

M20 PERSONAL COMPUTER

PCOS (Professional Computer Operating System)

User Guide



olivetti

PREFACE

This book describes the Professional Computer Operating System (PCOS) and may be used with any M20 model.

It is directed at the user who has some experience of computer programming and is familiar with computing terminology.

The book is made-up of two parts.

Part I comprises Chapters 1 to 12 and contains introductory and operational information. Part I should be read before attempting to use Part II which provides a command reference.

The first three chapters of Part I are introductory. They provide an overview of the M20, both hardware and software.

Chapter 4 describes the notation used throughout the book. It also describes the rules, and defines the terminology used, for entering PCOS commands.

Chapters 5 to 12 provide operational details of how to use PCOS.

Part II comprises Chapter 13 and should be used as a reference. It describes all the PCOS commands in alphabetical order. Each command description includes the command action, a syntax diagram, the characteristics of the command, and examples.

The following are trademarks of Ing. C. Olivetti & C., S.p.A.:
OLICOM, GTL, OLITERM, OLWORD, OLINUM, OLISTAT, OLITUTOR,
OLIENTRY, OLISORT, OLIMASTER.

MULTIPLAN is a registered trademark of MICROSOFT Inc.

MS-DOS is a trademark of MICROSOFT Inc.

CP/M and CP/M-86 are registered trademarks of Digital Research Inc.

CBASIC-86 is a trademark of Digital Research Inc.

Copyright © by Olivetti, 1983,
all rights reserved.

REFERENCE: BASIC Language
Reference Manual
Code 3982430 P

BASIC PCOS Pocket
Reference
Code 3985300 B

Installation and
Operations Guide
Code 3933340 V

I/O with External
Peripherals
Code 3982300 N

DISTRIBUTION: General (G)

EDITION: June 1983

RELEASE: 3.0

PUBLICATION ISSUED BY:

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

CONTENTS

PART I

1. INTRODUCTION		<u>PERIPHERAL INTERFACES</u>	2-12
<u>INTRODUCTION</u>	1-1	EIA RS-232-C SERIAL INTER-	2-12
		FACE	
2. HARDWARE COMPONENTS		CENTRONICS-LIKE PARALLEL	2-12
<u>MAIN UNIT</u>	2-1	IEEE-488 PARALLEL INTER-	2-12
<u>MEMORY</u>	2-2	FACE	
<u>THE VIDEO DISPLAY UNIT</u>	2-3	<u>PRINTERS</u>	2-13
<u>THE KEYBOARD</u>	2-4	3. SOFTWARE COMPONENTS	
LINE TERMINATOR KEYS	2-5	<u>INTRODUCTION</u>	3-1
THE CTRL KEY	2-5	<u>PCOS</u>	3-3
LOCKING THE SHIFT KEY	2-6	MEMORY OPTIMISATION	3-3
LOGICAL RESET	2-6	RESIDENT AND TRANSIENT	3-4
ASSIGNING VALUES TO KEYS	2-6	COMMANDS	
KEYBOARD BUFFER WARNING	2-7	PROGRAMMABLE KEYS	3-4
BUZZER		PROTECTION MECHANISMS	3-4
<u>PHYSICAL RESET</u>	2-7	LINE EDITOR FUNCTIONS	3-5
<u>HARD DISK AND DISKETTE</u>	2-8	REAL-TIME CLOCK	3-5
<u>DRIVES</u>		ROUTING INPUT/OUTPUT	3-5
<u>DISKETTE HANDLING</u>	2-9	USER-DEFINED FONTS	3-6
<u>LABELLING DISKETTES</u>	2-9	CONTROL CHARACTER DISPLAY	3-6
<u>WRITE-PROTECTION</u>	2-10	INITIALISATION FILES	3-6
INSERTING AND REMOVING	2-11	USER-CONFIGURABLE OPERA-	3-6
DISKETTES		TING SYSTEM	

<u>BASIC INTERPRETER</u>	3-7	<u>COMMAND SYNTAX</u>	4-2
<u>VIDEO FILE EDITOR</u>	3-7	<u>COMMAND NAMES AND KEYWORDS</u>	4-3
<u>ASSEMBLER</u>	3-8	<u>PARAMETERS</u>	4-3
<u>PASCAL</u>	3-9	<u>DEFAULT VALUES</u>	4-5
<u>LINKER</u>	3-9	<u>RESIDENT AND TRANSIENT COMMANDS</u>	4-5
<u>PROGRAM DEBUGGER</u>	3-9	<u>COMMAND SEARCH PROCEDURE</u>	4-6
<u>STANDARD INTERFACE HANDLERS</u>	3-9	<u>FILE AND VOLUME IDENTIFI- FIERS</u>	4-6
<u>PCOS COMMAND LIBRARY</u>	3-10	<u>WILD CARDS</u>	4-10
<u>CHANGING ENVIRONMENT COMMANDS</u>	3-10	<u>NO INTERACTION FLAG</u>	4-10
<u>PCOS CONFIGURING COMMANDS</u>	3-10	5. INITIALISATION AND CHANGING ENVIRONMENTS	
<u>SET SYSTEM GLOBAL COMMANDS</u>	3-11	<u>M20 ENVIRONMENTS</u>	5-1
<u>KEYBOARD-RELATED COMMANDS</u>	3-11	<u>PCOS</u>	5-1
<u>VOLUME HANDLING COMMANDS</u>	3-12	<u>BASIC</u>	5-1
<u>FILE HANDLING COMMANDS</u>	3-12	<u>VIDEO FILE EDITOR</u>	5-1
<u>STANDARD INTERFACE HANDLING COMMANDS</u>	3-14	<u>CHANGING ENVIRONMENTS</u>	5-2
<u>PCOS GRAPHIC FACILITY COMMANDS</u>	3-14	<u>MODES OF PCOS</u>	5-5
<u>USER AIDS</u>	3-15	<u>COMMAND MODE</u>	5-5
4. ENTERING A COMMAND		<u>EXECUTION MODE</u>	5-5
<u>NOTATION CONVENTION</u>	4-1	<u>CHANGING MODES</u>	5-5

<u>STANDARD INITIALISATION</u>	5-6	<u>SETTING THE ENVIRONMENT FOR AN RS-232-C COMMUNICATIONS PORT</u>	6-15
<u>INITIALISATION FILES</u>	5-7		
<u>NON-STANDARD INITIALISATION</u>	5-8	<u>SETTING THE PRINTING ENVIRONMENT</u>	6-15
<u>INITIALISATION FOLLOWING A PSAVE COMMAND</u>	5-9	<u>RECONFIGURING THE KEYBOARD LANGUAGE</u>	6-19
<u>INITIALISATION FOLLOWING A PRUN COMMAND</u>	5-9	<u>SAVING A USER-CONFIGURED PCOS</u>	6-21
<u>THE INITIALISATION FLOW-CHART</u>	5-9		
<u>BEGINNING AND ENDING A WORKING SESSION</u>	5-12		
6. CONFIGURING PCOS		7. DEVICE RE-ROUTING	
<u>INTRODUCTION</u>	6-1	<u>INTRODUCTION</u>	7-1
<u>MAKING TRANSIENT COMMANDS RESIDENT</u>	6-1	<u>LOCAL DEVICE RE-ROUTING</u>	7-2
<u>ASSIGNING STRINGS TO KEYS</u>	6-5	<u>GLOBAL DEVICE RE-ROUTING</u>	7-5
<u>CHANGING SYSTEM GLOBAL PARAMETERS</u>	6-7	<u>DEVICE RE-ROUTING FROM A BASIC PROGRAM</u>	7-7
<u>SETTING THE SYSTEM GLOBAL ENVIRONMENT</u>	6-7	8. PROTECTION TOOLS	
<u>DISPLAYING AND MODIFYING DEVICE NAMES</u>	6-11	<u>INTRODUCTION</u>	8-1
<u>MODIFYING THE BASIC ENVIRONMENT</u>	6-14	<u>VOLUME PASSWORDS</u>	8-1
		<u>FILE PASSWORDS</u>	8-2
		<u>WRITE-PROTECTION</u>	8-4
		<u>COPY-PROTECTION</u>	8-4
		<u>BASIC PROGRAM SECURITY</u>	8-5

9. VOLUME HANDLING

<u>FORMATTING AND INITIAL- ISING NEW VOLUMES</u>	9-1
FORMATTING A DISKETTE OR HARD DISK	9-1
INITIALISING A VOLUME	9-3
<u>LISTING A VOLUME</u>	9-4
LISTING A VOLUME USING THE VLIST COMMAND	9-4
LISTING A VOLUME USING THE VQUICK COMMAND	9-5
<u>COPYING VOLUMES</u>	9-6
COPYING VOLUMES ON A DUAL-DRIVE SYSTEM (VOLUMES OF EQUAL SIZE)	9-7
COPYING DISKETTES USING ONLY ONE DRIVE (VOLUMES OF EQUAL SIZE)	9-8
COPYING VOLUMES (OF DIFFERENT SIZES)	9-9
<u>NAMING AND PROTECTING A VOLUME</u>	9-9
WRITE-PROTECTION	9-9
PASSWORD PROTECTION	9-10
NAMING A VOLUME	9-10

