	#	$	@	[	ç	]	^	`	é	è	¢	¨
South African	#	$	@	Ê	'N	Ë	^	`	ê	'n	ë	¨
Portuguese	£	$	@	Ā	Ō	Ç	`	`	ā	ō	ç	~
Yugoslavian	#	$	Đ	Ć	Č	Š	Ž	đ	ć	č	š	ž

Special country keyboard definitions

ASCII CODE CHART

Binary b_4-b_1 ►		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
b_8-b_5 ▼	HEX	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**
0000	0	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
0001	1	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
0010	2	⧄	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
0011	3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0100	4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0101	5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	∧	_
0110	6	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0111	7	p	q	r	s	t	u	v	w	x	y	z	{	¦	}	~	DEL

LEGEND: (For Control Characters in USASI Code Set)

NUL	Null
SOH	Start of Heading
STX	Start of Text
ETX	End of Text
EOT	End of Transmission
ENQ	Enquiry
ACK	Acknowledge
BEL	Bell (Audible or Attention Signal)
BS	Backspace
HT	Horizontal Tabulation (Punched Card Strip)
LF	Line Feed
VT	Vertical Tabulation
FF	Form Feed
CR	Carriage Return
SO	Shift Out
SI	Shift In
DLE	Data Link Escape
DC1	Device Control 1
DC2	Device Control 2
DC3	Device Control 3
DC4	Device Control 4
NAK	Negative Acknowledge
SYN	Synchronous DLE (Sync Code)
ETB	End of Transmission Block
CAN	Cancel (Void Data)
EM	End of Media
SUB	Substitute
ESC	Escape
FS	File Separation (End of File)
GS	Group Separate
RS	Record Separator (End of Record)
US	Unit Separator (End of Field)
DEL	Delete

D.1 — D.5

ADVANCED CONFIGURATON

INDEX

INDEX

+ (command character) (MS-LINK) 9-10
.BAT 4-4
.EXE (MS-LINK) 9-6, 9-11
.EXE file (MS-LINK) 9-1
.LIB (MS-LINK) 9-5
.MAP (MS-LINK) 9-6, 9-12
.OBJ (MS-LINK) 9-5, 9-11
; (command character) (MS-LINK) 9-10
<COPY1> 7-4
<COPYALL> 7-6
<COPYUP> 7-5
<EXIT> 7-12
<INSERT> 7-10
<NEWLINE> 7-13
<SKIP1> 7-7
<SKIPUP> 7-8
<VOID> 7-9

Abort B1
Addresses (MS-LINK) 9-3
ASCII code chart C-9
AUTOEXEC.BAT 2-15, 4-6, 5-21
Automatic Program Execution 2-15

Backing-up your disks 2-14
BACKUP (command) 5-4
Backup (disk) 3-6
Basic 4-7
Batch processing 4-4
Binary files (FC) 8-1
BREAK 5-5
Buffer space (FC) 8-1

Change directory 3-13
CHDIR 5-6
CHKDSK 2-11, 2-17, 5-7
CIPHER 4-12, 5-10
Class name (MS-LINK) 9-3

Classes (MS-LINK) 9-3, 9-11
CLS 5-12
Command Characters
+ (MS-LINK) 9-10
; (MS-LINK) 9-10
Command Characters (MS-LINK) 9-10
Command processor 2-1, 4-1
Command Prompts
Libraries (MS-LINK) 9-8, 9-12
List File (MS-LINK) 9-8, 9-12
Object Modules (MS-LINK) 9-7, 9-11
Run File (MS-LINK) 9-7, 9-11
COMMAND.COM 2-1
Commands 5-1
Compiler 9-1
Concatenation 5-18
CONFIG 5-13
Control character functions 6-5
Control character functions table 6-6
CONTROL-C 4-3
CONTROL-S 4-4
CONTROL-Z 4-7, 4-9
COPY command 3-4, 5-17
Copying your master software disk 2-2
CTTY 5-20
Current date 2-12
Current time 2-12

Data Error B-1
Date 2-13
Default drive 2-14
Defining your disk configuration (table) 2-8
Defining a serial printer 2-12
Defining your fixed disks 2-7
Delimiters 4-3
Difference reporting (FC) 8-4
DIR 2-16, 3-11
Directories
Hierarchical 3-7
Directory 2-16, 3-6
Making 3-12
Removing 3-13
Working 3-7

Disk Errors
- Abort B-1
- Data B-1
- Disk B-1
- File Allocation Table Bad For Drive x B-2
- Ignore B-1
- Not Ready B-1
- Retry B-1
- Sector Not Found B-1
- Seek B-1
- Write Fault B-1
- Write Protect B-1

DISKCOPY 2-14, 5-25

Disks
- Backup 2-14

Drive designation 2-7, 3-2, 4-2

Drive designations (MS-LINK) 9-5

Dual-Operating Systems Considerations A-1

Dummy parameters 4-9

ECHO 5-27

EDLIN Commands
- Append 7-18
- Copy 7-19
- Delete 7-21
- Edit 7-23
- End 7-25
- Insert 7-26
- List 7-29
- Move 7-32
- Page 7-33
- Quit 7-34
- Replace 7-35
- Search 7-38
- Transfer 7-41
- Write 7-42

