M20 PERSONAL COMPUTER

MS-DOS Disk Operating System

User Guide

olivetti

M20 PERSONAL COMPUTER

MS-DOS Disk Operating System

User Guide

olivetti

PREFACE

This book describes the Microsoft Disk Operating System (MS-DOS) as implemented on your M20. It is directed at the user who has some experience of computer programming and is familiar with computing terminology.

Chapter 1 provides a general introduction to MS-DOS.

Chapters 2 to 6 tell the user how to start MS-DOS, and how to perform the more common operations.

Chapters 7 to 11 provide detailed references of MS-DOS commands, the Line Editor (EDLIN), the linker (MS-LINK), the debugger (DEBUG), and the File Comparison utility (FC), respectively. These chapters are organised alphabetically. Each command description includes the command purpose, its syntax, the characteristics of the command, and examples.

REFERENCES:

M20 Installation and Operations Guide
(Code 3933340 V (0))

DISTRIBUTION: General (G)

EDITION: October 1983

RELEASE: 2.0

1ST NEWSLETTER: November 1983

PUBLICATION ISSUED BY:

Ing. C. Olivetti & C., S.p.A.
Direzione Documentazione
77, Via Jervis - 10015 IVREA (Italy)

Title: MS-DOS USER GUIDE

Newsletter Code: 3984671 H
Date: 15 December 1983
Publication Code: 3984670 G
Previous Newsletters: None

This Newsletter provides updated pages for the subject publication.

The last level completed on the attached form, Updating Status, indicates the pages to be added, removed or replaced, the number of pages included, and the Newsletter Code. Pages marked with an asterisk should be removed from the publication. The form should be filed at the back of the publication as a permanent record of amended pages.

Each amended page is identified by the Newsletter Code shown above.
Amended pages remain valid unless otherwise noted in a subsequent Newsletter.
Modifications to text, figures or tables are indicated by a vertical bar in the outside margin next to the change.

Summary of Amendments:

This newsletter:

- corrects some typing and/or technical inaccuracies

- incorporates operational refinements to the HDISK command (formerly documented as the FDISK command)

- incorporates keyboard-generated codes for all national keyboards

Ing. C. Olivetti & C., S.p.A. — Direzione Documentazione — Via Jervis 77 10015 IVREA (TO) Italy

Printed in Italy

olivetti

JPDATING STATUS

LEVEL	DATE	UPDATED PAGES	PAGES	CODE
0				
1	15-12-83	Front cover, iii ÷ xi, 2-1, 2-3, 2-4, 2-11 ÷ 2-19, 3-3, 4-8, 5-5, 7-4 ÷ 7-8, 7-51, 8-6, 8-12, 8-39, 11-7, 11-8, A-1, B-1 ÷ B-26	63	3984671 H
2				
3				
4				
5				
6				
7				
8				
9				

Pages marked * must be suppressed

CONTENTS

1. INTRODUCTION

ABOUT THIS CHAPTER

This chapter introduces some of the more commonly used features of MS-DOS, provides some information on disk handling and defines the notation convention used throughout this book.

CONTENTS

WHAT IS MS-DOS?

The Microsoft Disk Operating System (MS-DOS ™) provides the interface between the M20 hardware and both you and the other system software. It is made up of a group of programs that enable you to manipulate information stored on diskette or hard disk.

Through MS-DOS you communicate with the Central Processing Unit (CPU), Video Display Unit (VDU), disk drives, printer, and other peripherals.

MS-DOS is supplied on your MS-DOS diskette.

The major features of MS-DOS are outlined in the sections that follow.

COMMAND LIBRARY

MS-DOS has a command library of over 40 commands that provide you with an environment suitable for handling files of information, developing programs, and running applications.

FILE AND DIRECTORY HANDLING

File handling commands not only allow you to copy and delete files, copy entire diskettes, display the contents of files, etc., but also to group files into directories at your convenience. For example, each user can have his own directory. Moreover, MS-DOS enables you to create directories within a directory, thereby creating a hierarchical directory structure. Refer to Chapter 4 for details.

PROGRAMMING TOOLS

A set of programming tools enable you to write and develop programs. You can edit program files using the Line Editor EDLIN (see Chapter 8), link object files using the LINK utility (see Chapter 9), and debug executable files using the DEBUG utility (see Chapter 10).

INTERNAL AND EXTERNAL COMMANDS

When MS-DOS is initialised some commands are loaded into memory and reside there. Other commands remain on disk. The former are known as internal commands and are the more commonly used commands. The latter are external commands and are loaded into memory and executed when required. After execution they are removed from memory, thereby optimising the use of memory.

BATCH PROCESSING

MS-DOS enables a commonly executed series of commands to be grouped into one file -- a batch file -- that can be executed simply by entering the file name. See Chapter 5 for details.

THE AUTOEXEC.BAT FILE

The AUTOEXEC.BAT file is a special batch file which, if present, is executed automatically at system initialisation. This is useful if your application requires a certain sequence of commands to be executed every time the system is initialised (see Chapter 5).

REAL-TIME CLOCK FUNCTIONS

MS-DOS has two commands that utilise the Real-Time Clock. These are:

- DATE which enables you to set the date
- TIME which enables you to set the time.

Moreover, on initialising the system you will be prompted to enter the date and time. These values are important not only for programs that use time-dependent functions, but also because MS-DOS provides you with information about the time and date of creation or modification of your files.

ADM5 TERMINAL EMULATOR

The ADM5 Terminal Emulator is a program that makes your M20 respond like a Lear Siegler ADM5 terminal. You will need to use this program if you wish to run any of the wide range of application packages that are compatible with that terminal. See Chapter 6 for details.

ANSI TERMINAL EMULATOR

The ANSI terminal emulator enables MS-DOS to recognise a set of ANSI escape sequences. Thereby enhancing console control. See Appendix C for details.

INTERFACE HANDLING

MS-DOS allows you to communicate with compatible devices (plotters, paper tape readers and punches, modems, etc.) via the standard RS-232-C serial interface. You will need to set the protocol for the interface using the MODE command (see Chapter 7) and load the RS-232-C Package using the RS232 command.

DISKS

Information is stored either on 5 1/4 in. floppy disks or, if your system has one, on hard disk. This manual will refer to the former as diskettes and the latter as the hard disk. The term "disk" will be used to mean either diskette or hard disk.

Your M20 will have one of the following combinations of diskette and/or hard disk drive:

- two 160/180 Kbyte diskette drives (drive A on the right, drive B on the left)
- two 320/360 Kbyte diskette drives (drive A on the right, drive B on the left)
- two 640/720 Kbyte diskette drives (drive A on the right, drive B on the left)
- one 320/360 Kbyte diskette drive (drive A on the left) and one hard disk (drive C on the right)
- one 640/720 Kbyte diskette drive (drive A on the left) and one hard disk (drive C on the right)

The drive letters (A, B and C) are the means by which commands can identify a particular drive.

The drive size (320/360 Kbyte, etc.) governs the type of diskette that can be used in that drive.

The types of diskette that can be used are:

- single-sided diskettes (one recording surface only) offering 160/180 Kbytes of storage. Such diskettes can be used in 160/180 Kbyte or 320/360 Kbyte diskette drives but not in 640/720 Kbyte diskette drives

- double-sided diskettes (two recording surfaces) offering 320/360 Kbytes of storage. These diskettes can only be used in 320/360 Kbyte diskette drives, or on 160/180 Kbyte drives, when formatted single-sided

- double-sided diskettes offering 640/720 Kbytes of storage. These diskettes can only be used in 640/720 Kbyte diskette drives

Each diskette can have one of two capacities (for example 320 or 360 Kbytes) depending on the way in which it is formatted. The higher of the two values provides compatibility with IBM DOS version 2.0, while the lesser value enables diskettes to be compatible with IBM DOS version 1.25 or previous Olivetti releases of MS-DOS.

DISKETTE HANDLING

Although diskettes are generally durable, damage to diskettes will be minimised if you take the following precautions:

- never bend diskettes

- do not touch the exposed surface of the diskette

- always keep the diskette in its cardboard envelope when not in use and store it in the diskette carton

- keep dust out of the diskette drives by keeping the drive covers closed when not in use

LABELLING DISKETTES

Every carton of diskettes contains a supply of self-adhesive labels for identifying diskettes. It is good practice to write all relevant details on the label before attaching it to the diskette. But if you do find it necessary to write on the label after sticking it to the diskette, you should avoid using sharp pencils or ball-point pens as these may damage the surface of the diskette. In this case a felt-tipped pen is recommended.

WRITE-PROTECTION

To protect your data from being accidentally overwritten, you can apply write-protection to your diskettes. A sheet of aluminised write-protect labels is provided with every carton of diskettes. To apply write-protection, simply fix an aluminised label over the write-protect notch in the diskette. To remove write-protection, simply remove the label.

NOTATION CONVENTION

The following syntax conventions are used throughout this book.

- Uppercase letters and words must be typed in the actual line exactly as shown.

  Example: In the statement shown as

  ```
  GOSUB line-num
  ```

  GOSUB should be typed as shown

  Uppercase letters and words are used simply as visual aids in this manual. They may be typed in lowercase if desired.

- Lowercase letters and words represent variable information the user is to provide.

  Example: In the statement

  ```
  PURGE filename
  ```

  filename should be replaced by a specific value; for example, myfile

- Hyphens may join lowercase letters or words to form a single variable item.

  Example: In the statement

  ```
  GOSUB line-num
  ```

  line-num is a single variable item that should be replaced by a single specific value; for example, 999

- A blank, a comma, a colon, or a semicolon is used to separate the items in a line

- The symbols listed below are used to define the syntax of a line, but should not be typed in the actual line:

| vertical stroke ("or" sign)

{ } braces

[ ] brackets

... ellipsis

- Items separated by vertical strokes (|) represent alternatives. You should select only one such alternative.

  Example: The representation

  A|B|C

  indicates that either A or B or C should be selected

  Note: This should not be confused with the "pipe" command (|) introduced in Chapter 5

- Braces group related items, such as alternatives.

  Example: The representation

  {A|B|C|}

  indicates that you must choose one of the items enclosed the braces

- Brackets also group related items; however, everything within the brackets is optional and may be omitted.

  Example: The representation

  [A|B|C]

  indicates that you may choose one of the items enclosed within the brackets or that you may omit all of the items

- An ellipsis indicates that the preceding item or group of items may be repeated more than once in succession.

  Example: The representation

  A [,B] ...

  indicates that A can be typed alone or can be followed by

  ,B

  more than once in succession

- Items enclosed within slashes indicate that that you must press a specific key. For example the key whose inscription is S2 is always referred to as /S2/

2. GETTING STARTED

ABOUT THIS CHAPTER

This chapter tells you how to load MS-DOS and how to end your working session. It also describes some of the steps you will take when you use MS-DOS for the first time, and how to perform some of the more common functions.

The last section of this chapter tells you how to prepare the hard disk. If you have a hard disk you should refer to this section first.

For further details of the commands mentioned in this chapter refer to Chapter 7.

CONTENTS

HOW TO LOAD MS-DOS

Switch on your M20 and a set of diagnostics will be run. Following successful execution of these diagnostics the following message will appear

Select Alternate CPU (y/n)? 

Entering "y" initiates the bootstrap from the APB 1086 processor board. Diagnostic tests held on that board are then performed. What happens after these diagnostics depends on the type of M20 you have:

- If you have a dual diskette drive system, you will be prompted to insert your MS-DOS disk in drive A. After doing so, strike any key and MS-DOS will begin to be loaded

- If you have a hard disk system then MS-DOS will be loaded from the diskette if either of the following conditions apply:

  . your MS-DOS diskette was already in the diskette drive when you switched the M20 on. In this case MS-DOS will be loaded automatically from the MS-DOS diskette

  . no bootable file (other than PCOS) exists on the hard disk. In this case you will be prompted to insert your MS-DOS disk in drive A. After striking any key MS-DOS will begin to be loaded

  If there are bootable files on the hard disk, however, then you can alternatively load an operating system from such a file. Refer to the section entitled "Booting MS-DOS from the Hard Disk" at the end of this chapter

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

When the command processor is loaded you will see a display on your screen similar to the following (underline represents the cursor):

```
Diskette number : MSDOS 2.x
MS-DOS version 2.00
Copyright (c) Olivetti 1983

Olivetti M20 CONIO Sept 1983
Copyright (c) Olivetti 1983 All rights reserved
Monochrome system
Command v. 2.02
Current date is Tue 1-01-1980
Enter new date:_
```

You may now enter today's date.

HOW TO ENTER THE DATE AND TIME

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

mm is a one- or two-digit number from 1-12 (representing month)

dd is a one- or two-digit number from 1-31 (representing day of month)

yy or a four-digit number from 1980 to 2099 (representing year)

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

```
6-1-82 or 06/01/82
```

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

If you enter an invalid date or form of date, the system will prompt you again with

```
Enter new date:
```

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

```
Current time is 0:00:19.25
Enter new time:_
```

The current time is displayed in four fields: hours; minutes; seconds; hundredths of seconds.

Enter the new current time in the hh:mm:ss format where:

hh is a one- or two-digit number from 0 to 23 representing hours

mm is a one- or two-digit number from 0 to 59 representing minutes

ss is a one- or two-digit number from 0 to 59 representing seconds

MS-DOS uses this time value to keep track of when you last updated and/or created files on the system. Notice that MS-DOS uses a 24-hour clock; for instance, 1.30 p.m. is written 13.30.

Example

```
Current time is 0.00:19.25
Enter new time: 9:05
```

You should use the colon (:) to separate hours, minutes and seconds. If you enter an invalid number separator, MS-DOS will respond

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

You have now completed the steps for starting MS-DOS.

Remarks

If you do not wish to enter date or time, simply answer "/CR/" to the corresponding prompts.

If you make a mistake while typing, press the /CTRL/ and /C/ keys simultaneously to abort your current entry. You can then reanswer the prompt or enter another command. To correct a line before you press /CR/, use the /S1/ (backspace) key to erase one letter at a time.

HOW TO TERMINATE YOUR WORKING SESSION

You can bring your working session to an end in any one of three ways:

- by switching off the M20 using the ON/OFF switch on the back panel
- by performing a physical reset either by pressing the /COMMAND/ and /RESET/ keys simultaneously, or by pressing the physical reset button with a pencil or ball-point pen. The physical reset button is located on the right-hand side of the main unit near the back. The effect of doing this is to terminate the current working session and to reinitialise the M20 starting with diagnostics followed by the "Select Alternate CPU (Y/N)?" prompt. In other words it has the same effect as switching the M20 off and back on again
- by pressing the /CTRL/ and /RESET/ keys simultaneously. The effect of doing this is to terminate the current working session and reinitialise the M20 but starting from the APB-resident diagnostics. That is, reinitialisation commences as if you had just replied "y" to the "Select Alternate CPU (y/n)?" prompt

HOW TO CHANGE THE DEFAULT DRIVE

After you have answered the new time prompt, a message is displayed that looks like this:

```
A>_
```

The A> is the MS-DOS prompt from the command processor. It tells you that MS-DOS is ready to accept commands. If MS-DOS has been loaded from a drive other than drive A, the command processor prompt will reflect that drive (for example, C>). For dual-diskette systems you will always load MS-DOS from drive A.

The A in the previous prompt represents the default disk drive. This means that MS-DOS will search only the disk in drive A for any file names you may enter and will write files to that disk unless you specify a different drive. You can ask MS-DOS to search the disk in drive B by changing the drive designation or by specifying "B:" in a command. To change the disk drive designation, enter the new drive letter followed by a colon. For example:

```
A>          (MS-DOS prompt)
A>B: /CR/   (you have typed "B:" in response to the prompt)
B>          (system responds with B> and drive B is now
            the default drive)
```

The system prompt B> will appear and MS-DOS will search only the disk in drive B until you specify a different default drive.

HOW TO FORMAT YOUR DISKETTES

You must format all new diskettes before they can be used by MS-DOS.

A blank diskette must be formatted with the FORMAT command. The FORMAT command changes the disk to a format that MS-DOS can use. If the disk is not already blank, formatting it will destroy any data that exists on the disk.

The syntax of the FORMAT command in its simplest form is:

```
FORMAT [drivespec:]
```

where:

drivespec is the drive designation (the drive that contains the disk to be formatted)

Note that the brackets identify optional information. If you do not specify a disk drive (A, B or C), MS-DOS will format the disk in the default drive.

Formatting places the tracks on a disk and creates an empty directory for that disk -- the directory is the means by which MS-DOS keeps track of what is on the disk.

If you enter

```
FORMAT B: /CR/
```

then MS-DOS issues the following message:

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

After you insert the new diskette in drive B and press any key on the keyboard, formatting begins and the system keeps you informed of the progress of the FORMAT command by responding

```
Formatted Track n
```

where n is the number of the last track formatted. When formatting is complete, you will see on your screen a message similar to this:

```
362496 bytes total disk space
362496 bytes available on disk

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

Enter "y" to format another disk using the same option flags that you specified in the command line. Type N to end the FORMAT program.

For disks that have already been used, however, it is not necessary to place the tracks on the disk. If you enter

```
FORMAT B:/C /CR/
```

then the disk is simply initialised. That is, the disk is cleared of all data and an empty directory is placed on it.

If you include /S at the end of the command line you will cause the FORMAT command to copy the system files to the target diskette as well as formatting that diskette. On completion the system will typically respond

```
362496 bytes total disk space
 41984 bytes used by system
320512 bytes available on disk
```

If you include /V at the end of the command line you will be given the option to label the disk. When formatting is complete you will be prompted

```
volume label (11 characters. RETURN for none)?
```

You can enter any valid string up to 11 characters followed by /CR/. For example you could assign the label PROGRAMS to a disk containing programs. You can use such a label to check that you are using the correct disk. The label will be displayed if you issue either a DIR command or a VOL command. The FORMAT command is the only command that can be used to label a disk so you are recommended to do so. However, if you do not wish to assign a label to the disk, simply enter "/CR/".

Formatting is normally 9 blocks per track which produces disks that are compatible with IBM PC-DOS release 2-0. However, if you include /8 at the

end of the command line you will format the disk eight blocks per track - compatible with previous Olivetti releases of MS-DOS. Moreover, if you enter /0/8 at the end of the command line the disk will be compatible with IBM PC-DOS release 1-1.

HOW TO BACK-UP YOUR DISKETTES

It is strongly recommended that you make back-up copies of all your disks. If a disk becomes damaged, or if files are accidentally erased, you will still have all of the information on your back-up disk. In particular, you must make a back-up of your MS-DOS disk. You can back-up diskettes using the MS-DOS DISKCOPY command as described below.

The DISKCOPY command copies the contents of one diskette onto another diskette. You can use this command to duplicate both the MS-DOS disk and a disk that contains your own files. DISKCOPY is the fastest way of copying a disk because it copies the entire disk in one operation, including MS-DOS system files if they exist. However, note that the DISKCOPY command can only be used where the source and target diskettes are the same size.

The format of the DISKCOPY command is

```
DISKCOPY [source-drive:] [target-drive:]
```

"source-drive" is the diskette drive that contains the diskette that you want to copy; "target-drive" is the diskette drive that contains the target diskette. The target diskette must be formatted prior to running DISKCOPY.

For example, if you want to make a copy of your MS-DOS disk which is in drive A, enter

```
DISKCOPY A:,B:  /CR/
```

MS-DOS responds

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

Make sure the MS-DOS disk is in drive A and insert a blank, formatted disk in drive B. Press any character key after you have done this and MS-DOS will begin copying the MS-DOS disk.

After MS-DOS has copied the disk, MS-DOS displays

```
Copying... Copy complete
Copy another (Y/N)?
```

Enter "y" (for Yes) if you wish to copy another disk with DISKCOPY. If you enter "n" (for NO), the default drive prompt is displayed.

Note

If either of the disks that you are using has defective tracks, DISKCOPY will not work. Use the COPY command to back-up your disks in these cases. COPY will skip over defective tracks. But note that if you need to copy the MS-DOS disk using the COPY command you must first format the disk using the /S option.

COPYING DISKETTES - SINGLE DISKETTE DRIVE

If your M20 has only one diskette drive - for instance if you have a hard disk system - you must first insert the system diskette into drive A and enter

```
DISKCOPY /CR/
```

MS-DOS then replies with the message

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

After striking any key MS-DOS responds

```
Insert source diskette into drive A:
Press any key when ready_
```

On doing so MS-DOS begins copying the contents of the diskette to memory. When available memory space is full, MS-DOS prompts

```
Insert target diskette in drive A:
Press any key when ready_
```

On doing so, MS-DOS transfers the data copied to memory onto the target diskette. You are then prompted to reinsert the source diskette.

After performing this sequence a number of times, the entire source diskette will have been copied to the target diskette. MS-DOS then displays

```
Copy complete

Copy another (Y/N)?
```

CAUTION

You must make sure that you insert the correct source and target diskettes as MS-DOS cannot identify the diskette. Moreover, you are strongly recommended to write-protect your source diskette to avoid inadvertently overwriting it.

HOW TO LIST THE FILES ON A DISK

When a disk is formatted a directory is placed on that disk. This directory provides the means by which MS-DOS keeps track of the files on that disk. The DIR command enables you to display the contents of this directory and hence obtain a list of the files held on that disk. For example, if your copy of the MS-DOS disk is in drive A and you enter

```
DIR A: /CR/
```

MS-DOS will respond with a directory listing of all the files in the current directory on your MS-DOS disk. The display will look similar to this:

```
Volume in drive A is REL 2-x
Directory of A:

COMMAND      COM     15480    8-22-83    2:06p
CONFIG       SYS        31    8-22-83    2:06p
CONIO        BIN      7314    8-22-83    2:06p
DEBUG        COM     11764    8-22-83    2:06p
CHKDSK       COM      6330    8-22-83    2:06p
EDLIN        COM      4389    8-22-83    2:06p
PRINT        COM      3808    8-22-83    2:06p
RECOVER      COM      2277    8-22-83    2:06p
SYS          COM       850    8-22-83    2:06p
MORE         COM      4364    8-22-83    2:06p
DISKCOPY     COM      1419    8-22-83    2:06p
BASIC        COM     29184    8-22-83    2:06p
RS232        COM       660    8-22-83    2:06p
HEXDUMP      COM       567    8-22-83    2:06p
FORMAT       COM     12357    8-22-83    2:06p
EXE2BIN      EXE      1649    8-22-83    2:06p
LINK         EXE     42368    8-22-83    2:06p
SORT         EXE      1216    8-22-83    2:06p
FIND         EXE      5796    8-22-83    2:06p
FC           EXE      2553    8-22-83    2:06p
MASM         EXE     77440    8-22-83    2:06p
CREF         EXE     13824    8-22-83    2:06p
SETLANG      EXE      4242    8-22-83    2:06p
MODE         EXE     13568    8-22-83    2:06p
UK           DAT      3072    8-22-83    2:06p
DENMARK      DAT      3072    8-22-83    2:06p
USA          DAT      3072    8-22-83    2:06p
ITALY        DAT      3072    8-22-83    2:06p
FRANCE       DAT      3072    8-22-83    2:06p
GERMANY      DAT      3072    8-22-83    2:06p
SPAIN        DAT      3072    8-22-83    2:06p
SWEDEN       DAT      3072    8-22-83    2:06p
PORTUGAL     DAT      3072    8-22-83    2:06p
NORWAY       DAT      3072    8-22-83    2:06p
SWFRENCH     DAT      3072    8-22-83    2:06p
SWGERMAN     DAT      3072    8-22-82    2:11p
     36 File(s)              21504 bytes free
```

Three MS-DOS system files, IO.SYS, MSDOS.SYS and REL_2.0B are "hidden" files and will not appear when you issue the DIR command. Entering "CHKDSK /V /CR/" will identify the hidden files.

For each entry in the directory the size of the file in bytes is given, along with the date and time at which the file was last modified.

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

```
DIR MYFILE.TXT  /CR/
```

will give you a display of all the directory information for the file MYFILE.TXT.

AUTOMATIC PROGRAM EXECUTION

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

When you start MS-DOS, the command processor searches for a file named AUTOEXEC.BAT on the MS-DOS disk. This file is a program that MS-DOS will run each time MS-DOS is started. Chapter 5 tells you how to create an AUTOEXEC.BAT file.

HOW TO SET THE KEYBOARD LANGUAGE

The MS-DOS disk that was delivered with your M20 is configured for a USA ASCII keyboard. If yours is not a USA ASCII keyboard, however, then you will need to reconfigure your MS-DOS disk to correspond to the appropriate keyboard. Appendix B describes all the supported keyboards. For this you require the SETLANG command.

The format of the SETLANG command is as follows:

```
SETLANG [filespec]
```

Where filespec is the data file containing the required keyboard tables. It can be one of the following:

- DENMARK.DAT
- FRANCE.DAT
- GERMANY.DAT
- ITALY.DAT
- NORWAY.DAT
- PORTUGAL.DAT
- SPAIN.DAT
- SWEDEN.DAT

- UK.DAT
- USA.DAT
- SWFRENCH.DAT (Switzerland French)
- SWGERMAN.DAT (Switzerland German)

Once you have entered the command line, SETLANG displays the following prompt:

```
Ready to modify DEFAULT drive?
```

Now insert your non write-protected copy of the MS-DOS disk in drive A and enter "y". The specified file then changes the contents of the IO.SYS file which is loaded next time the system is initialised, thereby making the selected keyboard language active.

If you do not specify a filespec in the command line, then after replying "y" to the above prompt the following prompt is displayed

```
Get tables from which file?_
```

You must then enter the file name from the above list. For example, if you enter

```
ITALY /CR/
```

then the Italy keyboard becomes active.

HOW TO PREPARE THE HARD DISK

If your M20 has a hard disk you will need to perform the following:

1. Update the bad block list for the hard disk using the PCOS VVERIFY command
2. Format the hard disk using the PCOS VFORMAT command
3. Check the hard disk for bad blocks using the PCOS VVERIFY command. This step is necessary to ensure that there are no bad blocks near the beginning of a partition
4. Partition the hard disk using the PCOS HDISK command to enable operating systems other than PCOS to use the hard disk
5. Logically format the PCOS partition using the PCOS VNEW command

6. Logically format the MS-DOS partition using the MS-DOS FORMAT command

7. Copy the MS-DOS diskette to the hard disk using the MS-DOS COPY command

Each of these steps is described in detail in the sections that follow.

HOW TO UPDATE THE BAD BLOCK LIST

The bad block list can only be updated in the PCOS environment. For this, proceed as follows:

1. Switch on the M20 (or perform a physical reset)

2. Answer "n" to the prompt "Select Alternate CPU? (y/n)"

3. Insert the PCOS system diskette and strike any key. PCOS will then be loaded into memory. On completion the PCOS prompt (0>) will appear

4. Update the bad block list by entering

```
vv %d  /CR/
```

   On completion, the PCOS prompt will return

HOW TO FORMAT THE HARD DISK

The hard disk can only be physically formatted in the PCOS environment.

Format the hard disk by entering

```
vf 10:  /CR/
```

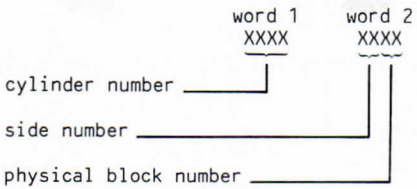
On completion, the PCOS prompt will return. Note that this process also isolates as unusable all blocks listed in the bad block list.

HOW TO CHECK THE BAD BLOCK LIST

The bootstrap loader, system tables, directory, etc. of each operating system will reside in the first 2 cylinders of its partition. It is therefore necessary to check that there are no bad blocks in these areas. If you enter

```
vv %p 10:  /CR/
```

then any bad blocks reported by the VFORMAT command will be displayed in the following format:



where the two words are in hexadecimal.

HOW TO PARTITION THE HARD DISK

1. Insert the diskette containing the HDISK command into drive 0

2. Enter

```
hd  /CR/
```

   and the following menu is displayed:

```
Fixed Disk Setup Program  4.x

Menu of options

  1. Initialize partition map.
  2. Display partition map.
  3. Change partition map entry.
  4. Exit program.
     q or Q to quit (same as Exit).

