DOS 3.10

Reference

Programming Family

Upgrade Kit

IBM

Personal
Computer
Software

6248221

UPGRADE TERMS AND CONDITIONS

This package contains materials to upgrade a prior version of the program to the current version. The contents of this package can only be used for this purpose. You are authorised to upgrade the materials of the prior version of the program with the materials contained in this package. The upgraded program will be the sole version of the program which you are entitled to use, and is subject to the IBM Conditions of Use of the previous version of the program.

Following its upgrade, all remaining materials of the previous version of the program, including all copies and modifications made, must be destroyed.

A copy of the IBM Conditions of Use is set out below for your convenience.

IBM CONDITIONS OF USE

Any other IBM Terms and Conditions and/or IBM Program License Agreement, which may appear printed inside the package, is inapplicable and should be ignored.

Copyright and Other Rights

IBM programs contain material in which IBM, and in many cases IBM's suppliers, retain proprietary rights. IBM wants these programs to be fully usable by you for the purpose for which they are supplied, that is, in connection with a computer. No infringement of the rights of IBM or of IBM's suppliers will occur provided that the following conditions are observed with respect to each program.

1. The program is used only on a single machine at any one time;
2. The program is copied into machine-readable or printed form for backup or modification purposes only in support of use on a single machine. However, certain diskettes marked "Copy Protected" may include mechanisms to limit or inhibit copying of the program;
3. The program is modified or merged into another program only for use on the single machine. Any portion so merged continues to be subject to these conditions;
4. The copyright notice is reproduced and included in any copy or modifications made of the program and in any portion merged into other programs; and
5. If the program package is transferred to another party, all copies and modifications made of the program must be transferred or destroyed. You do not retain any right with respect to the transferred package. The other party agrees to observe all of the IBM Conditions of Use.

Any other act involving reproduction or use of, or other dealing in the program is prohibited.

You are reminded that it may be necessary to obtain local and United States licenses to export or re-export this package.

No statements contained in this package shall affect the statutory rights of consumers.

UPGRADES

Upgrades to programs may be made available by IBM. Such upgrades are subject to these Conditions of Use together with such further conditions as may be applicable.

DOS 3.10

Reference

Programming Family

Upgrade Kit

Personal
Computer
Software

First edition (February 1985)

The following paragraph does not apply to the United Kingdom or any country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This publication could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time.

It is possible that this publication may contain reference to, or information about, IBM products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that IBM intends to announce such IBM products, programming, or services in your country.

Products are not stocked at the address below. Requests for copies of this publication and for technical information about IBM Personal Computer products should be made to your authorized IBM Personal Computer dealer, IBM Product Center, or your IBM Marketing Representative.

The following paragraph applies only to the United States and Puerto Rico: A Reader's Comment Form is provided at the back of this publication. If the form has been removed, address comments to: IBM Corporation, Personal Computer, P.O. Box 1328-C, Boca Raton, Florida 33432. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligations whatever.

About This Book

Read This First

This manual explains how to use the IBM Personal Computer Disk Operating System Version 3.10. Information is given on how to:

- Refer to DOS files
- Prepare your fixed disk for DOS
- Use directories to organize your disks
- Use redirection, piping, and filters with standard input and standard output
- Use DOS commands
- Create and edit files using EDLIN
- Link programs with the LINKER.

First Time Users

Before using your DOS diskette for the first time, read the sections "Using Your DOS Diskettes" and "About Diskette Drives and Diskettes" in Chapter 1 of this book.

Experienced Users

Experienced programmers can use this book together with the IBM *DOS Technical Reference* to develop application programs. Information is provided on:

- Installable device drivers
- Cursor control sequences that control cursor positioning and reassign keyboard keys
- File Management
- DOS disk allocation
- DOS interrupts and function calls
- DOS control blocks and work area
- Executing commands from within an application
- Fixed disk information
- EXE file structure and loading
- DOS memory management
- DEBUG

Terms Used

The terms "diskette," "fixed disk," and "disk" are used throughout this book. Where "diskette" is used, it applies only to diskette drives and diskettes. Where "fixed disk" is used, it applies only to the IBM non–removable fixed disk drive. Where "disk" is used, it applies to both fixed disks and diskettes.

The terms "source" and "target" are used to describe diskettes and drives. The source refers to the original or first diskette or drive, and the target refers to the second or new diskette or drive.

The terms "local" and "remote" describe the location of a disk, directory or printer relative to your computer. A local disk, directory, or printer is on *your* computer. A remote disk, directory, or printer is on a *network* computer.

Using Applications with DOS 3.10

To use applications with DOS Version 3.10, you need to follow the setup procedures given in the IBM *Application Setup Guide*. Do not follow the setup procedures given in your application manual.

How This Book is Organized

This book has 10 chapters and one appendix.

Chapter 1 is an introduction to DOS Version 3.10. It contains information about diskette types and compatibility. The new features of DOS Version 3.10 are also discussed.

Chapter 2 describes the DOS file specification. Information on valid filename characters and global filename characters is described.

Chapter 3 gives detailed instructions on how to prepare your fixed disk for use by DOS.

Chapter 4 describes system configuration. It contains the commands that you can include in a CONFIG.SYS file to configure your system.

Chapter 5 describes the use of tree–structured directories.

Chapter 6 describes how to use redirection, piping, and DOS filters with standard input and standard output.

Chapter 7 gives a detailed description of DOS commands, listed in alphabetic order. The descriptions define the purpose, format, and type (internal or external) of each command. Examples are given where appropriate.

Chapter 8 describes how to use EDLIN, the line editor to create, alter, and display source language files and text files.

Chapter 9 describes the use of the Linker program to link programs together before execution.

Chapter 10 tells you how to use the DEBUG program.

Appendix A explains general and device error messages and gives an appropriate response for each message.

Contents

Chapter 1. Introduction **1-1**
About Your DOS Books and Diskettes 1-3
Using Your DOS Diskettes 1-3
DOS 3.10 Features 1-4
New Commands 1-4
Enhanced Commands 1-5
About Diskette Drives and Diskettes 1-6
Types of Diskette Drives 1-6
Types of Diskettes 1-6
Diskette and Drive Compatibility 1-7
About Messages 1-8

Chapter 2. File Specification **2-1**
Introduction 2-3
The File Specification 2-3
DOS Device Names 2-5
Global Filename Characters 2-7
The ? Character 2-7
The * Character 2-8
Examples of Ways to Use ? and * 2-9

Chapter 3. Preparing Your Fixed Disk **3-1**
Introduction 3-3
Replacing a Previous Version of DOS 3-3
Referring to Disk Drives 3-4
Dividing Your Fixed Disk 3-4
Using FDISK 3-6
Starting FDISK 3-7
Creating a DOS Partition (Choice 1) 3-9
Using an Entire Fixed Disk for DOS . 3-10
Using Part of a Fixed Disk for DOS .. 3-11
Changing the Active Partition (Choice 2) . 3-13
Deleting a DOS Partition (Choice 3) 3-15
Displaying Partition Information (Choice 4) 3-17
Select the Next Fixed Disk Drive (Choice 5) 3-18
Formatting Your DOS Partition 3-18
Copying DOS to Your DOS Partition 3-22

Starting DOS from Your Fixed Disk 3-23

Chapter 4. Configuring Your System 4-1
Introduction 4-3
What is a Configuration File ? 4-4
Creating a CONFIG.SYS File 4-4
Configuration Commands 4-5
BREAK Command 4-6
BUFFERS Command 4-7
COUNTRY Command 4-11
DEVICE Command 4-13
Loading Standard Device Drivers 4-13
Installing Your Own Device Driver ... 4-13
ANSI.SYS 4-14
VDISK.SYS 4-14
FCBS (File Control Block) Command 4-19
With File Sharing 4-20
Without File Sharing 4-20
FILES Command 4-22
LASTDRIVE Command 4-24
SHELL Command 4-25

Chapter 5. Using Tree-Structured Directories 5-1
Introduction 5-3
Why Use Directories? 5-3
How Directories Are Organized 5-4
Directory Entries 5-5
Accessing Your Subdirectories 5-6
The Current Directory 5-6
Changing Directories with CHDIR 5-7
Specifying a Path to a File 5-7
Using the PATH Command 5-8
Using PATH in a Batch File 5-9
Using Directory Commands 5-10
Making a Subdirectory 5-11
Removing a Subdirectory 5-13
Displaying and Changing the Current Directory 5-14
Displaying the Directory Structure ... 5-15
Where DOS Looks for Commands and Batch Files 5-15

Chapter 6. Standard Input and Standard Output ... 6-1

Introduction 6-3
Redirection of Standard Input and Output Devices 6-3
Piping of Standard Input and Output 6-6
DOS Filters 6-7

Chapter 7. DOS Commands 7-1
Introduction 7-5
DOS Commands and the Network 7-7
Types of DOS Commands 7-8
Entering a DOS Command 7-9
Information Common to All DOS Commands 7-10
DOS Commands 7-13
ASSIGN (Drive) Command 7-14
ATTRIB (Attribute) Command 7-17
BACKUP Command 7-19
Batch File Commands 7-24
Creating a Batch File 7-26
Executing a Batch file 7-27
The AUTOEXEC.BAT File 7-27
Creating an AUTOEXEC.BAT file .. 7-28
Creating a Batch File with Replaceable Parameters 7-28
Executing a Batch File with Replaceable Parameters 7-29
ECHO Subcommand 7-30
FOR Subcommand 7-32
GOTO Subcommand 7-33
IF Subcommand 7-34
PAUSE Subcommand 7-38
REM (Remark) Subcommand 7-39
SHIFT Subcommand 7-40
BREAK (Control Break) Command 7-43
CHDIR (Change Directory) Command ... 7-45
CHKDSK (Check Disk) Command 7-48
CLS (Clear Screen) Command 7-52
COMMAND (Secondary Command Processor) Command 7-53
COMP (Compare Files) Command 7-55
COPY Command 7-60
CTTY (Change Console) Command 7-72
DATE Command 7-74

DEL (Delete) Command 7-77
DIR (Directory) Command 7-79
DISKCOMP (Compare Diskettes Only) Command 7-84
DISKCOMP Compatibility 7-88
DISKCOPY (Copy Diskettes Only) Command 7-90
DISKCOPY Compatibility 7-94
ERASE Command 7-96
EXE2BIN Command 7-98
FDISK Command 7-101
FIND Filter Command 7-102
FORMAT Command 7-105
FORMAT Compatibility 7-110
GRAFTABL (Load Graphics Table) Command 7-113
GRAPHICS (Screen Print) Command ... 7-115
JOIN Command 7-118
Why Use JOIN? 7-122
KEYBxx (Load Keyboard) Command ... 7-123
LABEL (Volume Label) Command 7-127
MKDIR (Make Directory) Command ... 7-130
MODE Command 7-132
MORE Filter Command 7-139
PATH (Set Search Directory) Command . 7-140
PRINT Command 7-143
PROMPT (Set System Prompt) Command 7-149
RECOVER Command 7-153
RENAME (or REN) Command 7-156
RESTORE Command 7-157
RMDIR (Remove Directory) Command . 7-161
SELECT Command 7-162
SET (Set Environment) Command 7-164
SHARE Command 7-167
SORT Filter Command 7-169
SUBST(Substitute) Command 7-172
SYS (System) Command 7-178
TIME Command 7-180
TREE Command 7-182
TYPE Command 7-185
VER (Version) Command 7-186
VERIFY Command 7-187
VOL (Volume) Command 7-188

Chapter 8. The Line Editor (EDLIN) 8-1
Introduction 8-3
How to Start the EDLIN Program 8-5
Editing an Existing File 8-5
Editing a New File 8-6
The EDLIN Command Parameters 8-7
The EDLIN Commands 8-9
Information Common to All EDLIN Commands 8-9
A (Append Lines) Command 8-11
C (Copy Lines) Command 8-12
D (Delete Lines) Command 8-13
Edit Line Command 8-16
E (End Edit) Command 8-18
I (Insert Lines) Command 8-19
L (List Lines) Command 8-22
M (Move Lines) Command 8-25
P (Page) Command 8-26
Q (Quit Edit) Command 8-27
R (Replace Text) Command 8-28
S (Search Text) Command 8-31
T (Transfer Lines) Command 8-34
W (Write Lines) Command 8-35

Chapter 9. The Linker (LINK) Program 9-1
Introduction 9-3
Files 9-4
Input Files 9-4
Output Files 9-5
VM.TMP (Temporary File) 9-5
Definitions 9-6
Segment 9-6
Group 9-7
Class 9-7
Command Prompts 9-7
Detailed Descriptions of the Command Prompts 9-9
Object Modules [.OBJ]: 9-9
Run File [filename.EXE]: 9-10
List File [NUL.MAP]: 9-10
Libraries [.LIB]: 9-12
Linker Parameters 9-14
How to Start the Linker Program 9-17

Before You Begin 9-17
Option 1 - Console Responses 9-17
Option 2 - Command Line 9-18
Option 3 - Automatic Responses 9-21
Example Linker Session 9-23
How to Determine the Absolute Address of a Segment 9-26
Messages 9-27

Chapter 10. DEBUG Program 10-1
Introduction 10-3
How to Start the DEBUG Program 10-4
The DEBUG Command Parameters 10-6
The DEBUG Commands 10-13
Information Common to All DEBUG Commands 10-13
A (Assemble) Command 10-15
C (Compare) Command 10-19
D (Dump) Command 10-20
E (Enter) Command 10-23
F (Fill) Command 10-26
G (Go) Command 10-27
H (Hexarithmetic) Command 10-30
I (Input) Command 10-31
L (Load) Command 10-32
M (Move) Command 10-35
N (Name) Command 10-36
O (Output) Command 10-38
P (Proceed) Command 10-39
Q (Quit) Command 10-40
R (Register) Command 10-41
S (Search) Command 10-46
T (Trace) Command 10-47
U (Unassemble) Command 10-49
W (Write) Command 10-52

Appendix A. Messages A-3
Introduction A-3
Responses A-3
Device Error Messages A-4
Other Messages A-11

Index Index-1

Chapter 1. Introduction

Contents

About Your DOS Books and Diskettes **1-3**

Using Your DOS Diskettes **1-3**

DOS 3.10 Features **1-4**
New Commands 1-4
Enhanced Commands 1-5

About Diskette Drives and Diskettes **1-6**
Types of Diskette Drives 1-6
Types of Diskettes 1-6
Diskette and Drive Compatibility 1-7

About Messages **1-8**

Notes:

About Your DOS Books and Diskettes

DOS Version 3.10 comes with three books:

- Application Setup Guide
- DOS User's Guide
- DOS Reference

DOS also comes on two diskettes. Both diskettes are in the back of the *DOS Reference* in a plastic pocket. The first diskette, labeled "DOS," contains the DOS programs and commands. In this book, the first diskette is referred to as the DOS diskette. The second diskette, labeled "DOS Supplemental Programs," contains the LINK Utility, EXE2BIN, a sample device driver listing, and BASIC sample programs. In this book, the second diskette is referred to as the Supplemental diskette.

Using Your DOS Diskettes

DOS Version 3.10 is different from previous versions of DOS. When you load DOS for the first time, you need to select the keyboard layout you want to use and the country whose date and time format you want to use. You can select the keyboard layout and the date and time format by using:

- The SELECT command described in Chapter 7
- The COUNTRY configuration command (Chapter 4) and the KEYB*xx* command (Chapter 7)

DOS 3.10 Features

This section describes the new and changed features of DOS Version 3.10. These are the new and changed features since DOS Version 3.00.

New Commands

The following new commands have been added to DOS Version 3.10. Refer to the command description in Chapter 7 of this book for further details and examples.

JOIN

Using JOIN, you can connect a drive to a directory on another drive to create a new directory structure. You can then access files on more than one drive using only one drive specifier.

SUBST

Use SUBST to substitute a *drive letter* for a drive or a directory. Then you can access files on that drive or directory by referring to the *drive letter*. SUBST is a way to reassign drives that is preferred over the ASSIGN command.

If you use applications that do not recognize paths or tree-structured directories, SUBST allows you to substitute a drive letter for a path.

Network Support

DOS Version 3.10 supports the IBM PC Network. You can use most DOS commands on a network disk, directory or printer.

Enhanced Commands

The following commands have been enhanced for DOS Version 3.10. Please refer to the command descriptions in Chapter 7 of this book for further details.

LABEL

The LABEL command now prompts you before deleting a volume label.

TREE

The TREE command now displays all files under the root directory when you specify the /F parameter.

Enhanced Linker

A new parameter (/X) is available with the Linker so that you can specify an increased number of segments.

About Diskette Drives and Diskettes

Types of Diskette Drives

Your IBM Personal Computer can have the following types of diskette drives:

- Single sided (160KB/180KB)
- Double sided (320KB/360KB)
- High capacity (1.2MB)

Types of Diskettes

You can use the following types of diskettes to read and write information:

- Single sided (160KB/180KB)
- Double sided (320KB/360KB)
- High capacity (1.2MB)

A Single-sided diskette contains 40 tracks, 8/9 sectors per track, and holds up to 160K/180K bytes of information (K equals 1024).

A Double-sided diskette contains 40 tracks, 8/9 sectors per track, and holds up to 320K/360K bytes of information (K equals 1024).

A High-capacity diskette is a double-sided diskette that contains 80 tracks, 15 sectors per track, and holds up to 1.2M bytes of information (M equals 1,048,576).

Diskette and Drive Compatibility

Some combinations for reading and writing between different diskette and drive types are not allowed. The following sections describe which diskette and drive combinations *are* allowed.

Single-Sided Drives

You can read and write to:

- Single-sided diskettes

Double-Sided Drives

You can read and write to:

- Single-sided diskettes
- Double-sided diskettes

High-Capacity Drives

You can read and write to:

- Single-sided diskettes*
- Double-sided diskettes*
- High-capacity diskettes

***Important:** If you write on any of these diskette types using a high-capacity drive, you may not be able to read the diskettes in a single- or double-sided drive.

You need to consider diskette and drive compatibility when you use DOS commands that read and write to diskettes. For example, the FORMAT command contains a section called "FORMAT Compatibility." Read the sections about compatibility before using the command.

About Messages

You may get messages on your screen when you use DOS commands. If you get a message and need help, refer to "Messages" in Appendix A for the *explanation* of the message and the *action* you should take.

Chapter 2. File Specification

Contents

Introduction **2-3**

The File Specification **2-3**

DOS Device Names **2-5**

Global Filename Characters **2-7**
The ? Character 2-7
The * Character 2-8
Examples of Ways to Use ? and * 2-9
Example 1 2-9
Example 2 2-9
Example 3 2-9

Notes:

Introduction

This chapter describes the file specification. It includes a description of the drive specifier, the filename, and the extension. It also contains information on the acceptable filename characters, reserved device names, and global filename characters.

The File Specification

The file specification tells DOS where to search for the specified file. A *filespec* consists of three parts: the drive specifier, the filename, and the filename extension. The following table describes each part of a filespec.

Parameter	Definition
d:	Denotes the drive specifier. It specifies the drive that contains the file you want to refer to. To specify the drive, type the drive letter followed by a colon. For example, A: is the drive specifier that represents drive A. If you omit the drive specifier, DOS assumes the file is located in the *default* drive.

Parameter	Definition

filename	Denotes the filename. The filename consists of one to eight characters. When you type a filename, DOS checks for invalid characters. The following characters are *invalid* in filenames.
	
	`.  "  /  \  [  ]  :`
	
	`¦  <  >  +  =  ;  ,`
	
	`ASCII characters less than 20H`
	
	Any other characters are valid.

ext	Denotes the filename extension. The filename extension consists of a period followed by one to three characters. The following characters are *invalid* in a filename extension:
	
	`.  "  /  \  [  ]  :`
	
	`¦  <  >  +  =  ;  ,`
	
	`ASCII characters less than 20H`
	
	Any other characters are valid.

DOS Device Names

Certain names have special meaning to DOS. They are called *DOS device names* Since they are reserved, do not name files with a DOS device name. DOS reserves the following names:

Reserved Name	Device
CON	Console keyboard/screen. If used as an input device, you can press the F6 key; then press Enter to generate an end–of–file indication, which ends CON as an input device.
AUX or COM1	First Serial/Parallel Adapter port.
COM2	Second Serial/Parallel Adapter port.
LPT1 or PRN	First Parallel Printer (as an output device only).
LPT2 and LPT3	Second Parallel Printer Third Parallel Printer
NUL	Nonexistent (dummy) device for testing applications. As an input device, immediate end–of–file is generated. As an output device, the write operations are simulated, but no data is actually written.

Notes:

1. Since these are reserved names, you cannot create files with these names.

2. When using a device name, you should assure that the device actually exists. Using the name of a nonexistent device can cause unpredictable errors in DOS operation.

3. The reserved device names can be used in place of a filename in DOS commands.

4. The colon after the device name is optional. For example, you can type:

```
CON
```

or

```
CON:
```

Global Filename Characters

Two special characters ? and * can be used within a filename and its extension. These special characters give you greater flexibility with the DOS commands.

The ? Character

A ? in a filename or in a filename extension indicates that any character can occupy that position. For example,

```
A>dir  ab?de.xyz
```

lists all directory entries on the default drive with filenames that have five characters, begin with AB, have any next character, are followed by DE, and have an extension of XYZ.

Here are some examples of the files that might be listed by the DIR command:

```
ABCED   XYZ
ABIDE   XYZ
ABODE   XYZ
```

The * Character

An * in a filename or in a filename extension indicates that any character can occupy that position and all the remaining positions in the filename or extension. For example,

```
A>dir  ab*.xyz
```

lists all directory entries on the default drive with filenames that begin with AB and have an extension of XYZ. In this case, the filenames may be from 2 to 8 characters in length.

Here are some example files that might be listed by the DIR command:

```
ABCDE   XYZ
ABC357  XYZ
ABIDE   XYZ
ABIIOU  XYZ
ABO$$$  XYZ
AB      XYZ
```

Examples of Ways to Use ? and *

Example 1

To list the directory entries for all files named INPUT on drive A (regardless of their filename extension), type:

```
A>dir a:input.???
  or
A>dir a:input.*
```


Example 2

To list the directory entries for all files in the current directory on drive A (regardless of their filenames) with a filename extension of XYZ, type:

```
A>dir a:????????.xyz
  or
A>dir a:*.xyz
```

Example 3

To list the directory entries for all files on drive A with filenames beginning with ABC and extensions beginning with E, type:

```
A>dir a:abc?????.e??
  or
A>dir a:abc*.e*
```

Notes:

Chapter 3. Preparing Your Fixed Disk

Contents

Introduction 3-3

Replacing a Previous Version of DOS 3-3

Referring to Disk Drives 3-4

Dividing Your Fixed Disk 3-4

Using FDISK 3-6
Starting FDISK 3-7

Creating a DOS Partition (Choice 1) 3-9
Using an Entire Fixed Disk for DOS 3-10
Using Part of a Fixed Disk for DOS 3-11

Changing the Active Partition (Choice 2) 3-13

Deleting a DOS Partition (Choice 3) 3-15

Displaying Partition Information (Choice 4) 3-17

Select the Next Fixed Disk Drive (Choice 5) 3-18

Formatting Your DOS Partition 3-18

Copying DOS to Your DOS Partition 3-22

Starting DOS from Your Fixed Disk 3-23

Notes:

Introduction

If your IBM Personal Computer has a fixed disk, you must prepare the fixed disk before DOS can use it.

This chapter tells you how to:

- Refer to your computer's disk drives
- Use FDISK to prepare your fixed disk for DOS
- Replace a previous version of DOS on your fixed disk with DOS 3.10
- Format your fixed disk
- Copy DOS to your fixed disk
- Start DOS from your fixed disk
- Do other tasks with FDISK

Replacing a Previous Version of DOS

If you already have a version of DOS on your fixed disk prior to DOS 3.10, follow these steps to replace it with DOS 3.10:

> **Note:** If you have *not* prepared your fixed disk with a previous version of DOS, skip this section. Go to "Referring to Disk Drives."

1. Insert your DOS 3.10 diskette in the appropriate diskette drive.
2. Press Ctrl-Alt-Del.
3. At the DOS prompt, type:

```
sys c:      (Press Enter)
```

If you have more than one fixed disk, be sure to type the drive letter for the fixed disk on which you want to place DOS 3.10.

4. Copy the rest of the DOS files to your fixed disk. Type:

```
copy *.* c:      (Press Enter)
```

5. Repeat step 4 with your Supplemental DOS diskette.

Now your fixed disk is ready to be used with DOS 3.10. For information about starting DOS see the section in this chapter entitled, "Starting DOS from Your Fixed Disk."

Referring to Disk Drives

DOS assigns drive letters to the diskette drives and to the fixed disk on your computer.

For example, if your computer has one diskette drive and one fixed disk drive, DOS assigns the drive letters A and B to the diskette drive and the letter C to the fixed disk. If your computer has two diskette drives and two fixed disk drives, the diskette drives are A and B, and the fixed disks are C and D. You can install a third fixed disk, to which DOS assigns the drive letter E.

Dividing Your Fixed Disk

You can divide a fixed disk into one, two, three or four sections called *partitions*. Partitions separate fixed disk space into individual areas. To prepare your fixed disk

for DOS, you create a partition for DOS called a *DOS partition*. To do this, you will use the fixed disk setup program supplied by DOS called *FDISK*.

If you are using more than one operating system, you need a separate partition for each operating system. If DOS is your only operating system, you only need one partition.

Using FDISK

With the FDISK program, you can create a DOS partition and do other fixed disk tasks.

FDISK allows you to:

- Create a DOS partition
- Change an active partition
- Delete a DOS partition
- Display partition data
- Select the next fixed disk drive for partitioning if you have more than one.

FDISK has menus and screens that guide you through the tasks. To find out how to use FDISK, continue with "Starting FDISK."

Starting FDISK

To start FDISK, follow these steps:

1. With your DOS diskette in drive A, at the DOS prompt A>, type:

   ```
   A>fdisk
   ```

2. Press Enter.

   The FDISK Options menu appears. Choose what you want to do. If you make a mistake or change your mind, press Esc to return to the FDISK Options menu. After making a selection, you see a series of screens.

```
IBM Personal Computer
Fixed Disk Setup Program Version 3.10
(C)Copyright IBM Corp. 1983,1984

FDISK Options

Current Fixed Disk Drive: 1

Choose one of the following:

    1.  Create DOS Partition
    2.  Change Active Partition
    3.  Delete DOS Partition
    4.  Display Partition Data
    5.  Select Next Fixed Disk Drive

 Enter choice: [1]


 Press ESC to return to DOS
```

Each screen displays a default choice value. If you want the default value as your choice, press Enter. If you want a different value than the default value, type the value you want, then press Enter.

Choices 1 through 4 appear if your computer has one fixed disk attached. Choice 5 appears if your computer has more than one fixed disk attached. When you first see this menu, the Enter choice field will have a 1 in it.

3. Do one of the following:

   - If you want to increase a DOS partition, go to "Creating a DOS Partition."
   - If you want to change an active partition, go to "Changing the Active Partition."
   - If you want to delete a DOS partition, go to "Deleting a DOS partition."
   - If you want to display information about the partitions on your fixed disk, go to "Displaying Partition Information."
   - If you want to use FDISK on another fixed disk drive, go to "Select the Next Fixed Disk Drive."

Creating a DOS Partition (Choice 1)

Do one of the following:

- If DOS is the only operating system that you intend to use with your fixed disk, follow the instructions in the section "Using an Entire Fixed Disk for DOS."
- If you intend to use part of the fixed disk with another operating system, follow the instructions for "Using Part of a Fixed Disk for DOS."

Choice 1 (Create a DOS partition) is the default value for the FDISK Options menu. To create a DOS partition, press Enter. The following screen appears:

```
Create DOS Partition

Current Fixed Disk Drive: 1


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

Using an Entire Fixed Disk for DOS

The following message appears if you told the computer you wanted to use an entire fixed disk for DOS:

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

1. Insert your DOS diskette in drive A and press any key.

   DOS restarts and prompts you to enter the date and time.

2. Enter the date.

3. Enter the time.

   When you see the DOS prompt, the DOS partition has been created. Before you can use the DOS partition, you must format the disk.

4. Go to "Formatting Your DOS Partition" later in this chapter to continue.

Using Part of a Fixed Disk for DOS

The following message appears if you told the computer you wanted to use only part of a fixed disk for DOS:

```
Total fixed disk space is xxxx cylinders
Maximum available space is xxxx
cylinders at xxxx.
```

"Total fixed disk space" tells you how many cylinders you have on your entire disk. "Maximum available space" tells you how many cylinders are left for you to partition. The following prompt appears:

```
Enter partition size.....: [xxxx]
```

1. Do one of the following:

   - If you want your DOS partition to use all the available space, press Enter.
   - If you do not want to use all the available space for DOS, type the size you want to use (in cylinders).

   The following prompt appears:

```
Enter starting cylinder number..:[xxxx]
```

2. The starting cylinder number you see depends on the partition size you specified. If you want the DOS partition to be located there, press Enter. If not, type the starting cylinder number you want and press Enter.

3. The DOS partition is created, but before you can use it, you must:

   - Change the active partition (FDISK Options menu Choice 2).

   - Format your DOS partition (refer to "Formatting Your DOS Partition" later in this chapter).

4. If there are bad spots at the start of the specified partition, FDISK automatically adjusts the start of the partition to the first good area on the disk that it finds. This may decrease the size of the partition.

Changing the Active Partition (Choice 2)

Your fixed disk can have many partitions, but only one partition can be *active*. The active partition is the one that gets control of your computer when you turn it on or reset the computer. Only one partition can be active at a time. If DOS is the active partition, DOS controls the computer when it is turned on or reset.

If you type **2** on the FDISK Options menu, a screen like the one shown here appears. This sample screen shows the partition status of a fixed disk that has three partitions.

```
Change Active Partition

Current Fixed Disk Drive: 1

Partition Status  Type   Start End Size
    1        A     DOS    000  149  150
    2        N   non-DOS  150  304  155
    3        N     xxx    xxx  xxx  xxx


Total disk space is  305  cylinders

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

Partition	The number assigned to the partition.
Status	The status of the partition—A for active, N for non–active.
Type	The kind of partition—DOS or non–DOS.
Start/End	The starting and ending cylinder numbers of the partition.
Size	The number of cylinders the partition uses.

Do the following:

1. Type the number of the partition that you want to be active. (If you want your DOS partition to be active, type the number for the DOS partition.)
2. Press Enter. The partition you selected then becomes the active partition.
3. Go to "Starting DOS from Your Fixed Disk."

Deleting a DOS Partition (Choice 3)

You can delete the fixed disk partition you use for DOS. When you do, remember any data in that partition is also deleted and cannot be recovered. The boundaries for that partition are also removed.

Note: If you delete the DOS partition but want to continue using DOS, insert a DOS diskette in drive A and start DOS again.

1. If you typed **3** on the FDISK Options menu, you will see a screen like the one shown here.

```
Delete DOS Partition

Current Fixed Disk Drive: 1

Partition Status  Type   Start End Size
    1        N     DOS    000  304  305

Total fixed disk space is 305 cylinders


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

When the screen first appears, you will see an **N** (no) next to the prompt "Do you wish to continue?"

2. Do one of the following:

   - If you do not want to delete the DOS partition, press Enter to return to the FDISK Options menu.

   - If you want to delete the DOS partition, type **y** (yes) and press Enter.

     You need to start another operating system from the fixed disk or restart DOS from a diskette to proceed.

Displaying Partition Information (Choice 4)

You can display information that tells how a fixed disk on your computer is partitioned.

If you typed **4** on the Fixed Disk Options menu, you will see a screen like the one shown here.

```
Display Partition Information

Current Fixed Disk Drive: 1

Partition Status  Type    Start End Size
    1        A     DOS     000  199  200
    2        N   non-DOS   200  304  105


Total fixed disk space is 305 cylinders
```


You see the:

- Partition number
- Partition status
- Partition type
- Starting cylinder number
- Ending cylinder number
- Size in cylinders

Select the Next Fixed Disk Drive (Choice 5)

If your computer has more than one fixed disk attached, you can use the same FDISK menu options for each one. If you type **5** on the FDISK Options menu, you can prepare another fixed disk.

Formatting Your DOS Partition

Formatting your DOS partition is the final step in preparing it for DOS use. You should format a DOS partition the first time you create it or if you want to erase all of the data contained in the DOS partition.

> **Note:** If your DOS partition is larger than 10M bytes and you format it using DOS Version 3.10, you cannot use versions of DOS prior to DOS 3.00 to access your fixed disk.

You use the DOS FORMAT command to format your DOS partition.

Follow these steps to format your DOS partition on drive C:

1. With your DOS diskette in drive A, at the DOS prompt A>, type:

   A>format c:/s/v

2. Press Enter.

   The following appears on the screen:

```
Warning, All data on Non-removable
Disk drive C: Will be lost!
Proceed with Format (Y/N)?
```

Warning: FORMAT destroys any data contained in your DOS partition. Be sure you wish to format the contents of your DOS partition disk before you continue.

3. Type **y**, then press Enter. The "In-Use" light on your fixed disk comes on and the message **Formatting...** is displayed. It takes several minutes to format your fixed disk so do not worry.

   Next you will see:

```
Formatting... Format complete
System transferred
Volume label (11 characters, ENTER for none)?
```

   The message "System transferred" is displayed because you specified the /S parameter. It means that a copy of the operating system files IBMBIO.COM, IBMDOS.COM, and COMMAND.COM has been placed in the DOS partition.

4. You are asked to enter a volume label consisting of up to 11 characters (for example MYFIXEDDISK).

   > **Note:** Characters that are valid for filenames are also valid for volume labels. Refer to Chapter 2 "The File Specification" for information on filename characters.

   The volume label is used to identify the fixed disk.

5. Do one of the following:

   - If you want to label your fixed disk, type a name and press Enter.
   - If you do not want to label your fixed disk, press Enter.

Note: You can add or change the volume label by using the LABEL command. Refer to Chapter 7 "DOS Commands" for more information.

After you have answered the volume label prompt, disk space statistics are displayed along with the DOS prompt A>. Now, your DOS partition is formatted and ready to be used by DOS.

6. If you want to copy the remaining DOS files from the DOS diskette, refer to the next section, "Copying DOS to Your DOS Partition." Otherwise, skip the next section and go to "Starting DOS from Your Fixed Disk" at the end of this chapter.

Copying DOS to Your DOS Partition

You can also copy the external DOS commands from the DOS diskette and the Supplemental diskette into the DOS partition. Use the COPY command:

COPY *d*:*.* *d*:

Note:

For more information on the COPY command, refer to Chapter 7 "DOS Commands."

Follow these steps to copy DOS to your DOS partition on drive C:

1. With your DOS diskette in drive A, at the DOS prompt A>, type:

   ```
   A>copy a:*.* c:
   ```

2. Press Enter.

3. To copy the commands from the Supplemental diskette, remove the DOS diskette from drive A and insert your Supplemental diskette. Type:

   ```
   A>copy a:*.* c:
   ```

4. Press Enter.

5. Go to "Starting DOS from Your Fixed Disk."

Now you can run all of the DOS commands from the fixed disk. You should store your DOS diskettes in a safe place.

Starting DOS from Your Fixed Disk

Starting DOS from your fixed disk means that when you switch on the power or do a system reset (restart DOS), DOS is loaded from the DOS Partition. DOS can only be loaded from the first fixed disk drive attached to your computer.

1. To start DOS from your fixed disk, make sure that:

   - You leave the drive door for Drive A open when you switch on the power or perform a system reset.
   - You have placed a copy of the operating system files in the DOS partition.
   - You have partitioned your entire disk for DOS *or* you have made the DOS partition the active partition.

2. Do one of the following:

   - If your computer is off, switch on your computer.
   - If your computer is on, press Ctrl–Alt–Del (system reset).

   After you have set up your fixed disk, you may want to refer to Section 5, "Using Directories," for information on how to organize your files.

Notes:

Chapter 4. Configuring Your System

Contents

Introduction **4-3**
What is a Configuration File ? 4-4

Creating a CONFIG.SYS File **4-4**

Configuration Commands **4-5**

BREAK Command **4-6**

BUFFERS Command **4-7**
What Is a Buffer ? 4-7
Read/Write Requests 4-8
Random/Sequential Applications 4-8
Size of Your Computer 4-9

COUNTRY Command **4-11**

DEVICE Command **4-13**
Loading Standard Device Drivers 4-13
Installing Your Own Device Driver 4-13
ANSI.SYS 4-14
VDISK.SYS 4-14
Installing VDISK 4-15

FCBS (File Control Block) Command **4-19**
With File Sharing 4-20
Without File Sharing 4-20

FILES Command **4-22**
Accessing a File 4-22
Number of Files Opened 4-22

LASTDRIVE Command **4-24**

SHELL Command **4-25**

Notes:

Introduction

This chapter describes how to configure your system by creating a file named CONFIG.SYS. A list of the configuration commands is given with a description of the purpose and format of the command. You can use configuration commands to:

- Set extended checking of Ctrl–Break (BREAK)
- Specify the number of disk buffers (BUFFERS)
- Specify the country whose date and time format you want to use (COUNTRY)
- Install device drivers (DEVICE)
- Specify the number of files that can be opened by file control blocks (FCBS)
- Specify the number of files that can be open at one time (FILES)
- Set the maximum drive letter that you may access (LASTDRIVE)
- Specify the name of a top–level command processor (SHELL)

What is a Configuration File ?

A configuration file contains commands that are used to configure your system. Each time you start DOS, DOS searches the root directory of the drive it was started from for the file named CONFIG.SYS. If the file CONFIG.SYS is found, DOS reads the file and interprets the commands within the file. If the file CONFIG.SYS is not found, DOS assigns default values for the configuration commands.

Create a configuration file named CONFIG.SYS to change the default values of the configuration commands. If you add or change any of the configuration file commands, the changes are not in effect until the *next* time you start DOS.

Creating a CONFIG.SYS File

Create a CONFIG.SYS file using an editor (such as EDLIN) or the DOS COPY command. This section describes how to create a CONFIG.SYS file using the COPY command directly from the standard input device.

> **Note:** If you have used the SELECT command, a CONFIG.SYS file has already been created that includes the the COUNTRY configuration command. You should use EDLIN or an editor to add other configuration commands.

Follow these steps to create a CONFIG.SYS file using the COPY command.

1. At the DOS prompt A>, type:

```
A>copy con config.sys
```

2. Press Enter.

3. Type the configuration commands you want in the CONFIG.SYS file. Press Enter after you type each command.

4. When you have finished typing the commands, press the F6 key and then Enter. This ends the COPY CON command and saves the file.

Now a file named CONFIG.SYS has been created. But the values for the commands are not in effect until the *next* time you start DOS.

Configuration Commands

The following section describes the commands that you can include in a CONFIG.SYS file.

BREAK Command

Purpose: Allows you to instruct DOS to check for control break whenever a program requests DOS to perform any functions.

Format: BREAK = [ON | OFF]

Remarks: The default value is set at BREAK = **OFF**. This means that DOS will only check for Ctrl–Break being entered during:

- Standard output operations
- Standard input operations
- Standard print operations
- Standard auxiliary operations

If you want DOS to check for Ctrl–Break whenever it is requested, set BREAK = **ON**. This allows you to *break–out* of a program that produces few or no standard device operations (such as a compiler). For instance, if a program is being compiled, it is important to have a way to stop compilation if an error or loop is encountered.

BUFFERS Command

Purpose: Allows you to determine the number of disk buffers that DOS will allocate in memory when it starts.

Format: BUFFERS = *x*

Remarks: The *x* is a number between 1 and 99. This is the number of disk buffers that DOS allocates in memory when it starts. The default value is 2, (3 for the IBM Personal Computer AT) and this value remains in effect until DOS is restarted with a different value specified in the configuration file.

What Is a Buffer ?

A disk buffer is a block of memory that DOS uses to hold data being read from, or written to a disk. For example, if an application reads a 128–byte record from a file, DOS reads the entire sector into one of its buffers, locates the correct 128–byte record in the buffer, and moves the record from the buffer into the application's area of memory. It then marks that buffer as having been used recently. On the next request to transfer data, DOS attempts to use a different buffer. In this way, all of the buffers eventually contain the most recently–used data. The more buffers DOS has, the more data is in memory.

Read/Write Requests

Before DOS reads or writes a record that is not an exact multiple of the sector size, it checks to see if the sector containing that record is already in a buffer. If not, it must read the sector as described above. But if the data is already in a buffer, DOS can simply transfer the record to the application's area without reading the sector from the disk. This saves time, both in reading and writing records, because DOS must first read a sector before it can insert a record that your application is attempting to write.

Random/Sequential Applications

For applications that read and write records in a random fashion (such as many BASIC and data base applications), the likelihood of finding the correct record already in a buffer increases if DOS has more buffers to work with. This can increase performance of those applications by speeding up the access time.

For applications doing sequential reads and writes, however (read an entire file, write an entire file), there is little advantage to having a large number of buffers allocated.

Because all applications are different, there is no specific number of buffers that will serve all applications equally well. If your applications do little random reading and writing of records, the system default of two buffers should be sufficient.

However, if you use data base type applications, or run programs that perform a lot of random reads and writes of records, you will want to increase the number of DOS buffers. The "best" number of buffers for your particular application can only be determined by using different values until the desired performance is achieved. For most data base applications, a value between 10 and 20 buffers usually provides the best results. For subdirectories, between 10 and 25 buffers usually provides desirable performance.

Beyond that point, the system may appear to start running slower. With a very large number of buffers, it can take DOS longer to search all the buffers for a record than it would take to read the record from disk.

Size of Your Computer

The final consideration in determining the number of buffers to allocate is the memory size of your computer. Since each additional buffer increases the resident size of DOS by 528 bytes, the amount of memory available to the application is reduced by that amount. Additional buffers may actually cause some applications to run more slowly because there is less memory available for the application to keep data. This could result in more frequent reads and writes than would otherwise be necessary.

BUFFERS Command

In summary, the optimum number of buffers must be determined by *you*, based on:

- The types of applications most often used
- The memory size of your computer
- Your analysis of system performance when using your applications with different numbers of buffers allocated
- For computers with fixed disks, a minimum of BUFFERS=3 is recommended

COUNTRY Command

Purpose: Use to specify the date and time format for a given country. Other information, such as the currency symbol and the decimal separator for a particular country, are also set using the COUNTRY command.

> **Note:** COUNTRY does not translate the text of DOS messages for the country you specify.

Format: COUNTRY = *xxx*

Remarks: *xxx* is the 3–digit international country code for the telephone system. The default value is the U.S. country code of 001. The following countries are supported for DOS Version 3.10. If your country is not supported, choose the most familiar country supported.

COUNTRY Command

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

Example: To specify the date and time format for the United States, include the following command in the CONFIG.SYS file.

```
country=001
```

The next time you start DOS, the date format is *mm-dd-yy*, the time format is *hh:mm:ss*, the decimal separator is a period (.), and the currency symbol is $.

DEVICE Command

Purpose: Allows you to specify the name of a file containing a device driver.

Format: DEVICE = [*d*:][*path*]*filename*[*.ext*]

Remarks: During startup, DOS loads the file into memory and gives it control as described in "Installation of Device Drivers" in Chapter 2 of *DOS Technical Reference*. Please refer to that section for technical information about installable device drivers.

Loading Standard Device Drivers

The standard device drivers loaded by DOS support the standard input, standard output, standard printer, diskette, and fixed disk devices. A clock driver is also loaded (see Chapter 2 of the *DOS Technical Reference*). You don't need to specify any DEVICE= commands for DOS to support these devices.

Installing Your Own Device Driver

For systems programmers and application developers—if you have written device drivers that you want DOS to load when it starts, include a DEVICE= command in the CONFIG.SYS file for each driver to be loaded.

You must install the device drivers that come with DOS (ANSI.SYS, VDISK.SYS) before you can use them.

DEVICE Command

ANSI.SYS

To use the "Extended Screen and Keyboard Control" features described in the *DOS Technical Reference*, create the file CONFIG.SYS on the disk you will be starting DOS from. The file should contain the command DEVICE=ANSI.SYS. This command causes DOS to replace the standard input and standard output support with the extended functions.

VDISK.SYS

The VDISK.SYS file on your DOS diskette is a device driver that simulates a disk drive by using a portion of your computer's memory as the storage medium. These simulated disks are called *virtual disks*. The following characteristics apply to virtual disks:

- Virtual disks are fast, since they operate at the speed of your computer's memory.
- You can install more than one virtual disk; each is referred to by a drive letter, in the same way you refer to disk drives. For example, if your computer has two diskette drives and no fixed disks, the diskette drives are referred to as drives A and B, the first virtual disk is referred to as C, the second as D, and so on.
- If you have an IBM Personal Computer AT with extended memory installed, (starting at the 1MB boundary), you can use the extended memory as one or more virtual disks. Otherwise, virtual disks are located in low memory.

- For each virtual disk, you can specify the amount of memory to be used (the "disk size"), the sector size, and the number of directory entries it is to contain.

- A volume label is created on each virtual disk to assist in its identification.

- Each virtual disk created increases the resident size of DOS by 720 bytes for the VDISK.SYS device driver plus the size of the virtual disk buffer you specify (if the driver is installed in low memory).

- The contents of a virtual disk are lost if you restart the system or if power is lost.

- Virtual disks cannot be formatted. Each VDISK is installed in formatted form.

Installing VDISK

To install the VDISK device driver, include this statement in the CONFIG.SYS file:

device=[*d*:][*path*]vdisk.sys [*bbb*][*sss*][*ddd*][/E[:*m*]]

[*d*:][*path*] is the drive and directory path containing the VDISK.SYS file.

The *bbb* is the virtual disk size in K bytes, and is specified as a decimal value. The default value is 64K bytes. The range of values is between 1 and the amount of available memory on your computer. VDISK may adjust the amount of memory actually used for the virtual disk as follows:

- If there is less than 64K of available memory at the time VDISK is being installed, VDISK issues an error message and does not install the virtual disk.

- If the size you specify is less than 1K byte or greater than the total amount of memory on your computer, VDISK uses the default value of 64K.

- If the specified size leaves less than 64K of available memory, VDISK adjusts the virtual disk size downward (VDISK always leaves a minimum of 64K of available memory after the installation of the virtual disk).

The *sss* is the sector size in bytes. Allowable sizes are 128, 256, or 512. If omitted, or if an incorrect value is typed, VDISK uses the default value of 128. If you use your virtual disk to hold relatively small files, you may wish to use a smaller sector size to minimize wasted space.

The *ddd* is the number of directory entries (number of files) that the virtual disk can contain (one directory entry is required for each file). The default value is 64. The range of values is between 2 and 512. VDISK may adjust the value you entered as follows:

- The value is adjusted upwards to the nearest sector size boundary. For example, if you specify a value of 10, and your sector size is 128, VDISK generates 12 directory entries (12 entries at 32 bytes each to round up to a multiple of the sector size).

- If the virtual disk size specified above is too small to hold the file allocation table, the

Note: If some interrupts are being lost, isolate the problem in the following manner. Install VDISK in non-extended memory. If the problem is resolved, adjust *m* (in extended memory) until no interrupts are lost. If an *m* of 1 and sss of 128 dos not improve the situation, then VDISK cannot be used in extended memory in that environment. If installing VDISK in non-extended memory does not resolve the problem, other areas should be investigated as the cause of the problem.

The next time you start DOS, VDISK displays the the following message:

VDISK Version 2.0 virtual disk x

This message is displayed as an informative message to tell you that VDISK is attempting to install a virtual disk. It also tells you the drive letter *x* that is assigned to the virtual disk.

directory, and 2 additional sectors, the directory size is adjusted downward by 1 sector at a time until these conditions are met. If the directory size reaches 1 sector and the conditions still cannot be met, VDISK issues an error message and the virtual disk is not installed.

- VDISK uses one of the directory entries to hold the volume label.

/E is a parameter that tells VDISK to use extended memory. Extended memory is memory at or above 1M bytes. The virtual disk buffer will be in extended memory and the device driver will be installed in low memory. More than one virtual disk can be installed in extended memory by including more than one DEVICE=VDISK.SYS command in the CONFIG.SYS file. The first device driver is installed at the 1 M-byte boundary, the second would immediately follow the first, and so on. This parameter is only valid for an IBM Personal Computer AT with extended memory. If you specify this parameter for a computer that does not have extended memory, an error message is displayed and the virtual disk is not installed.

The following example installs a 160K-byte virtual disk with 512-byte sectors and 64 directory entries.

```
device=vdisk.sys 160 512 64
```

The *m* is the maximum number of sectors (of size sss) of data VDISK transfers at a time. The default for *m* is 8. The possible values for *m* are 1, 2, 3, 4, 5, 6, 7, and 8. When VDISK is operating in extended memory, interrupt servicing is suspended during data transfers. If there are very frequent interrupts occurring during these periods (as in high speed communications), some interrupts can be lost.

FCBS (File Control Block) Command

Purpose: Allows you to specify the number of file control blocks (FCBs) that can be concurrently open by DOS.

Format: FCBS = *m*,*n*

Remarks: The *m* specifies the total number of files opened by FCBs that can be open at one time. The default value is 4. The range of values for *m* is from 1 to 255.

The *n* specifies the number of files opened by FCBs that cannot be closed automatically by DOS if a program tries to have more than *m* files opened by FCBs at one time. The first *n* files opened by FCBs are protected from being closed. The default value is 0. The range of values for *n* is between 0 and 255.

Some application programs use file control blocks to create, open, delete, read and write to files. DOS keeps track of the least recently used FCB. If the program tries to open more than *m* FCB files, the action DOS takes depends on whether file sharing is loaded.

Notes:

1. The value of *m* must be greater than or equal to the value of *n*.

2. If a program receives critical errors due to FCB files being closed by DOS, increasing the value of *m* may prevent these errors from occurring.

3. If a program uses two or more different FCBs to refer to the same file, DOS only counts this as one FCB used.

4. If you specify the FCBS command in your configuration file, the resident size of DOS is increased.

With File Sharing

If file sharing is loaded, and a program tries to open more than *m* files, DOS closes the least recently used FCB and opens the new file. Note that the first *n* files are not included in the list of files that DOS tracks for the least recently used FCB. Therefore, they are protected from being closed this way. If a program tries to read or write to a file that has been closed because it is the least recently used FCB, DOS issues the following message:

```
FCB unavailable
Abort, Retry, Ignore?
```

Note: If you set *m* equal to *n*, it does not allow any files to be closed by DOS if a program tries to open more than *m* files. If a program tries to open more than *m* files, DOS does not open the new file.

Without File Sharing

If sharing is not loaded, the number of files that can be open at the same time is not limited. The FCBS command is only applicable when file sharing is loaded.

FCBS (File Control Block) Command

Example: To set the total number of FCB files that can be open at one time to 3, and the number protected from being closed to 1, include this command in the CONFIG.SYS file.