ALPHABETISING A VOLUME 9-11

10. FILE HANDLING

<u>CREATING FILES</u>	10-1
CREATING AN EMPTY FILE	10-1
CREATING A FILE BY COPYING	10-2
<u>COPYING FILES</u>	10-2
COPYING FILES ON A DUAL DRIVE SYSTEM	10-2
COPYING FILES USING ONE DRIVE	10-5
<u>LISTING FILES</u>	10-6
<u>PROTECTING FILES</u>	10-8
PASSWORD PROTECTION	10-8
WRITE-PROTECTION	10-9
FREEDING UNUSED FILE BLOCKS	10-10
<u>DELETING AND RECOVERING FILES</u>	10-14
DELETING FILES	10-14
RECOVERING DELETED FILES	10-15
<u>RENAMING FILES</u>	10-15

11. PCOS GRAPHIC AND CONSOLE-RELATED FACILITIES

<u>INTRODUCTION</u>	11-1
<u>DISPLAYING LABELS</u>	11-1
<u>PRINTING THE SCREEN IMAGE</u>	11-3
<u>USING THE SPRINT AND LSCREEN COMMANDS</u>	11-4
<u>ENTERING CHARACTERS AT THE KEYBOARD</u>	11-5
RAW KEY CODES	11-5
ASCII TABLES	11-6
INTERPRETATION OF ASCII TABLE OUTPUT	11-8
USING THE PKEY COMMAND	11-8
<u>CREATING USER-DEFINED FONTS</u>	11-10
<u>CREATING A FONT MATRIX USING THE RFONT COMMAND</u>	11-11
<u>THE FONT MATRIX FILE</u>	11-11
<u>USING THE WFONT COMMAND</u>	11-14
<u>DISPLAYING CONTROL CHARACTERS</u>	11-15
<u>THE LINE TERMINATION KEYS</u>	11-19

12. VIDEO FILE EDITOR

<u>INTRODUCTION</u>	12-1
THE DISPLAY	12-1
THE KEYBOARD	12-2
<u>HOW TO INVOKE THE VIDEO FILE EDITOR</u>	12-4
EDIT.CMD	12-4
<u>GENERAL EDITING FUNCTION KEYS</u>	12-6
<u>WINDOW MOVING FUNCTION KEYS</u>	12-12
<u>EXITING AND SAVING FUNCTION KEYS</u>	12-14
<u>COMMANDS AND SEARCHING</u>	12-15
STRING SEARCHES	12-15
COMMANDS	12-16

PART II

13. PCOS COMMANDS		FUNPROT.CMD	13-40
BASIC.CMD	13-1	FWPROT.CMD	13-42
BKEYBOARD.BAS	13-2	HELP.BAS	13-44
BVOLUME.SAV	13-3	IEEE.SAV	13-44
CI.SAV	13-8	LABEL.CMD	13-45
CKEY.CMD	13-9	LSCREEN.CMD	13-50
COMMANDS.BAS	13-14	LTERM (ALWAYS RESIDENT)	13-52
DCONFIG.CMD	13-15	PKEY.CMD	13-53
EDIT.CMD	13-18	PLOAD (ALWAYS RESIDENT)	13-57
EPRINT.SAV	13-18	PRUN.CMD	13-59
ERROR.BAS	13-20	PSAVE.CMD	13-61
FCOPY.CMD	13-20	PUNLOAD (ALWAYS RESIDENT)	13-63
FDEPASS.CMD	13-26	RFONT.CMD	13-65
FFREE.CMD	13-28	RKILL.CMD	13-67
FKILL.CMD	13-30	RS232.SAV	13-69
FLIST.CMD	13-31	SBASIC.CMD	13-69
FMOVE.CMD	13-33	SCOMM.CMD	13-72
FNEW.CMD	13-35	SDEVICE.CMD	13-73
FPASS.CMD	13.37	SFORM.CMD	13.75
FRENAME.CMD	13-39		

SLANG.CMD	13-78
SPRINT.CMD	13-80
SSYS.CMD	13-82
VALPHA.CMD	13-86
VCOPY.CMD	13-88
VDEPASS.CMD	13-90
VFORMAT.CMD	13-91
VLIST.CMD	13-94
VMOVE.SAV	13-96
VNEW.CMD	13-98
VPASS.CMD	13-100
VQUICK.CMD	13-102
VRENAME.CMD	13-104
VVERIFY.CMD	13-106
WFONT.CMD	13-109

A. ASCII CODE		USA ASCII KEYBOARD	B-42
<u>ASCII CODE</u>	A-1	USA ASCII + BASIC KEYBOARD	B-45
B. NATIONAL KEYBOARDS		YUGOSLAVIA KEYBOARD	B-48
<u>ASCII CHARACTER EQUIVALENCES</u>	B-1	C. HARD DISK AND DISKETTE CHARACTERISTICS	
<u>NATIONAL KEYBOARD LAYOUTS AND CODES</u>	B-1	<u>THE HARD DISK UNIT</u>	C-1
DENMARK KEYBOARD	B-3	CHARACTERISTICS	C-2
FRANCE KEYBOARD	B-6	<u>DISKETTES</u>	C-3
GERMANY (ORIGINAL) KEYBOARD	B-9	CHARACTERISTICS	C-4
GERMANY (WEST) KEYBOARD	B-12	D. DIAGNOSTIC/BOOTSTRAP ERROR MESSAGES	
GREAT BRITAIN KEYBOARD	B-15	<u>DIAGNOSTIC ERROR MESSAGES</u>	D-1
GREECE KEYBOARD	B-18	<u>BOOTSTRAP ERROR MESSAGES</u>	D-3
ITALY KEYBOARD	B-21	E. PCOS AND BASIC ERROR MESSAGES	
NORWAY KEYBOARD	B-24	<u>INTRODUCTION</u>	E-1
PORTUGAL KEYBOARD	B-27	<u>PCOS AND BASIC ERRORS</u>	E-1
SPAIN KEYBOARD	B-30	F. GET/PUT CONVERSION - PCOS 1-3 TO PCOS 2-0/3-0	
SWEDEN/FINLAND KEYBOARD	B-33	<u>GET/PUT CONVERSION</u>	F-1
SWITZERLAND FRENCH KEYBOARD	B-36	GETCONV.BAS	F-1
SWITZERLAND GERMAN KEYBOARD	B-39		

GLOSSARY OF TERMS

G. GLOSSARY OF TERMS G-1

H. COMMAND INDEX

COMMAND KEYWORD INDEX H-1

PART I

1. INTRODUCTION

ABOUT THIS CHAPTER

This chapter provides a general introduction to the M20.

CONTENTS

<u>INTRODUCTION</u>	1-1
---------------------	-----

INTRODUCTION

The Olivetti Model 20 (M20) is a stand-alone system designed for professional use as a problem-solving tool. It has the versatility to help the businessman, the scientist, the student and the technician to process information quickly and accurately.

Processing power is provided by a 16-bit Zilog Z8001 microprocessor in conjunction with read only and random access memory. Bulk storage is provided by 5 1/4 in. floppy disks and (optionally) a 5 1/4 in. hard disk. A keyboard and alphanumeric/graphic video (in either its monochromatic or colour version) serve for the user interface, enabling commands to be entered, prompts to be displayed, etc. Access to a range of printers, other peripherals and other computers is made possible via serial and parallel interfaces.

The whole is managed by the Professional Computer Operating System (PCOS) thus providing an environment for a set of programming tools that enable you to develop and run application programs. Programming facilities include: an extensive BASIC Interpreter, an (optional) Assembler package comprising assembler and program debugger, an (optional) PASCAL compiler, and a linker. Moreover, the Video File Editor enables programs written in BASIC, Assembler or PASCAL to be created and modified.

The PCOS command library comprises resident and transient commands. Optimum use of memory is assured by the use of transient commands which are automatically removed from memory when no longer required. However, the PCOS command library contains a group of commands that enable transient commands to be made resident. A further group of commands enables you to set global parameters as required. Using these two groups you can tailor the operating system to suit your specific needs.

Further functional groups of commands facilitate volume and file handling (including protection mechanisms), keyboard handling, standard interface handling, graphic facilities and a set of user aids.

2. HARDWARE COMPONENTS

ABOUT THIS CHAPTER

This chapter describes the various hardware components that make up the M20. They are discussed from the viewpoint of the function performed, the options available and the physical controls.