Enter selection:
```

3. Enter

```
1  /CR/
```

   and the initial partition map is installed on the hard disk, destroying any map that already existed. The M20 responds

```
Disk partitions now initialized
Type [RETURN] when ready to continue:
```

4. Strike /CR/ to return to the menu of options then enter

```
2  /CR/
```

   to display the partition map. The initial partition map looks like this:

```
          Fixed Disk Partition Map

Partition       CPU     Bootable  Start   End     Operating
Number          Type      OS      Cyl     Cyl     System
  1             z8000     NO       0       178     PCOS
  2               -       -        -       -       NONE
  3               -       -        -       -       NONE
  4               -       -        -       -       NONE
  5               -       -        -       -       NONE
  6               -       -        -       -       NONE
  7               -       -        -       -       NONE
  8               -       -        -       -       NONE
  9               -       -        -       -       NONE
  10              -       -        -       -       NONE
  11              -       -        -       -       NONE
  12              -       -        -       -       NONE
  13              -       -        -       -       NONE
  14              -       -        -       -       NONE
  15              -       -        -       -       NONE
  16            z8000     NO      179     179     TEST

   Type [RETURN] when ready to continue:
```

Notice that the hard disk can be split into up to 16 partitions, the first of which is dedicated to PCOS and initially occupies all 179 available cylinders (cylinders 0 to 178). Cylinder 179 is taken up by partition 16 which always contains diagnostics

5. You must now decide how many partitions you require and partition the hard disk accordingly. To use the hard disk under PCOS, MS-DOS and CP/M-86 you will need to create partitions for MS-DOS and CP/M-86 as well as PCOS. First you must reduce the size of the partition allocated to PCOS

6. After striking /CR/ and returning to the menu of options, enter

```
3  /CR/
```

to enable you to change the partition map. The partition map will re-appear and you will be prompted

```
Which entry do you wish to modify (1..15)(0 to exit)?
```

7. Enter

```
1 /CR/
```

and the following prompt will appear

```
'c' for change, 'd' for delete:
```

Entering "c /CR/" enables you to modify the partition, entering "d /CR/" would delete the partition. Note, however, that partition 1 can

never be deleted. Any attempt to do so will generate an appropriate error message

8. Enter

```
c /CR/
```

to enable partition 1 to be modified. You will be prompted

```
Change End Cylinder number from: 178? (y to change)
```

9. Enter

```
y /CR/
```

and you will be prompted

```
Enter ending cyl number (00-178):
```

10. Enter, for example

```
59 /CR/
```

to allocate cylinders 0 to 59 to PCOS. (Even if you do not intend to use PCOS, you must still allow at least one cylinder.) The displayed partition map is then immediately updated and the following prompt is displayed

```
Change bootable status from: NO to YES? (y to change)
```

If the bootable status is NO, then the operating system to which the partition is assigned will not be initialised when you attempt to boot the system from the hard disk. If its status is YES, then it will.

11. Enter

```
y /CR/
```

to change the status to "YES". The updated partition map is redisplayed and you are prompted

```
Which entry do you wish to modify (1..15)(0 to exit)?
```

12. Enter

```
2 /CR/
```

to enable you to modify partition 2. Had the partition already been assigned an operating system (as a result of a previous partition map, or if you had assigned this partition earlier in the current HDISK session) then you would get the prompt "'c' for change, 'd' for delete". In such a situation 'c' would enable you to change the

boundaries or bootable status of partition 2 but not the operating system. 'd' would delete the entry and enable you to re-assign the partition to some other operating system. In this case, however, partition 2 has not yet been assigned an operating system and the following menu appears

```
Operating System Menu:     1. PCOS          2. MS-DOS
                           3. CP/M-86       4. UCSD p System

Enter selection:
```

13. Enter

```
2 /CR/
```

to dedicate partition 2 to MS-DOS. The displayed partition map is immediately updated and you are prompted

```
Enter beginning cyl number (60-178):
```

14. Enter

```
60 /CR/
```

(assuming that there are no bad blocks in the cylinders 60 and 61. If there are, you must redefine the previous partition to include these cylinders, then start this partition from two consecutive cylinders that do not contain bad blocks.) The displayed partition map is immediately updated and you are prompted

```
Enter ending cyl number (0-178):
```

15. Enter, for example

```
119 /CR/
```

to allocate cylinders 60 to 119, inclusive, to partition 2. The displayed partition map is again updated and you are prompted

```
Change bootable status from NO to YES? (y to change)
```

Since a number of operating systems can be initialised using the APB 1086 board, you may specify one of your partitions as the one whose operating system is to be initialised when the system is started from the hard disk. You do this by setting the bootable status of the corresponding partition to YES. Note that more than one partition may be designated as bootable. If you do this, however, when you attempt to initialise the system from the APB 1086 board the M20 will prompt you as to which of the bootable operating systems you require

16. Enter

        y /CR/

    to change the bootable status to "YES", thereby enabling MS-DOS to be initialised when the system is started from the hard disk. The updated partition map is then redisplayed and you are prompted

        Which entry do you wish to modify (1..15)(0 to exit)?

17. You may now repeat steps 12 to 16 to modify partition 3 such that the remaining cylinders (cylinders 120 to 178) are allocated to CP/M-86. Following this operation the partition map will be displayed thus:

        Fixed Disk Partition Map

Partition Number	CPU Type	Bootable OS	Start Cyl	End Cyl	Operating System
1	z8000	YES	0	59	PCOS
2	8086	YES	60	119	MS-DOS
3	8086	YES	120	178	CP/M-86
4	-	-	-	-	NONE
5	-	-	-	-	NONE
6	-	-	-	-	NONE
7	-	-	-	-	NONE
8	-	-	-	-	NONE
9	-	-	-	-	NONE
10	-	-	-	-	NONE
11	-	-	-	-	NONE
12	-	-	-	-	NONE
13	-	-	-	-	NONE
14	-	-	-	-	NONE
15	-	-	-	-	NONE
16	z8000	NO	179	179	TEST

        Which entry do you wish to modify (1..15)(0 to exit)?

18. Enter "0 /CR/" to exit and you will be prompted

        Type [RETURN] when ready to continue:

19. Strike /CR/ to return to the menu of options then enter

        4 /CR/

    to exit the program. You will then be prompted

```
*********************************************************
*  YOU MUST RUN VNEW FOR PCOS TO OPERATE PROPERLY  *
*********************************************************


        Hit any key to return to PCOS
```

20. Strike any key as instructed, and control will return to PCOS

Note: You will only be prompted to run VNEW if the HDISK session you have just completed modified the PCOS partition.

HOW TO LOGICALLY FORMAT THE PCOS PARTITION

To ensure that PCOS recognises the end of its partition it is necessary to logically format the PCOS partition using the VNEW command. You must therefore enter

```
        vn 10:  /CR/
```

HOW TO LOGICALLY FORMAT THE MS-DOS PARTITION

To logically format the MS-DOS partition of the hard disk you must insert your MS-DOS disk in drive A and boot MS-DOS. Having done so, you then need to use the FORMAT command. Moreover, if you intend to transfer your operating system onto the hard disk you must copy the "hidden" system files from the MS-DOS disk to hard disk. To do this you must specify the /S option with the FORMAT command thus:

```
        FORMAT C:/S  /CR/
```

As each cylinder is formatted and verified the message

```
        Verifying Cylinder xx
```

is displayed.

On completion a message similar to the following is displayed

```
        System transferred

        2924544 bytes total disk space
          41216 bytes used by system
        2883328 bytes available on disk
```

HOW TO COPY THE MS-DOS DISK TO THE HARD DISK

After the MS-DOS partition of the hard disk has been formatted using the /S option it is then possible to boot the system from the hard disk. However, you will probably want to transfer the remaining files from the system diskette to the hard disk. To do this you must enter

```
COPY *.* C:  /CR/
```

MS-DOS will display each file name as the file is copied until all the files have been transferred, at which point the MS-DOS prompt will reappear.

Note, however, that the bootstrap process defaults to drive A. Hence, if you have a diskette in drive A then the M20 will attempt to boot the system from that diskette. If you want to boot from the hard disk you must make sure that either there is no diskette in drive A or that the drive A door is open.

You have now completed the steps for preparing your hard disk.

BOOTING MS-DOS WITH A HARD DISK

Once you have MS-DOS on your hard disk you can boot it directly at system start-up. Respond "y" to the "Select Alternate CPU (y/n)?" prompt. If you do not have a system diskette in the diskette drive and you have more than one bootable 8086 operating system on your hard disk a message similar to the following appears:

```
Type letter of DOS to boot:

    A> MS-DOS
    B> CP/M-86
    C> Boot from floppy drive A:
```

You must then enter the appropriate letter followed by /CR/. In this case you would enter

```
B /CR/
```

to boot the MS-DOS operating system installed on the hard disk.

IF you select the last option you are prompted to insert a system disk in drive A and press any key when ready.

3. FUNCTION KEYS AND SPECIAL EDITING KEYS

ABOUT THIS CHAPTER

This chapter describes the keystroke combinations that are required to execute control functions such as carriage return, line feed, break, etc. It also indicates the keystrokes that perform special editing functions. For further details of the latter refer to Chapter 8.

CONTENTS

INTRODUCTION

Other than entering text in the manner of a typewriter, your keyboard is capable, under MS-DOS, of entering other functions by using certain combinations of keystrokes. Such features fall into two groups: function keys and special editing keys.

MS-DOS FUNCTION KEYS

MS-DOS function keys mainly comprise a key combination of an alphabetic key with the /CTRL/ key. They are used to correct typing errors, abort command execution, etc. These functions are described in the following table:

FUNCTION	KEY COMBINATION	MEANING
Carriage Return	/↵/ or /CTRL/ /M/	the command line is terminated and command execution begins. Either of these keystrokes will subsequently be referred to as /CR/
Backspace	/S1/ or /CTRL/ /H/	the last character entered is removed from the command line and the character erased from the VDU
TAB	/S2/ or /CTRL/ /I/	the cursor is moved to the next eight-character TAB position
Line-Feed	/CTRL/ /J/	a line-feed character is entered which moves the cursor to the beginning of the next line but without terminating the command line

Cancel	/CTRL/ /X/	cancels the current command line. A back-slash is displayed at the end of the cancelled line and the cursor is moved to the start of the next line
Break	/CTRL/ /C/	aborts the current command. The command cannot be recommenced at the point of termination. If you wish to reexecute the command you must enter it again
Shift lock	/COMMAND/ /?//	all alphabetic keys take on upper-case values. Shift lock is cleared by pressing /COMMAND/ /?// again Note that the key is /?// on the USA keyboard. It may be different on your keyboard but in any case it is always the bottom right-most key on the alpha-numeric section of the keyboard
No scroll	/CTRL/ /S/	output to the VDU is suspended. You can restart the display by striking any key

Print Output	/CTRL/ /P/	all output that is normally displayed on the VDU is sent to the printer. Press /CTRL/ /P/ again to terminate printer echo

EDITING FUNCTION KEYS

These commands are entered using the /CTRL/ key in conjunction with the top row of the alphanumeric section of the keyboard.

When you press /CR/ to execute a command, a copy of the command is kept in a special buffer called the source line. MS-DOS has a range of commands that you can use to enter a command line by copying some or part of the source line. For example if you wish to copy several files from drive B to drive A you might copy the first of those files by entering

```
COPY B:MYFILE A:  /CR/
```

This command would then be written to the source line. If the next file you want to copy is called YOURFILE, then the character strings "COPY B:" and "FILE A:" can be copied from the source line using editing function keys. The only part you need type in yourself is the string "YOUR".

The most common use of editing function keys, however, is in performing edit operations within a line of text when using the line editor EDLIN. A detailed description is therefore given in Chapter 9, but a brief description of the commands is given in the following table:

COMMAND	KEY COMBINATION	FUNCTION
COPY1	/CTRL/ /1/	copies one character from the source line to the command line

COPYTO	/CTRL/ /2/	copies all characters up to the first occurrence of a given character from the source line to the command line
COPYLINE	/CTRL/ /3/	copies all remaining characters in the source line to the command line
SKIP1	/CTRL/ /0/	deletes one character in the source line
SKIPTO	/CTRL/ /4/	deletes characters in the source line up to a given character
KILL	/CTRL/ /9/	terminates input and terminates the command line
INSERT	/CTRL/ /8/	enters/exits insert mode
NEWTEMP	/CTRL/ /5/	creates a new source line by copying the command line to the source line

4. FILES AND DIRECTORIES

ABOUT THIS CHAPTER

This chapter describes how to manipulate files and directories, how you can access files in directories by means of path names and how to create and delete directories.

For further details of commands mentioned in this chapter refer to Chapter 7.

CONTENTS

HOW MS-DOS KEEPS TRACK OF YOUR FILES

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

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

These two system areas, the directories and the File Allocation Table, enable MS-DOS to recognise and organise the files on your disks. The File Allocation Table is copied onto a new disk when you format it with the MS-DOS FORMAT command and one empty directory is created, known as the root directory.

HOW TO NAME YOUR FILES

A file name can comprise:

- one to eight characters (for legal characters see below). For example NEWFILE
- one to eight characters, followed by a period (.) and a one to three character file name extension. For example NEWFILE.EXE.

A file name may be made up of any of the following characters:

A-Z	0-9	$	&	#
%	'	(	)	-
@	^	{	}	!

Alphabetic characters within the file name can be entered in upper or lower case, but MS-DOS will translate lower case letters into upper case.

If the file you wish to specify is in the current directory, then the file name alone is sufficient to identify that file. If the file resides on the default drive but is not within the current directory, you will need to identify the directory using a path name to access that file (see the section entitled "File names and Paths"). If the file resides on a disk that is in a drive other than the default drive, you will need to identify the drive specifier followed by a colon and the file name; for example "B:NEWFILE.EXE".

WILD CARDS

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

THE ? WILD CARD

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

```
DIR TEST?RUN.EXE  /CR/
```

will list all entries in the current directory on the default drive that have 8 characters, begin with TEST, have any next character, end with the letters RUN, and have a file name extension of EXE. Here are some examples of files that might be listed by the above DIR command:

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

THE * WILD CARD

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

```
DIR TEST*.EXE /CR/
```

will list all entries in the current directory on the default drive with file names that begin with the characters TEST and have an extension of EXE. Here are some examples of files that might be listed by the above DIR command:

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

The wild card designation *.* refers to all files in the current directory. Note that this can be very powerful when used in MS-DOS commands. For example, the command

```
DEL *.*  /CR/
```

deletes all files on the current directory, regardless of file name.

Examples:

To list all files named NEWFILE in the current directory on drive A (regardless of their file name extensions), simply enter

```
DIR A:NEWFILE.*  /CR/
```

To list all files with file name extensions of .TXT in the current directory (regardless of their file names) on the disk in drive B, enter

```
DIR B:????????.TXT  /CR/
```

This command will list all files that have an eight character file name plus an extension of TXT.nsion of .TXT.

RESERVED FILE NAMES

MS-DOS treats some device names specially, and certain three-letter names are reserved for the names of these devices. These three-letter names cannot be used as file names or extensions. You must not name your files any of the following:

AUX	Used when referring to input from or output to an auxiliary device (such as a printer or modem)
CON	Used when referring to keyboard input or to output to the VDU
PRN	Used when referring to the printer
NUL	Used when you do not want to create a particular file, but the command requires an input or output file name

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

HOW TO COPY YOUR FILES

You often need more than one copy of a disk file. The COPY command allows you to copy one or more files to another disk, or another directory on the same disk (see the section entitled "File names and Paths"). You can also give the copy a different name if you specify the new name in the COPY command.

The COPY command can also make copies of files on the same disk. In this case, you must supply MS-DOS with a different file name or you will

overwrite the file. You cannot make a copy of a file on the same disk unless you specify a different file name for the new copy.

The format of the COPY command is:

```
COPY filespec [filespec]
```

For example,

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

will copy the file MYFILE.TXT on the diskette in drive A to a file named MYFILE.TXT on the diskette in drive B.

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

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

You now have two copies of your file on drive A - one named MYFILE.TXT and the other named NEWNAME.TXT.

You can also use the COPY command, along with the wild card feature, to copy groups of files from one disk to another, or to copy all files from one disk to another.

HOW TO PROTECT YOUR FILES

MS-DOS is a powerful and useful tool in processing your personal and business information. As with any information system, inadvertent errors may occur and information may be misused. If you are processing information that cannot be replaced or requires a high level of security, you should take steps to ensure that your data and programs are protected from accidental or unauthorised use, modification, or destruction. Simple measures you can take, such as removing your disks when they are not in use, keeping back-up copies of valuable information, write-protection, and installing your equipment in a secure facility, can help you maintain the integrity of the information in your files.

DIRECTORIES

The names of your files are kept in directories on each disk. Each directory also contains information on the size of the files and the dates that they were created and updated.

When there are multiple users on your M20, or when you are working on several different projects, the number of files in the directory can become large and unwieldy. You may want your own files kept separate from those of a coleague: or, you may want to organise your programs into categories that are convenient to you.

MS-DOS allows you to organise the files on your disks into directories. Directories are a way of dividing your files into convenient groups of files. For example, you may want all your accounting programs in one directory and text files in another. Any one directory can contain any reasonable number of files, and it may also contain other directories (referred to as subdirectories). This method of organising your files is called a hierarchical directory structure.

A hierarchical directory structure can be thought of as a "tree" structure: directories are branches of the tree and files are the leaves, except that the "tree" grows downward; that is, the "root" is at the top. The root is the first level in the directory structure. It is the directory that is automatically created when you format a disk and start putting files in it.

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

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

Hierarchical structures are particularly useful on hard disk systems.

The following figure illustrates a typical hierarchical directory structure.

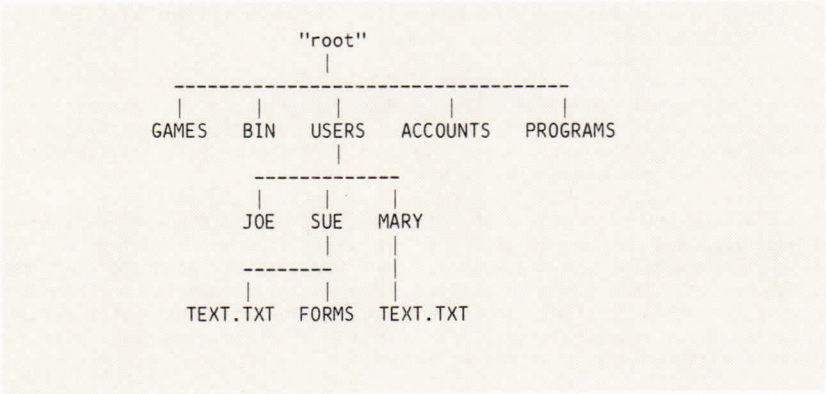


Fig. 4-1 A Sample Hierarchical Directory Structure

The root directory is the first level in the directory structure. It is the directory that you access when you enter the drive specifier. You can create subdirectories from the root by using the MKDIR command. The root directory may also contain files.

In this example, five subdirectories of root have been created. These include:

- a directory of games, named GAMES
- a directory of all external commands, named BIN
- a USER directory containing separate subdirectories for all users of the system
- a directory containing accounting information, named ACCOUNTS
- a directory of programs, named PROGRAMS

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

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

```
DIR \USER\SUE\FORMS  /CR/
```

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

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

```
DIR \USER\MARY  /CR/
```

FILE NAMES AND PATHS

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

PATH NAMES

A path name is a sequence of one or more directory names followed, optionally, by a simple file name, each separated from the previous one by a back-slash (\).

The syntax of path names is:

```
[drivespec: |\ ]directory[\directory]...[\filename]
```

If a path name begins with a backslash, MS-DOS searches for the file beginning at the root (or top) of the tree. Otherwise, MS-DOS begins at the users current directory and searches downward from there. For example, the path name of Sue's TEXT.TXT file is /USER/SUE/TEXT.TXT. That is, if you are in directory JOE and you want to access the file named TEXT.TXT in directory SUE you would use the path name

```
\USER\SUE\TEXT.TXT
```

If you were in directory USER, however, you would access the same file using the path name

```
SUE\TEXT.TXT
```

However, if you are already in directory SUE you simply use the file name

```
TEXT.TXT
```

For example, if you are in directory MARY and you wish to make a copy of Mary's file TEXT.TXT in Sue's directory, but rename the copy COPY.TXT you would enter

```
COPY TEXT.TXT, \USER\SUE\COPY.TXT  /CR/
```

MS-DOS provides special shorthand notations for the current directory and the parent directory (one level up) of the current directory:

. MS-DOS uses this shorthand notation to indicate the name of the current directory in all hierarchical directory listings. MS-DOS automatically creates this entry when a directory is made.

.. The shorthand name of the current directory's parent directory. If you enter

```
DIR .. /CR/
```

then MS-DOS will list the files in the parent directory of your current directory.

If you enter

```
DIR..\.. /CR/
```

then MS-DOS will list the files in the parent's parent directory.

PATHING AND EXTERNAL COMMANDS

External commands reside on disk as program files. They must be read from the disk before they execute.

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

For example, if your current directory is named \BIN\PROG, and all MS-DOS external commands are in \BIN\COMMANDS, you must tell MS-DOS to choose the \BIN\COMMANDS path to find an external command. The command

```
PATH \BIN\COMMANDS /CR/
```

tells MS-DOS to search in your current directory and the \BIN\COMMANDS directory for all commands. You only have to specify this path once to MS-DOS during your working session. MS-DOS will now search in \BIN for the external commands. If you want to know what the current path is, enter the command PATH (without a parameter) and the current value of PATH will be printed.

HOW TO DISPLAY YOUR CURRENT DIRECTORY

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

```
CHDIR /CR/
```

you will see

```
A:\USER\SUE
```

This is the complete path name of your current directory and comprises your current drive designation plus the current directory (USER SUE). If you now want to see what is in the \USER\SUE directory, you can issue the MS-DOS command DIR. The following is an example of the display you might receive for the DIR command for a subdirectory:

```
    Volume in drive A has no label
    Directory of A:\USER\SUE

.           <DIR>          8-09-82      10:09a
..          <DIR>          8-09-82      10:09a
FORMS       <DIR>          8-09-82      10:09a
TEXT   TXT       5243      8-04-82       9:30a
      4 File(s)  8376320 bytes free
```

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

Because files and directories are listed together (see previous display), MS-DOS does not allow you to give a subdirectory the same name as a file in that directory. For example, consider again the path USER\SUE\FORMS. FORMS is a subdirectory of SUE, therefore you cannot create a file named FORMS in directory SUE.

HOW TO CREATE A DIRECTORY

To create a subdirectory in your current directory, use the MKDIR (Make Directory) command. For example, if you are in directory SUE and you want to create a new directory named NEWDIR within your current directory, simply enter

```
MKDIR NEWDIR /CR/
```

After this command has been executed by MS-DOS, a new directory will exist in your tree structure under your current directory.

You can also create directories anywhere in the tree structure by specifying MKDIR and then a path name. For example, if your current directory is SUE and you want to create a directory named SPECS in directory MARY you would enter

```
MKDIR  \USER\MARY\SPECS   /CR/
```

MS-DOS will automatically create the . and .. entries in the new directory.

To create files in the new directory, use the MS-DOS line editor, EDLIN. For details refer to Chapter 8.

HOW TO CHANGE YOUR CURRENT DIRECTORY

To change your current directory to another directory simply issue the CHDIR (Change Directory) command and supply a path name. For example

```
CHDIR  USER  /CR/
```

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

HOW TO REMOVE A DIRECTORY

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

```
RMDIR NEWDIR  /CR/
```

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

```
RMDIR \BIN\USER\JOE  /CR/
```

If the directory is not empty an error is reported. To remove all the files in a directory (except for the . and .. entries), enter DEL and then the path name of the directory. For example, to delete all files in the BIN USER SUE directory, enter

```
DEL \BIN\USER\SUE  /CR/
```

and MS-DOS will display

```
ARE YOU SURE?  (Y/N)
```

Enter "y" and the files will be deleted. You cannot delete the . and ..

entries (except by deleting the directory).

HOW TO CHECK THE VALIDITY OF YOUR FILES

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

To check the disk in drive A enter

```
CHKDSK A: /CR/
```

MS-DOS will display a status report and any errors that it has found. Moreover, if you include the flag "/V" in the command line, thus

```
CHKDSK A: /V /CR/
```

then the CHKDSK command displays messages as it runs and also shows the hidden files. You should run the CHKDSK command occasionally for each disk to ensure the integrity of your files.

5. ENTERING AND USING MS-DOS COMMANDS

ABOUT THIS CHAPTER

This chapter defines the syntax for a command, explains how commands can be grouped into batch files, and how the output from a command can be redirected to some device other than the VDU. It also describes the concept of "piping", whereby the output from one command becomes the input to another.

For further details of commands mentioned in this chapter refer to Chapter 7.

CONTENTS

COMMAND SYNTAX

The general command format is defined as follows:

```
keyword [parameter]...
```

Where

SYNTAX ELEMENT	MEANING
keyword	a one to eight character mnemonic that specifies the command to be executed. It must exclude any file name extension
parameter	a parameter to the command defining the command action. The number of parameters depends on the command executed

PARAMETERS

Parameters are user-selected strings of alphabetic characters and of integers which can be optional. They are recognised by their position in the command line. The types of parameter are described in the following table:

PARAMETER TYPE	MEANING
filename	either a one to eight character string or a one to eight character string followed by a period (.) and a three character extension. A file name must be made up from the following characters: A-Z 0-9 $ & # % ' ( ) - @ ^ { } ! For example NEWFILE NEWFILE.TXT
filespec	{[drivespec:]filename|pathname} a file specifier can be a file name with or without a drive specifier, or it can be a path name that terminates in a file name. For example NEWFILE.TXT B:NEWFILE.TXT \USER\MARY\NEWFILE.TXT
pathname	[drivespec:|\]directory[\directory]...[\filename] a pathname may comprise one or more one-to-eight character directory names separated by back-slashes, and may terminate with either a file name or a directory name. The back-slash before the first directory name is optional - it must be entered if the pathname is to start from the root directory. For example PROGRAMS \USER\MARY\NEWFILE.TXT

drivespec	A|B|C drives A and B are the diskette drives, drive C is the hard disk drive
switch	an option which controls the way the command executes. It takes the form of a single letter preceded by a back-slash. For example /P
argument	provides more information to an MS-DOS command. For example ON or OFF

Nil Parameters

Some parameters are optional and take default values if they are not specified in the command line -- that is, no parameter is specified thereafter -- then the parameters may be omitted. For example, if you enter

```
MODE COM:11  /CR/
```

which takes four parameters, then the last three parameters take default values.

INTERNAL AND EXTERNAL COMMANDS

There are two types of MS-DOS command:

- Internal commands
- External commands

Internal commands are the simplest, most commonly used commands. You cannot see these commands when you do a directory listing on your MS-DOS disk; they are part of the command processor. Therefore they reside in memory whenever MS-DOS is loaded. When you enter these commands, they execute immediately. This class of command comprises:

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

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

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

When you enter an external command, do not include its file name extension. External commands include:

CHKDSK	MORE
DISKCOPY	PRINT
FIND	RECOVER
FORMAT	SORT
EXE2BIN	SYS
MODE	HEXDUMP
RS232	SETLANG
FC	

INFORMATION COMMON TO ALL MS-DOS COMMANDS

The following information applies to all MS-DOS commands:

- Commands are usually followed by one or more parameters

- Commands and parameters may be entered in upper case or lower case, or a combination of both. MS-DOS will convert all lower case letters to upper case

- Commands and parameters must be separated by delimiters. Space is usually used; for example:

  DEL MYFILE.OLD NEWFILE.TXT /CR/

  You can also use the comma (,), semicolon (;), the equal sign (=), or the /S2/ key as delimiters in MS-DOS commands.

  For brevity, this manual will use a space as the delimiter

- When you are instructed to "Press any key", you can press any alphabetic (A-Z) or numeric (0-9) key

- You must include the file name extension when referring to a file that already has a file name extension

- You can abort commands that perform input/output by pressing /CTRL/ /C/

- Commands take effect only after you have pressed /CR/

- Wild cards (global file name characters) and device names (for example, PRN or CON) are not allowed in the names of any commands

- When commands produce a large amount of output on the screen, the display will automatically scroll to the next screen. You can press /CTRL/ /S/ to suspend the display. Press any key to resume the display on the screen

- MS-DOS editing and function keys can be used when entering commands. Refer to Chapter 3 for a description of these keys

- The prompt from the command processor is the default drive designation plus a greater-than sign: for example, A>

- Disk drives will be referred to as source drive and target drive. A source drive is the drive you will be transferring information from. A target drive is the drive you will be transferring information to

BATCH PROCESSING

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

HOW TO CREATE AND EXECUTE A BATCH FILE

You can create a batch file by using the Line Editor (EDLIN) or by using the COPY command.

The MS-DOS command library contains a sub-set of batch processing commands. Among the more commonly used are REM and PAUSE. REM permits you to include remarks and comments in your batch files without these remarks being executed as commands. PAUSE prompts you with an optional message and permits you to either continue or abort the batch process at a given point.

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

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

You could create this file as follows:

STEP	IF you enter...	THEN...
1	COPY CON: NEWDISK.BAT /CR/	the file "NEWDISK" is created and awaits input from the keyboard

2	REM This is a file to check new disks /CR/	"REM This is a file to check new disks" is entered on the first line of the file
3	REM It is named NEWDISK.BAT /CR/	"REM It is named NEWDISK .BAT" is entered on the second line of the file
4	PAUSE Insert new disk in drive B /CR/	"PAUSE Insert new disk in drive B" is entered on the third line of the file
5	FORMAT B: /CR/	"FORMAT B:" is entered on the fourth line of the file
6	CHKDSK B: /CR/	"CHKDSK B:" is entered on the fifth line of the file
7	/CTRL/ /Z/ /CR/ or /CTRL/ /6/	the file creation is complete and the message "1 file(s) copied" appears on the screen

To execute this batch file, simply enter the file name without the extension

```
NEWDISK /CR/
```

The result is the same as if each of the lines in the batch file was entered at the terminal as individual commands. That is, the first three commands are executed successively and the following messages are displayed on the screen

```
This is a file to check new disks
It is named NEWDISK.BAT
Insert new disk in drive B
Strike a key when ready . . . _
```

After striking a key, the disk in drive B is formatted and checked. You will then be asked if you wish to format another disk. Following a negative reply, the diskette will be checked.

Remarks

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

2. Only the file name should be entered to execute the batch file. Do not enter the file name extension

3. The commands in the file named filename.BAT are executed

4. If you press /CTRL/ /C/ while in batch mode, this prompt appears:

   terminate batch job (Y/N)?

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

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

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

6. The last command in a batch file may be the name of another batch file. This allows you to call one batch file from another when the first is finished

THE AUTOEXEC.BAT FILE

An AUTOEXEC.BAT file is a batch file that allows you to automatically execute programs when you start MS-DOS. Automatic Program Execution is useful when you want to run a specific package (for example, Microsoft Multiplan) under MS-DOS, and when you want MS-DOS to execute a batch program automatically each time you start the system.

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

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

If MS-DOS does not find an AUTOEXEC.BAT file when you first load the MS-DOS disk, then the date and time prompts will be issued.

The AUTOEXEC.BAT file is created in exactly the same way as any batch file. It must, however, reside on the MS-DOS disk.

Example

If your AUTOEXEC.BAT file contains the following:

```
1. DATE
2. TIME
3. BASIC
```

then on initialising your system the date and time prompts will be maintained, and the system will automatically enter BASIC.

Remark

Remember that if you use an AUTOEXEC.BAT file, MS-DOS will not prompt you for a current date and time unless you include the DATE and TIME commands in the AUTOEXEC.BAT file. It is strongly recommended that you include these two commands in your AUTOEXEC.BAT file, since MS-DOS uses this information to keep your directory current.

BATCH FILES WITH REPLACEABLE PARAMETERS

You may require that commands within a batch file have replaceable parameters. For example, if your batch file contains a COPY command, you may wish to supply a different parameter to the COPY command each time you run the batch file. You can do this by specifying dummy (replaceable) parameters to the commands within the batch file. These parameters, named %0 to %9, can be replaced by values supplied when the batch file executes. For example, you may have created the following batch file named "MYFILE.BAT":

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

To execute this file you must enter the file name (without extension), followed by the replacement values for %1 and %2. For example, if you enter

```
MYFILE A:PROG1 B:PROG2  /CR/
```

then

- %0 is replaced by "MYFILE"
- %1 is replaced by "A:PROG1"
- %2 is replaced by "B:PROG2"

The effect is to execute the following sequence:

```
COPY A:PROG1.MAC B:PROG2.MAC

TYPE B:PROG2.PRN

TYPE MYFILE.BAT
```

Remarks

1. Up to 10 dummy parameters (%0-%9) can be specified in this way. Refer to the SHIFT command if you wish to specify more than 10 parameters.

2. If you use the percent sign as part of a file name within a batch file, you must enter it twice. For example, to specify the file ABC%.EXE, you must enter it as ABC%%.EXE in the batch file.

INPUT AND OUTPUT

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

REDIRECTING YOUR OUTPUT

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

```
DIR  /CR/
```

displays a directory listing of the current directory on the VDU. The same command can send this output to a file named MYFILES instead of the VDU by designating the output file on the command line

```
DIR >MYFILES  /CR/
```

If the file MYFILES does not already exist, MS-DOS creates it and stores your directory listing in it. If MYFILES already exists, MS-DOS

overwrites what is in the file with the new data.

Two greater-than signs (>>) can be used to tell MS-DOS to append the output of the command (such as a directory listing) to the end of a specified file. For example, the command

```
DIR >>MYFILES  /CR/
```

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

REDIRECTING YOUR INPUT

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

```
SORT <NAMES >LIST1  /CR/
```

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

FILTERS

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

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

FIND searches for a constant string of text in a file

MORE takes standard VDU output and displays it, one screen at a time

SORT sorts text

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

COMMAND PIPING

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

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

```
DIR | SORT /CR/
```

will sort your directory into alphabetical order. The vertical bar causes all output generated by the left side of the bar to be sent to the right side of the bar for processing.

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

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

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

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

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

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

```
DIR | SORT | MORE /CR/
```

will sort your directory, show it to you one screen at a time, and put -MORE- at the bottom of your screen when there is more output to be seen.

6. ADM5 TERMINAL EMULATOR

ABOUT THIS CHAPTER

This chapter describes the ADM5 Terminal Emulator.

For further details of the commands mentioned in this chapter refer to Chapter 7.

CONTENTS

INTRODUCTION

The ADM5 Terminal Emulator is a program that enables the M20 to emulate most of the functions of a Lear Siegler ADM5 terminal, thereby enhancing the range of application packages that you can run on the M20.

When you operate your M20 in ADM5 terminal emulation mode any MS-DOS command can be entered exactly as if you were operating in the standard M20 MS-DOS environment. Moreover, the control keys and special edit keys of MS-DOS also function in exactly the same way. However, in addition the ADM5 Terminal Emulator also supports the following functions of the Lear Siegler ADM5 terminal:

- bell
- backspace
- line feed
- up line
- forward space
- return
- clear screen
- home cursor
- clear to end-of-line
- clear to end-of-screen
- toggle inverse video
- initiate load cursor

How to perform each of these functions is described in the section entitled "The ADM5 Terminal Emulator Functions".

HOW TO INVOKE THE ADM5 TERMINAL EMULATOR

The ADM5 Terminal Emulator is invoked using the MODE command. You can either enter the command directly from the keyboard, or if you wish ADM5 emulation mode to be entered on initialisation you can include the MODE command in the AUTOEXEC.BAT file.

METHOD 1: ENTERING THE MODE COMMAND AT THE KEYBOARD

If, following the MS-DOS prompt, you enter