```
fcbs=3,1
```

FILES Command

Purpose: Allows you to specify the maximum number of file handles that can be open concurrently.

Format: FILES = *x*

Remarks: The *x* can be a number between 8 and 255. The default value is FILES=8.

Accessing a File

All file accesses (reads, writes, close) can be performed by telling DOS which handle to use. When an application opens a file in this manner, DOS constructs a control block in its own memory on behalf of the application, in an area that was set aside when DOS started. The size of this area (and consequently, the maximum number of file handles that can be concurrently open), depends on the value specified in the FILES= command.

The default value is FILES=8; that is, no more than 8 file handles can be open at the same time. There is no effect on the number of file that can be concurrently open using the traditional (OPEN FCB) functions. This default value is sufficient for the majority of operating environments. However, if applications are installed that result in error messages indicating an insufficient number of handles, the FILES= command should be used to provide DOS with additional handles.

Number of Files Opened

The value specified in FILES= becomes the new maximum number of file handles that DOS allows to be concurrently open.

Note that this value is the maximum number of handles allowed for the entire system. This includes handles in use by currently running foreground tasks (programs such as COMP and CHKDSK) and background tasks like PRINT and the network. The maximum number of file handles that a process can have open is 20 (this number includes the 3 used by DOS for the 5 predefined handles for standard input, standard output, standard error, auxiliary, and standard printer). So the limit on handles for the system is specified by FILES=, and the limit on handles for each process is is 20.

If you specify FILES= in your configuration file, the size of the resident portion of DOS increases by 48 bytes for each additional file above the default value of 8. Consequently, the memory available to the application is reduced by the same amount. See function calls 3CH through 46H in Chapter 5 of the *DOS Technical Reference* for descriptions of the new file–handling functions.

LASTDRIVE Command

Purpose: Sets the maximum number of drives that you may access.

Format: LASTDRIVE = *x*

Remarks: The *x* can be any alphabetic character A through Z. It represents the last valid drive letter that DOS may accept. The default value is LASTDRIVE = E.

The minimum number you can set LASTDRIVE equal to is the number of drives you have installed on your computer. If *x* is less than the number of physical drives on the your computer, this command is ignored in the configuration file.

Example: To set the number of drives equal to 16, include this command in the CONFIG.SYS file:

```
lastdrive=p
```

SHELL Command

Purpose: Allows you to specify the name and location of a top–level command processor that DOS initialization loads in place of COMMAND.COM.

Format: SHELL= [*d*:][path]*filename*[*.ext*]

Remarks: System programmers who develop their own top–level command processor should remember to include provisions for handling interrupts 22H, 23H and 24H, and for reading and executing commands. Because the internal commands and batch processor reside in COMMAND.COM., these functions will not be available to the user unless they are duplicated in your command processor.

SHELL does not affect COMSPEC= or BASIC's SHELL command. To assure that the same command processor is used for reloading (just the transient portion), you must set COMSPEC= to point to that command processor.

Notes:

Chapter 5. Using Tree-Structured Directories

Contents

Introduction **5-3**

Why Use Directories? **5-3**

How Directories Are Organized **5-4**
Directory Entries 5-5

Accessing Your Subdirectories **5-6**
The Current Directory 5-6
Changing Directories with CHDIR 5-7
Specifying a Path to a File 5-7
Using the PATH Command 5-8
Using PATH in a Batch File 5-9

Using Directory Commands **5-10**
Making a Subdirectory 5-11
Removing a Subdirectory 5-13
Displaying and Changing the Current Directory 5-14
Displaying the Directory Structure 5-15

Where DOS Looks for Commands and Batch Files **5-15**

Notes:

Introduction

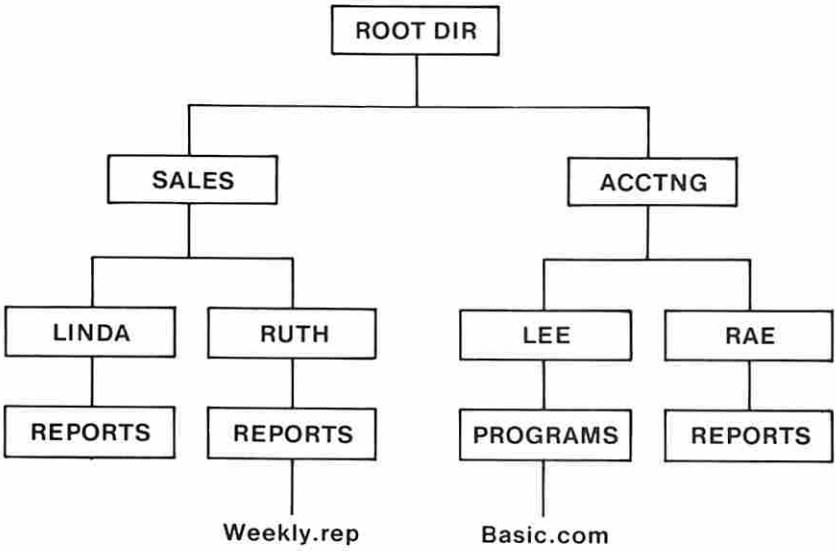
This chapter describes how to organize files into groupings called directories. It tells you how to create, access, remove and display the contents of your directories. It also tells you how to display the entire directory structure of your disk.

Why Use Directories?

A fixed disk can have many files. If you keep a large number of files in one place on your fixed disk, it can take longer for DOS to find a particular file. Keeping files in related groups in directories can reduce the time required to locate a file.

How Directories Are Organized

To give you an idea of how you might group files into directories, let's assume your company has two departments — sales and accounting — that use an IBM Personal Computer. All of the sales and accounting files are kept on the computer's fixed disk. The directory structure might look like this:



This directory structure reflects the organization of each of the two departments. Directories can be created on different levels. The highest level is the *root directory*. DOS creates a root directory on each disk when you format it. When you start DOS on your computer, you are in the root directory.

Just below the root directory shown in the previous diagram, are two *subdirectories* — one for sales and one for accounting. One level below the subdirectory named SALES, you find two more subdirectories. These two subdirectories can contain files for the salespersons, Ruth and Linda. The complete name of Ruth's subdirectory is actually \SALES\RUTH. The complete name of Linda's subdirectory is \SALES\LINDA.

Ruth and Linda can store files related to their work in their respective subdirectories. But suppose Ruth and Linda want to keep all files related to reports separate from their other files. On the next level, each has a subdirectory named REPORTS. The complete name of each subdirectory is \SALES\RUTH\REPORTS and \SALES\LINDA\REPORTS.

> **Note:** Sales files of a general nature, not related to a particular salesperson, can be stored in SALES.

Directory Entries

A directory *entry* can be a file, a subdirectory, or a volume label. The number of entries that the root directory of a disk can contain depends on the type of disk. The root directory of a single-sided diskette can hold 64 entries, while the root directory of a double-sided diskette can hold 112 entries. The root directory of a high-capacity diskette can hold 224 entries. There can be 512 entries in the root directory of a fixed disk.

Unlike the root directory, subdirectories can contain any number of entries, limited only by the amount of available space on the disk.

Subdirectory names follow the same format as filenames. They can consist of 1 to 8 characters followed by an optional extension of a period and 1 to 3 characters. All characters valid for filenames are also valid for subdirectory names.

Accessing Your Subdirectories

You need to know how to get to your subdirectories and how to access files located in a subdirectory other than the one you are working in. This section discusses:

- The current directory
- The CHDIR command (Change Directory command)
- Specifying a path to a file
- Using the PATH command
- Using PATH in a batch file

The Current Directory

The *current directory* is the one you are currently working in or the one you *were* working in on a different drive. DOS remembers which directory was current on each of your drives, even though you may not be presently accessing any of those drives. When DOS starts, the root directory is the current directory (until you change directory using the CHDIR command).

Changing Directories with CHDIR

Use the CHDIR (CD) command to move in and out of your directories (make a different directory the current directory). For example, if you are in the root directory, and want to work with files in \SALES\LINDA\REPORTS, type:

```
C>cd \SALES\LINDA\REPORTS      (Press Enter)
```

You must "go through" SALES and LINDA to get to REPORTS. The leading backslash is really not necessary if you are currently in the root directory.

If you are in LINDA and you want to change to the REPORTS subdirectory, you type:

```
C>cd reports      (Press Enter)
```

You always get back to the root directory no matter where you are in the directory structure, by typing:

```
C>cd \      (Press Enter)
```

The backslash (\) signifies the root directory.

The longest path that you can specify with CHDIR is 64 characters.

Specifying a Path to a File

When you want DOS to locate a file, and you do not specify a particular directory, DOS searches only the current directory of your default drive.

DOS can search for a file in a directory other than the current directory of your default drive. To do so, DOS must know three things — the drive, the name of the directory, and the name of the file.

These three components can be combined to create a *path* to a file. For example, if your default drive is A,

and you want to copy a file named WEEKLY.REP from drive C in the directory \SALES\RUTH\REPORTS to drive A, type:

```
A>copy c:\sales\ruth\reports\weekly.rep      (Press Enter)
```

This command copies the file WEEKLY.REP from a directory three levels below the root on drive C to the current directory on drive A. Note that the filename is the last item in the path.

Using the PATH Command

You can use the PATH command to tell DOS where to search for executable files that it does not find in the current directory. This search is valid only for files with an extension of .EXE, .BAT, or .COM. Data files cannot be located using PATH.

To start a search from the root directory, the path specified in the PATH command must begin with a backslash. Otherwise, the search begins at the current directory.

Let's suppose that you have several applications that require BASIC.COM in order to run. BASIC.COM is only on drive C in the directory \ACCTNG\LEE\PROGRAMS. Your application programs are on diskette and may be run from either drive A or drive B.

You can set the PATH so that DOS looks for BASIC.COM, or any other executable file that may be needed to run your applications or manage your files. DOS searches the directories in the exact order that you specify in the PATH command. Given the diagram at the beginning of the chapter, a valid path is:

```
C>path c:\sales\linda\reports\;c:\acctng\lee\programs
```

After you start the application program, at a specific point, BASIC.COM is required in order for the program to continue. In this example, DOS searches

the current directory of the default drive and then the path that you set when you started the computer. DOS finds the required file in \ACCTNG\LEE\PROGRAMS on drive C. Your application program can continue because BASIC.COM has been invoked.

For more information, see the PATH command in Chapter 7 of this book.

Using PATH in a Batch File

You can avoid having to set the PATH each time you turn on your computer. Include the names of your most-used directories in a PATH statement in an AUTOEXEC.BAT file in the root directory of the disk from which you start DOS.

AUTOEXEC.BAT is a special type of batch file that DOS looks for when you first start your computer. If DOS finds AUTOEXEC.BAT in the root directory, it executes the statements it finds there. The previous example is a valid path that can be included in an AUTOEXEC.BAT file.

Using Directory Commands

The following commands help you create and manage your subdirectories and directory structure:

- MKDIR (MD)—used in making a new subdirectory.
- RMDIR (RD)—used in removing a subdirectory.
- CHDIR (CD)—used in displaying and changing the current directory.
- TREE—used in displaying the entire directory structure of a disk.

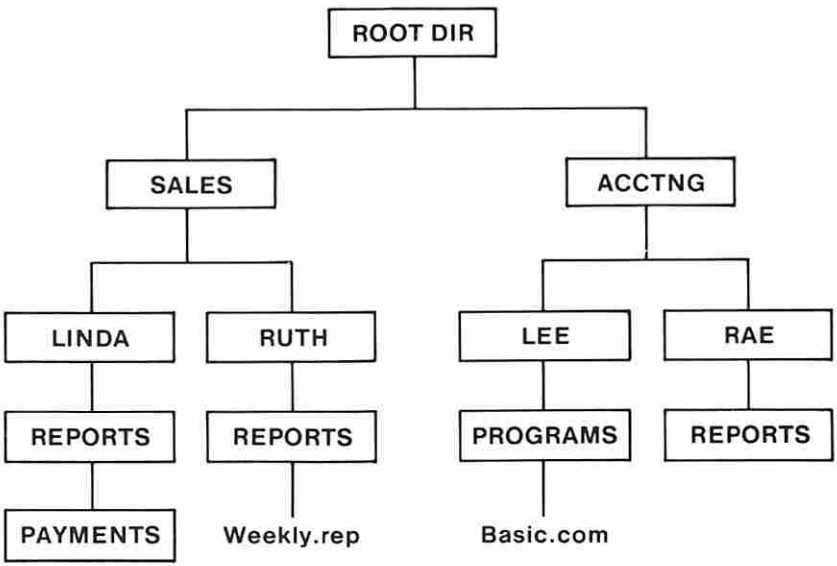
Making a Subdirectory

Use the MKDIR (MD) command to make a new subdirectory. Be sure to include the appropriate drive and path, ending with the name of the new subdirectory you want created.

Let's assume you want to add the PAYMENTS subdirectory one level below SALES\LINDA\REPORTS. Make sure you are in the root directory of the correct disk drive and type the following:

```
C>MD \SALES\LINDA\REPORTS\PAYMENTS    (Press Enter)
```

The resulting subdirectory structure looks like this:



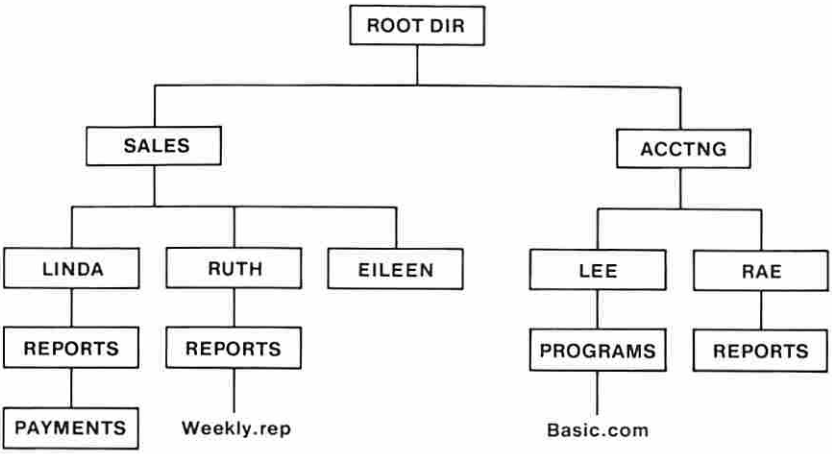
Assume you want to add a subdirectory for a third salesperson just below the SALES subdirectory. The salesperson is Eileen. If you are in the root directory, type:

```
C>md \sales\eileen      (Press Enter)
```

If you are in the SALES directory, type:

```
C>md eileen      (Press Enter)
```

The directory structure of your fixed disk now looks like this:



Removing a Subdirectory

Use the RMDIR (RD) command to remove a subdirectory. You cannot remove a subdirectory by using the ERASE or DEL commands. Before removing a subdirectory, keep the following in mind:

- A subdirectory can be removed only if it is empty. That is, it contains only the special entries (.) and (..). When you display the contents of a subdirectory, using the DIR command, you see those two special entries listed.
- Only one subdirectory can be removed at a time — only the last one specified in the path.
- The root directory and the current directory cannot be removed.

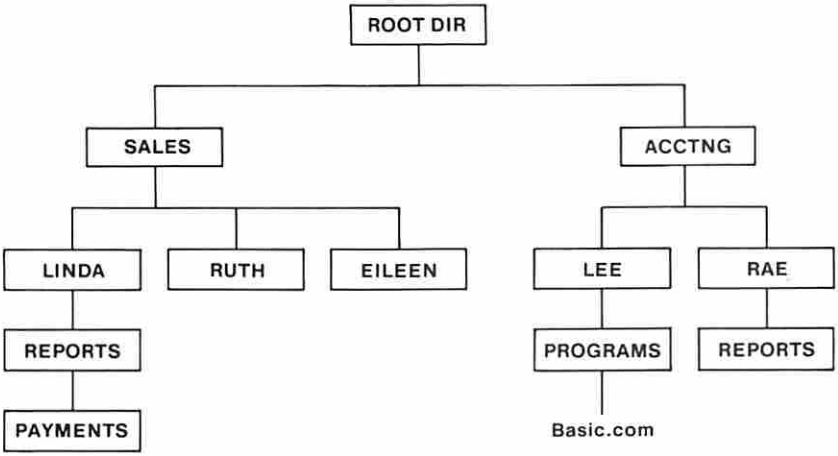
Let's assume you want to remove the REPORTS subdirectory from \SALES\RUTH\REPORTS.

- If you are in the root directory, change directory to the subdirectory you want to remove by typing **CD \SALES\RUTH\REPORTS.**
- Type **erase *.*** to remove all files in the \SALES\RUTH\REPORTS subdirectory.

  Note: No files are erased from the SALES or RUTH subdirectories, only from the REPORTS subdirectory.

- Change directory to the level above the REPORTS subdirectory that you want to remove by typing **cd \SALES\RUTH.**
- Type **RD REPORTS** to remove the subdirectory.

The directory structure now looks like this:



Displaying and Changing the Current Directory

As previously mentioned, you change from one subdirectory to another using the CHDIR (CD) command. See the section entitled, "Changing Directories with CHDIR" earlier in this chapter for more information.

You can also display which subdirectory you are currently working in by typing:

```
cd      (Press Enter)
```

Displaying the Directory Structure

Use the TREE command to find out the names of each subdirectory on a disk. For example, to see the directory structure for disk C, type the following:

```
A> TREE C:
```

For more information on the TREE command, see Chapter 7 of this book.

Where DOS Looks for Commands and Batch Files

When you type a command, if it is an external file, DOS must locate the command file in order to execute the command. You must specify the location (path) that describes which drive and directory the command file is in. You do this by typing the path before the command itself. You can use the PATH command to tell DOS what paths to search if it does not find the command file in the current directory or the path specified on the command line.

> **Note:** This search is only valid for files with a .EXE, .BAT, or .COM filename extension. Data files cannot be located by using the PATH command.

Notes:

Chapter 6. Standard Input and Standard Output

Contents

Introduction 6-3

Redirection of Standard Input and Output Devices . 6-3

Piping of Standard Input and Output 6-6

DOS Filters 6-7

Notes:

Introduction

This chapter describes how you can use redirection, piping, and filters with standard input and standard output. It also describes how to invoke a secondary command processor.

Use redirection of standard input and standard output to change the standard input device or standard output device.

Use piping of standard input and standard output to use the output of one program as the input to another program.

Use filters to sort the input or output of a program, or to find the occurrence of a string in a text file, or to display a screen of data and then pause with the message **--More--**.

Redirection of Standard Input and Output Devices

DOS provides internal functions that programs can use to receive input and display or print output. These functions are called *standard input* and *standard output*.

The DOS standard input and output redirection feature allows a program to receive its input from a source other than the standard input device, or direct its output to a device other than the standard output device.

When you first start DOS, the standard input device is the IBM Personal Computer keyboard, and the standard output device is the display unit (screen). However, DOS lets you specify input and output devices other than the keyboard and the screen. Refer

to the CTTY command, described in Chapter 7 of this book to see how you can specify a remote terminal as the standard input and output device instead of the keyboard and screen.

DOS handles the mapping of the logical (standard input and output) devices to the real devices in a way that is not apparent to the application program. The application program does not need to be aware of the physical device that is actually being used for standard input and standard output.

> **Note:** Any output sent to the standard error device cannot be redirected.

This means that you can run your application program using the keyboard and screen for your input and output device, and then run your application at another time using a remote terminal for your input and output device. You do not have to change the application program, which continues to use the standard input and standard output. DOS goes to the correct physical device on behalf of the application program.

The standard input and output devices can be redirected to or from files or other devices by the following DOS command line parameters:

>[*d*:][*path*]*filename*[.*ext*]

> Causes *filename*[.*ext*] to be created (or truncated to zero length) and then assigns standard output to that file. All output that would normally have gone to the screen from the command is placed in the file.

>>[*d*:][*path*]*filename*[.*ext*]

> Causes *filename*[.*ext*] to be opened (created if necessary) and positions the write pointer at the end of the file so that all output is appended to the file.

<[*d*:][*path*]*filename*[.*ext*]

Causes standard input to be assigned to *filename*[.*ext*]. All input to the program comes from this file instead of from the keyboard.

CAUTION
When using this method of providing input to a program, be sure *all* of the program's input is in the file. If the program attempts to obtain more input after end–of–file is reached, DOS is unable to supply the input, and processing stops. You may have to restart the system by pressing Ctrl-Alt-Del.

Note: If an application does not use DOS function calls to perform standard input and output (for example, it puts text directly into the video buffer), then redirection will not work for that application.

Example: In the following example, the output of the DIR command is sent to the printer:

```
A>dir >prn
```

In the following example, the output of the DIR command is sent to the file DIRLIST:

```
A>dir >dirlist
```

In the following example, the program MYPROG will receive its input from file INPUT.TXT, instead of from the keyboard:

```
A>myprog <input.txt
```

Piping of Standard Input and Output

The DOS piping feature allows the standard output of one program to be used as the standard input to another program. DOS uses temporary files to hold the input and output data being piped. These temporary files are created in the root directory of the default drive.

The programs being piped must not cause the piping files to be erased or modified.

Piping is the chaining of programs with automatic redirection of standard input and output (refer to "Redirection of Standard Input and Output Devices" in this chapter for additional information). The names of the programs to be chained are separated by the vertical bar (|) character on the command line.

The following are typical examples of using the piping feature for a program that does all of its input and output to the standard input and output devices. For example, if the program named SORTABC read all of its input from standard input, sorted it, and then wrote it to the standard output device, the command:

```
A>dir | sortabc
```

would generate a sorted directory listing. This causes all standard output generated by the DIR command to be sent to the standard input of the SORT program.

To put the sorted directory in a file, you would type:

```
A>dir | sort > file
```

If you want the file to contain only the directory entries for subdirectories, you could type:

```
A>dir | find"<DIR>" | sort > file
```

DOS Filters

A filter is a program or command that reads data from a standard input device, modifies the data, then writes the result to a standard output device. Thus, the data has been "filtered" by the program. For example, one of the filters on your DOS diskette is called SORT. SORT reads input from the standard input device (normally the keyboard), sorts the lines of data, then writes the sorted results to the standard output device (normally the screen). With the redirection capabilities described earlier in this chapter, you can cause SORT to receive its input from some other source, and to send its output to a different destination. For example,

```
A>sort <myfile >result
```

will cause SORT to read the file MYFILE, sort the lines within it, and write the sorted output to file RESULT.

By using the piping feature, you can cause a filter to receive its input from the output of another command, or to send its output to the input of another command. For example,

```
A>dir | sort
```

causes the output listing from the DIR command to be used by SORT as its input. The listing will be sorted and the result displayed on the standard output device.

There are three filters on your DOS diskette, and they are described as individual commands in Chapter 7. They are:

SORT Sorts text data.

FIND Searches files for occurrences of specified strings of text.

MORE Displays a screen full of data at a time, then pauses with the message **--More--**.

You can add your own filter to the filters that have been supplied; just write a program that reads its input from the standard input device, and writes its output to the standard output device.

> **Note:** If an application does not use DOS function calls to perform standard input or output (for example, it puts text directly into the video buffer), filters will not work for that application.

Chapter 7. DOS Commands

Contents

Introduction **7-5**

DOS Commands and the Network **7-7**

Types of DOS Commands **7-8**

Entering a DOS Command **7-9**

Information Common to All DOS Commands ... **7-10**

DOS Commands **7-13**

ASSIGN (Drive) Command **7-14**

ATTRIB (Attribute) Command **7-17**

BACKUP Command **7-19**

Batch File Commands **7-24**
- Creating a Batch File 7-26
- Executing a Batch file 7-27
- The AUTOEXEC.BAT File 7-27
- Creating an AUTOEXEC.BAT file 7-28
- Creating a Batch File with Replaceable Parameters 7-28
- Executing a Batch File with Replaceable Parameters 7-29
- ECHO Subcommand 7-30
- FOR Subcommand 7-32
- GOTO Subcommand 7-33
- IF Subcommand 7-34
- PAUSE Subcommand 7-38
- REM (Remark) Subcommand 7-39
- SHIFT Subcommand 7-40

BREAK (Control Break) Command **7-43**

CHDIR (Change Directory) Command **7-45**

CHKDSK (Check Disk) Command **7-48**

CLS (Clear Screen) Command **7-52**

COMMAND (Secondary Command Processor) Command **7-53**

COMP (Compare Files) Command **7-55**

COPY Command **7-60**

CTTY (Change Console) Command **7-72**

DATE Command **7-74**

DEL (Delete) Command **7-77**

DIR (Directory) Command **7-79**

DISKCOMP (Compare Diskettes Only) Command **7-84**
DISKCOMP Compatibility 7-88

DISKCOPY (Copy Diskettes Only) Command .. **7-90**
DISKCOPY Compatibility 7-94

ERASE Command **7-96**

EXE2BIN Command **7-98**

FDISK Command **7-101**

FIND Filter Command **7-102**

FORMAT Command **7-105**
FORMAT Compatibility 7-110

GRAFTABL (Load Graphics Table) Command **7-113**

GRAPHICS (Screen Print) Command **7-115**

JOIN Command **7-118**
Why Use JOIN? 7-122

KEYBxx (Load Keyboard) Command **7-123**
Allowable Dead Key Combinations ... 7-126

LABEL (Volume Label) Command **7-127**

MKDIR (Make Directory) Command **7-130**

MODE Command **7-132**

MORE Filter Command **7-139**

PATH (Set Search Directory) Command **7-140**

PRINT Command **7-143**

PROMPT (Set System Prompt) Command ... **7-149**

RECOVER Command **7-153**

RENAME (or REN) Command **7-156**

RESTORE Command **7-157**

RMDIR (Remove Directory) Command **7-161**

SELECT Command **7-162**

SET (Set Environment) Command **7-164**

SHARE Command **7-167**

SORT Filter Command **7-169**

SUBST(Substitute) Command **7-172**
Why Use SUBST? 7-177

SYS (System) Command **7-178**

TIME Command 7-180

TREE Command 7-182

TYPE Command 7-185

VER (Version) Command 7-186

VERIFY Command 7-187

VOL (Volume) Command 7-188

Introduction

You can use DOS commands to:

- Compare, copy, display, erase, rename files.
- Format fixed disks and diskettes.
- Execute system programs, such as EDLIN, DEBUG, and LINK, plus your own programs.
- Set various printer and screen options.
- Request DOS to pause.
- Transfer DOS to another diskette.
- Cause printer output to be directed to the Asynchronous Communications Adapter.
- Recover a specific file from a damaged disk, or recover the entire disk or diskette.
- Print the contents of a graphics display screen on a printer.
- Print files on the printer while the system is doing other work.
- Backup and restore files on a fixed disk.
- Define a remote device as your primary console.
- Sort text data.
- Search files for occurrences of specified strings of text.
- Display a screen full of data at a time.
- Set new system prompt.

- Set the system environment.
- Convert .EXE files to .COM files.
- Install file sharing.
- Make a file read–only.
- Add or change a disk volume label.
- Ask DOS to check for Ctrl–Break to exit a program or command at any time.
- Create, remove, and change subdirectories.
- Display all the directories on a disk.
- Check for disk errors.
- Set the date and time.
- Find out what's on a disk.
- Select the date and time format.
- Join a drive to a directory on another drive so that you can access a drive through a subdirectory.
- Substitute a drive letter for another drive or directory in order to access that drive or directory using only the drive letter.

DOS Commands and the Network

You can use most of the DOS commands on network disks, directories, and printers.

For example, use the DIR command to list the contents of a network disk or directory.

Use the COPY command to copy files from a network disk or directory. You can copy the files to your own disk or to a network disk or directory.

There are a few DOS commands that you cannot use with network disks, directories, or printers that you are using (or disks and directories that you are sharing) on the network. Don't use:

- CHKDSK
- DISKCOMP

  Note: Instead of DISKCOMP, use COMP (Compare) to compare files.

- DISKCOPY

  Note: Instead of DISKCOPY, use the COPY command to copy files.

- FORMAT
- JOIN (cannot JOIN a network drive to a local drive)
- RECOVER
- SUBST (cannot SUBST a drive letter for a network path)
- SYS

- PRINT (cannot use PRINT on a network server computer)
- FDISK
- LABEL (cannot change the label on a network disk)

  Note: Using these commands with a network device causes an error message.

Of course, these commands are valid for your real disks, directories, and printers.

Types of DOS Commands

The two types of DOS commands are *internal* commands and *external* commands.

Internal commands execute immediately because they are built into DOS. Therefore, once DOS is loaded, you do not need the DOS diskette in a drive to use these commands.

External commands are on disks as program files. They must be read from the disk before they are executed. This means that the disk containing the command must be in a drive, or DOS is unable to find the command.

Any file with a filename extension of .BAT, .COM or .EXE is considered an external command. This allows you to develop your own unique commands and add them to the system. (For example, programs such as FORMAT.COM and COMP.COM are external commands.)

When you select an external command, you do not have to include the filename extension.

Entering a DOS Command

Use the following format notation to enter DOS commands:

[ ] Items shown inside square brackets are optional. To include optional items, type only the information inside the brackets. Do not type the brackets.

CAPS Words shown in capital letters are called *keywords*. The DOS command names are keywords. You can type keywords in any combination of uppercase and lowercase letters.

italics Items shown in lowercase italic letters mean that you are to substitute the item. If italic items are inside brackets, then they are optional. For example,

filename

indicates that you should type the name of your file in place of the word *filename*.

| A vertical bar means either/or. Choose one of the separated items and type it as part of the command. For example,

```
ON|OFF
```

indicates that you should type either ON or OFF, but not both. Do not type the vertical bar.

... An ellipsis indicates that you can repeat an item.

Include all punctuation such as commas, equal signs, question marks, asterisks, colons, slashes and backslashes. Punctuation shown inside brackets is optional.

Information Common to All DOS Commands

The following information applies to all DOS commands:

- The DOS prompt consists of the default drive letter and the character ">." For example, A> is the DOS prompt that denotes A as the default drive. You can change the DOS prompt by using the PROMPT command.

- When a command has finished executing, the DOS prompt reappears on the screen. If no error messages appear before the DOS prompt returns, the command has been executed successfully.

- Commands are usually followed by one or more parameters.

- You can type commands in uppercase or lowercase or a combination of both. For example, you can type:

  ```
  Dir A:
  ```

- DOS searches the current directory of the drive you specify or the default drive (if you do not specify a drive) to find a command or batch file you typed. If not found, and you have specified a PATH command, DOS searches the directories listed in the path.

- Commands that allow you to enter filenames can accept a path (directory) name before the filename.

- Commands and parameters *must* be separated by delimiters (space, comma, semicolon, equal sign, or the tab key). The delimiters can be different within one command. For example, you could type:

```
A>copy oldfile.rel;newfile.rel
A>rename,thisfile thatfile
```

- Do not separate the three parts of a filespec (d:filename.ext). The colon and period already serve as separators.

- In this book, a space is used as the delimiter in the commands for readability.

- Also in this book, when we say "*Press any key*," we mean "Press any *character* key."

- To stop a command while it is executing, press Ctrl–Break. Ctrl–Break is recognized only while the system is reading from the keyboard or printing characters on the screen, unless you have used BREAK=ON in your configuration file or have issued a BREAK ON command. Thus, the command may not end immediately when you press Ctrl-Break.

- Commands start executing only after you press the Enter key.

- Global filename characters and device names are not allowed in a command name. You can only use them in filenames and filename extensions.

- For commands displaying a large amount of output, you can use the **Pause Screen** function to suspend the display of the output. Press any character key to continue the display.

- You can use the function keys and the DOS editing keys described in Chapter 2 of the *DOS User's Guide* while typing DOS commands.

- Drives are referred to as *source* drives and *target* drives. A source drive is the drive you transfer information *from*. A target drive is the drive you transfer information *to*.

- When an external command is typed, DOS searches for it in the current directory of the default or specified drive. If not found, DOS continues searching for it in the directories listed in the most recent PATH command.

- If you type any of these characters <, >, or ¦ in a command, DOS treats them as redirection and piping characters. Refer to Chapter 6 "Standard Input and Standard Output" for information on redirection and piping.

  For example, if you type the following ECHO command, the > character is treated as a redirection character, not the greater-than character.

```
A>echo file1 > file2
```

  Therefore, the string FILE1 is placed in the file FILE2.

- You can specify a drive and path before external commands. This means that the external command file can be in a directory other than the current directory. For example, if the file FORMAT.COM is in the directory \LEVEL1 on drive B, you can type:

```
A>b:\level1\format
```

DOS Commands

This section presents a detailed description of the DOS commands. The commands appear in alphabetic order. The description includes the purpose, format, and type of each command. Examples are provided where appropriate.

ASSIGN (Drive) Command

Purpose: Instructs DOS to route disk I/O requests for one drive into disk I/O requests for another drive.

Format: [*d*:][*path*]ASSIGN [*x*[=]*y* [...]]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before ASSIGN to specify the drive and path that contains the ASSIGN command file.

x to specify the drive to which current disk I/O requests are sent.

y to specify the drive letter that you want disk I/O requests to now be sent.

The first drive letter *x* is internally converted by DOS to the second drive letter *y*. Both *x* and *y* must physically exist (a diskette, fixed disk or block device driver such as VDISK). Do not type the colon after the drive letters *x* and *y*.

Type ASSIGN with no parameters to reset all drive assignments so that normal drive assignments resume.

> **Note:** This command has been included to assist you with applications that were designed to perform their disk operations specifically on drives A and B (those applications that do not allow you to specify a drive). By using a command such as,

ASSIGN (Drive) Command

```
A>assign a=c b=c
```

those applications can be made to use drives other than A and B, such as a fixed disk.

Reassignment of drives should *only* be used when necessary for these cases. It should never be used with the BACKUP, RESTORE, LABEL, JOIN, SUBST, or PRINT commands.

Also, do not reassign drives when running DOS in normal operations. Doing so can *hide* the true device type from commands and programs that require actual drive information. Note that FORMAT, DISKCOPY and DISKCOMP ignore any drive reassignments.

If you will be developing an application program, we recommend that you avoid using specific drive assignments within your program, but instead, allow the user to specify the drive(s) to be used.

Example: The following example assigns all requests for drive A to drive C. Thus, if you issue **DIR A:**, DOS displays the directory that is on physical drive C:

```
A>assign a=c
```

ASSIGN (Drive) Command

The following example assigns any requests for drive A or drive B to drive C.

```
A>assign a=c b=c
```

The following example resets any previous drive assignments so that requests for drive A go to drive A, etc.

```
A>assign
```

ATTRIB (Attribute) Command

Purpose: Allows you to set or reset the Read-Only file attribute, or to display the current setting of that attribute.

Format: [*d*:][*path*]ATTRIB [+R | –R] [*d*:][*path*]*filename*[.*ext*]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before ATTRIB to specify the drive and path that contains the ATTRIB command file.

+R to set the read attribute of the specified file to read–only.

–R to remove the read–only attribute of the specified file.

[*d*:][*path*]*filename*[.*ext*] to specify the file you want to mark as read–only. Global filename characters are allowed.

Example: The following example sets the attribute of the file name FILE1.TXT to read–only.

```
A>attrib +r file1.txt
```

The following example displays the current setting of the Read-Only attribute for the file FILE1.TXT.

```
A>attrib file1.txt
```

The result is:

```
R       A:\FILE1.TXT
```

The following example removes the read–only attribute from the previous example.

```
A>attrib -r file1.txt
```

The following example displays the current setting of the Read-Only attribute for the file FILE1.TXT.

```
A>attrib file1.txt
```

The result is:

```
        A:\FILE1.TXT
```

The following example sets the attribute of the file C:\PROG1.BAS to read–only.

```
A>attrib +r c:\prog1.bas
```

BACKUP Command

Purpose: Backs up one or more files from one disk to another. The drive specifiers of the disks must be different.

Format: [*d*:][*path*]BACKUP *d*:[*path*][*filename*[.*ext*]] *d*: [/S][/M][/A][/D:*mm-dd-yy*]

Type: Internal External
***

Remarks: You can back up files from a:

- Fixed disk to a diskette
- Diskette to a diskette
- Diskette to a fixed disk
- Fixed disk to a fixed disk

Specify the parameters:

[*d*:][*path*] before BACKUP to specify the drive and path that contains the BACKUP command file.

d: to specify the drive that contains the files you want to BACKUP (source). [*path*][*filename*[.*ext*]] to specify the name of the file you want to BACKUP.

d: to specify the drive that will contain the backed up files (target).

/S to back up subdirectory files in addition to the files in the specified or current directory.

/M to back up files that have been modified since the last backup.

BACKUP Command

/A to add the files to be backed up to the files already present on the backup disk.

/D to back up files that have been modified on or after the specified date. The format of the date specified will be [*mm-dd-yy*] or [*dd-mm-yy*] or [*yy-mm-dd*] depending on the country code you selected using the SELECT or COUNTRY commands.

Notes:

1. The disk that contains the file you want to back up is called the *source*. The disk where the backup file is placed is called the *target*. For example, if you want to back up a file from your fixed disk to a diskette, the fixed disk is the source and the diskette is the target.

2. Global filename characters are allowed in the filename and extension. Therefore, you can backup all files (including subdirectory files) from the source drive C to the target drive A by typing:

```
A>backup c:\*.* a:/s
```

3. BACKUP is not the same as COPY. COPY makes an exact duplicate of the file. BACKUP files have control data included that the RESTORE command uses. Therefore, you cannot use backed up files as data files until they are restored. Refer to the RESTORE command in this chapter for information on using RESTORE.

BACKUP Command

4. If you are sharing files, you can only BACKUP files that you have access to. If you attempt to access a file that you do not have access to, the following message appears:

```
PATHNAME\FILENAME.EXT
Not able to backup at this time
```

5. If the target is a new diskette, use DOS to format the diskette before you backup files.

6. If the diskette contains files, BACKUP erases the files already present on the diskette unless you specify /A parameter.

7. If the source or target is removable, you are prompted to insert the source or target diskette into the drive.

8. After BACKUP fills up a diskette, you are prompted to insert a new diskette. Label each diskette and record the date and diskette number.

9. BACKUP displays the name of each file as it is backed up. Label each backup diskette in consecutive order because, when the files are restored, you are prompted to insert the backup diskette in order.

10. The BACKUP command sets the exit code as follows:

    0 Normal completion

    1 No files were found to backup

    2 Some files not backed up due to file sharing conflicts

3 Terminated by user (Ctrl-Break)

4 Terminated due to error

These codes can be used with the batch processing IF ERRORLEVEL subcommand.

11. If the target is a fixed disk, the backup files are placed in a subdirectory named \BACKUP. If the target is a diskette, the backup files are placed in the root directory.

12. If the source is a diskette, it should not be write protected because BACKUP needs to mark the files as being backed up (/M).

13. Do not use BACKUP with a drive letter that has been assigned or substituted as this can hide the true drive letter that BACKUP is using.

14. Do not use BACKUP while a JOIN is in effect because the tree structure may be invalid when you RESTORE.

Example: The following example backs up all files from the fixed disk drive C to the diskette in drive A.

```
A>backup c:*.*  a:/s
```

Note: To back up a 10M byte fixed disk onto double–sided 9 sector diskettes, you need approximately 25 diskettes.

BACKUP Command

The following example backs up the file named FILE.TXT from drive A to drive C.

```
A>backup a:file.txt c:
```

The following example backs up all files from drive A that have been modified since the date 8–21–84.

```
A>backup a: c:/d:8-21-84
```

The following example adds the files on drive A to the backed up files already on the diskette in drive B.

```
A>backup a: b:/a
```

Batch File Commands

Purpose: Batch commands are DOS commands that are contained in a special file called a *batch* file. When you execute a batch file, DOS executes the commands you include in the batch file.

Format: [*d:*][*path*]*filename*[.BAT] [*parameters*]

Type: Internal External
***

Remarks: A batch file is a file containing one or more commands that DOS executes one at a time. All batch files must have a filename extension of .BAT.

You can pass parameters (using replaceable parameters) to the *filename*.BAT file when the file is executed. Therefore, the file can do similar work with different data during each execution.

You can create a batch file by using the Line Editor (EDLIN), or by using the COPY CON command directly from the standard input device.

Notes:

1. Do not enter the name BATCH (unless the name of the file you want to execute is BATCH.BAT).

2. Do not name batch files with internal command names.

3. You do not need to type the filename extension .BAT to execute a batch file.

Batch File Commands

4. The commands in the file named *filename*.BAT are executed.

5. There are seven subcommands that can be used to control batch processing: ECHO, FOR, GOTO, IF, SHIFT, PAUSE, and REM. They are explained in the following pages.

6. If you remove a diskette containing a batch file being processed, DOS prompts you to insert it again before the next command can be read.

7. The last command in a batch file may be the name of another batch file. This allows you to invoke one batch file from another when the first is finished. If a batch file references another batch file, the second batch file does not return to the first batch when if finishes unless the first batch file contains the COMMAND command.

8. DOS *remembers* which directory the batch file was started from. Therefore, the commands within the batch file may change the current directory during execution.

9. Batch files execute quickly if placed on a virtual disk. Refer to Chapter 4 "Configuring Your System" for information on virtual disks (VDISK.SYS).

Batch File Commands

Creating a Batch File

You can create a batch file by using an editor or by using the COPY command directly from the standard input device. Follow these steps to create a batch file using the COPY command.

1. At the DOS prompt type:

   A>copy con *filename*.bat

   Substitute the name you want for your batch file for *filename*.

2. Press Enter.

3. Type the commands you want to include in the batch file. Press Enter after you type each command.

   Note: If you make a mistake typing a command, press Ctrl–Break. Pressing Ctrl–break ends the COPY command *without* saving the batch file. Then repeat the procedure again.

4. Press F6 and then Enter when you have finished typing commands. This ends the COPY command and saves the batch file. This message is displayed:

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

Batch File Commands

Executing a Batch file

Batch files are executed by typing the name of the batch file at the DOS prompt and then pressing Enter. For example, to execute a batch file named EXAMPLE.BAT, at the DOS prompt type:

```
A>example
```

then press Enter.

You do not need to type the batch filename extension .BAT.

To stop the execution of a batch file, press Ctrl–Break. The command being executed ends and the following prompt is displayed:

Terminate batch job (Y/N)?

Type **y** to terminate the batch file. The remaining commands are ignored and the DOS prompt is displayed.

Type **n** to continue executing the remaining commands.

The AUTOEXEC.BAT File

Every time you start DOS, the command processor searches for a file named AUTOEXEC.BAT in the root directory on the disk that DOS was started from. An AUTOEXEC.BAT file is a special type of batch file that is automatically executed when you start or restart DOS. An AUTOEXEC.BAT file is useful if you want to execute certain commands

every time you start DOS. For example, you can create an AUTOEXEC.BAT file that sets the DOS path by including the PATH command in the file.

Creating an AUTOEXEC.BAT file

An AUTOEXEC.BAT file must be created in the root directory of the disk you start DOS from. You are not prompted for the date and time when you start DOS, unless you include the DATE and TIME commands in the AUTOEXEC.BAT file.

Creating a Batch File with Replaceable Parameters

Within a batch file you can include *dummy* parameters that can be replaced by values supplied when the batch file executes.

For example, if the batch file EXAMPLE.BAT contains these commands:

```
copy %1.mac %2.mac
type %2.prn
type %0.bat
```

the replaceable parameters %0, %1, and %2 are replaced sequentially by the parameters you supply when you execute the file. The replaceable parameter %0 is always replaced by the drive specifier, if specified, and the filename of the batch file. In the AUTOEXEC.BAT file, %0 is not set to any value.

Notes:

1. Up to 10 dummy parameters (%0 – %9) can be specified within a batch file, more than 10 parameters can be specified on a command line (see SHIFT subcommand).

2. If you want to use % as part of a filename *within* a batch file, you must specify it twice. For example, to specify the file ABC%.EXE you must type it as ABC%%.EXE in the batch file.

Executing a Batch File with Replaceable Parameters

To execute the EXAMPLE.BAT file and pass parameters, type the batch filename followed by the parameters you want sequentially substituted for %1, %2, etc.

For example, you can type:

```
A>example a:prog1 b:prog2
```

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

The result is the same as if you typed each of the three commands (in the EXAMPLE.BAT file) from the console with their parameters, as follows:

```
A>copy a:prog1.mac b:prog2.mac
A>type b:prog2.prn
A>type example.bat
```

Batch File Commands

ECHO Subcommand

Purpose: Allows or inhibits the screen display of DOS commands executed from a batch file. It does not interfere with messages produced while the commands are executing.

Format: ECHO [ON | OFF | *message*]

Type: Internal External
***

Remarks: Batch commands are normally displayed on the screen as they are read from the batch file. ECHO is ON after power–on or system reset. ECHO ON displays all the commands on the standard output device as they are executed. ECHO OFF stops the display of commands on the screen (including the REM command).

ECHO *message* displays **message** on the standard output device regardless of the current ON or OFF state. In this way, you can cause specific messages to be displayed even when ECHO has been turned off. If ECHO is issued with no parameters, the current ECHO state (ON or OFF) is displayed.

Example: In this example, the batch file contains the following:

```
echo off
rem **** command display is off
dir a:/w
echo on
dir a:/w
```

Batch File Commands

When this batch file is executed, the following display occurs:

```
A>echo off

  Volume on drive A has no ID
  Directory of A:\

  file1.ext  file2.ext

  2 file(s)    xxxxx bytes free

A>echo on

A>dir a:/w

  Volume on drive A has no ID
  Directory of  A:\

  file1.ext    file2.ext

  2 file(s)    xxxxx bytes free
```

In the above example, the ECHO OFF is displayed. The **rem** command and **dir a:/w** are not displayed because ECHO is OFF, but the output of the **dir** is still displayed. Then, **echo on** is executed, and the command **dir a:/w** is displayed.

Batch File Commands

FOR Subcommand

Purpose: Allows iterative execution of DOS commands.

Format: FOR %%*variable* IN (*set*) DO *command*

Type: Internal External
***

Remarks: The %% *variable* is sequentially set to each member of *set* and then the *command* is evaluated and executed. If a member of *set* is an expression involving * and/or ?, then the %% *variable* is set to each matching filename from disk. Path names are allowed in *set*.

> **Note:** %% is required if the FOR command is included in a batch file. To type the FOR command at the DOS prompt, only include one %. For example you can type:

```
A>for %h in (file1) do dir
```

Example: In the following example, if you type the command:

```
for %%f in (prog1.asm prog2.asm prog3.asm) do dir %%f
```

The result is the same as if you typed the following three commamds:

```
dir prog1.asm
dir prog2.asm
dir prog3.asm
```

> **Note:** FOR subcommands cannot be nested; that is, only one FOR subcommand can be specified on a command line.

GOTO Subcommand

Purpose: Transfers control to the line following the one containing the appropriate label. A label is inserted in a batch file as a colon (:) followed by the label name.

Format: GOTO *:label*

Type: Internal External
***

Remarks: The GOTO *label* causes commands to be executed beginning with the line immediately after *:label*. If *:label* is not defined, the current batch file terminates with the message **Label not found**. A label in a batch file is defined as a character string where the first 8 characters are significant (make it different).

Example: In this example, the following batch file produces an indefinite sequence of **rem looping...** and **goto loop** messages on the screen:

```
:loop
rem looping...
goto loop
```

Note that labels within a batch file are never displayed while the batch file is executing. In the example above, the line **:LOOP** would not be displayed. Thus, unreferenced labels provide a handy means for placing comments within your batch file that are not displayed when the file is executed.

Batch File Commands

IF Subcommand

Purpose: Allows conditional execution of DOS commands.

Format: IF [NOT] *condition command*

Type: Internal External
***

Remarks: The *condition* parameter is one of the following:

ERRORLEVEL *number*

string1==string2

EXIST [*d:*][*path*]*filename*[.ext]

When the *condition* is true, then the DOS command is executed. Otherwise, the DOS command is skipped, and the next command in the file is executed.

ERRORLEVEL *number* is true if the previous program had an exit code of *number* or higher. The number is specified as a decimal value.

string1==string2 is true when *string1* and *string2* are identical.

> **Note:** The corresponding characters of *string1* and *string2* must both be uppercase or lowercase to be identical.

EXIST is true if *filename* is found in the specified directory. Global filename characters (? and *) are allowed in *filename*.

The NOT *condition* is true if the *condition* is false.

Batch File Commands

Example: This example is for:

IF EXIST [*d*:][*path*]*filename*[.*ext*]

```
if exist file1 goto abc
        .
        .
        .
:abc
command
```

Execution of this command in a batch file is true provided FILE1 is found in the current directory on the default drive. The GOTO ABC is executed causing the system to skip to the command following the label :ABC. If FILE1 is not found, the GOTO ABC would not be executed. Processing would then continue with the next command in the batch file.

The following example is for:

IF *string1*==*string2*

```
if %1 == John echo John was here!
```

Execution of a batch file containing this command with JOHN given as the **%1** parameter would make the condition true. The ECHO batch command would then be executed displaying **John was here!** If JOHN was not given as the **%1** parameter, the condition would have been false. The ECHO batch command would not have been executed. Processing would continue with the next command in the batch file.

The following example is for:

IF ERRORLEVEL *number*

```
myprog1
if errorlevel 1 echo myprog1 failed
```

The above two commands are in a batch file; MYPROG1 is a program that sets the errorlevel when it completes its processing. In the simple case, MYPROG1 sets the errorlevel to 0 if it completed processing successfully and sets errorlevel to 1 if processing completed unsuccessfully. The batch file conditional **if errorlevel 1 echo ...** tests for the situation when MYPROG1 failed. If MYPROG1 completed processing unsuccessfully, the condition is true and the ECHO batch command is executed. The ECHO batch command displays the data (or message) immediately following *echo*. If MYPROG1 was successful, the condition would not be true and the ECHO batch command would not be executed. Processing would then continue with the next command in the batch file.

The following example is for:

IF NOT EXIST [*d*:][*path*]*filename*[.*ext*]

```
if not exist a:%1 copy b:%1 a:
program
```

Batch File Commands

The batch file that contains this command is going to execute a program that requires a particular file to be on drive A. The IF command is executed prior to the program to make sure that the required file is on drive A. If the file does not exist on drive A, the condition is true. The COPY command is then executed, copying the file from drive B to drive A to satisfy the requirements of the program. If the file does exist on drive A, the copy is not executed. Processing then continues to execute the program.

> **Note:** Only BACKUP and RESTORE commands set an ERRORLEVEL that can be tested. The facility is included to allow your own programs to set an error code that can then be tested by the IF ERRORLEVEL subcommand.

Batch File Commands

PAUSE Subcommand

Purpose: Suspends system processing and displays the message **Strike a key when ready....**

Format: PAUSE [*remark*]

Type: Internal External
***

Remarks: You can insert PAUSE commands within a batch file to display messages and to give you the opportunity to change diskettes between commands. To resume execution of the batch file, press any key *except* Ctrl–Break. (Ctrl–Break ends processing).

If you include the optional *remark*, the remark is also displayed. The optional remark can be any string of characters up to 121 characters long.

You can control how much of a batch file you want to execute by placing PAUSE commands at strategic points in the file. At each PAUSE command, the system stops and gives you time to decide whether to end processing. To end processing, press Ctrl–Break and then type **y**. To continue processing, press **n**.

Example: If you type this PAUSE command in a batch file, the following message is displayed:

```
A>pause Change diskette in drive A
Strike a key when ready..._
```

This PAUSE enables you to change diskettes between commands.

REM (Remark) Subcommand

Purpose: Displays remarks from within a batch file.

Format: REM [*remark*]

Type: Internal External
***

Remarks: The *remarks* are displayed when the batch execution reaches the REM command. If ECHO is OFF, then the remarks are not displayed.

Remarks can be any string of characters up to 123 bytes long.

You can use REM commands without remarks for spacing within your batch file, for readability.

Example: If the following REM command is issued in a batch file, this remark is displayed:

```
A>rem This is the daily checkout program
```

Batch File Commands

SHIFT Subcommand

Purpose: Allows command lines to make use of more than 10 (%0 through %9) replaceable parameters.

Format: SHIFT

Type: Internal External
***

Remarks: Replaceable parameters are numbered %0 through %9. If you wish to use more than 10 replaceable parameters on a command line, you can use SHIFT to get at parameters past the tenth. All parameters on the command line are shifted one position to the left, with the %0 parameter being replaced by the %1 parameter, etc.. Each subsequent shift command causes all the parameters to be shifted to the left by one position. For example:

```
%0 = A
%1 = B
%2 = C
%3 = D
.
.
.
%9
```

The SHIFT results are:

```
%0 = B
%1 = C
%2 = D
.
.
.
%9
```

Batch File Commands

Example: This example demonstrates how the SHIFT subcommand can be used in a batch file. If a batch file named MYFILE.BAT contains the following commands, and the default drive is A:

```
echo %0 %1 %2 %3
shift
echo %0 %1 %2 %3
shift
echo %0 %1 %2 %3
shift
echo %0 %1 %2 %3
shift
echo %0 %1 %2 %3
```

Invoke the batch file with the following parameters:

```
MYFILE   PROG1   PROG2   PROG3
```

The result is:

```
A>echo MYFILE   PROG1   PROG2   PROG3
MYFILE   PROG1   PROG2   PROG3

A>shift

A>echo PROG1   PROG2   PROG3
PROG1   PROG2   PROG3

A>shift

A>echo PROG2   PROG3
PROG2   PROG3

A>shift

A>echo PROG3
PROG3
A>shift
A>echo

A>
```

BREAK (Control Break) Command

Purpose: Allows you to instruct DOS to check for a control break whenever a program requests DOS to perform any functions.

Format: BREAK [ON | OFF]

Type: Internal External
***

Remarks: Specify the parameters:

ON to set BREAK=ON. This means that DOS checks for Ctrl–Break whenever it is requested. This allows you to *break–out* of a program that produces few or no standard device operations.

OFF to set BREAK=OFF. This means that DOS only checks for Ctrl–Break during:

- Standard input operations
- Standard output operations
- Standard print device operations
- Standard auxiliary device operations

The default value is set at BREAK **OFF**.

BREAK (Control Break) Command

If you type BREAK with no parameters, the current state of the BREAK command (**ON** or **OFF**) is displayed.

You can also turn on the extended checking by using BREAK=ON in your configuration file. Refer to "Configuring Your System" in Chapter 4.

CHDIR (Change Directory) Command

Purpose: Changes the DOS current directory of the specified or default drive, or displays the current directory path of a drive.

Format: CHDIR [*d:*][*path*]

or

CD [*d:*][*path*]

Type: Internal External
***

Remarks: Specify the parameters:

[*d:*] to specify the drive specifier of the disk whose current directory you want to change or display.

[*path*] to specify the current directory path. The path cannot be more than 64 characters starting from the root directory.

DOS looks in the current directory to find files whose names are entered without specifying a path. If you do not specify a drive, the default drive is assumed.

Type **CHDIR** or **CD** with no parameters to display the current directory path of the default drive.

Example: The following example changes the current directory of the default drive to its root directory:

```
A>chdir \
```

CHDIR (Change Directory) Command

The following example displays the current directory path of drive B.