CONTENTS

<u>MAIN UNIT</u>	2-1	<u>DISKETTE HANDLING</u>	2-9
<u>MEMORY</u>	2-2	<u>LABELLING DISKETTES</u>	2-9
<u>THE VIDEO DISPLAY UNIT (VDU)</u>	2-3	<u>WRITE-PROTECTION</u>	2-10
<u>THE KEYBOARD</u>	2-4	INSERTING AND REMOVING DISKETTES	2-11
LINE TERMINATOR KEYS	2-5	<u>PERIPHERAL INTERFACES</u>	2-12
THE CTRL KEY	2-5	EIA RS-232-C SERIAL INTERFACE	2-12
LOCKING THE SHIFT KEY	2-6	CENTRONICS-LIKE PARALLEL INTERFACE	2-12
LOGICAL RESET	2-6	IEEE-488 PARALLEL INTERFACE	2-12
ASSIGNING VALUES TO KEYS	2-6	<u>PRINTERS</u>	2-13
KEYBOARD BUFFER WARNING BUZZER	2-7		
<u>PHYSICAL RESET</u>	2-7		
<u>HARD DISK AND DISKETTE DRIVES</u>	2-8		

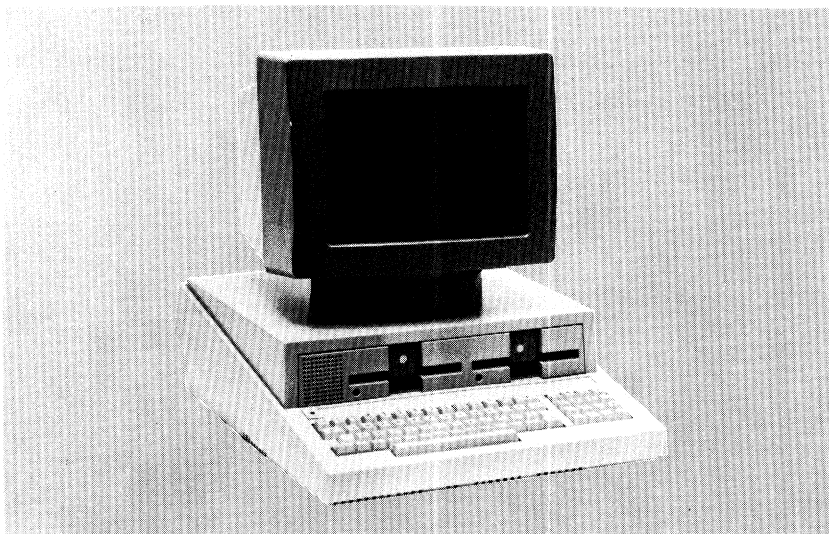
MAIN UNIT

Fig. 2-1 The M20

Figure 2-1 shows the location of the major physical components of the M20. It comprises a video display unit (VDU) and a main unit.

The main unit houses the following components:

- the keyboard, comprising a standard alphanumeric keypad and an additional 16-key numeric keypad for the rapid entry of numerical data and arithmetic functions
- standard 5 1/4 in. floppy disk (diskette) drive and/or a hard disk unit
- the central processing unit (CPU); a Z8001 microprocessor
- read-only memory (ROM)
- random access memory (RAM)

The back panel of the main unit is shown in Figure 2-2.

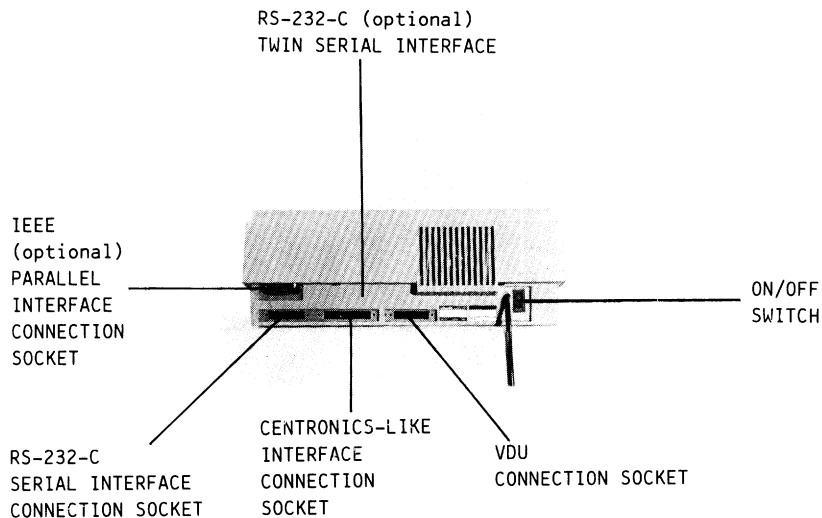


Fig. 2-2 The Back Panel

Figure 2-2 shows the location of the ON/OFF switch and the connection sockets. The latter provide connection points for the video and for peripherals utilising the standard interfaces.

MEMORY

The amount of memory that your M20 has depends on the number and type of memory expansion boards installed. The minimum configuration is one main memory board (the mother board) of 128K. Up to three 32K memory expansion boards or up to three 128K memory expansion boards may be added. 32K and 128K boards cannot be present in the same system. Four-colour systems require at least one memory expansion board (either 32K or 128K); eight-colour systems require at least two.

The mother board can contain up to four 16K blocks of read-only memory (ROM). The implemented version has 8K of ROM which contains self-testing diagnostics that are run on power-up, and the bootstrap loader. One block of RAM (16K) is used for the video bit map. The remaining seven blocks of RAM are available for the operating system, application packages, user programs, etc.

Each 32K memory expansion board adds two 16K blocks of RAM. For a four-colour system one block of the first expansion board is used for the video bit map, while the rest are added to the system memory. The video bit map for an eight-colour system requires one 16K block from each of

the first two expansion boards.

Each 128K memory expansion board adds eight blocks of RAM. Just as with 32K expansion boards, a four-colour system requires one 16K block from the first expansion board for the video bit map, and an eight-colour system requires one 16K block from each of the first two expansion boards. The remaining blocks are added to the system memory.

The memory is logically divided into segments. Each segment can contain up to 4 blocks. This is due to the architectural design of the CPU to enable different software components to occupy distinct areas of memory.

THE VIDEO DISPLAY UNIT (VDU)

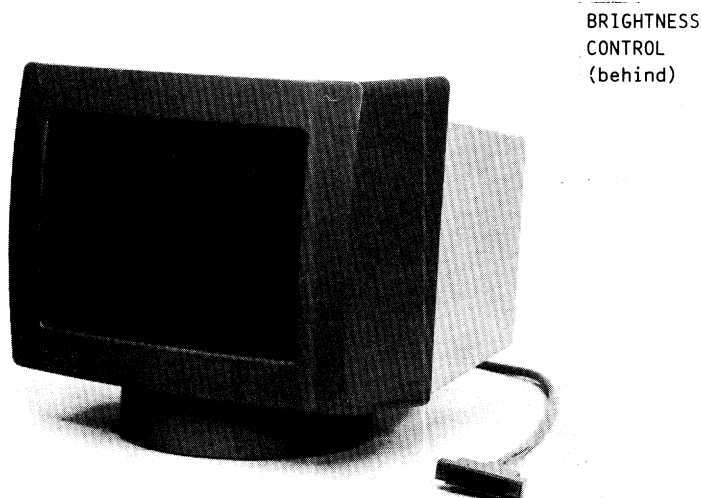


Fig. 2-3 The VDU

There are two types of VDU available with the M20:

- black and white; providing regular and reverse video
- eight colour; supporting the display of up to eight colours: black, red, green, yellow, blue, magenta, cyan and white. There are, however, two implementations of this VDU: one in which a maximum of four of these eight colours can be selected to be present at any one time; or one in which all eight can be displayed at once. This depends on the hardware present in the main unit.

Either VDU enables you to select either a 512 x 256 or a 480 x 256 pixel display; that is, 256 scanlines of either 512 or 480 pixels, where the term "pixel" is a contraction of "picture element" and "scanline" is a row of pixels.

The VDU can show up to 16 lines of 64 characters each, or 25 lines of 80 characters each. It can also display graphic images.

The VDU has a circular base on which it can be rotated and tilted to a viewing position which is comfortable for you. You can also alter the brightness of the images on the screen. This is done by adjusting the brightness control thumbwheel situated behind the top of the screen and to the right, as indicated in Figure 2-3.

THE KEYBOARD

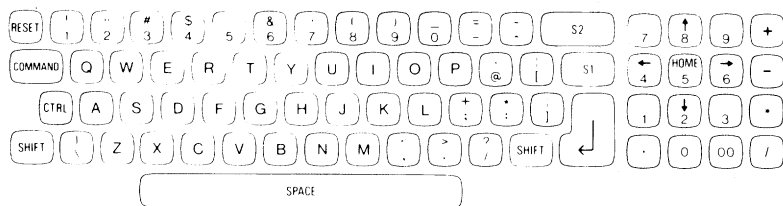


Fig. 2-4 The Keyboard

The keyboard shown in Figure 2-4 is the USA ASCII keyboard. Your keyboard may be that of another country, in which case the keyboard layout will be different. If so, you will find it described in Appendix B. The following description, however, applies to all keyboards.