```
MODE CON:ADM5  /CR/
```

then the M20 loads the ADM5 program into memory and executes it. The ADM5 Terminal Emulator is then active and the MS-DOS prompt (A>) is displayed.

METHOD 2: USING THE AUTOEXEC.BAT FILE

You can avoid having to enter the ADM5 command every time you want to run the ADM5 Terminal Emulator by creating the AUTOEXEC.BAT file and including the MODE command in it. By doing so, the ADM5 Terminal Emulator will be invoked automatically whenever the system is booted. In order not to lose the time and date dependent functions available in the standard M20 MS-DOS environment, you should include the TIME and DATE commands in the AUTOEXEC.BAT file, otherwise you will not be prompted for the time and date when the system is booted.

Before creating the AUTOEXEC.BAT file you must first remove the write-protect label from the copy you made of the MS-DOS disk, and insert the disk in drive A. Then create the AUTOEXEC.BAT file containing the following:

```
1. DATE
2. TIME
3. MODE CON:ADM5
```

The procedure for creating an AUTOEXEC.BAT file is described in detail in Chapter 5.

Having created the AUTOEXEC.BAT file on your system disk you must then affix the write-protect label. On booting from this disk in future the AUTOEXEC.BAT file will be executed automatically and will prompt you for the date and time, after which it will load and execute the ADM5 Terminal Emulator.

THE ADM5 TERMINAL EMULATOR FUNCTIONS

The additional functions available within the ADM5 Terminal Emulator are all invoked by generating an ASCII code or sequence of ASCII codes from a program. For example, you could do this using the PRINT statement from a BASIC program.

The following table explains which ASCII mnemonic is required to execute the corresponding function. The hexadecimal code is shown in parentheses under the corresponding ASCII mnemonic.

IF your program generates ASCII mnemonic (hex code)...	THEN the function...	is executed which...
BEL (07)	BELL	generates an audible 'beep'
BS (08)	BACKSPACE	deletes the character prior to the cursor and moves the cursor one character position to the left
LF (0A)	LINEFEED	moves the cursor to the start of the next line but without terminating the current entry
VT (0B)	UPLINE	moves the cursor one line up the screen. No action occurs if the cursor is already on the top line
FF (0C)	FORWARD SPACE	moves the cursor one position to the right. If the cursor was on the last position in the line, it is advanced to the start of the next line

CR (0D)	RETURN	causes a carriage return
SUB (1A)	CLEAR SCREEN	clears the screen of all text
RS (1E)	HOME CURSOR	moves the cursor to the start of the top line
ESC "T" (1B 54)	CLEAR TO END-OF-LINE	causes all text under and to the right of the cursor on the current line to be deleted and replaced with spaces
ESC "Y" (1B 59)	CLEAR TO END-OF-SCREEN	causes all text from the current cursor position to the end of the screen to be deleted and replaced with spaces
ESC "G" (1B 47)	TOGGLE INVERSE VIDEO	sets reverse video, or restores standard video
ESC "=" (1B 3D) /Ycharacter position/ /Xcharacter position/	INITIATE LOAD CURSOR	moves the cursor to the line number specified by the Y character position (range 1 to 24), and to the column position specified by the X character position (range 1 to 79)

7. COMMANDS

ABOUT THIS CHAPTER

This chapter gives the syntax and use of all the MS-DOS commands. The commands are presented in alphabetical order.

CONTENTS

INTRODUCTION

The table below lists the commands included in this chapter and gives a brief description of each. It also tells you which commands are internal (I) and which are external (E).

Command	Type	Function
BREAK	I	turns the /CTRL/ /C/ switch on and off for the compiler and assemble operations
CHDIR	I	changes the current directory
CHKDSK	E	scans a disk directory and checks for consistency
CLS	I	clears the screen
COPY	I	copies the file(s) specified
CTTY	I	changes the input and output device
DATE	I	displays and sets the date
DEL	I	deletes the file(s) specified
DIR	I	lists the requested directory entries
DISKCOPY	E	copies complete disks
ECHO	I	turns the batch file echo feature on/off
ERASE	I	is the same as DEL
EXE2BIN	E	converts executable files to binary format
EXIT	I	exits MS-DOS and returns to an application program or batch file
FIND	E	searches for a string within a file or files
FOR	I	batch command extension
FORMAT	E	formats a disk to receive MS-DOS files
GOTO	I	batch command extension
HEXDUMP	E	produces a hexadecimal dump of a file

IF	I	batch command extension
MKDIR	I	creates a directory
MODE	E	sets the display mode and serial transmission and printer environments
MORE	E	displays output one screen at a time
PATH	I	sets a command search path
PAUSE	I	pauses for input in a batch file
PRINT	E	background print feature
PROMPT	I	alters the command prompt
RECOVER	I	recovers bad files
REM	I	displays a comment in a batch file
REN	I	renames a file
RMDIR	I	removes a directory
RS232	E	invokes the RS-232-C software
SET	I	sets one string value to another
SETLANG	E	sets the keyboard language
SHIFT	I	increases the number of replaceable parameters in batch processing
SORT	E	sorts data alphabetically, forwards or backwards
SYS	E	transfers MS-DOS system files from one disk to another
TIME	I	displays and sets the time
TYPE	I	displays the contents of the specified file
VER	I	prints MS-DOS version number
VERIFY	I	verifies writes to disk
VOL	I	prints volume identification label

COMMANDS

BREAK

Turns off and on the abort facility provided by /CTRL/ /C/.

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

Type

Internal

Characteristics

MS-DOS normally checks /CTRL/ /C/ only for input/output operations involving VDU, keyboard and printer. /CTRL/ /C/ will not normally abort the assemble or compile operations. Specifying BREAK ON makes /CTRL/ /C/ effective for the assemble and compile operations also.

If you specify BREAK without a parameter the current BREAK setting is displayed.

CHDIR

Changes the current directory; displays the name of the current directory.

```
CHDIR [pathname]
```

Where

SYNTAX ELEMENT	MEANING
pathname	a path name that terminates with the directory you wish to enter

Type

Internal

Characteristics

Use CHDIR (or CD) with a pathname to change to any valid directory.

To change to the parent directory of your current directory enter:

```
CHDIR ..
```

Used without a parameter CHDIR displays the name of your current directory. This feature is useful if you have forgotten the name of the directory in which you are working.

To change to the root directory enter

```
CHDIR   /CR/
```

Examples

IF you enter...	THEN...
CHDIR \BIN\USER\JOE\FORMS /CR/	MS-DOS puts you in the directory \BIN\USER\JOE\FORMS
CHDIR .. /CR/	MS-DOS puts you in the parent of the current working directory (in the above example, \BIN\USER\JOE)

CHKDSK

Analyses the contents of the disk in the specified or default drive.

CHKDSK [drivespec] [filespec] [/F] [/V]

Where

SYNTAX ELEMENT	MEANING
drivespec	the drive containing the disk to be checked
filespec	a file or group of files specified using wild card characters. The number of contiguous areas occupied by the file(s) will be displayed

Type

External

Characteristics

CHKDSK produces a status report on the contents of the disk. If you specify the /F switch, CHKDSK attempts to correct any errors it finds. If you specify the /V switch, CHKDSK displays messages while it is running and will also list the hidden files.

Example

The following is an example of a CHKDSK status report:

```
322560  bytes total disk space
  2048  bytes in 2 directories
171008  bytes in 22 user files
149504  bytes available on disk

212922  bytes total memory
185408  bytes free
```

Remarks

To redirect CHKDSK output, use the > redirection symbol followed by the name of a file. Error messages will be sent to the file specified. Do not use the /F switch if you redirect CHKDSK output.

The following errors conditions will be corrected automatically if you specify the /F switch:

- Invalid drive specification
- Invalid parameter
- Invalid subdirectory entry
- Cannot CHDIR to <filename>
  Tree past this point not processed
- First cluster number is invalid entry truncated
- Allocation error, size adjusted
- Has invalid cluster, file truncated
- Disk error reading FAT
- Disk error writing FAT
- <filename> contains non-contiguous blocks
- All specified file(s) are contiguous

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

ERROR MESSAGE	MEANING
Incorrect DOS version	you cannot run CHKDSK on versions of MS-DOS that are not 2.0 or higher

Insufficient memory Processing cannot continue	there is not enough memory in your machine to process CHKDSK for this disk. You must obtain more memory to run CHKDSK
Errors found, F parameter not specified Corrections will not be written to disk	you must specify the /F switch if you want the errors corrected by CHKDSK
Invalid current directory Processing cannot continue	there is an error in the current directory. Restart the system and rerun CHKDSK
Cannot CHDIR to root Processing cannot continue	the root directory of the disk is inaccessible. Try to run RECOVER on on the disk
<filename> is cross linked on cluster	make a copy of the file you want to keep, and then delete both files that are cross linked
X lost clusters found in y chains Convert lost chains to files (Y/N)?	clusters have been found that do not correspond to a file. If you respond Y to the prompt, CHKDSK will create a directory entry and a file for you to resolve the problem (files created by CHKDSK are named FILEnnnnnnn) CHKDSK will then display: X bytes disk space freed If you respond N to this prompt, and have not specified the /F switch, CHKDSK frees the clusters and displays: X bytes disk space would be freed

Probable non-DOS disk Continue (Y/N)?	the disk you are using is a non-MS-DOS disk. You must indicate whether or not you want CHKDSK to continue processing
Insufficient room in root directory Erase files in root and repeat CHKDSK	CHKDSK cannot process until you delete files in the root directory
Unrecoverable error in directory Convert directory to file (Y/N)?	a directory has been found to be unusable. Enter Y to convert the directory into a file. You can then fix the directory yourself or delete it

CLS

Clears the screen.

```
CLS
```

Type

Internal

Characteristics

CLS causes MS-DOS to send the ANSI escape sequence ESC[2J to the console driver. This escape sequence clears the screen. This command will not work with the ADM5 Terminal Emulator.

COPY

Copies a file or concatenation of files to a file in the specified (or default) drive.

```
COPY [/A|/B] filespec1 [/A|/B] [+filespec2 [/A|/B]...] [pathname [/A|/B]]
```

Where

SYNTAX ELEMENT	MEANING
filespec1	the file specification of the file to be copied (excluding the drive only if the file is in the default drive)
filespec2	the file specification of any file to be concatenated with the file in filespec1 (excluding the drive only if the file is in the default drive)
pathname	the path of the destination directory or file (excluding the drive to place the file in the default drive, or the file name and extension to retain those given in filespec1)

Type

Internal

Characteristics

COPY either makes a copy of a file or concatenates two or more files. The resulting destination file may have the same name as the first source file (so long as it is in another directory), or a different name.

To copy a file and retain its name, use the following syntax:

```
COPY filespec [pathname]
```

where "pathname" gives the drive and/or directory in which the copy is to be placed.

Omit the pathname parameter to place the copy in the current directory of the default drive. Specify simply a drive to place the copy in the current directory of that drive. If you specify a directory path, make sure that you end the path with the name of an existing directory.

You cannot place a copy of a file in the directory that contains the original.

To copy a file and give the copy a different name, use the following syntax:

```
COPY filespec pathname
```

where "pathname" gives the name of the file in which the copy is to be placed.

Omit drive and directory names from "pathname" to place the file in the current directory of the default drive. Omit just directory names from "pathname" to place the file in the current directory of the specified file. Because you are renaming the file, the source and destination directories may be the same.

If you identify in "pathname" a file that does not exist, MS-DOS creates it for you. If the file already exists, its previous contents are destroyed in the copy operation.

To concatenate two or more files, use the following syntax:

```
COPY filespec1+filespec2... [pathname]
```

Combine the names of each file to be concatenated with the plus sign (+). COPY appends each file in turn to the previous one.

The result of the concatenation is a single file. This file is given the drive, directory, name and extension specified in "pathname".

If "pathname" is absent, the resulting file replaces the file specified in "filespec1" (that is, the first file to be concatenated). If the drive only is supplied, a new file with the name of the file specified in "filespec1" is copied into the drive's current directory. If drive and directory names only are supplied, a new file with the name given in "filespec1" is copied into the specified subdirectory. If the name and extension only are supplied, the file is placed in the current directory of the default drive.

Examples

IF you enter...	THEN...
COPY B:SECRETS /CR/	the file SECRETS is copied from drive B to the default drive
COPY *.* B: /CR/	all files on the default drive that have extensions are copied onto drive B
COPY \DOCS\SECRETS B:INFO /CR/	the file SECRETS is copied to the file INFO in the current directory of drive B (or the subdirectory INFO in the current directory if it exists)
COPY \DOCS\SECRETS \DOCS\NEWS /CR/	the file SECRETS is copied to the file NEWS (or the subdirectory NEWS if it exists)
COPY INFO+NEWS+VIEWS ALL.LST /CR/	the files NEWS and VIEWS are appended to the file INFO, and the resulting concatenation is copied to the file ALL.LST

COPY ALL.LST+NEWS /CR/	the file NEWS is appended to the file ALL.LST, resulting in an enlarged ALL.LST
COPY *.LST COMBIN.PRN /CR/	all files with the extension .LST are concatenated, and the result is placed in COMBIN.PRN
COPY *.LST+*.REF *.PRN /CR/	each file with the extension .REF is appended to the corresponding file with the extension .LST, and each resulting file is given the extension .PRN
COPY *.LST+*.REF COMBIN.PRN /CR/	all files with the extension .LST then all files with the extension .REF, are concatenated, the result being placed in COMBIN.PRN
COPY ALL.LST+*.LST /CR/	all files with the extension .LST, with the exception of ALL.LST, are appended to ALL.LST
COPY PROG.COM/B+ERRS.TXT/A /CR/	the text file ERRS.TST is appended to the binary file PROG.COM, leaving the result in the binary file PROG.COM

Remarks

As the examples show, you can use the wildcards * and ?, to do both simple copy operations (where no destination file is specified) and file concatenation.

When wildcards are present in two or more source parameters combined with the concatenation symbol (+), the result is a single destination file where one has been specified. If the destination parameter itself contains a wildcard, a series of concatenated files is produced.

Note that the penultimate example shows the correct way of concatenating files where one of the source files is also the destination. Had the

command COPY *.LST ALL.LST been entered, the previous contents of ALL.LST would have been destroyed and the following message would have appeared:

```
Content of destination lost before copy
```

You may use the following reserved (illegal) file names in place of standard file name parameters:

```
CON
AUX
PRN
NUL
```

For example, to copy text that you are about to enter from the keyboard into a file, use the following format:

```
COPY CON filename
```

You can then enter text directly into the file you have named, terminating your input by pressing /CTRL/ /Z/ followed by /CR/.

The parameters /A and /B shown in the syntax of the COPY command apply when you wish to regulate the amount of data to be copied. The following table shows the effect of /A and /B on files to which they are attached and to all remaining files in the command until another parameter is found.

IF you enter...	WITH...	THEN...
/A	a source file	the file is regarded as a text (ASCII) file, and its contents copied up to but excluding the first end-of-file character (/CTRL/ /Z/)
/A	a destination file	the file is regarded as a text (ASCII) file, and an end-of-file character (/CTRL/ /Z/) is added as its last character

/B	a source file	the file is regarded as a binary file, and the entire file including any number of end-of-file characters is copied
/B	a destination file	the file is regarded as a binary file, and no end-of-file character (/CTRL/ /Z/) is added

The default value is /A when you are using COPY to concatenate files, /B when you are not.

The final example COPY PROG.COM/B+ERRS.TXT/A shows the use of /A and /B to append a file of error messages to a program file. The default for concatenated files being /A, the /B attached to the program file is obligatory. The /A must then be attached to the text file in order to cancel the previous /B parameter.

CTTY

Lets you change the device from which you issue commands.

CTTY devicename

Where

SYNTAX ELEMENT	MEANING
devicename	the reserved name of the device you wish to use (AUX, CON, PRN or NUL)

Type

Internal

Examples

IF you enter...	THEN...
CTTY AUX /CR/	command I/O is moved to the device attached to the AUX port
CTTY CON /CR/	command I/O is returned to the console

DATE

Displays and lets you change the date known to the system.

DATE [mm-dd-yy]

Where

SYNTAX ELEMENT	MEANING
mm	the one- or two-digit identifier of the month (1-12)

dd	the one- or two-digit identifier of the day (1-31)
yy	the two- or four-digit identifier of the year (80-99 or 1980-2099)

Type

Internal

Characteristics

Separate month, day and year entries by either hyphens (-) or slashes (/).

If you leave out the parameter, DATE prompts you as in the following example:

```
Current date is Mon 3-28-83
Enter new date:
```

Enter the date in the correct format, without entering a value for the day of the week. To accept the current date simply press /CR/.

Example

IF you enter...	THEN...
DATE 1-2-84 /CR/	2nd January 1984 is established as the current date

Remarks

If the values or separators you enter are not valid, DATE displays the message:

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

DATE then waits for you to enter a valid date.

DEL

Deletes the specified file(s).

DEL pathname
or
ERASE pathname

Where

SYNTAX ELEMENT	MEANING
pathname	the specification of the file(s) to be deleted (excluding the drive to delete file(s) in the default drive, excluding the directory path to delete file(s) in the current directory)

Type

Internal

Characteristics

You may use the wildcards * and ? in the file name and extension.

To delete all the files in a directory enter the wildcards "*.*". Alternatively, enter a path name ending in a directory. In these cases, MS-DOS prompts you to confirm your choice as follows:

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

Press "Y" to carry out the deletion, or "N" to return to the MS-DOS prompt.

To delete all files without a file extension, enter "*.".

Example

IF you enter...	THEN...
DEL B:*.TMP /CR/	all files in drive B with the extension .TMP are deleted

Remarks

To delete an actual directory (as opposed to all the files that a directory contains) you must use the RMDIR command.

ERASE is synonymous with DEL.

DIR

Lists files in the current or a specified directory.

DIR [pathname] [/P] [/W]

Where

SYNTAX ELEMENT	MEANING
pathname	the specification of the file or directory to be listed (excluding the drive to list a file or directory in the default drive, excluding the directory path to list a file in the current directory

Type

Internal

Characteristics

If you do not specify a file name and extension, all files in the specified (or current) directory are listed.

You may use the wildcards * and ? in the file name and extension. If you omit either the name or the extension, the wildcard * is assumed in its place.

To list a file that does not have an extension but exclude any that do, enter the file name followed by a period (.). To list all files that do not have an extension simply enter a period.

DIR produces a display in which the size in bytes and date and time of last modification appear alongside the file name(s) and extension(s).

The /P parameter halts the directory display as soon as the screen is filled. To resume the listing, press any key.

The /W parameter displays file names and extensions, without supplementary information, five to a line across the screen.

Examples

IF you enter...	THEN...
DIR B: /CR/ (or DIR B:*.*)	all files in the drive B root directory are listed
DIR .COM /CR/ (or DIR *.COM)	all files in the default drive root directory the extension COM are listed
DIR AUTHORS /CR/ (or DIR AUTHORS.*)	all files in the default drive with the name AUTHORS are listed

DISKCOPY

Copies the contents of a disk in one drive onto a disk in another drive.

DISKCOPY [sourcedrive] [targetdrive]

Where

SYNTAX ELEMENT	MEANING
sourcedrive	the drive that contains the disk to be copied

targetdrive	the drive that contains the disk to receive the copy

Type

External

Characteristics

The disk in the destination drive must be formatted prior to using DISKCOPY.

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

After copying, DISKCOPY prompts:

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

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

Remarks

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

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

For example:

```
COPY A:*.* B: /CR/
```

copies all files from the disk in drive A to the disk in drive B

- DISKCOPY automatically determines the number of sides to copy, based on the source drive and disk

- If the source and destination diskettes have not been formatted with the same number of sectors per track then the message is displayed

```
Source and destination disks are not of the
same format. Cannot do copy
```

- If disk errors are encountered during a DISKCOPY, MS-DOS displays:

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

- You can use the CHKDSK command to determine the capacity (and hence the format) of the source diskette

ECHO

Turns the batch file echo feature off and on.

```
ECHO [ON|OFF]
```

Type

Internal

Characteristics

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

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

ERASE

Deletes the specified file(s)

See the "DEL" command.

EXE2BIN

Converts files from .EXE format to binary format.

```
EXE2BIN filespec1 [filespec2]
```

Where

SYNTAX ELEMENT	MEANING
filespec1	the file specification of the file to be converted (excluding the drive if it is in the default drive, excluding a directory path if it is in the current directory, excluding the extension to accept the default extension of .EXE)
filespec2	the file specification of the output file (excluding the drive and/or file name to accept the drive and/or file name in filespec1, excluding a directory path to accept the current directory, excluding the extension to accept the default extension of .BIN

Type

External

Characteristics

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

Two kinds of conversion are possible, depending on the initial CS:IP in the .EXE file:

- 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 by the assembler statement ORG at 100H; the first 100H bytes of the file are deleted. Providing no segment fixups are present in the file, the output file can be renamed with a .COM extension and used just like any other .COM program available on the MS-DOS diskette

- If CS:IP is not specified, a pure binary conversion is assumed. If the program contains instructions requiring segment relocation, the following prompt appears:

  Fix-up needed - base segment (hex):

  Enter a hexadecimal value for the segment in which the program is to be loaded. In this case the converted output file cannot be used as a relocatable .COM file but only when loaded into the specified segment by a user application

Remarks

If the input file does not meet one of the two sets of requirements given above, the following message appears:

```
File cannot be converted
```

Note that to produce a standard .COM file with the MACRO-86 assembler you must both use the statement ORG to set the location pointer at 100H and specify the first location as the start address (with the END statement). For example:

```
ORG 100H
START:
.
.
.
END START
```

EXIT

Exits from the command processor COMMAND.COM and returns to an application program running previously.

```
EXIT
```

Type

Internal

Characteristics

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

```
A>COMMAND
```

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

FIND

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

```
FIND [/V][/C][/N]"string" [filespec]
```

Where

SYNTAX ELEMENT	MEANING
"string"	a string of valid characters contained in quotes (")
filespec	the specification of the file(s) to be searched

Type

External

Characteristics

FIND displays all lines that contain a specified string from the file or files listed in the command line.

Put in double quotes any string that itself contains quotes.

You cannot use wild cards in your file specifications.

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

Switches for FIND are:

SWITCH	MEANING
/V	causes FIND to display all lines not containing the specified string
/C	causes FIND to print only the count of lines that contain a match in each of the files
/N	causes each line to be preceded by its relative line number in the file

Example

IF you enter...	THEN...
FIND "COLOR" BOOK1 BOOK2 /CR/	FIND displays each line in BOOK1 and BOOK2 (in that order) that contains the string COLOR
DIR B: |FIND/V "DAT" /CR/	FIND displays each file name on the disk in drive B that does not contain the string DAT Note: when using piping in this manner your system disk must not be write-protected

Remarks

If you use more than one switch with the FIND command, you will get the results summarised in the following table:

COMBINATION	RESULT
all three switches OR /V and /C	the lines not containing the string are displayed
/C and /N	the count of lines that contain the string is displayed
/V and /N	the lines not containing the string are displayed, together with their line numbers

If you use the FIND command on BASIC text files, the text must have been saved in ASCII format.

FOR

Passes multiple parameters to a command.

Interactive entry: - **FOR %char IN (item...) DO command % char**

Batch entry: - **FOR %%char IN (item...) DO command %% char**

Where

SYNTAX ELEMENT	MEANING
char	any single character other than the digits 0-9
item	a parameter valid for the command required (separated from another such item by a space)
command	the command you wish to invoke

Type

Internal

Characteristics

Use the %char variable in a command line you enter interactively. Use the %%char variable in a command line within a batch file. The %char or %%char variable is assigned the value of each item listed in turn. The command specified is activated for each resulting parameter.

You may include the wildcards * and ? in an item. In this case only one item is processed and any others you enter are ignored.

Remember to separate each item with a space, and to surround the complete item list by brackets.

Examples

IF you enter...	THEN...
FOR %%f IN (*.ASM) DO MASM %%f /CR/	all .ASM files are submitted to the assembler

FORMAT

Formats to MS-DOS standards the disk in the specified or default drive.

```
FORMAT [d:] [/1] [/S] [/0] [/V] [/8] [/C]
```

Where

SYNTAX ELEMENT	MEANING
d:	the name of the drive that contains the disk. If the drive is not specified, the default drive is assumed

Type

External

Characteristics

You must run FORMAT on any new disk you wish to use with MS-DOS. Any information already on the disk is destroyed.

FORMAT places at the beginning of the disk a bootstrap loader, directory and file allocation table. It also checks for any faulty sectors on the disk.

Switch options available with the FORMAT command have the following effect:

SWITCH	MEANING
/1	formats a diskette single-sided. You would use this facility for preparing 180 Kbyte diskettes on a 360 Kbyte drive (or 160 Kbyte on a 320 Kbyte drive if the /8 option is also specified)
/S	copies the system files to the disk being formatted. These are the IO.SYS MSDOS.SYS REL2_0B As hidden files these will not show in a directory listing
/0	can only be used in conjunction with the /8 option to produce disks of compatible format to IBM PC-DOS release 1.1
/V	allows you to enter a volume label. The FORMAT command issues a prompt that enables you to enter a unique volume label of up to 11 characters. This label will appear in subsequent directory listings
/8	formats diskettes 8 blocks per track instead of the default 9 blocks per track. Diskettes formatted in this manner are compatible with previous releases of MS-DOS
/C	formats disks logically rather than physically. That is, it deletes all data from the disk, but does not reformat the tracks. This option should be used on all previously used disks and Olivetti-supplied diskettes

Example

IF you enter...	THEN...
FORMAT B: /S /CR/	the disk in drive B: is formatted and operating system files are copied onto it

Remarks

FORMAT prompts you with a message such as:

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

When you have struck a key to continue MS-DOS formats the disk track-by-track. When it has finished you will receive a message such as:

```
Formatted Track 39  ... Formatting complete

   362496  bytes total disk space
   362496  bytes available on disk

Format another (Y/N)?
```

GOTO

Causes MS-DOS to jump to a specified position in a batch file.

GOTO label

Where

SYNTAX ELEMENT	MEANING
label	a sequence of characters placed in a batch file one line above the next command to be activated

Type

Internal

Characteristics

To define a label in a batch file, precede a sequence of characters by a colon (:). Batch processing then ignores the line until it encounters the GOTO command with the label as parameter. It then jumps to the line below the one that contains the label.

Do not enter the colon when using the label as a parameter of GOTO.

Example

IF you execute a .BAT file containing	THEN...
:foo /CR/ REM looping... /CR/ GOTO foo /CR/	an infinite number of "REM looping..." messages are produced

HEXDUMP

Displays the contents of a file, byte by byte, in hexadecimal.

```
HEXDUMP filespec
```

Where

SYNTAX ELEMENT	MEANING
filespec	the file whose contents are to be displayed

Type

External.

Characteristics

Each line of the display shows 16 bytes of information. At the left-hand end of the line the address of the first byte in the line is given. The hexadecimal value of each of the next 16 bytes then follows -- two characters per byte. The right-hand columns show the ASCII equivalent (if any) of the bytes displayed in that line.

Example

IF you enter...	THEN...
HEXDUMP b:myfile /CR/	the content of the file named "myfile" on the disk in drive B are displayed in hexadecimal

HEXDUMP produces a display similar to the following:

```
Dumping File: B:MYFILE

0000: 1b 5c 61 62 63 64 65 66 67 68 69 6a 6b 6c 6d 6e  ".\abcdefghijklmn"
0010: 6f 70 71 72 73 74 75 77 77 78 79 7a 30 31 32 33  "opqrstuvwxyz0123"
0020: 34 35 36 37 38 39 2d 5e 40 5b 3b 3a 5d 2c 2e 2f  "456789-^@[;:],./"
0030: 00 7c 41 42 43 44 45 46 47 48 49 4a 4b 4c 4d 4e  ".|ABCDEFGHIJKLMN"
0040: 4f 50 51 52 53 54 55 56 57 58 59 5a 5f 21 22 23  "OPQRSTUVWXYZ_!"#"
0050: 24 25 26 27 28 29 3d 7e 60 7b 2b 2a 7d 3c 3e 3f  "$%&'()=~`{+*}<>?"
0060: 00 7f 01 02 03 04 05 06 07 08 09 0a 0b 0c 0d 0e  "................"
0070: 0f 10 11 12 13 14 15 16 17 18 19 1a e0 e1 e2 e3  "................"

HEXDUMP complete
```

IF

Causes conditional execution of a command in batch processing.

IF [NOT] condition command

Where

SYNTAX ELEMENT	MEANING
condition	one of the three valid conditions listed below
command	the command you wish to conditionally execute

Type

Internal

Characteristics

The specified command is only executed if the condition is true. If it is false the command is ignored. Valid conditions are as follows:

CONDITION	MEANING
EXIST filename	the command is executed only if the specified file exists
string1==string2	the command is executed only if the two strings are identical after parameter substitution
ERRORLEVEL number	the command is executed only if the previous program executed had an exit code of the specified number, or higher

Examples

IF you enter...	THEN
IF NOT EXIST \TMP\FOO ECHO no such file /CR/	if the file \TMP\FOO does not exist the message "no such file" appears
IF NOT ERRORLEVEL 3 LINK $1,,; /CR/	if the error level is under three the linker runs, taking as object module the first parameter of the batch file invocation

MKDIR

Creates a new directory.

MKDIR pathname

Where

SYNTAX ELEMENT	MEANING
pathname	the path of the directory you want to create

Type

Internal

Characteristics

Use MKDIR to create or add to a hierarchical directory structure on the disk in the default or specified drive.

You may enter either MKDIR or MD in invoke this command.

Example

IF you enter...	THEN...
MKDIR \USER /CR/	the subdirectory USER is created beneath the root directory in the default drive

MODE

This command enables you to:

- set the protocol for the RS-232-C port
- set the VDU display mode
- set the end of line character for the printer

MODE COM:

Sets the protocol for the RS-232-C port.

```
MODE COM1:baud[,parity[,databits[,stopbits]]]
```

Where

SYNTAX ELEMENT	MEANING
baud	the baud rate. This must be one of the following: 110, 150, 300, 600, 1200, 2400, 4800 or 9600. Only the first two digits need be specified
parity	one of: - E (even) - O (odd) - N (none) The default is N (none)
databits	the number of data bits. This must be either 7 or 8. The default is 7
stopbits	either 1 or 2. If the baud rate is specified as 110, then the default is 2, otherwise the default is 1

Characteristics

The baud rate must be specified. All other parameters are optional and will take default values if omitted.

Examples

IF you enter...	THEN...
MODE COM1:11 0 8 1 /CR/	the baud rate is set to 110, odd parity is specified, and the data bits and stop bits are specified as 8 and 1, respectively
MODE COM1:96 /CR/	the baud rate is set to 9600. All other parameters take default values

MODE CON:

Sets the VDU display mode.

MODE CON:[console [lines [type]]]

Where

SYNTAX ELEMENT	MEANING
console	one of: - ADM5 - the ADM5 Terminal Emulator is invoked. See Chapter 6 for details - ANSI - ANSI escape sequences are enabled. See Appendix C - RAW - neither ADM5 nor ANSI escape features are enabled The default value is ANSI
lines	the number of lines per screen. This must be either 24 or 25. The default is 25
type	one of: - C (colour VDU) - M (black and white VDU) The default is black and white

Example

IF you enter...	THEN...
MODE CON:ADM5 24 C /CR/	the ADM5 Terminal Emulator is invoked with a 24 line display for a colour VDU

MODE PRT:

Sets the end of line character for the printer.

```
MODE PRT:character [interface]
```

Where

SYNTAX ELEMENT	MEANING
character	one of the following: CR - carriage return is specified as the end of line character LF - line-feed is specified as the end of line character CRLF - carriage-return/line-feed is specified as the end of line character
interface	one of the following: P - for a parallel interface S - for a serial interface The default is P

Characteristics

The following table shows the proper EOL character for each Olivetti supported printer:

Printer	EOL Character
PR-1450	LF
PR-2400	LF
PR-1471	LF
PR-1481	CRLF
PR-2300	LF
PR-430	CRLF
PR-2835	LF
PR-320	CRLF
ET-121	CRLF
ET-231	CRLF

Example

IF you enter...	THEN...
MODE PRT:CRLF /CR/	carriage-return/line-feed is recognised as the end of line character

MORE

Sends output to the terminal one screen at a time.