```
A>cd b:
```

The following example changes the current directory of drive B to the path \LEVEL1\LEVEL2.

```
A>cd b:\level1\level2
```

The following example changes the directory of drive B to the current directory path plus LEVEL3:

```
A>cd b:level3
```

Thus, if the second example is used, the resultant path would be:

```
\LEVEL1\LEVEL2\LEVEL3
```

The search for the LEVEL3 directory begins in the directory that was current when the command was issued, because no leading backslash (\) was used.

The following example changes the current directory of drive B to \LEVEL1.

```
A>cd b:\level1
```

The leading backslash (\) tells DOS to start at the root directory. DOS remembers the current directory for each drive on the system, and any reference to a drive will access the current directory.

CHDIR (Change Directory) Command

Important: The true directory can be hidden by ASSIGN, SUBST, and JOIN. The following example describes a method of using C: while actually on A:.

```
A>join c: a:\cdrive
A> cd \cdrive
A>dir
```

CHKDSK (Check Disk) Command

Purpose: Analyzes the directories, files, and the File Allocation Table on the designated or default drive and produces a disk and memory status report.

Format: [*d*:][*path*]CHKDSK
[*d*:][*path*][*filename*[.*ext*]][/F][/V]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before CHKDSK to specify the drive and path that contains the CHKDSK command file.

[*d*:][*path*][*filename*[.*ext*]] to specify the filename. If you specify a filename, CHKDSK displays the number of noncontiguous areas occupied by the file or files.

/F to have CHKDSK fix errors that are found in the directory or file allocation table. The corrections are written on the disk. If /F is not specified, CHKDSK functions as though it were preparing to correct the disk so that you can analyze the possible results of correction, but does not write the corrections to the disk. When you have subdirectories that cannot be reached, you get the message **Cannot CHDIR to d:path, tree past this point not processed.** If this occurs, CHKDSK does not free any allocation units on the disk.

/V to display all files and their paths on the default or specified drive.

CHKDSK (Check Disk) Command

After checking the disk, CHKDSK displays any error messages, followed by a status report. A complete listing of error messages can be found in Appendix A of this book.

CHKDSK does not wait for you to insert a diskette. It assumes that the diskette you want to check is in the specified drive. Therefore, on a one–diskette drive system, it is especially important that the specified drive is different from the default drive, unless you are checking the DOS diskette itself.

You should run CHKDSK occasionally for each disk to ensure the integrity of the file structures.

Notes:

1. All yes or no **(Y/N)** prompts from CHKDSK require you to press Enter after typing **y** or **n**, to prevent accidental changes to your disk.

2. If you specified a filename, the number of non–contiguous areas occupied by the file will be reported. Badly fragmented files (many non–contiguous areas) can cause system performance to slow down when those files are accessed, since DOS can not read them sequentially. You can determine the extent of file fragmentation by using *.* in the filename field of the CHKDSK command.

3. If CHKDSK finds *lost* allocation units (clusters) on the disk, it asks if you wish to recover the lost data into files. If you say yes, and the **/F** parameter was used, CHKDSK recovers each chain of lost allocation units into a file whose name is in the form:

   FILE*nnnn*.CHK

Where *nnnn* is a sequential number starting with 0000. These files are created in the root directory of the specified drive. You can then look at these files to see if they have any useful information. If not, you can erase them.

4. If you redirect CHKDSK's output to a file, for example:

```
A>chkdsk b:>file
```

It will report errors to that file. In this case, be sure not to use the **/F** parameter.

5. CHKDSK does not work on network drives or drives involved in a substitution (SUBST) or JOIN.

Example: Following is an example CHKDSK status report:

```
Volume MYDISK      Created AUG 12, 1984 10:00

   179712 bytes total disk space
    18944 bytes in 3 hidden files
      512 bytes in 1 directories
    26112 bytes in 4 user files
   134144 bytes available on disk

   196608 bytes total memory
   170736 bytes free
```

Note that in this status report, *three hidden* files were reported. These are the volume label, and the DOS system files IBMBIO.COM and IBMDOS.COM, that are hidden from the normal directory searches. Some application programs also create hidden files.

CHKDSK (Check Disk) Command

The following example produces a CHKDSK status report for the diskette in drive A.

```
A>chkdsk a:
```

The following example produces a CHKDSK status report for the diskette in drive A and fixes any errors found in the directory of file allocation table.

```
A>chkdsk a:/f
```

The following example produces a CHKDSK status report for the diskette in drive A and displays all the files and their paths on drive A.

```
A>chkdsk a:/v
```

The following example produces a CHKDSK status report for disk drive C and lists the filenames on drive C that contain non–contiguous areas.

```
A>chkdsk c:*.*
```

In the previous example, if the files FILE1, FILE3, and FILE6 contain non–contiguous blocks, CHKDSK would display these messages:

```
C:\FILE1
    Contains 2 non-contiguous blocks
C:\FILE3
    Contains 2 non-contiguous blocks
C:\FILE6
    Contains 2 non-contiguous blocks
```

CLS (Clear Screen) Command

Purpose: Clears the display screen.

Format: CLS

Type: Internal External
***

Remarks: This command clears the display on the standard output device. If screen attributes have been selected using the "Extended Screen and Keyboard Control" functions in Chapter 3 of the *DOS Technical Reference*, the attributes remain unchanged.

COMMAND (Secondary Command Processor) Command

Purpose: Use the following command to invoke a secondary command processor:

Format: COMMAND [*d*:][*path*][/P][/C *string*]

Remarks: Specify the parameters:

[*d*:][*path*] is the drive and path that DOS searches to find the command processor you want to invoke. If a COMMAND.COM is not found in the specified directory, DOS searches the path in your environment for it. COMMAND.COM loads the transient portion from the file specified in COMSPEC= as part of its initialization.

/P causes the copy of the new command processor to become permanent in memory. If you specify /P, the second command processor does not return to the primary command processor. You must restart DOS to remove the second command processor.

[**/C** *string*] allows you to pass a *string* and then automatically exit back to the primary command processor after the command is completed.

string is a command that you want to pass to the command processor. The command is interpreted and acted upon as if you had typed it at the DOS prompt. For example, if you type:

```
A>command /c dir b:
```

a secondary command processor is loaded, and it executes the command **dir b:** and then exits back to the primary command processor.

COMMAND (Secondary Command Processor) Command

Issuing COMMAND without any parameters causes a new copy of the command processor to be loaded, and this new copy will inherit the environment known to the previous level of the command processor. If you use the SET command to change the environment known to the secondary command processor, that change is known *only* to the secondary copy. Exiting back to the primary command processor causes a resumption of the environment that the primary command processor knew before the secondary copy existed.

If you specify the /P and /C parameters together, then the /P parameter is ignored.

Example: Let's assume that the primary command processor used the normal DOS prompt of **ng** (which is the default prompt). If you invoke a secondary copy of the command processor, the secondary copy "inherits" that prompt. If you change the secondary's prompt to something else, and then you exit back to the primary, the primary's prompt will still be **ng**.

When a secondary command processor is loaded with no parameters specified, you can cause it to return to the previous level of command processor by issuing the special command **EXIT**.

> **Note:** Application programmers: please refer to Chapter 7 of *DOS Technical Reference* for additional information.

COMP (Compare Files) Command

Purpose: Compares the contents of the first set of specified files to the contents of the second set of specified files.

> **Note:** This command compares two sets of *files*; the DISKCOMP command compares two *entire diskettes*.

Format: [*d*:][*path*]COMP [*d*:][*path*][*filename*[.*ext*]] [*d*:][*path*][*filename*[.*ext*]]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before COMP to specify the drive and path that contains the COMP command file.

[*d*:][*path*][*filename*[.*ext*]] to specify the first set of filenames that you want to compare. It is also referred to as the *primary* file.

[*d*:][*path*][*filename*[.*ext*]] to specify the second set of filenames that you want to compare. It is also referred to as the *secondary* file.

Notes:

1. The files that you compare may be on the same drive or on different drives. They can also be in the same directory or different directories.

2. Global filename characters are allowed in both filespecs, and cause all files matching the first filespec to be compared with the corresponding files from the second filespec. Thus, typing:

```
A>comp a:*.asm b:*.bak
```

   causes each file that has an extension of .ASM from the current directory of drive A to be compared with a file of the same name (but with an extension of .BAK) from the current directory of drive B.

3. If no parameters are specified with the COMP command, or if the second parameter is missing, you are prompted for them. If either parameter contains only a drive or a path with no filename, COMP assumes a filename of *.*. You can enter a complete path with either of the two filenames.

4. The paths and names of the files being compared are displayed as the comparing process proceeds. An error message will follow if a file matching the second filespec cannot be found, or the files are different sizes, or a specified directory path is invalid.

5. It is possible to compare all files in one directory with all corresponding files in another directory.

6. If no file matches the primary filename, COMP will prompt you for both the primary and secondary parameters again.

7. During the comparison, an error message appears for any location that contains mismatching information in the two files. The message indicates the offset into the files of the

COMP (Compare Files) Command

mismatching bytes, and the contents of the bytes themselves (all in hexadecimal), as follows:

```
Compare error at OFFSET XXXXXXXX
File 1 = XX
File 2 = XX
```

In this example, FILE1 is the first filename typed; FILE2 is the second filename typed.

After ten unequal comparisons, COMP concludes that further comparing would be useless; processing ends; and the following message is displayed:

```
10 Mismatches - ending compare
```

After a successful comparison, COMP displays:

```
Files compare OK
```

After the comparison of the two files ends, comparing will proceed with the next pair of files that match the two filenames, until no more files can be found that match the first parameter. Then COMP displays:

```
Compare more files (Y/N)?_
```

You now have the option to compare two more files or to end the comparison. If you want to compare two more files, enter **y**. You will be prompted for new primary and secondary filenames.

If you want to end COMP processing, type **n**. You will return to the DOS prompt.

8. In all compares, COMP looks at the last byte of the files being compared to assure that it contains a valid end–of–file mark (Ctrl–Z, which is the hexadecimal character 1A). If found, no action is taken by COMP. If the end–of–file mark is *not* found, COMP produces the message:

   ```
   EOF mark not found
   ```

   This is done because some products produce files whose sizes are always recorded in the directory as a multiple of 128 bytes, even though the actual usable data in the file is usually a few bytes less than the directory size. In this case, COMP may produce **compare error** messages when comparing the few bytes beyond the last real data byte in the last block of 128 bytes (COMP always compares the number of bytes reflected in the directory). Thus, the **EOF mark not found** message indicates that the compare errors may not have occurred in the usable data portion of the file.

9. The two sets of files you want to compare can have the same path and filenames—provided they are on different drives.

10. If you only specify a drive for the second file, it is assumed that the second filename is the same as the first filename.

11. A comparison does not take place if the file sizes are different.

12. COMP does not wait for you to insert a diskette containing a file to be compared. Therefore, if a file to be compared is not on the same diskette as the COMP command itself, you should type

COMP (Compare Files) Command

COMP with no parameters. When COMP prompts for the filenames, you can insert the desired diskette and reply with the name of the file to be compared.

Example: The following example compares all .ASM files on drive B with the files on drive C with the same filenames.

```
A>comp b:*.asm c:
```

The following example compares all files in the directory A:\LEVEL1 with corresponding files in the directory A:\LEVEL2.

```
A>comp a:\level1 a:\level2
```

COPY Command

Purpose: Copies one or more files to the specified disk.

Format: COPY [/A][/B][*d*:][*path*]*filename*[.*ext*][/A][/B]

[*d*:][*path*][*filename*[.*ext*]][/A][/B][/V]

or

COPY [/A][/B][*d*:][*path*]*filename*[.*ext*][/A][/B]

[+[*d*:][*path*]*filename*[.*ext*][/A][/B]...]

[*d*:][*path*][*filename*[.*ext*]][/A][/B][/V]

or

COPY [/A][/B][*d*:][*path*]*filename*[.*ext*][/A][/B]

[+[[,,]*d*:][*path*]*filename*[.*ext*][/A][/B]...]

[*d*:][*path*][*filename*[.*ext*]][/A][/B][/V]

Type: Internal External
***

Remarks: The first file specified is the source file. The second file specified is the target file. If the second parameter is a directory (*path* with no filename), files are copied into that directory without changing their names.

> **Note:** COPY is not the same as BACKUP. Use BACKUP if you want all files including subdirectories. COPY only copies files from the current or specified directory.

COPY Command

COPY also copies files to the same disk. In this case, you *must* give the copies different names unless different directories are specified; otherwise, the copy is not permitted. Concatenation (combining of files) can be performed during the copying process.

You can also use COPY to transfer data between any of the system devices. An example of how to copy information that you type at the keyboard to a file is provided at the end of the description of COPY Option 2.

Specify **/V** to cause DOS to verify that the sectors written on the target diskette are recorded properly. Although errors in recording data are very rare, this option has been provided for those of you who wish to verify that critical data has been correctly recorded. This option causes the COPY command to run more slowly, due to the additional overhead of verification.

The **/V** parameter provides the same check as does the VERIFY ON command. **/V** is redundant if the VERIFY ON command has been executed previously. The difference is that **/V** is effective only during the duration of the COPY command. The VERIFY ON command is in effect until VERIFY OFF is entered.

The parameters **/A** and **/B** indicate the amount of data to be processed by the COPY command. Each applies to the filespec preceding it and to all remaining filespecs on the command line until another **/A** or **/B** is encountered. These parameters have the following meanings:

When used with a *source* filespec:

/A Causes the file to be treated as an ASCII (text) file. The file's data is copied up to, but not including, the first end–of–file character (Ctrl–Z, which is 1AH) found in the file the remainder of the file is not copied.

/B Causes the entire file (based on the directory file size) to be copied.

When used with a *target* filespec:

/A Causes a Ctrl–Z character to be added as the last character of the file.

/B Causes no end–of–file character (Ctrl–Z) to be added.

The default values are **/A** when concatenation is being performed (see Option 3 below), and **/B** when concatenation is not being performed (Options 1 and 2).

Notes:

1. When copying to or from a reserved device name, the copy is performed in ASCII (**/A**) mode. The first Ctrl–Z character encountered will end the copy unless **/B** was specified.

2. If you make a copy of a file that is marked read–only, the copy will not be marked read–only.

3. You cannot use COPY to transfer a file using the COM or AUX serial ports.

COPY Command

You can use the global characters ? and * in the filename and in the extension parameters of both the source and target files. If you type a ? or * in the source *filespec*, the names of the files will be displayed as the files are being copied. For more information about global characters, refer to "Global Filename Characters" in Chapter 2.

The COPY command has three format options:

Option 1 - Copy with Same Name

Use this option to copy a file with the target file having the *same* filename and extension as the source file. For example:

COPY [*d:*][*path*]*filename*[.*ext*]

or

COPY [*d:*][*path*]*filename*[.*ext*] *d:*[*path*]

In the first example, we want to copy a file to the current directory of the default drive. In the second example, we specify the target drive and/or directory. In both examples, because we did not specify the second filename, the copied file will have the same filename as the source file. Because we did not specify a name for the second file, the source drive and the target drive must be different unless different directories were specified or implied; otherwise, the copy is not permitted.

For example, assume the default drive is A. The command:

```
A>copy b:myprog
```

COPY Command

copies the file MYPROG from drive B to the current directory on the default drive A, with no change in the filename. The command:

```
A>copy *.* b:
```

copies all the files in the current directory from the default drive A to drive B, with no change in the filenames or in the extensions. The filenames are displayed as the files are copied. This method is very useful if the files on drive A are fragmented. The command:

```
A>copy b:\myprog    b:\level1
```

copies the file MYPROG from the root directory of drive B to the directory path:

```
\level1
```

on the same drive. The copy has the same filename as the original file. Note that the above example assumes that directory \LEVEL1 exists on drive B. If it did not, then the file MYPROG would have been copied into a file named LEVEL1 in the root directory of drive B. In other words, if the second parameter specifies a directory that exists, the file (or files) will be placed in that directory, keeping the same filename. If the second parameter does not specify a directory that exists, DOS will treat it as a filename.

COPY Command

Option 2 - Copy with Different Name

Use this option when you want the copied file to have a different name from the file being copied. For example:

COPY [*d*:][*path*]*filename*[.*ext*]
[*path*]*filename*[.*ext*]

or

COPY [*d*:][*path*]*filename*[.*ext*]
d:[*path*]*filename*[.*ext*]

In the first example, we copied a file (first file specified), and renamed the copy (second file specified). We did not specify a drive, so the default drive was used. In the second example, we copied a file and renamed the copy also. In this example, we did specify the target drive. Because we changed the name of the file, the source drive and the target drive do not have to be different. The current directory can be the same or different.

For example:

```
A>copy myprog.abc b:*.xxx
```

Copies the file MYPROG.ABC from the diskette in the default drive (drive A in this example) to drive B, naming the copy MYPROG.XXX. The current directory of each drive was used.

You can also use reserved device names for the copy operation. For example:

COPY Command

```
copy con fileA
copy con aux
copy con lpt1
copy filea con
copy fileb aux
copy filec lpt2
copy aux lpt1
copy aux con
```

Also, NUL can be used in any variation.

Refer to "Reserved Device Names" in Chapter 2 for information about system devices.

This example shows how to use COPY to put what you type from the keyboard into a file:

```
A>copy con filea

Type a line and press Enter.
Type your next line and press Enter.
.
.
.
Type your last line and press Enter.
Now, press F6 and then press Enter.
```

When you press F6, and then press Enter, the COPY operation ends and saves the information you entered. In this example, the information is saved in a file named FILEA.

> **Note:** This example assumes that you have not altered the meaning of F6 through the "Extended Screen and Keyboard Control" functions described in Chapter 3 of the *DOS Technical Reference*. If you have, then substitute the key that you have assigned Ctrl–Z for F6 in this example.

COPY Command

Option 3 - Copy and Combine Files

Use this option when you want to combine files while copying. That is, you can combine two or more files into one file by adding the additional files to the end of the first. The date and time recorded in the result file directory are the current date and time. The message indicating the number of files copied refers to the number of result files created.

To combine files, list any number of source files, separated by plus (+) signs in the COPY command. Use the following format:

COPY [/A][/B][*d*:][*path*]*filename*[.*ext*][/A][/B]

[+[*d*:][*path*]*filename*[.*ext*][/A][/B]...]

[*d*:][*path*][*filename*[.*ext*]][/A][/B][/V]

For example:

```
A>copy a.xyz+b.abc+b:c.txt bigfile.txt
```

This command creates a new file called BIGFILE.TXT on the default drive. The combination of A.XYZ, B.ABC, and B:C.TXT is put into BIGFILE.TXT.

If you do not specify a result *filename*, the additional files are added to the end of the first file, leaving the result in the first file. For example,

```
A>copy a.asm+b.asm
```

In this case, COPY appends B.ASM to the end of A.ASM and leaves the result in A.ASM.

> **Note:** Combining files is normally performed in text (or ASCII) mode. That is, the first Ctrl–Z (hex 1A) character in the file is interpreted as an end–of–file mark. To combine binary files, use the **/B** parameter to force COPY to use the physical end–of–file (the file length shown in the DIR command).

You can also combine ASCII and binary files by using the following parameters:

- ASCII - /A
- Binary - /B

For example,

```
A>copy a.xyz+b.com/b+b:c.txt/a bigfile.txt
```

A **/A** or **/B** takes effect on the file it is placed after, and it applies to all subsequent files on the command line until another **/A** or **/B** is found. A **/A** or **/B** on the result file causes a Ctrl-Z to be added (/A), or not to be added (/B), as the last character in the result file.

You can use the global characters ? and * in the filenames of both the files to be combined and the result file. For example:

```
A>copy  *.lst combin.prn
```

In this example, all files matching *.LST are combined into one file called COMBIN.PRN:

```
A>copy  *.lst+*.ref combin.prn
```

COPY Command

This example combines all files matching *.LST and then all files matching *.REF into one file called COMBIN.PRN:

```
A>copy *.lst+*.ref *.prn
```

In this example, each file matching *.LST is combined with the corresponding .REF file, with the result having the same name but with extension .PRN. Thus, a file FILE1.LST would be combined with FILE1.REF to form FILE1.PRN; XYZ.LST would be combined with XYZ.REF to form XYZ.PRN; etc. Note that in this case (when multiple files are to be created), only one file from each of the source filespecs is used to create a given target file.

For more information about global characters, refer to "Global Filename Characters" in Chapter 2.

It is easy to enter a COPY command to combine files where one of the source files is the same as the target, yet this often cannot be detected. For example:

```
A>copy *.lst all.lst
```

This would produce an error if ALL.LST already existed. The error would not be detected, however, until it was time for ALL.LST to be appended; by this time, ALL.LST could already have been altered.

COPY handles this situation as follows: As each input file is found, its name is compared with the target filename. If the names are the same, that one input file is skipped, and the following message is displayed on the screen:

Content of destination lost before copy

Further copying proceeds normally. This allows *summing* files, with a command like:

```
A>copy all.lst + *.lst
```

This command appends all .LST files, except ALL.LST itself, to ALL.LST. In this case, the error message is suppressed, because this is a true *physical append* to ALL.LST.

The following are special cases. Remember to use the **/B** parameter whenever you use the plus (+) sign with non–ASCII files:

```
A>copy b:xyz.asm+
```

This command copies the file XYZ.ASM to the default drive and gives it a new date and time. To simply change the date and time, leaving the file in place, you can use the following command:

```
A>copy b:xyz.asm+,, b:
```

In these special cases, if global filename characters are used in the filename or extension, then all of the matching files will be appended together into the first filename that matches. Thus, the command:

```
A>copy b:*.*+,, b:
```

will not update the dates and times of all files on drive B, but will append all of drive B's files together into a single file that will replace the first file found on drive B.

COPY Command

Note: When combining files, COPY considers the copying process to be successful if at least one, but not necessarily all, of the named source files is found. If none of the source files can be found, you receive the message

```
0 file(s) copied
```

CTTY (Change Console) Command

Purpose: Changes the standard input and output console to an auxiliary console, or restores the keyboard and screen as the standard input and output devices.

Format: CTTY *device–name*

Type: Internal External
***

Remarks: Specify the parameter:

device–name to define the device to use as the primary console. Specify AUX, COM1, or COM2 to use that device as the primary console. Specify CON to reset the primary standard input and output devices to the primary console.

Notes:

1. The CTTY command accepts the name of *any* character–oriented device to allow you to install your own device drivers, and to specify their device names. You must be certain that the named device is capable of both input and output operations. For example, you should not specify the name of a printer, because DOS will attempt to read from that device.

2. The CTTY command is effective only for programs that use DOS function calls. Other programs, such as BASIC (that do not use DOS function calls), are not able to use the CTTY command to change the standard input and output devices. Therefore if you load BASIC

CTTY (Change Console) Command

while you are using CTTY, the standard input and output console are reset to the keyboard and screen.

Example: The following example causes DOS to use the AUX device for its standard input and output operations:

```
A>ctty aux
```

The following example reverses the previous assignment, causing DOS to switch back to the standard screen and keyboard for its operations:

```
A>ctty con
```

DATE Command

Purpose: Use to enter or change the date known to DOS. The date is recorded in the directory when you create or change a file.

Format: DATE [*mm-dd-yy*] | [*dd-mm-yy*] | [*yy-mm-dd*]

Type: Internal External
***

Remarks: Specify the parameters:

mm to specify the month. Type one or two numbers from 1 to 12 for the month.

dd to specify the date. Type one or two numbers from 1 to 31 for the day.

yy to specify the year. Type two numbers between 80 and 99 or four numbers between 1980 and 1999 for the year.

Notes:

1. If you type DATE with no parameters, the following prompt is displayed:

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

   The format for the date (mm-dd-yy or dd-mm-yy or yy-mm-dd) depends on the country you selected with the SELECT command. You also can change the format for the date by creating a CONFIG.SYS file that

contains the COUNTRY command. Refer to Chapter 4, "Configuring Your System" for more information on the COUNTRY command.

2. Separate the parts of the date with a dash (-), a slash (/) or a period (.). For example, to set the date to September 23, 1984, type:

```
9-23-84
```

or

```
9/23/84
```

or

```
9.23.84
```

3. If you type a valid date, the new date is accepted and the DOS prompt is displayed. If the date is invalid, the following prompt is displayed:

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

4. DOS displays the day of the week (Tue for example) for information purposes only. Do not include the day of the week when you type the date.

5. You can change the date from the standard input device or from a batch file. Remember, when you start the system, it does not prompt you for the date if you use an AUTOEXEC.BAT file. You can include a DATE command in that file. For more information about the AUTOEXEC.BAT file, refer to "Batch File Commands" in this chapter.

DATE Command

6. To leave the date as–is, press Enter.

7. If you are using an IBM Personal Computer AT, typing a new date does not change the system clock. Refer to your *GTO* for information on the system clock.

8. Leaving your computer on for more than 24 hours with no activity may cause the date to be incorrectly updated.

Example: The following example changes the date to July 24, 1984.

```
A>date
Current date is Mon  1-18-1984
Enter new date: 7/24/84_
```

DEL (Delete) Command

Purpose: Deletes the specified file.

Format: DEL [*d*:][*path*]*filename*[.*ext*]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:] to specify the drive that contains the file you want to delete.

[*path*] to specify the directory path that contains the file you want to delete.

filename[.*ext*] to specify the name of the file you want to delete.

Notes:

1. If the drive specifier is not specified, the default drive is assumed.

2. If the path is not specified the current directory is assumed.

3. If the filename is not specified, *.* is assumed and all files are deleted.

4. You can use the global filename characters ? and * in the filename and extension. However, use global filename characters with caution because multiple files can be deleted with a single command.

DEL (Delete) Command

5. If you use the filespec *.* to delete all the files on a disk, the following message is displayed to verify that you actually want to delete all files:

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

   Type **y** (yes) and press Enter if you *do* want to delete all the files on the disk.

   Type **n** (no) and press Enter if you do not want to erase all files on the disk.

6. You cannot delete files that are marked as read–only.

7. You cannot use DEL to delete a subdirectory. To delete a subdirectory, use the RMDIR (remove directory) command.

8. Extra care should be exercised when using DEL after using ASSIGN, JOIN, or SUBST.

Example: The following example deletes the file name FILE.BAT from the diskette in drive A.

```
A>del a:file.bat
```

The following example deletes all files from the directory \LEVEL1 on drive C.

```
A>del c:\level1
```

DIR (Directory) Command

Purpose: Lists either all the directory entries, or only those for specified files.

> **Note:** Directory entries for hidden system files such as IBMBIO.COM and IBMDOS.COM are not listed, even if present.

Format: DIR [*d*:][*path*][*filename*[.*ext*]][/P][/W]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*][*filename*[.*ext*]] to specify the file whose directory you want to list.

/P to pause the display when the screen is full. The following prompt is displayed:

Strike a key when ready . . .

To continue press any key.

/W to display the information in a wide display format. Only the filenames and directory names are shown. This parameter is only recommended for 80-column displays.

Notes:

1. The information provided in the directory listing includes the volume identification and the amount of free space left on the disk. The freespace amount is rounded up to the nearest

1024K bytes. The display line for each file includes its size in decimal bytes and the date and time the file was last written to.

> **Note:** If you set the COUNTRY configuration command to a country other than U.S., the date and time format displayed may be different. The examples in this section show the date displayed for the U.S. date and time format.

2. Entries that name other directories are clearly identified with <DIR> in the file size field.

3. You can use the global characters ? and * in the filename and extension parameters. For more information about the global characters, refer to "Global Filename Characters" in Chapter 2.

4. If you do not specify a filename extension, the default is *.

5. To display the directory entry for a file that does not have an extension, type the filename followed by a period. In this case, the *.ext* does not default to *.

6. The DIR command has two format options (the **/P** and **/W** parameters may be used with either option):

   - List All Files

     or

   - List Selected Files

DIR (Directory) Command

Example:

Option 1 - List All Files

The following example lists all the directory entries on the default drive.

```
A>dir
```

The directory may look like this:

```
 Volume in drive A is MYDISK
 Directory of  A:\

FILE1     A        10368       7-20-83  12:13p
FILE3     A         1613       5-27-83  12:14p
9X                    31       8-17-82  10:59a
LEVEL2          <DIR>          9-09-82  12:10p
FILE1               2288       9-02-82  12:25p
        5 File(s) 141312 bytes free
```

The following example lists all the directory entries for current directory of drive C.

```
A>dir c:
```

The following example lists all the directory entries for the directory path \LEVEL2:

```
A>dir \level2
```

The screen will look like this:

DIR (Directory) Command

```
Volume in drive A is MYDISK
Directory of  A:\LEVEL2

 .           <DIR>        9-09-84   1:30p
 ..          <DIR>        9-09-84   2:45p
MYPROG   COM      2463     7-30-84   8:55a
         3 File(s)   141312 bytes free
```

Note that all files in directory LEVEL2 have been listed, including the two special entries found in all subdirectories. The entry marked with a single period denotes the directory being listed (LEVEL2), and the double period denotes this directory's parent directory (in this case, the root directory). Thus, if your *current* directory is LEVEL2 and you wish to see the files in its parent directory, you can enter:

```
A>dir ..
```

Option 2 - List Selected Files

The following example lists the directory entry of the file named FILE3.A in the current directory of the default drive.

```
A>dir file3.a
```

the screen may look like this:

```
 Volume in drive A is MYDISK
 Directory of  A:\

FILE3    A      1613     5-27-84  12:14p
        1 File(s) 141312 bytes free
```

DIR (Directory) Command

If you type:

```
A>dir *.a
```

the screen may look like this:

```
 Volume in drive A is MYDISK
 Directory of  A:\

FILE1     A     10368       7-20-84   12:13p
FILE3     A      1613       5-27-84   12:14p
        2 File(s) 141312 bytes free
```

If you type:

```
A>dir file1
```

the screen may look like this (omission of *.ext* defaults to *):

```
 Volume in drive A is MYDISK
 Directory of  A:\

FILE1     A     10368       7-20-84   12:13p
FILE1           2288       9-02-84   12:25p
        2 File(s)  141312 bytes free
```

To display only the entry for a file that has no extension, enter the filename followed by a period. In this case, the *.ext* does *not* default to *. For example,

```
A>dir file1.
```

DISKCOMP (Compare Diskettes Only) Command

Purpose: Compares the contents of the diskette in the first specified drive to the contents of the diskette in the second specified drive.

Notes:

1. This command is used only for comparing diskettes. If a fixed disk drive letter is specified, an error message is displayed.

2. This command compares two *entire diskettes*; the COMP command compares two *files*.

Format: [*d*:][*path*]DISKCOMP [*d*: [*d*:]] [/1] [/8]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before DISKCOMP to specify the drive and path that contains the DISKCOMP command file.

[*d*:] to specify the source drive.

[*d*:] to specify the target drive.

/1 to compare only the first side of the diskettes, even if the diskettes and drives are dual-sided.

/8 to compare only 8 sectors per track, even if the first diskette contains 9/15 sectors per track.

DISKCOMP (Compare Diskettes Only) Command

You can specify the same drive or different drives in this command. If you specify the same drive, a one–drive comparison is performed. You are prompted to insert the diskettes at the appropriate time. DISKCOMP waits for you to press any key before it continues.

DISKCOMP compares all tracks on a track–for–track basis and issues a message if the tracks are not equal. The message indicates the track number and the side (0 or 1) where the mismatch was found.

After completing the comparison, DISKCOMP prompts:

```
Compare more diskettes (Y/N)?_
```

If you type **y**, the next comparison is done on the same drives that you originally specified, after you receive prompts to insert the proper diskettes.

To end the command, type **n**. If the following message is displayed, insert the DOS diskette in drive *x* and press any key when ready.

```
Insert disk with \COMMAND.COM in drive A
and strike any key when ready
```

Notes:

1. If you omit both parameters, a one–drive comparison is performed on the default drive.

2. If you omit the second parameter, the default drive is used as the secondary drive.

3. On a one–drive system, all prompts are for drive A, regardless of any drive specifiers entered.

DISKCOMP (Compare Diskettes Only) Command

4. DISKCOMP usually does not issue a **Diskettes compare OK** message if you try to compare a backup diskette created by the COPY command with the diskette you copied from. The COPY operation produces a copy that contains the same information, but may place the information at different locations on the target diskette from those locations used on the source diskette. In this case, you should use the COMP command to compare individual files on the diskettes.

5. If a diskette error occurs while DISKCOMP is reading the diskette, a message is produced that indicates where (track and side) the error occurred. Then DISKCOMP continues to compare the rest of the diskette. Because the remainder of the data to be compared cannot be read correctly from the indicated track and side, you can expect to receive a **Compare error** message.

6. DISKCOMP automatically determines the number of sides and sectors per track to be compared, based on the diskette that is to be read first (the first drive parameter entered).

   If the first diskette or drive can be read on only one side, or if the **/1** parameter is used, only the first side is read from both diskettes. If the first diskette contains 9 sectors per track, then DISKCOMP will compare 9 sectors per track unless you used the **/8** parameter. If the first diskette contains 15 sectors per track, then DISKCOMP will compare 15 sectors per track. If the first drive and diskette are double-sided, and **/1** is not specified, a two–sided comparison

DISKCOMP (Compare Diskettes Only) Command

is done. In this case an error message is produced if either the second drive or the diskette is a single–sided diskette.

7. The source and target drives cannot be virtual drives, such as those created by the SUBST command.

8. DISKCOMP does not recognize assigned drives.

9. DISKCOMP should not be used while a JOIN is in effect.

10. DISKCOMP does not work on network drives.

DISKCOMP (Compare Diskettes Only) Command

DISKCOMP Compatibility

This section describes the possible diskette combinations that you may use with DISKCOMP:

If your computer has single-sided diskette drives, you can use DISKCOMP to compare:

- A single-sided diskette to a single-sided diskette

If your computer has double-sided diskette drives, you can use DISKCOMP to compare:

- A single-sided diskette to a single-sided diskette
- A double-sided diskette to a double-sided diskette

If your computer has high-capacity diskette drives, you can use DISKCOMP to compare:

- A single-sided diskette to a single-sided diskette
- A double-sided diskette to a double-sided diskette
- A high-capacity diskette to a high-capacity diskette

No other combinations are allowed. If you specify an invalid combination, the following message is displayed:

```
Drive types (double, single-sided) or
diskette types not compatible
```

DISKCOMP (Compare Diskettes Only) Command

Example: The following example compares the diskette in drive A to the diskette in drive B.

```
A>diskcomp a: b:
```

The following example compares the first diskette in drive A to the second diskette in drive A. Notice, that by not specifying the source and target drive, a one–drive compare is performed. You have to switch diskettes as prompted.

```
A>diskcomp
```

The following example compares the first diskette in drive B to the second diskette in drive A.

```
A>diskcomp b:
```

DISKCOPY (Copy Diskettes Only) Command

Purpose: Copies the contents of the diskette in the source drive to the diskette in the target drive. The target diskette is formatted if necessary, during the copy.

Notes:

1. This command is used only for copying diskettes. If a fixed disk drive letter is specified, an error message is displayed.

2. This command copies two *entire diskettes*; the COPY command copies two *files*.

Format: [*d*:][*path*]DISKCOPY [*d*: [*d*:]][/1]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before DISKCOPY to specify the drive and path that contains the DISKCOPY command file.

[*d*:] to specify the source drive.

[*d*:] to specify the target drive.

/1 to copy only the first side of the diskette, regardless of the diskette or drive type.

You can specify the same drives or different drives. If the drives are the same, a one-drive copy operation

DISKCOPY (Copy Diskettes Only) Command

is performed. You are prompted to insert the diskettes at the appropriate times. DISKCOPY waits for you to press any key before continuing.

After copying, DISKCOPY prompts:

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

If you type **y**, the next copy is done on the same drives that you originally specified, after you are prompted to insert the proper diskettes.

To end the command, type **n**. If the following message is displayed, insert the DOS diskette in drive *x* and press any key when ready.

```
Insert diskette with \COMMAND.COM in drive A
and strike any key when ready
```

Notes:

1. If the target diskette has not been formatted with the same number of sides and sectors per track as the source diskette, DISKCOPY will format the target diskette during the copy operation.

2. If you omit both drive parameters, a one-drive copy operation is performed on the default drive.

3. If you omit the second parameter, the default drive is used as the target drive.

4. If you omit the second parameter and you specify the default drive as the source drive, a one–drive copy operation is performed.

5. On a one–drive system, all prompts will be for drive A, regardless of any drive letter you may enter.

6. Diskettes that have had a lot of file creation and deletion activity become *fragmented*, because diskette space is not allocated sequentially. The first free sector found is the next sector allocated, regardless of its location on the diskette.

   A fragmented diskette can cause degraded performance due to excessive head movement and rotational delays involved in finding, reading, or writing a file.

   If this is the case, it is recommended that you use the COPY command, instead of DISKCOPY, to eliminate the fragmentation.

   For example, place a freshly formatted diskette in drive B, and the diskette you wish to copy in drive A. The command:

   ```
   A>copy a:*.* b:
   ```

   copies all the files from the diskette in drive A to the diskette in drive B. The resultant files (in drive B) are now copied sequentially. You should get better performance when you use these files from now on.

7. You can run DISKCOMP after a successful DISKCOPY to verify that the diskettes are identical.

8. If disk errors are encountered on either diskette, DISKCOPY indicates the drive, track, and side in error and proceeds with the copy. In this

case, the target diskette (copy) may or may not be usable, depending on whether the affected diskette location was to contain valid data.

9. DISKCOPY automatically determines the number of sides and sectors per track to copy, based on the source drive and diskette. If only the first side of the source diskette can be read, then only the first side can be copied. If the source drive and diskette are double-sided, both sides can be copied (unless you override it with the **/1** parameter). In this case, if the target drive is single-sided, an error message will indicate that the drives are incompatible.

   If the source diskette has ever been physically formatted with 9 sectors per track, then all 9 sectors on each track will be copied.

10. The source and target drive cannot be a virtual drive, such as those created by the SUBST command.

11. DISKCOPY does not recognize an assigned drive.

12. Double-sided diskettes written on in a high-capacity diskette drive will not be reliably read in a single- or double-sided diskette drive.

13. DISKCOPY should not be used while a JOIN is in effect.

14. DISKCOPY does not work with network drives.

DISKCOPY (Copy Diskettes Only) Command

DISKCOPY Compatibility

This section describes the possible diskette combinations that you may use with DISKCOPY:

If your computer has single-sided diskette drives, you can use DISKCOPY to copy:

- a single-sided diskette to a single-sided diskette

If your computer has double-sided diskette drives, you can use DISKCOPY to copy:

- a single-sided diskette to a single-sided diskette
- a double-sided diskette to a double-sided diskette

If your computer has high-capacity diskette drives, you can use DISKCOPY to copy:

- a single-sided diskette to a single-sided diskette*
- a double-sided diskette to a double-sided diskette*
- a high-capacity diskette to a high-capacity diskette

*Caution: You may not be able to reliably read these diskette types in a single- or double-sided diskette drive.

DISKCOPY (Copy Diskettes Only) Command

No other diskette combinations are allowed. If you specify an invalid combination, the following message is displayed:

```
Drive types (double- or single-sided) or
diskette types not compatible
```

If you want to copy the contents of a double-sided diskette onto a high-capacity diskette, use the COPY *.* command. Refer to the COPY command in this chapter for more information.

Example: The following example copies the diskette in drive A to the diskette in drive B.

```
A>diskcopy a: b:
```

The following example copies the source diskette in drive A to the target diskette in drive A. Notice, that by not specifing the source and target drive, a one-drive copy is performed. You must switch diskettes as prompted.

```
A>diskcopy
```

The following example copies the source diskette in drive B to the target diskette in drive B. Notice that by not specifing the source and target drive, a one-drive copy is performed. You must switch diskettes as prompted.

```
B>diskcopy
```

ERASE Command

Purpose: Erases the specified file.

Format: ERASE [*d*:][*path*]*filename*[.*ext*]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:] to specify the drive that contains the file you want to erase.

[*path*] to specify the directory path that contains the file you want to erase.

filename[.*ext*] to specify the name of the file you want to erase.

Notes:

1. If the drive specifier is not specified, the default drive is assumed.

2. If the path is not specified the current directory is assumed.

3. If the filename is not specified, *.* is assumed and all files are erased.

4. You can use the global filename characters ? and * in the filename and extension. However, use global filename characters with caution because multiple files can be erased with a single command.

ERASE Command

5. The operating system files IBMBIO.COM and IBMDOS.COM cannot be erased.

6. If you use the filespec *.* to erase all of the files on a disk, the following message is displayed to verify that you actually want to erase all files:

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

   Type **y** (yes) and press Enter if you *do* want to erase all the files on the disk.

   Type **n** (no) and press Enter if you do not want to erase all files on the disk.

7. You cannot erase files that are marked as read–only.

8. You cannot use ERASE to erase a subdirectory. To delete a subdirectory you need to use the RMDIR (remove directory) command.

9. Extra care should be exercised when using ERASE after using ASSIGN, JOIN, or SUBST.

Example: The following example erases the file name FILE.BAT from the diskette in drive A.

```
A>erase a:file.bat
```

The following example erases all files from the directory \LEVEL1 on drive C.

```
A>erase c:\level1
```

EXE2BIN Command

Purpose: Converts .EXE files to .COM or .BIN files.

Format: [*d*:][*path*]EXE2BIN [*d*:][*path*]*filename*[.*ext*]
[*d*:][*path*][*filename*[.*ext*]]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before EXE2BIN to specify the drive specifier and path that contains the EXE2BIN command file.

[*d*:][*path*]*filename*.[*ext*] to specify the input file. If you *do not* specify:

[*d*:] the default drive is assumed

[*path*] the current directory is assumed

[.*ext*] .EXE is assumed

[*d*:][*path*]*filename*.[*ext*] to specify the output file. If you *do not* specify:

[*d*:] the drive of the input file is assumed

[*path*] the current directory is assumed

filename the input filename is assumed

[.*ext*] .BIN is assumed

EXE2BIN Command

The input file is converted to .COM file format (memory image of the program) and placed in the output file.

The input must be in valid .EXE format as 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 the specified initial CS:IP:

- If CS:IP is not specified in the program (the .EXE file contains 0:0), a pure binary conversion is assumed. If segment fixups are necessary (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.

  In this case, the resultant program is usable only when loaded at the absolute memory address specified by a user application. The DOS command processor will not be capable of properly loading the program.

- If CS:IP is specified as 0000:100H, it is assumed that the file is to be run as a .COM file, with the location pointer set at 100H by the assembler statement ORG. No segment fixups are allowed, as .COM files must be segment relocatable, that is, they must assume the entry conditions explained in Chapters 5 of the *DOS Technical Reference*. Once the conversion is complete, you may rename the resultant file to a .COM extension. Then, the command processor is capable of loading and executing the program in the same manner as the .COM programs supplied on your DOS diskette.

If CS:IP does not meet one 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 .EXE file.

To produce standard .COM files with the assembler, you must both use the assembler statement ORG to set the location pointer of the file at 100H and specify the first location as the start address. (This is done in the END statement.) Also, the program must not use references that are defined only in other segments. For example, with the IBM Personal Computer MACRO Assembler:

```
ORG 100H
START:
.
.
.
END START
```

EXE2BIN resides on your DOS Supplemental Program diskette.

FDISK Command

See Chapter 3, "Preparing Your Fixed Disk."

FIND Filter Command

Purpose: Sends all lines from the specified filenames that contain the specified string to the standard output device.

Format: [*d*:][*path*]FIND [/V][/C][/N]"*string*"[[*d*:][*path*]*filename*[.*ext*]...]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before FIND to specify the drive specifier and path that contains the FIND command file.

/V to display all lines *not* containing the *string*.

/C to display a count of the number of lines containing *string*. If /C is specified with /V or /N, then FIND ignores /V and /N.

/N to display the relative line number of each matching line ahead of the line from the file.

Enclose the string in double quotes ("). Two quotes in succession are taken as a single quote. An uppercase string does not match a lowercase string.

Global filename characters are not allowed in the filenames or extensions.

Example: The following example displays all lines from BOOK1.TXT, BOOK2.TXT, and BOOK3 (in that order) that contain the string "Fool's Paradise."

```
A>find "Fool's Paradise" book1.txt book2.txt book3
```

FIND Filter Command

The following example displays the names of all files in the current directory of drive B that *do not* contain the string "DAT."

```
A>dir b: ¦ find /v "DAT"
```

The following example displays all the subdirectory entries in the current directory (redirects the output of the DIR command to the FIND filter and then displays the directory entries that contain DIR).

```
A>dir ¦ find "<DIR>"
```

For the following examples, assume the file PROG contains the following lines:

```
This is a
beautiful
day for a picnic.
```

The following example displays the lines in the file PROG that *do not* contain the string "f."

```
A>find /v"f"prog
```

The result is:

```
----------a:prog
This is a
```

FIND Filter Command

The following example displays the relative line number of the lines that contain the string "f."

```
A>find /n"f"prog
```

The result is:

```
----------a:prog
[2]beautiful
[3]day for a picnic.
```

The following example displays a count of the number of lines in the file PROG that contain the string "a."