The keyboard is divided into two sections: one for the entry of alphabetic, numeric, and control characters; the other - the numeric keypad - for rapid entry of numeric data.

LINE TERMINATOR KEYS

The `/↵/`, `/S1/` and `/S2/` keys often serve the same function. Any one of these keys can be used to complete a command, a program statement, a data entry request, or a request for an immediate calculation.

For brevity, these keys will subsequently be referred to as `/CR/`.

In a BASIC program, however, you can test to determine which of the three `/CR/` keys was used in response to a data entry request. For some applications this is a valuable feature - see the `LTERM` command.

THE CTRL KEY

The `/CTRL/` key is always used in conjunction with other keys to add other meanings to existing keys.

Break

`/CTRL/ /C/` activates the break facility and can be used to cancel a line you are currently entering, or it can also be used to terminate most system activities. When `/CTRL/ /C/` is used, the characters " C" appear on the screen and the PCOS prompt and the cursor move to the next line.

Hiding What You Enter

`/CTRL/ /G/` suppresses the display of subsequently entered characters. Thus you can enter some secret data or a password. Hide mode, as this feature is termed, remains in effect until either `/CR/` is pressed or `/CTRL/ /G/` is pressed again.

Deleting Characters and Correcting Errors

`/CTRL/ /H/` performs a backspace function. That is, it deletes the last character entered and moves the cursor one position to the left. In PCOS, you use this facility to correct any error you spot in a line before you have pressed `/CR/`. Simply delete the characters back to the point of the error, and then re-enter the rest of the line correctly.

8-Character Tab

`/CTRL/ /I/` advances the cursor to the next eight-character tab position on the screen.

Stopping and Starting a Listing

`/CTRL/ /S/` suspends the display of a text listing. To resume the listing after scanning the screen for the information you need, press any key.

LOCKING THE SHIFT KEY

The `/COMMAND/` key is used with the bottom right-most key (`/?//` on the USA ASCII keyboard) to provide a "shift lock" for the letters A-Z. After you press `/COMMAND/` with `/?//`, all letters subsequently keyed-in appear as upper-case letters. Furthermore, when the `/SHIFT/` key is used, subsequently keyed characters appear in lower-case. The shift lock stays effective until you press `/COMMAND/ /?//` again.

LOGICAL RESET

The key combination `/CTRL/ /RESET/` causes a logical reset of the system. This re-initialises the system as described in chapter 5.

ASSIGNING VALUES TO KEYS

All of the keys of the M20 are "programmable", with the exception of the `/CTRL/`, `/COMMAND/` and `/SHIFT/` keys, and can be assigned values other than those shown on the keytops. These keys can be assigned a command name, an arithmetic expression, a numeric value, an algorithm, or any string of characters you may find useful to have at a single key-stroke. Assignment is made by means of the `PKEY` or `CKEY` commands.

A "template" comes with the M20 to help you remember what values you have assigned to the keys. The template fits into the slot just above the keys.

KEYBOARD BUFFER WARNING BUZZER

When data is entered at the keyboard it is stored in a buffer which is subsequently read. In rare situations, however, it is possible that the buffer will become full; that is, data has been entered faster than it is read. In such situations a buzzer will be heard (when the buffer contains 56 characters), warning you that input may be lost if you continue to key-in data.

PHYSICAL RESET

Physical reset has the effect of switching the power off and on again. All system parameters are reset to their default values and the system is re-initialised, including a set of diagnostic tests. (Note that logical reset, as explained above, does not reset all system parameters, neither does it cause diagnostic tests to be performed.) See Chapter 5 for details.

The physical reset switch is located in the small hole on the right hand side of the main unit. It is operated by inserting a ball-point pen or pencil into the hole (see Figure 2-5). A 'bleep' will be heard when contact is made.

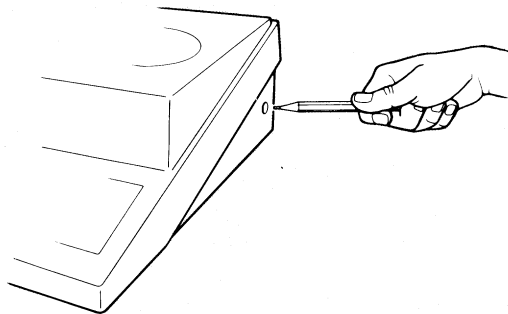


Fig. 2-5 Physical Reset

HARD DISK AND DISKETTE DRIVES

Hard disk and diskettes provide the bulk storage medium for information on the M20.

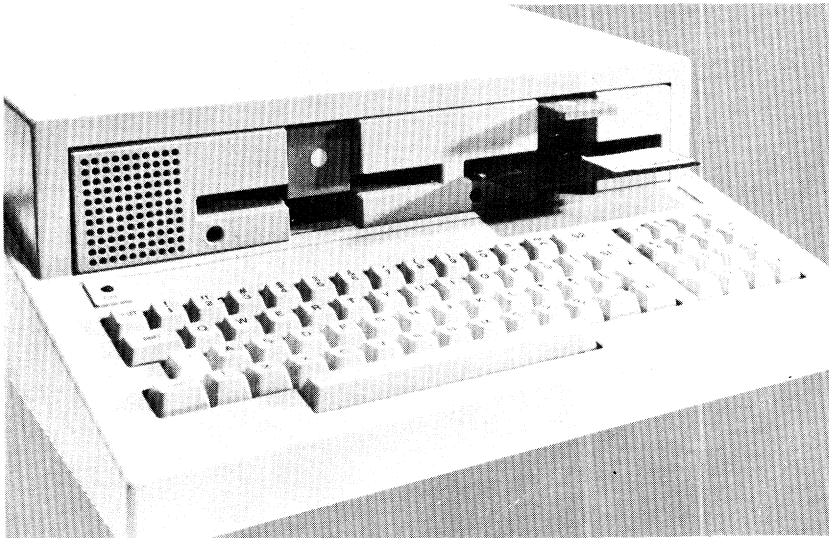


Fig. 2-6 The Diskette Drives

Figure 2-6 shows an M20 having two diskette drives - drive 0 on the right, drive 1 on the left. It also shows the position of the two indicator lights which tell you which drive is being accessed.

Your M20 will not necessarily have the configuration shown in Figure 2-6. In fact several combinations of disk and/or diskette drive are possible. Yours will be one of the following:

- one 160 Kbyte diskette drive
- one 320 Kbyte diskette drive
- one 640 Kbyte diskette drive
- two 160 Kbyte diskette drives
- two 320 Kbyte diskette drives
- two 640 Kbyte diskette drives

- one (fixed) hard disk drive (drive 10 on the right) and a 640 Kbyte diskette drive (drive 0 on the left)

Three types of diskette can be used:

- 160 Kbyte diskette (single-sided double-density)
- 320 Kbyte diskette (double-sided double-density)
- 640 Kbyte diskette (double-sided quadruple-density)

However note that while both 160 and 320 Kbyte diskettes can be used on a 320 Kbyte drive, only 160 Kbyte diskettes can be used on a 160 Kbyte drive. Moreover, while any diskette can be read via a 640 Kbyte drive, only 640 Kbyte diskettes can be written to on a 640 Kbyte drive.

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 this may damage the surface of the diskette. In this case a felt-tipped pen is recommended.

WRITE-PROTECTION

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 as indicated in figure 2-7.

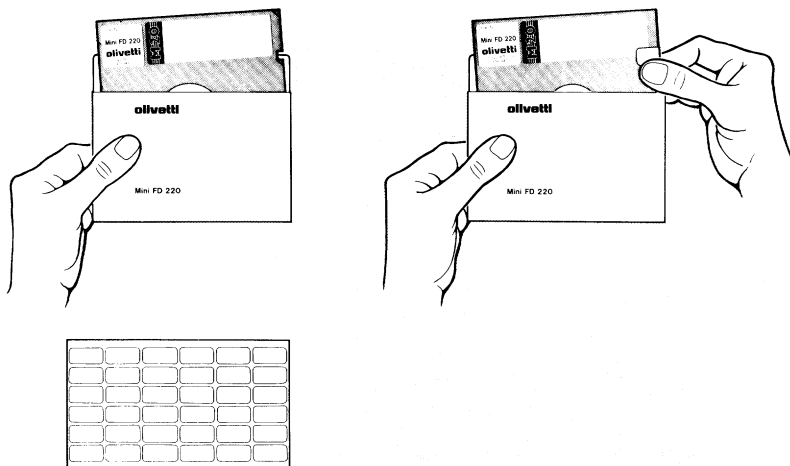


Fig. 2-7 Diskette Write-Protection

To remove write-protection simply peel off the aluminised label.