EDLIN Errors
- Cannot edit .BAK file – rename file 7-43
- Disk full 7-44
- Entry error 7-43
- Filename must be specified 7-45
- File not found 7-45
- Incorrect DOS version 7-44
- Insufficient memory 7-45

Invalid drive name or file 7-44
Invalid parameter 7-45
Line too long 7-44
Must specify destination number 7-45
No room in directory for file 7-43
Not enough room to merge the entire file 7-46
EDLIN Command Options 7-16
EDLIN Format Conventions 7-14
EXIT 5-28
External commands 3-10
External reference (MS-LINK) 9-1

FC 8-1
File Allocation Table 2-16
File Allocation Table Bad For Drive x B-2
File specification 3-2
File system 3-8
Filename 3-1, 4-2
Filename extension 3-1, 4-2
.COM 4-2
.EXE 4-2
Filename extensions -- default
.EXE (MS-LINK) 9-6
.MAP (MS-LINK) 9-6
.OBJ (MS-LINK) 9-5
Filenames
Illegal 3-4
Files 2-15
Binary (FC) 8-1
Comparing 8-1
Naming 3-1
Source (FC) 8-1
Files that MS-LINK uses (MS-LINK) 9-5
Filespec 4-3
Filters 4-12
FIND 4-13, 5-29
FOR 5-31
FORMAT 2-12, 5-33
Functional characteristics information C-3

GOTO 5-35
Groups (MS-LINK) 9-4

Hidden files 2-17
Hierarchical directory 3-7
How to obtain and install software C-1

IF 5-36
Ignore B-1
Illegal filenames 3-4
Input 4-11
    Redirection 4-11
Installing the software C-2
Internal commands 3-11
Intraline Commands
    Replace mode 7-12
    Copy multiple characters 7-5
    Copy template 7-6
    Enter insert mode 7-10
    Exit insert mode 7-12
    New template 7-13
    Quit 7-9
    Quit input 7-9
    Skip multiple characters 7-8
    Skip one character 7-7

LINK (FC) 8-1
LINKER PROGRAM (MS-LINK) 9-1
Loading MS-DOS 2-1
LOCATE 5-38

Miscellaneous information table C-5
MKDIR 3-12, 5-41
MORE 4-12, 5-42
MS-LINK Switches 9-13
MS-DOS ANSI driver table C-3
MS-DOS Command Summary 5-2
Music codes table C-6, C-7

Not Ready Error B-1

Object modules 9-1
Options 4-2
Output 4-11
    Redirection 4-11
Output Redirection (FC) 8-5

Overview
  MS-LINK 9-1

Parameters 4-9
Parameters
  Replaceable 4-9
Parent directory 3-9, 5-6
PATH 5-43
Pathing 3-9
Pathname 3-9
PAUSE 4-5
Pipes 4-11
Piping 4-12
PRINT 5-45
Programmable function keys 2-15
PROMPT 5-48

RDCPM 5-50
RECOVER 5-52
Redirecting output (FC) 8-5
Relative address (FC) 8-2
Relocatable load module (MS-LINK) 9-1
REM 4-5, 5-53
RENAME (synonym for REN) 5-54
Replaceable parameters 4-9
Reserved filenames
  AUX 3-4
  CON 3-4
  LST 3-4
  NUL 3-4
  PRN 3-4
Response file (MS-LINK) 9-8
Retry B-1
RMDIR 3-13, 5-56
Root directory 2-16

Sector Not Found Error B-1
Seek Error B-1
Segment name (MS-LINK) 9-3
Segments (MS-LINK) 9-3, 9-11
SET 5-57
SHIFT 4-9, 5-58
Shorthand notation 3-9
SORT 4-12, 5-59

Source code 9-1
Source drives 4-4
Source files (FC) 8-1
Special characters 3-2
Special country keyboard definitions C-8
Special editing functions table 6-2
Special editing keys 6-1, 7-3
Starting MS-LINK 9-6
Subdirectory 3-11
Switches
  FC
    /# (FC) 8-3
    /B (FC) 8-2
    /C (FC) 8-3
    /W (FC) 8-3
  MS-LINK
    – DSALLOCATE (MS-LINK) 9-13
    – HIGH (MS-LINK) 9-13
    – LINENUMBERS (MS-LINK) 9-14
    – MAP (MS-LINK) 9-14
    – PAUSE (MS-LINK) 9-14
    – STACK (MS-LINK) 9-15
SYS 5-61
System set up 2-1

Terminal function codes table C-4, C-5
Time 2-12
TIME (command) 5-63
Translation/conversion information C-8
TYPE 5-65

VER 5-66
VERIFY 5-67
VM.TMP (MS-LINK) 9-6
VOL 5-68

Wild cards 3-2, 4-4
  The * wild card 3-3
  The ? wild card 3-32
Working directory 2-16, 3-7, 3-9, 5-6
  Displaying 3-11
Write Fault Error B-1
Write Protect Error B-1