```
A>find /c" a "a:prog
```

The result is:

```
----------a:prog:2
```

FORMAT Command

Purpose: Initializes the disk in the designated or default drive to a recording format acceptable to DOS; analyzes the entire disk for any defective tracks; and prepares the disk to accept DOS files by initializing the directory, File Allocation Table, and system loader.

> **CAUTION**
> Please note that formatting destroys all data on the disk. Because of this, you should be very careful before you decide to format any disk, particularly a fixed disk. If you attempt to format your fixed disk, please note that the entire contents of any previously created DOS partition, including all subdirectories and their contents, are destroyed. If you are not certain which drive is the default drive, do NOT enter FORMAT without a drive letter. If you do not specify a drive letter, you could unintentionally destroy all of the data on whatever disk happens to be the default. It's safer to specify a drive letter.

Format: [*d*:][*path*]FORMAT [*d*:][/S][/1][/8][/V][/B][/4]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before FORMAT to specify the drive specifier and path that contains the FORMAT command file.

[*d*:] to specify the drive that contains the disk you want to format.

/S to copy the operating system files from the DOS diskette in the default drive to the new disk in the following order:

```
IBMBIO.COM
IBMDOS.COM
COMMAND.COM
```

If the system does not reside on the default drive, and the default drive is non–removable, FORMAT prompts you to put a system diskette in diskette drive A. If the system is not on the default drive and the default drive is removable, FORMAT prompts you to insert a system diskette in the default drive.

/1 to format a diskette for single-sided use regardless of the drive type.

/8 to format a diskette for 8 sectors per track. FORMAT defaults to 9/15 sectors per track usage if you do not specify **/8**. Note that format always creates 9/15 physical sectors on each diskette track, but that it instructs DOS to use only 8 sectors per track if you use the **/8** parameter.

/V to give the disk a volume label. We strongly recommend that you use the **/V** parameter. This uniquely identifies each disk.

The volume label cannot be used in place of filenames as input to any of the DOS commands. The volume label is for your use in keeping track of your disks.

/B to format a diskette for 8 sectors per track diskette with space allocated for the IBMBIO.COM and IBMDOS.COM system modules. It does not place the system modules or the command processor on the diskette. This parameter is used to create a

FORMAT Command

diskette on which any version of DOS can be placed through that version's SYS command. If the **/B** parameter is not used, only DOS Version 3.10 can be placed on the diskette through the SYS command.

/4 to format a double-sided diskette in a high-capacity drive.

> **CAUTION**
> This parameter is intended to allow use of double-sided diskettes in high-capacity drives. However, the diskettes formatted with the /4 parameter specified may not be reliably read or written in a single- or double-sided drive.

The following table shows which parameters are valid for certain diskette types:

Disk Type	Parameters Allowed
320KB	/S, /V, /1, /8, /B, /4
1.2MB	/S, /V
fixed disk	/S, /V

FORMAT Command

Notes:

1. All new diskettes and fixed disks must be formatted before they can be used by DOS. Refer to Chapter 3 for more information before formatting your fixed disk.

2. A fixed disk must also be formatted again if you change the size of its DOS partition through the FDISK command.

3. Formatting destroys any previously existing data on the disk.

4. During the formatting process, any defective tracks are marked as *reserved* to prevent the tracks from being allocated to a data file.

5. Directory entries for IBMBIO.COM and IBMDOS.COM are marked as *hidden files*, and therefore, they do not appear in any directory searches—including the DIR command.

6. FORMAT will prompt you to enter a volume label (volume identification) if you have used the /V parameter. The label can consist of from 1 to 11 characters. All characters acceptable in filenames are acceptable in the volume label. Unlike filenames, however, the volume label does not contain a period between the eighth and ninth characters.

   You can add or change a volume label using the LABEL commmand. For more information refer to the LABEL command in this chapter.

7. FORMAT produces a status report, that indicates:

   - Total disk space
   - Space marked as defective
   - Space currently allocated to the DOS system files (when **/S** is used)
   - Amount of space available for your files

8. FORMAT determines the target drive type and formats the disk or diskette accordingly. For diskettes, if the diskette can be successfully read and written on only one side, the diskette is formatted for single-sided use, 8 sectors per track; it can be used in either type of drive. If the target drive is double-sided and you do not use the **/1** parameter, the diskette is formatted for double-sided use; it will not be usable in a single-sided drive.

9. Fixed disks are already physically formatted (proper recording format) when shipped by IBM. When formatting a fixed disk, FORMAT checks all locations within the DOS partition, but does not physically format them again.

10. If the **/S** parameter is used and the system has insufficient available memory for FORMAT to load all three system modules, it will load as many modules as it can, format the target disk, and write the modules that are in memory. It must then read the remaining modules from the source disk so they can be placed on the target disk. If the source diskette has been removed

from the drive, an appropriate message will prompt you to reinsert it before FORMAT can continue.

11. The parameters /8 and /V cannot be specified with the /B parameter.

12. If you specify the /S parameter, the system files are copied from the default drive. If the default drive is a fixed disk drive that does not contain the system files, then you are prompted to insert the DOS diskette in drive A.

13. FORMAT ignores any drive reassignments (ASSIGN).

14. FORMAT should not be used with drives involved in a JOIN or substitution (SUBST).

15. FORMAT does not work on network drives.

FORMAT Compatibility

The following table shows the possible diskette combinations that may be used with FORMAT:

Drive Type	Diskette Type
single sided	single-sided diskettes
double sided	single-sided or double-sided diskettes

FORMAT Command

Drive Type	Diskette Type
high capacity	single-sided*, double-sided* or high-capacity diskettes

* To format a single-sided or double-sided diskette in a high-capacity drive, refer to the FORMAT/4 command in this chapter.

No other combinations are allowed.

Example: By issuing the following command, the diskette in drive B is formatted and the operating system files are also copied:

```
A>format b:/s/v
```

The system displays the following message:

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

After you insert the appropriate diskette and press ENTER, the system displays this message:

```
Formatting...
```

while the diskette formatting is taking place.

Once the formatting is complete, the system displays this message:

FORMAT Command

```
Formatting...Format complete
System transferred

Volume label (11 character, ENTER for none)? mydisk

    xxxxxx bytes total disk space
     xxxxx bytes used by system
    xxxxxx bytes available on disk

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

In the above example, note that MYDISK was typed as the volume label.

Type **y** and press Enter to format another diskette.

Type **n** and press Enter to end the FORMAT program.

When you format a fixed disk, you will see the following message instead of the prompt to insert a diskette:

```
Warning, All Data on Non-Removable
Disk Drive x: Will Be Lost!
Proceed with Format (Y/N)?
```

The *x* is replaced by the drive letter you typed. If you want to format your fixed disk, type **y** and press Enter. If you do not want to format your fixed disk, type **n** and press Enter.

Fixed disk formatting can take several minutes because of the large size that can be allocated to DOS, so don't be alarmed if it takes some time before you are prompted for the volume label. You can tell that FORMAT is working by noting that your fixed disk drive light is on.

GRAFTABL (Load Graphics Table) Command

Purpose: Loads a table of additional character data for the color/graphics adapter into memory.

Format: [*d*:][*path*]GRAFTABL

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before GRAFTABL to specify the drive and path that contains the GRAFTABL command file.

Use GRAFTABL to display foreign language characters when in graphics mode on the color/graphics adapter. GRAFTABL loads a table of data in memory which defines these additional characters for the color/graphics adapter to use. This allows the ASCII characters 128 thru 255 to be displayed when using the Color/Graphics adapter in graphics mode. This command increases the resident size of DOS in memory.

- After loading the character table and initializing the interrupt vector, GRAFTABL responds with the message:

  ```
  GRAPHICS CHARACTERS LOADED
  ```

  and then the DOS prompt is displayed.

GRAFTABL (Load Graphics Table) Command

- GRAFTABL should only be invoked once each time DOS is started. If you try to load GRAPHICS once it is already loaded, the following message is displayed:

```
GRAPHICS CHARACTERS ALREADY LOADED
```

Example: The following example loads the table of graphics characters:

```
A>graftabl
```

The system now supports display of ASCII characters 128 through 255 in the graphics mode on the Color/Graphics Adapter.

GRAPHICS (Screen Print) Command

Purpose: Allows the contents of a graphics display screen to be printed on an IBM Personal Computer Printer when using a color/graphics monitor adapter.

Format: [*d*:][*path*]GRAPHICS [*printer type*] [/R][/B]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before GRAPHICS to specify the drive and path that contains the GRAPHICS command file.

[*printer type*] to specify the type of printer you are using. You can choose from the following type:

COLOR1 - IBM Personal Computer Color Printer with black ribbon

COLOR4 - IBM Personal Computer Color Printer with RGB (red, green, blue, black) ribbon

COLOR8 - IBM Personal Computer Color Printer with CMY (cyan, magenta, yellow, black) ribbon

COMPACT - IBM Personal Computer Compact Printer

GRAPHICS - IBM Personal Graphics Printer

If you do not specify a printer type, the default is the GRAPHICS printer.

GRAPHICS (Screen Print) Command

/R to print black (as seen on the monitor) and white (as seen on the monitor) on the printer. If you do not specify /R, the default is to print black as white and white as black.

/B to print the background color. This parameter is only for printer types COLOR4 and COLOR8. If you do not specify /B, the default is not to print the background color.

This command increases the resident size of DOS in memory.

Press the Shift–PrtSc keys to print the screen contents on the printer. If the screen is in text mode, the text is printed in less than 30 seconds. If the screen is in the graphics mode, (modes 4, 5, 6 only) each time the PrtSc key is pressed, the following happens:

- In the 320x200 color graphics mode with printer types GRAPHICS and COLOR1, the screen contents are printed in up to four shades of gray.

- In the 640x200 color graphics mode, the screen is printed sideways on the paper. The upper right corner of the screen is printed on the upper left corner of the paper.

- Printing may take as long as three minutes.

- To invoke the screen print from an assembly language program, use the following coding example:

```
PUSH BP
INT  5
POP  BP
```

GRAPHICS (Screen Print) Command

Example: The following example loads graphics support needed to print the screen contents on a COLOR8 printer. The background colors are printed and black prints as black and white as white.

```
A>graphics color8 /b /r
```

JOIN Command

Purpose: Connects a drive to a directory on another drive to produce a single directory structure from two separate directories.

Format: [*d*:][*path*]JOIN

or

[*d*:][*path*]JOIN *d*: *d*: \ *directory*

or

[*d*:][*path*]JOIN *d*: /D

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] to specify the drive and path that contains the JOIN command file, if it is not in the current directory of the default drive.

d: to specify the drive to be connected to a directory on another drive.

d:\ *directory* to specify the directory that you will join a drive under. The directory must be at the root and only one level deep.

If the directory you specify in the path does not exist, it is created on the drive you specify. If the directory does exist, it must be empty. An empty directory only contains the entries . and .. For example, if the directory of C:\LEVEL1 is empty, the following is displayed if you type **dir c:\level1**

JOIN Command

```
Volume in drive C is EMPTY
Directory for C:\LEVEL1

.     <DIR>              1-01-84    12:03.00 AM
..    <DIR>              1-01-84    12:03.00 AM
   2 Files(s)  1048956 bytes free
```

/D to disconnect a join. You must specify the drive letter of the drive whose join you want to delete. For example, if you joined drive B to the directory on drive C:\JOINB, to disconnect the directories, type:

```
A>join b: /d
```

Notes:

1. The message **Directory not Empty** is displayed if the directory you try to JOIN is not empty.

2. The path you specify cannot be the current directory. If you try to JOIN to the current directory, the message **Invalid parameter** appears.

3. The message **Invalid drive specification** is displayed if you refer to a joined drive. For example, if you join drive A, you cannot refer to the drive letter A until you remove the JOIN (/D parameter). This means you cannot JOIN the default drive because you will then be on an invalid drive.

4. The directory path you specify cannot be the root directory (\).

5. Type JOIN with no parameters specified to display the drives and directories that are currently joined. The information displayed indicates the drive that is joined and the path the

drive is joined to. For example, the following message indicates that drive **A** is joined to the directory path **C:\LEVEL1** .

```
A: => C:\LEVEL1
```

6. The entire tree (starting from the root) of the specified drive is joined, regardless of the current directory of that drive.

7. JOIN does not allow network drives for either parameter. The message **Cannot JOIN a network drive** appears if you attempt to do so.

8. Unpredictable results occur if the drive being JOINed is part of a substitution (SUBST) or reassignment (ASSIGN).

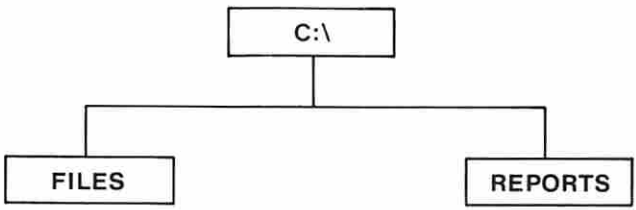
9. BACKUP, RESTORE, FORMAT, DISKCOPY, and DISKCOMP should *not* be used while a JOIN is in effect.

Example: For the following examples, assume:

- The default drive is drive B
- The current directory of drive C is C:\

Joining a Drive to a Directory Path

Assume the directory structure of drive C looks like this:

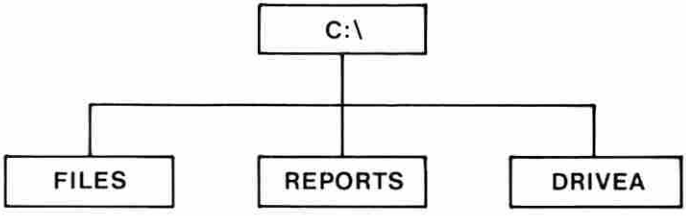


JOIN Command

You can join drive A to the path C:\DRIVEA by typing:

```
B>join a: c:\drivea
```

The directory structure of drive C now looks like this:



Note: Notice that the directory DRIVEA was created on drive C by the JOIN command.

If you type **dir c:**, the following is displayed:

```
Volume in drive C is FIXEDC
Directory for C:\

FILES        <DIR>      8-21-84    2:10.00 PM
REPORTS      <DIR>      9-23-84    5:55.00 PM
DRIVEA       <DIR>      9-30-84    12:03.00 AM
   3 Files(s)  1048956 bytes free
```

If you type **dir a:**, the message **Invalid drive specification** is displayed because you cannot refer to a joined drive. If you enter **dir c:\drivea** , the directory of the disk in drive A: is displayed. The "**bytes free**" displayed are those of C:.

JOIN Command

Displaying the Current Joins

The following example displays the current joins from the previous example.

```
B>join
```

The following is displayed:

```
A: => C:\DRIVEA
```

Deleting a Join

The following example deletes the join from the previous example.

```
B>join a: /d
```

Now you can refer to drive A because the join is removed.

Why Use JOIN?

JOIN can be used to help applications take advantage of fixed and virtual disks. The following example shows a way that files on A: can be accessed through C:

```
D>join a: c:\datafils.dir
D>cd c:\datafils.dir
D>path=c:\
D>dataprog c:
```

KEYBxx (Load Keyboard) Command

Purpose: Loads a keyboard program that replaces the keyboard program resident in ROM BIOS to support non-U.S. English keyboards. The *xx* in the command represents one of the five keyboard programs provided on the DOS diskette. Each command increases the resident size of DOS in memory by a different amount.

> **Note:** The keyboard programs provided on DOS 1.10 diskettes are not compatible with DOS 3.10 and should not be executed when running under DOS 3.10.

Format: [*d*:][*path*]KEYBUK
or
[*d*:][*path*]KEYBGR
or
[*d*:][*path*]KEYBFR
or
[*d*:][*path*]KEYBIT
or
[*d*:][*path*]KEYBSP

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before KEYB*xx* to specify the drive and path that contains the KEYB*xx* command file.

KEYBxx (Load Keyboard) Command

These commands load a program into memory that replaces the ROM BIOS keyboard program.

- Only one keyboard program should be loaded after starting DOS. The program remains resident in memory until you perform a system reset or power off the computer. If you load a second keyboard program, it will gain control of the keystrokes you type. However, the earlier program will still be in memory and cannot be returned to.

- You can change from the KEYB*xx* program to the U.S. keyboard format and back again at any time by holding down the Ctrl and Alt keys and pressing F1 to change to U.S. format or Ctrl–Alt–F2 to return to the selected memory-resident keyboard program.

  Note: If you enter a KEYB*xx* command on a computer with multilingual keyboard support in the ROM BIOS, the program simply sets the current ROM keyboard to whichever one is appropriate and then exits leaving nothing resident. Therefore, you cannot press Ctrl–Alt–F1 to return to a U.S. keyboard on a machine with ROM multilingual keyboard support. The reason for this is that the KEYB*xx* program does not remain resident in memory. Refer to your *Guide to Operations* for more information on how to change the formats on these types of computers.

- The copy of the DOS diskette you created with the SELECT command, loads one of these programs when you start DOS. Refer to Chapter 4 of the *DOS User's Guide* for information on the SELECT command.

KEYBxx (Load Keyboard) Command

- The table below lists which keyboard program supports each keyboard. The memory allocated to each when it is loaded is approximately 2K bytes.

Command	Keyboard Type
KEYBUK	United Kingdom
KEYBGR	Germany
KEYBFR	France
KEYBIT	Italy
KEYBSP	Spain

Non-U.S. English keyboards have some keys with characters used by programmers. To use these characters, press and hold the Ctrl and Alt keys and press the appropriate character key.

You can get accented characters through the use of "dead keys," that is, keys that do not produce characters unless they are used in combination with another key. To "build" an accented character, press and release the accent key and then press the appropriate letter key. To use the accent character by itself, press the accent key and then press the space bar.

KEYBxx (Load Keyboard) Command

Note: If the dead-key/letter-key combination is not allowed (not in the following table), you see the accent followed by the character key you pressed, and a beep sounds. If you press the diaeresis, an invalid second key produces a box.

Allowable Dead Key Combinations

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

LABEL (Volume Label) Command

Purpose: Allows you to create, change or delete a volume label on a disk.

Format: [*d:*][*path*]LABEL [*d:*][*volume label*]

Type: Internal External
***

Remarks: Specify the parameters:

[*d:*][*path*] before LABEL to specify the drive and path that contains the LABEL command file.

[*d:*] to specify the drive letter of the disk you want to label. If you do not specify a drive letter, the default drive is assumed.

[*volume label*] to specify the volume label. Volume labels are used to identify a disk. They can be up to 11 characters and are in the same format as volume labels created by FORMAT/V. If you do not specify a volume label, you are prompted with the following messages:

```
Volume in drive X is xxxxxxxxxxx

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

To give the disk a volume label, type the label you want and press Enter.

To change an existing volume label, type the new volume label and press Enter. The new label you typed replaces the existing volume label.

LABEL (Volume Label) Command

To delete a volume label, do not specify a volume label; just press Enter. Then you are prompted:

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

Type **y** and press Enter. The volume label on the disk is deleted.

If you type more than 11 characters for the volume label, only the first 11 characters are used.

Example: **Creating a Volume Label**

The following example creates a volume label called ADDRESS on the diskette in drive A.

```
A>label a:address
```

The following example gives the fixed disk C the label FIXEDISKC.

```
A>label c:fixediskc
```

Changing a Disk's Volume Label

The following example changes the volume label of the diskette in drive A from ADDRESS to PROGRAMS.

```
A>label a:
```

Then you are prompted:

```
Volume in drive A is ADDRESS

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

LABEL (Volume Label) Command

Type:

```
programs
```

Then press Enter.

Deleting a Volume Label

The following example deletes the volume label PROGRAMS from the diskette in drive A.

```
A>label a:
```

You are prompted:

```
Volume in drive A is PROGRAMS

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

Press Enter. The following prompt is displayed:

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

Type **y** and then press Enter. The volume label for drive A is deleted.

> **Note:** LABEL should not be used with SUBSTed drives. The *root* directory of the actual drive will be the target of LABEL. LABEL should not be used with ASSIGNed drives or to label network drives.

MKDIR (Make Directory) Command

Purpose: Creates a subdirectory on the specified disk.

Format: MKDIR [*d*:]*path*

or

MD [*d*:]*path*

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:] to specify the drive letter of the disk you want to create the subdirectory on. If you do not specify a drive, the default drive is assumed.

[*path*] to specify the path of directory names. Subdirectory names are in the same format as filenames. All characters that are valid for a filename are also valid for a directory name. For more information on paths, refer to Chapter 5, "Tree-Structured Directories" in this manual.

> **Note:** You can create as many subdirectories as you wish, limited only by available disk space. However, you should ensure that the maximum length of any single path from the root directory to the desired level is 63 characters, including imbedded backslashes.

Each directory can contain file and directory names that also appear in other directories. In other words, two or more files or directories can have the same name, as long as they are defined in separate directories.

MKDIR (Make Directory) Command

Example: The following example creates the subdirectory called LEVEL1 under the root directory.

```
A>md \level1
```

The following example creates the subdirectory LEVEL2 under the subdirectory LEVEL1.

```
A>md \level1\level2
```

If the current directory is \LEVEL1, the following example creates the subdirectory LEVEL2 under the subdirectory LEVEL1.

```
A>md level2
```

The last example does the same thing as the second. Note that in the second example, the first \ tells DOS to begin its directory search with the *root* directory. The absence of a leading \ in the last example causes DOS to begin at the *current* directory.

> **Note:** Care should be taken when making directories while an ASSIGN, JOIN, or a SUBST is in effect.

MODE Command

Purpose: Sets the way that a printer, a Color/Graphics monitor adapter, or an Asynchronous Communications Adapter operates.

Format: [*d*:][*path*]MODE LPT#[:][*n*][,[*m*][,P]]

or

[*d*:][*path*]MODE *n*

or

[*d*:][*path*]MODE [*n*],*m*[,T]

or

[*d*:][*path*]MODE
COM*n*[:]*baud*[,*[parity]*[,*[databits]*[,*[stopbits]*[,P]]]]

or

[*d*:][*path*]MODE LPT#[:]=COM*n*

Type: Internal External
***

Remarks: A missing or invalid *n* or *m* parameter means that the mode of operation for that parameter is not changed.

> **Technical Note:** When used in Option 1 (P option), Option 2 (shifting the screen (R)ight or (L)eft), Option 3 (P option), or Option 4, the MODE command causes printer and Asyncronous Communications Adapter intercept code to be resident in memory. This increases the resident size of DOS in memory.

MODE Command

The resident portion is common for all three operations that cause it to be loaded. Once loaded, invoking another option causing residency does not cause any additional code to become resident.

Notes:

1. Before you can use MODE to redirect parallel printer output to a serial device, you must initialize the Asynchronous Communications Adapter by using Option 3 (see below). If that serial device is a printer, your serial initialization command should also include the **P** parameter.

2. MODE LPT#[:][*n*][,*m*] disables the redirection for the printer designated by the #. Redirection causes a portion of MODE to remain resident.

3. Because LPT#[:] and COM[:] are DOS device names, you can specify them with or without the colon (:). For example, you can specify:

```
A>mode lpt1: 132,8

or
A>mode lpt1 132,8

or
A>mode lpt1=com1
```

MODE Command

The MODE command has four format options:

Option 1 (For the printer)

MODE LPT#[:][*n*][,[*m*][,P]]

where:
is 1, 2, or 3 (the printer number)
n is 80 or 132 (characters per line)
m is 6 or 8 (lines per inch vertical spacing)
P specifies continuous retry on time-out errors

For example:

```
A>mode lpt1:132,8
```

sets the mode of operation of printer number 1 to 132 characters per line and 8 lines per inch vertical spacing. The power-on default options for the printer are 80 characters per line and 6 lines per inch. If the printer is reset or initialized, the default values are set (BASIC initializes the printer).

If you specify an invalid **n** or **m** value, the values are ignored and the previous value is unchanged.

The retry loop can be stopped by pressing Ctrl-Break. To stop time-out errors from being continuously retried when you have entered **P**, you must use MODE Option 1 without specifying **P**.

MODE Command

Option 2 (For switching Display Adapters, and setting the display mode of the Color/Graphics Monitor Adapter)

MODE *n*

or

MODE [*n*],*m*[,T]

where:

n is 40, 80, BW40, BW80, CO40, CO80, or MONO

40 sets the display width to 40 characters per line (for Color/Graphics Monitor Adapter).

80 sets the display width to 80 characters per line (for Color/Graphics Monitor Adapter).

BW40 switches the active display adapter to the Color/Graphics Monitor Adapter, and sets the display mode to Black and White (disables color) with 40 characters per line.

BW80 switches the active display adapter to the Color/Graphics Monitor Adapter, and sets the display mode to Black and White (disables color) with 80 characters per line.

CO40 switches the active display adapter to the Color/Graphics Monitor Adapter, enables color, and sets the display width to 40 characters per line.

MODE Command

CO80 switches the active display adapter to the Color/Graphics Monitor Adapter, enables color, and sets the display width to 80 characters per line.

MONO switches the active display adapter to the Monochrome Display Adapter (which always has display width of 80 characters per line).

m is **R** or **L** (shift display right or left)

T requests a test pattern used to align the display

For readability, you can shift a display connected to a color/graphics monitor adapter 1 character (for 40 columns) or 2 characters (for 80 columns) in either direction. If you specify **T** in the MODE command, a prompt asks you if the screen is aligned properly. If you type **Y** the command ends. If you type **N** the shift is repeated followed by the same prompt. For example,

```
A>mode 80,r,t
```

sets the mode of operation to 80 characters per line and shifts the display 2 character positions to the right. The test pattern is displayed to give you the opportunity to further shift the display without having to enter the command again.

> **Note:** Shifting the display on an IBM Personal Computer AT causes all of MODE's resident code to be loaded.

MODE Command

Option 3 (For Asynchronous Communications Adapter)

MODE COM*n*[:]*baud*[,*parity*[,*databits*[,*stopbits*[,P]]]]

where:

n	Either 1 or 2 (Asynchronous Communications Adapter number)
baud	110, 150, 300, 600, 1200, 2400, 4800, or 9600
	Note: Only the first 2 characters are required; subsequent characters are ignored.
parity	Either N (none), O (odd), or E (even)—(default = E)
databits	Either 7 or 8 (default = 7)
stopbits	Either 1 or 2 (if baud equals 110, default = 2; if baud does not equal 110, default = 1)

These are the *protocol* parameters. They are used to initialize the Asynchronous Communications Adapter. When you specify the protocol, you must specify at least the baud rate. The other parameters can be omitted, with the defaults accepted, by entering only commas. For example,

```
A>mode com1:12,n,8,1,p
```

sets the mode of operation to 1200 baud rate, no parity, 8 databits, and 1 stopbit. To use the defaults listed in the definitions above, you enter:

```
A>mode com1:12,,,,p
```

The *parity* defaults to even, the *databits* defaults to seven, and the *stopbits* defaults to one.

The **P** option indicates that the asynchronous adapter is being used for a serial interface printer. If you enter the **P**, time-out errors are continuously retried. You can stop the retry loop by pressing Ctrl-Break. To stop the time-out errors from being continuously retried when you have entered **P**, you must reinitialize the asynchronous adapter without entering the **P**. P option causes a portion of MODE to remain resident.

Option 4 (To redirect parallel printer output to an Asynchronous Communications Adapter)

MODE LPT#[:]=COM*n*

where:

#	Either 1, 2, or 3 (printer number)
n	Either 1 or 2 (Asynchronous Communications Adapter number)

All output directed to printer LPT# is redirected to the asynchronous adapter *n*.

MORE Filter Command

Purpose: Reads data from the standard input device, and sends one screen of data to the standard output device, and then pauses with the message **--More--**.

Format: [*d*:][*path*]MORE

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before MORE to specify the drive specifier and path that contains the MORE command file.

Pressing any character key causes another screen of data to be sent to the standard output device. This process continues until all input data is read.

Example: The following example displays the contents of the file TEST.ASM one screen at a time. When the screen is full, the message **--More--** appears on the bottom line. You can press any key to see the next screen.

```
A>more <test.asm
```

The following example pipes the output of the TREE command to the MORE filter. This is useful because when a full screen of output from TREE is displayed, the output pauses and the message --More-- is displayed. When you want to continue with the display, press any key.

```
A>tree ¦ more
```

PATH (Set Search Directory) Command

Purpose: Searches specified directories for commands or batch files that were not found by a search of the current directory.

Format: PATH [[*d*:]*path*[[;[*d*:]*path*]...]]

or

PATH ;

Type: Internal External
***

Remarks: You may specify a list of drives and path names, separated by semicolons (note that path names must be specified and will not default to the current directory). Then, when you enter a command that is not found in the current directory of the drive that was specified (or implied) with the command, DOS searches the named directories in the sequence you entered them. The current directory is not changed.

Typing PATH with no parameters displays the current path. Typing PATH with only a semicolon (PATH ;) resets the search path to null (no extended search path). This is the default when DOS is started. In this case, DOS searches only the current directory for commands and batch files.

PATH (Set Search Directory) Command

Notes:

1. Erroneous information in the paths, such as invalid drive specifications or imbedded delimiters, will not be detected until DOS actually needs to search the specified paths.

2. If a path is specified that no longer exists at the time DOS uses it to search for a command or batch file, DOS ignores that path and goes on to the next.

3. PATH only finds files that can be executed; such as .COM, .EXE, and .BAT files. PATH will not find files with any other extensions.

4. A copy of the environment is saved with terminate and stay-resident programs. Invoking programs with a resident portion (MODE, PRINT, GRAPHICS) before a large path is set saves usable memory.

5. Terminate and stay-resident programs are loaded above the environment area so growth of the environment is limited to 128 bytes or the current size, whichever is greater.

Example: For the following examples, assume the program MYPROG.COM is only in directory MYDIR on drive B, and that the default drive is drive A.

The following example instructs DOS to look in the current directory of the specified drive, followed by A:\LEVEL2\LEVEL3, then B:\MYDIR for a command you specify.

```
A>path a:\level2\level3;b:\mydir
```

PATH (Set Search Directory) Command

If the command typed is *not* found in any of the directories specified in PATH, the following message is displayed:

Bad command or filename

In the previous example, if you type the command:

```
A>myprog
```

DOS searches the three specified directories, finding the program MYPROG in B:\MYDIR.

To display the current path, type:

```
A>path
```

The result is:

```
PATH=A:\LEVEL2\LEVEL3;B:\MYDIR
```

The following example causes DOS to search the current directory of C:, then D:\tools.

```
A>path c:.;d:\tools
```

PRINT Command

Purpose: Prints a queue (list) of data files on the printer while you are doing other tasks on the computer.

Format: [*d*:][*path*]PRINT [/D:*device*][/B:*buffsiz*][/U:*busytick*] [/M:*maxtick*][/S:*timeslice*] [/Q:*quesiz*] [/C][/T][/P][[*d*:][*path*][*filename*][.ext]...]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before PRINT to specify the drive and path that contains the PRINT command file.

/D:*device* to specify the print device. If not specified, the default device PRN is assumed.

Important: If you specify /D, it must be the first parameter.

/B:*buffsiz* to set the size in bytes of the internal buffer. The default value is 512 bytes. Increasing the value of **B** may enhance the performance of the PRINT command.

/Q:*quesiz* to specify how many print files you can have in the queue. The range of values is from 1 to 32 files. The default value is 10.

/S:*timeslice* to specify the time–slice value. The default is 8 time slices. The range of values is 1 to 255.

/U:*busytick* to specify the number of clock ticks that PRINT will wait for until the print device is

available. /U is called *busyticks*. The default value for busytick is 1. If PRINT waits longer than U busyticks, it gives up its time slice.

/M:*maxtick* to specify how many clock ticks PRINT can have to print characters on the print device. /M is called *maxticks*. The default value for maxtick is 2 maxticks. The range of values is from 1 to 255 maxticks. This parameter does not need to be specified each time you use PRINT. Specify it only the first time.

/T to set the terminate mode. All queued files are canceled from the print queue. If a file is currently being printed, the printing stops, a cancelation message is printed, the paper is advanced to the next page, and the printer's alarm sounds.

/C to set the cancel mode. Allows you to select which file or files to cancel. The preceding filename and all following filenames entered on the command line are canceled from the print queue until a **/P** is found on the command line, or until you press the Enter key.

/P to set the print mode. The preceding filename and all following filenames are added to the print queue until a **/C** is found on the command line, or until you press the Enter.

The parameters /D, /B, /Q, /S, /U, and /M can only be specified the first time you use PRINT. If you specify them again, the following message is displayed:

Invalid Parameters

You can enter more than one filename on the command line, each with appropriate parameters.

PRINT Command

Global filename characters * and ? are allowed in the filename. Once a file has been queued, you can change the current directory without affecting the printing of the files already in the print queue.

The first time this command is issued, it increases the resident size of DOS in memory.

If no parameters are specified and you press Enter, the files listed on the command line are queued for printing (/P is assumed).

If PRINT is typed with no parameters, PRINT displays the names of the files currently in the print queue.

If you did not specify the device name using /D the first time the PRINT command is executed after you start your system, the following message is displayed on the display screen:

```
Name of list device [PRN]:
```

This allows you to specify the output list device, LPT1, LPT2, LPT3, PRN, COM1, COM2, AUX, etc. The default is PRN, and will be selected if you press Enter.

> **Note:** Be sure the device you name is physically attached to your system; naming a nonexistent device will cause unpredictable system behavior.

The files are queued for printing in the order entered. After each file is printed, the printer paper is advanced to the next page. Any tab characters found are expanded with blanks to the next 8–column boundary.

PRINT Command

If PRINT encounters a disk error while attempting to read the file to be printed, PRINT will cause:

- The file currently printing to be canceled
- The disk error message to be printed on the printer
- The printer paper to be advanced to the next page and the alarm to be sounded
- The remaining files in the print queue to be printed

If the /T or /C parameters are used to cancel a file or files currently being printed:

- The printer alarm sounds.
- A file cancelation message prints on the printer. If /T, the following message prints:

  All files canceled by operator.

  If /C, the name of the canceled file is printed followed by this message:

  File canceled by operator

- The printer paper advances to the next page.
- If all files in the print queue have not been canceled, printing resumes with the first file remaining in the print queue.

PRINT Command

Notes:

1. The disk containing the files being printed must remain in the specified drive until all printing is complete. Any file in the print queue must not be altered or erased until after it has been printed.

2. The printer cannot be used for any other purpose while PRINT has data to print. Any attempt to use the printer (Shift–PrtSc, LLIST, Ctrl–PrtSc, LPRINT, etc.) results in an "out–of–paper" indication until all files have been printed or printing is terminated (/T).

3. You cannot use PRINT on a network server computer.

Example: In this example, the PRINT command is being used for the first time since the system was started. The command:

```
A>print a:temp1.tst
```

has just been entered, DOS responds with:

```
Name of list device [PRN]:
```

Press the Enter key to send output to the printer.

Then it adds the file TEMP1.TST from drive A to the print queue and sends the output to the device "PRN" printer. The command:

```
A>print /t
```

empties the print queue. Any other information on the line is ignored.

PRINT Command

The command:

```
A>print temp.*/c
```

removes all TEMP.??? files from the print queue that have the same drive letter as the default drive. The command:

```
A>print a:temp1.tst/c a:temp2.tst a:temp3.tst
```

removes the three files TEMP1, TEMP2, and TEMP3 on drive A from the print queue. The command:

```
A>print temp1.tst/c temp2.tst/p temp3.tst
```

removes file TEMP1.TST from the print queue, adds the files TEMP2.TST and TEMP3.TST to the print queue. The command:

```
A>print temp1.tst temp2.tst temp3.tst/c
```

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

PROMPT (Set System Prompt) Command

Purpose: Sets a new DOS prompt.

Format: PROMPT [*prompt–text*]

Type: Internal External
***

Remarks: Specify the parameter:

prompt–text to specify the text for the new system prompt. The prompt–text can contain special meta–strings that are in the form $c. The following are meta–strings that can be included in the prompt–text:

Where *c* is one of the following:

$ The $ character.
t The time.
d The date.
p The current directory of the default drive.
v The version number.
n The default drive letter
g The > character.
l The < character.
b The | character.
q The = character.
h A backspace; the previous character is erased.
e The ESCape character.
_ The CR LF sequence (go to beginning of new line on the display screen).

Any other value for *c* is treated as a null character and is ignored by PROMPT.

PROMPT (Set System Prompt) Command

Type PROMPT with no parameters to reset the prompt to the normal DOS prompt.

Example: The following example sets the DOS prompt to the default drive letter plus the character >.

```
A>prompt $n$g
```

The following example sets the DOS prompt to the message hello.

```
A>prompt hello
```

The following example sets the DOS prompt to the current directory of the default drive plus the > character.

```
A>prompt $p$g
```

> **Note:** Include this statement in an AUTOEXEC.BAT file so that every time you start DOS, the DOS prompt tells you what directory you are in (the current directory).

If the current directory of drive A is \LEVEL1, the DOS prompt would display:

```
A:\level1>
```

The following example sets the DOS prompt to display the date and time as follows:

Time = (current time)

Date = (current date)

```
A>prompt time = $t$_date = $d
```

PROMPT (Set System Prompt) Command

If you want to create a prompt that begins with any of the DOS command delimiters (such as semicolon or blank), you can precede that character with a null meta–string. In this case, the character is treated as the first character of the prompt, rather than as a delimiter between the word PROMPT and its parameter. For example:

```
A>prompt $a;abc
```

causes the PROMPT command to interpret the $A as a null character, because **A** is not one of the defined characters in the above list. All characters following the null character become the new system prompt.

You can use the PROMPT command to issue escape sequences to ANSI.SYS. If you started your system with the command, DEVICE =ANSI.SYS in your CONFIG.SYS file, the following example displays the current directory on the top row of the screen, and a minus - on the current line.

```
A>prompt $e[s$e[24A$p$e[u-
```

The **A** in the escape sequence must be uppercase. See Chapter 3, "Using Extended Screen and Keyboard Control" in the *DOS Technical Reference*, Version 3.10.

Notes:

1. A copy of the environment is saved with terminate and stay-resident programs. Invoking programs with a resident portion (MODE, PRINT, GRAPHICS) before a large path is set saves usable memory.

2. Terminate and stay-resident programs are loaded above the environment area so growth of the environment is limited to 128 bytes or the current size, whichever is greater.

RECOVER Command

Purpose: Recovers files from a disk that has a defective sector. You can recover the file that contains the bad sector (minus the data in the bad sector). Or, all the files on the disk can be recovered if the directory has been damaged.

Format: [*d*:][*path*]RECOVER [*d*:][*path*]*filename*[.*ext*]

or

[*d*:][*path*]RECOVER *d*:

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before RECOVER to specify the drive and path that contains the RECOVER command file.

[*d*:][*path*]*filename*[.*ext*] to specify the name of the file you want to recover.

Use the first RECOVER format to recover the specified file. If you do not specify a drive, the default drive is used. If you do not specify a path, the current directory is used. The size of the recovered file is a multiple of the DOS allocation unit size. In most cases, this is larger than the original file size. The data in the bad sector is not recovered.

Use the second RECOVER format to recover all files.

If the sum of the number of files in the root and the number of files in subdirectories is greater than the

possible number of entries in the root (112 for double-sided diskettes), you have to do multiple recovers to save the entire disk. Issue RECOVER using the second format to recover as many entries as will fit in the root directory. Examine the recovered files and copy the ones you want to save to another disk. Delete some or all of the files you have already recovered from the damaged disk from which you are recovering. Issue RECOVER again to recover more files. Continue this procedure until all files are recovered.

This form of the RECOVER command should only be used if the directory of the disk has become unusable. Because RECOVER has no way to know whether the data in the directory is valid or not, it *must* assume that the entire directory is invalid, and therefore recovers all files into filenames of the form shown below, including any files for which there may still have been valid directory entries.

Text files will normally require re–editing to remove unwanted data from the end of the recovered file before they can be used for normal processing.

> **Note:** You cannot use the RECOVER command on a network disk.

Example: The following example recovers the file MYPROG from the disk in drive A.

```
A>recover a:myprog
```

The disk file MYPROG on drive A is read sector-by-sector, skipping the bad sectors.

The following example recovers the contents of an entire disk from drive A.

RECOVER Command

```
A>recover a:
```

The disk file allocation table on drive A is scanned for chains of allocation units. A directory is created for each chain of allocation units in the form:

```
FILEnnnn.REC
```

Where *nnnn* is a sequential number starting with 0001. Each FILE*nnnn*.REC points to one of the recovered files on the disk.

RENAME (or REN) Command

Purpose: Changes the name of the file specified in the first parameter to the name and extension given in the second parameter.

Format: REN[AME] [*d*:][*path*]*filename*[.*ext*] *filename*[.*ext*]

Type: Internal External
***

Remarks: You can use the abbreviated form REN for the RENAME command. You can also use the global characters ? and * in the parameters. For more information about global characters, refer to "Global Filename Characters" in Chapter 2. A path can be specified only with the first filename; the file will remain in the same directory after its name has been changed.

Example: The following example renames the file ABODE on drive B to HOME.

```
A>rename b:abode home
```

The following example renames the file ABODE on drive B to ABODE.XY.

```
A>ren b:abode *.xy
```

The following example renames the file MYPROG.COM in the directory \LEVEL1 on drive B to YOURPROG.COM.

```
A>ren b:\level1\myprog.com yourprog.com
```

RESTORE Command

Purpose: Restores one or more backup files from a disk to another disk.

Format: [d:][path]RESTORE *d*:
[*d*:][*path*]*filename*[.*ext*][/S][/P]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before RESTORE to specify the drive and path that contains the RESTORE command file.

d: to specify the drive that contains the BACKUP files (source).

[*d*:][*path*]*filename*[.*ext*] to specify where you want to restore the files and what files from the source you want to restore.

/*S* to restore all files in subdirectories in addition to the files in the specified directory. This includes subdirectories at all levels beyond the specified directory.

/**P** to have RESTORE prompt you before restoring files that have changed since they were last backed up, or that are marked read–only. You can then choose to restore the file or not. Read–only is a file attribute that you can set by using the ATTRIB command. The two DOS system files (IBMBIO.COM and IBMDOS.COM) are marked read–only when they are created by the FORMAT and SYS commands.

The files being restored must have been placed on the source by the BACKUP command.

Files are restored to the current directory if you do not specify a path. If you specify a path, you must also specify a filename.

Global filename characters are allowed in the filename, and cause all of the files matching the filename to be restored.

When RESTORE prompts you to insert the source, make sure you insert the first diskette that might contain the file you want to restore. If you are not sure, insert source number 1. If the file is not on the diskette you inserted, RESTORE prompts you to insert the next target.

If you are sharing files, you can only restore files that you have access to. If you attempt to access a file that you do not have access to, the following message is displayed:

```
PATHNAME\FILENAME
Not able to restore at this time
```

If you use global filename characters, RESTORE prompts you to insert the next diskette after it has restored all files on the backup diskette that match the specified filename.

If you are restoring files that were backed up using a previous version of DOS, it is recommended that you use the RESTORE /P parameter. This will prompt you before restoring the DOS system files IBMBIO.COM and IBMDOS.COM. You should respond with a **N** to the prompt asking you if you want to restore the file. This will avoid overlaying the DOS 3.10 version of these files with versions

RESTORE Command

from previous levels of DOS. We recommend that you use the SYS command or FORMAT /S and then copy the current version on COMMAND.COM to the root directory before restoring if you want the disk to be bootable.

The RESTORE command sets the ERRORLEVEL (see Batch Commands) as follows:

0 Normal completion

1 No files were found to restore

2 Some files not restored due to sharing conflicts

3 Terminated by user (Ctrl–Break or ESC)

4 Terminated due to error

These codes can be used with the batch processing IF subcommand to control subsequent error level processing.

Example: The following example restores all files (including subdirectories /S) on the backup diskettes to fixed disk drive C:

```
A>restore a: c:\*.*  /s
```

The following example restores files from the backup disk in drive A that have a filename extension of .DAT to drive C.

```
A>restore a: c:*.dat
```

RESTORE Command

The following example restores three different files from the backup diskettes to the default fixed disk drive:

```
C>restore a: \level1\file1.dat
C>restore a: \level1\level2\file2.dat
C>restore a: \level1\level3\file3.dat
```

The following example restores all files from the backup disk drive C, and prompts you if any files on drive C have changed since the last backup or if any files are marked read–only (/P).

```
A>restore c:/p
```

Important: Do not RESTORE if a JOIN, ASSIGN, or SUBST was in effect during the BACKUP.

RMDIR (Remove Directory) Command

Purpose: Removes a subdirectory from the specified disk.

Format: RMDIR [*d*:]*path*

or

RD [*d*:]*path*

Type: Internal External
***

Remarks: The directory must be empty before it can be removed with the exception of the "." and ".." entries. You cannot remove subdirectories that contain hidden files. The last directory name in the path is the directory to be removed.

> **Note:** The root directory and the current directory cannot be removed.

Example: The following example removes the entry for LEVEL2 from the directory LEVEL1.

```
A>rd b:\level1\level2
```

> **Note:** Exercise care in using RMDIR while a JOIN or an ASSIGN is in effect. You cannot remove a directory if it has been substituted (SUBST).

SELECT Command

Purpose: Allows you to select the keyboard layout and the date and time format.

Format: [d:][path]SELECT *xxx yy*

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before SELECT to specify the drive and path that contains the SELECT command file.

xxx to specify the *country code*. The country code tells DOS the date and time format. It also tells DOS the currency symbol and the decimal separator. Choose the value for *xxx* from the table below. Other values can be found in the COUNTRY command.

yy to specify the *keyboard code*. The keyboard code tells DOS which keyboard layout you want to use. Choose the values for *xxx* and *yy* from the table below.

Country	Country Code	Keyboard Code
United States	001	US
France	033	FR
Spain	034	SP

SELECT Command

Country	Country Code	Keyboard Code
Italy	039	IT
United Kingdom	044	UK
Germany	049	GR

The SELECT command uses DISKCOPY to make a copy of your DOS diskette. It also creates the following two files on the copy of the DOS diskette:

- A CONFIG.SYS file that contains the COUNTRY = command
- An AUTOEXEC.BAT file that contains the KEYb*xx* command (except in the case of a U.S. keyboard)

Refer to Chapter 4 "Starting DOS" in the *DOS User's Guide* for more information.

SET (Set Environment) Command

Purpose: This command inserts strings into the command processor's environment. A copy of the entire series of strings in the environment is made available to all commands and applications.

Format: SET [*name*=[*parameter*]]

Type: Internal External
***

Remarks: The entire string (beginning with *name*) is inserted into a block of memory reserved for environment strings. Any lowercase letters in the name are converted to uppercase letters when added to the environment (including foreign language characters); the remainder of the line is inserted as you typed it. If the name already exists in the environment, it is replaced with the new *parameter*.

If the SET command is typed with no *name* specified, then the current set of environment strings is displayed.

If a *name* is specified, but the *parameter* is not specified, then the current occurrence of *name=parameter* is removed from the environment.

A copy of the environment (series of names and parameters) is made available to all DOS commands and applications (see "Program Segment Prefix" in Chapter 7 of the *DOS Technical Reference*).

SET (Set Environment) Command

Notes:

1. DOS automatically adds any PROMPT or PATH commands to the environment when you enter them. You do not need to use the SET command to add either of these two commands to the environment.

2. One of the strings in the environment (placed there by DOS when it starts up) will always be a COMSPEC = parameter. That parameter describes the path that DOS uses to reload the command processor when necessary.

3. If you have *not* loaded a program that remains resident (such as MODE, PRINT, or GRAPHICS), DOS expands the environment string area to hold additional strings. If you *have* loaded a program that remains resident, DOS is unable to expand the environment area beyond 127 bytes. If the environment area has already expanded beyond 127 bytes when you load a program that is to remain resident, DOS is unable to expand the environment area beyond that point. The message **Out of environment space** appears if you issue a SET command that would cause the combined environment strings to exceed 127 bytes.

4. A copy of the environment is saved with terminate and stay-resident programs. Invoking programs with a resident portion (MODE, PRINT, GRAPHICS) before a large environment is set saves usable memory.

SET (Set Environment) Command

Example: This example adds the string PGMS=\LEVEL1 to the environment. When an application program receives control, it could search the environment for the name PGMS, and use the supplied parameter as the directory name to use for its files:

```
A>set pgms=\level1
```

The following example removes PGMS=\LEVEL1 from the environment:

```
A>set pgms=
```

You can select the strings in the environment. For example, typing:

```
A>set abc=xyz
```

adds the string ABC=xyz to the other strings already in the environment (note the conversion of abc to uppercase ABC). In this way, it is possible for you to type keywords and parameters that are not meaningful to DOS, but can be found and interpreted by applications designed to examine the environment.

SHARE Command

Purpose: Loads support for file sharing.

> **Note:** Refer to Int 21H function calls 3DH and 5CH in Chapter 6 of the *DOS Technical Reference* for more information on file sharing and locking.

Format: [*d*:][*path*]SHARE [/F:*filspace*][/L:*locks*]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before SHARE to specify the drive and path that contains the SHARE command file.

/F:*filespace* to allocate file space in bytes for the area used for recording the information necessary for file sharing. Each open file requires the length of the full filename plus 11 bytes. The default value is 2048 bytes.

/L:*locks* to allocate space for the number of locks you want. The default value is 20 locks.

SHARE Command

If you load SHARE, all read or write requests are validated against the file sharing code. If filesharing is installed and you try to re-install it, the following message is displayed:

Share Already Installed

Example: The following example loads file sharing support.

```
A>share
```

Note: If you load SHARE the table for FCB control is checked. If you specified FCBS = 4,0 (the default) in CONFIG.SYS, the table is adjusted to 16,8.

SORT Filter Command

Purpose: Reads data from the standard input device, sorts the data, then writes the data to the standard output device.

Format: [*d*:][*path*]SORT [/R] [/+ *n*]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before SORT to specify the drive and path that contains the SORT command file.

/R to sort in reverse order. For example, Z comes before A.

/ +*n* to start sorting with column *n*. The *n* is an integer value. If you do not specify *n* the default is column 1. The maximum file that you can sort is 63K.

The output filename must be different than the input filename.

Notes:

Characters are sorted according to their binary value with the following exceptions:

1. Lowercase letters a-z are equated to uppercase A-Z.

2. Some characters above 127 are collated according to the following chart:

SORT Filter Command

ASCII value (decimal)	ASCII weight	ASCII value (decimal)	ASCII weight
128	67	**150**	85
129	85	**151**	85
130	69	**152**	89
131	65	**153**	79
132	65	**154**	85
133	65	**155**	36
134	65	**156**	36
135	67	**157**	36
136	69	**158**	36
137	69	**159**	36
138	69	**160**	65
139	73	**161**	73
140	73	**162**	79
141	73	**163**	85
142	65	**164**	78
143	65	**165**	78
144	69	**166**	166
145	65	**167**	167
146	65	**168**	63
147	79	**169**	169
148	79	**170**	170
149	79	**171**	171

SORT Filter Command

ASCII value (decimal)	ASCII weight	ASCII value (decimal)	ASCII weight
172	172	**175**	34
173	33	**225**	83
174	34		

Example: The following example reads the file UNSORT.TXT, sorts it in reverse order, and then writes the output to the file SORT.TXT.

```
A>sort /r <unsort.txt >sort.txt
```

The following example pipes the output of the DIR command to the SORT filter. Then the directory listing is sorted starting with column 14 (column 14 of a directory listing contains the file size). The output is sent to the standard input device.