INSERTING AND REMOVING DISKETTES

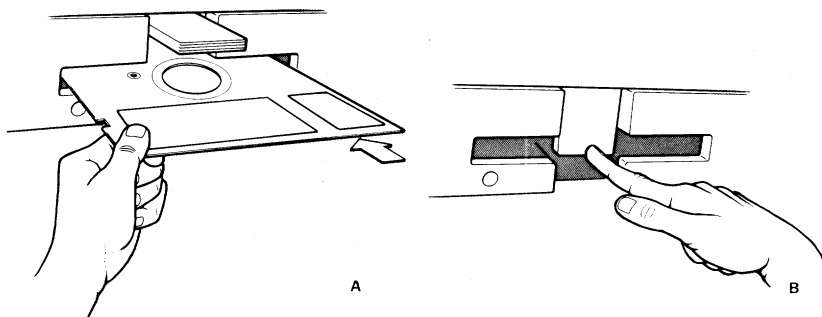


Fig. 2-8 Inserting a Diskette

To insert a diskette you do the following:

- Open the drive cover by pulling it outwards. It will spring open
- Insert the diskette into the slot with its label facing upward and nearest you (Figure 2-8A)
- Push the diskette gently into the drive until you feel it click into position. Do not attempt to force it in; if it will not go, withdraw the diskette and re-insert it
- Once the diskette has clicked into position, close the drive cover (Figure 2-8B)

To remove a diskette you merely open the drive cover. This automatically pushes the diskette out of the drive so you can withdraw it easily.

You can insert and remove diskettes while the M20 is powered-up or with the power off.

DO NOT ATTEMPT TO WITHDRAW THE DISKETTE WHILE IT IS BEING ACCESSED (DRIVE INDICATOR LIGHT ON). ATTEMPTING TO DO SO WILL CAUSE AN ERROR CONDITION AND IT MAY DESTROY THE INFORMATION ON THE DISKETTE.

PERIPHERAL INTERFACES

A standard M20 provides the following interfaces:

- EIA RS-232-C Serial Interface
- Centronics-like Parallel Interface

The following are optional:

- IEEE-488 Parallel Interface
- Twin EIA RS-232-C/20mA Current Loop Interface

EIA RS-232-C SERIAL INTERFACE

This standard I/O interface is offered on all M20 models to connect compatible devices (plotters, paper tape readers and punches, modems, etc...). It is programmable to correspond to baud rate (50 - 9600 baud), character length (in bits), presence or absence of parity, number of STOP bits in the data to be transmitted to or from the M20.

For further details refer to "I/O with External Peripherals User Guide".

CENTRONICS-LIKE PARALLEL INTERFACE

The M20 is fitted with this interface so that any compatible parallel printer may be connected to it.

To interact correctly with individual printers, the SFORM command allows the user to choose the desired print format.

IEEE-488 PARALLEL INTERFACE

This is an optional interface available on the M20.

Once the IEEE-488 driver is invoked by the IEEE command the M20 offers most of the IEEE-488 standard features to:

- read/write data from and to other devices
- assign 'talker'/'listener' status to other devices so that one talker may transmit data to several listeners

- receive service requests, conduct serial polls to identify the requesting device and respond with a user-programmed action
- transfer devices from 'remote' to 'local' control and vice versa
- act as controller and send commands to any device

All these features are invoked by use of BASIC IEEE-488 extension statements.

For further details refer to the "I/O with External Peripherals User Guide".

PRINTERS

The M20 is compatible with a wide range of printers. These can be connected either via the Centronics-like parallel interface, or the RS-232-C interface. For details of how to install and operate your particular printer(s) consult the appropriate manual:

- PR 1450 Operator Guide
- PR 1471 Operator Guide
- PR 1481 Operator Guide
- PR 2300 Operator Guide
- PR 2400 Operator Guide
- PR 2835 Operator Guide
- PR 320 Operator Guide
- PR 430 Operator Guide
- ET 121 Operator Guide
- ET 231 Operator Guide

3. SOFTWARE COMPONENTS

ABOUT THIS CHAPTER

This chapter describes the PCOS-related software components of the M20.

CONTENTS

<u>INTRODUCTION</u>	3-1	<u>BASIC INTERPRETER</u>	3-7
<u>PCOS</u>	3-3	<u>VIDEO FILE EDITOR</u>	3-7
MEMORY OPTIMISATION	3-3	<u>ASSEMBLER</u>	3-8
RESIDENT AND TRANSIENT COMMANDS	3-4	<u>PASCAL</u>	3-9
PROGRAMMABLE KEYS	3-4	<u>LINKER</u>	3-9
PROTECTION MECHANISMS	3-4	<u>PROGRAM DEBUGGER</u>	3-9
LINE EDITOR FUNCTIONS	3-5	<u>STANDARD INTERFACE HANDLERS</u>	3-9
REAL-TIME CLOCK	3-5	<u>PCOS COMMAND LIBRARY</u>	3-10
ROUTING INPUT/OUTPUT	3-5	CHANGING ENVIRONMENT COMMANDS	3-10
USER-DEFINED FONTS	3-6	PCOS CONFIGURING COMMANDS	3-10
CONTROL CHARACTER DISPLAY	3-6	SET SYSTEM GLOBAL COMMANDS	3-11
INITIALISATION FILES	3-6	KEYBOARD-RELATED COMMANDS	3-11
USER-CONFIGURABLE OPERATING SYSTEM	3-6	VOLUME HANDLING COMMANDS	3-12

FILE HANDLING COMMANDS	3-14
STANDARD INTERFACE HANDLING COMMANDS	3-14
USER AIDS	3-15

INTRODUCTION

The M20 software comprises a number of independent but closely related functional components. These can be considered to be at three levels: user, language and operating system (see Figure 3-1).

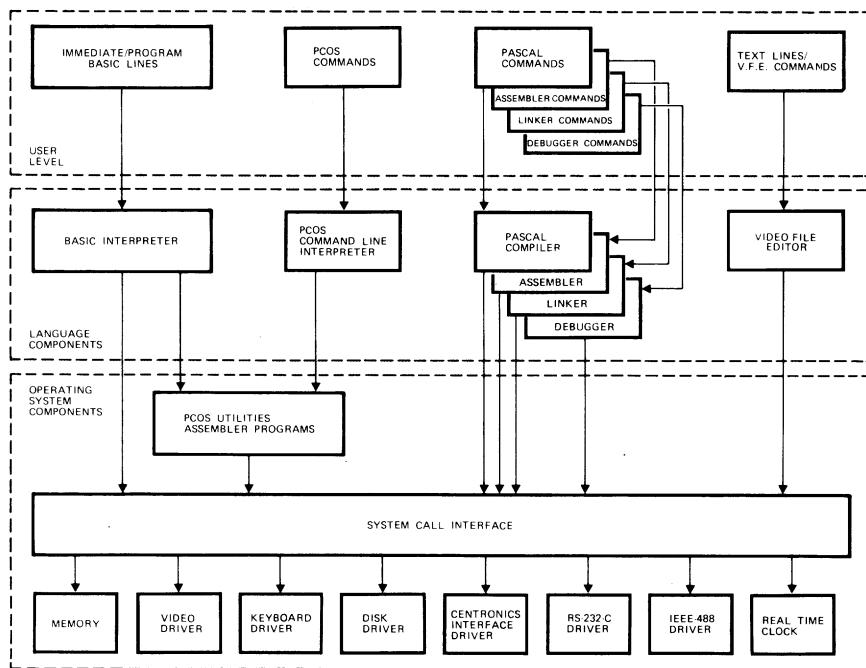


Fig. 3-1 M20 Software Components

The user level components are:

- immediate BASIC lines (that is, one or more BASIC statements or commands separated by colons). For example

```
RUN "1:NEWFILE" /CR/
```

It is executed as soon as you enter /CR/

- BASIC program lines (that is, a line number followed by one or more BASIC statements or commands separated by colons). For example:

```
100 PRINT "SIN of X is"; SIN (X) : IF X>2 THEN 1000 /CR/
```

It is stored in memory as soon as you enter /CR/

- PCOS commands. For example

```
vf 1: /CR/
```

- Assembler and Linker commands. These are special purpose PCOS commands used to invoke the Assembler and Linker, respectively
- Program Debugger commands. These are special commands for debugging programs
- PASCAL commands. These are special commands for invoking the PASCAL compiler
- text lines. That is, normal text entered at the keyboard
- Video File Editor commands. For example

```
/CTRL/ /6/ (EXIT AND SAVE)
```

The language level components are:

- The BASIC Interpreter, to interpret the BASIC statements, commands and programs
- The PCOS Command Line Interpreter, to interpret PCOS commands
- The Assembler, to generate an object file from an Assembly language source file
- The PASCAL compiler, to compile object code from PASCAL source files specified by the corresponding PASCAL command
- The Linker, to create an executable file from the object file(s) generated by the Assembler or PASCAL compiler and specified by the corresponding linker command. (Executable files are in turn treated exactly like PCOS utilities)
- The Video File Editor, to create and modify text files in response to text and Video File Editor commands entered at the keyboard

The Operating System components are:

- PCOS utilities (such as BASIC.CMD, BVOLUME.SAV, ...WFONT.CMD) and Assembler or PASCAL programs. These are executable routines to which are passed user-specified parameters. These in turn generate system calls
- The system calls. These are system routines which allow access to the drivers of the M20 hardware components, and perform operations such as moving strings of bits into memory, activating pixels on the video, reading characters from the keyboard, opening or closing a file, writing strings of characters to a file, etc.