```
MORE
```

Type

External

Characteristics

Use MORE as a filter added at the end of a command line. Whatever screen output the command you have selected produces is displayed one screen at a time, instead of scrolling through its entire contents. At the bottom of each screenful, the prompt ---MORE--- is displayed. Press /CR/ to display the next screenful.

Example

IF you enter...	THEN...
TYPE MYFILE|MORE /CR/	the file MYFILE on the default drive is displayed one screen at a time. Note: The diskette cannot be write-protected

PATH

Establishes a command search path.

```
PATH [pathname[;pathname]...]
```

Where

SYNTAX ELEMENT	MEANING
pathname	the path of a directory you wish MS-DOS to search

Type

Internal

Characteristics

PATH tells MS-DOS where to look for external commands after it has searched your working directory.

You can specify a single directory path or a number of paths each separated by a semicolon (;). The default is no search path; in this case MS-DOS only searches your working directory.

If you enter "PATH" with no parameter, MS-DOS displays the current search path. If you enter "PATH;" any previously established path is cancelled and only your working directory is searched.

You only need to set the search path once in any terminal session.

Example

IF you enter...	THEN...
PATH \BIN\USER;\BIN\DEV /CR/	MS-DOS searches first your working directory, then \BIN\USER, and finally \BIN\DEV

PAUSE

Suspends execution of the batch file in which it is contained.

PAUSE [comment]

Where

SYNTAX ELEMENT	MEANING
comment	a string of up to 121 characters long

Type

Internal

Characteristics

When PAUSE is encountered during the execution of a batch file, any comment you have entered shows on the screen followed by this prompt:

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

At this point the batch file is suspended, allowing you to change disks or perform any other necessary action.

To resume batch execution press any key with the exception of /CTRL/ /C/.

Press /CTRL/ /C/ to cancel processing of the batch file. The following prompt appears:

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

Press "Y" to cancel the batch operation and return to the MS-DOS prompt. Press "N" to return to the previous prompt.

Example

IF you enter...	THEN...
PAUSE have you updated your boss? /CR/	the batch job is suspended and the appropriate reminder is displayed

PRINT

Queues text files on a line printer while you are using other MS-DOS commands.

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

Where

SYNTAX ELEMENT	MEANING
filespec	the file specification of a file to be printed

Type

External

Characteristics

If you enter other commands during printing, printing is suspended until the command is completed.

When you run PRINT for the first time in a terminal session, you are prompted as follows:

```
Name of list device [PRN:]
```

Type the name of a valid line printer, or simply press /CR/ to accept the default line printer.

You cannot specify a path name as a PRINT parameter. You can, however, use lists of files and wild cards. The following switches are possible with this command:

SWITCH	MEANING
/T	TERMINATE: this switch deletes all files in the print queue (those waiting to be printed). A message to this effect will be printed
/C	CANCEL: This switch turns on cancel mode. The preceding filespec and all following filespecs will be suspended in the print queue until you enter a /P switch

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

Print with no parameters displays the contents of the print queue on your screen without affecting the queue.

Examples

IF you enter...	THEN...
PRINT /T /CR/	the print queue is emptied
PRINT /T *.ASM /CR/	the print queue is emptied then all .ASM files on the default drive are queued to the printer
PRINT TEMP1/C TEMP2 TEMP3 /CR/	the three files indicated are removed from the print queue
PRINT TEMP1/C TEMP2/P /CR/	TEMP1 is removed from the queue, whereas TEMP2 is added

PROMPT

Changes the MS-DOS prompt.

PROMPT [prompt]

Where

SYNTAX ELEMENT	MEANING
prompt	the text of the screen prompt you wish to use

Type

Internal

Characteristics

If no text is entered, the prompt will be set to the default prompt, which is the default drive designation plus the > symbol. You can set the prompt to a special prompt, such as the current time, by using the characters indicated below.

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

SPECIAL CHARACTER	MEANING
$	the '$' character
t	the current time
d	the current date
p	the current directory of the default drive
v	the version number
n	the default drive
g	the '>' character
l	the '<' character
b	the '|' character

_	a CR LF sequence
s	a space (leading only)
h	a backspace
e	ASCII escape

Example

IF you enter...	THEN...
PROMPT ng /CR/	the normal MS-DOS prompt (drive letter plus > symbol) is established

RECOVER

Recovers a file or an entire disk containing faulty sectors.

RECOVER filespec|d:

Where

SYNTAX ELEMENT	MEANING
filespec	the file specification of a file that contains a faulty sector

d:	the name of a drive that contains a disk with a faulty sector

Type

External

Characteristics

Specify a file name to recover a particular file. RECOVER reads the file sector by sector and marks any faulty sectors it finds. MS-DOS will no longer allocate data to such sectors.

Specify a drive name to recover a complete disk. RECOVER reads the contents of the disk sector by sector and marks any faulty sectors it finds. If there is not enough space in your root directory for information on all the files on the disk, a message prompts you to rerun RECOVER to complete the recovery procedure.

When you run RECOVER on a disk any directory tree that may be present is destroyed. Files are placed under the root directory and renamed FILE0001.REC, FILE0002.REC and so on.

You can neither run RECOVER on a directory nor use a list of files or wild cards.

REM

Displays a remark during the execution of a batch file.

```
REM [remark]
```

Where

SYNTAX ELEMENT	MEANING
remark	a string of up to 123 bytes long

Type

Internal

Characteristics

A remark inserted in a batch file shows on the screen as soon as it is encountered during batch execution.

REN

Renames the specified file.

REN[AME] filespec filename

Where

SYNTAX ELEMENT	MEANING
filespec	the file specification of the file to be renamed (excluding the drive only if the file is in the default drive)

filename	the new name including any extension you wish to give the file

Type

Internal

Characteristics

REN changes the name and extension of the file specified in the first parameter to those given in the second parameter. The drive and directory do not change; therefore any drive or directory path included in the second parameter is ignored.

You may use the wildcards * and ? in either parameter. If wild cards appear in the second parameter, the corresponding characters in the first parameter remain unchanged.

Examples

IF you enter...	THEN...
REN B:PRESENT PAST /CR/	the file PRESENT in the current directory in drive B is renamed PAST
RENAME *.JON *.?A? /CR/	any file in the current directory in the default drive with the extension JON has its extension changed to JAN

Remarks

You may use either REN or RENAME to call this command.

An attempt to give a file a name already in the file directory results in the following message:

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

RMDIR

Deletes a directory from a hierarchical directory structure.

```
RMDIR pathname
```

Where

SYNTAX ELEMENT	MEANING
pathname	the path of the directory you wish to delete

Type

Internal .

Characteristics

RMDIR deletes a directory that is empty apart from the . and .. shorthand symbols. If the directory contains subdirectories or files, these must first be removed by means of the RMDIR or DEL command respectively.

You may either enter RMDIR or RD to invoke this command.

Example

IF you enter...	THEN...
RMDIR \BIN\USER\JOE /CR/	the specified empty directory is deleted

RS232

Invokes the RS-232-C package.

```
RS232
```

Characteristics

This command loads the RS-232-C software.

Before invoking this command you must set up the RS-232-C protocol using the MODE command.

This command is only useful for communication with a device (such as an Olivetti serial printer) that requires the use of the XON/XOFF handshake.

Example

IF you enter...	THEN...
RS232 /CR/	the RS-232-C package is loaded into memory

SET

Assigns a substitute string value to a string used in an application program or batch file.

```
SET [key=value]
```

Where

SYNTAX ELEMENT	MEANING
key	the key string you wish to replace
value	the string you wish to assign to the key string

Type

Internal

Characteristics

Use SET to assign a value to a standard parameter included in an application program. The value remains operative during a terminal session until another SET command is issued. Within the application program, you must issue the SET command with no parameters. This tells the application what assignment commands have been issued prior to program invocation.

You can also use the SET command with batch files. Instead of passing string values to a batch file by means of positional parameters in the command line assigned to numeric keys in the file text, you can use SET to assign string values to string keys. Within the batch file the form of the key must be as follows:

```
%string%
```

That is, the string must be preceded and followed by a percentage sign (whereas numeric keys are only preceded by a percentage sign).

Outside an application program, use of the SET command with no parameters causes all the current SET assignments to be displayed on the screen.

Example

IF you enter...	THEN...
SET TTY=VT52 /CR/	the TTY value is set to VT52, wherever it appears in your application program

SETLANG

Sets the keyboard language.

```
SETLANG [filespec]
```

Where

SYNTAX ELEMENT	MEANING
filespec	the file containing the keyboard data to be configured (excluding, if you wish, the .DAT extension)

Characteristics

The files containing the font data are all held on the system diskette and can be any one of

- DENMARK.DAT
- FRANCE.DAT
- GERMANY.DAT

- ITALY.DAT
- NORWAY.DAT
- PORTUGAL.DAT
- SPAIN.DAT
- SWEDEN.DAT
- UK.DAT
- USA.DAT
- SWFRENCH.DAT (Switzerland French)
- SWGERMAN.DAT (Switzerland German)

Once you have entered the command line MS-DOS displays the following prompt:

```
Ready to modify DEFAULT drive (Y/N)?
```

You must then insert a non write-protected copy of the system diskette in drive A and enter "y". The corresponding keyboard language becomes active at the next and all subsequent initialisations.

If you do not specify a filespec in the command line, then after replying 'y' to the above prompt the following prompt is displayed:

```
Get tables from which file?
```

You must then enter the name from the above list.

Examples

IF you enter...	THEN...
SETLANG FRANCE /CR/	the France keyboard is configured

SETLANG /CR/	the SETLANG command is invoked and the following prompt is displayed: Get tables from which file?
ITALY /CR/	the Italy keyboard is configured

SHIFT

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

SHIFT

Type

Internal

Characteristics

Usually, command files are limited to handling 10 parameters, %0 through %9. To allow access to more than ten parameters, use SHIFT to alter the numbering of your command line parameters. For example:

```
if          %0 = "foo"
            %1 = "bar"
            %2 = "name"
            %3...%9 are empty
```

then a SHIFT will result in the following:

```
            %0 = "bar"
            %1 = "name"
            %2...%9 are empty
```

If you have entered more than 10 parameters on the command line, those that appear after the 10th (%9) will be shifted one at a time into %9 by successive shifts. You can then refer to these parameters in your batch file, along with any other parameter that retains an association with a numeric variable.

SORT

Reads input and sorts it into an order.

```
SORT [/R] [/+number]
```

Where

SYNTAX ELEMENT	MEANING
number	the column number from which you wish to begin your sort

Type

External

Characteristics

SORT takes the terminal screen as its input and output unless you specify otherwise with the redirection symbols > and <.

Sort takes place in alphabetical order (from A to Z) unless you specify the /R switch, in which case a reverse sort (from Z to A) is done. It starts with the first column of input unless you specify otherwise using the /+number switch.

Examples

IF you enter...	THEN...
SORT /R <UNSORT.TXT>SORT.TXT /CR/	the file UNSORT.TXT is sorted in reverse order and the result placed in the file SORT.TXT
DIR|SORT /+14 /CR/	the directory listing produced by the DIR command is sorted starting with the fourteenth column (the column that contains the file size)

SYS

Updates the specified disk with the system files IO.SYS, MSDOS.SYS and KB_FONT.SYS.

```
SYS d:
```

Where

SYNTAX ELEMENT	MEANING
d:	the drive that contains the target disk

Type

External

Characteristics

The target disk must either be completely blank or have been formatted with the /S parameter to contain previous versions of the system files. If this is not the case the following message appears:

```
No room for system on destination disk
```

If IO.SYS, MSDOS.SYS and DEL2_0B are on the destination disk, they must take up the same amount of space on the disk as the new system will need. This means that you cannot transfer system files from an MS-DOS 2.0 disk to an MS-DOS 1.1 disk. If you try, the following message appears:

```
Incompatible system size
```

You must reformat the MS-DOS 1.1 disk with the MS-DOS FORMAT command before the SYS command will work.

When the SYS command has been successfully carried out the following message appears:

```
System transferred
```

Remarks

The file COMMAND.COM is not transferred.

IO.SYS and MSDOS.SYS and KB_FONT.SYS are hidden files and will not appear in any directory searches.

TIME

Displays and lets you change the time known to the system.

```
TIME [hh[:mm[:ss[.xx]]]]
```

Where

SYNTAX ELEMENT	MEANING
hh	the one- or two-digit identifier of the hour (0-24)
mm	the one- or two-digit identifier of the minute (0-59)
ss	the one- or two-digit identifier of the second (0-59)
xx	the one- or two-digit identifier of the hundredth second (0-99)

Type

Internal

Characteristics

If you enter one or more value(s) and then press /CR/, TIME sets the remaining value(s) to zero. For example, if you enter "8" the time is set to 8:00:00.00

If you leave out the complete parameter, TIME prompts you as in the following example:

```
Current time is 16:36:00.00
Enter new time:
```

You can then enter a new time in the correct format. To accept the current time simply press /CR/.

Example

IF you enter...	THEN...
TIME 8:30 /CR/	half past eight in the morning is established as the current time

TYPE

Displays the contents of the specified file on the video screen.

TYPE filespec

Where

SYNTAX ELEMENT	MEANING
filespec	the file specification of the file to be displayed

Type

Internal

Characteristics

Use this command to examine a file without modifying it. (Use DIR to find the name of a file and EDLIN to alter the contents of a file.)

The complete contents of the file, including any nonalphabetic and non-numeric characters, appear on the screen. As such, the file may appear unreadable.

Tab characters are expanded on the screen to tab stops every eighth column.

VER

Displays the MS-DOS version number.

```
VER
```

Type

Internal

Characteristics

This command displays on your screen the version number of the MS-DOS system you are using.

VERIFY

Turns the verify switch on or off.

```
VERIFY [ON|OFF]
```

Type

Internal

Characteristics

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

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

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

VOL

Displays the volume label of the disk in the specified or default drive.

```
VOL[UME] [d:]
```

Where

SYNTAX ELEMENT	MEANING
d:	the drive that contains the disk to be checked

Type

Internal

Characteristics

If the disk does not have a volume label, VOL displays the following message:

```
Volume in drive x has no label
```

You may enter either VOL or VOLUME to invoke this command.

8. LINE EDITOR (EDLIN)

ABOUT THIS CHAPTER

This chapter tells you how to use the Line Editor EDLIN.

CONTENTS

INTRODUCTION

The Line Editor (EDLIN) can be used to edit files that contain lines of text, where each line is a maximum of 255 characters, the last of which must be a carriage return. The files can contain legible text or source programs.

Within EDLIN, two types of command may be used:

- commands that enable you to perform editing operations on specified lines, a range of lines or an entire file in order to

  . list, edit, delete and insert lines of text

  . search for a specified text string

  . search for and replace a specified text string

  . create, edit and save new files

  . edit an existing file, save the modified file and keep a back-up of the original file

  These commands are termed "inter-line" commands.

- commands that enable you to perform editing operations within a line of text. These are termed "intra-line" commands and utilise the source line facility as described in Chapter 3.

  The control keys described in Chapter 3 can also be used within EDLIN.

In the disk files, the lines of text are not numbered. But when a file is displayed, lines are numbered dynamically. When you create or edit a file, line numbers begin at 1 and are incremented by one through to the end of the file. If you insert new lines between existing lines, all line numbers following the inserted text are automatically incremented by the number of lines inserted. When lines are deleted, all line numbers following the deleted text are decremented automatically by the number of lines deleted. Consequently, lines are always numbered consecutively, starting from 1, through to the last line in the file.

HOW TO INVOKE THE EDLIN PROGRAM

EDLIN

The line editor (EDLIN) is invoked as follows:

```
EDLIN filespec
```

Where

SYNTAX ELEMENT	MEANING
filespec	the file specifier of the file to be edited This must include the drive specifier unless you want to default to the system disk in which case you need to remove any write-protection

Characteristics

When you invoke EDLIN, the back-up file to the file you specify (if it exists) is erased to ensure there is sufficient room on disk for the output file.

If the file specified exists and is smaller than 75% of the available memory, then the entire file is loaded into memory and the following is displayed

```
End of input file
*
```

You can then edit the file. Note that the EDLIN prompt is an asterisk (*), and the cursor is an underline (_).

If the file specified exists and is larger than 75% of the available memory, then only the first part of the file is loaded, until 75% of the available memory is full. The EDLIN prompt (*) and cursor (_) will then appear but not the "End of input file" message. You can then edit that

part of the file loaded into memory. To access unloaded lines you must use the Write Lines and Append Lines commands described later in this chapter.

If the specified file does not exist on the drive then a new file is created with the specified name. But note that the drive you wish the output file to be written to must be specified when you invoke EDLIN, otherwise the output file will be written to the default drive. The following message is displayed

```
New file
*
```

You can then begin to create the file.

There are two edit commands that can be used to terminate the edit session:

- End Edit, which terminates EDLIN, renames the input file filename.BAK and writes the edited file in memory to the output file which is given the same name as the input file. See the "E (END EDIT)" command later in this chapter

- Quit Edit, which terminates EDLIN without creating a back-up or an output file. The input file remains unchanged. See the "Q (QUIT EDIT)" command later in this chapter

Note that a file with the extension .BAK cannot be edited. Any attempt to do so will generate the message

```
.BAK file--rename file
```

You must rename the file using the RENAME command (See Chapter 4), then invoke EDLIN on the renamed file.

If, when attempting to create a new file, the following message appears

```
No room in directory for file
```

then either

- the file directory is full, or

- you have specified an illegal disk drive or file name

The latter can be checked by examining the command line. (If the command line is no longer on the screen it can be recalled using the /COPYLINE/ edit key.) To check the former you can run the CHKDSK command on the specified disk drive. See Chapter 4 for details.

INTER-LINE COMMANDS

This section describes the EDLIN commands that operate on entire lines of text.

The lines you wish to work on may be specified either by entering a line number as a parameter to the command, or by entering a period (.). The latter indicates that the "current line" is to be worked on.

The current line is the location of the last change to the file. It is not necessarily the last line displayed. The current line is indicated by an asterisk between the line number and the first text character. For example:

```
15:* This is the current line
```

Each command description summarises the purpose of the command, defines the command syntax and explains each syntax element. This is followed, for each command, by a detailed account of the command characteristics and some working examples.

Remarks

1. Commands can be entered in either upper or lower case

2. Command keywords and command parameters can be separated from each other by spaces or commas for readability but need not be, except where two line numbers are entered as parameters, in which case they must be separated by a comma or space. For brevity the syntax of this chapter will always indicate comma where separation is obligatory, but note that a space can alternatively be used

3. Commands only become effective after entering /CR/

4. If you make a syntax error when entering a command the message "Entry Error" will be displayed. You must re-enter the command using the correct syntax

line (EDIT LINE)

Enables you to edit a specified line

```
[{line|.}]
```

Where

SYNTAX ELEMENT	MEANING
line	the number of the line to be edited
.	the current line is to be edited

Characteristics

When you enter a line number followed by /CR/, EDLIN displays the line number and the corresponding text, then, on the next screen line, reprints the line number followed by the EDLIN prompt (*) and the cursor (_) . The displayed line serves as the source line and is ready to be edited.

If, instead of a number, you enter a period (.), the current line is displayed and ready for editing. If you enter /CR/ without a line number or a period, then the line immediately following the current line is displayed, unless the current line is the last one in the file, in which case the edit prompt (*) will re-appear.

To edit the line you can use any of the control or edit (intraline) keys, or re-enter the entire line by entering text.

If you wish to abort the operation without changing the source line, press /CTRL/ /C/. Pressing /CR/ with the cursor at the start of the line also aborts the operation without changing anything.

To save the edited line and overwrite the original line, enter /CR/ with the cursor at the end of the edited line. If you enter /CR/ with the cursor in any position other than the start or end of the line, text to the left of the cursor will be written to the file in memory, but all text under and to the right of the cursor will be lost.

Example

Assuming that the contents of the current edit file are as follows:

```
1: This file demonstrates how
2: the line command can
3: be used to edit line
4: four.
```

IF you enter...	THEN EDLIN displays...
4/CR/	4:*four. 4:*_
/INS/ number /SPACE/ /INS/ /COPYTO/ /CR/	4:*four. 4:*number four.

A (APPEND LINES)

Adds lines from the input file on disk to that part of the file currently in memory.

[n] A

Where

SYNTAX ELEMENT	MEANING
n	the number of lines to be added to the file in memory from the input file on disk

Characteristics

This command is only useful for files that are too large to fit into the available memory.

When EDLIN is invoked on a file that is too large to fit into memory, it loads as many lines as possible (as much as will fit into 75% of the available memory). Before using the Append command it is therefore

necessary to write some lines of text to the output file on disk by means of the Write Lines command (see later).

If you enter A without a parameter, lines are appended from the disk file until the available memory is 75% full, or until there are no more lines to append.

Any attempt to read beyond the end of the disk file will yield the following message

```
End of input file
```

Example

IF you enter...	THEN...
100 A /CR/	the next 100 lines of the input file on disk are read into memory and appended to that part of the file that is already there

C (COPY LINES)

Copies a range of lines to a specified line.

[line-a],[line-b],line-c[,count]C

Where

SYNTAX ELEMENT	MEANING
line-a	the first line in the range to be copied
line-b	the last line in the range to be copied
line-c	the line at which the copied lines are to start
count	the number of times the range is to be copied

Characteristics

If line-a is omitted, then the first line defaults to the current line.

If line-b is omitted, then the last line defaults to the current line.

If both line-a and line-b are omitted, then only the current line is copied.

Following the copy operation, lines that previously followed line-c are moved to follow the copied block.

If line-c is beyond the current end of file then the lines are copied to line numbers contiguous to the end of file.

line-b must be greater than or equal to line-a.

On completion line-c becomes the current line.

Examples

Assuming the contents of the current edit file are as follows:

```
1:*This is a sample file
2: to demonstrate the use
3: of the Copy lines command.
```

IF you enter...	THEN the edited file becomes...
1,3,4C /CR/	1: This is a sample file 2: to demonstrate the use 3: of the Copy lines command. 4:*This is a sample file 5: to demonstrate the use 6: of the Copy lines command
,,25,3C /CR/	1: This is a sample file 2: to demonstrate the use 3: of the Copy lines command. 4: This is a sample file 5: to demonstrate the use 6: of the Copy lines command. 7:*This is a sample file 8: This is a sample file 9: This is a sample file
2,3,8C /CR/	1: This is a sample file 2: to demonstrate the use 3: of the Copy lines command. 4: This is a sample file 5: to demonstrate the use 6: of the Copy lines command. 7: This is a sample file 8:*to demonstrate the use 9: of the Copy lines command. 10: This is a sample file 11: This is a sample file

D (DELETE LINES)

Deletes all lines within a specified range.

[line-a][,line-b] D

Where

SYNTAX ELEMENT	MEANING
line-a	the first line in the range to be deleted
line-b	the last line in the range to be deleted

Characteristics

If line-a is omitted, then the first line defaults to the current line.

If line-b is omitted, then the specified line only is deleted.

If D is entered alone, then only the current line is deleted.

After the command has been executed the numbers of the lines following the deleted section are changed to follow the numbers of the lines preceding the deleted section.

The line immediately following the deleted section becomes the current line.

line-b must be greater than (or equal to) line-a.

Example

Assuming the contents of the current edit file are as follows:

```
 1: This is a sample file
 2: to demonstrate the use
 3: of the Delete command.
 4: "line-a D" deletes just the specified line
 5: if ",line-b D" is specified then
 6: all lines from the current line
 7: to line-b, inclusive, are deleted.
 8: Specifying both line-a and
 9: line-b causes that range
10: to be deleted.
11: Specifying D alone deletes the current line.
12: The line subsequent to the
13: deleted line(s) becomes the
14: current line.
```

IF you enter...	THEN the edited file becomes...
8,10 D /CR/	1: This is a sample file 2: to demonstrate the use 3: of the Delete command. 4: "line-a D" deletes just the specified line 5: if ",line-b D" is specified then 6: all lines from the current line 7: to line-b, inclusive, are deleted. 8:*Specifying D alone deletes the current line. 9: The line subsequent to the 10: deleted line(s) becomes the 11: current line.

4 D /CR/	1: This is a sample file 2: to demonstrate the use 3: of the Delete command. 4:*if ",line-b D" is specified then 5: all lines from the current line 6: to line-b, inclusive, are deleted. 7: Specifying D alone deletes the current line. 8: The line subsequent to the 9: deleted line(s) becomes the 10: current line.
, 6 D /CR/	1: This is a sample file 2: to demonstrates the use 3: of the Delete command. 4:*Specifying D alone deletes the current line. 5: The line subsequent to the 6: deleted line(s) becomes the 7: current line.
D /CR/	1: This is a sample file 2: to demonstrate the use 3: of the Delete command. 4:*The line subsequent to the 5: deleted line(s) becomes the 6: current line.

E (END EDITING)

Exits EDLIN and saves the edited file on disk.

```
E
```

Characteristics

The edited file is written to the drive selected when EDLIN was invoked, or to the default drive if no drive was specified. The input file is renamed "filename.BAK". If the file was created during the editing session, no back-up file is created.

You must make sure that enough free space is available on disk to take the output file, otherwise only a portion (at most) will be saved. The remainder will be lost and the message

```
Disk full--write not completed
```

will be displayed, and EDLIN will exit

Example

IF you enter...	THEN...
E /CR/	the current edit file is saved on the diskette specified when EDLIN was invoked. The input file is renamed "filename.BAK", and EDLIN is exited

I (INSERT LINES)

Allows you to insert lines of text before the specified line number.

[{line|.|#}]I

Where

SYNTAX ELEMENT	MEANING
line	the number of the line before which subsequently entered text is to be inserted
.	subsequent text is to be inserted before the current line
#	subsequent text is to be appended to the file

Characteristics

If a line number is entered, then subsequently entered text is inserted immediately before the specified line. If, instead of a line number you enter a period (.), or you enter I on its own, then subsequent text is inserted before the current line. Moreover, if you enter # instead of the line number, text will be appended to the file.

EDLIN remains in insert mode until either /CTRL/ /Z/ or /CTRL/ /C/ is entered. Whilst in insert mode, successive line numbers appear each time /CR/ is pressed.

When you exit insert mode, the line immediately following the last inserted line becomes the current line. The line numbers of this and all subsequent lines are automatically incremented by the number of lines inserted.

When a file is created, you must enter I before entering text. In this case the line number is 1.

Example

Assuming that the contents of the current edit file are as follows:

```
1: This is a sample file
2: to demonstrate
3: using the Insert Lines
4: command
```

IF you enter...	THEN...
3 I /CR/	EDLIN enters insert mode and displays 3:*_
how lines of text /CR/ can be inserted into a file /CR/ at a specified line number /CR/	EDLIN displays 3: how lines of text 4: can be inserted into a file 5: at a specified line number 6:*_
/CTRL/ /Z/ /CR/	EDLIN exits insert mode
L /CR/	The List command (see later) is invoked and EDLIN displays 1: This is a sample file 2: to demonstrate 3: how lines of text 4: can be inserted into a file 5: at a specified line number 6:*using the Insert Lines 7: command

I /CR/ or .I /CR/	EDLIN enters insert mode and displays 6:*_
or before the current line /CR/	EDLIN displays 6: or before the current line 7:*_
/CTRL/ /Z/ /CR/	EDLIN exits insert mode
L /CR/	The List command is invoked and EDLIN displays 1: This is a sample file 2: to demonstrate 3: how lines of text 4: can be inserted into a file 5: at a specified line number 6: or before the current line 7:*using the Insert Lines 8: command
# I /CR/ or 9 /CR/	EDLIN enters insert mode and displays 9:*_
and how lines of /CR/ text can be appended /CR/ to a file. /CR/	EDLIN displays 9: and how lines of 10: text can be appended 11: to a file. 12:*

/CTRL/ /Z/ /CR/	EDLIN exits insert mode
L /CR/	The List command is invoked and EDLIN displays 1: This is a sample file 2: to demonstrate 3: how lines of text 4: can be inserted into a file 5: at a specified line number 6: or before the current line 7: using the Insert Lines 8: command 9: and how lines of 10: text can be appended 11: to a file. 12:*_

L (LIST TEXT)

Displays a specified range of lines.

[line-a][,line-b] L

Where

SYNTAX ELEMENT	MEANING
line-a	the first line in the range to be listed
line-b	the last line in the range to be listed

Characteristics

If you specify both line-a and line-b, then the entire range of lines is displayed, unless this is in excess of 23 lines, in which case the display starts from line-a, but this and subsequent lines are scrolled off the top of the screen until line-b appears on the 23rd line.

If line-a is omitted, but line-b is specified, then the display starts 11 lines before the current line and ends at line-b. If this is more than 23 lines, the screen scrolls down the file until line-b appears on the 23rd line.

If line-b is omitted, but line-a is specified, then 23 lines are displayed, starting from line-a.

If you enter L on its own, then the current line appears on the centre line of the screen (unless the current line is less than line 12), with the preceding 11 lines displayed before it, and the subsequent 11 lines displayed after it. If the current line is before line 12, EDLIN displays the first 23 lines.

Example

Assuming the contents of the current edit file are as follows:

```
   1: This is a sample file
   2: to demonstrate the
   3: use of the List
   4: command
      .
      .
      .
  14:*This is the current line
      .
      .
      .
  23: The List command can be
  24: used to examine
  25: different parts of the
  26: file, up to 23 lines
  27: at once.
```

IF you enter...	THEN EDLIN displays...
2,4 L /CR/	2: to demonstrate the 3: use of the List 4: command
,24 L /CR/	3: use of the List 4: command . . . 14:*This is the current line . . . 23: The List command can be 24: used to examine
24 L /CR/	24: used to examine 25: different parts of the 26: file, up to 23 lines 27: at once.
L /CR/	3: use of the List 4: command . . . 14:*This is the current line . . . . 24: used to examine 25: different parts of the

M (MOVE LINES)

Moves a range of lines to a specified line.

[line-a],[line-b],[line-c]M

Where

SYNTAX ELEMENT	MEANING
line-a	the first of the range of lines to be moved
line-b	the last of the range of lines to be moved
line-c	the line to which the text is to be moved.

Characteristics

If line-a is omitted, then the first line defaults to the current line.

If line-b is omitted, then the last line defaults to the current line.

line-b must be greater than or equal to line-a and may be specified as relative to line-a (by preceding the number with +) or as the actual line number.

Following the move, lines are renumbered depending on the direction of the move. For instance, moving lines 10 to 20 to line 100 would effectively delete lines 10 to 20 (thereby causing all subsequent lines to be moved up the file 11 lines), then the moved lines would become lines 79 to 99.

The first of the moved lines becomes the current line.

Examples

Assuming the contents of the current edit file are as follows:

```
  1: This is a sample file
  2: to demonstrate the use
  3: of the Move lines command.
  4: New first line
     .
     .
     .
100: Next line.
```

IF you enter...	THEN the edit file becomes...
1,+3,100M	1: New first line . . . 97: This is a sample file 98: to demonstrate the use 99: of the Move command 100: Next line.

P (PAGE)

Lists a specified range of lines.

[line-a][,line-b]P

Where

SYNTAX ELEMENT	MEANING
line-a	the first of the range of lines to be displayed
line-a	the last of the range of lines to be displayed

Characteristics

If line-a is omitted, then the first line defaults to the line following the current line.

If the line-b is omitted 23 lines are displayed.

The last line displayed becomes the current line.

The difference between the Page command and the List command is that the Page command changes the current line.

Q (QUIT EDITING)

Quits the editing session but does not save any changes you have made.