```
A>dir ¦ sort /+14
```

SUBST(Substitute) Command

Purpose: Allows you to use a different drive specifier to refer to another drive or path.

> **Note:** If you use applications that do not recognize paths, SUBST allows you to specify a drive letter for a path.

Format: [*d*:][*path*]SUBST *d*: *d*:*path*

or

[*d*:][*path*]SUBST *d*: /D

or

[*d*:][*path*] SUBST

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] to specify the drive and path that contains the SUBST command file.

d: to specify the drive letter that you want to use to refer to another drive or path.

d:*path* to specify the drive or path that you want to refer to with a nickname. The path you specify should start from the root directory.

/D to delete a substitution. You must specify the drive letter of the drive whose substitution you want

SUBST(Substitute) Command

to delete. For example, if you substituted the drive G for the path C:\LEVEL1, to remove the substitution, type:

```
A>subst g: /d
```

Notes:

1. The first drive letter you specify depends on the value of the LASTDRIVE configuration command. If you do not specify the LASTDRIVE command in your CONFIG.SYS file, the default value is LASTDRIVE=E. This means that you can substitute the drive letters A through E. To specify a drive letter greater than E, set LASTDRIVE equal to a letter A - Z. For example, to SUBST the drive letter G, the LASTDRIVE command must be greater than or equal to G. Refer to Chapter 4 "Configuring Your System" for more information on the LASTDRIVE command.

   You can think of this drive letter as a "nickname" for a drive or path that you can access. For example, if you substitute the drive letter G for the path C:\LEVEL1\LEVEL2 to refer to C:\LEVEL1\LEVEL2\FILE1 you can type:

   ```
   A>dir g:file1
   ```

   Instead of typing:

   ```
   A>dir c:\level1\level2\file1
   ```

2. The drive specifiers for both drives you specify must be different.

SUBST(Substitute) Command

3. The first drive letter that you specify cannot be the same as the default drive.

4. If you specify a path, it is assumed that the path starts from the root directory.

5. Neither parameter can refer to a network drive. If you do refer to a network drive, the message "Cannot SUBST a network drive" appears.

6. The following commands should be used with care while a substitution is in effect: CHDIR, MKDIR, RMDIR, and PATH.

7. The following commands should *not* be used while a substitution is in effect: ASSIGN, BACKUP, DISKCOMP, DISKCOPY, FDISK, FORMAT, JOIN, LABEL, and RESTORE.

8. Type SUBST with no parameters to display the current substitutions. The output is displayed in this form:

```
G: => D:\PATH
```

9. The message **Invalid Parameter** is displayed if:

   You type an incorrect drive specifier

   You try to substitute the default drive

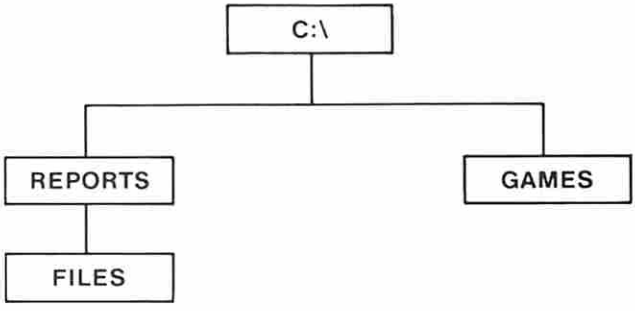
   You specify /D without specifying the drive letter

10. The message **Invalid Path** is displayed if you specify a path that doesn't exist.

SUBST(Substitute) Command

Example: For the following examples, assume:

- The default drive is drive B
- The current directory of drive C: is C:\REPORTS\FILES
- LASTDRIVE=H



Creating a Substitution

The following example substitutes drive G for the path C:\REPORTS\FILES

```
b>subst g:  c:\reports\files
```

If you want to list the directory of C:\REPORTS\FILES, you can type:

```
B>dir g:
```

Remember, the drive specifiers must be less than or equal to the value specified for LASTDRIVE in your CONFIG.SYS file. For example, in order for drive G to be valid, the LASTDRIVE command must be greater than or equal to g (LASTDRIVE = g).

SUBST(Substitute) Command

Displaying the Current Substitutions

The following example displays the current substitutions:

```
b>subst
```

The following is displayed:

```
G: =>  C:\REPORTS\FILES
```

The message indicates that drive G: is substituted for the path C:\REPORTS\FILES. Now when you refer to drive G, the directory C:\REPORTS\FILES is used.

SUBST(Substitute) Command

Deleting a Substitution

The following example removes the substitution as it appeared in the previous example:

```
subst g: /d
```

Why Use SUBST?

Substituting a drive letter for a path can be useful if you use applications that do not recognize paths. For example, if you want to edit the file RECIPES in the directory C:\FOOD\FILES, you can substitute the drive letter **F** for this directory path. Then when you want to edit RECIPES use the drive letter **F** instead of the path C:\FOOD\FILES. You can then refer to the file as F:RECIPES. Remember, you must have LASTDRIVE=F to do this substitution.

SYS (System) Command

Purpose: Transfers the operating system files IBMDOS.COM and IBMBIO.COM from the first drive specified to the second drive specified.

> **Note:** SYS does not transfer the file COMMAND.COM. You must copy this file into the root directory of the disk.

Format: [*d*:][*path*]SYS *d*:

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before SYS to specify the drive and path that contains the SYS command file.

d: to specify the disk drive that you want to transfer the operating system files to.

The directory of the disk in the specified drive must be completely empty, or the disk must have been formatted by a FORMAT *d*:/S or FORMAT *d*:/B command to contain directory entries for the DOS files IBMBIO.COM and IBMDOS.COM. This is necessary because DOS startup requires these files to occupy the first two directory entries, and because IBMBIO.COM must reside on consecutive sectors on the disk.

SYS (System) Command

Notes:

1. SYS lets you transfer a copy of the operating system files to an application program diskette designed to use DOS, but sold without it. In this case, the space required for the DOS files has already been allocated, although the DOS files are not actually present. The SYS command transfers the files to the allocated space.

2. You cannot use the SYS command on a network drive.

TIME Command

Purpose: Permits you to enter or change the time known to the system. Whenever you create or add to a file, the time is recorded in the directory. You can change the time from the console or from a batch file.

Format: TIME [*hh*:*mm*:[:*ss*[.*xx*]]]

Type: Internal External
***

Remarks: Specify the parameters:

hh to specify the hours. Type one or two numbers from 0 to 23 for the hours.

mm to specify the minutes. Type one or two numbers from 0 to 59 for the minutes.

ss to specify the seconds. Type one or two numbers from 0 to 59 for the seconds.

xx to specify the hundredths of seconds. Type one or two numbers from 0 to 99 for the hundredths of seconds.

Notes: If you type TIME with no parameters, the following prompt is displayed:

```
Current time is hh:mm:ss.xx
Enter new time:_
```

To leave the time as-is, press Enter. To change the time, type the new time and then press Enter.

1. Separate the hours, minutes, and seconds using a colon (:) or period (.). Separate the hundredths

of seconds using a period (.) or a comma (,), depending on the decimal separator that is displayed on the screen.

2. If you type a valid time, the new time is accepted and the DOS prompt is displayed. If the time is invalid, the following prompt is displayed:

```
Invalid time
Enter new time_
```

3. If you enter any information (for example, just the hours and minutes, and press Enter), the remaining fields are set to zero.

4. Any time is acceptable as long as the digits are within the defined ranges.

5. You can change the format of the time by using the COUNTRY configuration command. Refer to Chapter 4 "Configuring Your System" for information on the COUNTRY command.

6. If you are using an IBM Personal Computer AT, changing the time does not change the system clock. Refer to your *GTO* for information on the system clock.

Example: The following example sets the time to 13:55:00.00.

```
A>time
Current time is 00:25:16.65
Enter new time: 13:55_
```

TREE Command

Purpose: Displays all of the directory paths found on the specified drive, and optionally lists the files in the root directory and each subdirectory.

Format: [*d*:][*path*]TREE [*d*:][/F]

Type: Internal External
***

Remarks: Specify the parameters:

[*d*:][*path*] before TREE to specify the drive and path that contains the TREE command file.

[*d*:] to specify the drive whose directory paths you want to display. If not specified, the default drive is used.

/F to also display the names of files in the root directory and in all subdirectories.

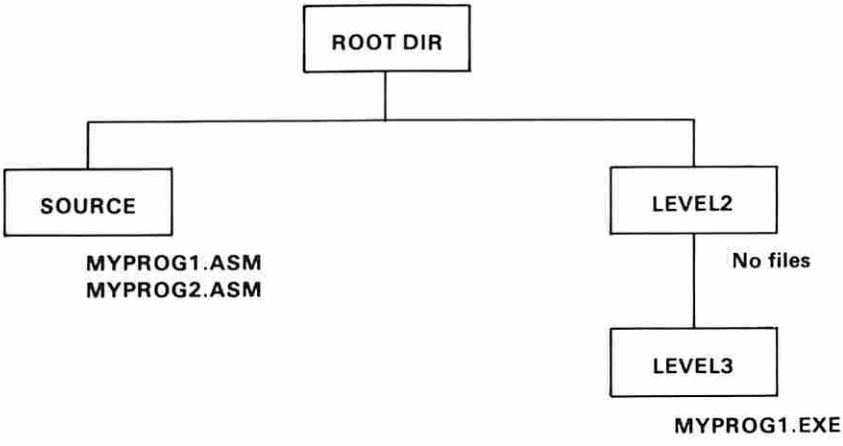
For each directory found, its full path name is displayed, along with the names of any directories defined within it (these are called subdirectories in the output). To pause the screen output, use the Pause Screen function or pipe the output to the MORE filter.

Example: The following example lists all the directory paths and the names of all files in the root directory and each subdirectory on drive B. The output is placed in the file TREE.LST in the current directory of drive A.

```
A>tree b:/f >tree.lst
```

TREE Command

Following is an example of a directory path listing. If the disk called MYDISK in drive A had the following directory structure:



Then TREE /F would display:

```
DIRECTORY PATH LISTING FOR VOLUME MYDISK

Files:        None

Path: \SOURCE

Subdirectories: None

Files:        MYPROG1 .ASM
              MYPROG2 .ASM


Path: \LEVEL2

Subdirectories: LEVEL3

Files:        None


Path: \LEVEL2\LEVEL3

Subdirectories: None

Files:        MYPROG1 .EXE
```

The following example lists all the directory paths and filenames of the root directory and of all subdirectories on drive A. The output is sent to the printer.

```
A>tree a:/f >prn
```

TYPE Command

Purpose: Displays the contents of the specified file on the standard output device.

Format: TYPE [*d*:][*path*]*filename*[.*ext*]

Type: Internal External
***

Remarks: The data is unformatted except that tab characters are expanded to an 8–character boundary; that is, columns 8, 16, 24, etc.

Notes:

1. Use the Print Screen function to print the contents of a file as it is displayed. You can also redirect the output to a file or the printer.

2. Text files appear in a legible format; however, other files, such as object program files, may appear unreadable due to the presence of nonalphabetic or nonnumeric characters.

3. Global filename characters are not allowed in the filename or extension.

Example: The following example displays the file MYFILE.ONE on drive B on the standard output device.

```
A>type b:myfile.one
```

VER (Version) Command

Purpose: Displays the DOS version number that you are working with on the display screen or the standard output device.

Format: VER

Type: Internal External
***

Remarks: The DOS version consists of a single–digit major version number, followed by a period, followed by a two–digit minor version number.

Example: The following example displays the current DOS version number.

```
A>ver
```

The result is:

```
IBM Personal Computer DOS Version 3.10.
```

The major version number is 3 and the minor version number is 10.

VERIFY Command

Purpose: Verifies that the data written on a disk has been correctly recorded.

Format: VERIFY [ON | OFF]

Type: Internal External
***

Remarks: VERIFY ON remains on until it is turned off through the SET VERIFY System Call or a VERIFY OFF command. When ON, DOS performs a verify operation following each disk write operation, to verify that the data just written can be read without error. Because of the extra time required to perform the verification, the system runs slower when programs write data to disk.

Typing VERIFY with no parameters displays the current state (ON or OFF) of the verify feature.

> **Note:** VERIFY is not supported with data written to a network disk.

Example: The following example sets VERIFY ON.

```
A>verify on
```

To display the current state of VERIFY, type:

```
A>verify
```

The result is:

```
VERIFY is on
A>
```

VOL (Volume) Command

Purpose: Displays the disk volume label of the specified drive.

Format: VOL [*d*:]

Type: Internal External
***

Remarks: If you do not specify a drive, the default drive is assumed.

Example: To display the volume label of drive A, type: A.

```
A>vol
```

The result is:

```
Volume in drive A is MYDISK

A>
```

If drive A has no label, the result is:

```
Volume in drive A has no label
```

Chapter 8. The Line Editor (EDLIN)

Contents

Introduction **8-3**

How to Start the EDLIN Program **8-5**
Editing an Existing File 8-5
Editing a New File 8-6

The EDLIN Command Parameters **8-7**

The EDLIN Commands **8-9**
Information Common to All EDLIN Commands 8-9

A (Append Lines) Command **8-11**

C (Copy Lines) Command **8-12**

D (Delete Lines) Command **8-13**

Edit Line Command **8-16**

E (End Edit) Command **8-18**

I (Insert Lines) Command **8-19**

L (List Lines) Command **8-22**

M (Move Lines) Command **8-25**

P (Page) Command **8-26**

Q (Quit Edit) Command **8-27**

R (Replace Text) Command **8-28**

S (Search Text) Command **8-31**

T (Transfer Lines) Command 8-34

W (Write Lines) Command 8-35

Introduction

This chapter describes how to use the Line Editor (EDLIN) program.

You can use the Line Editor (EDLIN) to create, change, and display source files or text files. Source files are unassembled programs in source language format. Text files appear in a legible format.

EDLIN is a line text editor that you can use to:

- 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 of files created or edited by EDLIN is divided into lines of varying length, up to 253 characters per line.

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 inserted. When you delete lines, all line numbers following the deleted text decrease automatically by the number of lines deleted. Consequently, line numbers always go consecutively from 1 through the last line number.

> **Note:** EDLIN erases the original backup copy (.BAK) of the file when you issue an **E** (end edit) command, or if the disk space is required during the editing session to satisfy a **W** (write lines) command.

How to Start the EDLIN Program

To start EDLIN, type:

[*d*:][*path*]EDLIN [*d*:][*path*]filename[.*ext*][/B]

Editing an Existing File

If the specified file exists on the designated or default drive, the file is loaded into memory until memory is 75% full. If the entire file is loaded, the following message and prompt is displayed:

```
End of input file
*
```

You can then edit the file.

> **Note:** If you have not used the **/B** parameter, EDLIN will stop loading the file when the first Ctrl-Z is encountered in the file's text. If you wish to edit a file that is known to contain embedded Ctrl-Z characters (end-of-file marks), you should use the **/B** parameter. EDLIN then processes the entire file regardless of any embedded end-of-file marks.

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

If the entire file cannot be loaded into memory, EDLIN loads lines until memory is 75% full, then displays the * prompt. You can then edit the portion of the file that is in memory.

To edit the remainder of the file, you must write some of the edited lines to disk to free memory so that you can load unedited lines from disk into memory. Refer to the Append Lines and Write Lines commands in this chapter for the procedure.

Editing a New File

If the specified file does not exist on the drive, a new file is opened with the specified name. The following message and prompt are displayed:

```
New file
*_
```

Begin creating the file by entering the desired lines of text. To begin entering text, enter an **I** command to insert lines.

When you have completed the editing session, you can save the file using the End Edit command. The End Edit command is discussed in this chapter in the section called "The EDLIN Commands." The new file is saved with the filename and extension you specified in the EDLIN command when you first opened the file.

> **Note:** If you edit an existing file, when you end the edit session with the End Edit command, the original file is saved with an extension of .BAK. You cannot edit a file with a filename extension of .BAK with EDLIN because the system assumes it is a backup file. If you find it necessary to edit such a file, rename the file to another extension; then start EDLIN and specify the new name.

The EDLIN Command Parameters

Parameter	Definition
line	Denotes when you must specify a line number. There are three possible entries that you can make using this parameter: 1. Enter a decimal integer from 1-65529. If you specify a number greater than the number of lines in memory, the line is added after the last line. Line numbers must be separated from each other by a comma or space. OR 2. Enter a hash sign (#) to specify the line after the last line in memory. Entering a # has the same effect as specifying a number greater than the number of lines in memory.

Parameter	Definition
line	OR 3. Enter a period (.) to specify the current line. The current line indicates the location of the last change to the file, but it is not necessarily the last line displayed. The current line is marked by an asterisk (*) between the line number and the first character of text in the line. For example: 10:*FIRST CHARACTER OF TEXT
n	Denotes when you must specify lines. Enter the number of lines that you want to write to disk or load from disk. You only use this parameter with the Write Lines and Append Lines commands. These commands are meaningful only if the file to be edited is too large to fit in memory.
string	Denotes when you must enter one or more characters to represent text to be found, replaced, deleted, or to replace other text. You only use this parameter with the Search Text and Replace Text commands.

The EDLIN Commands

This section describes the EDLIN commands and tells how to use them. The commands are in alphabetic order, each with its purpose, format and remarks. Examples are provided where appropriate.

Information Common to All EDLIN Commands

The following information applies to all EDLIN commands:

- With the exception of the Edit Line command, all commands are a single letter.
- With the exception of the End Edit and Quit Edit commands, commands are usually preceded and/or followed by parameters.
- Enter commands and string parameters in uppercase, or lowercase, or a combination of both.
- Separate commands and parameters with delimiters for readability; however, a delimiter is only required between two adjacent line numbers. Remember, delimiters are spaces or commas.
- Commands become effective only after you press the Enter key.
- Stop commands by pressing the Ctrl-Break keys.
- For commands producing a large amount of output, press Ctrl-Num Lock to suspend the display so that you can read it before it scrolls away. Press any other character to restart the display.

- Use the control keys and DOS editing keys, described in the *DOS User's Guide*, while using EDLIN. They are very useful for editing *within a line*, while the EDLIN commands can be used for editing operations on *entire lines*.

- The prompt from EDLIN is an asterisk (*).

- It is possible to refer to line numbers relative to the current line. Use a minus (-) sign and a number to indicate a line before the current line. Use a plus (+) sign and a number to indicate a line after the current line. For example:

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

  This command displays 10 lines before the current line, the current line, and 10 lines after the current line.

- Multiple commands can be entered on one command line. When you enter the command to edit a single line using [*line*], you must use a semicolon to separate the commands on the line. In the case of the Search or Replace commands the [*string*] can be terminated by Ctrl-Z (F6) instead of the Enter key. Otherwise, one command can follow another without any special delimiting characters. For example:

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

  edits line 15 and then displays lines 10 through 20 on the screen.

- Control characters can be inserted into the text, or can be used in the strings for the Search and Replace text commands. To enter a control character, press Ctrl-V, then enter the desired control character in uppercase. For example, the sequence Ctrl-V, followed by Z generates the control character Ctrl-Z.

A (Append Lines) Command

Purpose: Adds the specified number of lines from disk to the file being edited in memory. The lines are added at the end of the current lines in memory.

Format: [*n*]A

Remarks: This command is only meaningful if the file being edited is too large to fit in memory. As many lines as possible are read into memory for editing when you start EDLIN. If you do not specify the number of lines, lines are appended to memory until available memory is 75% full. No action is taken if available memory is already 75% full.

To edit the remainder of the file that will not fit into memory, you must write edited lines in memory to disk before you can load unedited lines from disk into memory by using the Append Lines command. Refer to the Write Lines command for information on how to write edited lines to disk.

The message **End of input file** is displayed when the Append Lines command has read the last line of the file into memory.

C (Copy Lines) Command

Purpose: Copies the lines in the specified range to the line number specified by the third parameter. The new data is placed ahead of the line that was specified in the third parameter. This third parameter is not optional. The operation is repeated the number of times specified in *count*.

Format: [*line*],[*line*],*line*[,*count*]C

Remarks: The parameter *count* defaults to 1. To repeat text, specify the number of times the operation is to be performed in *count*. If the first parameter or the second parameter is omitted, the default is the current line. This effectively copies the current line to the specified line. The file is renumbered accordingly. The first of the copied lines becomes the current line. For example:

```
1,5,8C
```

copies lines 1 through 5 to line 8. Line 8 becomes the current line.

The line numbers must not overlap or an error is reported. Also, the characters - and + are not allowed in the *count* field.

D (Delete Lines) Command

Purpose: Deletes a specified range of lines.

Format: [*line*][,*line*] D

Remarks: The line following the deleted range becomes the current line, even if the deleted range includes the last line in memory. The current line and any following lines are renumbered.

If you omit the first parameter, as in:

```
,lineD
```

deletion starts with the current line and ends with the line specified by the second parameter. The beginning comma is required to indicate the omitted first parameter.

If you omit the second parameter, as in:

```
lineD   or   line,D
```

only the one specified line is deleted. If you omit both parameters, as in:

```
D
```

only the current line is deleted, and the line that follows becomes the current line.

Example: Assume that you want to edit the following file. The current line is line 22.

D (Delete Lines) Command

```
 1: This is a sample file used to demonstrate
 2: line deletion and dynamic
 3: line number generation.
 .              .
 .              .
 .              .
20: See what happens to the lines
21: and line numbers when lines are
22:*deleted.
23: See how easy this is to use.
```

If you want to delete a range of lines, from 3–20, enter:

```
3,20D
```

The result is:

```
1: This is a sample file used to demonstrate
2: line deletion and dynamic
3:*and line numbers when lines are
4: deleted.
5: See how easy this is to use.
```

Lines 3–20 are deleted from the file. Lines 21–23 are renumbered to 3–5. Line 3 becomes the current line. If you want to delete the current and the following line, enter:

```
,4D
```

The result is:

```
1: This is a sample file used to demonstrate
2: line deletion and dynamic
3:*See how easy this is to use.
```

Lines 3 and 4 are deleted from the file. Line 5 is renumbered to 3. Line 3 is still the current line, but now it has different text.

D (Delete Lines) Command

If you want to delete a single line, say line 2, enter:

```
2D
```

The result is:

```
1: This is a sample file used to demonstrate
2:*See how easy this is to use.
```

Line 2 is deleted. Line 3 is renumbered to 2. The new line 2 becomes the current line. If you want to delete only the current line, enter:

```
D
```

The result is:

```
1:*This is a sample file used to demonstrate
```

The current line, line 2, is deleted. The new line 1 becomes the current line.

Edit Line Command

Purpose: Allows you to edit a line of text. You must enter the line number of the line to be edited, or enter a period (.) to indicate the current line.

Format: [*line*]

Remarks: If you just press Enter, you specify that the line after the current line is to be edited. The line number and its text are displayed and the line number is repeated on the line below.

You can use the control keys and the editing keys, described in the *DOS User's Guide*, to edit the line, or you can replace the entire line by typing new text. When you press the Enter key, the edited line is placed in the file and becomes the current line.

If you decide not to save the changed line, press either Esc or Ctrl-Break instead of Enter. The original line remains unchanged. Pressing the Enter key with the cursor at the beginning of the line has the same effect as pressing Esc or Ctrl-Break.

If the cursor is in any position other than the beginning or the end of a line, pressing Enter erases the rest of the line.

Example: Assume that you want to edit line 6. The following display would appear on the screen:

```
*6
   6: This is a sample unedited line.
   6: _
```

The first line is your request to edit line 6, followed by the two-line display response.

If you want to move the cursor to the letter **u**, press F2 and enter:

```
u
```

The result is:

```
*6
  6: This is a sample unedited line.
  6: This is a sample __
```

If you want to delete the next two characters and keep the remainder of the line, press Del twice; then press F3.

The result is:

```
*6
  6: This is a sample unedited line.
  6: This is a sample edited line.__
```

Now you can take one of the following actions:

- Press Enter to save the changed line.
- Extend the changed line by typing more text. You are automatically in insert mode when the cursor is at the end of a line.
- Press F5 to do additional editing to the changed line without changing the original line.
- Press Esc or Ctrl-Break to cancel the changes you made to the line. The original contents of the line are preserved.

E (End Edit) Command

Purpose: Ends EDLIN and saves the edited file.

Format: E

Remarks: The edited file is saved by writing it to the drive and filename specified when you started EDLIN.

The original file, the one specified when EDLIN was started, is given a .BAK filename extension. A .BAK file will not be created if there is no original file; that is, if you created a new file instead of updating an old file during the editing session.

EDLIN returns to the DOS command processor, which displays the command prompt.

Be sure your disk has enough free space to save the entire file. If your disk does not have enough free space, only a portion of the file is saved. The portion in memory that is not written to disk is lost. In this case, your original file is not renamed to .BAK, and the portion of data that was written to disk will have a filename extension of $$$.

EDLIN appends a carriage return, line feed sequence to the end of the file, if they were not already present, to delimit the last line of text in the file. Also, a Ctrl-Z character is added as the last character in the saved file. This serves as an end-of-file mark.

I (Insert Lines) Command

Purpose: Inserts lines of text immediately *before* the specified line. When you create a new file, you must enter the Insert Lines command before text can be inserted.

Format: [*line*]I

Remarks: If you do not specify line, or if you specify line as a period (.), the insert is made immediately before the current line. If the line number you specify is greater than the highest existing line number, or if you specify # as the line number, the insertion is made after the last line in memory.

EDLIN displays the appropriate line number so that you can enter more lines, ending each line by pressing Enter. During the insert mode of operation, successive line numbers appear automatically each time Enter is pressed.

You must press Ctrl-Break to discontinue the insert mode of operation.

The line that follows the inserted lines becomes the current line, even if the inserted lines are added to the end of the lines in memory. The current line and any remaining lines are renumbered.

Example: Assume that you want to edit the following file. Line 2 is the current line.

```
1: This is a sample file used to demonstrate
2:*line deletion
3: and dynamic line number generation.
```

If you want to insert text before line 3, the entry and immediate response look like this:

I (Insert Lines) Command

```
*3I
   3:*_
```

Now, if you want to insert two new lines of text, enter:

```
*3 I
   3:*First new line of text
   4:*Second new line of text
   5:*
```

and press Ctrl-Break.

The original line 3 is now renumbered to line 5.

If you display the file with a List Lines command, the file looks like this:

```
1: This is a sample file used to demonstrate
2: line deletion
3: First new line of text
4: Second new line of text
5:*and dynamic line number generation.
```

If the two lines that were inserted had been placed at the beginning of the file, the screen would look like this:

```
1: First new line of text
2: Second new line of text
3:*This is a sample file used to demonstrate
4: line deletion
5: and dynamic line number generation.
```

If the two lines that were inserted had been placed immediately before the current line (2 I or . I or I), the screen would look like this:

```
1: This is a sample file used to demonstrate
2: First new line of text
3: Second new line of text
4:*line deletion
5: and dynamic line number generation.
```

If the two inserted lines had been placed at the end of the file (4 I or # I), the screen would look like this:

```
1: This is a sample file used to demonstrate
2: line deletion
3: and dynamic line number generation.
4: First new line of text
5: Second new line of text
```

L (List Lines) Command

Purpose: Displays a specified range of lines. The current line remains unchanged.

Format: [*line*][,*line*]L

Remarks: Default values are provided if either one or both of the parameters are omitted. If you omit the first parameter, as in:

```
,lineL
```

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 parameter.

> **Note:** If the specified line is more than 11 lines before the current line, the display is the same as if you omitted both parameters. (An example is provided in this section showing both parameters omitted.)

If you omit the second parameter, as in:

```
lineL   or   line,L
```

a total of 23 lines are displayed, starting with the specified *line*.

If you omit both parameters, as in:

```
L
```

a total of 23 lines are displayed–the 11 lines before the current line, the current line, and the 11 lines after the current line. If there aren't 11 lines before the current line, then extra lines are displayed after the current line to make a total of 23 lines.

Example: Assume that you want to edit the following file. Line 15 is the current line.

```
 1: This is a sample file used to demonstrate
 2: line deletion and dynamic
 3: line number generation.
 .              .
 .              .
 .              .
15:*This is the current line (note the asterisk)
 .              .
 .              .
 .              .
25: See what happens to the lines
26: and line numbers when lines are
27: deleted.
```

If you want to display a range of lines, from 3–25, enter:

```
3,25L
```

The screen looks like this:

```
 3: line number generation.
 .              .
 .              .
 .              .
15:*This is the current line (note the asterisk)
 .              .
 .              .
 .              .
25: See what happens to the lines
```

If you want to display the first three lines, enter:

```
1,3L
```

The screen looks like this:

L (List Lines) Command

```
 1: This is a sample file used to demonstrate
 2: line deletion and dynamic
 3: line number generation.
```

If you want to display 23 lines of the file, starting with line 3, enter:

```
3L
```

The screen looks like this:

```
 3: line number generation.
  .              .
  .              .
  .              .
15:*This is the current line (note the asterisk)
  .              .
  .              .
  .              .
 25: See what happens to the lines
```

If you want to display 23 lines centered around the current line, enter:

```
L
```

The screen looks like this:

```
 4: Fourth line of text
 5: Fifth line of text
  .              .
  .              .
  .              .
15:*This is the current line (note the asterisk)
  .              .
  .              .
  .              .
 25: See what happens to the lines
 26: and line numbers when lines are
```

M (Move Lines) Command

Purpose: Moves the range of lines specified by the first two *line* parameters ahead of the line specified in the third *line* parameter. The third parameter is not optional.

Format: [*line*],[*line*],*line*M

Remarks: Use this command to move a block of data from one location in the file to another. If the first or second line parameter is omitted, it will default to the current line. After the move, the first of the moved lines becomes the current line. The lines are renumbered according to the direction of the move. For example:

```
,+25,100M
```

moves the data from the current line plus 25 lines to line 100. If the arguments overlap an entry error is reported.

P (Page) Command

Purpose: Lists the specified block of lines.

Format: [*line*][,*line*]P

Remarks: If the first *line* parameter is omitted, it defaults to the current line plus one. If the second *line* parameter is omitted, 23 lines are listed. The new current line becomes the last line displayed by the Page command and is marked with an asterisk. This command pages through a file displaying 23 lines at a time. It differs from the List Lines command in that it changes the current line.

Q (Quit Edit) Command

Purpose: Quits the editing session without saving any changes you may have entered.

Format: Q

Remarks: EDLIN prompts you to make sure you really don't want to save the changes.

Enter **Y** if you want to quit the editing session. No editing changes are saved and no .BAK file is created. Refer to the End Edit command for information about the .BAK file.

Enter **N**, or any other character, if you want to continue the editing session.

Example: When you type the letter Q, the following is displayed:

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

R (Replace Text) Command

Purpose: Replaces all occurrences of the first string in the specified range of lines with the second string.

Notes:

1. If you omit the second string, Replace Text deletes all occurrences of the first string within the specified range of lines. If you omit both strings, EDLIN re-uses the search string entered with the most recent (previous) **S** or **R** command, and the Replace Text string entered with the last **R** command.

2. This command uses the F6 key as normally setup by DOS. If you have changed the meaning of the F6 key through "Extended Keyboard Control" (see Chapter 3 of the *DOS Technical Reference* manual), you should press Ctrl-Z where F6 is referred to below.

EDLIN displays the changed lines each time they are changed. The last line changed becomes the current line.

Format: [*line*][,*line*] [?]R[*string*][<F6>*string*]

Remarks: You can specify the optional parameter **?** to request a prompt (**O.K.?**) after each display of a modified line. Press **Y** or the Enter key if you want to keep the modification.

Enter any other character if you don't want the modification. In either case, the search continues for further occurrences of the first string within the range of lines, including multiple occurrences within the same line.

Defaults occur if either one or both of the *line* parameters are missing.

If you omit the first *line*, the search begins with the line after the current line. If you omit the second *line*, the search ends with the last line in memory. If you omit both *line* parameters, the system searches from the line following the current line to the last line in memory.

> **Note:** The first string begins with the character in the position immediately following the R, and continues until you press F6 or Ctrl-Z (or the Enter key if the second string is omitted).
>
> The second string begins immediately after you press F6 or Ctrl-Z and continues until you press Enter.

Example: Assume that you want to edit the following file. Line 7 is the current line.

```
1: This is a sample file
2: used to demonstrate
3: the Replace and Search Text commands.
4: This includes the
5: optional parameter ?
6: and required string
7:*parameter.
```

R (Replace Text) Command

To replace all occurrences of **and** with **or** in the lines in memory, enter:

```
1,7 Rand
```

Then press F6, type **or**, and press Enter.

The result is:

```
3: the Replace or Search Text commands.
3: the Replace or Search Text commors.
6: or required string
```

Line 6 becomes the current line in the file, because line 6 was the last line changed. Notice that lines 1, 2, 4, 5, and 7 are not displayed because they were not changed.

Greater selectivity can be achieved by requesting a prompt (by using the ? parameter) after each display of a modified line. If you request a prompt, the screen looks like this:

```
*1,7 ? Rand (Press F6, type or, and press Enter)
     3: the Replace or Search Text commands
O.K.? Y
     3: the Replace or Search Text commors
O.K.? N
     6: or required string
O.K.? Y
*
```

Lines 3 and 6 are displayed like this:

```
3: the Replace or Search Text commands.
6: or required string
```

S (Search Text) Command

Purpose: Searches a specified range of lines in order to locate a specified string.

Format: [*line*][,*line*] [?]S[*string*]

Remarks: The first line to contain the specified string is displayed and the search ends (unless you use the **?** parameter). The first line found that contains the specified string becomes the current line.

> **Note:** The Search command always searches for the exact same character in text. That is, it searches for UPPERCASE if you enter UPPERCASE, and lowercase if you enter lowercase.

You should specify the optional parameter **?** if you would like a prompt (O.K.?) after each display of a line containing the specified string.

If you do not enter a string, the **S** command uses the last search string entered on a Replace or Search command. If the specified string is not found, the search ends and the message **Not found** is displayed. The current line remains unchanged. If you enter **Y** or press the Enter key, the line that matches the specified string becomes the current line and the search ends. Enter any other character to continue the search until another string is found, or until all lines within the range are searched. Once all the lines within the range are searched, the **Not found** message is displayed.

The system provides default values if you omit the first, second, or both line parameters. If you omit the first line parameter, the system defaults to the

line following the current line. If you omit the second line parameter, the system defaults to the last line in memory. If you omit both line parameters, the system searches from the line following the current line to the last line in memory.

Notes:

1. The string begins with the character in the position immediately following the **S** and continues until you end the string by pressing the Enter key.

2. If you wish to place more than one command on a line containing a Search Text command, the Search Text command should end in a Ctrl-Z (F6), and the next command should begin in the following character position.

Example: Assume that you want to edit the following file. Line 7 is the current line.

```
1: This is a sample file
2: used to demonstrate
3: the Search Text command.
4: This includes the
5: optional parameter ?
6: and required string
7:*parameter.
```

If you want to search for the first occurrence of **and** in the file, enter:

```
1,7 Sand
   or
1, Sand
   or
1Sand
```

The result is:

```
     3: the Search Text command.
*
```

The **and** is part of the word **command**. Notice that line 3 becomes the current line in the file.

Perhaps this is not the **and** you were looking for. To continue the search, simply enter the letter **S** and press Enter. The search continues with the line following the current line (the line just found).

The screen looks like this:

```
*1,7 Sand
     3: the Search Text command.
*S
     6: and required string
*
```

Line 6 now becomes the current line in the file.

You can also search for strings by requesting a prompt (by means of the ? parameter) after each display of a matching line. In this case, the screen looks like this:

```
*1,7 ? Sand
     3: the Search Text command.
O.K.? N
     6: and required string
O.K.? Y
*
```

T (Transfer Lines) Command

Purpose: Transfers (merges) the contents of a specified file into the file currently being edited.

Format: [*line*]T:[*d*:]*filename*

Remarks: The *filename* contents are inserted ahead of the *line* in the file being edited. If *line* is omitted, the current line is used.

> **Note:** The file being merged is read from the current directory of the specified or default drive. If a path was specified when you issued the EDLIN command, that path becomes the current directory for that drive for the duration of the EDLIN session, and any Transfer Lines commands for that drive must be satisfied from the same directory.

W (Write Lines) Command

Purpose: Writes a specified number of lines to disk from the lines being edited in memory. Lines are written beginning with line number 1.

Format: [*n*]W

Remarks: This command is only meaningful if the file you are editing is too large to fit in memory. When you start EDLIN, it reads lines into memory until memory is 75% full.

To edit the remainder of the file not in memory, you must write edited lines in memory to disk. Then you can load additional unedited lines from disk into memory by using the Append Lines command.

> **Note:** If you do not specify the number of lines, lines are written until 25% of available memory is used. No action is taken if available memory is already less than 25% used. All lines are renumbered so that the first remaining line becomes number 1.

Notes:

Chapter 9. The Linker (LINK) Program

Contents

Introduction **9-3**

Files **9-4**
Input Files 9-4
Output Files 9-5
VM.TMP (Temporary File) 9-5

Definitions **9-6**
Segment 9-6
Group 9-7
Class 9-7

Command Prompts **9-7**

Detailed Descriptions of the Command Prompts .. **9-9**
Object Modules [.OBJ]: 9-9
Run File [filename.EXE]: 9-10
List File [NUL.MAP]: 9-10
Libraries [.LIB]: 9-12
Linker Parameters 9-14
/DSALLOCATION 9-14
/HIGH 9-14
/LINE 9-15
/MAP 9-15
/PAUSE 9-15
/STACK:size 9-16
/X 9-16
/O 9-17

How to Start the Linker Program **9-17**
Before You Begin 9-17
Option 1 - Console Responses 9-17
Option 2 - Command Line 9-18
Option 3 - Automatic Responses 9-21

Example Linker Session **9-23**
How to Determine the Absolute Address of a Segment 9-26

Messages **9-27**

Introduction

The linker (LINK) program:

- Combines separately produced object modules
- Searches library files for definitions of unresolved external references
- Resolves external cross-references
- Produces a printable listing that shows the resolution of external references and error messages
- Produces a relocatable load module

The LINK program resides on your DOS Supplemental Program Diskette. This chapter shows you how to use LINK. Read all of this chapter before you start LINK.

Files

The linker processes input, output, and temporary files.

Input Files

Type	Default .ext	Override .ext	Produced by
Object	.OBJ	Yes	Compiler[1] or MACRO Assembler
Library	.LIB	Yes	Compiler and user
Automatic Response	(None)	N/A*	User

Figure 9-1 **Input Files Used by the Linker**

*N/A - Not applicable.

[1] One of the optional compiler packages available for use with the IBM Personal Computer DOS.

Type	Default .ext	Override .ext	Used by
Listing	.MAP	Yes	User
Run	.EXE	No	Relocatable loader (COMMAND.COM)

Figure 9-2 Output Files Used by the Linker

VM.TMP (Temporary File)

LINK uses as much memory as is available to hold the data that defines the load module being created. If the module is too large to be processed with the available amount of memory, the linker may need additional memory space. If this happens, a temporary file called VM.TMP is created on the DOS default drive.

When the overflow to the VM.TMP file has begun, the linker displays the following message:

```
VM.TMP has been created
Do not change diskette in drive x
```

If the VM.TMP file has been created on diskette, you should not remove the diskette until LINK ends. When LINK ends, it erases the VM.TMP file.

If the DOS default drive already has a file by the name of VM.TMP, it is deleted by LINK and a new file is

allocated; the contents of the previous file are destroyed. Therefore, you should avoid using VM.TMP as one of your own filenames.

Definitions

Segment, *group*, and *class* are terms that appear in this chapter and in some of the messages in Appendix A. These terms describe the underlying function of LINK. An understanding of the concepts that define these terms provides a basic understanding of the way LINK works.

Segment

A *segment* is a contiguous area of memory up to 64K bytes in length. A segment may be located anywhere in memory on a *paragraph* (16-byte) boundary. Each of the four segment registers defines a segment. The segments can overlap. The contents of a segment are addressed by a segment register/offset pair.

The contents of various portions of the segment are determined when machine language is generated.

Neither size nor location is necessarily fixed by the compiler or assembler because this portion of the segment may be combined at link time with other portions forming a single segment.

A program's ultimate location in memory is determined at load time by the relocation loader facility provided in COMMAND.COM, based on whether you specified the /HIGH parameter. The /HIGH parameter is discussed later in this chapter.

Group

A *group* is a collection of segments that fit together within a 64K byte segment of memory. The segments are named to the group by the assembler or compiler. A program may consist of one or more groups.

The group is used for addressing segments in memory. The various portions of segments within the group are addressed by a segment base pointer plus an offset.

Class

A *class* is a collection of segments. The naming of segments to a class affects the order and relative placement of segments in memory. The class name is specified by the assembler or compiler. All portions assigned to the same class name are loaded into memory contiguously.

The segments are ordered within a class in the order that the linker encounters the segments in the object files. One class precedes another in memory only if a segment for the first class precedes all segments for the second class in the input to LINK. Classes are not restricted in size.

Command Prompts

After you start the linker session, you receive a series of four prompts. You can respond to these prompts from the keyboard, on the command line, or by using a special diskette file called an *automatic response file*. An example of an automatic response file is provided in this chapter.

LINK prompts you for the names of the object, run, list, and library files. When the session is finished, LINK returns to DOS and the DOS prompt is displayed. If linking is unsuccessful, LINK displays a message.

The prompts are described in order of their appearance on the screen. Defaults are shown in square brackets ([ ]) after the prompt. In the response column of the table, square brackets indicate optional entries. **Object Modules** is the only prompt that requires a response from you.

PROMPT	RESPONSE
Object Modules [.OBJ]:	[*d*:][path]*filename*[.*ext*] [+.[*d*:][path]*filename*[.*ext*]]...
Run File [filename.EXE]:	[*d*:][path][*filename*[.*ext*]]
List File [NUL.MAP]:	[*d*:][path][*filename*[.*ext*]]
Libraries [.LIB]:	[*d*:][[path]*filename*[.*ext*]] [+.[*d*:][[path]*filename*[.*ext*]]]...

Notes:

1. If you enter a filename without specifying the drive, the default drive is assumed. If you enter a filename without specifying the path, the default path is assumed. The libraries prompt is an exception–the linker will look for the libraries on the default drive and if not found, look on the drive specified by the compiler.

2. You can end the linker session prior to its normal end by pressing Ctrl–Break.

Detailed Descriptions of the Command Prompts

The following detailed descriptions contain information about the responses that you can enter to the prompts.

Object Modules [.OBJ]:

Enter one or more file locations for the object modules to be linked. Multiple file locations must be separated by single plus (+) signs or blanks. If the extension is omitted from any filename, LINK assumes the filename extension **.OBJ**. If an object module has a different filename extension, the extension must be specified. Object filenames can not begin with the @ symbol (@ is reserved for using an automatic response file).

LINK loads segments into classes in the order encountered.

If you specify an object module on a diskette drive, but LINK cannot locate the file, it displays the following prompt:

```
Cannot find file object module
change diskette <press ENTER>
```

If you specify an object module on a non-removable media (like a fixed disk), the linker session ends with the following message:

```
Cannot find file object module
```

You should insert the diskette containing the requested module. This permits **.OBJ** files from several diskettes to be included. On a single-drive system, diskette exchanging can be done safely *only* if

VM.TMP has *not* been opened. As explained in the discussion of the VM.TMP file earlier in this chapter, a message will indicate if VM.TMP has been opened.

Important: If a VM.TMP file has been opened on a diskette, you should *not* remove the diskette containing the VM.TMP file.

After a VM.TMP file has been opened, if you specified an object module on the same disk that VM.TMP is on and LINK cannot find it, the linker session ends with the message:

```
Cannot find file object module
```

Run File [filename.EXE]:

The file specification you enter is created to store the run (executable) file that results from the LINK session. All run files receive the filename extension **.EXE**, even if you specify another extension. If you specify another extension, it is ignored.

The default filename for the run file prompt is the first filename specified on the object module prompt.

You can specify just a drive letter, or a path on the run file prompt. This changes the place where the run file *filename*.EXE is placed.

List File [NUL.MAP]:

The linker list file is sometimes called the linker *map*.

The list file is not created unless you specifically request it. You can request it by overriding the default with a drive letter, path, or *filename*[*.ext*]. If you do not include a filename extension, the default extension **.MAP** is used. If you do not enter anything, the DOS reserved filename **NUL** specifies that no list file is created.

The list file contains an entry for each segment in the input (object) modules. Each entry also shows the offset (addressing) in the run file.

You can specify just a drive letter or a path on the list file prompt. This changes the place where the list file is placed.

Important: If the list file is allocated to a file on diskette, that diskette must not be removed until the LINK has ended.

> **Note:** There is one exception. If /P is specified, the diskette containing the list file may be removed while the .EXE file is being written. The linker prompts you to put back the diskette containing the list file when it finishes writing the .EXE file.

If you specify an object module on the same diskette drive as the diskette drive to which the list file is allocated, and LINK cannot find the object module, the linker session ends with the message:

```
Cannot find file object module
```

To avoid generating the list file on a diskette, you can specify the display or printer as the list file device. For example:

```
List File [NUL.MAP]: CON
```

If you direct the output to your display, you can also print a copy of the output by pressing Ctrl-PrtSc.

Libraries [.LIB]:

You may either list the file locations for your libraries, or just press the Enter key. If you press the Enter key, LINK defaults to the library provided as part of the Compiler package.

The LINK program looks for the Compiler package library on the default drive. If it cannot find the library there, it looks for the library on the drive specified by the Compiler package. For linking objects from just the MACRO Assembler, there is no automatic default library search.

If you answer the library prompt, you specify a list of drive letters and [*path*]*filename.ext* separated by plus signs (+) or spaces. You can enter from 1 to 16 library file locations. Specifying a drive letter tells linker to look on that drive instead of the Compiler package supplied drive for all subsequent libraries on the library prompt. The automatically searched library file specifications are conceptually placed at the end of the response to the library prompt.

LINK searches the library files in the order they are listed to resolve external references. When LINK finds the module that defines the external symbol, the module is processed as another object module.

If two or more libraries have the same filename, regardless of the location, only the first library in the list is searched.

When LINK cannot find a library file, it displays a message like this:

```
Cannot find library A:library file
Enter new drive letter:
```

The drive that the indicated library is located on must be entered.

The following library prompt responses may be used:

Libraries [.LIB]: B:

Look for compiler.LIB on drive B.

Libraries [.LIB]: B:USERLIB

Look for USERLIB.LIB on drive B and compiler.LIB on drive A.

Libraries [.LIB]: A:LIB1+LIB2+B:LIB3+A:

Look for LIB1.LIB and LIB2.LIB on drive A, LIB3.LIB on drive B, and compiler.LIB on drive A.

Linker Parameters

At the end of any of the four linker prompts, you may specify one or more parameters that instruct the linker to do something differently. Only the / and first letter of any parameter are required.

/DSALLOCATION

The /DSALLOCATION (/D) parameter directs LINK to load all data defined to be in DGROUP at the *high end* of the group. If the /HIGH parameter is specified, (module loaded high), any available storage below the specifically allocated area within DGROUP is allocated dynamically by your application. It still is addressable by the same data space pointer.

> **Note:** The maximum amount of storage which can be dynamically allocated by the application is 64K-bytes (or the amount actually available) minus the allocated portion of DGROUP.

If the /DSALLOCATION parameter is not specified, LINK loads all data defined to be in the group whose group name is DGROUP at the *low end* of the group, beginning at an offset of 0. The only storage thus referenced by the data space pointer should be that specifically defined as residing in the group.

All other segments of any type in any GROUP other than DGROUP are loaded at the low end of their respective groups, as if the /DSALLOCATION parameter were not specified.

For certain compiler packages, /DSALLOCATION is automatically used.

/HIGH

The /HIGH (/H) parameter causes the loader to place the run image as high as possible in storage. If

you specify the /HIGH parameter, you tell the linker to cause the loader to place the run file as high as possible without overlaying the transient portion of COMMAND.COM, which occupies the highest area of storage when loaded. If you do not specify the /HIGH parameter, the linker directs the loader to place the run file as low in memory as possible.

The /HIGH parameter is used with the /DSALLOCATION parameter.

/LINE

For certain IBM Personal Computer language processors, the /LINE (/L) parameter directs LINK to include the line numbers and addresses of the source statements in the input modules in the list file.

/MAP

The /MAP (/M) parameter directs LINK to list all public (global) symbols defined in the input modules. For each symbol, LINK lists its value and segment-offset location in the run file. The symbols are listed at the end of the list file.

/PAUSE

The /PAUSE (/P) parameter tells LINK to display a message to you as follows:

```
About to generate .EXE file
Change diskette in drive A: and press ENTER
```

This message allows you to insert the diskette that is to contain the run file.

/STACK:size

The *size* entry is any positive decimal value up to 65536 bytes. This value is used to override the size of the stack that the MACRO Assembler or compiler has provided for the load module being created. If the size of the stack is too small, the results of executing the resulting load module are unpredictable.

If you do not specify /STACK (/S), the orginal stack size provided by the MACRO Assembler or compiler is used. This parameter can be used to reduce the stack size provided by the application only if the original stack has uninitialized data. In any case, it may increase the stack up to the 64K limit.

If the stack size is an odd number, either on the /S parameter or in the application's definition of the stack, LINK subtracts one to force the stack words to be on an even boundary for better efficiency when running on the 80286.

At least one input (object) module must contain a stack allocation statement, unless you plan to use the EXE2BIN program. This is automatically provided by compilers. For the MACRO Assembler, the source must contain a SEGMENT command that has the combine type of STACK. If a stack allocation statement was not provided, LINK returns the message **Warning: No Stack statement**.

/X

Use the /X parameter at runtime to adjust the total number of segments that an .EXE file can contain. You can vary the limit from 0 to 1024. The default is 256 segments. This limit represents the number of distinct segments from all sources (object files and libraries) that the .EXE may contain.

Although the limit on the total number of segments may be set as high as 1024, the limit on the total

number of segments that are not *absolute segments* is 1000. For a definition of *absolute segments*, see the assembler manual.

/O

To link object modules created by version 1 of the Pascal compiler or version 1 of the FORTRAN compiler using the 2.30 linker, specify the /O (old) switch.

How to Start the Linker Program

Before You Begin

- Make sure the files you use for linking are on the appropriate disks.
- Make sure you have enough free space on your disks to contain your files and any generated data.

You can start the linker program by using one of three options:

Option 1 - Console Responses

From your keyboard, type:

```
LINK
```

The linker is loaded into memory and displays a series of four prompts, one at a time, to which you must enter the requested responses. (Detailed descriptions of the responses that you can make to the prompts are discussed in this chapter.)

If you enter a wrong response, such as an incorrectly spelled filename, you must press Ctrl-Break to exit LINK, then restart LINK. If the response in error has been typed but you haven't pressed Enter yet, you can delete the wrong characters (on that line only).

An example of a linker session using the console response option is provided in this chapter in the section called "Example Linker Session."

As soon as you have entered the last filename, the linker begins to run. If the linker finds any errors, it displays the errors on the screen as well as in the listing file.

> **Note:** After any of these responses, before pressing Enter, you can continue the response with a comma and the answer to what would be the next prompt, without having to wait for that prompt. If you end any with the semicolon (;), the remaining responses are all assumed to be the default. Processing begins immediately with no further prompting.

Option 2 - Command Line

From your keyboard, type:

LINK *objlist,runfile,mapfile,liblist* [*parm*]...;

objlist is a list of object modules separated by spaces or plus signs (+).

runfile is the name you want to give the run file.

mapfile is the name you want to give the linker map.

liblist is a list of the libraries to be used, separated by plus signs (+) or spaces.

parm is an optional linker parameter. Each parameter must begin with a slash (/).

The linker is loaded and immediately performs the tasks indicated by the command line.

When you use this command line, the prompts described in Option 1 are not displayed if you specified an entry for all four files or if the command line ends with a semicolon.

If an incomplete list is given and no semicolon is used, the linker prompts for the remaining unspecified files.

Each prompt displays its default, which is accepted by pressing the Enter key, or overridden with an explicit filename or device name. However, if an incomplete list is given and the command line is terminated with a final semicolon, the unspecified files default without further prompting. The *parms* are never prompted for, but may be added to the end of the command line or to any file specification given in response to a prompt.

Certain variations of this command line are permitted.

Examples:

LINK module

The object module is MODULE.OBJ. A prompt is given, showing the default of MODULE.EXE. After the response is entered, a prompt is given showing the default of NUL.MAP. After the response is given, a prompt is displayed showing the default extension of .LIB.

LINK module;

If the semicolon is added, no further prompts are displayed. The object module of MODULE.OBJ is linked, the run file is put into MODULE.EXE, and no list file is produced.

LINK module,,;

This is similar to the preceding example, except the list file is produced in MODULE.MAP.

LINK module,,

Using the same example, but without the semicolon, MODULE.OBJ is linked, and the run file is produced in MODULE.EXE, but a prompt is given with the default of MODULE.MAP.

LINK module,,NUL;

No list file is produced. The run file is in MODULE.EXE. No further prompts are displayed.

Option 3 - Automatic Responses

It is often convenient to save responses to the linker for use at a later time. This is especially useful when long lists of object modules need to be specified.

Before using this option, you must create the automatic response file. It contains several lines of text, each of which is the response to a linker prompt. These responses must be in the same order as the linker prompts that were discussed earlier in this chapter. If desired, a long response to the object module or libraries prompt may be contained on several lines by using a plus sign (+) to continue the same response onto the next line.

To specify an automatic response file, you enter a file specification preceded by an @ symbol in place of a prompt response or part of a prompt response. The prompt is answered by the contents of the diskette file. The file specification cannot be a reserved DOS filename.

From your keyboard, type:

LINK @[*d*:][path]*filename*[*ext*]

Use of the filename extension is optional and can be any name. There is no default extension.

Use of this option permits the command that starts LINK to be entered from the keyboard or within a batch file without requiring any response from you.

Example

```
Automatic Response File - RESP1

    MODA+MODB+MODC+
    MODD+MODE+MODF

Automatic Response File - RESP2

    runfile/p
    printout

Command line

    LINK @RESP1+mymod,@RESP2;
```

Notes:

1. The plus sign at the end of the first line in RESP1 causes the modules listed in the first two lines to be considered as the input object modules. After reading RESP1, the linker returns to the command line and sees **+mymod**, so it includes MYMOD.OBJ in the first list of object modules as well.

2. Each of the above lines ends when you press the Enter key.

Example Linker Session

This example shows you the type of information that is displayed during a linker session.

When you type:

```
b:link
```

in response to the DOS prompt, the system responds with the following messages and prompts, which you answer as shown:


```
IBM Personal Computer Linker
Version 2.30 (C)Copyright IBM Corp. 1981, 1982, 1983, 1984

Object Modules [.OBJ]: example
Run File [EXAMPLE.EXE]: /map
List File [NUL.MAP]: prn/line
Libraries [.LIB]:
```

Notes:

1. By specifying **/map**, you get both an alphabetic listing and a chronological listing of public symbols.

2. By responding **prn** to the list file prompt, you send your output to the printer.

3. By specifying the /LINE parameter, LINK gives a listing of all line numbers for all modules. (The /LINE parameter can generate a large amount of output.)

4. By just pressing Enter in response to the libraries prompt, an automatic library search is performed.

Once LINK locates all libraries, the linker map displays a list of segments in the relative order of their appearance within the load module. The list looks like this:

```
Start  Stop   Length  Name           Class

00000H 00028H 0029H   MAINQQ         CODE
00030H 000F6H 00C7H   ENTXQQ         CODE
00100H 00100H 0000H   INIXQQ         CODE
00100H 038D3H 37D4H   FILVQQ_CODE    CODE
038D4H 04921H 104EH   FILUQQ_CODE    CODE
.
.
.
074A0H 074A0H 0000H   HEAP           MEMORY
074A0H 074A0H 0000H   MEMORY         MEMORY
074A0H 0759FH 0100H   STACK          STACK
075A0H 07925H 0386H   DATA           DATA
07930H 082A9H 097AH   CONST          CONST
```

The information in the **Start** and **Stop** columns shows a 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 of where these segments are loaded. To find the absolute address of a segment, you must determine where the segment listed as being at relative zero is actually loaded; then add the absolute address to the relative address shown in the linker map. The procedure used to determine where relative zero is actually located is discussed in this chapter, in the section called "How to Determine the Absolute Address of a Segment."

Because you specified the /MAP parameter, the public symbols are displayed by name and by value. For example:

```
Address         Publics by Name

0492:0003H    ABSNQQ
06CD:029FH    ABSRQQ
0492:00A3H    ADDNQQ
06CD:0087H    ADDRQQ
0602:000FH    ALLHQQ
.
.
.
0010:1BCEH    WT4VQQ
0010:1D7EH    WTFVQQ
0010:1887H    WTIVQQ
0010:19E2H    WTNVQQ
0010:11B2H    WTRVQQ

Address         Publics by Value

0000:0001H    MAIN
0000:0010H    ENTGQQ
0000:0010H    MAINQQ
0003:0000H    BEGXQQ
0003:0095H    ENDXQQ
.
.
.
F82B:F31CH    CRCXQQ
F82B:F31EH    CRDXQQ
F82B:F322H    CESXQQ
F82B:F5B8H    FNSUQQ
F82B:F5E0H    OUTUQQ
```

The addresses of the public symbols are in the *segment:offset* format, showing the location relative to zero as the beginning of the load module. In some cases, an entry may look like this:

```
F8CC:EBE2H
```

This entry appears to be the address of a load module that is almost 1 megabyte in size. Actually, the area being referenced is relative to a segment base that is pointing to a segment below the relative zero beginning of the load module. This condition produces a pointer that has effectively gone negative. The memory map that follows illustrates this point.

When LINK has completed, the following message is displayed:

```
Program entry point at 0003:0000
```

How to Determine the Absolute Address of a Segment

The linker map displays a list of segments in the relative order of their appearance within the load module. The information displayed shows a 20-bit hex address of each segment relative to location zero. The addresses displayed are not the absolute addresses where these segments are located. To determine where relative zero is actually located, you must use DEBUG. DEBUG is described in detail in Chapter 12 of the *DOS Technical Reference*.

Using DEBUG,

1. Load the application. Note the segment value in CS and the offset within that segment to the entry point as shown in IP. The last line of the linker

map also describes this entry point, but uses relative values, not the absolute values shown by CS and IP.

2. Subtract the relative entry as shown at the end of the map listing from the CS:IP value. For example, let's say CS is at 05DC and IP is at zero.

   The linker map shows the entry point at 0100:0000. (0100 is a segment ID or paragraph number; 0000 is the offset into that segment.)

   In this example, relative zero is located at 04DC:0000, which is 04DC0 absolute.

If a program is loaded low, the relative zero location is located at the end of the Program Segment Prefix, in the location DS plus 100H.

Messages

All messages, except for the warning messages, cause the LINK session to end. Therefore, after you locate and correct a problem, you must rerun LINK.

Messages appear both in the list file and on the display unless you direct the list file to CON, in which case the display messages are suppressed.

All of the linker messages are included in Appendix A.

Notes:

Chapter 10. DEBUG Program

Contents

Introduction **10-3**

How to Start the DEBUG Program **10-4**

The DEBUG Command Parameters **10-6**

The DEBUG Commands **10-13**
Information Common to All DEBUG Commands 10-13

A (Assemble) Command **10-15**

C (Compare) Command **10-19**

D (Dump) Command **10-20**

E (Enter) Command **10-23**

F (Fill) Command **10-26**

G (Go) Command **10-27**

H (Hexarithmetic) Command **10-30**

I (Input) Command **10-31**

L (Load) Command **10-32**

M (Move) Command **10-35**

N (Name) Command **10-36**

O (Output) Command **10-38**

P (Proceed) Command **10-39**

Q (Quit) Command 10-40

R (Register) Command 10-41

S (Search) Command 10-46

T (Trace) Command 10-47

U (Unassemble) Command 10-49

W (Write) Command 10-52

Introduction

This chapter explains how to use the DEBUG program.

The DEBUG program can be used to:

- Provide a controlled testing environment so you can monitor and control the execution of a program to be debugged. You can fix problems in your program directly, and then execute the program immediately to determine if the problems have been resolved. You do not need to reassemble a program to find out if your changes worked.
- Load, alter, or display any file.
- Execute *object files*. Object files are executable programs in machine language format.

How to Start the DEBUG Program

To start DEBUG, type:

DEBUG [*d:*][*path*][*filename*[*.ext*]][*parm1*][*parm2*]

If you enter *filename*, the DEBUG program loads the specified file into memory. You may now type commands to alter, display, or execute the contents of the specified file.

If you do *not* enter a filename, you must either work with the present memory contents, or load the required file into memory by using the Name and Load commands. Then you can type commands to alter, display, or execute the memory contents.

The optional parameters, *parm1* and *parm2*, represent the optional parameters for the named *filespec*. For example,

```
DEBUG DISKCOMP.COM A: B:
```

In this command, the A: and B: are the parameters that DEBUG prepares for the DISKCOMP program.

When the DEBUG program starts, the registers and flags are set to the following values for the program being debugged:

- The segment registers (CS, DS, ES, and SS) are set to the bottom of free memory; that is, the first segment after the end of the DEBUG program.
- The Instruction Pointer (IP) is set to hex 0100.
- The Stack Pointer (SP) is set to the end of the segment, or the bottom of the transient portion of the program loader, whichever is lower. The segment size at offset 6 is reduced by hex 100 to allow for a stack that size.

- The remaining registers (AX, BX, CX, DX, BP, SI, and DI) are set to zero. However, if you start the DEBUG program with a filespec, the CX register contains the length of the file in bytes. If the file is greater than 64K, the length is contained in registers BX and CX (the high portion in BX).

- The initial state of the flags is:

```
NV UP EI PL NZ NA PO NC
```

- The default disk transfer address is set to hex 80 in the code segment.

All of available memory is allocated; therefore, any attempt by the loaded program to allocate memory fails.

Notes:

1. If a file loaded by DEBUG has an extension of .EXE, DEBUG does the necessary relocation and sets the segment registers, stack pointer, and Instruction Pointer to the values defined in the file. The DS and ES registers, however, point to the Program Segment Prefix at the lowest available segment. The BX and CX registers contain the size of the program (smaller than the file size).

   The program is loaded at the high end of memory if the appropriate parameter was specified when the linker created the file. Refer to ".EXE File Structure and Loading" in Chapter 9 of the *DOS Technical Reference* manual for more information about loading .EXE files.

2. If a file loaded by DEBUG has an extension of .HEX, the file is assumed to be in INTEL hex format, and is converted to executable form while being loaded.

The DEBUG Command Parameters

Parameter	Definition
address	Enter a one- or two-part designation in one of the following formats: • An alphabetic segment register designation, plus an offset value, such as: CS:0100 • A segment address, plus an offset value, such as: 4BA:0100 • An offset value only, such as: 100 (In this case, each command uses a default segment.) **Note**: 1. In the first two formats, the colon is required to separate the values.

Parameter	Definition
address	2. All numeric values are *hexadecimal* and may be entered as 1-4 characters. 3. The memory locations specified in address must be valid; that is, they must actually exist. Unpredictable results occur if an attempt is made to access a nonexistent memory location.
byte	Enter a 1 or 2 character *hexadecimal* value.
drive	Enter 1 or 2 digits (for example, 0 for drive A or 1 for drive B) to indicate which drive data is to be loaded from or written to. (Refer to the Load and Write commands.)
filespec	Enter a one- to three-part file specification consisting of a drive designation, filename, and filename extension. All three fields are optional. However, for the Name command to be meaningful, you should at least specify a drive designator or a filename. (Refer to the Name command.)
list	Enter 1 or more byte and/or string values. For example, `F3 'XYZ' 8D 4 "abcd:"` has five items in the list (that is, three byte entries and two string entries having a total of 10 bytes).

Parameter	Definition
portaddress	Enter a 1-4 character *hexadecimal* value to specify an 8- or 16-bit port address. (Refer to the Input and Output commands.)
range	Enter either of the following formats to specify the lower and upper addresses of a range: • *address address* For example: CS:100 110 **Note:** Only an offset value is allowed in the second address. The addresses must be separated by a space or comma.

Parameter	Definition
range	• *address* L *value* where *value* is the number of bytes in *hexadecimal* to be processed by the command. For example: CS:100 L 11 **Notes:** 1. The limit for *range* is hex 10000, so the sum of *value* and the offset part of *address* cannot exceed 64K bytes. To specify a *value* of 64K bytes within four *hexadecimal* characters, enter 0000 (or 0). 2. The memory locations specified in *range* must be valid; that is, they must actually exist. Unpredictable results will occur if an attempt is made to access a non-existent memory location.
registername	Refer to the Register command.

Parameter	Definition
sector sector	Enter 1-3 character *hexadecimal* values to specify: 1. The starting relative sector number 2. The number of sector numbers to be loaded or written In DEBUG, relative sectors are obtained by counting the sectors on the disk surface. The sector at track 0, sector 1, head 0 (the first sector on the disk) is relative sector 0. The numbering continues for each sector on that track and head, then continues with the first sector on the next head of the same track. When all sectors on all heads of the track have been counted, numbering continues with the first sector on head 0 of the next track. **Note:** This is a change from the sector mapping used by DOS Version 1.10. The maximum number of sectors that can be loaded or written with a single command is hex 80. A sector contains 512 bytes. (Refer to the Load and Write commands.)

Parameter	Definition
string	Enter characters enclosed in quotation marks. The quotation marks can be either single (') or double (" ").
	The ASCII values of the characters in the string are used as a list of byte values.
	Within a string, the *opposite* set of quotation marks can be used freely as characters. However, if the *same* set of quotation marks (as the delimiters) must be used within the string, then the quotation marks must be doubled. The doubling does not appear in memory. For example:
	1. 'This "literal" is correct'
	2. 'This ' 'literal' ' is correct'
	3. 'This 'literal' is not correct'
	4. 'This "'literal"' is not correct'
	5. "This 'literal' is correct"
	6. "This "'literal"' is correct"
	7. "This 'literal' is not correct"
	8. "This ' 'literal' ' is not correct"
	In the second and sixth cases above, the word *literal* is enclosed in one set of quotation marks in memory. In the fourth and eighth cases above, the word *literal* is not correct unless you really want it enclosed in two sets of quotation marks in memory.

Parameter	Definition
value	Enter a 1-4 character *hexadecimal* value to specify: • The numbers to be added and subtracted (refer to the Hexarithmetic command), or • The number of instructions to be executed by the Trace command, or • The number of bytes a command should operate on. (Refer to the Trace, Proceed, and Hexarithmetic commands.)

The DEBUG Commands

This section presents a detailed description of how to use the commands to the DEBUG program. The commands appear in alphabetic order; each with its format and purpose. Examples are provided where appropriate.

Information Common to All DEBUG Commands

The following information applies to the DEBUG commands:

- A command is a single letter, usually followed by one or more parameters.
- Commands and parameters can be entered in uppercase or lowercase, or a combination of both.
- Commands and parameters may be separated by delimiters. Delimiters are only required, however, between two consecutive hexadecimal values. Thus, these commands are equivalent:

```
dcs:100 110
d cs:100 110
d,cs:100,110
```

- Press Ctrl-Break to end commands.
- Commands become effective only after you press the Enter key.
- For commands producing a large amount of output, you can press Ctrl-NumLock to suspend the display to read it before it scrolls away. Press any other character to restart the display.

- You can use the control keys and the DOS editing keys, described in Chapter 3 of *DOS User's Guide* while using the DEBUG program.

- If a syntax error is encountered, the line is displayed with the error pointed out as follows:

```
d cs:100 CS:110
             ^error
```

  In this case, the Dump command is expecting the second address to contain only a hexadecimal offset value. It finds the S, which is not a valid hexadecimal character.

- The prompt from the DEBUG program is a hyphen (-).

- The DEBUG program resides on your DOS Supplemental Program diskette.

A (Assemble) Command

Purpose: To assemble IBM Personal Computer Macro Assembler language statements directly into memory.

Format: A[*address*]

Remarks: All numeric input to the Assemble command is in hexadecimal. The assembly statements you enter are assembled into memory at successive locations, starting with the address specified in *address*. If no address is specified, the statements are assembled into the area at CS:0100, if no previous Assemble command was used, or into the location following the last instruction assembled by a previous Assemble command. When all desired statements have been entered, press Enter when you are prompted for the next statement, to return to the DEBUG prompt.

DEBUG responds to invalid statements by displaying:

```
^error
```

and redisplaying the current assemble address.

DEBUG supports standard 8086/8088 assembly language syntax (and the 8087 instruction set), with the following rules:

- All numeric values entered are hexadecimal and can be entered as 1-4 characters.
- Prefix mnemonics must be entered in front of the opcode to which they refer. They can also be entered on a separate line.

- The segment override mnemonics are CS:, DS:, ES:, and SS:.

- String manipulation mnemonics must explictly state the string size. For example, MOVSW must be used to move word strings and MOVSB must be used to move byte strings.

- The mnemonic for the far return is RETF.

- The assembler will automatically assemble short, near, or far jumps and calls depending on byte displacement to the destination address. These can be overridden with the NEAR OR FAR prefix. For example:

```
0100:0500 JMP 502          ;a 2 byte short jump
0100:0502 JMP NEAR 505     ;a 3 byte near jump
0100:0505 JMP FAR 50A      ;a 5 byte far jump
```

  The NEAR prefix can be abbreviated to NE, but the FAR prefix cannot be abbreviated.

- DEBUG cannot tell whether some operands refer to a word memory location or a byte memory location. In this case, the data type must be explicity stated with the prefix WORD PTR or BYTE PTR. DEBUG will also accept the abbreviations WO and BY. For example:

```
NEG   BYTE PTR [128]
DEC   WO [SI]
```

- DEBUG also cannot tell whether an operand refers to a memory location or to an immediate operand. DEBUG uses the common convention that operands enclosed in square brackets refer to memory. For example:

```
MOV    AX,21     ;Load AX with 21H
MOV    AX,[21]   ;Load AX with the
                  contents of
                  memory location
                  21H
```

- Two popular pseudo-instructions have also been included. The DB opcode assembles byte values directly into memory. The DW opcode assembles word values directly into memory. For example:

```
DB    1,2,3,4,"THIS IS AN EXAMPLE"
DB    "THIS IS A QUOTE: "'
DB    "THIS IS A QUOTE:'"

DW    1000,2000,3000:", BACH:"
```

- All forms of the register indirect commands are supported. For example:

```
ADD   BX,34[BP+2].[SI-1]
POP   [BP+DI]
PUSH  [SI]
```

- All opcode synonyms are supported. For example:

```
LOOPZ   100
LOOPE   100

JA      200
JNBE    200
```

A (Assemble) Command

- For 8087 opcodes the WAIT or FWAIT prefix must be explictly specified. For example:

```
FWAIT FADD ST,ST(3)     ;This line will assemble
                        a FWAIT prefix
FLD TBYTE PTR [BX]      ;This line will not
```

Example:

```
C>debug
-a200
08B4:0200 xor ax,ax
08B4:0202 mov [bx],ax
08B4:0204 ret
08B4:0205
```

C (Compare) Command

Purpose: Compares the contents of two blocks of memory.

Format: C *range address*

Remarks: The contents of the two blocks of memory are compared; the length of the comparison is determined from the *range*. If unequal bytes are found, their addresses and contents are displayed, in the form:

```
addr1  byte1  byte2  addr2
```

where, the first half (addr1 byte1) refers to the location and contents of the mismatching locations in *range*, and the second half (byte2 addr2) refers to the byte found in *address*.

If you enter only an offset for the beginning address of *range*, the C command assumes the segment contained in the DS register. To specify an ending address for *range*, enter it with only an offset value.

Example:

```
C 100 L20 200
```

The 32 bytes (hex 20) of memory beginning at DS:100 are compared with the 32 bytes beginning at DS:200. L20 is the range.

D (Dump) Command

Purpose: Displays the contents of a portion of memory.

Format:

D [*address*]

or

D [*range*]

Remarks: The dump is displayed in two parts:

1. A hexadecimal portion. Each byte is displayed in hexadecimal.

2. An ASCII portion. The bytes are displayed as ASCII characters. Unprintable characters are indicated by a period (.).

With a 40-column system display format, each line begins on an 8-byte boundary and shows 8 bytes.

With an 80-column system display format, each line begins on a 16-byte boundary and shows 16 bytes. There is a hyphen between the 8th and 9th bytes.

> **Note:** The first line may have fewer than 8 or 16 bytes if the starting address of the dump is not on a boundary. In this case, the second line of the dump begins on a boundary.

D (Dump) Command

The Dump command has two format options:

Option 1

Use this option to display the contents of hex 40 bytes (40-column mode) or hex 80 bytes (80-column mode). For example:

```
D address
```

or

```
D
```

The contents are dumped starting with the specified address.

If you do not specify an address, the D command assumes the starting address is the location following the last location displayed by a previous D command. Thus, it is possible to dump consecutive 40-byte or 80-byte areas by entering consecutive D commands without parameters.

If no previous D command was entered, the location is offset hex 100 into the segment originally initialized in the segment registers by DEBUG.

> **Note:** If you enter only an offset for the starting address, the D command assumes the segment contained in the DS register.

D (Dump) Command

Option 2

Use this option to display the contents of the specified address range. For example:

```
D range
```

> **Note:** If you enter only an offset for the starting address, the D command assumes the segment contained in the DS register. If you specify an ending address, enter it with only an offset value.

For example:

```
D cs:100 10C
```

A 40-column display format might look like this:

```
04BA:0100  42 45 52 54 41 20 54 00
                  BERTA T.

04BA:0108  20 42 4F 52 47
                     BORG
```

E (Enter) Command

Purpose: The Enter command has two modes of operation:

- Replaces the contents of one or more bytes, starting at the specified address, with the values contained in the list. (See Option 1.)
- Displays and allows modification of bytes in a sequential manner. (See Option 2.)

Format: E *address* [*list*]

Remarks: If you enter only an offset for the address, the E command assumes the segment contained in the DS register.

The Enter command has two format options:

Option 1

Use this option to place the list in memory beginning at the specified address.

```
E  address list
```

For example:

```
E ds:100 F3 "xyz" 8D
```

Memory locations ds:100 through ds:104 are filled with the 5 bytes specified in the list.

Option 2

Use this option to display the address and the byte of a location, then the system waits for your input.

E (Enter) Command

For example:

```
E  address
```

Enter a 1- or 2-character *hexadecimal* value to replace the contents of the byte; then take any of the next three actions:

1. Press the space bar to advance to the next address. Its contents are displayed. If you want to change the contents take action 1, above.

   To advance to the next byte without changing the current byte, press the space bar again.

2. Enter a hyphen (-) to back up to the preceding address. A new line is displayed with the preceding address and its contents. If you want to change the contents take option 1, above.

   To back up one more byte without changing the current byte, enter another hyphen.

3. Press the Enter key to end the Enter command.

   Note: Display lines can have 4 or 8 bytes of data, depending on whether the system display format is 40- or 80-column. Spacing beyond an 8-byte boundary causes a new display line, with the beginning address, to be started.

For example:

```
E cs:100
```

might cause this display:

```
04BA:0100 EB._
```

To change the contents of 04BA:0100 from hex EB to hex 41, enter 41.

```
04BA:0100 EB.41_
```

To see the contents of the next three locations, press the space bar three times. The screen might look like this:

```
04BA:0100  EB.41  10. 00.  BC._
```

To change the contents of the current location (04BA:0103) from hex BC to hex 42, enter 42.

```
04BA:0100  EB.41  10. 00.  BC.42_
```

Now, suppose you want to back up and change the hex 10 to hex 6F. This is what the screen would look like after entering two hyphens and the replacement byte:

```
04BA:0100 EB.41 10.00.  BC.42-
04BA:0102 00.-
04BA:0101 10.6F_
```

Press the Enter key to end the Enter command. You will see the hyphen (-) prompt.

F (Fill) Command

Purpose: Fills the memory locations in the range with the values in the list.

Format: F *range list*

Remarks: If the list contains fewer bytes than the address range, the list is used repeatedly until all the designated memory locations are filled.

If the list contains more bytes than the address range, the extra list items are ignored.

> **Note:** If you enter only an offset for the starting address of the range, the Fill command assumes the segment contained in the DS register.

Example: F 4BA:100 L 5 F3 "XYZ" 8D

Memory locations 04BA:100 through 04BA:104 are filled with the 5 bytes specified. Remember that the ASCII values of the list characters are stored. Thus, locations 100-104 will contain F3 58 59 5A 8D.

G (Go) Command

Purpose: Executes the program you are debugging.

Stops the execution when the instruction at a specified address is reached (breakpoint), and displays the registers, flags, and the next instruction to be executed.

Format: G [=*address*] [*address* [*address*...]]

Remarks: Program execution begins with the current instruction, whose address is determined by the contents of the CS and IP registers, unless overridden by the =*address* parameter (the = must be entered). If =*address* is specified, program execution begins with CS:=*address*.

The Go command has two format options:

Option 1

Use this option to execute the program you are debugging without breakpoints. For example:

```
G [=address]
```

This option is useful when testing program execution with different parameters each time. (Refer to the Name command.) Be certain the CS:IP values are set properly before issuing the G command, if not using =*address*.

G (Go) Command

Option 2

This option performs the same function as Option 1 but, in addition, allows breakpoints to be set at the specified addresses. For example:

```
G [=address] address
  [address...]
```

This method causes execution to stop at a specified location so the system/program environment can be examined.

You can specify up to ten breakpoints in any order. You may wish to take advantage of this if your program has many paths, and you want to stop the execution no matter which path the program takes.

The DEBUG program replaces the instruction codes at the breakpoint addresses with an interrupt code (hex CC). If *any one* breakpoint is reached during execution, the execution is stopped, the registers and flags are displayed, and all the breakpoint addresses are restored to their original instruction codes. If no breakpoint is reached, the instructions are *not* restored.

Notes:

1. Once a program has reached completion (DEBUG has displayed the "Program terminated normally" message), it is necessary to reload the program before it can be executed again.

2. Make sure that the address parameters refer to locations that contain valid 8088 instruction

codes. If you specify an address that does not contain the first byte valid instruction, unpredictable results occur.

3. The stack pointer must be valid and have 6 bytes available for the Go command; otherwise, unpredictable results occur.

4. If only an offset is entered for a breakpoint, the G command assumes the segment contained in the CS register.

5. Do not set breakpoints at instructions in read-only memory (ROM BIOS or ROM BASIC).

For example:

```
G 102 1EF 208
```

Execution begins with the current instruction, whose address is the current values of CS:IP. The *=address* parameter was not used.

Three breakpoints are specified; assume that the second is reached. Execution stops before the instruction at location CS:1EF is executed, the original instruction codes are restored, all three breakpoints are removed, the display occurs, and the Go command ends.

Refer to the Register command for a description of the display.

H (Hexarithmetic) Command

Purpose: Adds the two hexadecimal values, then subtracts the second from the first.

Displays the sum and difference on one line.

Format: H *value value*

Example:

```
H 0F 8
17 07
```

The hexadecimal sum of 000F and 0008 is 0017, and their difference is 0007.

I (Input) Command

Purpose: Inputs and displays (in hexadecimal) 1 byte from the specified port.

Format: I *portaddress*

Example:

```
I 2F8
6B
```

The single hexadecimal byte read from port 02F8 is displayed (6B).

L (Load) Command

Purpose: Loads a file or absolute disk sectors into memory.

Format: L [*address* [*drive sector sector*]]

Remarks: The maximum number of sectors that can be loaded with a single Load command is hex 80.

> **Note:** DEBUG displays a message if a disk read error occurs. You can retry the read operation by pressing F3 to re-display the Load command. Then, press the Enter key.

The Load command has two format options:

Option 1

Use this option to load data from the disk specified by *drive*, and place the data in memory beginning at the specified *address*. For example:

```
L  address drive sector sector
```

The data is read from the specified starting relative sector (first sector) and continues until the requested number of sectors is read (second sector).

> **Note:** If you only enter an offset for the beginning address, the L command assumes the segment contained in the CS register.

For example, to load data, you might enter:

```
L DS:100 1 0F 6D
```

The data is loaded from the diskette in drive B and placed in memory beginning at DS:100. 6DH (109) consecutive sectors of data are transferred, starting with relative sector hex 0F (15) (the 16th sector on the diskette).

> **Note:** Option 1 cannot be used if the drive specified is a network drive.

Option 2

When issued without parameters, or with only the address parameter, use this option to load the file whose filespec is at CS:80. For example:

```
L
```

or

```
L address
```

This condition is met by specifying the filespec when starting the DEBUG program, or by using the Name command.

> **Note:** If DEBUG was started with a filespec and subsequent Name commands were used, you may need to enter a new Name command for the proper filespec before issuing the Load command.

The file is loaded into memory beginning at CS:100 (or the location specified by *address*), and is read from the drive specified in the filespec (or from the default drive, if none was specified). Note that files with extensions of .COM or .EXE are always loaded at CS:100–if you specified an address, it is ignored.

L (Load) Command

The BX and CX registers are set to the number of bytes read; however, if the file being loaded has an extension of .EXE, BX and CX are set to the actual program size. The file may be loaded at the high end of memory. Refer to the notes in "How to Start the DEBUG Program" at the beginning of this chapter for the conditions that are in effect when .EXE or .HEX files are loaded.

For example:

```
DEBUG
-N myprog
-L
-
```

The file named **myprog** is loaded from the default diskette and placed in memory beginning at location CS:0100.

M (Move) Command

Purpose: Moves the contents of the memory locations specified by *range* to the locations beginning at the *address* specified.

Format: M *range address*

Remarks: Overlapping moves are always performed without loss of data during the transfer. (The source and destination areas share some of the same memory locations.)

The data in the source area remains unchanged unless overwritten by the move.

Notes:

1. If you enter only an offset for the beginning address of the range, the M command assumes the segment contained in the DS register. If you specify an ending address for the range, enter it with only an offset value.

2. If you enter only an offset for the address of the destination area, the M command assumes the segment contained in the DS register.

Example:

```
M CS:100 110 500
```

The 17 bytes of data from CS:100 through CS:110 are moved to the area of memory beginning at DS:500.

N (Name) Command

Purpose: The Name command has two functions:

- Formats file control blocks for the first two filespecs, at CS:5C and CS:6C. (Starting DEBUG with a filespec also formats a file control block at CS:5C.)

  The file control blocks are set up for the use of the Load and Write commands, and to supply required filenames for the program being debugged.

- All specified filespecs and other parameters are placed exactly as entered, including delimiters, in a parameter save area at CS:81, with CS:80 containing the number of characters entered. Register AX is set to indicate the validity of the drive specifiers entered with the first two filespecs.

Format: N [d:][path]*filename*[*.ext*]

Remarks: If you start the DEBUG program without a filespec, you must use the Name command before a file can be loaded with the L command.

Example:

```
DEBUG
-N myprog
-L
-
```

To define filespecs or other parameters required by the program being debugged, enter:

```
DEBUG myprog
-N file1 file2
-
```

In this example, DEBUG loads the file **myprog** at CS:100, and leaves the file control block at CS:5C formatted with the same filespec. Then, the Name command formats file control blocks for *file1* and *file2* at CS:5C and CS:6C, respectively. The file control block for **myprog** is overwritten. The parameter area at CS:81 contains all characters entered after the N, including all delimiters, and CS:80 contains the count of those characters (hex 0C).

O (Output) Command

Purpose: Sends the *byte* to the specified output port.

Format: O *portaddress byte*

Example: To send the byte value 4F to output port 2F8, enter:

```
O 2F8 4F
```

P (Proceed) Command

Purpose: Causes the execution of a subroutine call, a loop instruction, an interrupt, or a repeat string instruction to stop at the next instruction.

Format: P[=address][value]

Remarks: When at a subroutine call, a loop instruction, an interrupt, or a repeat string instruction, issue the Proceed command to execute the instruction (as an atomic operation), and return control at the next instruction. The Proceed command has the same syntax as the Trace command. Specifing P0, is the same as specifying T0.

Example: If the following instructions are executed:

```
0100  CALL 10000103  JC   2000
.
.
.
1000  XOR AX,AX
.
.
.
1XXX  RET
```

And CS:IP was pointing to the CALL 1000 instruction, typing **P** causes the execution of the subroutine and returns control to DEBUG at the JC instruction.

Q (Quit) Command

Purpose: Ends the DEBUG program.

Format: Q

Remarks: The file that you are working on in memory is *not* saved by the Quit command. You must use the Write command to save the file.

DEBUG returns to the command processor which then issues the normal command prompt.

Example:

```
-Q
A>
```

R (Register) Command

Purpose: The Register command has three functions:

- It displays the hexadecimal contents of a single register, with the option of changing those contents.
- It displays the hexadecimal contents of all the registers, plus the alphabetic flag settings, and the next instruction to be executed.
- It displays the eight 2-letter alphabetic flag settings, with the option of changing any or all of them.

Format: R [*registername*]

Remarks: When the DEBUG program starts, the registers and flags are set to certain values for the program being debugged. (Refer to "How to Start the DEBUG Program" at the beginning of this chapter.)

Display a Single Register

The valid *registernames* are:

```
AX    BP    SS
BX    SI    CS
CX    DI    IP
DX    DS    PC
SP    ES    F
```

Both IP and PC refer to the instruction pointer.

For example, to display the contents of a single register, you might enter:

```
R AX
```

The system might respond with:

```
AX F1E4
:_
```

Now you may take one of two actions:

- Press Enter to leave the contents unchanged.

  or

- Change the contents of the AX register by entering a 1-4 character hexadecimal value, such as hex FFF.

```
AX F1E4
:FFF_
```

Now pressing Enter changes the contents of the AX register to hex 0FFF.

Display All Registers and Flags

To display the contents of all registers and flags (and the next instruction to be executed), type:

```
R
```

The system might respond with:

```
AX=0E00 BX=00FF CX=0007 DX=01FF
SP=039D BP=0000 SI=005C DI=0000
DS=04BA ES=04BA SS=04BA CS=04BA
IP=011A  NV UP DI NG NZ AC PE NC
04BA:011A  CD21          INT     21
```

The first four lines display the hexadecimal contents of the registers and the eight alphabetic flag settings. The last line indicates the location of the next instruction to be executed, and its hexadecimal and unassembled formats. This is the instruction pointed to by CS:IP.

Note: A system with an 80-column display shows:

1st line - 8 registers

2nd line - 5 registers and 8 flag settings

3rd line - next instruction information

A system with a 40-column display shows:

1st line - 4 registers

2nd line - 4 registers

3rd line - 4 registers

4th line - 1 register and 8 flag settings

5th line - next instruction information

Display All Flags

There are eight flags, each with 2-letter codes to indicate either a *set* condition or a *clear* condition.

The flags appear in displays in the same order as presented in the following table:

R (Register) Command

Flag Name	Set	Clear
Overflow (yes/no)	OV	NV
Direction (decrement/increment)	DN	UP
Interrupt (enable/disable)	EI	DI
Sign (negative/positive)	NG	PL
Zero (yes/no)	ZR	NZ
Auxiliary carry (yes/no)	AC	NA
Parity (even/odd)	PE	PO
Carry (yes/no)	CY	NC

Figure 10-1 **Alphabetic Flag Settings**

To display all flags, enter:

```
R F
```

If all the flags are in a *set* condition, the response is:

```
OV DN EI NG ZR AC PE CY -_
```

Now you can take one of two actions:

1. Press Enter to leave the settings unchanged.
2. Change any or all of the settings.

To change a flag, just enter its opposite code. The opposite codes can be entered in any order—with or without intervening spaces. For example, to change the first, third, fifth, and seventh flags, enter:

```
OV DN EI NG ZR AC PE CY - PONZDINV
```

They are entered in reverse order in this example.

Press Enter and the flags are modified as specified, the prompt appears, and you can enter the next command.

If you want to see if the new codes are in effect, enter:

```
R F
```

The response is:

```
NV DN DI NG NZ AC PO CY -_
```

The first, third, fifth, and seventh flags are changed as requested. The second, fourth, sixth, and eighth flags are unchanged.

> **Note:** A single flag can be changed only once per R F command.

S (Search) Command

Purpose: Searches the *range* for the character(s) in the *list*.

Format: S *range list*

Remarks: All matches are indicated by displaying the addresses where matches are found.

A display of the prompt (-) without an address means that no match was found.

> **Note:** If you enter only an offset for the starting address of the range, the S command assumes the segment contained in the DS register.

Example: If you want to search the range of addresses from CS:100 through CS:110 for hex 41, type:

```
S CS:100 110 41
```

If two matches are found the response might be:

```
04BA:0104
04BA:010D
```

If you want to search the same range of addresses as in the previous example for a match with the 4-byte-long list, enter:

```
S CS:100 L 11 41 "AB" E
```

The starting addresses of all matches are listed. If no match is found, no address is displayed.

T (Trace) Command

Purpose: Executes one or more instructions starting with the instruction at CS:IP, or at =*address* if it is specified. The = must be entered. One instruction is assumed, but you can specify more than one with *value*. Displays the contents of all registers and flags *after each* instruction executes. For a description of the display format, refer to the Register command.

Format: T [=*address*][*value*]

Remarks: The display caused by the Trace command continues until *value* instructions are executed. Therefore, when tracing multiple instructions, remember you can suspend the scrolling at any time by pressing Ctrl-NumLock. Resume scrolling by entering any other character.

Notes:

1. The Trace command disables all hardware interrupts before executing the user instruction, and then reenables the interrupts when the trap interrupt occurs following the execution of the instruction.

2. TRACE should not be used with any steps that change the contents of the 8259 interrupt mask (ports 20 and 21).

3. If you trace an INT3 instruction, the breakpoint is set at the INT3 location.

T (Trace) Command

Example: T

If the IP register contains 011A, and that location contains B40E (MOV AH,0EH), this might be displayed:

```
AX=0E00 BX=00FF CX=0007 DX=01FF
SP=039D BP=0000 SI=005C DI=0000
DS=04BA ES=04BA SS=04BA CS=04BA
IP=011C  NV UP DI NG NZ AC PE NC
04BA:011C  CD21     INT     21
```

This displays the results *after* the instruction at 011A is executed, and indicates the next instruction to be executed is the INT 21 at location 04BA:011C.

```
T 10
```

Sixteen instructions are executed (starting at CS:IP). The contents of all registers and flags are displayed after each instruction. The display stops after the 16th instruction has been executed. Displays may scroll off the screen unless you suspend the display by pressing the Ctrl-NumLock keys.

U (Unassemble) Command

Purpose: Unassembles instructions (translates the contents of memory into assembler-like statements) and displays their addresses and hexadecimal values, together with assembler-like statements. For example, a display might look like this:

```
04BA:0100  206472    AND [SI+72],AH
04BA:0103  FC        CLD
04BA:0104  7665      JBE 016B
```

Format: U [*address*]

or

U [*range*]

Remarks: The number of bytes unassembled depends on your system display format (whether 40 or 80 columns), and which option you use with the Unassemble command.

Notes:

1. In all cases, the number of bytes unassembled and displayed may be slightly more than either the amount requested or the default amount. This happens because the instructions are of variable lengths; therefore, to unassemble the last instruction may include more bytes than expected.

2. Make sure that the address parameters refer to locations containing valid 8088 instruction codes. If you specify an address that does not contain the first byte of a valid instruction, the display will be erroneous.

3. If you enter only an offset for the starting address, the U command assumes the segment contained in the CS register.

The Unassemble command has two format options:

Option 1

Use this option to either unassemble instructions without specifying an address, or to unassemble instructions beginning with a specified address. For example:

```
U
```

or

```
U address
```

Sixteen bytes are unassembled with a 40-column display. Thirty-two bytes are unassembled while in 80-column mode.

Instructions are unassembled beginning with the specified address.

If you do not specify an address, the U command assumes the starting address is the location following the last instruction unassembled by a previous U command. Thus, it is possible to unassemble consecutive locations, producing continuous unassembled displays, by entering consecutive U commands without parameters.

If no previous U command is entered, the location is offset hex 0100 into the segment originally initialized in the segment registers by DEBUG.

Option 2

Use this option to unassemble instructions in a specified address range. For example:

```
U range
```

All instructions in the specified address range are unassembled, regardless of the system display format.

> **Note:** If you specify an ending address, enter it with only an offset value.

For example:

```
U04ba:0100 108
```

The display response might be:

```
04BA:0100  206472      AND [SI+72],AH
04BA:0103  FC          CLD
04BA:0104  7665        JBE 016B
04BA:0106  207370      AND [BP+DI+70],DH
```

The same display appears if you enter:

```
U04BA:100 L 7
```

or

```
U 04BA:100 L 8
```

or

```
U 04BA:100 L 9
```

W (Write) Command

Purpose: Writes the data being debugged to disk.

Format: W [*address* [*drive sector sector*]]

Remarks: The maximum number of sectors that can be written with a single Write command is hex 80.

DEBUG displays a message if a disk write error occurs. You can retry the write operation by pressing F3 to redisplay the Write command, then press the Enter key.

The Write command has two format options:

Option 1

Use this option to write data to disk beginning at a specified address. For example:

```
W address drive sector sector
```

The data beginning at the specified address is written to the disk in the indicated drive. The data is written starting at the specified starting relative sector (first sector) and continues until the requested number of sectors are filled (second sector).

Notes:

1. Be extremely careful when you write data to absolute sectors because an erroneous sector specification destroys whatever was on the disk at that location.

2. If only an offset is entered for the beginning address, the W command assumes the segment contained in the CS register.

3. Remember, the starting sector and the sector count are both specified in *hexadecimal.*

4. Option 1 cannot be used if the drive specified is a network drive.

For example:

```
W 1FD 1 100 A
```

The data beginning at CS:01FD is written to the diskette in drive B, starting at relative sector hex 100 (256) and continuing for hex 0A (10) sectors.

Option 2

This option allows you to use the Write command without specifying parameters or only specifying the address parameter. For example:

```
W
```

or

```
W address
```

When issued as shown above, the Write command writes the file (whose filespec is at CS:80) to disk.

This condition is met by specifying the filespec when starting the DEBUG program, or by using the Name command.

> **Note:** If DEBUG was started with a filespec and subsequent Name commands were used,

you may need to enter a new Name command for the proper filespec before issuing the Write command.

In addition, the BX and CX registers must be set to the number of bytes to be written. They may have been set properly by the DEBUG or Load commands, but might have been changed by a Go or Trace command. You must be certain the BX and CX registers contain the correct values.

The file beginning at CS:100, or at the location specified by *address*, is written to the diskette in the drive specified in filespec or the default drive if none was specified.

The debugged file is written over the original file that was loaded into memory, or into a new file if the filename in the FCB didn't previously exist.

Note: An error message is issued if you try to write a file with an extension of .EXE or .HEX. These files must be written in a specific format that DEBUG cannot support.

If you find it necessary to modify a file with an extension of .EXE or .HEX, and the exact locations to be modified are known, use the following procedure:

1. RENAME the file to an extension other than .EXE or .HEX.

2. Load the file into memory using the DEBUG or Load command.

3. Modify the file as needed in memory, but do not try to execute it with the Go or Trace commands. Unpredictable results would occur.

4. Write the file back using the Write command.

5. RENAME the file to its correct name.

Notes:

Appendixes

Contents

Appendix A. Messages **A-3**

Introduction **A-3**
Responses A-3

Device Error Messages **A-4**

Other Messages **A-11**

Notes:

Appendix A. Messages

Introduction

This appendix contains *device errors* (the message that DOS uses to indicate errors while reading or writing to devices on your system), and *Other messages* (the remainder of the DOS messages) in alphabetic order. Messages are listed in **bold** type and the explanation and action follow the message.

The first word of the description of each message is the name of the program or command that generated the message.

In some cases, the message is generated by several different programs or commands. In that case, the first word is **command**. Where the message is generated by an internal DOS file, the first word is **DOS**.

Responses

When any of the device error messages are displayed, the system waits for you to respond. If you know what caused the problem, take corrective action before choosing a response. The system waits until you make *one* of these responses.

To recover from an error condition, the responses should be made in the following order:

R **Retry** the operation because the error may not occur again. The system tries the disk read or write operation again. We strongly recommend that you use **Retry** first.

A **Abort** the program. The system ends the program that requested the disk read or write.

I **Ignore** the error condition and continue the program. (Be careful when choosing this response because data may be lost.) If the system determines that the condition is not dangerous, it ignores the error and continues the program.

Device Error Messages

When an error is detected during reading or writing to any of the devices (disk drives, printer, etc.) on your system, DOS displays a message in the following format:

<*type*> error reading <*device*>
Abort, Retry, Ignore?
or
<*type*> error writing <*device*>
Abort, Retry, Ignore?

In these messages, <*device*> is the name of the device in error, such as **PRN**, or **B**:, and <*type*> is one of the types listed on the following pages:

Bad call format

Explanation: A device driver was passed an incorrect length request header.

Action: Refer to "Responses" at the beginning of this group of device messages.

- Use DEBUG.
- Review your programming specifications. Patch and reassemble.
- If you are using a purchased program, contact the dealer you purchased the device driver from.

Bad command

Explanation: A device driver has issued an invalid command to <*device*>.

Action: Refer to "Responses" at the beginning of this group of device messages.

- Review your device interface specification and DOS driver implementation to make sure everything you are trying to do is supported.
- Check your program to see if you have a coding problem that needs debugging.

Bad unit

Explanation: A device driver has been passed an invalid sub–unit number.

Action: Refer to "Responses" at the beginning of this group of device messages. If you are using a purchased program contact the dealer you purchased the device driver from.

Data

Explanation: DOS was unable to read or write the data correctly. This message usually means a disk has developed a defective spot.

Action: Refer to "Responses" at the beginning of this group of device messages.

FCB unavailable

Explanation: With file sharing loaded, you tried to concurrently open more file control blocks than were specified by the FCBS configuration command.

Action: Choose **Abort** as your response. Then, increase the value of the FCBS configuration command.

General failure

Explanation: An error of a type not described elsewhere in this list has occurred.

Action: Refer to "Responses" at the beginning of this group of device messages. Choose **Retry** first. Then choose **Abort** if this problem requires further investigation by a programmer.

If you are using a purchased program, contact the dealer you purchased it from.

Lock violation

Explanation: SHARE. The file you tried to access is locked by someone else. This is usually a temporary situtation.

Action: Choose **Retry** as your first response. If the problem still exists, choose **Abort**.

No paper

Explanation: The indicated printer is either out of paper or not turned on.

Action: Turn the printer ON, press the ONLINE switch, or add paper and retry.

Refer to "Responses" at the beginning of this group of device messages.

Non-DOS disk

Explanation: The file allocation table contains invalid information. The disk needs to be reformatted.

Action: Refer to "Responses" at the beginning of this group of device messages.

Try running CHKDSK to see if any corrective action is possible. Reformatting will correct the disk, but the files are lost forever.

Not ready

Explanation: The named device is not ready to accept or transmit data.

Action: Check that the disk drive door is closed and choose **Retry** for your response if this is the problem.

Refer to "Responses" at the beginning of this group of device messages.

Read fault

Explanation: DOS was unable to read the data from the device.

Action: Refer to "Responses" at the beginning of this group of device messages.

- Make sure the diskette is properly inserted in the drive.
- If you get the same message, choose **Abort** and rerun the command with a different disk.

Sector not found

Explanation: The sector containing the data could not be located on the disk.

Action: Refer to "Responses" at the beginning of this group of device messages.

If you get the same message, choose **Abort** and rerun the command with a different disk.

Seek

Explanation: The fixed disk or diskette drive was unable to locate the proper track on the disk.

Action:

- Make sure the diskette is properly inserted in the drive.
- Try a different drive
- Run CHKDSK

Refer to "Responses" at the beginning of this group of device messages.

Sharing violation

Explanation: SHARE. You tried to access a file using a sharing mode not allowed at this time. Normally this occurs when someone else is accessing the file in compatibility mode for writing, or in a sharing mode that doesn't allow you to access the file at the same time.

Action: Choose **Retry** as your first response. If that fails, choose **Abort**.

Write fault

Explanation: DOS was unable to write the data to the device.

Action:

- Make sure the diskette is properly inserted in the drive.
- If the diskette is not the problem, choose **Retry**.
- If you get the same message, choose **Abort** and rerun the command with a different disk.

Refer to "Responses" at the beginning of this group of device messages.

Write protect

Explanation: An attempt was made to write on a write protected diskette.

Action: Investigate carefully before you decide to write on a write protected diskette.

Important: If you attempt to use a dual-sided diskette in a single-sided drive, or if you attempt to use a 9-sector-per-track diskette on a version of DOS previous to version 2.00, one of the

preceding messages appears. If any of the preceding messages appear for a diskette drive, DO NOT change diskettes before responding with **Abort**, **Retry**, or **Ignore**.

Other Messages

The following messages are in alphabetic order.

A

Abort edit (Y/N)?

Explanation: EDLIN. This message is displayed if you specify the /Q command to quit the edit.

Action: Type a **Y** to end the edit. Type a **N** to continue editing the file.

About to generate .EXE file
Change diskette in drive A: and press <ENTER>

Explanation: LINK. This message appears before the .EXE has been written if the /P switch is given.

Action: Insert diskette that the .EXE file is to be written to into the specified drive (A: for example).

Access denied

Explanation: Commands. Executing the requrested command violates the access mode of the file, subdirectory, or device involved. For example, an attempt was made to write to a file marked read-only, or read a file that is write-only. Or an attempt was made to open a subdirectory as a file.

Action: Use a different filename.

All files canceled by operator

Explanation: PRINT/T. Informational message. You use the PRINT command with the /T parameter to cancel the printing of all queued files.

Action: This message appears on the printer.

All specified file(s) are contiguous

Explanation: CHKDSK. Informational message. The file or files you named are all written sequentially on the disk.

Action: No action required.

Allocation error for file, size adjusted

Explanation: CHKDSK or CHKDSK/F. A filename precedes this message. An invalid cluster number was found in the file allocation table.

Action:

- If you specified the /F parameter, the file is truncated at the end of the last valid cluster.
- If you did not enter the /F parameter, the message is for your information and no action is needed. Enter:

  ```
  CHKDSK/F
  ```

  to correct the file size.

Amount read less than size in header

Explanation: EXE2BIN. The program portion of the file was smaller than indicated in the file's header.

Action: Recompile or reassemble the program, and then

reLINK it.

Array element size mismatch

Explanation: LINK. A far communal array has been declared with two or more different array element sizes (for example, declared once as an array of characters and once as an array of reals).

Action: Match definitions and recreate object module.

Attempt to access data outside of segment bounds

Explanation: LINK. An object file is invalid.

Action:

- Review the .ASM file or assembled listing for segmentation violations.
- Look for a bad reference or invalid instruction.

Attempt to put segment *xxxxxx* **in more than one group in file** *xxxxxx*

Explanation: LINK. A segment was declared to be part of two different groups.

Action: Correct the source and recreate the object files.

Attempted write protect violation

Explanation: FORMAT. The diskette being formatted is write protected and cannot be written on.

Action: In response to the displayed prompt, insert a new diskette and press any key to restart formatting.

B

**** Backing up files to target drive *x*****
Target Number: *x*

Explanation: BACKUP. Informational message telling you the sequence of file being backed up.

Action: No action required.

Backup file sequence error

Explanation: RESTORE. A file to be restored was backed up on more than one diskette. You did not insert the diskette with the first part of the file.

Action: Rerun RESTORE and start with the correct diskette.

Bad command or filename

Explanation: DOS. The command you just entered is not a valid DOS command.

Action: Check the spelling of the command and re-enter it.

If the command name is spelled correctly, check to see that the default drive contains the external command or batch file you are trying to execute.

Bad internal reloc table

Explanation: LINK. Internal linker error.

Action: Record the scenario that produced this message, and contact your IBM dealer.

Bad or missing Command Interpreter

Explanation: DOS. The disk that DOS is being started from does not contain a copy of COMMAND.COM, or an error occurred while the disk was being loaded.

This message also appears if COMMAND.COM has been removed from the directory it was in originally when DOS was started; or if the **COMSPEC=** parameter in the environment points to a directory not containing COMMAND.COM and DOS is trying to reload the command processor.

Action: Do a System Reset. If System Reset fails to solve the problem, start DOS with your backup DOS diskette. Then copy COMMAND.COM from the backup diskette to the root directory of the disk that failed.

Bad or missing <filename>

Explanation: DOS. This message appears only at startup and indicates one of the following:

a. The name of a driver named in a **DEVICE=** <filename> parameter in the CONFIG.SYS file was not found.

b. A break address was out of bounds for the machine size.

c. An error occurred while the driver was being loaded. That driver is not installed by DOS.

Action:

a. For an incorrect device driver name, use the correct spelling for the device driver name, or use the diskette with the *named* device driver.

b. For items B and C, correct the coding in the device driver.

c. If you still cannot correct the problem, see your IBM Personal Computer dealer.

Bad Unit

Explanation: Commands. A disk I/O device driver encountered a critical error in or within an unrecognized or invalid device.

Action: Record the environment and sequence of events and notify the author of the device driver or your authorized IBM dealer.

Bad tracks found at start of partition
Partition size adjusted

Explanation: FDISK. Bad tracks were encountered in the area of the fixed disk required by DOS. Some space was lost at the beginning of the requested partition.

Action: No action required.

Batch file missing

Explanation: DOS. DOS could not locate the batch file it was processing. The file may have been erased or renamed by one of the steps within it. Batch processing stops and the DOS prompt appears.

Action:

1. If the filename was changed or if the file was deleted, correct the command that changed the name(s).

2. If the file was erased, use your backup copy. If you used EDLIN to create the file or make changes, rename the .BAK file to .BAT. Correct the command that deleted the file.

BF

Explanation: DEBUG. Bad flag. An invalid flag code setting was specified.

Action: Try the Register (R F) command again with the correct code.

BP

Explanation: DEBUG. Breakpoints. More than ten breakpoints were specified for the GO command.

Action: Re-enter the GO (G) command again with ten or fewer breakpoints.

BR

Explanation: DEBUG. Bad register. An invalid register name was specified.

Action: Re-enter the Register (R) command using a correct register name.

BREAK is on/off

Explanation: BREAK. This message indicates the status of BREAK, either on or off.

Action: Enter the command you want. For example if **Break is off** is displayed and **Break is on** is desired, enter the command:

```
BREAK ON
```

Buffer size adjusted

Explanation: VDISK. VDISK found it necessary to adjust the buffer size value in the Device=VDISK.SYS in the CONFIG.SYS command.

Action: No action required.

Buffer size:
Sector size:
Directory entries:
Transfer size:

Explanation: VDISK. VDISK has successfully installed a virtual disk indicating the virtual disk size, the sector size, the number of directory entries in use, and the maximum transfer size for the virtual disk.

Action: No action required.

C

Cannot CHDIR to root

Explanation: CHKDSK. CHKDSK, during its scan of the disk, attempted to CHDIR to the root but failed. This failed attempt is due to a damaged disk.

Action: No action required.

Cannot CHKDSK a network drive

Explanation: You cannot CHKDSK a drive that is either a network disk or a disk that is on your computer, but is currently being shared on the network.

Action: If the disk is being shared, you can PAUSE the server, do CHKDSK, then CONTINUE the server.

Cannot CHKDSK a SUBSTed or ASSIGNed drive

Explanation: SUBST. SUBST hides disk/device information necessary to CHKDSK.

Action: Remove the substitution and try again.

Cannot DISKCOMP to or from a Network drive

Explanation: DISKCOMP. You cannot use DISKCOMP to compare files on a network drive or on a drive that is on your computer but is currently being shared on the network.

Action: Use COMP *.* instead of DISKCOMP. If the disk is being shared, you can PAUSE the server, do DISKCOMP, then CONTINUE the server.

Cannot DISKCOPY to or from a Network drive

Explanation: DISKCOPY. You cannot use DISKCOPY to copy files to or from a network drive or a drive that is on your computer but is currently being shared on the network.

Action: Use COPY *.* instead of DISKCOPY. You can also use the COPY command to copy individual files instead of the whole diskette. Be sure your diskette is formatted. If the disk is being shared, you can PAUSE the server, do DISKCOPY, then CONTINUE the server.

Cannot do binary reads from a device

Explanation: COPY. You used the **/B** parameter with a device name while trying to copy from the device. The copy cannot be performed in binary mode because COPY must be able to detect end-of-file from the device.

Action: Re-enter COPY and omit the **/B** parameter or use the **/A** parameter after the device name.

Cannot edit .BAK file--rename file

Explanation: EDLIN. Files with the extension .BAK are considered to be backup files, with more up-to-date versions of the files assumed to exist. Therefore, .BAK files shouldn't ordinarily be edited.

Action: If it is necessary to edit the .BAK file, rename the file giving it an extension other than **.BAK**. Or else, copy the file and give the copy a different filename extension.

Cannot FDISK with network loaded

Explanation: FDISK. You cannot format the fixed disk when the network is loaded.

Action: No action required.

Cannot find file *object file*
Change diskette and press <ENTER>

Explanation: LINK. The linker could not locate the specified object module on the drive.

Action: Insert the correct diskette with the *object file* on it and press Enter.

> **Warning:** If it is necessary to change the diskette that contains the open VM.TMP file, you need to exit LINK by pressing Ctrl-Break instead of changing diskettes, and then restart LINK with the changing drive specified to locate the object file. Otherwise, there may be a loss of data on the diskette inserted. This usually occurs when using default drive value for the object file.

Cannot find library *library file*
Enter new file spec:

Explanation: LINK. The specified library could not be found on the drive.

Action: Enter the correct letter for the drive the library is on.

Warning: If it is necessary to change the diskette that contains the open VM.TMP file, you need to exit LINK by pressing Ctrl-Break instead of changing diskettes, and then restart LINK with the changing drive specified to locate the object file. Otherwise, there may be a loss of data on the diskette inserted. This usually occurs when using default drive value for the object file.

Cannot find system files

Explanation: FORMAT, SYS. The hidden files IBMBIO.COM and/or IBMDOS.COM were not found, and the current drive is not removable.

Action: Change the current drive to one that has the system files in the root and try again.

Cannot FORMAT a Network drive

Explanation: FORMAT. You cannot use the FORMAT command to format a network drive or a drive on your computer being shared on the network.

Action: If the drive is being shared, you can PAUSE the server, do FORMAT, then CONTINUE the server.

Cannot format a SUBSTed or ASSIGNed drive

Explanation: FORMAT. The ASSIGN *x=y* was executed prior to the attempted execution of FORMAT.

Action: Execute ASSIGN to restore the original drive letter assignment. Then proceed to execute FORMAT.

Cannot JOIN to a network drive

Explanation: You cannot use the JOIN command to join a local drive to a network drive or to join a network drive to a local drive.

Action: No action required.

Cannot LABEL a Network drive

Explanation: You tried to create a new, or change an existing volume label on a redirected block device.

Action: The suggested action is to retain the existing label on that drive.

Cannot load COMMAND, system halted

Explanation: DOS. DOS attempted to reload the command processor, but the area in which DOS keeps track of available memory was destroyed; or the command processor was not found in the path specified by the COMSPEC parameter.

Action: Restart DOS. Don't forget to put the DOS diskette in the drive.

Cannot nest response file

Explanation: LINK. You used @*filespec* within an automatic response file. Automatic response files cannot be nested.

Action:

1. Change the initial auto response file to eliminate nested auto response file.

2. Or, fix the syntax error if you did not intend this to be an auto response file.

Cannot open list file

Explanation: LINK. The directory or disk is full.

Action: Insert another disk, or delete some files from the disk that is full.

Cannot open overlay

Explanation: LINK. The directory or disk is full.

Action: Insert another disk, or delete some files from the disk that is full.

Cannot open response file: *filename*

Explanation: LINK. The automatic response file could not be found.

Action: Include drive specifier and/or path for the response file. Place the file on the proper disk.

Cannot open run file

Explanation: LINK. The directory or disk is full.

Action: Insert another diskette, or delete some files from the disk that is full.

Cannot open temporary file

Explanation: LINK. The directory or disk is full.

Action: Insert another disk, or delete some files from the disk that is full.

Cannot RECOVER Entry, Processing Continued

Explanation: CHKDSK.

Action: No action required.

Cannot RECOVER to a network drive

Explanation: RECOVER. You cannot use the RECOVER command to recover files from a network drive or from a drive on your computer that is being shared on the network.

Action: If the drive is being shared, you can PAUSE the server, do RECOVER, then CONTINUE the server.

Cannot reopen list file

Explanation: LINK. The original diskette was not actually replaced.

Action: Restart the linker.

Cannot start COMMAND, exiting

Explanation: DOS. While DOS was attempting to load another copy of the command processor, either the FILES= parameter in the configuration file was found to contain too small a value, or there is not enough available memory to contain the new copy of COMMAND.COM.

Action:

- Restart DOS.
- If necessary, increase the parameter value of FILES= parameter in CONFIG.SYS.

Cannot use PRINT - use NET PRINT

Explanation: PRINT. You cannot use the PRINT command on a network server computer. Use NET PRINT.

Action: Use NET PRINT to print the files.

CHDIR . . Failed Trying Alternate Method

Explanation: CHKDSK.

Action: Restart DOS and CHKDSK again.

Cannot SUBST to a network drive

Explanation: SUBST. You cannot use the SUBST command to substitute a drive for a network path or substitute a network drive for a local path.

Action: No action required.

Cannot SYS to a network drive

Explanation: SYS. You cannot use SYS to transfer system files to a network drive or to a drive on your computer that is currently being shared on the network.

Action: If the drive is being shared, you can PAUSE the server, do SYS, then CONTINUE the server.

Common area longer than 65536 bytes

Explanation: LINK. User's program has more than 64K of communal variables. **Note**: At the present time, only Microsoft C programs can possibly cause this message to appear.

Action: Rewrite your program using fewer or smaller communal variables.

COMn: bbbb,p,d,s,t initialized

Explanation: MODE. Informational message. The Asynchronous Communications Adapter is initialized. The values represent:

n adapter (COM1 or COM2)

bbbb baud rate

p parity

- **e** even
- **o** odd
- **n** none

d data bits

s stop bits (1 or 2)

t type of serial device

p serial printer (serial timeouts will be retried)

- other serial device (serial timeouts will not be retried)

Action: No action required. The feedback message from MODE shows its interpretation of the MODE command and the parameters you entered.

Compare error at offset *XXXXXXXX*

Explanation: COMP. Informational message. The files being compared contain different values at the displayed offset (in hexadecimal) into the file. The differing values are also displayed in hexadecimal.

Action: No action required This message is for your information to give you the location that contains mismatching information in two files.

Compare process ended

Explanation: DISKCOMP. This is an informative message indicating that DISKCOMP is finished comparing.

Action: No action required.

COM: ,e,7,1
Compare error(s) on
Track *xx*, side *xx*

Explanation: DISKCOMP. One or more locations on the indicated track and side contain different information when the diskettes are compared.

Action: This message is to inform you that there is a difference between diskettes. If you want an exact copy of a diskette, use DISKCOPY.

Compare more diskettes (Y/N)?

Explanation: DISKCOMP.

- This message indicates completion of DISKCOMP.
- You may compare more than one set of diskettes without re-entering the DISKCOMP command.

Action: If you wish to compare another pair of diskettes, enter **Y**, and DISKCOMP will ask you to insert the required diskettes. If you do not want to compare any more diskettes, enter **N**.

Compare more files (Y/N)?

Explanation: COMP. COMP has finished comparing files. You may compare more files without re-entering the COMP command.

Action:

- If you wish to compare the contents of two more files, enter **Y**, and COMP will ask you for the names of the files to compare.
- If you do not wish to compare more files, enter **N**.

Comparing *x* sectors per track, *n* side(s)

Explanation: DISKCOMP. The *x* indicates the number of sectors per track found on the first diskette (8 or 9). If you use **/8**, then the number 8 appears. The *n* will be either 1 or 2, indicating the number of sides that DISKCOMP will compare on the two diskettes. This number is determined by the number of sides DISKCOMP was able to successfully read from the first track of the first diskette.

Action: If *x* or *n* is not what you expected, let DISKCOMP finish comparing diskettes. Re-enter the DISKCOMP command with or without additional parameters.

Configuration too large for memory

Explanation: DOS. Occurs when the number specified for the FILES or BUFFERS commands in the CONFIG.SYS file does not leave enough room for DOS to be loaded.

Action: Restart DOS with a different diskette, and reduce the value specified for the FILES and/or BUFFERS commands.

Contains *xxx* non-contiguous blocks

Explanation: CHKDSK. Informational message. The file name preceding this message means that the file is not written sequentially on the disk—it is written in *xxx* pieces on different areas of the disk.

This message is for your information and does not indicate a problem with the disk.

Action: Since fragmented files take longer to read, you should consider copying badly fragmented files to another disk with the COPY command. This will record the file sequentially, resulting in better system performance when the file is read.

Convert directory to file (Y/N)?

Explanation: CHKDSK. The directory name preceding this message means the directory contains too much invalid information to be usable as a directory.

Action:

- If you reply **Y**, CHKDSK will convert the directory to a file so that you may examine it with DEBUG.
- If you reply **N**, the entry is not changed.

***xxx* lost clusters found in *yyy* chains.**
Convert lost chains to files (Y/N)?

Explanation: CHKDSK. Ctrl-Break was entered during a disk I/O operation. CHKDSK did not clean up the disk after encountering the Ctrl-Break.

Action:

- If you reply **Y** and you have used the /F parameter, CHKDSK will recover each chain into a separate file.
- If you reply **N**, CHKDSK frees the blocks up so they can be allocated to new files.
- If CHKDSK was specified (no /F), then messages displayed afterwards are informational (no corrective action was taken).

Copy another (Y/N)?

Explanation: DISKCOPY. This message allows you to make exact images of additional diskettes without re-entering the DISKCOPY command.

Action:

- If you wish to copy another entire diskette, enter **Y**. DISKCOPY asks you to insert the required diskettes.
- If you do not wish to make any more copies, enter **N**.

Copy complete

Explanation: DISKCOPY. Informational message.

Action: No action required. The source diskette contents have been successfully copied to the target diskette.

Copying *xxx* **tracks**
x **sectors per track,** *n* **sides**

Explanation: DISKCOPY. The *xxx* will be 40 or 80 tracks. The *x* will be 8, if the diskette was formatted using DOS 1.1; or 9, if the diskette was formatted with DOS 2.00 or 2.10; or 15 if the diskette was formatted using DOS 3.00. The *n* is either 1 or 2 indicating the number of sides. If the diskette has been formatted double sided and subsequently formatted single sided, DISKCOPY will say it is copying *2 side(s)* In this situation, use DISKCOPY /1.

If the diskette was formatted with FORMAT /8, DISKCOPY will say **Copying 9 sectors**. This is due to the fact that FORMAT will format the disk with 9 sectors, but initializes 8. DISKCOPY mimics FORMAT in that it says, **Copying 9 sectors**, but the target diskette ends up with 8 sectors per track.

Action: No action required.

Current drive is no longer valid

Explanation: COMMAND. While attempting to get the current drive for the DOS prompt ($p), COMMAND found that the drive is no longer valid. This can occur if your current drive is a network drive and you delete the network drive.

Action: Change your current drive to a valid drive.

D

Data record too large

Explanation: LINK. LEDATA record contains more than 1024 bytes of data.

Action: This is a translator error. Note the translator (compiler or assembler) that produced the incorrect object module and the circumstances under which it was produced, and report the information to your dealer.

Delete current volume label (Y/N)?

Explanation: LABEL. You can avoid deleting the existing label by pressing Enter rather than typing a new volume label.

Action Type **n** to keep the current label.

DF

Explanation: DEBUG. Double flag. Conflicting codes were specified for a single flag.

Action: DEBUG is informing you that a flag can be changed only once per Register (R F) command.

Dir path listing for volume *xxxxxxx*

Explanation: TREE. Informational message telling you the volume label of the disk.

Action: No action required.

Directory entries adjusted

Explanation: VDISK. VDISK found it necessary to adjust the number of directory entries in the DEVICE=VDISK.SYS in the CONFIG.SYS command.

Action: No action required.

Directory is joined,
tree past this point not processed.

Explanation: CHKDSK. This is an informative message. Joining a drive to another tree will extend the tree. The allocation on the joined device is completely separate, and there is no reason to CHKDSK across devices. CHKDSK is skipping over the joined device.

Action: No action required.

Directory is totally empty, no . or ..,
tree past this point not processed.

Explanation: A subdirectory was found that did not properly contain a . or .. entry. This usually happens when DOS is not given a chance to update the disk properly. During the updating process, the system may have shut down, or you may have reloaded the system before DOS finished the update.

Action: Try using the RECOVER command to try to recover files on the damaged disk.

Disk boot failure

Explanation: DOS. An error occurred when you tried to load DOS into memory.

Action: Restart the system. If subsequent attempts to start the system also fail, place a backup DOS diskette in drive A and restart your system.

Disk error reading drive *x*

Explanation: You tried to read absolute sectors on a network drive. This cannot be done.

Action: If possible, us the Loader Write filespec option of this DEBUG command.

Disk error writing drive *x*

Explanation: You tried to write to absolute sectors on a network disk. This cannot be done.

Action: If possible, use the Loader Write filespec option of this DEBUG command.

Disk error reading FAT *X*

Explanation: CHKDSK. The File Allocation Table indicated is invalid. This can be caused by a power failure while a file is open.

Action: If this message appears twice for FATs 1 and 2, format the disk to make it usable again. If FORMAT fails, the disk is probably unusable.

Disk error writing FAT *X*

Explanation: CHKDSK. A disk error was encountered while CHKDSK was attempting to update the file allocation table (FAT) on the specified drive. *X* will be 1 or 2, depending on which of the 2 copies of the file allocation table could not be written.

Action: If this message appears twice, for FAT's 1 and 2, format the disk to make it usable again. If FORMAT fails, discard the disk since it is probably unusable.

Disk full. Edits lost

Explanation: EDLIN. An End Edit command ended abnormally because the disk is full (not enough free space to save the entire file). Any editing done to the file is lost.

Action: Obtain a fresh diskette, copy the file on to the fresh diskette, and start editing again.

Disk not compatible

Explanation: FORMAT. You cannot use the DOS FORMAT command to format a diskette using the drive you specified. This message informs you that the drive you specified is not supported by the IBM device interfaces that FORMAT requires.

Action: Obtain a compatible disk drive.

Disk unsuitable for system disk

Explanation: FORMAT. A defective track was detected where the DOS files were to reside. The diskette can be used only for data.

Action: The diskette can be used only for data. Use another disk if you wish to copy DOS files.

Diskette/Drive not compatible

Explanation:. DISKCOPY, DISKCOMP. The destination diskette/drive is different from the source diskette/drive in a way that prohibits the copy or compare. For example, you cannot DISKCOPY a double-sided diskette to a single-sided diskette or DISKCOMP a 96 tpi diskette in a 48 tpi drive with a 96 tpi diskette in a 96 tpi drive.

Action: Match the diskette/drive types and try again.

Diskettes compare OK

Explanation: DISKCOMP. Informational message. The two diskettes just compared contain identical information.

Action: No action required.

Divide overflow

Explanation: DOS. A program tried to divide a number by zero, or a logic error caused an internal malfunction. The program ends and you return to DOS.

Action: Correct the programming error and continue. If this is a purchased program, take it back to your dealer.

Do you see the leftmost 9? (Y/N)

Explanation: MODE. ,R,T was specified.

Action: Respond **Y** or **N**. This prompt is repeated until you respond **Y**.

Do you see the rightmost 9? (Y/N)

Explanation: MODE. ,L,T was specified.

Action: Respond **Y** or **N**. This prompt is repeated until you respond **Y**.

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

Explanation: FDISK. When the "Create DOS Partition" option is used on the current fixed disk and the fixed disk has never been set up, this question is asked.

Action:

- If you enter **Y**, the entire current fixed disk is used for DOS and it will be made active.
- If you enter **N**, you are asked to enter the limits of the DOS partition you want to create.

Command format:DISKCOPY d: d: [/1]

Explanation: DISKCOPY. An invalid parameter or filename was typed.

Action: Check the command format, and re-enter the command.

DOS partition already exists

Explanation: FDISK. You chose the "Create DOS Partition" option for a fixed disk that already has a DOS partition.

Action: Return to the Main Menu and choose option 4, **Display Partition Data**. Read Chapter 3 before continuing.

DOS partition created

Explanation: FDISK. Informational message. A DOS partition has been created on the fixed disk.

Action: You will need to run the FORMAT command on the DOS partition before you can store files on the fixed disk.

DOS partition deleted

Explanation: FDISK. Informational message. A DOS partition no longer exists on the fixed disk.

Action: No action required.

Drive types or diskette types not compatible

Explanation: DISKCOMP or DISKCOPY. The source and target diskettes or drives are not compatible.

Action: Refer to the DISKCOMP or DISKCOPY commands for the allowable combinations.

Dup record too complex

Explanation: LINK. A DUP record in an assembler source program is too deeply nested or too complicated for the linker to expand. A single DUP requires 1024 bytes before expansion.

Action: Reduce the number of structures or DUP statements in the assembler source program, create a new object module, and retry LINK.

Duplicate filename or file not found

Explanation: RENAME. You tried to rename a file to a filename that already exists on the diskette, or the file to be renamed could not be found on the specified (or default) drive. RENAME is warning you that you are using the same name for two files, or else it cannot find the file you are trying to rename.

Action: Did you type the filename correctly? Take a second look at the filename you want to change, and re-enter the RENAME command.

E

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

Explanation: FDISK. The "Change Active Partition" option is requesting you to enter the number of the partition you want to make active.

Action: Type the number of the partition that you want to make active on the current fixed disk. Then press the Enter key.

Note: The partitions are displayed above the prompt.

Enter partition size.....: [dddd]

Explanation: FDISK. The "Create DOS Partition" option is requesting that you enter the size of the partition you wish to create.

Action: The number shown in the brackets is the default size. If you only press Enter, that size will be used as the partition size. Otherwise, enter the desired size, then press Enter.

Enter primary file name

Explanation: COMP. DOS asks you for primary filename.

Action: Enter the filespec of the first of two files to be compared.

Enter 2nd file name or drive id

Explanation: COMP. DOS asks you for filespec of second of the two files you want compared.

Action: Enter the filespec of the second of two files to be compared, or just enter the drive letter if the filename is the same as the primary filename.

Enter starting cylinder number..: [dddd]

Explanation: FDISK. The "Create DOS Partition" option is requesting that you enter the starting cylinder number for the DOS partition you are creating. The value in the brackets is the default value. It is the starting cylinder of the largest piece of free space on the current fixed disk.

Action: Type the starting cylinder number and press Enter, or just press Enter to use the default value.

Entry Error

Explanation: EDLIN. EDLIN has detected a syntax error.

Action: Correct the syntax error on the last command.

Entry has a bad attribute
(or size or link)

Explanation: CHKDSK. This message may begin with one or two periods, indicating which entry in the subdirectory was in error. One period indicates the current directory is

in error. Two periods mean the parent directory is in error. If you did *not* enter the /F parameter, no corrective action is taken.

Action: Enter: CHKDSK /F. CHKDSK will then try to correct the error.

EOF mark not found

Explanation: COMP. COMP could not find the end of valid data in the last block of the files being compared. This message usually occurs when comparing non-text files; it should not occur when comparing text files.

Action: For more details, see the COMP command in Chapter 2.

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

Explanation: CHKDSK. Informational message. An error was found and you have not used the /F parameter. CHKDSK performs its analysis as though it were going to correct any errors detected, so that you can see the results of its analysis, but it will not actually write the corrections on the disk.

Action: No action required.

Error in EXE file

Explanation: DOS. An error was detected in the relocation information placed in the file by the LINK program. This may be due to a modification to the file.

Action:

- If you are using a purchased program, rerun the program using your backup copy.
- If you still have trouble see your authorized dealer.
- If you are using a program you wrote yourself, go through the LINK procedure again.

Error in EXE or HEX file

Explanation: DEBUG. The file contained invalid records or characters.

Action: Get another copy of the program, and run DEBUG again.

Error loading operating system

Explanation: DOS. A disk error occurred while attempting to load your operating system from fixed disk.

Action: Restart the system. If the error persists after several tries, restart the system, (you should start DOS from your DOS diskette) and use the SYS command to transfer a new copy of DOS to your fixed disk.

Error reading fixed disk

Explanation: FDISK. The FDISK program was unable to read the startup record of the current fixed disk after five tries.

Action: Try the FDISK program again. If after several tries you still get the same error, consult the IBM *Guide to Operations*, "Problem Determination" section. If you still cannot solve the problem, see your IBM Personal Computer Dealer.

Error writing fixed disk

Explanation: FDISK. The FDISK program was unable to write the startup record of the current fixed disk after five tries.

Action: Try the FDISK program again. If after several tries

you still get the same error, consult the IBM *Guide to Operations*, "Problem Determination" section. If you still cannot solve the problem see your IBM Personal Computer, Dealer.

Error writing to device

Explanation: Commands. Informational message. DOS encountered an I/O error when writing output to a device. The device is unable to handle the number of bytes requested.

Action: Change amount of data in the file and retry the command.

Errors on list device indicate that it may be off-line. Please check.

Explanation: PRINT. The device being used for background printing is off line. This message only appears when the device is off line and you enter a new PRINT command.

Action: Make sure the printing device is connected and switched on.

EXE and HEX files cannot be written

Explanation: DEBUG. This error normally occurs when you load a .HEX or .EXE file, modify it, and then attempt to write the file back to a diskette.

You should understand that .EXE and .HEX files contain loading information that is used to load the file. When DEBUG is executed, it loads the .EXE file while at the same time discarding the information. During direct execution of an .EXE file, the same thing happens. When you use DEBUG to write an .EXE file, the information is gone; therefore, a correct .EXE file cannot be generated. That is why you get this error message.

The error might also be caused by the fact that the data consists of a .COM file loaded in with DEBUG and you are now trying to write it to an .EXE or HEX file. This is not possible. The data requires a backward conversion that DEBUG doesn't support.

Action: Rename the file using a different extension, then execute DEBUG. DEBUG then *reads* the file in instead of loading it, and the file can now be written out. Reading the file in does not alter the records or data in the file.

Extender card switches do not match the system memory size

Explanation: VDISK. The Extender card switch settings do not reflect the total amount of memory in the system unit. Although the extender card switch settings may be set correctly for your system, VDISK does not support memory in an expansion unit.

Action: Check the settings on the extender card.

EXEC failure

Explanation: Commands. DOS encountered an error while reading a command from disk, or the FILES= command in the configuration file (CONFIG.SYS) does not specify a large enough value.

Action: Increase the FILES= value. Restart DOS. If restarting DOS does not work, then there may be a problem with the disk itself.

File allocation table bad, drive *x*
Abort, Retry, Ignore?

Explanation: DOS.

Action: See the message Disk error reading drive *x* under "Device Error Messages" at the beginning of this appendix. If this error persists, the disk is unusable and should be formatted again.

File AND File

Explanation: COMP. Informational message. This message indicates the full path and filenames of the two files being compared.

Action: No action required.

File *xxx* canceled by operator

Explanation: PRINT. Informational message. This message appears on the printer after you cancel the printing of a file to serve as a reminder that the printout is incomplete.

Action: No action required.

File Cannot be Converted

Explanation: EXE2BIN.

Action: No action required.

File cannot be copied onto itself

Explanation: COPY. You tried to COPY a file and place the copy (with the same name as the original) in the same directory and on the same disk as the original file.

Action: Change the name given to the copy, or put it in a different directory, or put it on another disk.

File creation error

Explanation: DOS and commands. An unsuccessful attempt was made to add a new filename to the directory or to replace a file that was already there.

Action: If the file was already there, check whether the file is marked read only and cannot be replaced. Otherwise, run CHKDSK to determine if the directory is full, or if some other condition caused the error.

File is cross-linked:
on cluster *xx*

Explanation: CHKDSK. This message appears twice for each cross-linked cluster number, naming the two files in error. The same data block is allocated to both files.

Action: No corrective action is taken automatically. You must correct the problem by doing the following:

1. Make copies of both files (use COPY command).
2. Delete the original files (use ERASE command).
3. Review the files for validity and edit as necessary.

filename **is currently being printed**
filename **is in queue**

Explanation: PRINT. Informational message. These messages appear together when you issue a PRINT command with no parameters. They occur individually when you queue the first or a subsequent file for printing.

Action: No action required.

File is READ-ONLY

Explanation: EDLIN. The file you specified is read-only.

Action: Use the ATTRIB command to change the attribute of the file to read-write.

File name must be specified

Explanation: EDLIN. You typed EDLIN without specifying the file that you wanted to edit.

Action: Type the filename on the command line when you use EDLIN.

File not found

Explanation: DOS and commands. A file named in a command or command parameter does not exist in the directory of the specified (or default) drive.

Action: Retry the command using the correct filename.

File not in print queue

Explanation: PRINT. The file you want to cancel is not in the print queue.

Action: No action required.

File sharing conflict

Explanation: COMP. Unable to compare files because one file is in use by another process.

Action: Try the compare again at a later time.

Files are different sizes

Explanation: COMP. Informational message. The sizes of the files to be compared do not match. This means that a comparison cannot be done because one of the files contains data which does not match the data in the other file.

Action: No action required.

Files compare OK

Explanation: COMP. Informational message. The two files just compared contain identical information.

Action: No action required.

Files were backed up xx/xx/xx

Explanation: RESTORE. Informational message telling you the date the files were backed up on.

Action: No action required.

First cluster number is invalid, entry truncated

Explanation: CHKDSK. Informational message. The file whose name precedes this message contains an invalid pointer to the data area. If you specify **/F** parameter the file will be truncated to a zero length file.

Action: No action required.

Fixed Backup device is full

Explanation: BACKUP. The fixed disk media target is full. Therefore, no more files can be backed up onto that device.

Action: No action required.

Fixup offset exceeds field width near *xxxx* **in** *xxxxxx* offset *xxxx*

Explanation: LINK. Some possible causes: 1) A group is larger than 64K bytes. 2) The linker has been asked to fix up an intersegment short jump or intersegment short call. 3) A data item's name conflicts with that of a subroutine in a library included in the link. 4) In an assembly language source file, the user has an EXTRN declaration inside the body of a segment. 5) A data item (DB, DW, etc.) was declared outside all segments.

Action: Revise the source and recreate the object file.

Fixups needed - base segment (hex):

Explanation: EXE2BIN. The source (.EXE) file contains information indicating that a load segment is required for the file.

Action: Specify the absolute segment address at which the finished module is to be loaded.

> **Note:** We do not recommend using such a program as a .COM file because the program is dependent upon being loaded at a specific memory location.

FOR cannot be nested

Explanation: BATCH. More than one FOR subcommand was found on one command line in the batch file.

Action: Use only one FOR subcommand per command line. Then retry command.

Format failure

Explanation: FORMAT. A disk error was encountered while creating the target disk.

Action: The disk is unusable. Use another disk and retry the command.

Formatting while copying

Explanation: DISKCOPY. Informational message. The target diskette was found to contain unformatted tracks. DISKCOPY will format the remainder of the target diskette as it copies data.

Action: No action required.

Note: If this message is followed by the message **Incompatible drive types**, you tried to copy a dual-sided diskette to a drive that does not have dual-sided capability. This cannot be done. Processing ends and the target diskette contains no useful data.

H

Has invalid cluster, file truncated

Explanation: CHKDSK. The file name preceding this message means that the file contains an invalid pointer to the data area.

Action: Use the /F parameter to truncate the file at the last valid data block. No corrective action occurs if CHKDSK is executed without the **/F** parameter.

I

Illegal Device Name

Explanation: MODE. The specified printer must be:

- LPT1:
- LPT2:

  or,

- LPT3:

The specified Asynchronous Communications Adapter must exist and be:

- COM1:

  or,

- COM2:

There must be no more than one blank between MODE and its parameters.

Action: Use the correct device name and retry the command.

Incompatible system size

Explanation: SYS. The target diskette contained a copy of DOS that is smaller than the one being copied. The system transfer does not take place.

Action: Format a blank diskette (use the FORMAT /S command) and then copy any files to the new diskette.

Incorrect DOS version

Explanation: Commands. The command you just entered requires a different version of DOS than the one you are using.

Action: Obtain correct version of DOS and retry command.

Incorrect DOS version, use DOS 2.00 or later

Explanation: LINK. Linker will not run on versions of DOS prior to DOS 2.00.

Action: Reboot your system with DOS 2.00 or a later version, and try linking again.

Incorrect parameter

Explanation: SHARE. A parameter specified in the SHARE command is invalid.

Action: Check the command syntax and retry the command.

Infinite retry on parallel printer time-out

Explanation: MODE. "P" was specified in Option 1, requesting continuous retry on time-out errors.

Action: No action required.

Insert backup diskette *xx* in drive *x*:
Strike any key when ready

Explanation: RESTORE.

Action: Insert the backup diskette(s) in sequence in accordance with the prompt. Press any key and RESTORE will continue.

Insert backup source diskette in drive *x*
Strike any key when ready

Explanation: BACKUP. Informational prompt telling you to insert the source backup.

Action: Insert the source in drive *x* and strike any key.

Insert backup target diskette *y* in drive *x*
Strike any key when ready

Explanation: BACKUP. Informational prompt telling you to insert the target backup diskette.

Action: Insert the target in drive *x* and strike any key.

Insert disk with \COMMAND.COM in drive A
and strike any key when ready

Explanation: DOS. DOS is attempting to reload the command processor, but COMMAND.COM is not on the drive that DOS was started from.

Action: Insert the DOS diskette in the indicated drive and press any key.

Insert diskette for drive *x* and press any key when ready

Explanation: DOS. In a single diskette system, Drive A: or B:, which is not the current default drive, is being referenced, so DOS is asking for the diskette corresponding to that drive.

Action: If the diskette for *x* is different than the one currently in the drive, insert the appropriate diskette and press any key.

Insert disk with batch file and strike any key when ready

Explanation: DOS. The diskette that contained the batch file being processed was removed. The batch processor is trying to find the next command in the file.

Action: Insert the diskette in the appropriate drive and press any key. Processing will continue.

Insert DOS disk in *x* and strike any key when ready

Explanation: FORMAT. FORMAT is trying to load the DOS files, but the indicated drive *x* does not contain the DOS diskette.

Action: Follow the prompt and insert the DOS diskette. Press any key and processing continues.

Insert DOS diskette in drive A:
Press any key when ready . . .

Explanation: FDISK. You have successfully created the DOS partition on the current fixed disk.

Action: Insert the DOS diskette into drive A and press any key. This restarts your IBM Personal Computer.

The current fixed disk is now assigned a fixed disk letter and you can now FORMAT the fixed disk.

Insert first diskette in drive *x*
Insert second diskette in drive *x*

Explanation: DISKCOMP.

Action: Insert the first (or second) of the two diskettes to be compared into the indicated drive. One or both of these messages is followed by the message **Strike any key when ready**. Press a key and the comparison starts.

Insert last backup target in drive *x*

Explanation: BACKUP. Informational prompt. The /A parameter was specified.

Action: Insert the last backup target used in the previous backup.

Insert new diskette for drive *x*
and press ENTER when ready

Explanation: FORMAT. Prompt asking you to insert the diskette you want to format.

Action: Insert the diskette you want to format and press Enter.

Insert source diskette in drive *x*
Insert target diskette in drive *x*

Explanation: DISKCOPY.

Action: Insert the appropriate diskette into the indicated drive, and press any key when asked. The copying process starts.

Insert restore target *xx* in drive *yy*
Strike any key when ready

Explanation: RESTORE. This is the informational prompt for removable media.

Action: Insert the target in drive *yy* and strike any key.

Insert System disk in *x*
and strike any key when ready

Explanation: SYS. SYS is trying to load the DOS files, but the indicated drive *x* does not contain the DOS diskette.

Action: Follow the prompt and insert the DOS diskette. Press any key and processing continues.

Insufficient disk space

Explanation: DOS and commands. The disk does not contain enough free space to contain the file being written.

Action: If you suspect this condition is invalid, run CHKDSK to determine the status of the disk. Otherwise, use another disk and retry the command.

Insufficient memory

Explanation: Commands. The amount of available memory is too small to allow these commands to function.

Action: Change the BUFFERS= parameter in the CONFIG.SYS file to a smaller value (if you have specified BUFFERS=), restart the system and try the command again. If the message still appears, your system does not have enough memory to execute the command.

Insufficient memory for system transfer

Explanation: FORMAT and SYS. There is not enough memory on the disk.

Action: No action required.

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

Explanation: CHKDSK. You instructed CHKDSK to create files from the "lost" data blocks it has found, but the root directory is full, and all of the lost chains could not be recovered into files.

Action:

1. Copy some of the recovered files to another disk for further examination.

2. Delete the recovered files from the disk you are checking.

3. Run CHKDSK again to recover the remainder of the lost data.

Insufficient space on disk

Explanation: DEBUG. A write command was issued to a disk that doesn't have enough free space to hold the data being written.

Action: If you are writing to a diskette, you can insert a diskette that has enough free space, then reissue the write command. Otherwise, you should erase files from the disk and run DEBUG again.

Insufficient stack space

Explanation: LINK. There is not enough memory to run the linker.

Action: Restart the system to free some memory that was used by installable device drivers such as VDISK, and terminate and stay resident programs such as PRINT and GRAPHICS.

Intermediate file error during pipe

Explanation: DOS. DOS is unable to create one or both of its intermediate files because the default drive's root directory was full, or DOS is unable to locate the piping files, or the disk does not have enough space to hold the data being piped.

Action: Erase some files from the default drive's root directory, and reissue the command that failed. If you get the same message, one of the programs in the command line has erased one or both of the piping files. Correct the program and reissue the command line.

Invalid baud rate specified

Explanation: MODE.

Action: Specify the baud rate as 110, 150, 300, 600, 1200, 2400, 4800, or 9600 (you need specify only the first two characters of the number).

Invalid characters in volume label

Explanation: FORMAT. One or more of the characters you entered in the volume label is not a valid filename character, or the name contained a period (volume labels contain 1 to 11 valid characters without a period).

Action: Retry using valid characters.

Invalid COMMAND.COM in drive n

Explanation: DOS. When DOS tried to reload the command processor, the copy of COMMAND.COM on the disk was found to be an incorrect version.

Action: Insert the correct DOS diskette and press any key to continue.

Invalid country code

Explanation: SELECT, COUNTRY. The country code specified in COUNTRY= command in CONFIG.SYS or input to the SELECT command is not available.

Action: See SELECT command in Chapter 7 or "Creating a CONFIG.SYS File" in Chapter 4 for the correct country code and try again.

Invalid current directory

Explanation: CHKDSK. CHKDSK attempted to read the current directory. It found an unrecoverable error on the disk.

Action: No action required.

Invalid date

Explanation: DATE. You entered an invalid date or delimiter. The only valid delimiters in a date entry are hyphens (-) and slashes (/).

Action: Re-enter valid date.

Invalid device

Explanation: CTTY. The device name you specified is an invalid name to DOS.

Action: Retry command using valid device name.

Invalid directory

Explanation: DOS and commands. One of the directories in the specified path does not exist.

Action: Retry command using valid directory.

Invalid disk change

Explanation: The diskette in the high-capacity drive was changed while files were still open on the diskette.

Action: Reinsert the correct diskette.

Invalid drive in search path

Explanation: DOS. An invalid drive specifier was found in one of the paths specified in the PATH command.

This message appears when DOS attempts to locate a command or batch file, not at the time you issued the erroneous PATH command.

Action:

1. Enter PATH. This displays the *PATHs* you previously defined.

2. Find the invalid specifier.

3. Re-enter the PATH command with the valid drive specifier and the paths you desire.

Invalid drive or filename

Explanation: RECOVER. The drive or filename specified is invalid.

Action: No action required.

Invalid drive specification
Specified drive does not exist,
or is non-removable

Explanation: DOS and commands. An invalid drive specification was just entered in a command or in one of its parameters.

Action: Re-enter the command using a valid drive specifier.

Invalid drive specification
Source and Target drives are the same

Explanation: BACKUP and RESTORE. You cannot have the same drive specifiers for the source and target drives.

Action: Specify a different drive letter for the source and target drives.

Invalid filename or file not found

Explanation: RENAME or TYPE. You tried to rename a file that was either invalid or not found in the specified directory.

TYPE does not allow global filename characters.

Action: No action required.

Invalid format file

Explanation: LINK. A library is in error.

Action: Restore library file from your backup disk and try again.

Invalid media or track 0 bad - disk unusable

Explanation: FORMAT. FORMAT was unable to format track 0 on the specified media. This error occurs if:

- Track 0 is unusable. Track 0 is where the boot record, file allocation table, and directory must reside. If track 0 is bad, the disk is unusable.
- The diskette type and drive type are incompatible. You tried to format a double-sided diskette in a high-capacity drive, or a high-capacity diskette in a double-sided drive.

Action: For the first case, obtain another disk and retry the FORMAT command. For the second case, retry the FORMAT command specifying the /4 parameter.

Invalid number of parameters

Explanation: Commands. You have specified too few or too many parameters for the command you issued.

Action: Review this command in the "Commands" section of this book.

Invalid numeric parameter

Explanation: LINK. The numeric value is not in digits.

Action: Run LINK again and name the numeric parameter with values of 0 through 9 for each digit.

Invalid numeric switch specification
switch error: "s: *xxx*"

Explanation: LINK. Typographical error entering value for one of the LINKER switches, such as entering a character string for a switch that requires a numeric value.

Action: No action required. Linker will abort.

Invalid object module

Explanation: LINK. Object module(s) was incorrectly formed or unobserved errors occurred during compilation. The disk may be bad.

Action: Recompile the module to either the same or a different disk.

Invalid parameter

Explanation: DOS and commands. One or more of the parameters entered for these commands is not valid.

Action: If the program expects a drive specifier, enter a colon following the drive letter. In other cases, make sure the character following the slash (/) is valid for the program being run. For JOIN and SUBST see the specific command in Chapter 7 for valid parameters.

Invalid parameters

Explanation: MODE.

- No parameters were entered
- The first parameter character was other than **L**, or **C**
- The first parameter was other than **40**, **80**, **BW40**, **BW80**, **CO40**, **CO80**, **MONO**, **L**, **R**
- The display adapter the parameter refers to is not present in the machine.

Action: Check the preceding list and correct accordingly.

Invalid partition table

Explanation: Start-up. While attempting to start DOS from your fixed disk, the start-up procedures detected invalid information in the disk's partition information.

Action:

1. Start DOS from the diskette.
2. Use the FDISK command to examine and correct the fixed disk partition information.

Invalid path

Explanation: TREE, BACKUP, RESTORE. The command was unable to use a directory whose name was found in another directory.

Action: Run CHKDSK to find out what is wrong with the directory structure.

Invalid path, not directory or directory not empty

Explanation: RMDIR.

- The specified directory was not removed because one of the names you specified in the path was not a valid directory name.

  or,

- The directory you specified still contains entries for files or other subdirectories (with the exception of the . and .. entries).

- You cannot remove a current directory.

Action: Try one of the following:

- Correct the invalid directory name in the path.
- Delete any files or remove any subdirectories in the directory.
- Change to a different subdirectory and try again.

Invalid path or file name

Explanation: ATTRIB or COPY. You specified a directory or file name that does not exist.

Action: Use correct name. Check for the following. Then retry command:

- Correct spelling of names
- Valid directory names
- Existence of file in the subdirectory specified

Invalid subdirectory

Explanation: CHKDSK. Invalid information was detected in the subdirectory whose name precedes this message.

Action: CHKDSK attempts to correct the error if you have used the /F parameter. For more specific information about the nature of the error, run CHKDSK with the /V parameter.

Invalid switch

Explanation: LINK. The character indicated on the preceding line is not a valid linker parameter (switch).

Action: Review Chapter 9, "The Linker (Link) Program" to determine which characters are a valid link parameters (switches).

Invalid switch character

Explanation: VDISK. VDISK encountered a forward slash in the DEVICE=VDISK.SYS CONFIG.SYS command, but the following character was not an E. VDISK will attempt to install the virtual disk in low memory.

Action: No action required.

Invalid time

Explanation: TIME. An invalid time or delimiter was entered.

Action: Re-enter the correct time. The only valid delimiters are:

- A colon (:) between the hours and minutes
- A colon (:) between the minutes and seconds
- A period (.) between the seconds and hundredths of a second.

L

Label not found

Explanation: Batch. Informational message. A GOTO command named a label that does not exist in the batch file. This caused the system to read to the end of the batch file, ending batch processing.

Action: If you do not want the GOTO to exit the batch file, edit the batch file and put the label in the desired location.

Last backup target not inserted

Explanation: BACKUP. The /A parameter was specified, but the removable target was not the last in the backup sequence.

Action: No action required.

***** Last file not backed up *****

Explanation: BACKUP. The fixed target became full while the file was being backed up. There is no room for the entire file, so the backup file is deleted and this message is issued.

Action: No action required.

Line too long

Explanation: EDLIN. Upon replacing a string, the replacement caused the line to expand beyond the 253-character limit. The REPLACE text command is ended abnormally.

Action: Split the long line into shorter lines; then issue the REPLACE text command again.

List output is not assigned to a device

Explanation: PRINT. The list device specified is not a valid device to PRINT to.

Action: Invoke PRINT again and specify a valid list device. Trailing colon is not needed.

LPT#: not rerouted.

Explanation: MODE. Informational message. The parallel printer will now receive its own output, even if this printer's output had previously been rerouted to a serial device.

This message is provided for your information and indicates cancellation of any previous redirection that may have been in effect, because you have set the printer width or vertical spacing.

Action: No action required.

LPT#: rerouted to COM*n*:

Explanation: MODE. Informational message

This message is provided for your information and indicates that any request that would normally have gone to the parallel printer LPT# (#=1, 2, or 3) is sent instead to the serial device COM*n* (*n*=1 or 2).

Action: No action required.

LPT#: set for 80

Explanation: MODE. Informational message. You tried to set the printer line length to 80 characters by requesting standard type format.

This message is provided for your information. If the attempt was unsuccessful, an error message will follow this message on the screen.

Action: No action required

LPT#: set for 132

Explanation: MODE. Informational message. You tried to set the printer line length to 132 characters by requesting compressed type format.

If the attempt is unsuccessful, an error message will follow this message on the screen.

Action: No action required.

Make sure a diskette is inserted into the drive and the door is closed

Explanation: DISKCOMP and DISKCOPY. The drive is empty or the drive door is left open.

Action: Insert a diskette or close the drive door.

Maximum available space is *xxxx* **cylinders at cylinder** *xxxx*.

Explanation: FDISK. Informational message. The "Create DOS Partition" option displays the largest available piece of space on the current fixed disk. These numbers are also used as the defaults for the two prompts that will follow.

Action: No action required.

Memory allocation error
Cannot load COMMAND, system halted

Explanation: DOS. A program destroyed the area in which DOS keeps track of available memory.

Action: Restart DOS.

Mismatch DOS level number

Explanation: LINK. Internal linker error.

Action: Record the scenario that produced this message, and contact your IBM dealer.

Missing operating system

Explanation: Startup. When you tried to start DOS from fixed disk, the startup procedures determined that the DOS partition was marked "bootable" (startable), but that it doesn't contain a copy of DOS.

Action: Start DOS from diskette and use FORMAT with the /S parameter to place a copy of DOS on the fixed disk. You should back up your files before doing the FORMAT or they will be lost.

Must specify ON or OFF

Explanation: BREAK. You entered something other than on or off.

Action: Try again, specifying on or off.

Must specify destination line number

Explanation: EDLIN. A Move or Copy command was entered without a destination line number.

Action: Re-enter the command with a valid destination line number.

N

Name of list device [PRN]:

Explanation: PRINT. This message appears the first time you start print after DOS has been restarted.

Action: Reply with the reserved device name which is to receive the printed output, or simply press Enter if the first parallel printer [PRN] is to be used.

NEAR/FAR conflict

Explanation: LINK. Conflicting near and far definitions for a communal variable.

Action: Revise definitions to be consistent.

Network support loaded
Unable to FDISK

Explanation: FDISK. FDISK cannot modify a remote fixed disk.

Action: Restart system without Network support and try FDISK again.

No DOS partition to delete.

Explanation: FDISK. You chose the "Delete DOS Partition" option but there was no DOS partition on the current fixed disk.

Action:

1. Return to the Main Menu.
2. Select the **Display Partition Data** to review.
3. Proceed with your next choice.

No free file handles
Cannot start COMMAND, exiting

Explanation: DOS. An attempt to load a second copy of the command processor failed because there are too many files open.

Action: Increase the number in the FILES= command in the configuration file (CONFIG.SYS), and restart DOS.

No fixed disks present

Explanation: FDISK. The FDISK program was run on an IBM Personal Computer that:

- Does not have a fixed disk, or
- Has a fixed disk in the expansion unit and the expansion unit is not powered on, or
- Has a fixed disk that is not properly installed.

Action: From the above list find out what caused the problem and take appropriate action. Check to make sure the expansion unit is powered ON first.

No object modules specified

Explanation: LINK. You did not name any object modules in the command line or in response to the prompt.

Action: Name the object modules, since the linker needs some files to link.

No partitions to make active.

Explanation: FDISK. You chose the "Change Active Partition" option but there were no partitions on the current fixed disk to be made active.

Action: Use the "Create DOS Partition" option to create a partition, then the "Change Active Partition" option to make it the active partition.

No path

Explanation: PATH. Informational message. There is currently no alternate path for DOS to search to find commands and batch files if it does not find them in the specified (or current directory).

Action: Informational message unless you want to define a set of paths. If so, enter PATH and the set of paths you want. Then press Enter.

No retry on parallel printer time-out

Explanation: MODE. P was not specified in Option 1, requesting no retry on time-out errors.

Action: No action required.

No room for system on destination disk

Explanation: SYS. The destination diskette does not contain the required reserved space for DOS; therefore, the system cannot be transferred.

Action: Format a blank diskette (use the FORMAT /S command), then copy any other files to the new diskette.

No room in directory for file

Explanation: EDLIN. The directory on the specified disk is full. Your editing changes are lost.

Action: Make sure that your disk has available directory entries and run EDLIN again.

No room in root directory

Explanation: LABEL. An error occurred while creating the volume label, probably due lack of room in the root directory for another entry.

Action: If the length of the label is less than 11 characters, and you used only valid filename characters (no periods), then delete a file from the root directory to make room for another entry. Try LABEL again.

No space for a *xxxx* cylinder partition.

Explanation: FDISK. You entered a "Partition Cylinder Size" that is larger than the largest piece of free space on the disk.

Action: Enter a smaller number.

No space for a *xxxx* cylinder partition at cylinder *xxxx*

Explanation: FDISK. You requested a partition to be created at a place on the current fixed disk and there is no space at that place to create a DOS partition.

Action: Look for typographical errors. Reinvestigate your partitioning requirements.

No space to create a DOS partition

Explanation: FDISK. You chose the "Create DOS Partition" option on the current fixed disk which has no space to create a DOS partition.

Action: Remove or reduce the size of the existing partition. Then run FDISK again to create the DOS partition(s).

No subdirectories exist

Explanation: TREE. Informational message. The specified drive contains only a root directory. Therefore, there is no directory path to display.

Action: No action required.

Non-DOS diskette

Explanation: CHKDSK and Commands. The format of the diskette accessed was not recognized.

Action: Format the diskette using the FORMAT command.

Non-System disk or disk error
Replace and strike any key when ready

Explanation: Startup. No entry exists for IBMBIO.COM or IBMDOS.COM in the directory; or a disk read error occurred when you started up the system.

Action: Insert a DOS diskette in drive A and then restart your system.

***** Not able to back up file *****

Explanation: BACKUP. The files cannot be backed up due to a file sharing conflict.

Action: Retry the request at a later time by specifying the same backup command but include the /M parameter.

***** Not able to restore file *****

Explanation: RESTORE. The file you want to restore cannot be opened due to a sharing conflict.

Action: No action required.

Not enough memory

Explanation: SHARE,REDIR. The available memory is less than what SHARE and REDIR need to start. They terminate without installation.

Action: No action required.

Not enough room to merge the entire file

Explanation: EDLIN. Informational message. A Transfer command was unable to merge the entire contents of the specified file because of insufficient memory. Only part of the file was merged.

Action: Either reduce size of one of the files being merged, or install more memory.

Not found

Explanation: EDLIN. Informational message. Edlin could not find the string specified by the REPLACE text or SEARCH text commands within the specified range of lines. Or, if a search is resumed by replying **N** to the **OK?** prompt, no further occurrences of the string were found.

Action: Check to be sure you properly used upper and lower case for the string to be searched.

Out of environment space

Explanation: DOS. Informational message. DOS was unable to accept the SET command you just issued because it was unable to expand the area in which the environment information is kept.

This normally occurs when you try to add to the environment after loading a program which makes itself resident (PRINT, MODE, or GRAPHICS for example).

Action: No action required.

Out of space on list file

Explanation: LINK. There is not enough disk space for the list file.

Action: Use a disk with enough free space to hold the file.

Out of space on run file

Explanation: LINK. This error usually occurs when there is not enough disk space for the Run file (.EXE).

Action: Use a disk with enough free space to hold the file.

Out of space on scratch file

Explanation: LINK. The disk that the linker is using for scratch file is full.

Action: Delete some files on that disk, or replace with another diskette, and restart the linker.

Out of space on VM.TMP

Explanation: LINK. No more disk space remained to expand the VM.TMP file.

Action: Use a disk with enough free space to hold this file.

P

Parameters not compatible

Explanation: FORMAT. You attempted to use two parameters that are not compatible with each other (/**B** and /**V** for example).

Action: Review the FORMAT command. Correct parameter and re-enter the command.

Parameter not compatible with fixed disk

Explanation: FORMAT. You wrongly specified the /**1** or /**8** parameter while formatting a fixed disk. Neither of these parameters is valid for a fixed disk.

Action: Review the FORMAT command. Correct parameter and re-enter the command.

Parity error or nonexistent memory error detected

Explanation: DEBUG.

Action: No action required.

Partition 1 is already active

Explanation: FDISK. Informational message. Partition 1 is the only partition defined and it is already marked as active.

Action: No action required.

Partition *xx* made active

Explanation: FDISK. Informational message. Partition *xx* is now marked as bootable.

Action: No action required.

Path not found

Explanation: DOS and commands. A file or path named in a command or command parameter does not exist in the directory of the specified (or default) drive.

Action: Retry the command using the correct path and filename.

Pathname too long

Explanation: PRINT. The path for the file you specified is longer than 64 characters.

Action: No action required.

Please replace original diskette in drive A: and press <ENTER>

Explanation: LINK. This message appears after the .EXE file has been written if the /P switch is given.

Action: Insert the diskette with the list file so that it can be reopened.

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

Explanation: RECOVER.

Action: Insert the diskette to be recovered in the indicated drive and press any character key.

Print queue is empty

Explanation: PRINT. Informational message. There are currently no files being processed by PRINT.

Action: No action required.

Print queue is full

Explanation: PRINT. You tried to add more than the limit of ten files to the print queue. Ten files is the default. You can set the limit to 32 files. See the PRINT command.

Action: Wait until a file is printed before you add another file to the print queue.

Printer error

Explanation: MODE. The MODE command (option 1) was unable to set the printer mode because:

- There is an I/O error
- The printer is out of paper (or POWER OFF)
- There is a printer time out (not ready) condition
- The printer is offline

Action: Determine which of above conditions caused the error message and take corrective action.

Printer lines per inch set

Explanation: MODE. Informational message. You tried to set the printer vertical spacing to the specified 6 or 8 lines per inch.

Action: If the attempt was unsuccessful, an error message follows this message on the screen.

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

Explanation: CHKDSK. The file allocation table identification byte contains invalid information. Either the disk was not formatted by DOS or has become badly damaged.

Action: If you did not use the /**F** parameter, and you reply **Y**, CHKDSK will indicate its possible corrective actions without actually changing the disk. We recommend doing this first, before you consider using the /F switch and replying **Y**.

Processing cannot continue

Explanation: CHKDSK. Informational message. This message is followed by another message which explains why CHKDSK cannot continue.

This message is normally issued when there is not enough memory.

Action: No action required.

Program size exceeds capacity of LINK, limit 704K

Explanation: LINK. Load module was too large for processing.

Action: Reduce the size of the program.

Program terminated normally

Explanation: DEBUG. This message informs you that the program that was executed by issuing G(o),T(race), or P(roceed) has completed via an INT 20, INT 27, INT 21, or other expected manner.

Action: No action required.

Program too big to fit in memory

Explanation: DOS. The file containing the external command cannot be loaded because it is larger than the available free memory.

Action: Reduce the number in the BUFFERS= parameter in your CONFIG.SYS file (if you have specified BUFFERS=), restart your system, and reissue the command.

If the message reappears, your system does not have enough memory to execute the command.

R

Read error in:
***x*:\level 1\level 2.**

Explanation: EDLIN. An error occurred while reading file *x:\xxxx\xxxx* into memory.

Action: Copy the file or a backup of the file to a different disk and try again.

Reinsert diskette for drive *x*
and strike Enter when ready

Explanation: FORMAT. This message, which usually occurs after you enter FORMAT /S, means:

- DOS filled the memory with system files, but could not read all of the files, into memory because of insufficient memory size.
- After asking for the new diskette, FORMAT started formatting it and all of the files in memory on the new diskette.
- FORMAT then asked that the DOS diskette be inserted so it could finish loading the rest of the DOS files into the memory.

Action: FORMAT is now asking you to insert the new diskette again so it can finish the task of writing the DOS files onto the new diskette.

Relocation table overflow

Explanation: LINK. More than 13000 long calls or long jumps or other long pointers in the program.

Action: Rewrite program replacing long references with short references where possible and recreate object module. **Note:** Pascal and FORTRAN users should first try turning debugging off.

Resident part of PRINT installed

Explanation: PRINT. Informational message. The message appears the first time you use the PRINT command.

This message indicates that a program has been loaded into memory to handle subsequent PRINT commands. Available memory for your applications has been reduced by approximately 3200 bytes.

Action: No action required.

Resident portion of MODE loaded

Explanation: MODE. Informational message.

This message indicates that when MODE is invoked for a non-screen-setting function it is sometimes necessary to load a portion of code to be made permanently resident.

Action: No action required.

Restore file sequence error

Explanation: RESTORE. The file was not restored because the diskettes were not inserted in sequential order.

Action: Retry the restore, inserting the diskettes in sequential order.

***** Restoring files from drive** *y* *******
Source: *x*

Explanation: RESTORE. This is an informational message telling you that the files on the source drive are being restored.

Action: No action required.

S

Sector size adjusted

Explanation: VDISK. VDISK found it necessary to adjust the sector size value in the DEVICE=VDISK.SYS in the CONFIG.SYS command.

Action: No action required.

Sector size too large in file <filename>

Explanation: Startup. The device driver named in <filename> specifies a device sector size larger than the devices previously defined to DOS.

Action: Reduce sector size to conform with the sector size of DOS.

If this is a purchased program, return it to your dealer.

Segment limit set too high, exceeds 1024

Explanation: LINK. The count specified for segments in the /X: parameter is too large.

Action: No action required. Linker will abort.

Segment limit too high

Explanation: LINK. There is insufficient memory for the linker to allocate tables to describe the number of segments requested (either the value specified with /X or the default: 256).

Action: Either try the link again using /X to select a smaller number of segments (e.g. 128, if the default was used previously) or restart the system to free some memory that was used by installable device drivers such as VDISK, and terminate and stay resident programs such as PRINT and GRAPHICS.

Segment size exceeds 64K

Explanation: LINK. This message indicates that you attempted to combine identically named segments which resulted in a segment requirement of greater than 64K bytes. You cannot address more than 64K bytes.

Action: Change segment names in object modules and try link again.

Source and target drives are the same

Explanation: BACKUP and RESTORE. The drive specifier for the source and target drive cannot be the same.

Action: Specify a different drive letter for the source and target drives.

Source does not contain backup files

Explanation: The source media does not contain files created by the BACKUP command.

Action: No action required.

SHARE already installed

Explanation: SHARE. You have already loaded SHARE. SHARE can only be loaded once when you start DOS.

Action: No action required.

Specified command search directory bad

Explanation: Commands. Invalid path name specified.

Action: Specify valid path name containing command processor.

Specified drive does not exist, or is non-removable

Explanation: DISKCOPY and DISKCOMP. The drive specifier is for a fixed disk drive, or does not exist on your computer.

Action: Check the drive specifier and reenter the command.

Source diskette bad or incompatible

Explanation: DISKCOPY. Read errors on the source diskette indicate that it may not be suitable for a DISKCOPY.

Action: Make sure that there is no necessary data on the bad part of the diskette, and compare the files copied to assure that the copy was successful.

Stack size exceeds 65536 bytes

Explanation: LINK. Informational message. The size specified for the stack must be less than or equal to 65536.

Action: No action required.

Symbol defined more than once

Explanation: LINK. The Linker found two or more modules that define a single symbol name.

Action: Check for the following:

- Are you sure you're not linking the same files twice?
- Does one of the modules being linked have a symbol incorrectly identified as *public* instead of *external*?

Symbol table overflow

Explanation: LINK. The user's program has greater than 256K of symbolic information (publics, externs, segments, groups, classes, files).

Action: Combine modules and/or segments and recreate the object files. Eliminate as many public symbols as possible.

Syntax error

Explanation: DOS. The command format you typed is improper.

Action: Check to make sure you have used the correct format for this command.

System files restored
The target disk may not be bootable

Explanation: RESTORE. DOS system files IBMBIO.COM or IBMDOS.COM were restored. The diskette may not be bootable if the system files from a previous version of DOS were restored.

Action: SYS the disk with DOS Version 3.00 and copy COMMAND.COM to the root directory of the target disk.

System transferred

Explanation: FORMAT. This message is displayed when you specify FORMAT /S. It is an informational message telling you that the system files have been installed on the formatting disk.

Action: No action required.

System will now restart
Insert DOS diskette in drive A:
Press any key when ready

Explanation: FDISK. Informational message. FDISK requires the system to restart in order to update the partition information.

Action: Place DOS diskette in drive A: and press any key.

T

Target can not be used for backup

Explanation: BACKUP. Your attempt to create files on the target disk failed.

Action: Replace the disk if it is removable. If not, BACKUP

to a different device or restart the system and try again.

Target diskette bad or incompatible

Explanation: DISKCOPY. Errors encountered during the copy indicate that the target diskette is dirty, damaged or of poor quality, or the drive is malfunctioning on some tracks.

Action: Change the target drive or the diskette and try again.

Target diskette may be unusable

Explanation: DISKCOPY. This message follows an unrecoverable read, write, or verify error message. The copy on the target diskette may be incomplete because of the unrecoverable I/O error.

Action:

- If error is on the target diskette, get a fresh diskette for your target, and retry the DISKCOPY command.
- If the error is on the source diskette, copy all files from the source diskette to another diskette. Then try to reformat the source diskette.

Target diskette write protected
Correct, then strike any key

Explanation: DISKCOPY. You are trying to produce a copy on a diskette that is write protected.

Action: Either remove the write protect tab, or use another diskette that is not write protected.

Target is full

Explanation: RESTORE. The disk you are restoring to is full.

Action: Delete any unnecessary files you have and try again, or RESTORE to a destination disk with more space.

Terminate batch job (Y/N)?

Explanation: DOS. This message appears when you press Ctrl-Break while DOS is processing a batch file.

Action: Press Y to stop processing the batch file. Pressing N only ends the command that was executing when Ctrl-Break was pressed; processing resumes with the next command in the batch file.

Terminated by user

Explanation: LINK. The user entered Ctrl-Break in response to one of the linker prompts.

Action: No action required.

The current active partition is *x*.

Explanation: FDISK. Informational message. The "Change Active Partition" option displays the active partition on the current fixed disk.

Action: No action required.

The last file was not restored

Explanation: RESTORE. You stopped RESTORE before it completely restored the last file listed, or there was not enough room on the fixed disk and RESTORE deleted the partially restored file.

Action: If RESTORE has ended, you can re-enter the RESTORE command with the filename of the file(s) not restored to continue from the point where RESTORE stopped.

If the problem occurred because you ran out of room on the fixed disk, you must evaluate which files to keep and which ones to delete. Then continue the execution of RESTORE.

There was/were *number* errors detected

Explanation: LINK. This message is displayed for your information at the end of the link session.

Action: No action required.

Too many block devices

Explanation: CONFIG.SYS. You attempted to install more than the system limit of 26 block device units.

Action: Change your CONFIG.SYS "DEVICE=" command so that only 26 block device units (including those automatically installed by DOS for disk/diskette drives) are installed. See Chapter 4, "Configuring Your System," of the *DOS Reference*, and Chapter 2, "Installable Device Drivers," of the *DOS Technical Reference* for more information.

Too many external symbols, limit 510 per module

Explanation: LINK. More than the limit of 510 external symbols were specified in a module.

Action: Break up the module.

Too many files open

Explanation: EDLIN. EDLIN tried to open the specified file, but was not able to.

Action: Increase the FILES= value in the CONFIG.SYS file.

Too many group-, segment-, and class-names, limit 254 per module

Explanation: LINK. User's program contains too many group, segment, and class names.

Action: Reduce the number of groups, segments, or classes and recreate the object files.

Too many groups

Explanation: LINK. The limit is 9 including DGROUP.

This message indicates that the limit was exceeded.

Action: Reduce the number of groups.

Too many GRPDEFs, limit 8 per module.

Explanation: LINK. Linker encountered more than 8 GRPDEFs in a single module.

Action: Reduce the number of groups or split up the module.

Too many libraries, limit is 16

Explanation: LINK. User tried to open more than 16 libraries.

Action: Combine libraries or link modules that require fewer libraries.

Too many overlays

Explanation: LINK. The limit is 64 overlays. This message indicates that the limit was exceeded.

Action: Reduce the number of overlays.

Too many public symbols

Explanation: LINK. The limit is 2048 public symbols. This message indicates that the limit was exceeded.

Action: Reduce the number of public symbols.

Too many segments, limit 255 per module

Explanation: LINK. The user's object module has more than 255 segments.

Action: Split the modules or combine segments.

Too many segments, use /X:N (256<N<1025)

Explanation: LINK. The user has specified more than the maximum of 1024 segments.

Action:Relink using the /X parameter with appropriate number of segments specified.

Too many TYPDEFs, limit 255 per module

Explanation: LINK. TYPDEFs are records emitted by the compiler to describe communal variables.

Action: Create two sources from the old source, dividing the communal variable definitions between them; recompile and relink.

Top level process aborted, cannot continue

Explanation: During boot, COMMAND detected a disk error, and you chose to ABORT. The error is unrecoverable and COMMAND has no choice but to halt.

Action: The recommended course of action is to use another disk.

Total disk space is *xxxx* cylinders.

Explanation: FDISK. Informational message. This message shows the total space on the current fixed disk.

Action: No action required.

Transfer size adjusted

Explanation: VDISK. The maximum number of sectors to transfer specified (m) was not in the range 1-8. The value 8 will be used by VDISK.

Action: No action required.

Tree past this point not processed

Explanation: CHKDSK. Informational message. CHKDSK is unable to continue processing past the directory path currently being examined of the error displayed in the previous message.

Action: No action required.

Unable to create directory

Explanation: MKDIR. The directory you want to create already exists, or one of the directory path names you specified could not be found, or you attempted to add a directory to the root directory and it is full. Make sure a file by that name does not already exist in that directory.

Action: Do the following:

- Check to see if a directory by that name exists in the parent directory (or current directory)
- Recheck all your directory names to make sure they are valid
- Use CHKDSK to see if your directory is full

Unable to write BOOT

Explanation: FORMAT. The first track of the diskette or DOS partition is bad. The BOOT record could not be written on it. The diskette or DOS partition is not usable.

Action: Obtain another diskette and retry the FORMAT command.

Unable to shift screen left

Explanation: MODE. Shifting the screen further to the left would exceed the allowable limit.

Action: No action required.

Unable to shift screen right

Explanation: MODE. Shifting the screen further to the right would exceed the allowable limit.

Action: No action required.

Unexpected end-of-file on library

Explanation: LINK. Informational message. This is caused by an error in the library file.

This usually means that the object file contains bytes that are of the same value as end-of-file. If this occurs, LINK continues processing to the physical end-of-file as given in the directory.

Action: No action required.

Unexpected end-of-file on scratch file

Explanation: LINK. Diskette containing VM.TMP was removed.

Action: Restart linker.

Unexpected end of file on VM.TMP

Explanation: LINK. Informational message. The diskette containing VM.TMP was removed.

Action: No action required.

Unrecognized command in CONFIG.SYS

Explanation: Startup. An invalid command was detected in the configuration file CONFIG.SYS.

Action: Edit the file, correct the invalid command, and restart DOS.

Unrecognized switch error: *xxxxxx*

Explanation: LINK. User entered an unrecognized character after the switch indicator /.

Action: No action required. Linker will abort.

Unrecoverable error on directory

Explanation: CHKDSK. CHKDSK encountered an error while checking the directory.

Action: No action required.

Unrecoverable file sharing error

Explanation: SHARE. A file sharing conflict occurred. Therefore, files cannot be restored.

Action: No action required.

Unrecoverable read error on drive *x*
Track *xx*, **side** *x*

Explanation: DISKCOMP. Four attempts were made to read the data from the diskette in the specified drive. The data could not be read from the indicated track and side.

Action: If the error occurred on the target diskette (just created by DISKCOPY) get a fresh diskette and retry the DISKCOPY and DISKCOMP commands. Otherwise, use:

```
COPY *.*
```

to copy all files from the damaged diskette to another diskette. Then reformat the bad diskette or else discard it.

Unrecoverable read error drive A:
Track *xx*, side *x*

Explanation: DISKCOPY. Four attempts were made to read the data from the source diskette. DISKCOPY continues copying, but the copy may contain incomplete data.

Action: Use the following form of the COPY comand:

```
COPY *.*
```

to copy all files from the damaged diskette to another diskette. Reformat the bad diskette, or else discard it.

Unrecoverable write error on target
Track *xx*, **side** *x*

Explanation: DISKCOPY. Several attempts were made to write the data to the target diskette. DISKCOPY continues copying, but the copy may contain incomplete data.

Action: Obtain a fresh diskette and re-enter the DISKCOPY command. Use FORMAT on the bad diskette to see if can be reused. If it is a bad diskette, discard it.

Unresolved externals: list

Explanation: LINK. The external symbols listed were not defined in the modules or library files that you specified.

Action: Do not attempt to run the executable file created by the linker.

Make sure you specified all appropriate object modules and libraries. Check the source code for the program that caused the message and make corrections to that program.

V

VDISK not installed - insufficient memory

Explanation: VDISK. This message occurs if:

- Less than 64K of available memory would exist even after attempting to adjust the virtual disk size and number of directory entries.

- You have specified the /E parameter and the computer does not contain extended memory.

- The amount of available extended memory is too small to install the virtual virtual disk, even after adjusting the parameters.

Action: No action required.

VDISK Version 2.0 Virtual Disk *x*

Explanation: VDISK. Issued when VDISK receives control to install a virtual disk, and *x* tells you which drive letter is being assigned to the virtual disk.

Action: No action required.

VERIFY is off/on

Explanation: VERIFY. Informational message.

Action: No action required.

VM.TMP is an illegal file name and has been ignored

Explanation: LINK.VM.TMP cannot be used for an object filename. Informational message that is meant as a a warning.

Action: No action required.

Volume label (11 characters, ENTER for none) ?

Explanation: FORMAT. You are requested to enter a 1 to 11 character volume label that will be written on the disk being formatted.

Action: If you do not want a volume label on the disk, press only the ENTER key.

Warning! All data on non-removable disk drive *x*
will be lost
Proceed with Format (Y/N)?

Explanation: FORMAT. Prompt telling you that the fixed disk will be formatted.

Action: If you do not want to format the fixed disk, type **n**. If you want to format the fixed disk, type **y**.

Warning! Data in the DOS partition
could be DESTROYED. Do you wish to
continue.....? [d]

Explanation: FDISK. The "Delete DOS Partition" option is warning you that if you continue, all data in the DOS partition on the current fixed disk could be destroyed.

Action: If you press Enter, the DOS partition will NOT be destroyed. If you do wish to delete the DOS partition, type Y and press Enter.

Warning-directory full
xxx **file(s) recovered**

Explanation: RECOVER. There is insufficient directory space to recover more files.

Action: Copy some of the files to another disk, erase them from this disk, and run RECOVER again.

Warning! Diskette is out of sequence
Replace the diskette or continue if okay
Strike any key when ready

Explanation: RESTORE. The backup diskette is not the next one in sequence.

Action: Replace the diskette unless you are sure no files on the diskette(s) you skipped would be restored. RESTORE will continue when you press a key. This message will be repeated if you try to skip a diskette that contains part of a file being restored.

Warning! File *xx*
is a read only file
Replace the file (Y/N)?

Explanation: RESTORE. The indicated file is read-only.

Action: Enter **Y** if you want to replace it or **N** if you do not. RESTORE will continue after you press ENTER. You will see this message only if you specified the **/P** option.

Warning! File *xx*
was changed after it was backed up
Replace the file (Y/N)?

Explanation: RESTORE. The indicated file on the fixed disk has a later date and time than the corresponding file on the backup diskette.

Action: Enter **Y** if you want to replace it with the backed up version or **N** if you do not. RESTORE will continue after you press ENTER. You will see this message only if you specified the **/P** option.

Warning! Files in the target
root directory will be erased
Strike any key when ready

Explanation: RESTORE. This is a warning prompt to tell you that the files in the root directory will be erased.

Action: If you do not want to proceed, press Ctrl-Break. To continue, strike any key.

Warning! No files were found to back up

Explanation: BACKUP. Informational message telling you that no files were found to back up.

Action: No action required.

Warning! No files were found to restore

Explanation: RESTORE. No backup files were found that matched the restore file specification.

Action: Make sure the criteria you specified for BACKUP is what you want. Otherwise, this is an informational message.

Warning: No stack segment

Explanation: LINK. Informational message. None of the object modules specified contain a statement allocating stack space.

Action: No action required.

Warning! Target is Full

Explanation: BACKUP or RESTORE. The target device is full. No more files can be restored.

Action: No action required.

Writing *xxx* bytes

Explanation: DEBUG. Informational message telling how many bytes are being written.

Action: No action required.

X

***x* is not a choice. Enter a choice.**

Explanation: FDISK. You entered *x* which is not a choice for this question.

Action: Enter a valid choice.

x is not choice. Enter Y or N

Explanation: FDISK. You entered *x* which is not a choice for this question.

Action: Enter Y or N.

xxxxxx **is not a valid library**

Explanation: LINK. The file specified as a library is invalid.

Action: No action required. Linker will abort.

xxxxxxxxxx **bytes disk space freed**

Explanation: CHKDSK. Informational message. Disk space marked as allocated was not associated with a file. If you used the **/F** parameter, the space was freed and made available.

Action: No action required.

xxxx **error on file** *yyyy*

Explanation: PRINT. This message appears on the printer. While attempting to read data from file *yyyy* for printing, a disk error of type *xxxx* was encountered. Printing of that file is stopped.

Action: Check that disk drive is ready.

xxx **lost clusters found in** *yyy* **chains.**
Convert lost chains to files (Y/N)?

Explanation: CHKDSK. Ctrl-Break was entered during a disk I/O operation. CHKDSK did not clean up the disk after encountering the Ctrl-Break.

Action: If you reply **Y** and you have used the /F parameter, CHKDSK will recover each chain into a separate file, otherwise, if you reply **N**, CHKDSK frees the blocks up so they can be allocated to new files. If CHKDSK was specified (no /F), then messages displayed afterwards are informational (no corrective action was taken).

xxxxxx **of** *xxxxxx* **bytes recovered**

Explanation: RECOVER. This is an informative message indicating the number of bytes of the specified file that were recovered.

Action: No action required.

***** Files were backed up xx/xx/xxxx *****

Explanation: RESTORE. Informational message. The files on the backup diskette were backed up on the indicated date.

Action: No action required.

*** Restoring files from drive A:-
diskette xx

Explanation: RESTORE. Informational message. This message is followed by a list of files that were restored from the indicated diskette.

Action: No action required.

--More--

Explanation: MORE. The screen is full and there is more data waiting to be displayed.

Action: Press any character to see the next full screen.

Numbers

10 Mismatches - ending compare

Explanation: COMP. Informational message. Ten mismatched locations were detected in the files being compared. COMP assumes that the files are so different that further comparisons would serve no purpose.

Action: No action required.

Notes:

Index

Special Characters A

. - period 8-8, 8-16
.COM file format 7-99
+ (plus sign)
  in automatic response file 9-21
$$$ - filename extension 8-5
*
  EDLIN prompt 8-5, 8-10
  global filename character 2-8
* - EDLIN prompt 8-5, 8-10
* - global filename character 2-8
- (DEBUG prompt) 10-14
/P parameter 7-79
/V parameter 7-61
/W parameter 7-79
/1 parameter 7-90
/1 parameter, DISKCOMP 7-84
/8 parameter, DISKCOMP 7-84
% (percent sign) 7-28
? - global filename character 2-7
- pound sign 8-7
@ symbol (linker)
= equal sign 7-10
A (Append Lines) Command 8-11
A (Assemble) Command 10-15
A> prompt 7-10
abort read/write operation A-3
about messages 1-8
absolute sector 10-52
absolute segment address
  how to determine 9-26
AC flag set condition 10-44
accessing a file 4-22
adding hexadecimal values 10-30
address - DEBUG parameter 10-6
address, disk transfer 10-5
analyze
  diskettes 7-105
  the directory 7-48
  the File Allocation Table 7-48
Append Lines command 8-11
applications, random/sequential 4-8
ASCII characters 10-20
ASCII representation 10-5
ASCII values 10-11
Assemble Command 10-15
assembler 9-4
ASSIGN (Drive) Command 7-14
ASSIGN drive command 7-14

Asynchronous
Communications
Adapter 7-132, 7-137, 7-138
ATTRIB (Attribute)
Command 7-17
ATTRIB command 7-17
AUTOEXEC.BAT file 7-27,
7-75
automatic program
execution 7-27
automatic response file
linker 9-21
AUX - reserved device
name 2-5
auxiliary carry flag 10-44
AX register 10-4

B

BACKUP Command 7-19
backup diskette 7-86
backup file, edit 8-5
BAK filename extension 8-6,
8-18, 8-27
BAT filename extension 7-24
Batch commands
ECHO 7-30
FOR 7-32
GOTO 7-33
IF 7-34
SHIFT 7-40
batch file 7-24, 7-38, 7-39,
7-180
Batch File Commands 7-24
batch file, using with
PATH 5-9
batch file, where DOS looks
for 5-15
batch processing 7-24
books, about DOS 1-3
boundary, paragraph 9-6
boundary, 16-byte 10-20
boundary, 8-byte 10-20
BP register 10-4
brackets, square 7-10
BREAK (Control Break)
Command 7-43
BREAK Command 4-6, 7-43
breakpoint 10-27
buffer, what is a 4-7
BUFFERS Command 4-7
BX register 10-4, 10-34
byte - DEBUG
parameter 10-7
byte contents
display 10-23
fill 10-26
replace 10-23

C

C (Compare)
Command 10-19
C (Copy Lines)
Command 8-12
carry flag 10-44
change console, CTTY 7-72
change date 7-74
change diskettes 7-38
change filenames 7-156
change time 7-180
changing directories 5-7
changing the current
directory 5-14
CHDIR (Change Directory)
Command 7-45
CHDIR command 7-45
check for break 7-43
CHKDSK (Check Disk)
Command 7-48

CHKDSK command 7-48
class 9-7
clear condition 10-43
clear screen 7-52
CLS (Clear Screen) Command 7-52
codes, 8088 instruction 10-28
colon 7-10
color graphics adapter 7-113
Color/Graphics Monitor Adapter 7-132
COM filename extension 7-8
comma 7-10
COMMAND (Secondary Command Processor) Command 7-53
COMMAND command 7-53
command line
  linker 9-18
command parameters
  DEBUG 10-6
  EDLIN 8-7
command prompt, DEBUG 10-13
command prompts, linker 9-8
COMMAND.COM 7-106, 10-4
commands
  DEBUG 10-13
  DOS 7-5
  EDLIN 8-9
commands for directories 5-10
commands, end 7-11
commands, new 1-4
commands, summary of
commands, where DOS looks for 5-15
Communications Adapter 7-137
COMP (Compare Files) Command 7-55
COMP command 7-55, 7-86
Compare command 10-19
comparing diskettes 7-84
comparing files 7-55
comparing memory 10-19
compatibility, DISKCOMP 7-88
compatibility, DISKCOPY 7-94
compatibility, drives and diskettes 1-7
computer, size of your 4-9
COMSPEC 7-165
COM1 - reserved device name 2-5
CON - reserved name for console/keyboard 2-5
concatenation 7-60, 7-67
CONFIG.SYS 4-3
Configuration Commands 4-5
  BREAK 4-6
  BUFFERS 4-7
  COUNTRY 4-11
  DEVICE 4-13
  FCBS 4-19
  FILES 4-22
  LASTDRIVE 4-24
  SHELL 4-25
console/keyboard 2-5
control keys 7-11, 8-9, 10-13
copy and combine files 7-67
COPY command 7-24, 7-60, 7-86, 7-92
Copy Lines command 8-12
copy with different filename 7-65
copy with same filename 7-63
copying diskettes 7-90
copying DOS to DOS partition 3-22
copying files 7-60
copying your DOS diskette 1-3, 7-162
country code 7-162
COUNTRY Command 4-11

creating a .BAT file 7-28
creating a batch file 7-24
Creating a CONFIG.SYS File 4-4
creating a new file 8-19
creating a subdirectory 5-11
CS register 10-4, 10-27, 10-29, 10-32, 10-50, 10-53
Ctrl-Break 9-8
Ctrl-Break keys 7-11, 8-9, 8-16, 8-19, 10-13
Ctrl-Num Lock keys 7-11, 8-9
Ctrl-NumLock keys 10-13
Ctrl-Z keys 8-29
Ctrl–Break keys 7-38
Ctrl–Z character 7-68
Ctrl—PrtSc keys 7-185
CTTY (Change Console) Command 7-72
CTTY command 7-72
current directory 5-6
current directory, changing or displaying 5-14
CX register 10-4, 10-5, 10-34, 10-54
CY flag set condition 10-44

D

d:
  default 2-3
  parameter 2-3
D (Delete Lines) Command 8-13
D (Dump) Command 10-20
Date
  change 7-74
  enter 7-74
DATE Command 7-74
date format, select 7-162
DEBUG commands
  A (Assemble) 10-15
  C (Compare) 10-19
  D (Dump) 10-20
  E (Enter) 10-23
  F (Fill) 10-26
  G (Go) 10-27
  H (Hexarithmetic) 10-30
  I (Input) 10-31
  L (Load) 10-32
  M (Move) 10-35
  N (Name) 10-36
  O (Output) 10-38
  P (Proceed) 10-39
  Q (Quit) 10-40
  R (Register) 10-41
  S (Search) 10-46
  T (Trace) 10-47
  U (Unassemble) 10-49
  W (Write) 10-52
DEBUG program
  command parameters 10-6
  commands 10-13
  common information 10-13
  ending 10-40
  how to start 10-4
  prompt 10-14
  what it does 10-3
default disk transfer address 10-5
default drive
  linker 9-8
  parameter 2-3
default segment 10-6
defective tracks 7-105
DEL (Delete) Command 7-77
DEL command 7-77
Delete Lines command 8-13
deleting a directory or subdirectory 5-13
deleting a file 7-77
delimiters 7-11, 8-9, 10-13
destination area 10-35

DEVICE Command 4-13
device error messages A-4
device names, reserved 2-5, 7-11, 7-65
device redirection 6-3
DGROUP 9-14
DI flag clear condition 10-44
DI register 10-4
DIR (Directory) Command 7-79
DIR command 2-8, 7-79, 7-108
direction flag 10-44
director, analyze 7-48
directories 5-3
directories, accessing 5-6
directories, levels 5-4
directory commands 5-10
directory entries 5-5
directory entries, listing 7-79
directory names 5-6
directory search, PATH 7-140
directory structure, displaying 5-15
directory, current 5-6
directory, display 7-182
directory, displaying the current 5-14
directory, how to organize 5-4
directory, make 7-130
directory, remove 7-161
disk transfer address 10-5
DISKCOMP (Compare Diskettes Only) Command 7-84
DISKCOMP command 7-84, 7-92
DISKCOMP compatibility 7-88
DISKCOPY (Copy Diskettes Only) Command 7-90
DISKCOPY command 7-84, 7-90
DISKCOPY compatibility 7-94
diskette and drive compatibility 1-7
diskette drives, types 1-6
diskette, naming 7-127
diskette, restore 7-157
diskettes
  analyze 7-105
  back up 7-86
  change 7-38
  comparing 7-84
  copying 7-90
  defective tracks 7-105
  filenames 2-4
  fragmented 7-92
  initialize 7-105
  preparing 7-105
  recording format 7-105
  status report 7-48
diskettes, types 1-6
display
  byte contents 10-23
  flags 10-42
  lines 8-22
  registers 10-42
  remarks 7-39
display contents of directory 7-182
display directory entries 7-79
Display DOS version number 7-186
display instructions 10-49
displaying a file's contents 7-185
displaying directory structure 5-15
displaying memory 10-20
displaying the current directory 5-14
DN flag set condition 10-44
DOS books 1-3

DOS command, how to enter 7-9
DOS commands
.bat 7-24
ASSIGN 7-14
ATTRIB 7-17
BACKUP 7-19
Batch processing 7-24
BREAK 4-6, 7-43
CHDIR 7-45
CHKDSK 7-48
CLS 7-52
COMMMAND 7-53
common information 7-10
COMP 7-55
COPY 7-60
CTTY 7-72
DATE 7-74
DEL 7-77
DIR 7-79
DISKCOMP 7-84
DISKCOPY 7-90
entering a DOS command 7-9
ERASE 7-96
EXE2BIN 7-98
external 7-8
FDISK 7-101
FIND Filter 7-102
FORMAT 7-105
GRAFTABL 7-113
GRAPHICS 7-115
internal 7-8
JOIN 7-118
KEYBxx 7-123
LABEL 7-127
MKDIR 7-130
MODE 7-132
MORE Filter 7-139
PATH 7-140
PAUSE 7-38
PRINT 7-143
PROMPT 7-149
RECOVER 7-153
REM 7-39
RENAME (or REN) 7-156
RESTORE 7-157
RMDIR 7-161
SELECT 7-162
SET 7-164
SHARE 7-167
SHIFT 7-40
SORT Filter 7-169
SUBSTITUTE 7-172
SYS 7-178
TIME 7-180
TREE 7-182
TYPE 7-185
types of 7-8
VER 7-186
VERIFY 7-187
VOL 7-188
DOS device names 2-5
DOS diskette, about your 1-3
DOS diskette, copying 7-162
DOS diskette, using 1-3
DOS Editing Keys
entering DOS commands 7-11
using DEBUG 10-13
using EDLIN 8-9
DOS files
displaying contents of 7-185
fragmented 7-65
object 10-3
object program 7-185
source 8-3
text 7-185, 8-3
DOS filters 6-7
DOS partition, formatting 3-18
DOS version number, display 7-186

DOS, replacing previous version 3-3
drive 2-3
drive - DEBUG parameter 10-7
drive, assign new 7-14
DS register 10-4, 10-5, 10-23, 10-26, 10-35
/DSALLOCATION linker parameter 9-14
dummy device 2-5
dummy parameters 7-28, 7-29
Dump command 10-20
DX register 10-4

E

E (End Edit) Command 8-18
E (Enter) Command 10-23
ECHO Subcommand 7-30
edit
  backup file 8-5
  existing file 8-5
  partial file 8-11
Edit Line Command 8-16
editing a new file 8-6
editing keys 7-11, 8-9, 10-13
EDLIN
  A (Append Lines) 8-11
  C (Copy Lines) 8-12
  command parameters 8-7
  commands 8-9
  common information 8-9
  creating a batch file 7-24
  D (Delete Lines) 8-13
  E (End Edit) 8-18
  Edit Line 8-16
  how to start 8-5
  I (Insert Lines) 8-19
  L (List Lines) 8-22
  M (Move Lines) 8-25
  P (Page) 8-26
  program 8-3
  prompt 8-5
  Q (Quit Edit) 8-27
  R (Replace Text) 8-28
  S (Search Text) 8-31
  T (Transfer Lines) 8-34
  W (Write Lines) 8-35
EI flag set condition 10-44
End Edit command 8-6, 8-18
end–of–file 7-68
ending commands 7-11
Enter command 10-23
enter date 7-74
Enter key 2-5, 8-9
enter time 7-180
entering a DOS command 7-9
entries, directory 5-5
environment, set 7-164
equal sign (=) 7-10
ERASE Command 7-96
erasing a file 7-96
error message, device A-4
error messages A-3
error, recover from A-3
error, syntax 10-13
ES register 10-4, 10-5
Esc key 8-16
EXE filename extension 7-8, 9-10, 10-5, 10-34, 10-55
execute instructions 10-47
execute program 10-27
executing a .BAT file 7-29
executing files quickly 7-25
EXE2BIN Command 7-98
existing file, edit 8-5
ext 2-4
extension
  .BAK 8-6, 8-18, 8-27
  .COM 7-8
  .EXE 7-8, 10-5, 10-34, 10-55

.EXE filename
extension 9-10
.HEX 10-5, 10-34, 10-55
.MAP 9-11
.OBJ 9-9
$$$ 8-5
extensions 2-4
external commands 7-8

F

F (Fill) Command 10-26
FCB (see File Control Block)
FCBS (File Control Block)
Command 4-19
FDISK 3-6
change active
partition 3-13
create DOS partition 3-9
delete DOS partition 3-15
display partition
information 3-17
loading FDISK 3-7
next fixed disk 3-18
partitioning 3-4
FDISK Command 7-101
FDISK program, loading 3-7
FDISK, DOS partition 3-6
features, DOS 3.10 1-4
file attribute, read–only 7-17
File Control Block
(FCB) 10-36
file sharing 4-20, 7-167
file specification 2-3
file, editing a new 8-6
filename characters, global 2-7
filename extensions
.BAK 8-6, 8-18, 8-27
.BAT 7-24
.COM 7-8
.EXE 7-8, 10-5, 10-34,
10-55
.EXE filename
extension 9-10
.HEX 10-5, 10-34, 10-55
.MAP 9-11
.OBJ 9-9
$$$ 8-5
characters, valid 2-4
filenames
length of 2-4
renaming 7-156
FILES Command 4-22
files, list all 7-81
files, list selected 7-82
files, number opened 4-23
filespec 2-3
filespec - DEBUG
parameter 10-7
Fill command 10-26
filter commands
FIND command 7-102
MORE command 7-139
SORT Command 7-169
filtering data 6-7
FIND Filter Command 7-102
First Asynchronous
Communications Adapter
port 2-5
fixed disk specifiers 3-4
fixed disk, formatting 3-18
fixed disk, partitioning 3-4
fixed disk, preparing your 3-4
fixed disk, restore 7-157
fixups, segment 7-99
flag values 10-41
flags 10-4
flags, display 10-42
FOR Subcommand 7-32
FORMAT Command 7-105
FORMAT compatiblity 7-110
FORMAT status report 7-109

formatting, DOS
partition 3-18
fragmented diskettes 7-92
fragmented files 7-65
F3 key 10-32
F5 key 8-17
F6 key 2-5, 7-66, 8-29

G

G (Go) Command 10-27
generating line numbers 8-3
global filename characters
* 2-7
? 2-7
examples using 2-9
in command name 7-11
in COPY 7-62
in DELETE. 7-78
in DIR 7-79
in ERASE 7-97
in RENAME 7-156
Go command 10-27, 10-54
GOTO Subcommand 7-33
GRAFTABL (Load Graphics Table) Command 7-113
GRAFTABL command 7-113
GRAPHICS (Screen Print) Command 7-115
GRAPHICS command 7-115
group 9-7

H

H (Hexarithmetic) Command 10-30
HEX filename extension 10-5, 10-34, 10-55
Hexarithmetic command 10-30
hidden files 7-50, 7-108
HIGH 9-14
HIGH linker parameter 9-6
high memory 9-14, 10-5

I

I (Input) Command 10-31
I (Insert Lines) Command 8-19
IBMBIO.COM 7-79, 7-106
IBMDOS.COM 7-79, 7-106
IF Subcommand 7-34
initialize
diskettes 7-105
initializing the asynchronous adapter 7-137
Input command 10-31
input file
linker 9-4
input files
Insert Lines command 8-19
insert mode 8-19
inserting lines 8-3
instruction codes, 8088 10-28
Instruction Pointer (IP) 10-4
instructions
display 10-49
execute 10-47
unassemble 10-49
variable length 10-49
internal commands 7-8
interrupt codes 10-28
interrupt flag 10-44
invoking a secondary processor 7-53

invoking one batch file from another 7-25
IP (Instruction Pointer) 10-4
IP register 10-27, 10-41

J

JOIN Command 7-118

K

keyboard 2-5
keyboard code 7-162
keyboard layout, select 7-162
KEYBxx (Load Keyboard) Command 7-123
KEYBxx Command 7-123
keys, control 7-11, 8-9, 10-13
keys, DOS editing 7-11, 8-9, 10-13

L

L (List Lines) Command 8-22
L (Load) Command 10-32
LABEL (Volume Label) Command 7-127
LABEL command 7-127
LASTDRIVE Command 4-24
LINE 9-15
line - EDLIN parameter 8-7
Line Editor Program 8-3
line numbers 8-3
lines, renumber 8-13, 8-19
LINK
  See linker(LINK) Program
linker files
  automatic response 9-4, 9-21
  input 9-4
  library 9-4, 9-12
  listing 9-4, 9-10
  object 9-4, 9-9
  output 9-4
  run 9-4, 9-10
linker parameters 9-14
  /DSALLOCATION 9-14
  /HIGH 9-14
  /LINE 9-15
  /MAP 9-15
  /PAUSE 9-15
  /STACK 9-16
linker prompts 9-9
linker(LINK) Program
  command line 9-18
  command prompts 9-7
  example session 9-23
  messages 9-27
  starting 9-17
list - DEBUG parameter 10-7
list all files 7-81
List Lines command 8-22
list selected files 7-82
listing data lines 8-26
listing directory entries 2-8, 7-79
Load command 10-32, 10-54
load graphics table 7-113
load module 9-16, 9-26
loading programs 7-98
loading standard device drivers 4-13
loading, FDISK program 3-7
  create DOS partition

entire fixed disk for DOS 3-9
part of fixed disk for DOS 3-9
local v
LPT1 - reserved name for printer 2-5

M

M (Move Lines) Command 8-25
M (Move) Command 10-35
make directory, MKDIR 7-130
making a file read-only 7-17
making a subdirectory 5-11
/MAP linker parameter 9-15
MAP extension 9-11
memory
high 9-14, 10-5
low 9-14
memory status report 7-48
memory, loading files into 8-5
messages 1-8, A-3
messages, linker 9-27
MKDIR (Make Directory) Command 7-130
MKDIR command 7-130
MODE Command 7-132
MORE Filter Command 7-139
Move command 10-35
Move Lines command 8-25

N

N (Name) Command 10-36
n - EDLIN parameter 8-8
NA flag clear condition 10-44
Name command 10-33, 10-36
naming a diskette 7-127
NC flag set condition 10-44
new DOS commands 1-4
NG flag set condition 10-44
NUL: - reserved device name 2-5
numbers, line 8-3, 8-13
NV flag clear condition 10-44
NZ flag clear condition 10-44

O

O (Output) Command 10-38
OBJ extension 9-9
object files 10-3
object modules 9-9
in response to linker prompt 9-9
object program files 7-185
one-drive system 7-85, 7-92
operation, suspend system 7-38
optional remarks, PAUSE command 7-38
Output command 10-38
output files
linker 9-5
OV flag set condition 10-44
overflow flag 10-44

P

P (Page) Command 8-26
P (Proceed) Command 10-39
Page command 8-26
paragraph boundary 9-6
parallel printer to
  Asynchronous
  Communications
  Adapter 7-138
parameter
    DEBUG 10-6
    dummy 7-28
    EDLIN 8-7
    testing with different 10-27
parameters, Linker 9-14
parity flag 10-44
partial file, edit 8-11
partition, FDISK 3-6
partitioning, fixed disk 3-4
path 5-6
PATH (Set Search Directory)
  Command 7-140
PATH command 5-6, 5-8,
  7-140
path to a file 5-7
PATH, using in
  AUTOEXEC.BAT 5-9
PAUSE 9-15, A-11
PAUSE command 7-38
PC register 10-41
PE flag set condition 10-44
percent sign (%) 7-28
period (.) 8-8
period (.) 8-16
physical append 7-70
Piping standard I/O 6-6
piping, what is 6-6
PL flag clear condition 10-44
plus sign
    in automatic response
      file 9-21
PO flag clear condition 10-44
portaddress - DEBUG
  parameter 10-8
pound sign (#) 8-7
prepare diskettes 7-105
preparing your fixed disk 3-4
print buffer size 7-143
PRINT Command 7-143
print device, specifying 7-143
print queue size 7-143
printer 2-5, 7-132
printing files 7-143
printing graphics 7-115
PRN - reserved name for
  printer 2-5
Proceed command 10-39
program execution, stop 10-27
Program Segment Prefix 10-5
program, FDISK 3-6
prompt
    DEBUG 10-14
    EDLIN 8-5
PROMPT (Set System Prompt)
  Command 7-149
PROMPT Command 7-149
protecting a file 7-17
protocol parameters 7-137
public symbols 9-25
punctuation 7-10

Q

Q (Quit Edit) Command 8-27
Q (Quit) Command 10-40
questions mark 7-10
queue print files 7-143
Quit command 10-40
Quit Edit command 8-27
quotation marks 10-11

R

R (Register) Command 10-41
R (Replace Text) Command 8-28
random/sequential applications 4-8
range - DEBUG parameter 10-8, 10-9
read-only 7-17
read/write requests 4-8
read–only attribute 7-17
recording format, diskette 7-105
RECOVER Command 7-153
recover from error A-3
redirection of I/O devices 6-3
Register command 10-41
registername - DEBUG parameter 10-9
registernames, valid 10-41
registers, display 10-42
relative sector number 10-10
relative zero 9-25
relocatable loader 9-4
REM command 7-39
remarks, display 7-39
remarks, PAUSE command 7-38
remote v
remove directory, RMDIR 7-161
removing a directory 5-13
REN Command 7-156
RENAME (or REN) Command 7-156
RENAME Command 7-156, 10-55
renumber lines 8-13, 8-19
replace byte contents 10-23
Replace Text command 8-28
replacing previous version of DOS 3-3
reserved device names 2-5, 7-11, 7-65
responses to the system A-3
RESTORE Command 7-157
retry read/write operation A-3
RMDIR (Remove Directory) Command 7-161
RMDIR Command 7-161
root directory, getting back to 5-7
run file 9-10

S

S (Search Text) Command 8-31
S (Search) Command 10-46
saving diskette space 7-98
screen 2-5
screen print, graphics 7-115
screen, clear CLS 7-52
Search command 10-46
search directory, PATH 7-140
Search Text command 8-31
secondary command processor 7-53
sector - DEBUG parameter 10-10
sector number, relative 10-10
sector, absolute 10-52
sectors 7-92, 10-10
segment 9-6, 9-9
SEGMENT command 9-16
segment fixups 7-99
segment registers 10-4, 10-50
segment, default 10-6
segments, class 9-7
SELECT Command 7-162

SELECT procedure 1-3
select, date and time format 7-162
select, keyboard layout 7-162
semicolon delimiter 7-10
SET (Set Environment) Command 7-164
SET Command 7-164
set condition 10-43
set system prompt command 7-149
SHARE Command 7-167
sharing, file 7-167
SHELL Command 4-25
SHIFT Subcommand 7-40
SI register 10-4
sign flag 10-44
slashes 7-10
SORT Filter Command 7-169
source area 10-35
source drive 7-10
source files 8-3
SP (Stack Pointer) 10-4
space delimiter 7-10
special characters 2-7
specifying a drive 2-3
specifying, print device 7-143
square brackets 7-10
SS register 10-4
stack allocation statement 9-16
/STACK linker parameter
Stack Pointer (SP) 10-4
standard I/O device redirection 6-3
standard I/O, piping 6-6
starting DEBUG 10-4
starting DOS from fixed disk 3-23
starting EDLIN 8-5
starting the linker 9-17
status report 7-48
stop program execution 10-27
string - DEBUG parameter 10-11
string - EDLIN parameter 8-8, 8-9
subdirectories 5-3
subdirectories, accessing 5-6
subdirectories, changing 5-6
subdirectory, creating a 5-11
SUBST 7-172
SUBST(Substitute) Command 7-172
SUBSTITUTE Command 7-172
summing files 7-70
suspend system operation 7-38
symbols, global and public 9-15
syntax error 10-13
SYS (System) Command 7-178
SYS Command 7-178
system configuration commands 4-3
system devices 2-5
system files, transfer 7-178
system prompt 7-10
system prompt command, set 7-149
system prompt, command is complete 7-10

T

T (Trace) Command 10-47
T (Transfer Lines) Command 8-34
target drive 7-12
temporary file, VM.TMP
terminate commands 7-11
text files 7-185, 8-3

Time command 7-180
enter 7-180
time format, select 7-162
time, change 7-180
top–level command processor 4-25
Trace command 10-47, 10-54
tracks, defective 7-105
Transfer Lines command 8-34
transfer system files 7-178
TREE Command 7-182
tree-structure, displaying 5-15
tree-structured directories 5-3
TYPE Command 7-185
types of diskette drives 1-6
double sided (320/360K) 1-6
high capacity (1.2MB) 1-6
single sided (160/180K) 1-6
types of diskettes 1-6
double sided (320/360K) 1-6
high capacity (1.2MB) 1-6
single sided (160/180K) 1-6
types of DOS commands 7-8
typing the contents of a file 7-185

U

U (Unassemble) Command 10-49
Unassemble command 10-49
unassemble instructions 10-49
unprintable characters 10-20
UP flag clear condition 10-44
using applications v
using your DOS diskettes 1-3

V

value 10-12
variable length instructions 10-49
VDISK.SYS 4-14
installing in config.sys 4-15
VER (Version) Command 7-186
VER Command 7-186
VERIFY Command 7-187
virtual disk 4-14
VM.TMP temporary files
VOL (Volume) Command 7-188
VOLUME Command 7-188
volume label 7-127

W

W (Write Lines) Command 8-35
W (Write) Command 10-52
where DOS looks for commands and batch files 5-15
Write command 10-52
Write Lines command 8-35

Z

zero flag 10-44
ZR flag set condition 10-44

The Personal Computer
Programming Family

Reader's Comment Form

DOS Reference 6138519

Your comments assist us in improving the usefulness of our publication; they are an important part of the input used for revisions.

IBM may use and distribute any of the information you supply in any way it believes appropriate without incurring any obligation whatever. You may, of course, continue to use the information you supply.

Please do not use this form for technical questions regarding the IBM Personal Computer or programs for the IBM Personal Computer, or for requests for additional publications; this only delays the response. Instead, direct your inquiries or request to your authorized IBM Personal Computer dealer.

Comments:

Notes:

Notes:

Notes:

Notes:

Notes:

Notes:

Notes:

Notes:

IBM United Kingdom
International Products Limited
PO Box 41, North Harbour
Portsmouth, PO6 3AU
England

Printed in Great Britain by Collins, Glasgow