PCOS

The M20 has its Professional Computer Operating System (PCOS) to manage:

- interaction with the CPU, memory, keyboard, diskette (and/or hard disk) drives and the VDU
- interaction with any connected peripherals through the four possible interfaces
 - . Centronics-like parallel interface suitable for a range of printers
 - . EIA RS-232-C serial interface suitable for access to peripherals or computers
 - . EIA RS-232-C twin serial interface (as an option) for access to RS-232-C and/or 20mA Current Loop peripherals and/or computers
 - . IEEE-488 parallel interface (as an option) suitable for access to other talkers and listeners such as counters, heat sensors, signal generators and measuring instruments
- handling of the real-time clock for all timing functions including date and time

The Comand Line Interpreter enables you to communicate with PCOS using a library of over 50 commands.

MEMORY OPTIMISATION

The M20 has a system of memory optimisation which is handled dynamically by allocating memory according to need and usage.

Memory optimisation is achieved by:

- using transient commands which are executed and then removed
- creating and then purging all temporary PCOS tables
- using the global command SBASIC to set resource level according to requirements of the application package and user program

For details of how to examine your particular memory configuration refer to the DCONFIG command in Chapter 13.

RESIDENT AND TRANSIENT COMMANDS

In an attempt to maximise user memory space, only three commands are always loaded into memory when the system is initialised. They cannot be removed from memory. These commands are:

- PLOAD - used to load transient commands into memory
- PUNLOAD - used to remove PLOADed commands from memory
- LTERM - used to differentiate among the three line-terminator keys / \downarrow /, /S1/ and /S2/.

The remaining commands are transient and can be executed then removed from memory. However, these commands can also remain in memory by means of the PLOAD command, and can become permanently resident by use of the PLOAD and PSAVE commands (see Chapter 6).

PROGRAMMABLE KEYS

Any key or key struck in combination with the /CTRL/, /COMMAND/ or /SHIFT/ key can have a special meaning assigned to it. This may be a BASIC or PCOS command, an expression, a constant, or any group of characters that may be found useful to have at the touch of a single keystroke or key combination. Assignment is made using the PKEY command, and can be made a permanent feature by means of the PSAVE command (see Chapter 6).

PROTECTION MECHANISMS

PCOS offers the following protection mechanisms:

- volume password protection using the VPASS command such that the protected volume cannot be accessed without knowledge of the password
- file password protection using the FPASS command such that the corresponding file cannot be accessed without knowing the password

- file write-protection using the FWPROT command to inhibit writing to the specified file(s)

For further details, and for information about other M20 protection mechanisms, see Chapter 8.

LINE EDITOR FUNCTIONS

PCOS offers line editor functions to:

- backspace (by pressing /CTRL/ /H/ simultaneously)
- cancel the current line (by pressing /CTRL/ /C/ simultaneously)
- hide what you enter (by pressing /CTRL/ /G/ simultaneously)

REAL-TIME CLOCK

The CPU includes an oscillator that generates a clock pulse every 50 ms used to strobe the real-time clock. The real-time clock provides the user with the local time in the ISO 24-hour format of hours:minutes:seconds (for example, 23:59:59 for one second to midnight) and the date in the format of month/day/year (for example, 12/01/82 for 1st December 1982). Note that the date format is month/day/year only for the USA keyboard versions; for all other national keyboards the date format is day/month/year. The internal calendar keeps track of days, months and years provided you set the time and date at switch-on using the SSYS command. The real-time clock stops at switch-off or physical reset, but not on logical reset.

ROUTING INPUT/OUTPUT

The M20 normally expects to receive input from the keyboard and sends output to the screen. However, PCOS enables both input and output to be redirected to other devices connected to the M20 by specifying "device re-routing parameters". This can be done in two ways:

- in a command line; where they will only be effective for the command in question
- by themselves; thus remaining in effect for all subsequent commands until they are changed, or until the system is re-initialised

For more details about device re-routing see Chapter 7.

USER-DEFINED FONTS

Characters are displayed on the VDU with a shape defined by the system font tables. But PCOS enables you to define and implement your own character font sets using the RFONT and WFONT commands. See Chapter 11 for details.

CONTROL CHARACTER DISPLAY

ASCII control characters are normally unprintable. However, PCOS contains a facility whereby each of these control characters is assigned a unique font, which will be displayed on the occurrence of the corresponding control character if control character display is specified. This can be done in one of two ways:

- in a command line; where control characters will only be displayed for the command in question
- by itself; thus remaining in effect until either actively cancelled, or the system is re-initialised

For more details about control character display refer to Chapter 11.

INITIALISATION FILES

PCOS enables you to create an initialisation file that will be executed automatically every time the system is initialised. Any sequence of commands, programs, BASIC statements, etc., can be executed in this way to initialise the system to suit your own needs. See Chapter 5 for details.

USER-CONFIGURABLE OPERATING SYSTEM

Resident and transient commands, programmable keys, routing input/output, user-defined fonts and control character display are all features of PCOS that enable you to define your working environment. In addition, a subset of PCOS commands enable you to change the global parameters of your system, such as the amount of memory available to BASIC, the system date and time, etc. Moreover, the current state of the system, as defined by these features, can be saved at any time using the PSAVE command. By subsequently re-initialising the system from the PSAVEd file, you will then restore the system to the state it was at the time it was saved.

For details on how to configure PCOS, refer to Chapter 6.

BASIC INTERPRETER

The BASIC Interpreter allows you to create, debug, and execute BASIC Language programs. The comprehensive instruction set includes sophisticated graphics facilities and special features for logical control of the IEEE-488 interface.

The instruction set is composed of commands and statements.

Statements are preceded by line numbers and grouped to form BASIC programs, which can then be executed. The user can halt the execution of a program, issue BASIC commands, and return to program execution without destroying the program variables. The BASIC statements include the following features:

- program segmentation through CHAINING and COMMON areas
- ability to CALL and EXECUTE Assembly Language routines and PCOS commands
- simple but powerful control statements (FOR/NEXT, GOTO, IF/GOTO, IF/THEN, ON/GOTO, WHILE/WEND, IF/THEN/ELSE, IF/GOTO/ELSE, ON ERROR/RESUME NEXT, GOSUB, ON/GOSUB)
- effective character string handling
- powerful print/display formatting statements
- Predictable Error handling (ON ERROR etc.)
- IEEE-488 control statements
- sophisticated graphics statements

For details refer to the "M20 BASIC Language Reference Guide".

VIDEO FILE EDITOR

The Video File Editor enables you to create and edit files of text, where a text file can be a file of normal text or a program written in any programming language.

The VDU displays a 21 line window of text which can be moved up or down within the file. Editing functions are entered from the keyboard and include :

- line and general editing functions. These include facilities to :
 - . move the cursor around the screen
 - . insert text either as a new line between adjacent lines or within an existing line

- . delete text either one line at a time or one character at a time
- . recall a deleted line
- . delete a block of text and restore it elsewhere
- . split and join lines of text
- window moving functions. These enable you to move the window up or down the file
- exit and save functions. These enable you to:
 - . save the edited text and exit the editor
 - . exit the editor without saving the edited text
 - . save the edited text without exiting the editor
- search functions to search the file for a specified string
- a subset of "high-level" commands. These commands enable you to:
 - . move the window to a specified line in the file
 - . delete blocks of text
 - . suspend processing of the current file and invoke the editor on another file

For further details see Chapter 12.

ASSEMBLER

The (optional) M20 Assembler processes an Assembly Language source file of ASCII text and produces an object file containing Z8000 machine code. Optionally, a listing file can be produced. This displays the source file program lines along with the generated code. The Assembler package also contains a number of commands to enable you to examine and manipulate your program files.

For details refer to the "Assembler Language User Guide".

PASCAL

The (optional) M20 PASCAL compiler processes a source program file written in PASCAL - a high-level programming language suitable for structured programming - and produces the corresponding object code. The M20 PASCAL is an extended version of the Microsoft PASCAL in that it can call run-time routines including graphics features.