```
Q
```

Characteristics

After entering the Quit command EDLIN replies with the message

```
Abort edit (Y/N)?_
```

Entering "n" or any key other than /y/ or /CTRL/ /C/ continues the editing session. Entering "y" terminates the editing session. No BAK file

is created, and any changes made during the editing session are lost. The file on disk remains exactly as it was when EDLIN was invoked. Note also that any previous BAK file is also lost since the current BAK file is always deleted when EDLIN is invoked.

Example

IF you enter...	THEN...
Q /CR/	the editing session is terminated without saving the changes made during the editing session

R (REPLACE TEXT)

Searches a specified range to replace all occurences of one string with another string.

[line-a][,line-b] [?] Rstring-a[/CTRL/ /Z/ string-b]

Where

SYNTAX ELEMENT	MEANING
line-a	the number of the first line in the range on which the Replace Text command is to be executed

line-b	the number of the last line in the range
?	for each occurence of the specified string the O.K.? prompt appears, enabling you to accept or reject the replacement
string-a	the string of characters that are to be replaced
string-b	the string of characters that are to replace string-a

Characteristics

For each line in which a replacement occurrs the modified line is displayed on the screen. If you entered the ? parameter then the prompt

O.K.?

will appear after each replacement. You must then enter "y" or /CR/ to confirm the replacement, or strike any other key to reject it. In either case the search will recommence for the next occurrence of string-a. If ? is not specified, all occurrences of string-a will be replaced by string-b without confirmation.

If you omit string-b, then all occurrances of string-a are deleted.

If you omit line-a then the search will begin from line 1. If you omit line-b then the search will continue to the end of the file in memory. If neither line-a nor line-b is entered, then the entire file in memory will be searched and modified.

Once all replacements have been made, the Replace Text command terminates and the last line in which string-a occurred becomes the current line.

Example

Assuming that the contents of the current edit file are as follows:

```
1: This is a sample file
2: to demonstrate the
3: use of te Replace Text
4: command.
5: Using this command a
6: specified group of characters
7: can be replaced by
8: another group of characters
9:*and can be deleted entirely.
```

IF you enter...	THEN...
5,8 R group of characters /CTRL/ /Z/ string /CR/	all occurrences of "group of characters" in the range line 5 to line 8 are replaced with "string". EDLIN displays 6: specified string 8: another string and line 8 becomes the current line
? Rand /CTRL/ /Z/ or /CR/	each occurrence of "and" within the entire file can optionally be replaced with "or". EDLIN displays 4: commor. O.K.?
N	5: using this commor a O.K.?
N	9: or can be deleted entirely? O.K.?
Y	* and line 9 becomes the current line

L /CR/	the List command is executed to display the file and enable you to see the changes you have made. EDLIN displays 1: This is a sample file 2: to demonstrate the 3: use of the Replace Text 4: command. 5: Using this command a 6: specified string 7: can be replaced by 8: another string 9:*or can be deleted entirely.

Remark

If the replacement string causes a line to expand beyond the limit of 254 characters then the following message is displayed

```
Line too long
```

S (SEARCH TEXT)

Searches a specified range of lines for a specified string.

[line-a][,line-b] [?] Sstring

Where

SYNTAX ELEMENT	MEANING
line-a	the number of the line from which the search is to start
line-b	the number of the last possible line to be searched
?	on finding a matching string the O.K.? prompt is to be displayed, thereby enabling you to accept or reject the particular occurrence
string	the string of characters to be searched for

Characteristics

The command searches the range line-a to line-b for the specified string. If line-a is not specified then the search begins from the line after the current line. If line-b is not specified then the last line in the range is the last line of the file in memory.

When a matching string is found, the corresponding line is displayed. Then if the ? parameter is not specified, the search is terminated and the displayed line becomes the current line. If ? is specified, then the displayed line will be followed by the prompt

O.K.?

To accept the string you must enter either "y" or /CR/. The search will then terminate and the displayed line becomes the current line. Striking any other key, however, will re-commence the search for the next occurence of the string.

If no matching string is found, or if ? is specified and all matching strings are rejected, the message

Not found

is displayed.

Example

Assuming that the contents of the current edit file are as follows:

```
   1: This is a sample file to demonstrate
   2: The use of the Search Text command.
   3: The search can either display the
   4: first occurrence of a specified string
   5: and terminate, or, if specified
   6: to do so, it will enable you to
   7: interactively examine each
   8: occurrance of a string allowing
   9: you to confirm or reject the string.
  10: Once a string is accepted the
  11:*search terminates.
```

IF you enter...	THEN...
2,5 Sstring /CR/	the Search Text command will search lines 2 to 5 of the file, inclusive, for the first occurrence of the string "string", then display the following 4: first occurrence of a specified string and terminate. Line 4 becomes the current line
,8 Sstring /CR/	the search will be made on lines 5 (one after the current line) to 8, inclusive, for the first occurrence of "string". EDLIN will display 8: occurrence of a string allowing and the search will terminate with line 8 as the current line

1? Sstring /CR/	the Search Text command searches the file for the first occurrence of "string". The search starts from line 1. The result is 4: first occurrence of a specified string O.K.?
N	The string is rejected and the search continues for the next occurrence. The result is 8: occurrences of a string allowing O.K.?
Y	the search is terminated and line 8 becomes the current line
Ssample file /CR/	the Search Text command searches for the string "sample file" starting from line 9 (one after the current line), up to the end of the file. The string is not found, hence the message Not found is displayed. The search terminates and line 8 remains the current line.

T (TRANSFER LINES)

Inserts an entire file before a specified line of the current edit file.

```
[line]T[filespec]
```

Where

SYNTAX ELEMENT	MEANING
line	the number of the line before which the file is to be inserted
filespec	the file to be inserted

Characteristics

If the line parameter is omitted then the current line is assumed.

The specified file must be in the same directory as the edit file.

W (WRITE LINES)

Writes a specified number of lines from the file being edited in memory to the output file on disk.

```
[n] W
```

Where

SYNTAX ELEMENT	MEANING
n	the number of lines to be written to diskette, starting from line 1. If this parameter is omitted, then lines of text are written to the output file until the available memory is is 25% full

Characteristics

The Write Lines command is used in conjunction with the Append Lines command when editing files that are too large to fit into the available memory. Lines written to the output file are deleted from memory, and the remaining lines renumbered, starting from line 1. This leaves space available at the end of the file in memory for additional lines to be read from the input file on disk using the Append Lines command.

INTRA-LINE COMMANDS

The intra-line commands are executed using the special editing keys that can be used to perform edits within the current line taking advantage of the source line facility. They enable you to:

- copy one character from the source line to the current line (COPY1)
- copy a specified portion of the source line to the current line (COPYTO)
- copy all remaining characters in the source line to the current line (COPYLINE)
- delete a specified character in the source line (SKIP1)
- delete a specified portion of the source line (SKIPTO)
- kill the current input (KILL)
- enter insert mode to insert text into the current line (INS)
- exit insert mode (enter overstrike mode) (INS)

- make the current line the source line (NEWTEMP)

Moreover, you can also use the MS-DOS control keys when in EDLIN.

For more details about control keys refer to Chapter 3.

The remainder of this section describes each of the intra-line commands in turn. The syntax shows the command as the name of the editing key; refer to Chapter 3 for the actual keystroke combination.

Before starting to edit the current line, a copy of the current line exists in the source line. You then begin editing the current line by entering an edit line by entering text and by using the intra-line commands. Not until you complete the edit line by entering /CR/ does the edit line replace the contents of the current line.

You can select a line to work on using the line command described in the previous section.

COPY1

Copies one character from the source line to the edit line.

/CTRL/ /1/

Characteristics

Pressing the /CTRL/ /1/ copies one character from the source line to the edit line. Insert mode, if active, is automatically turned off.

Example

Assuming that the line to be edited is displayed as follows:

```
1:*This is the COPY1 command
1:*_
```

IF you enter...	THEN...
/CTRL/ /1/	the first character is copied from the source line into the edit line thus: 1:*This is the COPY1 command 1:*T_
/CTRL/ /1/	the next character is copied from the source line into the edit line thus: 1:*This is the COPY1 command 1:*Th_

COPYTO

Copies up to a given character from the source line to the edit line.

/CTRL/ /2/character

Where

SYNTAX ELEMENT	MEANING
character	a character whose first occurrence in the source line will terminate the copy operation. If the character does not appear in the source line nothing will be copied

Characteristics

Pressing /CTRL/ /1/ copies all characters up to but not including a given character from the source line to the edit line. The cursor is moved to the position of the given character. The given character is not displayed.

Example

Assuming that the line to be edited is displayed as follows:

```
1:*This is the COPYTO command
1:*_
```

IF you enter...	THEN...
/CTRL/ /2/c	the characters "This is the COPYTO " are copied from the source line to the edit line thus: 1:*This is the COPYTO command 1:*This is the COPYTO _

COPYLINE

Copies the source line to the edit line.

/CTRL/ /3/

Characteristics

Pressing /CTRL/ /3/ copies all remaining characters from the source line to the edit line regardless of cursor position. Following the copy, the cursor is positioned after the last character on the line. Insert mode, if active, is automatically turned off.

Examples

Assuming that the line to be edited is displayed as follows:

```
1:*This is the COPYLINE command
1:*_
```

IF you enter...	THEN...
/CTRL/ /3/	all remaining characters are copied from the source to the edit line thus : 1:*This is the COPYLINE command 1:*This is the COPYLINE command_

SKIP1

Deletes one character in the source line.

/CTRL/ /O/

Characteristics

Pressing /CTRL/ /O/ skips over one character in the source line without copying it to the edit line. It does not affect insert mode.

Example

Assuming that the line to be edited is displayed as follows:

```
1:*This is the SKIP1 command
1:*
```

IF you enter...	THEN...
/CTRL/ /0/	you skip the letter "T" in the source line
/CTRL/ /3/	the remaining characters are copied from the source line to the edit line thus 1:*This is the SKIP1 command 1:*his is the SKIP1 command_

SKIPTO

Deletes all characters in the source line up to the first occurrence of a specified character.

/CTRL/ /4/character

Where

SYNTAX ELEMENT	MEANING
character	the character in the source line that terminates the string that is to be skipped

Characteristics

Pressing the /CTRL/ /4/ causes characters in the source line to be skipped from the edit character up to but not including the the first occurrence of the given character. If the source line does not contain the given character then no characters are skipped. Nothing is copied to the edit line by this command. Insert mode remains unaffected.

Examples

Assuming that the line to be edited is displayed as follows:

```
1:*This is the SKIPTO command
1:*_
```

IF you enter...	THEN...
/CTRL/ /4/c	all characters in the source line up to the first "c" are skipped over
/CTRL/ /3/	the remaining characters in the source line are copied to the edit line thus: 1:*This is the SKIPTO command 1:*command_

KILL

Terminates input and empties the edit line.

/CTRL/ /9/

Characteristics

Pressing /CTRL/ /9/ clears the edit line, but the source line remains unchanged. The KILL command also displays a back-slash (\) and inserts a carriage return and a line-feed. The cursor is placed immediately under the first character of the terminated line. You can then begin again to edit the line. Insert mode is turned off by this command.

Examples

Assuming your current and edit lines are displayed as follows:

```
1:*This is the KILL command
1:*This is the_
```

IF you enter...	THEN...
/CTRL/ /9/	the edit line is emptied thus: 1:*This is the KILL command 1:*This is the\ _
/CTRL/ /3/	the source line is copied to the edit line thus: 1:*This is the KILL command 1:*This is the\ This is the KILL command

INS

Enters/exits insert mode.

/CTRL/ /8/
or
/./

Characteristics

Pressing /CTRL/ /8/ enters or exits insert mode. On entering insert mode, subsequently entered characters will be inserted before the character under the cursor when insert mode was entered.

Examples

Assuming the line to be edited is displayed as follows:

```
1:*This is INS command
1:*This is _
```

IF you enter...	THEN...
/CTRL/ /8/the /SPACE/	the characters "the" are inserted in the edit line thus: 1:*This is INS command 1:*This is the _

/CTRL/ /8/ /CTRL/ /3/	insert mode is switched off and the remainder of the source line is copied to the edit line thus: 1:*This is INS command 1:*This is the INS command_

NEWTEMP

Creates a new source line by copying the edit line to the source line.

/CTRL/ /5/

Characteristics

Pressing the NEWLINE key copies the edit line to the source line. The original contents of the source line are deleted. An "@" sign appears at the end of the edit line and a carriage return line-feed is inserted. The edit line is also cleared and and insert mode is exited.

Example

Assuming the source and edit lines are displayed as follows:

```
1:*This is the NEWLINE command
1:*_
```

and you want to change the source line to read "This is the NEWTEMP command"

IF you enter...	THEN...
/CTRL/ /2/L	all characters up to the first "L" are copied from the source line to the edit line thus: 1:*This is the NEWLINE command 1:*This is the NEW_
TEMP	the next four characters in the source line are replaced in the edit line by the characters "TEMP" thus: 1:*This is the NEWLINE command 1:*This is the NEWTEMP_
/CTRL/ /3/	the remaining characters in the source line are copied to the edit line thus: 1:*This is the NEWLINE command 1:*This is the NEWTEMP command_
/NEWLINE/	the contents of the source line are replaced with those of the edit line. The display appears thus: 1:*This is the NEWLINE command 1:*This is the NEWTEMP command@ _

9. THE LINKER

ABOUT THIS CHAPTER

This chapter describes the LINK utility.

CONTENTS

INTRODUCTION

MS-LINK is an executable program available on your MS-DOS system disk. It links separate object modules that are the output of the MACRO-86 assembler or a compatible compiler, resolving external references by searching multiple library files. Its output is a relocatable run file, along with a list file that shows external references and error messages.

To run MS-LINK you must provide appropriate object, run, list and library file parameters. In addition, you may enter switches that modify the way in which MS-LINK processes your input. Parameters and switches are fully described in the section "Interactive Entry".

There are three related methods of running MS-LINK. These are described later in this chapter in the following sections:

- Interactive Entry
- Command Line Entry
- Automatic Response File Entry

Interactive entry is the primary method and its section contains all the information common to the three methods.

TEMPORARY FILES

MS-LINK uses available memory for the link session. If the files to be linked create an output file that exceeds available memory, MS-LINK creates a temporary file on the default drive and names it VM.TMP. If MS-LINK needs to create VM.TMP, it displays the message:

```
VM.TMP has been created.
Do not change diskette in drive  <drv:>
```

Once this message is displayed, you must not remove the disk from the default drive until the link session ends. If the disk is removed, the operation of MS-LINK is unpredictable, and MS-LINK might return the error message:

```
Unexpected end of file on VM.TMP
```

MS-LINK uses VM.TMP as virtual memory. The contents of VM.TMP are subsequently written to the file named following the run file prompt. VM.TMP is a working file only and is deleted at the end of the linking session.

If the default drive already has a file by the name of VM.TMP, it will be deleted by MS-LINK and a new file will be allocated; the contents of the previous file are destroyed. You should therefore avoid using VM.TMP as one of your own file names.

CHANGING DISKS

You may wish to change disks during the link operation. For example, if MS-LINK cannot find an object file on the specified disk it prompts you to change disks instead of aborting the session. Or if you enter the /PAUSE switch, MS-LINK pauses and prompts you to change disks before creating the run file. You may change disks when prompted except in the following cases:

- When the disk you wish to change has a VM.TMP file created on it (see the previous section)

- when you have requested a list file on the disk you wish to change

SEGMENTS, GROUPS AND CLASSES

Some of the terms used in this chapter are explained below to help you understand how MS-LINK works. Generally, if you are linking object modules compiled from BASIC, Pascal, or any high-level language, you will not need to know these terms. If you are writing and compiling programs in assembly language, however, you will need to understand MS-LINK and the definitions described below.

In MS-DOS, memory can be divided into segments, classes, and groups. For example:

Group Contents	Segment Names	Segment Class Names
Segment 1	PROG.1	CODE
Segment 2	PROG.2	CODE
Segment 12	PROG.3	DATA

Note that segments 1, 2, and 12 have different segment names but may or may not have the same segment class name. Segments 1, 2, and 12 form a group with a group address of the lowest address of segment 1 (that is, the lowest address in memory).

Each segment has a segment name and a class name. MS-LINK loads all segments into memory by class name from the first class encountered to the last. All segments assigned to the same class are loaded into memory contiguously.

During processing, MS-LINK references segments by their addresses in memory. MS-LINK does this by finding groups of segments.

A group is a collection of segments that fit within a 64K byte area of memory. The segments do not need to be contiguous to form a group. The address of any group is the lowest address of the segments in that group.

At link time, MS-LINK analyses the groups, then references the segments by the address in memory of that group. A program may consist of one or more groups.

If you are writing in assembly language, you may assign the group and class names in your program. In high-level languages (BASIC, COBOL, FORTRAN, Pascal), the naming is done automatically by the compiler.

INTERACTIVE ENTRY

With interactive entry MS-LINK prompts you for each parameter in turn.

Type the following:

```
LINK
```

Four prompts appear, one at a time, requesting the appropriate parameters. These are summarized in the following table.

PROMPT	RESPONSE
Object Modules [.OBJ]:	objfile[+objfile...][switch...]
Run File [objfile.EXE]:	[runfile][switch...]
List File [runfile.MAP]:	[listfile][switch...]
Libraries []:	[libfile[+libfile...]][switch...]

Where

SYNTAX ELEMENT	MEANING
objfile	the file specification of an object module to be linked. Such specifications must be separated from each other with a plus sign (+) or a /SPACE/. The default file extension is .OBJ. If any extension is different from .OBJ it must be specified; otherwise it may be omitted. Segments are loaded by class name, from the first class encountered to the last. The order in which you list object files is therefore significant
runfile	the file specification of the run (executable) file that results from the link session. All run files receive the extension .EXE. Any other extension you may enter is ignored. The default run file name is the first object file name entered
listfile	the file specification of the list file that contains an entry for each segment in the object files and its offset in the run file. The default list file name is the run file name. The default extension is .MAP
libfile	the file specification of each library. Up to eight libraries may be searched; separate each specification with a plus sign (+) or a /SPACE/. The default is standard library search. The default extension is .LIB. Libraries are searched in the order they are listed
switch	the name of any of the switches described in the table below. You may enter any number of switches, at the end of any number of prompt resposes (before pressing /CR/). Switches may be abbreviated to the / together with the first letter or any sequential substring starting with the first letter

SWITCH	MEANING
/DSALLOCATE (or /D)	all data defined to be in DGROUP is loaded at the high end of the group. If the switch is not used MS-LINK loads all data at the low end of the group. At runtime the data space pointer is set to the lowest possible address, allowing the entire storage to be used. Use of the /DSALLOCATE switch in combination with the default load low (that is, the /HIGH switch is not used) permits the user application to allocate dynamically any available memory below the area specifically allocated within DGROUP, yet to remain addressable by the same data space pointer. This dynamic allocation is needed for Pascal and FORTRAN programs. The maximum amount of memory that can be allocated by the application is 64K (or the amount actually available) minus the allocated portion of DGROUP
/HIGH (or /H)	MS-LINK loads the run file as high as possible in memory. If /HIGH is not specified the run file is loaded as low as possible. DO NOT USE THIS SWITCH WITH PASCAL OR FORTRAN PROGRAMS
/LINENUMBERS (or /L)	MS-LINK includes in the list file the line numbers and addresses of the source statements in the input modules. If /LINENUMBERS is not specified, line numbers and addresses are not included. (Not all compilers produce object modules that contain line number information. In these cases, of course, MS-LINK cannot include line numbers).
/MAP (or /M)	MS-LINK lists all public (global) symbols defined in the output modules. If /MAP is not given, MS-LINK will list only errors (including undefined globals). The symbols are listed alphabetically. For each symbol, MS-LINK lists its value and its segment: offset location in the run file. The symbols are listed at the end of the list file

/NODEFAULTLIBRARY SEARCH (or /N)	MS-LINK does not automatically search the default library to resolve external references. For example, linking Pascal object modules with the /N switch stops MS-LINK automatically searching the file PASCAL.LIB
/PAUSE (or /P)	MS-LINK pauses in the link session when the switch is encountered. Normally, MS-LINK performs the linking session without stopping from beginning to end. /PAUSE allows the user to swap disks before MS-LINK outputs the run (.EXE) file. When MS-LINK encounters the /PAUSE switch, it displays the message: About to generate .EXE file Change disks <hit any key> MS-LINK resumes processing when the user presses any key. DO NOT SWAP A DISK THAT IS TO RECEIVE A LIST FILE, OR A DISK USED FOR A TEMPORARY (VM.TMP) FILE
/STACK:size (or /S)	the size of the stack provided for the load module by the MACRO-86 assembler or compiler is overridden. The stack size becomes that specified in the 'size' parameter, which must follow the switch name and a colon. If a value from 1 to 511 is entered, MS-LINK uses 512. At least one object (input) module must contain a stack allocation statement. If not, MS-LINK will return a WARNING: NO STACK STATEMENT error message

Characteristics

After any of these responses, before pressing /CR/, you may enter a comma (,) followed by the answer to what would be the next prompt, without having to wait for that prompt.

If you conclude any response with a semicolon (;) the remaining responses are all assumed to be the default. Linking begins immediately with no further prompting.

Use the plus sign (+) not only to separate lists of object files and libraries but to extend these lists, where necessary, onto more than one line. Enter the plus sign followed by /CR/ at the end of a physical line. This repeats the object file or library prompt, and enables you to continue the logical line with further file names.

Example

This sample shows you the enter of information that is displayed during an MS-LINK session.

In response to the MS-DOS prompt, enter:

```
LINK  /CR/
```

The system displays the following messages and prompts (your answers are underlined):

```
Microsoft Object Linker V.2.00
(C) Copyright 1982 by Microsoft Inc.

Object Modules [.OBJ]: IO+SYSINIT
Run File [IO.EXE]:
List File [NUL.MAP]: IO /MAP
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 can redirect your output to the printer.

3. By specifying the /LINE switch, MS-LINK gives you a listing of all line numbers for all modules. (Note that the /LINE switch can generate a large volume of output.)

4. By pressing /CR/ in response to the Libraries: prompt, an automatic library search is performed.

Once MS-LINK locates all libraries, the linker map displays a list of segments in the order of their appearance within the load module. The list might look like this:

```
Start     Stop      Length    Name
00000H    009ECH    09EDH     CODE
009F0H    01166H    0777H     SYSINITSEG
```

The information in the Start and Stop columns shows the 20-bit hex address of each segment relative to location zero. Location zero is the

beginning of the load module.

The addresses displayed are not the absolute addresses where these segments are loaded.

Because the /MAP switch was used, MS-LINK displays the public symbols by name and value. For example:

```
    ADDRESS             PUBLICS BY NAME
    009F:0012           BUFFERS
    009F:0005           CURRENT DOS LOCATION
    009F:0011           DEFAULT DRIVE
    009F:0008           DEVICE LIST
    009F:0013           FILES
    009F:0009           FINAL DOS LOCATION
    009F:000F           MEMORY SIZE
    009F:0000           SYSINIT

    ADDRESS             PUBLICS BY VALUE
    009F:0000           SYSINIT
    009F:0005           CURRENT DOS LOCATION
    009F:0009           FINAL DOS LOCATION
    009F:000B           DEVICE LIST
    009F:000F           MEMORY SIZE
    009F:0011           DEFAULT DRIVE
    009F:0012           BUFFERS
    009F:0013           FILES
```

COMMAND LINE ENTRY

With command line entry you enter the MS-LINK command along with its parameters, without waiting to be prompted.

You must separate each complete parameter entry from the next with a comma (,). Apart from this, what you enter is the same as with interactive entry. The command syntax is therefore the following:

```
LINK objfile[+objfile...][switch...],[runfile][switch...],
[listfile][switch...],[libfile[+libfile...]][switch...]
```

Where

SYNTAX ELEMENT	MEANING
objfile	an object module to be linked
runfile	the run (executable) file to be created
listfile	the list file to be output
libfile	a library file to be searched
switch	a switch to be applied

See the section "Interactive Entry" for full descriptions of these parameters.

Characteristics

Use the plus sign (+) as with interactive entry not only to separate lists of object files and library files but to extend those lists, where necessary, onto more than one line.

You may enter switches after any one of the four parameter entries (that is, before any of the commas or the final /CR/).

To accept the default parameter for a syntax element, enter a second comma with no space between the two commas. To accept a further default enter a third comma and so on. Remember that you must make an entry for the object file parameter.

If you enter a semicolon (;) at any time, the unspecified parameters all assume default values. Linking begins immediately.

If you enter an incomplete list of parameters and no semicolon is used, MS-LINK prompts you for the next remaining entry. See "Interactive Entry" for a full list of MS-LINK prompts.

Examples

IF you enter...	THEN...
LINK FUN+TEXT+TABLE+ CARE/P/M,,FUNLIST,COBLIB.LIB /CR/	MS-LINK is loaded, then the object modules FUN.OBJ, TEXT.OBJ, TABLE.OBJ and CARE.OBJ are loaded. MS-LINK then pauses (because of the /P switch). The object modules are linked when you press any key, and a global symbol map is produced (because of the /M switch). The default run file FUN.EXE is created, along with a list file called FUNLIST. MAP. The library file COBLIB. LIB is searched for external references
LINK FUN,,/CR/	MS-LINK is loaded, then the object module FUN.OBJ is loaded. The default run file FUN.EXE is created. MS-LINK then prompts for a list file, offering the file FUN.MAP as default

AUTOMATIC RESPONSE FILE ENTRY

With automatic response file entry you enter the name of a file that already contains the answers to some or all of the MS-LINK parameter prompts. Precede this file name by the symbol @. The command syntax is therefore the following:

```
LINK @filespec
```

Where

SYNTAX ELEMENT	MEANING
filespec	the file specification of the file that contains responses to the MS-LINK prompts. Use of file extensions is optional; there is no default extension

Characteristics

Automatic response files can contain several lines of text, each corresponding to an MS-LINK prompt. Responses must be in the same order as with interactive input.

Press /CR/ to indicate the conclusion of each response and the beginning of the next. Type a plus sign (+) followed by /CR/ to continue a response to the object module or libraries prompt on a new line. See the section "Interactive Entry" for full details of each MS-LINK prompt.

Use switches, commas and colons in an automatic response file just as you would in interactive entry.

You can enter the name of more than one automatic response file on the command line, and combine response file names with additional parameters. The combined series of resulting parameters must be a valid sequence of MS-LINK prompts.

When the MS-LINK session begins, each prompt is displayed in order with the responses from the response file. If the response file does not contain answers for all the prompts, (in the form of file names, the semicolon command character or carriage returns), MS-LINK displays the prompt which does not have a response, then waits for you to enter a legal response. When a legal response has been entered, MS-LINK continues the link session.

Example

IF you enter...	THEN...
FUN+TEXT+TABLE+CARE /CR/ /PAUSE/MAP /CR/ FUNLIST /CR/ COBLIB.LIB /CR/	this response file tells MS-LINK to load the four object modules named FUN, TEXT, TABLE, and CARE. MS-LINK pauses before producing a public symbol map to permit you to swap disks. When you press any key, the output files are named FUN.EXE and FUNLIST.MAP. MS-LINK searches the library file COBLIB.LIB

10. THE DEBUGGER

ABOUT THIS CHAPTER

This chapter describes the DEBUG utility.

CONTENTS

INTRODUCTION

The DEBUG utility is a program that provides a controlled testing environment for binary and executable object files. It eliminates the need to re-assemble a program to see if a problem has been corrected by a minor change. Moreover, it enables you to change the contents of a file or CPU register, then to immediately re-execute a program to check the validity of the changes.

HOW TO INVOKE THE DEBUG PROGRAM

DEBUG

The DEBUG program is invoked as follows:

```
DEBUG[filespec[,arglist]]
```

Where

SYNTAX ELEMENT	MEANING
filespec	the specifier of the program file to be debugged
arglist	a list of file name parameters and switches separated by commas. These will be passed to the program specified by the filespec parameter. Thus, when the program is loaded into memory, it is loaded as if it had been invoked with the command filespec arglist

	That is, filespec indicates the file to be debugged, and arglist is the rest of the command line that is used when the file is invoked and loaded into memory

Characteristics

On entering the DEBUG environment DEBUG responds with the hyphen (-) prompt and underline (_) cursor. You are then free to enter any DEBUG command.

If you enter DEBUG without parameters, since no file name has been specified, current memory, disk sectors, or disk files can be worked on using other debugging commands.

If you include the filespec in the command line then the specified file is loaded into memory starting at location 100 (hexadecimal). However, if the file has an EXE extension, then it is relocated to the address specified in the header of the file. Moreover, if the file has the HEX extension, then the file is loaded beginning at the address specified in the HEX file.

Examples

IF you enter...	THEN...
DEBUG /CR/	the DEBUG environment is entered, but without loading a file
DEBUG b:myprog /CR/	the DEBUG environment is entered and the file named "myprog" is loaded into memory from drive B
DEBUG progs dev myrtn, prog1,prog2 /CR/	the DEBUG environment is entered and the file named "myrtn" is loaded into memory from the sub-directory "dev" of directory "progs". The the loaded file also takes two filename parameters: "prog1" and "prog2"

Remarks

When you invoke DEBUG, it sets up a program header at offset 0 in the program work area. You can overwrite this area if you enter DEBUG without parameters. Moreover, if you are debugging a file with a COM or EXE extension you must not tamper with the program header below location 5CH, or DEBUG will terminate.

Do not restart a program after the "Program terminated normally" message is displayed. You must reload the program with the N and L commands for it to run properly.

DEBUGGING COMMANDS

This section describes the DEBUG commands in alphabetical order for ease of reference. Each such command description summarises the purpose of the command, defines the command syntax and explains each syntax element. This is followed, for each command, by a detailed account of the command characteristics and some working examples.

Remarks

1. Commands can be entered in either upper or lower case

2. Command keywords and command parameters can be separated from each other by spaces or commas for readability but need not be, except where two hexadecimal numbers are entered as parameters, in which case they must be separated by a comma or space. For brevity, the syntax of this chapter will always indicate comma where separation is obligatory, but note that a space can alternatively be used

3. Commands only become effective after entering /CR/

4. If you make a syntax error when entering a command the message "ΩError" will be displayed. You must re-enter the command using the correct syntax. You must re-enter the command using the correct syntax

COMMAND PARAMETERS

The following DEBUG command parameters require special definition.

PARAMETER	DEFINITION
address	a hexadecimal value in one of the following formats: - a segment register designation and a hexadecimal offset separated from each other by a colon. For example DS:0300 - a hexadecimal segment and offset separated from each other by a colon. For example 9D0:0100 - a hexadecimal offset value. For example 200 The DEBUG command will append a default segment value from either the DS or CS registers, depending on the command
byte	a one or two character hexadecimal value
drive	0, 1 or 2 depending on whether you wish to select drive A, drive B or drive C, respectively
range	a range of addresses specified as follows address L value where address specifies the start of the range and value specifies the length of the range. For example DS:300L30 indicates a range of 48 locations starting at address 300 in the sector indicated by the DS register. The specified range cannot be greater than 1000 (hexadecimal). To specify this value enter 0000 (or 0) as the value parameter

value	a 1 to 4 character hexadecimal value

A (ASSEMBLE)

Assembles 8086 mnemonics directly into memory.

A[address]

Where

SYNTAX ELEMENT	MEANING
address	the start address at which the subsequently entered line of mnemonics is to be assembled. If this parameter is omitted, location 100 is assumed

Characteristics

After you enter the Assemble command, DEBUG displays the specified address followed by the cursor. You may then enter a line of 8086 assembler mnemonics. On terminating the line with /CR/ the line will be assembled into memory starting at the specified location. The address of the byte subsequent to the assembled code will be displayed on the next line along with the cursor to enable you enter the next line of code. If, instead of a line of 8086 mnemonics, you simply enter /CR/, the Assemble command terminates and the DEBUG prompt re-appears.

All numeric values are hexadecimal and must be entered as 1 to 4 characters. Prefix mnemonics must be specified in front of the opcode to which they refer. You may also enter them on a separate line.

The segment override mnemonics are CS:, DS:, ES: and SS:. The mnemonic for the far return is RETF. String manipulation mnemonics must

explicitly state the string size. For example, use MOVSW to move word strings and MOVSB to move byte strings.