For details refer to the "PASCAL Language User Guide".

LINKER

The (optional) Link utility creates an executable file from one or more object files.

For details refer to either the "Assembler Language User Guide" or the "PASCAL User Guide".

PROGRAM DEBUGGER

The (optional) Program Debug utility enables you to enter a range of commands for debugging and testing programs.

For details refer to the "Assembler User Guide".

STANDARD INTERFACE HANDLERS

PCOS contains two communications packages to manage input/output with peripherals and/or computers via the built-in RS-232-C interface, the (optional) twin RS-232-C interface, and the (optional) IEEE-488 parallel interface. User interaction is via a group of PCOS commands that enable BASIC programs to communicate via these interfaces. These commands are:

- IEEE which loads the IEEE-488 extension package
- RS232 which loads the RS-232-C interface package
- SCOMM which sets the protocol for an RS-232-C port
- CI which provides a BASIC interface with the RS-232-C driver

For details refer to the "I/O with External Peripherals User Guide".

PCOS COMMAND LIBRARY

This section lists the PCOS commands in functional groups. It lists both the full mnemonic and the two-character short form. The latter is the shortest form that PCOS will recognise as a keyword. This is explained in chapter 4.

For details of a particular command see chapter 13.

CHANGING ENVIRONMENT COMMANDS

KEYWORD		COMMAND TITLE
SHORT FORM	FULL MNEMONIC	
ba	BASIC.CMD	Loads the BASIC Interpreter
ed	EDIT.CMD	Loads the Video File Editor

Table 3-1 Changing Environment Commands

PCOS CONFIGURING COMMANDS

KEYWORD		COMMAND FUNCTION
SHORT FORM	FULL MNEMONIC	
pl	PLOAD (resident)	Loads commands
pr	PRUN.CMD	Reloads an operating system
ps	PSAVE.CMD	Saves PCOS
pu	PUNLOAD (resident)	Unloads commands

Table 3-2 PCOS Configuring Commands

SET SYSTEM GLOBAL COMMANDS

KEYWORD		COMMAND FUNCTION
SHORT FORM	FULL MNEMONIC	
sb	SBASIC.CMD	Sets the BASIC environment
sc	SCOMM.CMD	Sets the RS-232-C communications Port Environment
sd	SDEVICE.CMD	Changes device names
sf	SFORM.CMD	Sets the printer environment
sl	SLANG.CMD	Sets the national keyboard language
ss	SSYS.CMD	Sets the system environment

Table 3-3 Set System Global Commands

KEYBOARD-RELATED COMMANDS

KEYWORD		COMMAND FUNCTION
SHORT FORM	FULL MNEMONIC	
bk	BKEYBOARD.BAS	Activates the BASIC verbs facility
ck	CKEY.CMD	Changes the value of a key
lt	LTERM (resident)	Returns an integer (0, 1 or 2) depending on which of the three carriage return keys (, S1 or S2) was last used.
pk	PKEY.CMD	Assigns a string to a key

Table 3-4 Keyboard-Related Commands

VOLUME HANDLING COMMANDS

KEYWORD		COMMAND FUNCTION
SHORT FORM	FULL MNEMONIC	
bv	BVOLUME.CMD	Searches the volume directory, returns free disk space, or returns the name of the current volume (from BASIC only)
va	VALPHA.CMD	Alphabetises a directory
vc	VCOPY.CMD	Copies a volume (drive to drive)
vd	VDEPASS.CMD	Removes a password from a volume
vf	VFORMAT.CMD	Formats a volume
vl	VLIST.CMD	Lists a volume directory (full form)
vm	VMOVE.SAV	Copies a volume (using one drive)
vn	VNEW.CMD	Initialises a volume
vp	VPASS.CMD	Assigns a password to a volume
vq	VQUICK.CMD	Lists a volume directory (filename only)
vr	VRENAME.CMD	Renames a volume
vv	VVERIFY.CMD	Checks the hard disk for faulty blocks

Table 3-5 Volume Handling Commands

FILE HANDLING COMMANDS

KEYWORD		COMMAND FUNCTION
SHORT FORM	FULL MNEMONIC	
fc	FCOPY.CMD	Copies a file
fd	FDEPASS.CMD	Removes a password from a file
ff	FFREE.CMD	Frees unused file sectors
fk	FKILL.CMD	Deletes a file
fl	FLIST.CMD	Lists ASCII files
fm	FMOVE.CMD	Copies a file (diskette to diskette on a single-drive system)
fn	FNEW.CMD	Creates a new file
fp	FPASS.CMD	Assigns a password to a file
fr	FRENAME.CMD	Renames a file
fu	FUNPROT.CMD	Removes write-protection from a file
fw	FWPROT.CMD	Assigns write-protection to a file
rk	RKILL.CMD	Recovers a killed file

Table 3-6 File Handling Commands

STANDARD INTERFACE HANDLING COMMANDS

KEYWORD		COMMAND FUNCTION
SHORT FORM	FULL MNEMONIC	
ci	CI.SAV	Provides the BASIC interface to the RS-232-C driver
ie	IEEE488.SAV	Loads the IEEE-488 package
rs	RS232.SAV	Loads the RS-232-C package

Table 3-7 Standard Interface Handling Commands

PCOS GRAPHIC FACILITY COMMANDS

KEYWORD		COMMAND FUNCTION
SHORT FORM	FULL MNEMONIC	
la	LABEL.CMD	Displays a label string
ls	LSCREEN.CMD	Prints the displayed text
rf	RFONT.CMD	Creates an ASCII font matrix file from the currently active font
sp	SPRINT.CMD	Prints the text and graphic contents of a specified window
wf	WFONT.CMD	Makes a font matrix file active

Table 3-8 PCOS Graphic Facility Commands

USER AIDS

KEYWORD		COMMAND FUNCTION
SHORT FORM	FULL MNEMONIC	
co	COMMANDS.BAS	Displays details about PCOS commands
dc	DCONFIG.CMD	Displays the hardware and/or memory configuration
ep	EPRINT.SAV	Displays error messages
er	ERROR.BAS	Displays details about PCOS/BASIC error messages
he	HELP.BAS	Provides a series of display frames as a guide to using PCOS

Table 3-9 User Aids

4. ENTERING A COMMAND

ABOUT THIS CHAPTER

This chapter describes the format of a PCOS command, and the syntax used by this manual to describe the PCOS commands. For detailed descriptions of the commands mentioned in this chapter refer to Chapter 13.

CONTENTS

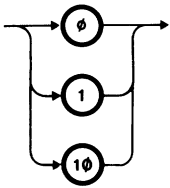
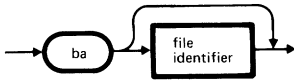
<u>NOTATION CONVENTION</u>	4-1
<u>COMMAND SYNTAX</u>	4-2
<u>COMMAND NAMES AND KEYWORDS</u>	4-3
<u>PARAMETERS</u>	4-3
<u>DEFAULT VALUES</u>	4-5
<u>RESIDENT AND TRANSIENT COMMANDS</u>	4-5
<u>COMMAND SEARCH PROCEDURE</u>	4-6
<u>FILE AND VOLUME IDENTIFIERS</u>	4-6
<u>WILD CARDS</u>	4-10
<u>NO INTERACTION FLAG</u>	4-10

NOTATION CONVENTION

PCOS commands are represented using syntax diagrams, in which:

- the keyword, written in its short form (using the first two characters) in lower case letters, is enclosed in an oval
- parameters are enclosed in rectangles
- punctuation, single characters and drive numbers are enclosed in circles
- flow lines connecting the above mentioned elements indicate any options which can be taken. Furthermore, a loop indicates any elements that may be repeated

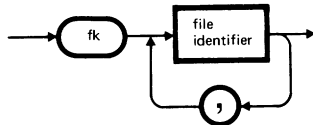
The following table shows some examples of the use of syntax diagrams. Note in particular the use of the flow lines.

No.	RULE	EXAMPLE
1	<p>A fork indicates a choice. One of the two paths must be followed in the direction of the arrow.</p> <p>For example, when a drive number has to be specified you can enter one of</p> <p>0 - drive 0</p> <p>1 - drive 1</p> <p>or</p> <p>10 - if you have a hard disk</p>	
2	<p>An empty branch or a by-pass indicates an optional element.</p> <p>In the example the file identifier in the PCOS command BASIC is optional</p>	

3

A loop indicates that the parameter may be repeated at least once.

For example, in the FKILL command the file identifier may be repeated several times using a comma as a separator



COMMAND SYNTAX

The general format is defined as follows:

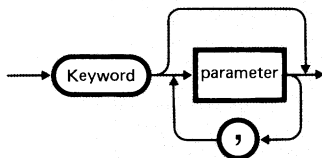


Fig. 4-1 General Format of a PCOS Command

Where

SYNTAX ELEMENT	MEANING
keyword	a mnemonic that specifies the command to be executed
parameter	a parameter to the command defining the command action. The number of parameters depends on the command executed but must be in the range 0 to 20

Remarks

A command may be entered over one or more lines (up to 255 characters) and terminates with the first occurrence of a /CR/.