The Assemble command will automatically assemble short, near or far jumps and calls, depending on byte displacement with respect to the destination address. These may be overridden with the NEAR or FAR prefix. For example

```
0100:0500 JMP  502            ;a two-byte short jump
0100:0502 JMP  NEAR 505       ;a three-byte near jump
0100:505  JMP  FAR  50A       ;a five-byte far jump
```

The NEAR prefix may be abbreviated to NE, but the FAR prefix cannot be abbreviated.

DEBUG cannot tell whether some operands refer to a word memory location or to a byte memory location. In this case the data type must be explicitly stated with the prefix "WORD PTR" or "BYTE PTR". Acceptable abbreviations are "WO" and "BY". For example

```
NEG  BYTE PTR [128]
DEC  WO [SI]
```

DEBUG also cannot tell whether an operand refers to a memory location or to an immediate operand. DEBUG uses the convention that operands enclosed within square brackets refer to memory. For example

```
MOV  AX,21              ;Load AX with 21H
MOV  AX,[21]            ;Load AX with the contents of location
                        ;21H
```

Two popular pseudo-instructions are available with the Assemble command. The DB opcode will assemble word values directly into memory. For example

```
DB   1,2,3,4,"THIS IS AN EXAMPLE"
DB   'THIS IS A QUOTE: "'
DB   "THIS IS A QUOTE'"

DW   1000,2000,3000,"BACH'
```

The Assemble command supports all forms of register indirect commands. For example

```
ADD  BX,34[BP+2].[SI-1]
POP  [BP+DI]
PUSH [SI]
```

All opcode synonyms are also supported. For example

```
LOOPZ   100
LOOPE   100
JA      200
JNBE    200
```

Example

IF you enter...	THEN...
A200 /CR/	DEBUG displays 09AC:0200 _
MOV AX,[21] /CR/	the 8086 mnemonics are assembled starting at location 200. The byte location subsequent to the assembled code is then displayed thus 09AC:0203 _
/CR/	the Assemble command terminates and the DEBUG prompt re-appears

C (COMPARE)

Compares the contents of two areas of memory.

Crange,address

Where

SYNTAX ELEMENT	MEANING
range	the range of addresses defining the first area to be compared. If no segment is specified then the segment specified in the DS register is assumed
address	the start of the area to be compared with the area specified by the range parameter

Characteristics

The Compare command compares the area of memory specified by the range parameter with an area of the same size starting at the location specified by the address parameter.

If the contents of the two areas are identical nothing is displayed. If there are differences, then the differences are displayed in the form

```
address1 contents1 contents2 address2
```

where: address1 indicates the address in the first area and contents1 its contents; and address2 indicates the corresponding address in the second area and contents2 its contents.

Example

IF you enter...	THEN...
C100, 1FF, 300 /CR/ or C100L100, 300 /CR/	the area of memory from 100 to 1FF is compared with the area of memory from 300 to 3FF

D (DUMP)

Displays an area of memory.

D[{range/address}]

Where

SYNTAX ELEMENT	MEANING
range	the range of addresses whose contents are to be displayed. If you enter only an offset, then the segment specified in the DS register is assumed
address	the address from which the display is to start. The contents of this address and the subsequent 127 locations are displayed

Characteristics

If D is specified without parameters then the 128 bytes following the last address to be displayed are displayed. If no location has yet been accessed then the dump will start from location DS:100.

If D and the range parameter are specified then the contents of that range of addresses are displayed. If this takes more than 24 display lines the display is scrolled until the contents of the final address in the range are displayed on line 24.

If D and a single address are specified, then the contents of the 128 locations starting from the specified address are displayed.

The dump is displayed in two portions:

- a hexadecimal dump, where each byte is represented by its hexadecimal value

- an ASCII dump, where the equivalent ASCII character for the byte is displayed. If there is no corresponding printable ASCII character then the dump displays a period (.)

Each line of the dump begins with an address which is followed by the hexadecimal contents of the 16 bytes starting from the addressed location. The eighth and ninth bytes are separated by a hyphen (-). The right-hand columns display the equivalent ASCII values. Each line of the display, except possibly the first, begins on a 16 byte boundary.

Examples

IF you enter...	THEN...
D 100,110 /CR/ or D100L11 /CR/	a hexadecimal and ASCII dump of lines 100 to to 110 (hexadecimal), inclusive, are displayed
D /CR/	a hexadecimal and ASCII dump of the 128 bytes starting from location 111 hexadecimal) is displayed
D200 /CR/	a hexadecimal and ASCII dump of the 128 bytes starting from location 200 (hexadecimal) is displayed

E (ENTER)

Replaces the contents of memory locations at the byte addresses specified.

Eaddress[,bytevalue[,bytevalue...]]

Where

SYNTAX MEANING	MEANING
address	the address of the location whose value is to be replaced; or the address of the first of a succession of locations whose contents are to be replaced. If only an offset is specified then the segment indicated by the DS register is assumed
bytevalue	the hexadecimal byte value that is to replace the contents of the specified address. The first bytevalue parameter will replace the contents of the location specified by the address parameter. A second byte value will replace the contents of the location following that specified by the address parameter, and so on

Characteristics

If the command is entered without the byte value list, then DEBUG displays the specified address and its contents. The Enter command then waits for you to perform one of the following:

- replace the displayed byte value by entering another value. You simply enter the new value after the current value. If you enter an illegal value, or if you type more than two digits then the illegal or extra character is not echoed

- advance to the next byte by pressing /SPACE/. To change the value of this byte simply enter the value as described above. If you advance beyond an eight-byte boundary, DEBUG starts a new display line with the address displayed at the start of the line. To advance to the next byte without changing the current byte simply press /SPACE/ again

- to return to the previous byte enter hyphen (-). On doing so DEBUG starts a new display line with the address of the byte you have returned to and its contents. You can then change the contents of this location as described above. To move back one byte further without changing this value simply enter hyphen again, and another new display line will be generated

- terminate the Enter command by pressing /CR/. This key may be pressed in any byte position

If you specify byte values in the command line then the first of these byte values will replace the contents of the location specified by the address parameter. Subsequent entries in the list of byte values will replace subsequent bytes in memory.

Examples

IF you enter...	THEN...
E100 /CR/	DEBUG displays something like 058D:0100 CD._
26	the value of location 100 is changed to 26 and DEBUG displays 058D:0100 CD.26_
/SPACE/	the next byte (location 101) is displayed 058D:0100 CD.26 20._
/SPACE/	the next byte (location 102) is displayed 058D:100 CD.26 20. 00._
/-/	the previous byte (location 101) is displayed on the next line 058D:0100 CD.26 20. 00. 058D:0101 20._

30 /CR/	the contents of location 101 are changed to 30 and the Enter command is terminated 058D:0100 CD.26. 20. 00. 058D:0101 20.30 >_
E200,26,0A,19,23 /CR/	the contents of byte locations 200, 201, 202 and 203 are changed to 26, 0A, 19 and 23, respectively

F (FILL)

Fills an area of memory with specified byte values.

Frange,bytevalue[,bytevalue...]

Where

SYNTAX ELEMENT	MEANING
range	the range of addresses whose contents are to be overwritten with the specified byte values. If only the offset is specified then the segment indicated by the DS register is assumed
bytevalue	a two digit hexadecimal value that is to overwrite the contents of the specified address

Characteristics

If the specified range contains more bytes than the list of byte values, then the list of byte values is repeated until the specified range is filled.

If the list of byte values is longer than the specified range then the extra byte values are ignored.

Example

IF you enter...	THEN...
F04BA:100L100,42,45, 48,37,20 /CR/	DEBUG fills memory locations 04BA:100 to 04BA:1FF with the byte values specified. The five values are repeated until all 256 locations are filled

G (GO)

Executes the program currently in memory, optionally halting a specified breakpoint and displaying information about the system/program environment.

G[=address][,address]...

Where

SYNTAX ELEMENT	MEANING
=address	the address in memory at which program execution is to start."=" must be entered to distinguish a start address from a breakpoint address
address	the breakpoint address at which program execution is to halt and the register and flag states to be displayed along with the next instruction to be executed

Characteristics

If you enter G without parameters then the program currently in memory is executed starting from the address specified by the CS and IP registers.

If you specify the =address parameter, then the contents of the CS and IP registers are changed to those specified by the =address parameter, and the program in memory is executed, starting from that point.

If you specify one or more breakpoint addresses then program execution stops at the first such address encountered and displays the contents of the registers, the state of the flags and the next instruction to be executed (see the Register command for a description of the display).

You may specify up to ten breakpoint addresses, in any order. If your program has many paths you can use this feature to ensure that your program halts, whichever path it takes.

If you enter more than ten breakpoints DEBUG will display

```
BP Error
```

Before executing the program the Go command replaces the contents of the breakpoint locations with an interrupt instruction (hexadecimal CC). When program execution halts at such a location DEBUG restores the original values of all the specified breakpoint locations. However, if the program terminates normally (that is, not at a specified breakpoint), then the breakpoint values are not restored.

Each breakpoint address that you specify must point to the first byte of an 8086 instruction, otherwise unpredictable results will occur.

The user stack pointer must have six bytes available for this command, otherwise unpredictable results will occur.

Example

IF you enter...	THEN...
G=200,1AF,141 /CR/	the program currently in memory is executed starting from location 200. Assuming location 141 is encountered before 1AF, then the program halts at location 141 and the register and flag values are displayed along with the next instruction to be executed. If neither breakpoint location is encountered, then the program terminates normally
G /CR/	if, in the previous example, the progrm halted at location 141, then program execution continues from that address. If program execution terminated normally in the previous example, then program execution again starts at location 200

H (HEX)

Calculates and displays the sum and the difference of two hexadecimal values.

Hvalue-a,value-b

Where

SYNTAX ELEMENT	MEANING
value-a	the first of two hexadecimal values
value-b	the hexadecimal value that is to be added to or subtracted from value-a

Characteristics

The hexadecimal values may be up to four characters long.

The Hex command displays two four digit values:

- the first is the result of adding value-b to value-a
- the second is the result of subtracting value-b from value-a

Example

IF you enter...	THEN...
H19F,10A /CR/	DEBUG displays 02A9 0095
HFFFF,2 /CR/	DEBUG displays 0001, FFFD

I (INPUT)

Inputs and displays (in hexadecimal) one byte from the specified port.

Ivalue

Where

SYNTAX ELEMENT	MEANING
value	the address of the port that the byte is to be input from

Characteristics

The port address can be up to 16 bits.

Example

IF you enter...	THEN...
I2F8	the byte at the addressed port is input and displayed

L (LOAD)

Loads a file or absolute disk blocks into memory.

```
L[address[,drive,block-a,block-b]]
```

Where

SYNTAX ELEMENT	MEANING
address	the address in memory at which the file or specified blocks is to be loaded. If only an offset is entered then the segment indicated by the CS register is assumed
drive	the drive from which disk blocks are to be loaded. For drive A you must enter 0, for drive B you must enter 1, or for drive C you must enter 2
block-a	the first of a range of blocks to be loaded from the disk specified by the drive parameter
block-b	the number of blocks to be loaded

Characteristics

If all parameters are specified then DEBUG loads blocks of information from disk into memory. The first such block to be loaded is that specified by the block-a parameter. It is loaded at the memory location specified by the address parameter. This block is the first of a continuous range of blocks to be loaded, the number of which is specified by the block-b parameter.

If you enter L without parameters, or with just the address parameter, the file whose file control block is correctly formatted at location CS:5C is loaded into memory. The file control block at CS:5C is set either to the filespec specified at the invokation of the DEBUG command, or to the filespec specified by the most recent subsequent Name command.

If L is entered alone, then the file is loaded at location CS:100. If you specify L and the address parameter, the file is loaded at the specified address. In either case DEBUG sets the BX:CX registers to the number of bytes loaded.

If the file has an EXE extension, then it is relocated to the load address specified in the loader of the EXE file. That is, the address parameter to the Load command is ignored. Note that the header itself is stripped off the EXE file before the file is loaded into memory. Thus the size of the EXE file on disk will differ from its size in memory.

If the file is a HEX file, then entering the Load command with no parameters causes the file to be loaded starting at the address specified within the HEX file. If the address parameter, however, is specified then loading starts at the address which is the sum of the address specified and the address in the HEX file.

Examples

The following examples assume the system to be initially in MS-DOS.

IF you enter...	THEN...
debug /CR/ Nb:file.com /CR/ L /CR/	the debugger is entered and the subsequent Name command sets the file control block at CS:5C to identify file "file.com" on the diskette inserted in drive B. The Load command then loads this file into memory starting at CS:100 (the default address)
debug b:file.com /CR/ L300 /CR/	file.com is loaded into memory at location CS:100 by the DEBUG command. It is then relocated to CS:300 by the Load command
debug /CR/ L500,1,0F,6D /CR/	109 blocks are loaded into memory from drive B starting from block 0F. They are placed in memory starting at location CS:500

M (MOVE)

Moves the contents of a range of memory addresses to the locations beginning at a specified address.

```
Mrange,address
```

Where

SYNTAX ELEMENT	MEANING
range	the area of memory whose contents are to be moved. If you only entered an offset then the segment indicated in the DS register is assumed
address	the start of the destination area. If you only an offset then the segment indicated by the DS register is assumed

Characteristics

If the source and destination areas overlap the move is performed without loss of data.

The contents of the source area are not changed by the move, unless the destination area overlaps it.

If you specify an address as the end of the range you must only enter the offset. The segment specified, or defaulted to, in the start address of the range is assumed.

Example

IF you enter...	THEN...
MCS:100,110,CS:500 /CR/ or MCS:100L11,CS:500 /CR/	the 17 bytes starting at location CS:100 are copied to the 17 bytes starting at location CS:500

N (NAME)

Provides filenames for the Load and Write commands or filename parameters for the program to be debugged.

Nfilespec[,filespec]...

Where

SYNTAX ELEMENT	MEANING
filespec	the file specifiers of a file to be loaded into memory, written to diskette, or used as a filename parameter to the file currently in memory

Characteristics

The name command can be used to provide:

- the name of the disk file to be loaded into memory by a subsequent Load command

- the name to be assigned to the file currently in memory when the file is subsequently written to disk

- file name parameters to the file in memory to be debugged

The first case enables you to specify the file you wish to debug after entering the DEBUG environment. That is, you can enter DEBUG without specifying parameters, then use the Name command to name the disk file you wish to debug, then load the file into memory using the Load command. This has the same effect as entering the filename as the first parameter to the DEBUG command. In either case the file control block for the file to be debugged is set up at location CS:5C and the file is loaded.

In the second case the file is already in memory and the Name command sets up the file control block for the specified file name at location CS:5C. When a Write command is subsequently entered the file in memory is written to disk with the file name whose file control block is set up at location CS:5C.

In the third case the Name command provides file name parameters for the program currently in memory. Whatever file control block was set at CS:5C is replaced by that of the first such parameter. If a second file parameter is specified then its file control block is set up at location CS:6C. Only two file control blocks are set up although additional file name parameters may be included if required. All the file names specified are placed in a save area at CS:81, with CS:80 containing a character count. Parameters specified in this way are analogous to file names specified in the argument list to the DEBUG command.

Examples

IF you enter...	THEN...
DEBUG /CR/ Nb:file.com /CR/ L /CR/	the system enters the DEBUG environment and the file named file.COM resident on drive B has its file control block set up at location CS:5C. The Load command subsequently loads this file into memory. This sequence has the same effect as entering "DEBUG b:file.com"

Nb:newfile.com /CR/ W /CR/	the file control block is set up at location CS:5C for the file specifier "b:newfile.com". The subsequent Write command writes the file currently in memory to drive B and names the file "newfile.com"
DEBUG b:file1.com /CR/ Nfile2.dat,file3. dat /CR/ G /CR/	the DEBUG command loads the file named "file1.com from drive B to be debugged. The Name command sets up two file control blocks at locations CS:5C and CS:6C for the file specifiers b:file2.dat and b:file3.dat, respectively. These files then become parameters to file1.COM when the subsequent Go command executes file1.COM

O(OUTPUT)

Sends a specified byte to an output port.

Ovalue,byte

Where

SYNTAX ELEMENT	MEANING
value	the address of the output port. It must be specefied in hexadecimal and can be up to 16 bits
byte	a two-digit hexadecimal value to be sent to the specified port

Example

IF you enter...	THEN...
02F8, 4F /CR/	the byte value 4F is output to the port 2F8

Q(QUIT)

Terminates the DEBUG program.

```
Q
```

Characteristics

The Quit command terminates the debugger without saving the file you are working on. Control is returned to MS-DOS command mode.

Example

IF you enter...	THEN...
Q	the DEBUG program terminates and returns you to MS-DOS command mode

R(REGISTER)

Displays the contents of the registers and flag settings, or displays the contents of a specified register with the option to change that value, or displays the flag settings with the option of reversing any number of those settings.

```
R[register-name|F]
```

Where

SYNTAX ELEMENT	MEANING
register-name	any valid register name whose contents are to be examined and optionally changed. This may be one of: AX DX SI ES IP BX SP DI SS PC CX BP DS CS Note: IP and PC both refer to the Instruction Pointer
F	the flag settings are to be displayed and, optionally, changed

Characteristics

If you enter R without parameters, then the contents of all registers are displayed along with the flag settings and the next instruction to be executed. For example:

```
AX=058D BX=0000 CX=0000 DX=0000 SP=FFF0 BP=0000 SI=0000 DI=0000
DS=058D ES=058D SS=058D CS=058D IP=013B NV UP EI PL NZ NA PO NC
058D:013B 83D8          MOV DS,AX
```

If you enter R with a register name, then DEBUG displays the contents of that register. The command then waits for you to do one of the following:

- press /CR/ to terminate the Register command without changing the value of the displayed register
- change the value of the register by entering the two-digit hexadecimal value then terminate the Register command by entering /CR/

The valid flag values are shown in the following table:

FLAG NAME	SET	CLEAR
Overflow	OV (yes)	NV (no)
Direction	DN (decrement)	UP (increment)
Interrupt	EI (enabled)	DI (disabled)
Sign	NG (negative)	PL (plus)
Zero	ZR (yes)	NZ (no)
Auxiliary carry	AC (yes)	NA (no)
Parity	PE (even)	PO (odd)
Carry	CY (yes)	NC (no)

If you enter RF, then the current flag settings are displayed. You can then either

- press /CR/ to terminate the Register command without changing the flag values, or
- change the setting of one or more flags by entering the alternate value of the appropriate flags. The new values may be entered in any order, with or without delimiters.

Examples

IF you enter...	THEN...
R /CR/	DEBUG displays the contents of all registers, flag settings and the next instruction to be executed
IP /CR/	DEBUG displays the contents of the Instruction Pointer. For example IP 0139 :_
013B /CR/	the contents of the Instruction Pointer are changed to 013B
RF /CR/	DEBUG displays the flag settings. For example NV UP EI PL NZ NA PO NC-_
PE R Z DING /CR/	The Parity flag is set to even (PE), the Zero flag is set (ZR), the Interrupt flag is cleared (DI), and the Sign flag is set (NG).
RF /CR/	DEBUG displays the new state of the flags NV UP DI NG ZR NA PE NC-_

S (SEARCH)

Searches a specified range for a list of bytes.

Srange,list

Where

SYNTAX ELEMENT	MEANING
range	the range of addresses within which the search is to be made. If you only enter the offset then the segment indicated by the DS register is assumed
list	the list of bytes to be searched for. Bytes in the list must be separated by a space or a comma

Characteristics

For each occurrence of the list of bytes within the specified range, DEBUG returns the address of the first byte. If no address is returned, no match was found.

Example

IF you enter...	THEN...
S100L100,20 /CR/ or S100,1FF,20 /CR/	DEBUG displays the address of every occurrence of byte value 20 in the address range 100 to 1FF, inclusive. For example 058D: 010C 058D: 0110 058D: 0115 058D: 0118 058D: 0120 058D: 0128 058D: 0125

T (TRACE)

Executes one or more instructions and displays the register contents, flag settings and the next instruction to be executed.

```
T[=address][,value]
```

Where

SYNTAX ELEMENT	MEANING
=address	DEBUG is to commence execution at this address
value	the number of instructions to be executed

Characteristics

If the =address parameter is not specified then execution begins at CS:IP.

If the value parameter is not specified then only one instruction is executed.

The display generated is of the same format as that of the Register command (without parameters).

Examples

IF you enter...	THEN...
T = 200,5 /CR/	five instructions, starting with the one at location 200 are executed, and the register and flag values following each instruction are displayed along with the next instruction to be executed
T /CR/	the instruction pointed to by CS:IP is executed and the register and flag contents are displayed along with the next instruction to be executed

U (UNASSEMBLE)

Disassembles strings of bytes in memory and displays them as assembler-like statements along with their corresponding addresses.

```
U[range|address]
```

Where

SYNTAX ELEMENT	MEANING
range	the range of addresses whose byte values are to be disassembled. If you do not specify the segment then the segment indicated by the CS register is assumed

address	the start of a 32 byte area of memory to be disassembled. If you only enter an offset then the segment indicated by the CS register is assumed

Characteristics

If neither the range nor address parameter is specified, then 32 bytes are disassembled starting at location CS:IP.

The number of bytes disassembled may be slightly more than the number you specified. This is because instructions are not always the same length and the final address in a range will not always contain the last byte of an instruction.

The first address of a range, or the address parameter, must always refer to the first byte of an 8086 instruction, otherwise results will be unpredictable.

Example

IF you enter...	THEN...
U058D:204L8 /CR/	eight bytes starting at location 058D:204 are disassembled and the result displayed: 058D:0204 8D16DF0D LEA DX,[0DDF] 058D:0208 42 INC DX 058D:0209 03D0 ADD DX,AX 058D:020B 8916E50B MOV [0BE5],DX

W (WRITE)

Writes the file being debugged to disk.

W[address[,drive,block-a,blocks]

Where

SYNTAX ELEMENT	MEANING
address	the start address of the code in memory that is to be written to disk. If you enter only an offset then the segment indicated in the CS register is assumed
drive	the drive containing the specified blocks to which code in memory is to be written. For drive A you must enter 0, for drive B you must enter 1, or for drive C you must enter 2
block-a	the block number on disk that is the first of a contiguous range of blocks to be overwritten with code from memory
blocks	the number of disk blocks to be overwritten with code from memory

Characteristics

If you enter the Write command without parameters, then the file is written to disk starting from memory address CS:100. If you specify the address parameter then the file in memory, starting from the specified address, is written to disk.

In either case, before executing the Write command, BX:CX must be set to the number of bytes to be written. This value was set up correctly when the file was loaded (either by the Load command or the DDEBUG command itself). However, if, since loading the file, you have executed a Go or Trace command, then the value of BX:CX will have been changed. You must be sure this value is set up correctly.

When the Write command writes a file to disk it obtains the drive specifier and file name via the file control block set up at CS:5C. If no drive specifier is set up then the default is assumed. This file control block is set up either by the DEBUG command or a subsequent Name command. If it does not indicate the file specifier you require, you must set up this file control block using the Name command.

When the file is written to disk it overwrites the version currently on disk unless the specified file name does not exist, in which case a new file is created.

If all parameters are specified then the code in memory is written to the drive specified by the parameter. The data to be written starts at the memory location specified by the address parameter, and is written to the blocks on disk specified by the block-a and blocks parameters. You must therefore be **extremely** careful to specify the required blocks, since information held there will be destroyed by this operation.

Examples

IF you enter...	THEN...
W /CR/	the file in memory, starting from location CS:100, is written to disk with the file specifier defined by the file control block set up at location CS:5C
W200 /CR/	the file in memory, starting from location CS:200, is written to disk with the file specifier defined by the file control block set up at location CS:5C
W200,1,1F,20 /CR/	blocks 1F and 20 on drive B are overwritten with the data starting at memory location CS:200

11. FILE COMPARISON UTILITY

ABOUT THIS CHAPTER

This chapter describes the FC (File Comparison) utility.

CONTENTS

FILE COMPARISON

FC

It is sometimes useful to compare files on your disk. If you have copied a file and modified the copy, you may later want to compare copies to check the changes you have made. You can do this using the MS-DOS File Comparison utility (FC).

The File Comparison utility compares the contents of two files. The difference between the two files can be output to the terminal screen or to a third file. The files being compared may be either source files (files containing source statements of a programming language), or binary files (output from the MACRO-86 assembler, the MS-LINK Linker utility, or a high-level language compiler).

The comparisons are either on a line-by-line or a byte-by-byte basis. The line-by-line comparison isolates blocks of lines that are different between the two files and prints those blocks of lines. This is the default method. The byte-by-byte comparison displays the bytes that are different between the two files.

```
FC [/#][/B][/C][/W] filename1 filename2
```

Where

SYNTAX ELEMENT	MEANING
#	a digit from 1 to 9 (see switch table below)
filename1	the name of the first file to be compared
filename2	the name of the second file to be compared

Type

External

Characteristics

The following table describes the switches available with the File Comparison utility.

SWITCH	MEANING
/#	the number of lines required to match for the files to be considered as matching again after a difference has been found. # can be any number from 1 to 9. If this switch is not specified, the number defaults to 3. This switch is used only in source comparisons
/B	a binary comparison of both files is performed. The two files are compared byte-to-byte, with no attempt to re-synchronise after a mismatch. The mismatches are printed as follows: --ADDRS----F1----F2- xxxxxxxx yy zz (where xxxxxxxx is the relative address of the pair of bytes from the beginning of the file). Addresses start at 00000000; yy and zz are the mismatched bytes from file1 and file2, respectively. If one of the files contains less data than the other, then a message is printed out. For example, if file F1 ends before file F2, then the utility displays: ***Data left in F2***

/C	the utility ignores the case of letters. All letters in the files are considered uppercase letters. For example, Much_MORE_data_IS_NOT_FOUND will match much_more_data_is_not_found This switch is used only in source comparisons
/W	the utility compresses whites (tabs and spaces) during the comparison. Thus, multiple contiguous whites in any line will be considered as a single white space. Note that although FC compresses whites, it does not ignore them. The two exceptions are beginning and ending whites in a line, which are ignored. For example (note that an underscore represents a white) ___More__data_to_be_found____ will match with More_data_to_be_found and with _______More_______data_to_be_____found_______ but will not match with ____Moredata_to_be_found This switch is used only in source comparisons

The File Comparison utility reports the difference between the two files you specify by displaying the first filename, followed by the lines that differ between the files, followed by the first line to match in both files. FC then displays the name of the second file followed by the lines that are different, followed by the first line that matches. The default for the number of lines that must match before FC recognises a match is 3. (If you want to change this default, specify the number of lines with the /# switch.) For example:

```
        ...
        ...
_________<filename1>
<difference>
<1st line to match file2 in file1>

_________<filename2>
<difference>
<1st line to match file1 in file2>
_____________________________________________
        ...
        ...
```

If there are too many differences (involving too many lines), the program simply reports that the files are different and stops.

If no matches are found after the first difference is found, FC displays:

```
*** Files are different ***
```

and returns to the MS-DOS default drive prompt.

The comparison report is sent to the video screen unless you specify output redirection to a file.

FC uses a large amount of memory as buffer (storage) space to hold the source files. If source files are larger than available memory, FC compares only what can be loaded into the buffer space. If no lines match in those portions of the files that have been loaded into the buffer space, FC simply displays the message:

```
*** Files are different ***
```

For binary files larger than available memory, FC compares both files completely, overlaying the portion in memory with the next portion from disk. All differences are output in the same manner as those files that fit completely in memory.

Examples

Assume these two ASCII files are on disk:

ALPHA.ASM	BETA.ASM
A B C D E F G H I M N O P Q R S T U V W X Y Z	A B C G H I J 1 2 P Q R S T U V 4 5 W X Y Z

The following examples show three possible ways of using FC to compare the contents of these two files.

IF you enter...	THEN...
FC ALPHA.ASM BETA.ASM /CR/	FC compares ALPHA.ASM with BETA.ASM and displays the differences on the terminal screen. All the defaults remain intact. The output appears as follows on the terminal screen (the Notes do not appear): ----------ALPHA.ASM D E F G ----------BETA.ASM G ---------- ----------ALPHA.ASM M N O P ----------BETA.ASM J 1 2 P ---------- ----------ALPHA.ASM W ----------BETA.ASM 4 5 W NOTE: ALPHA file contains DEFG, BETA contains G NOTE: ALPHA file contains MNO where BETA contains J12 NOTE: ALPHA file contains W where BETA contains 45W

```
FC /4 ALPHA.ASM BETA.ASM
>PRN  /CR/
```

the following output appears on the line printer (the Notes do not appear):

```
----------ALPHA.ASM
D
E
F
G
H
I
M
N                    NOTE: P is the 1st of
O                    a string of 4 matches
P

----------BETA.ASM
G
H
I
J
1
2
P

---------------------------------

----------ALPHA.ASM
W
                     NOTE: W is the 1st of
----------BETA.ASM    a string of 4 matches
4
5
W

---------------------------------
```

```
FC /B ALPHA.ASM BETA.ASM
/CR/
```

the following binary comparison report appears:

```
--ADDRS----F1---F2--

 00000009   44   47
 0000000C   45   48
 0000000F   46   49
 00000012   47   4A
 00000015   48   31
 00000018   49   32
 0000001B   4D   50
 0000001E   4E   51
 00000021   4F   52
 00000024   50   53
 00000027   51   54
 0000002A   52   55
 0000002D   53   56
 00000030   54   34
 00000033   55   35
 00000036   56   57
 00000039   57   58
 0000003C   58   59
 0000003F   59   5A
 00000042   5A   1A
*** DATA left in F1 ***
```

A. ASCII CODE

ABOUT THIS APPENDIX

This appendix provides a table of binary, decimal and hexadecimal ASCII codes.

CONTENTS

ASCII CODE

This table shows decimal (a), hexadecimal (b), and binary (c) representation of the ASCII code. (Boxed items are different on national keyboards.)

a	b	c	d	a	b	c	d	a	b	c	a	b	c
0	00	0000 0000	NUL	64	40	0100 0000	@	128	80	1000 0000	192	C0	1100 0000
1	01	0000 0001	SOH	65	41	0100 0001	A	129	81	1000 0001	193	C1	1100 0001
2	02	0000 0010	STX	66	42	0100 0010	B	130	82	1000 0010	194	C2	1100 0010
3	03	0000 0011	ETX	67	43	0100 0011	C	131	83	1000 0011	195	C3	1100 0011
4	04	0000 0100	EOT	68	44	0100 0100	D	132	84	1000 0100	196	C4	1100 0100
5	05	0000 0101	ENQ	69	45	0100 0101	E	133	85	1000 0101	197	C5	1100 0101
6	06	0000 0110	ACK	70	46	0100 0110	F	134	86	1000 0110	198	C6	1100 0110
7	07	0000 0111	BEL	71	47	0100 0111	G	135	87	1000 0111	199	C7	1100 0111
8	08	0000 1000	BS	72	48	0100 1000	H	136	88	1000 1000	200	C8	1100 1000
9	09	0000 1001	HT	73	49	0100 1001	I	137	89	1000 1001	201	C9	1100 1001
10	0A	0000 1010	LF	74	4A	0100 1010	J	138	8A	1000 1010	202	CA	1100 1010
11	0B	0000 1011	VT	75	4B	0100 1011	K	139	8B	1000 1011	203	CB	1100 1011
12	0C	0000 1100	FF	76	4C	0100 1100	L	140	8C	1000 1100	204	CC	1100 1100
13	0D	0000 1101	CR	77	4D	0100 1101	M	141	8D	1000 1101	205	CD	1100 1101
14	0E	0000 1110	SO	78	4E	0100 1110	N	142	8E	1000 1110	206	CE	1100 1110
15	0F	0000 1111	SI	79	4F	0100 1111	O	143	8F	1000 1111	207	CF	1100 1111
16	10	0001 0000	DLE	80	50	0101 0000	P	144	90	1001 0000	208	D0	1101 0000
17	11	0001 0001	DC_1	81	51	0101 0001	Q	145	91	1001 0001	209	D1	1101 0001
18	12	0001 0010	DC_2	82	52	0101 0010	R	146	92	1001 0010	210	D2	1101 0010
19	13	0001 0011	DC_3	83	53	0101 0011	S	147	93	1001 0011	211	D3	1101 0011
20	14	0001 0100	DC_4	84	54	0101 0100	T	148	94	1001 0100	212	D4	1101 0100
21	15	0001 0101	NAK	85	55	0101 0101	U	149	95	1001 0101	213	D5	1101 0101
22	16	0001 0110	SYN	86	56	0101 0110	V	150	96	1001 0110	214	D6	1101 0110
23	17	0001 0111	ETB	87	57	0101 0111	W	151	97	1001 0111	215	D7	1101 0111
24	18	0001 1000	CAN	88	58	0101 1000	X	152	98	1001 1000	216	D8	1101 1000
25	19	0001 1001	EM	89	59	0101 1001	Y	153	99	1001 1001	217	D9	1101 1001
26	1A	0001 1010	SUB	90	5A	0101 1010	Z	154	9A	1001 1010	218	DA	1101 1010
27	1B	0001 1011	ESC	91	5B	0101 1011	[	155	9B	1001 1011	219	DB	1101 1011
28	1C	0001 1100	FS	92	5C	0101 1100	\	156	9C	1001 1100	220	DC	1101 1100
29	1D	0001 1101	GS	93	5D	0101 1101	]	157	9D	1001 1101	221	DD	1101 1101
30	1E	0001 1110	RS	94	5E	0101 1110	^	158	9E	1001 1110	222	DE	1101 1110
31	1F	0001 1111	US	95	5F	0101 1111	—	159	9F	1001 1111	223	DF	1101 1111
32	20	0010 0000	SPACE	96	60	0110 0000	`	160	A0	1010 0000	224	E0	1110 0000
33	21	0010 0001	!	97	61	0110 0001	a	161	A1	1010 0001	225	E1	1110 0001
34	22	0010 0010	"	98	62	0110 0010	b	162	A2	1010 0010	226	E2	1110 0010
35	23	0010 0011	#	99	63	0110 0011	c	163	A3	1010 0011	227	E3	1110 0011
36	24	0010 0100	$	100	64	0110 0100	d	164	A4	1010 0100	228	E4	1110 0100
37	25	0010 0101	%	101	65	0110 0101	e	165	A5	1010 0101	229	E5	1110 0101
38	26	0010 0110	&	102	66	0110 0110	f	166	A6	1010 0110	230	E6	1110 0110
39	27	0010 0111	'	103	67	0110 0111	g	167	A7	1010 0111	231	E7	1110 0111
40	28	0010 1000	(	104	68	0110 1000	h	168	A8	1010 1000	232	E8	1110 1000
41	29	0010 1001	)	105	69	0110 1001	i	169	A9	1010 1001	233	E9	1110 1001
42	2A	0010 1010	*	106	6A	0110 1010	j	170	AA	1010 1010	234	EA	1110 1010
43	2B	0010 1011	+	107	6B	0110 1011	k	171	AB	1010 1011	235	EB	1110 1011
44	2C	0010 1100	,	108	6C	0110 1100	l	172	AC	1010 1100	236	EC	1110 1100
45	2D	0010 1101	-	109	6D	0110 1101	m	173	AD	1010 1101	237	ED	1110 1101
46	2E	0010 1110	.	110	6E	0110 1110	n	174	AE	1010 1110	238	EE	1110 1110
47	2F	0010 1111	/	111	6F	0110 1111	o	175	AF	1010 1111	239	EF	1110 1111
48	30	0011 0000	0	112	70	0111 0000	p	176	B0	1011 0000	240	F0	1111 0000
49	31	0011 0001	1	113	71	0111 0001	q	177	B1	1011 0001	241	F1	1111 0001
50	32	0011 0010	2	114	72	0111 0010	r	178	B2	1011 0010	242	F2	1111 0010
51	33	0011 0011	3	115	73	0111 0011	s	179	B3	1011 0011	243	F3	1111 0011
52	34	0011 0100	4	116	74	0111 0100	t	180	B4	1011 0100	244	F4	1111 0100
53	35	0011 0101	5	117	75	0111 0101	u	181	B5	1011 0101	245	F5	1111 0101
54	36	0011 0110	6	118	76	0111 0110	v	182	B6	1011 0110	246	F6	1111 0110
55	37	0011 0111	7	119	77	0111 0111	w	183	B7	1011 0111	247	F7	1111 0111
56	38	0011 1000	8	120	78	0111 1000	x	184	B8	1011 1000	248	F8	1111 1000
57	39	0011 1001	9	121	79	0111 1001	y	185	B9	1011 1001	249	F9	1111 1001
58	3A	0011 1010	:	122	7A	0111 1010	z	186	BA	1011 1010	250	FA	1111 1010
59	3B	0011 1011	;	123	7B	0111 1011	{	187	BB	1011 1011	251	FB	1111 1011
60	3C	0011 1100	<	124	7C	0111 1100	|	188	BC	1011 1100	252	FC	1111 1100
61	3D	0011 1101	=	125	7D	0111 1101	}	189	BD	1011 1101	253	FD	1111 1101
62	3E	0011 1110	>	126	7E	0111 1110	~	190	BE	1011 1110	254	FE	1111 1110
63	3F	0011 1111	?	127	7F	0111 1111	DEL	191	BF	1011 1111	255	FF	1111 1111

B. NATIONAL KEYBOARDS LAYOUTS AND CODES

ABOUT THIS APPENDIX

This chapter provides the keyboard layouts for all keyboards supported under MS-DOS on the M20.

CONTENTS

NATIONAL KEYBOARDS LAYOUTS AND CODES

Each of the national keyboards is described by a figure that illustrates the keyboard layout, and a table that relates the key or key combination struck to the code generated. That is, the table shows the 16 bit code (in hexadecimal) generated for each key whether struck on its own, or in conjunction with the /SHIFT/, /CTRL/, or /COMMAND/ key.

The first eight bits (first two hexadecimal digits) are returned in the AH register. The second eight bits (the third and fourth hexadecimal digits) contain the ASCII code and are returned in the AL register.

The keyboards are each toured in the same physical sequence, in ascending order of raw key codes of the key struck on its own. The raw key code is the code that is immediately generated when a key or key combination is struck before it is translated by system tables. That is, it depends entirely on the physical position of the key. The raw key codes are shown in the following figure.

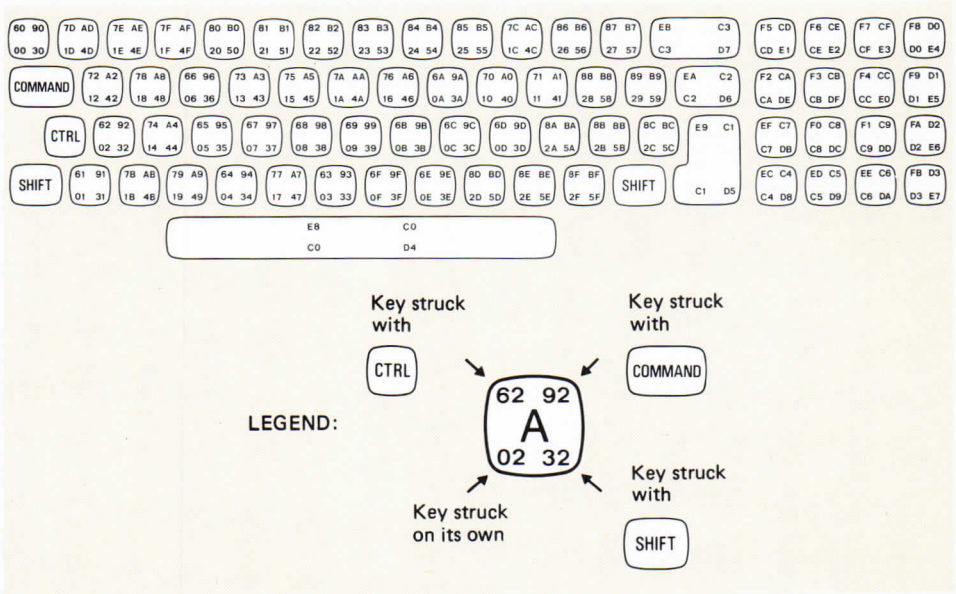


Fig. B-1 Raw Key Codes

Remark

The shift-lock and cursor-lock functions are enabled by the bottom right-hand key (/?// on the USA ASCII keyboard) struck in conjunction with the /COMMAND/ or /CTRL/ key respectively. Where:

shift-lock — infers that all alpha keys on the alphanumeric keypad subsequently take on shifted values. That is, an alpha key struck on its own will generate an upper case character. Moreover, an alpha key struck in conjunction with the /SHIFT/ key will generate a lower case character. The shift-lock is disabled by re-entering /COMMAND/ /?//

cursor-lock — infers that all keys on the numeric keypad subsequently take on shifted values. That is, if such a key is struck on its own it will generate the code normally associated with pressing the same key in conjunction with the /SHIFT/ key. Moreover, if such a key is pressed in conjunction with the shift key, it will generate the unshifted value. The cursor lock is disabled by re-entering /CTRL/ /?//.

DENMARK KEYBOARD

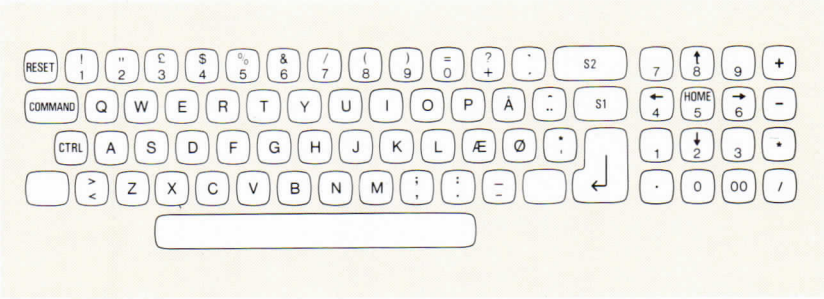


Fig. B-2 Denmark Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	<	013C	313E	611C	7F00
02	a	0261	3241	6201	1E00
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	m	0E6D	3E4D	6E0D	3200
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	q	1271	4251	7211	1000
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	w	1877	4857	7817	1100

19	x	1978	4958	7918	2D00
1A	y	1A79	4A59	7A19	1500
1B	z	1B7A	4B5A	7B1A	2C00
1C	0	1C30	4C3D	4400	5D00
1D	1	1D31	4D21	3B00	5400
1E	2	1E32	4E22	3C00	5500
1F	3	1F33	4F23	3D00	5600
20	4	2034	5024	3E00	5700
21	5	2135	5125	3F00	5800
22	6	2236	5226	4000	5900
23	7	2337	532F	4100	5A00
24	8	2438	5428	4200	5B00
25	9	2539	5529	4300	5C00
26	+	262B	563F	861F	6800
27	/	2740	5760	871E	6900
28	å	287D	585D	0300	6A00
29	··	297E	595E	891B	6B00
2A	æ	2A7B	5A5B	7D00	6C00
2B	Ø	2B7C	5B5C	7E00	6D00
2C	'	2C27	5C2A	8C1D	6E00
2D	,	2D2C	5D3B	8D1C	6F00
2E	.	2E2E	5E3A	7100	7000
2F	-	2F2D	5F5F	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

FRANCE KEYBOARD

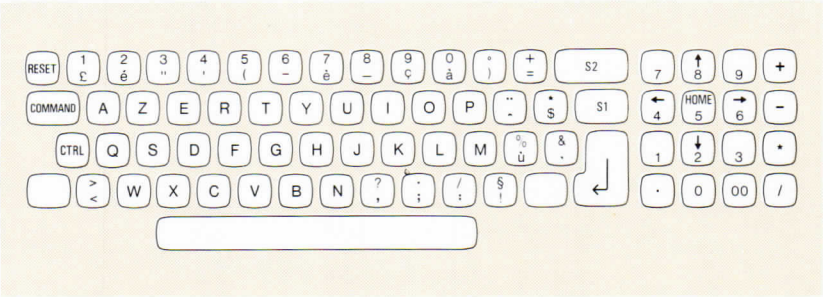


Fig. B-3 France Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	<	013C	313E	611C	7F00
02	q	0271	3251	6211	1000
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	,	0E2C	3E3F	6E1C	6C00
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	a	1261	4241	7201	1E00
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	z	187A	485A	781A	2C00

19	x	1978	4958	7918	2D00
1A	y	1A79	4A59	7A19	1500
1B	w	1B77	4B57	7B17	1100
1C	à	1C40	4C30	4400	5D00
1D	£	1D23	4D31	3B00	5400
1E	é	1E7B	4E32	3C00	5500
1F	"	1F22	4F33	3D00	5600
20	'	2027	5034	3E00	5700
21	(	2128	5135	3F00	5800
22	-	222D	5236	4000	5900
23	è	237D	5337	4100	5A00
24	_	245F	5438	4200	5B00
25	ç	255C	5539	4300	5C00
26	)	2629	565B	861F	6800
27	=	273D	572B	871E	6900
28	^	285E	587E	0300	6A00
29	$	2924	592A	891B	6B00
2A	m	2A6D	5A4D	8A0D	3200
2B	ù	2B7C	5B25	7E00	6D00
2C	\	2C60	5C26	8C1D	6E00
2D	;	2D3B	5D2E	7D00	6F00
2E	:	2E3A	5E2F	7100	7000
2F	!	2F21	5F5D	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

GERMANY KEYBOARD

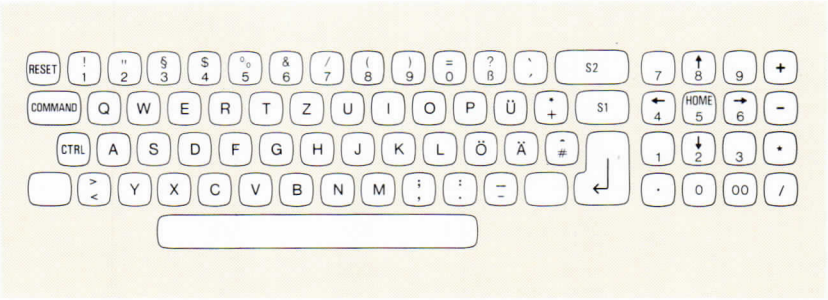


Fig. B-4 Germany Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	<	013C	313E	611C	7F00
02	a	0261	3241	6201	1E00
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	m	0E6D	3E4D	6E0D	3200
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	q	1271	4251	7211	1000
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	w	1877	4857	7817	1100

19	x	1978	4958	7918	2D00
1A	z	1A7A	4A5A	7A1A	1500
1B	y	1B79	4B59	7B19	2C00
1C	0	1C30	4C5F	4400	5D00
1D	1	1D31	4D21	3B00	5400
1E	2	1E32	4E22	3C00	5500
1F	3	1F33	4F23	3D00	5600
20	4	2034	5024	3E00	5700
21	5	2135	5125	3F00	5800
22	6	2236	5226	4000	5900
23	7	2337	5327	4100	5A00
24	8	2438	5428	4200	5B00
25	9	2539	5529	4300	5C00
26	ß	267E	563F	861F	6800
27	'	2727	5760	871E	6900
28	ü	287D	585D	0300	6A00
29	+	292B	592A	891B	6B00
2A	ö	2A7C	5A5C	7D00	6C00
2B	ä	2B7B	5B5B	7E00	6D00
2C	£	2C23	5C5E	8C1D	6E00
2D	,	2D2C	5D3B	8D1C	6F00
2E	.	2E2E	5E3A	7100	7000
2F	-	2F2D	5F5F	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

GREAT BRITAIN KEYBOARD

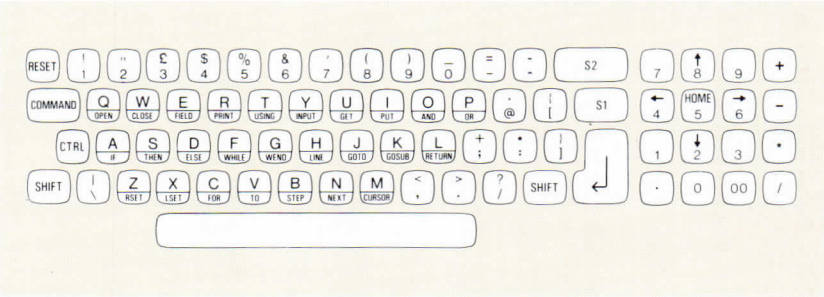


Fig. B-5 Great Britain Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	\	015C	317C	611C	7F00
02	a	0261	3241	6201	1E00
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	m	0E6D	3E4D	6E0D	3200
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	q	1271	4251	7211	1000
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	w	1877	4857	7817	1100

19	x	1978	4958	7918	2D00
1A	y	1A79	4A59	7A19	1500
1B	z	1B7A	4B5A	7B1A	2C00
1C	0	1C30	4C5F	4400	5D00
1D	1	1D31	4D21	3B00	5400
1E	2	1E32	4E22	3C00	5500
1F	3	1F33	4F23	3D00	5600
20	4	2034	5024	3E00	5700
21	5	2135	5125	3F00	5800
22	6	2236	5226	4000	5900
23	7	2337	5327	4100	5A00
24	8	2438	5428	4200	5B00
25	9	2539	5529	4300	5C00
26	-	262D	563D	861F	6800
27	^	275E	577E	871E	6900
28	@	2840	5860	0300	6A00
29	[	295B	597B	891B	6B00
2A	;	2A3B	5A2B	7D00	6C00
2B	:	2B3A	5B2A	7E00	6D00
2C	]	2C5D	5C7D	8C1D	6E00
2D	,	2D2C	5D3C	8D1C	6F00
2E	.	2E2E	5E3E	7100	7000
2F	/	2F2F	5F3F	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

ITALY KEYBOARD

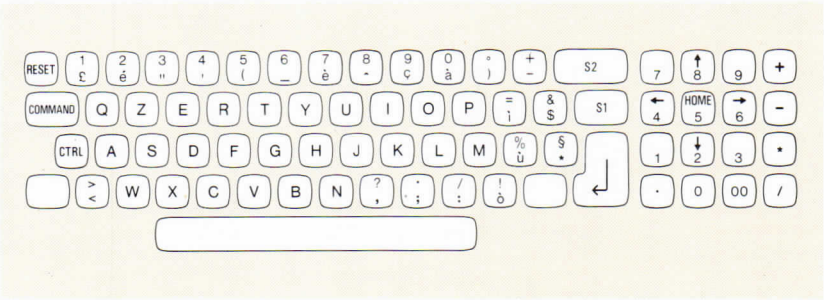


Fig. B-6 Italy Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	<	013C	313E	611C	7F00
02	a	0261	3241	6201	1E00
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	,	0E2C	3E3F	6E1C	6C00
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	q	1271	4251	7211	1000
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	z	187A	485A	781A	2C00

19	x	1978	4958	7918	2D00
1A	y	1A79	4A59	7A19	1500
1B	w	1B77	4B57	7B17	1100
1C	à	1C7B	4C30	4400	5D00
1D	£	1D23	4D31	3B00	5400
1E	é	1E5D	4E32	3C00	5500
1F	"	1F22	4F33	3D00	5600
20	'	2027	5034	3E00	5700
21	(	2128	5135	3F00	5800
22	-	225F	5236	4000	5900
23	è	237D	5337	4100	5A00
24	^	245E	5438	4200	5B00
25	ç	255C	5539	4300	5C00
26	)	2629	565B	861F	6800
27	-	272D	572B	871E	6900
28	ì	287E	583D	0300	6A00
29	$	2924	5926	891B	6B00
2A	m	2A6D	5A4D	8A0D	3200
2B	ù	2B60	5B25	7E00	6D00
2C	*	2C2A	5C40	8C1D	6E00
2D	;	2D3B	5D2E	7D00	6F00
2E	:	2E3A	5E2F	7100	7000
2F	ò	2F7C	5F21	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

NORWAY KEYBOARD

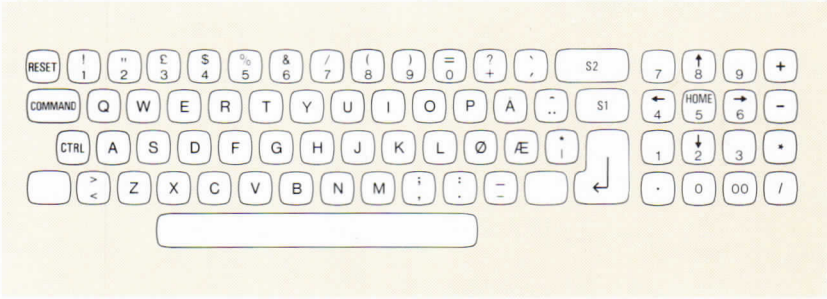


Fig. B-7 Norway Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	<	013C	313E	611C	7F00
02	a	0261	3241	6201	1E00
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	m	0E6D	3E4D	6E0D	3200
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	q	1271	4251	7211	1000
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	w	1877	4857	7817	1100

19	x	1978	4958	7918	2D00
1A	y	1A79	4A59	7A19	1500
1B	z	1B7A	4B5A	7B1A	2C00
1C	0	1C30	4C3D	4400	5D00
1D	1	1D31	4D21	3B00	5400
1E	2	1E32	4E22	3C00	5500
1F	3	1F33	4F23	3D00	5600
20	4	2034	5024	3E00	5700
21	5	2135	5125	3F00	5800
22	6	2236	5226	4000	5900
23	7	2337	532F	4100	5A00
24	8	2438	5428	4200	5B00
25	9	2539	5529	4300	5C00
26	+	262B	563F	861F	6800
27	'	2740	5760	871E	6900
28	&	287D	585D	0300	6A00
29	··	297E	595E	891B	6B00
2A	ø	2A7C	5A5C	7D00	6C00
2B	æ	2B7B	5B5B	7E00	6D00
2C	'	2C27	5C2A	8C1D	6E00
2D	,	2D2C	5D3B	8D1C	6F00
2E	.	2E2E	5E3A	7100	7000
2F	-	2F2D	5F5F	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

PORTUGAL KEYBOARD

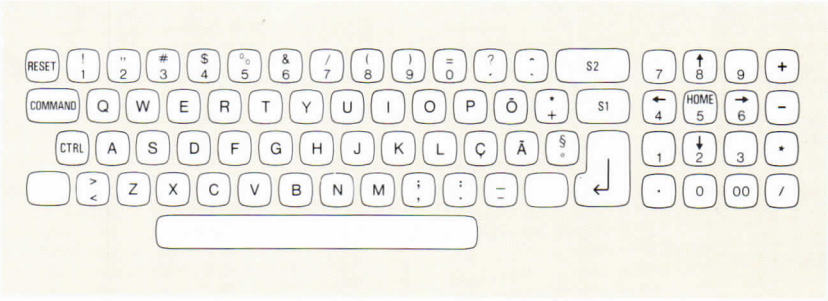


Fig. B-8 Portugal Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	<	013C	313E	611C	7F00
02	a	0261	3241	6201	1E00
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	m	0E6D	3E4D	6E0D	3200
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	q	1271	4251	7211	1000
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	w	1877	4857	7817	1100

19	x	1978	4958	7918	2D00
1A	y	1A79	4A59	7A19	1500
1B	z	1B7A	4B5A	7B1A	2C00
1C	0	1C30	4C3D	4400	5D00
1D	1	1D31	4D21	3B00	5400
1E	2	1E32	4E22	3C00	5500
1F	3	1F33	4F23	3D00	5600
20	4	2034	5024	3E00	5700
21	5	2135	5125	3F00	5800
22	6	2236	5226	4000	5900
23	7	2337	5327	4100	5A00
24	8	2438	5428	4200	5B00
25	9	2539	5529	4300	5C00
26	/	2627	563F	861F	6800
27	/	2760	575E	871E	6900
28	õ	287D	585D	0300	6A00
29	+	292B	592A	891B	6B00
2A	ç	2A7C	5A5C	7D00	6C00
2B	ã	2B7B	5B40	7E00	6D00
2C	o	2C7E	5C2A	8C1D	6E00
2D	,	2D2C	5D3B	8D1C	6F00
2E	.	2E2E	5E3A	7100	7000
2F	-	2F2D	5F5F	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

SPAIN KEYBOARD

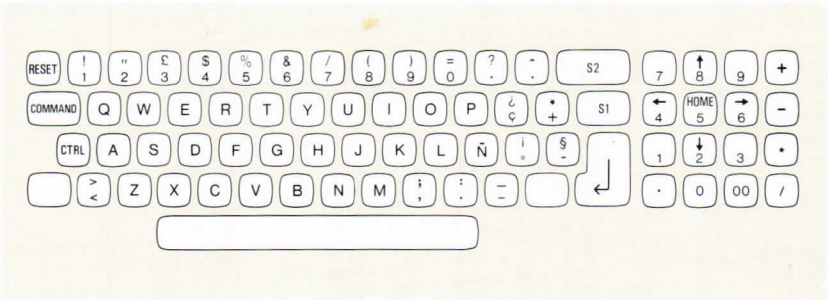


Fig. B-9 Spain Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	<	013C	313E	611C	7F00
02	a	0261	3241	6201	1E00
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	m	0E6D	3E4D	6E0D	3200
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	q	1271	4251	7211	1000
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	w	1877	4857	7817	1100

19	x	1978	4958	7918	2D00
1A	y	1A79	4A59	7A19	1500
1B	z	1B7A	4B5A	7B1A	2C00
1C	0	1C30	4C3D	4400	5D00
1D	1	1D31	4D21	3B00	5400
1E	2	1E32	4E22	3C00	5500
1F	3	1F33	4F23	3D00	5600
20	4	2034	5024	3E00	5700
21	5	2135	5125	3F00	5800
22	6	2236	5226	4000	5900
23	7	2337	5327	4100	5A00
24	8	2438	5428	4200	5B00
25	9	2539	5529	4300	5C00
26	/	2627	563F	861F	6800
27	\	2760	575E	871E	6900
28	ç	287D	585D	0300	6A00
29	+	292B	592A	891B	6B00
2A	ñ	2A7C	5A5C	7D00	6C00
2B	o	2B7B	5B40	7E00	6D00
2C	~	2C7E	5C2A	8C1D	6E00
2D	,	2D2C	5D3B	8D1C	6F00
2E	.	2E2E	5E3A	7100	7000
2F	-	2F2D	5F5F	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

SWEDEN/FINLAND KEYBOARD

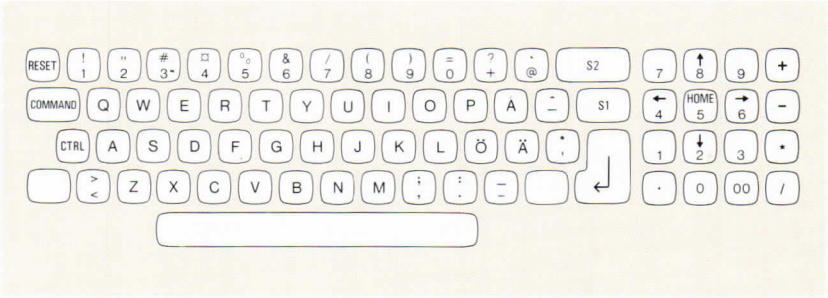


Fig. B-10 Sweden/Finland Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	<	013C	313E	611C	7F00
02	a	0261	3241	6201	1E00
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	m	0E6D	3E4D	6E0D	3200
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	q	1271	4251	7211	1000
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	w	1877	4857	7817	1100

19	x	1978	4958	7918	2D00
1A	y	1A79	4A59	7A19	1500
1B	z	1B7A	4B5A	7B1A	2C00
1C	0	1C30	4C3D	4400	5D00
1D	1	1D31	4D21	3B00	5400
1E	2	1E32	4E22	3C00	5500
1F	3	1F33	4F23	3D00	5600
20	4	2034	5024	3E00	5700
21	5	2135	5125	3F00	5800
22	6	2236	5226	4000	5900
23	7	2337	532F	4100	5A00
24	8	2438	5428	4200	5B00
25	9	2539	5529	4300	5C00
26	+	262B	563F	861F	6800
27	@	2740	5760	871E	6900
28	å	287D	585D	0300	6A00
29	-	297E	595E	891B	6B00
2A	ö	2A7C	5A5C	7D00	6C00
2B	ä	2B7B	5B5B	7E00	6D00
2C	'	2C27	5C2A	8C1D	6E00
2D	,	2D2C	5D3B	8D1C	6F00
2E	.	2E2E	5E3A	7100	7000
2F	-	2F2D	5F5F	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

SWITZERLAND FRENCH KEYBOARD

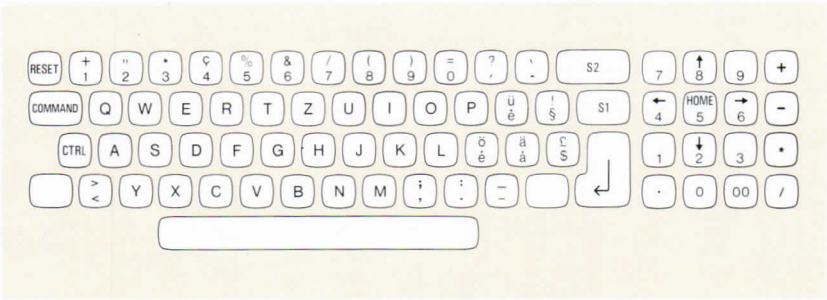


Fig. B-11 Switzerland French Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	<	013C	313E	611C	7F00
02	a	0261	3241	6201	1E00
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	m	0E6D	3E4D	6E0D	3200
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	q	1271	4251	7211	1000
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	w	1877	4857	7817	1100

19	x	1978	4958	7918	2D00
1A	y	1A79	4A59	7A19	1500
1B	z	1B7A	4B5A	7B1A	2C00
1C	0	1C30	4C3D	4400	5D00
1D	1	1D31	4D2B	3B00	5400
1E	2	1E32	4E22	3C00	5500
1F	3	1F33	4F23	3D00	5600
20	4	2034	502A	3E00	5700
21	5	2135	515C	3F00	5800
22	6	2236	5226	4000	5900
23	7	2337	532F	4100	5A00
24	8	2438	5428	4200	5B00
25	9	2539	5529	4300	5C00
26	/	2627	563F	861F	6800
27	^	275E	5760	871E	6900
28	é	285D	587D	0300	6A00
29	§	2940	5921	891B	6B00
2A	è	2A7E	5A7C	7D00	6C00
2B	à	2B5B	5B7B	7E00	6D00
2C	$	2C24	5C23	8C1D	6E00
2D	,	2D2C	5D3B	8D1C	6F00
2E	.	2E2E	5E3A	7100	7000
2F	-	2F2D	5F5F	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

SWITZERLAND GERMAN KEYBOARD

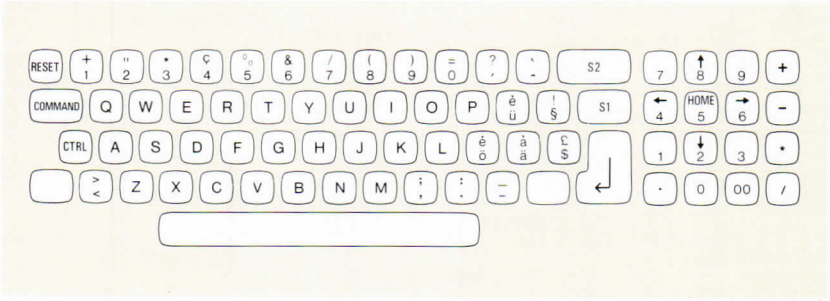


Fig. B-12 Switzerland German Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	<	013C	313E	611C	7F00
02	a	0261	3241	6201	1E00
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	m	0E6D	3E4D	6E0D	3200
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	q	1271	4251	7211	1000
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	w	1877	4857	7817	1100

19	x	1978	4958	7918	2D00
1A	y	1A79	4A59	7A19	1500
1B	z	1B7A	4B5A	7B1A	2C00
1C	0	1C30	4C3D	4400	5D00
1D	1	1D31	4D2B	3B00	5400
1E	2	1E32	4E22	3C00	5500
1F	3	1F33	4F23	3D00	5600
20	4	2034	502A	3E00	5700
21	5	2135	515C	3F00	5800
22	6	2236	5226	4000	5900
23	7	2337	532F	4100	5A00
24	8	2438	5428	4200	5B00
25	9	2539	5529	4300	5C00
26	'	2627	563F	861F	6800
27	^	275E	5760	871E	6900
28	ü	287D	585D	0300	6A00
29	§	2940	5921	891B	6B00
2A	ö	2A7C	5A7E	7D00	6C00
2B	ä	2B7B	5B5B	7E00	6D00
2C	$	2C24	5C23	8C1D	6E00
2D	,	2D2C	5D3B	8D1C	6F00
2E	.	2E2E	5E3A	7100	7000
2F	-	2F2D	5F5F	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

USA ASCII KEYBOARD

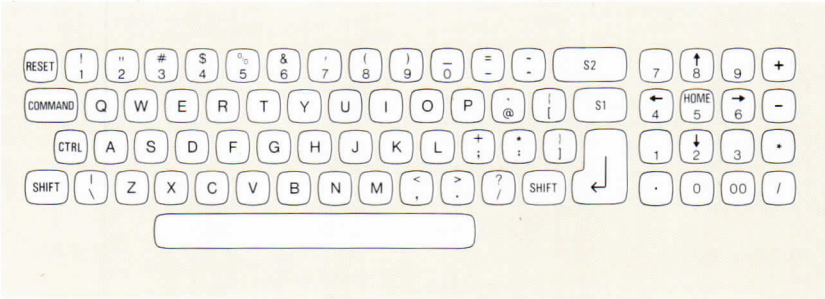


Fig. B-13 USA ASCII Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	001B	301B	trapped	trapped
01	\	015C	317C	611C	7F00
02	a	0261	3241	6201	1E00
03	b	0362	3342	6302	3000
04	c	0463	3443	6403	2E00
05	d	0564	3544	6504	2000
06	e	0665	3645	6605	1200
07	f	0766	3746	6706	2100
08	g	0867	3847	6807	2200
09	h	0968	3948	6908	2300
0A	i	0A69	3A49	6A09	1700
0B	j	0B6A	3B4A	6B0A	2400
0C	k	0C6B	3C4B	6C0B	2500
0D	l	0D6C	3D4C	6D0C	2600
0E	m	0E6D	3E4D	6E0D	3200
0F	n	0F6E	3F4E	6F0E	3100
10	o	106F	404F	700F	1800
11	p	1170	4150	7110	1900
12	q	1271	4251	7211	1000
13	r	1372	4352	7312	1300
14	s	1473	4453	7413	1F00
15	t	1574	4554	7514	1400
16	u	1675	4655	7615	1600
17	v	1776	4756	7716	2F00
18	w	1877	4857	7817	1100

19	x	1978	4958	7918	2D00
1A	y	1A79	4A59	7A19	1500
1B	z	1B7A	4B5A	7B1A	2C00
1C	0	1C30	4C5F	4400	5D00
1D	1	1D31	4D21	3B00	5400
1E	2	1E32	4E22	3C00	5500
1F	3	1F33	4F23	3D00	5600
20	4	2034	5024	3E00	5700
21	5	2135	5125	3F00	5800
22	6	2236	5226	4000	5900
23	7	2337	5327	4100	5A00
24	8	2438	5428	4200	5B00
25	9	2539	5529	4300	5C00
26	-	262D	563D	861F	6800
27	^	275E	577E	871E	6900
28	@	2840	5860	0300	6A00
29	[	295B	597B	891B	6B00
2A	;	2A3B	5A2B	7D00	6C00
2B	:	2B3A	5B2A	7E00	6D00
2C	]	2C5D	5C7D	8C1D	6E00
2D	,	2D2C	5D3C	8D1C	6F00
2E	.	2E2E	5E3E	7100	7000
2F	/	2F2F	5F3F	trapped	trapped
C0	SPACE	C020	D420	E820	C020
C1	↵	C10D	D50D	E90A	C10D
C2	S1	C208	D608	EA7F	C208
C3	S2	C309	D709	0F00	C309

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	5200	D82E	7700	5200
C5	0	5300	D930	6700	5300
C6	00	7200	trapped	7800	7200
C7	1	4F00	DB31	5E00	4F00
C8	2	5000	DC32	5F00	5000
C9	3	5100	DD33	6000	5100
CA	4	4B00	DE34	6100	4B00
CB	5	4700	DF35	6200	4700
CC	6	4D00	E036	6300	4D00
CD	7	4700	E137	6400	4700
CE	8	4800	E238	6500	4800
CF	9	4900	E339	6600	4900
D0	+	7300	E42B	7C00	7300
D1	-	7400	E52D	7B00	7400
D2	*	7500	E62A	7A00	7500
D3	/	7600	E72F	7900	7600

C. ANSI ESCAPE SEQUENCES

ABOUT THIS APPENDIX

This appendix describes the ANSI escape sequences.

CONTENTS

INTRODUCTION

An ANSI escape sequence is a series of characters (beginning with an escape character) that your program can generate to define functions to MS-DOS. Specifically you can affect cursor movement, erase specific areas of the screen and set the graphics mode.

By default the console driver is in ANSI mode, but if you have entered either ADM5 or RAW mode since initialisation, then you should refer to the MODE command in Chapter 7 for details of how to return to ANSI mode.

Remarks

The following notes are general to all escape sequences:

1. Pn represents a numeric parameter. This is a decimal number

2. Ps represents a selective parameter. The parameter is still a decimal integer but is one that must be selected from a list of alternatives

3. Where no parameter is specified, or where zero is entered, a default value is assumed

CURSOR MOVING FUNCTIONS

The following escape sequences affect the position of the cursor on the screen.

MNEMONIC	SEQUENCE	MEANING
CUP (Cursor Position) or HVP (Horizontal and Vertical Position)	ESC Pn ; Pn H ESC Pn ; Pn f	the cursor is moved to the line and column specified by the first and second parameters, respectively. The default values are 1. If no parameters are specified the cursor is moved to the home position

CUU (Cursor Up)	ESC [Pn A	moves the cursor up the screen by the number of rows specified by the parameter. If no parameter is specified one line is assumed. No action is taken if the cursor is already on the top line of the screen
CUD (Cursor Down)	ESC [Pn B	moves the cursor down the screen by the number of rows specified by the parameter. If no parameter is specified one line is assumed. No action is taken if the cursor is already on the bottom line of the screen
CUF (Cursor Forward)	ESC [Pn C	moves the cursor to the right by the number of columns specified by the parameter. If no parameter is specified then one column is assumed. No action is taken if the cursor is already on the right-most column
CUB (Cursor Backward)	ESC [Pn D	moves the cursor left by the number of columns specified by the parameter. If no parameter is specified then one column is assumed. No action is taken if the cursor is already on the left-most column
PSCP or SCP (Save Cursor Position)	ESC [s	the current cursor position is saved. The saved value can subsequently be restored by issuing an RCP (Restore Cursor Position) sequence

PRCP or RCP (Restore Cursor Position)	ESC [u	restores the cursor position to what it was at the time that the console driver received the SCP sequence

ERASING FUNCTIONS

The following sequences affect graphic functions.

MNEMONIC	SEQUENCE	MEANING
ED (Erase Display)	ESC [2 J	the screen is erased and the cursor is moved to the home position
	ESC [J or ESC [0 J	the screen is erased from the cursor position to the end of the display
EL (Erase Line)	ESC [2 K	the entire current line is erased
	ESC [K or ESC [0 K	the current line from the cursor position to the end of the line is erased, including the cursor position itself
	ESC [1 K	the current line from the left hand edge up to the cursor position is erased

D. CONFIGURING MS-DOS

ABOUT THIS APPENDIX

This appendix describes how you can configure MS-DOS to suit the requirements of your application.

CONTENTS

INTRODUCTION

Whenever MS-DOS is initialised, it searches the directory from which it is loaded for a file named CONFIG.SYS. If this file exists, it is read and the configuring commands within it are executed.

You can create or modify the CONFIG.SYS file using the Line Editor EDLIN, and include in it any of the commands described in this appendix.

CONFIGURING COMMANDS

BREAK

Sets or resets the /CTRL/ /C/ break facility.

```
BREAK={ON|OFF}
```

Characteristics

The action of the BREAK command is exactly the same as described in Chapter 7. However, it can only occur once in the CONFIG.SYS file. The state set by including this command in the CONFIG.SYS file can later be reversed by entering a BREAK command with the opposite argument at the keyboard.

The default is OFF.

BUFFERS

Sets the number of buffers to be allocated.

```
BUFFERS=number
```

Where

SYNTAX ELEMENT	MEANING
number	a decimal number in the range 1 to 99 that defines the number of buffers to be allocated. The default is 5

Characteristics

The optimum number of buffers depends on factors such as

- the type of applications most commonly used
- the amount of memory

If your applications perform a lot of random accesses you will probably benefit by using a larger number of buffers. However, beyond a certain number it may take MS-DOS longer to scan the buffers than it would to access a value from disk. For most data base applications between 10 and 20 buffers is a reasonable number.

Conversely, applications that perform mostly sequential accesses gain little by increasing the number of buffers beyond the default of five.

Memory size affects the optimum number of buffers because each additional buffer requires 528 bytes of memory.

The optimum number of buffers can only be determined by trial and error.

DEVICE

Installs a device driver.

DEVICE=[filespec]

Where

SYNTAX ELEMENT	MEANING
filespec	the file containing the device driver

FILES

Defines the number of files that can be open at any one time.

```
FILES=number
```

Where

SYNTAX ELEMENT	MEANING
number	a decimal number in the range 1 to 99 defining the number of files that can be open concurrently. The default is 8

Characteristics

Each additional file above the default value of 8 requires an extra 39 bytes of memory.

SHELL

Loads a top-level command processor.

```
SHELL=filespec
```

Where

SYNTAX ELEMENT	MEANING
filespec	the file containing the top-level command processor to be loaded

E. ERROR MESSAGES

ABOUT THIS APPENDIX

This appendix explains the various messages that can be displayed by MS-DOS.

CONTENTS

ERROR MESSAGES

This appendix lists in alphabetical order the error messages you may encounter when using MS-DOS.

DEVICE ERRORS

A large group of error messages has to do with reading from or writing to devices such as disks or printers. These messages all take the same form, which is:

<type> error reading/writing <device>
Abort, Retry, Ignore?

where <type> represents one of the following:

Bad call format
Bad command
Bad unit
Data
Disk
No paper
Non-DOS disk
Not ready
Read fault
Sector not found
Seek
Write fault
Write protect

When you receive one of these messages, do one of the following:

- enter A for Abort. The system ends the program that requested the read or write.

- enter R for Retry. The system tries the read or write operation again.

- enter I for Ignore. The system ignores the error and attempts to continue the program. (This method may cause loss of data.)

These device error messages are described along with the rest of the error messages in the following table.

MESSAGE TABLE

ERROR MESSAGE	MEANING
Allocation error for file, size adjusted (CHKDSK)	an invalid sector number was found in the file allocation table. The file was truncated at the end of the last valid sector
Ambiguous switch: z (LINK)	the characters specified by z do not uniquely identify a linker parameter. Use more characters from the parameter name
Amount read less than size in header (EXE2BIN)	the file is smaller than indicated in its header. Recompile, or reassemble, and relink the program
An internal failure has occurred (LINK)	an error has occurred in the linker Program. Report the conditions under which the message appeared to your authorised dealer
Attempt to access data outside of segment bounds, possibly bad object module (LINK)	an object file is invalid
Attempted write-protect violation (FORMAT)	the diskette being formatted cannot be written on because it is write-protected. You are prompted to insert a new diskette and press a key to restart formatting

Aux I/O error (MS-DOS)	an input or output error occurred while trying to use the Asynchronous Communications Adapter
Bad call format error (device error)	a device driver was passed an incorrect length request header. Contact your dealer
Bad command error (device error)	a device driver has issued an invalid command to the named device
Bad command or file name (MS-DOS)	the command just entered is not a valid internal command, and a file called command-name.COM or command-name.EXE could not be found on the specified (or default) drive
Bad numeric parameter (LINK)	the value you specified with the /STACK parameter is not a valid numeric constant
Bad or missing Command Interpreter (STARTUP)	the diskette in drive A does not contain a copy of COMMAND.COM, or an error occured while the diskette was being loaded. If system reset fails to solve the problem, copy COMMAND.COM from a backup diskette to the diskette that failed
Bad unit error (device error)	a device driver has been sent an invalid subunit number. Contact your dealer
Batch file missing (MS-DOS)	the batch file that MS-DOS is processing has been deleted or renamed by one of the commands within it

BF (DEBUG)	an invalid flag code setting was specified. Try the Register (RF) command again with the correct code
BP (DEBUG)	more than ten breakpoints were specified for the Go command. Try the Go (G) command again with ten or fewer breakpoints
BR (DEBUG)	an invalid register name was specified. Try the Register (R) command again with a correct register name
Cannot CHDIR to root Processing cannot continue (CHKDSK)	the disk you are checking is bad. Try restarting MS-DOS and RECOVER the disk
Cannot do binary reads from a device (COPY)	you have tried to use the /B switch with the name of a device. Place a /A switch after the device name to do the copy in ASCII mode
Cannot edit .BAK file - rename file (EDLIN)	.BAK files are considered to be backup files, with more up-to-date versions of the files assumed to exist. Therefore, .BAK files cannot be edited
Cannot find file filespec Change diskette <hit any key> (LINK)	the linker could not locate the specified object module on the drive. Insert the diskette with the specified module on it and press any key
Cannot find library libname Enter new drive letter (LINK)	the specified library could not be found on the drive. Enter the letter for the drive the library is on

Cannot nest response file (LINK)	you used @filespec within an automatic response file. Automatic response files cannot be nested
Cannot open list file (LINK)	the directory is full
Cannot open overlay (LINK)	the directory is full
Cannot open response file (LINK)	the automatic response file could not be found
Cannot open temporary file (LINK)	the directory is full and so LINK cannot create the VM.TMP file. Insert a new disk
Copy another (Y/N)? (DISKCOPY)	if you wish to copy another entire diskette, enter Y; DISKCOPY will prompt you to insert the required diskette. If you do not wish to make another copy, enter N
Copy complete (DISKCOPY)	the source diskette contents have been successfully copied to the target diskette
Copying n side(s) (DISKCOPY)	the n will be either 1 or 2, indicating the number of sides that DISKCOPY has successfully read from the first track of the source diskette
Data error (device error)	data could not be read/written correctly because of a faulty disk

DF (DEBUG)	conflicting codes were specified for a single flag. A flag can be changed only once per Register (RF) command
Directory error-file: filename (CHKDSK)	no valid sectors were allocated to the file. The file is removed from the directory
Disk boot failure (MS-DOS)	an error occurred while trying to load MS-DOS into memory. If subsequent attempts to start the system also fail, use a backup MS-DOS diskette
Disk error (device error)	an error has occurred reading from or writing to a disk
Disk full - file write not completed (EDLIN)	the disk you are using does not have enough space to save the entire file. Some of file may be saved on diskette, but the portion in memory not saved is lost
Disk unsuitable for system disk (FORMAT)	a defective track was detected where the DOS files were to reside. The diskette can be used only for data
Diskette not initialised (CHKDSK)	during its analysis of the diskette, CHKDSK could not recognise the directory or file allocation table. The diskette should be formatted again before further use (it may be possible first to copy files to another diskette in order to preserve as much data as possible)

Divide overflow (MS-DOS)	a program attempted to divide a number by zero, or the program had a logic error that caused an internal malfunction
Drive not ready (PRINT)	if this message occurs when PRINT attempts a disk access, PRINT will keep trying until the drive is ready. Any other error causes the current file to be cancelled. An error message would be output on your printer in such a case
Dup record too complex (LINK)	there is a problem with an object module created from an assembler source program. A single DUP requires 1024 bytes before expansion. Debug the source program; then rerun LINK
Duplicate filename or file not found (REN)	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
Enter 2nd file name or drive id (COMP)	enter the filespec of the second of two files to be compared, or just enter the drive designator if the file name is the same as the primary file name
Entry error (EDLIN)	you have entered an EDLIN command correctly. Correct the syntax error on the last command
Error in EXE file (MS-DOS)	an error was detected in the relocation information placed in the file by the LINK program

Error in EXE/HEX file (DEBUG)	the file contained invalid records or characters
Errors found, F parameter not specified Corrections will not be written to disk (CHKDSK)	you must specify the /F switch if you want the errors corrected by CHKDSK
EXE/HEX file cannot be written (DEBUG)	the data would require a backwards conversion that DEBUG does not support
File allocation table bad, drive x Abort, Retry, Ignore? (MS-DOS)	see the message 'Disk error reading drive x'. If this error persists, the disk is unusable and should be formatted again
File cannot be converted (EXE2BIN)	the input file you have specified does not have the correct format for conver-
File cannot be copied into itself (MS-DOS)	a request is made to COPY a file and place the copy (with the same name) on the same diskette as the original. Either change the name given to the copy or put it on another diskette
File creation error (MS-DOS and commands)	an unsuccessful attempt was made to add a new file to the directory. Run CHKDSK to determine if the directory is full, or if some other condition caused the errors
<filename> is cross linked on cluster (CHKDSK)	you have two files cross linked. Make a copy of the file you want to keep, and then delete both files that are cross linked

File not found (EDLIN)	the file you named in the Transfer command does not exist
Filename must be specified (EDLIN)	you did not specify a file name when you started EDLIN
File not found (MS-DOS and commands)	a file named in a command or command parameter does not exist on the diskette in the specified (or default) drive
File size error for file filename (CHKDSK)	the file size shown in the directory is different from the actual size allocated for the file. The size in the directory is adjusted, up or down, to show the actual size (rounded to a 512-byte boundary)
File cross-linked: filename and filename (CHKDSK)	the same data block is allocated to both files. No corrective action is taken automatically, so you must correct the problem. For example, you can: - make copies of both files (use COPY command) - delete the original files (use ERASE command) - review the files for validity and edit as necessary
FIND: Invalid number of parameters (FIND)	you did not specify a string when issuing the FIND command

FIND: Invalid parameter <option-name> (FIND)	you specified an option that does not exist
FIND: Read error in <filename> (FIND)	an error occurred when FIND tried to read the file specified in the command
FIND: Syntax error (FIND)	you entered an illegal string when issuing the FIND command
Fixup offset exceeds field width (LINK)	a machine language processor instruction refers to an address with a short instruction instead of a long instruction. Edit assembler source program and process again
Fixups needed-base segment (hex): (EXE2BIN)	the source (.EXE) file contained information indicating that a load segment is required for the file. Specify the absolute segment address at which the finished module is to be loaded
Format failure (FORMAT)	a disk error was encountered while creating the target diskette. The diskette is unusable
Formatting while copying (DISKCOPY)	the target diskette was found to contain unformatted tracks. DISKCOPY will format the remainder of the target diskette as it copies data. If this message is followed by the message 'Incompatible drive types', you have tried to copy a dual sided diskette to a drive that does not have dual sided capability: processing will end, and the target diskette will not contain any useful data

Incompatible system size (SYS)	the target diskette contained a copy of MS-DOS that is smaller than the one being copied. The system transfer does not take place. A possible solution might be to format a blank diskette (use the FORMAT /S command) and then copy any files to the new diskette
Incorrect DOS version (Commands)	certain commands will only run on versions of MS-DOS that are 2.0 or higher
Input file read error (LINK)	there is probably a bad object file in the command line
Insert disk with batch file and strike any key when ready (MS-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. Processing will continue when you insert the diskette in the appropriate drive and press a key
Insufficient disk space (MS-DOS)	the diskette does not contain enough free space to contain the file being written. if you suspect this condition is invalid, run CHKDSK to determine the status of the diskette
Insufficient memory (Commands)	the amount of available memory is too small to allow the specified command to function

Insufficient room in root directory Erase files in root and repeat CHKDSK (CHKDSK)	CHKDSK cannot function until you delete files in the root directory
Insufficient space on disk (DEBUG)	write command was issued to a diskette that does not have enough free space to hold the data being written
Invalid COMMAND.COM in drive n (MS-DOS)	while trying to reload the command processor, the copy of COMMAND.COM on the diskette was found to be an incorrect version. You are prompted to insert a correct MS-DOS diskette and press a key to continue
Invalid current directory Processing cannot continue (CHKDSK)	CHKDSK has found an error in the disk's current directory. Restart the system and rerun CHKDSK
Invalid date (DATE)	an invalid date or delimiter was entered. The only valid delimiters in a date entry are hyphens(-) and slashes (/)
Invalid drive name or file (EDLIN)	you did not specify a valid drive or file name when starting EDLIN
Invalid drive specification (MS-DOS and commands)	an invalid drive specification was just entered in a command or one of its parameters

Invalid format file (LINK)	a library is in error
Invalid number of parameters (Commands)	you have specified the wrong number of parameters on the command line. Check the syntax of the command you are using
Invalid numeric parameter (LINK)	you have entered a numeric value using not only digits
Invalid object module (LINK)	object module(s) are incorrectly formed or incomplete
Invalid parameter (Commands)	the parameter entered for a command was incorrect. Check the syntax of the command you are using
Invalid switch: z (LINK)	the characters indicated by z do not form a valid linker parameter
Invalid time (TIME)	an invalid time or delimiter was entered. The only valid delimiter is the colon
Invalid Y/N parameter (LINK)	a response to a prompt did not begin with Y,N, or /CR/

Line too long (EDLIN)	upon replacing a string, the replacement causes the line to expand beyond the 253-character limit. The Replace command is ended abnormally. Split the long line into shorter lines; then issue the Replace command again
List output is not assigned to a device (PRINT)	the list device you have specified is invalid. Subsequent attempts will return the same message until a valid de- is specified
Missing file name (REN)	the second of the two required file names is not specified
Must specify destination number (EDLIN)	a destination line number was not specified for a Copy or Move command
No files match d:XXXXXXXX.XXX (PRINT)	a filespec was given for files to add to the print queue, but no files match a specification. NOTE: if there are no files in the queue, no error message will appear
No object modules specified (LINK)	you did not name any object modules in the command line or in response to the prompt. The linker needs some files to link

No paper error (device error)	the named printer is either out of paper or is not switched on
Non-DOS disk error (device error)	the file allocation table contains invalid information and must be reformatted
No room for system on destination disk (SYS)	the destination diskette did not already contain the required reserved space for MS-DOS; therefore, the system cannot be transferred. A possible solution would be to format a blank diskette (use the FORMAT /S command) and then copy any other files to the new diskette
No room in directory for file (EDLIN)	the specified diskette already contains the maximum of 64 (or 112) files. Alternatively, you may have specified an illegal disk drive or an illegal file name
No room in disk directory (DEBUG)	the diskette in the drive specified by the Write command already contains the maximum of 64 (or 112) files
Non-System disk or disk error Replace and strike any key when ready (STARTUP)	there is no entry for IO.SYS or MS-DOS.SYS in the directory; or a disk read error occurred while starting up the system. Insert a MS-DOS diskette in the drive
Not enough room to merge the entire file (EDLIN)	there was not enough room in memory to hold the file you were attempting to transfer. Write some files to disk or delete them

Not found (EDLIN)	either the specified range of lines does not contain the string being searched for by the Replace Text or Search Text commands; or if a search is resumed by replying N to the OK? prompt no further occurrences of the string were found
Not ready error (device error)	the named device is not available for the read/write operation required
Out of space on list file (LINK)	this error usually occurs when there is not enough disk space for the list file
Out of space on run file (LINK)	this error usually occurs when there is not enough disk space for the run file (.EXE)
Out of space on VM.TMP (LINK)	no more disk space remained to expand the VM.TMP file
Printer fault (MS-DOS)	the printer cannot except data because the printer is offline
PRINT queue is empty (PRINT)	there are no files in the print queue
PRINT queue is full (PRINT)	there is room for 10 files in the queue. If you attempt to put more than 10 files in the queue, this message appears on the console

Probable non-DOS disk Continue (Y/N)? (CHKDSK)	the disk you are using is a non-MS-DOS disk. You must indicate whether or not you want CHKDSK to continue processing
Program size or number of segments exceeds capacity of LINKER (LINK)	load module is too large for processing. The total size may not exceed 384K bytes and the number of segments may not exceed 255
Program too big to fit in memory (MS-DOS)	the file containing the external command cannot be loaded because it is larger than the available memory
Requested stack size exceeds 64K (LINK)	specify a size no greater than 64K bytes when the Stack Size: prompt appears
Read error in: filename (Commands)	the command you are using cannot read the entire file specified
Read fault error (device error)	MS-DOS cannot read the requested data from the named device
Sector not found error (device error)	the sector containing the data cannot be located, usually due to a defective area on the disk

Seek error (device error)	the disk drive cannot locate the proper track on the disk
Segment size exceeds 64K (LINK)	LINK attempted to combine identically named segments which resulted in a segment requirement of greater than 64K bytes. The addressing limit is 64K bytes
Stack size exceeds 65535 bytes (LINK)	the size specified for the stack must be less than or equal to 65535 bytes
Symbol defined more than once (LINK)	the Linker found two or more modules that define a single symbol name
Symbol table capacity exceeded (LINK)	very many, very long names were entered. The names exceeded approximately 50K bytes. Use shorter and/or fewer names
Target diskette may be unusable (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
Target diskette write protected Correct, then strike any key (DISKCOPY)	you are trying to produce a copy on a diskette that is write-protected

There was/were x error(s) detected (LINK)	this message is displayed for your information at the end of the link session
Too many external symbols in one module (LINK)	there are too many external symbols in one module. The limit is 256 external symbols per module
Too many groups (LINK)	too many groups are defined. The limit is 10, including DGROUP
Too many libraries specified (LINK)	too many libraries have been named. The limit is eight libraries
Too many public symbols in one module (LINK)	there are too many public symbols in one module. The limit is 1024 public symbols
Too many segments or classes (LINK)	you have too many segments or classes in your source files. The limit is 256 (segments and classes taken together)
Track 0 bad-disk unusable (FORMAT)	track 0 is unusable. Track 0 is where the boot record, file allocation table, and directory must reside
Unable to write BOOT (FORMAT)	the first track of the diskette is bad; the BOOT record could not be written on it

Unexpected end-of-file on library (LINK)	this is caused by an error in the file
Unexpected end-of-file on VM.TMP (LINK)	the diskette containing VM.TMP has been removed
Unrecoverable error in directory Convert directory to file (Y/N)? (CHKDSK)	if you respond Y to this prompt, CHKDSK will convert the bad directory into a file. You can then fix the directory yourself or delete it
Unrecoverable format error on target Target diskette unusable (DISKCOPY)	an unrecoverable error was encountered while formatting the target diskette. The diskette contains no usable data
Unrecoverable read error on source Track xx, side x (DISKCOPY)	four attempts were made to read the the data from the source diskette. DISKCOPY continues copying, but the copy may contain incomplete data
Unrecoverable verify error on target Track xx, side x (DISKCOPY)	four attempts were made to verify the write operation to the target diskette. DISKCOPY continues copying but the copy may contain incomplete data
Unrecoverable write error on target track xx, side x (DISKCOPY)	four attempts were made to write the data to the target diskette. DISKCOPY continues copying, but the copy may contain incomplete data

Unresolved externals: list (LINK)	the external symbols listed were not defined in the modules or library files that you specified. If this error occurs, do not attempt to run the executable file created by the linker
VM.TMP is an illegal file name and has been ignored (LINK)	VM.TMP cannot be used for an object filename. This message is only a warning
Warning: no stack segment (LINK)	none of the object modules specified contain a statement allocating stack space, but you responded with a nonzero entry to the STACK SPACE: prompt
Warning: Segment of absolute or unknown type	there is a bad object module or an attempt has been made to link modules that MS-LINK cannot handle (e.g., an absolute object module)
Write error in TMP file	no more disk space remains to expand the VM.TMP file
Write error on Run file	there is not enough disk space for the Run file
Write fault error (device error)	MS-DOS cannot successfully write data from/to the named device
Write protect error (device error)	you have tried to write data to a disk that is write-protected

X lost clusters found in y chains Convert lost chains to files (Y/N)?	CHKDSK has found lost clusters on your disk. If you respond Y to this prompt, CHKDSK will create a directory entry and a file for you to resolve this problem (files created by CHKDSK are named FILEnnnnnnn). CHKDSK will then display: X bytes disk space freed If you respond N to this prompt, and have not specified the /F switch, CHKDSK frees the clusters and displays: X bytes disk space would be freed

F. GLOSSARY OF TERMS

ABOUT THIS APPENDIX

This appendix defines the terminology used in this book where the meaning differs from general EDP terminology.

CONTENTS

GLOSSARY OF TERMS

The following table defines the non-standard terminology in this manual.

TERM	MEANING
bootable file	a file of a specific format that the bootstrap loader can load into memory to initialise the system
current directory	the directory in which you are working
disk	a diskette or hard disk
diskette	a single or double-sided 5 1/4in. floppy disk
drive specifier	an letter referring to the diskette drive or hard disk drive in question. It may be A - first diskette drive B - second diskette drive C - hard disk drive
editing function keys	the key combinations that invoke the intra-line commands
external command	a command that is not loaded into memory at initialisation. Such commands reside on disk from where they are loaded, executed and purged from memory
hard disk	a 5 1/4in. Winchester disk unit

inter-line commands	the EDLIN commands that operate on entire lines of text
intra-line commands	the commands invoked by the special editing function function keys that perform editing operations within a single line of text
internal command	a command that is embeded in the COMMAND.COM file and resides in memory whenever MS-DOS is booted
logical formatting	deletes any information on a disk and creates an empty directory for that disk. It assumes that the tracks are correctly placed and have the required number of blocks per track
logical reset	a reset of all system parameters and reinitialisation of the system (without performing diagnostic tests). It is caused by pressing /CTRL/ /RESET/, simultaneously. Any AUTOEXEC.BAT file or CONFIG.SYS file is executed
nil parameter	a parameter to a command where the parameter in question is not specified in the command line. The parameter therefore assumes a default value
path name	a sequence of one or more directory names separated by backslashes optionally beginning with a drive specifier and optionally terminating in a file name. It specifies a path through a directory structure to access a file or directory
physical formatting	tracks are physically placed on a disk and an empty directory is created for the disk

physical reset	a system re-initialisation caused by pressing the physical reset button. The subsequent initialisation includes diagnostic tests and a reset of all system parameters. Any AUTOEXEC.BAT file or CONFIG.SYS file is executed
source line	a line of text containing either the last command line entered or the current line in a file being edited. It can be retrieved in whole or part or modified using the special editing function keys
system file	an MS-DOS file that is present on the MS-DOS disk but will not show on a directory listing. There are three such files MSDOS.SYS KB_FONT.SYS IO.SYS
text file	an ASCII file whose records are separated either by CR/LF, or by record separator (RS) characters
volume	the entire contents of a diskette or hard disk
volume label	a name that can be assigned to a disk by the FORMAT command. It will subsequently be displayed in a directory listing, or by the VOL command
wild card character	a special symbol used to represent any single character (?), or any string of characters (*)
working session	the time between booting MS-DOS and the next boot of MS-DOS or switch-off

G. INDEX

ABOUT THIS APPENDIX

This appendix provides an alphabetical index.

CONTENTS

INDEX

NOTICE

Ing. C. Olivetti & C. S.p.A. reserves the right to make improvements in the product described in this manual at any time and without notice.

This material was prepared for the benefit of Olivetti customers. It is recommended that the package be test run before actual use.

Anything in the standard form of the Olivetti Sales Contract to the contrary not withstanding, all software being licensed to Customer is licensed "as is". THERE ARE NO WARRANTIES EXPRESS OR IMPLIED INCLUDING WITHOUT LIMITATION THE IMPLIED WARRANTY OF FITNESS FOR PURPOSE AND OLIVETTI SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, CONSEQUENTIAL OR INCIDENTAL DAMAGES IN CONNECTION WITH SUCH SOFTWARE.

GU Code 3984670 G (0)
Printed in Italy

olivetti

GU Code 3984670 G (0)
Printed in Italy

olivetti