A space may alternatively be used as the parameter separator except where nil parameters (see the section on "Parameters") are specified.

COMMAND NAMES AND KEYWORDS

A command name comprises a mnemonic string of up to 14 alphanumeric characters (the first of which must be a letter), optionally including a period and an extension. PCOS commands can have one of the following extensions; CMD, SAV or BAS. For example, BASIC.CMD. These extensions serve a purpose that will become evident in the following sections.

The keyword can be entered in either upper or lower case and must comprise at least the first two characters of the command name. For example

```
ba /CR/
```

```
basic /CR/
```

```
basic.cmd /CR/
```

will execute the command named BASIC.CMD.

Standard PCOS commands, user-written Assembly Language and PASCAL programs are all executed in this way. For details on executing Assembly Language programs refer to the "Assembler Language User Guide". For details of executing PASCAL programs refer to the "PASCAL User Guide".

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 different types of parameter are described below:

PARAMETER TYPE	MEANING												
integer parameter	<p>a decimal integer, or a hexadecimal integer (up to four characters) preceded by an "&" (ampersand).</p> <p>For example 10 &A</p>												
string parameter	<p>a string of alphabetic characters. Upper and lower case are interpreted differently. Leading and trailing quotation marks (either single or double) are optional except where the string contains any of the following:</p> <table><tr><td>+</td><td>(plus)</td><td>"</td><td>(double quotation mark)</td></tr><tr><td>&</td><td>(ampersand)</td><td>'</td><td>(single quotation mark)</td></tr><tr><td>SPACE</td><td></td><td>,</td><td>(comma)</td></tr></table> <p>In which case the string must be enclosed in quotation marks. If the string contains a single quotation mark, then it must be enclosed in double quotation marks, and vice versa. The syntax diagrams indicate when it is necessary or useful to include quotation marks. For brevity single quotation marks are always indicated</p>	+	(plus)	"	(double quotation mark)	&	(ampersand)	'	(single quotation mark)	SPACE		,	(comma)
+	(plus)	"	(double quotation mark)										
&	(ampersand)	'	(single quotation mark)										
SPACE		,	(comma)										
nil parameter	<p>a parameter that does not have a value specified in the command line. Such parameters assume default values. Nil parameters can be designated in one of two ways depending on their position in the command line:</p> <ul style="list-style-type: none">- before the last specified parameter. Such nil parameters are designated by a comma, with no preceding information. For example 1a 'title',,,5,2 /CR/ has two nil parameters after 'title'. These will assume their default values.- after the last specified parameter. In this case nil parameters are not entered. For example 1a 'title' /CR/ ' has four nil parameters after 'title'. All of these will assume their default values												

device re-routing parameter	<p>a parameter that re-routes input/output from/to specified devices or files. These are recognised by a "+" or a "-" sign as the first character.</p> <p>For example +dprt:</p> <p>For details refer to Chapter 7</p>
control character display	<p>a special parameter that enables or disables the display of control characters (unprintable ASCII characters 00 to 1F, hexadecimal).</p> <p>For example +cc</p> <p>For details refer to Chapter 11</p>

DEFAULT VALUES

Default values are automatically assumed when a nil parameter is detected.

Parameters which are governed by global commands (SBASIC, SSYS, SFORM, SCOMM, SLANG and SDEVICE), assume default values in the absence of any command. If a global command has been executed then the values specified by it are assumed by the system from then on until a new global command changes those values or until the system is re-initialised. In the latter case, the default values are again assumed.

RESIDENT AND TRANSIENT COMMANDS

There are only three resident commands; PLOAD, PUNLOAD and LTERM. These commands can never become transient. All the other commands are transient. Any transient command can be made resident, but the technique for doing this depends on the command file extension; that is, whether the command has a CMD or SAV extension. The former require the PLOAD command to make them resident, while the latter become resident simply by executing the command. Once a command has been made resident it remains so until the end of the current working session (that is, when the system is switched off or a physical or logical reset is performed), whereupon it becomes transient once again. Such commands, however, can be made permanently resident by means of the PSAVE command (see Chapter 6).

COMMAND SEARCH PROCEDURE

When a command is entered, PCOS will first search RAM for the first resident command that matches the characters entered. If found, the command is executed.

If no command is found in RAM then both drives are searched starting with the last drive selected, for:

1. A transient command with a CMD extension. If such a command is found, it is loaded into RAM, executed, and subsequently removed from RAM
2. A transient command with a SAV extension. If such a command is found, it is loaded into RAM, and executed. However, it is not removed from RAM. This means that the command can be used again even if the diskette it resides on is removed from its drive
3. A transient command with a BAS extension. If found the M20 will load the BASIC Interpreter, enter into BASIC execution mode, and run the file with the BAS extension
4. If a command cannot be found an error message (ERROR 92) is issued

Note: You may alternatively specify a drive number before the command keyword, thereby limiting the search to the specified drive. For example

```
1:vl 0: /CR/
```

will search only drive 1 for the VLIST command.

FILE AND VOLUME IDENTIFIERS

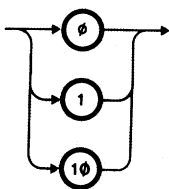
A disk may contain one or more program and/or data files. A single file, however, may not extend beyond one disk.

A group of files stored on the same diskette or disk forms a "volume". Each file and each volume has an identifier. Each file name must be unique on any one volume. Saving a program file which already exists on a volume causes the original file to be overwritten.

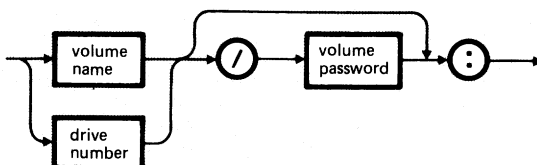
You may assign an identifier to a file either by an OPEN statement (data files), or by a SAVE command (program files), or by an FNEW, FCOPY, FMOVE, FRENAME or EDIT command.

You may assign an identifier to a volume by a VFORMAT, VNEW, VCOPY, VMOVE or a VRENAME command.

drive number



volume identifier



file identifier

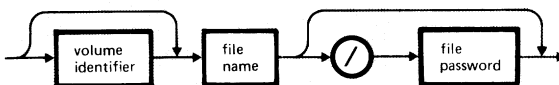


Fig. 4-2 File and Volume Identifier

SYNTAX ELEMENT	MEANING
volume name	<p>the name of a volume. This name must be a string of up to 14 printable ASCII characters (for illegal characters see below). The first character cannot be numeric.</p> <p>To select a specific volume in a PCOS or BASIC command or in an OPEN statement you must specify a volume name or a drive number.</p> <p>The volume name (or the drive number) may be followed by a volume password. At the end of a volume identifier a colon must be entered. For example</p> <p style="text-align: center;">fp VOL1/vpass:myfile,newpass /CR/</p>

	<p>Here "VOL1" is a volume name and "vpass" is the volume password, "myfile" is a file name and "VOL1:myfile" is a file identifier.</p> <p>Note: When specifying a file or volume identifier in a BASIC statement or command you must either include the identifier in a pair of quotation marks or write a string variable or a string expression whose value is the identifier. When specifying a file or volume by name in a PCOS command you need not include the quotation marks. For example</p> <pre>SAVE "VOL1:FILE1" /CR/ (in BASIC) vn VOL1: /CR/ (in PCOS)</pre>
drive number	<p>the drive number may be either 0 or 1 for a diskette, or 10 for a hard disk. For example</p> <pre>flist 1:FILE002 /CR/</pre> <p>Here "1:" indicates that file "FILE002" resides on the disk inserted in drive 1</p>
file name	<p>the name of a file, which must be a string of up to 14 printable ASCII characters (for illegal characters see below), optionally including a file extension. It must include at least one non-numeric character or be enclosed in quotation marks.</p> <p>To select a file in a PCOS or BASIC command or in an OPEN statement you must specify the file name. The file name may be preceded by a volume identifier and followed by a (file) password. For example</p> <pre>ba 1:MYPROG/MYPASS /CR/</pre> <p>IF YOU DO NOT SPECIFY ANY VOLUME IDENTIFIER BEFORE THE FILE NAME, THE SEARCH IS LIMITED TO THE LAST SELECTED DRIVE.</p> <p>The file extension is a string of up to 12 printable ASCII characters, preceded by a period (.). (For illegal characters see below.)</p> <p>Note: 1. filename.extension cannot exceed 14 characters</p>

	2. The extensions BAS, CMD and SAV have special meanings
file password or volume password	<p>the password to the file or volume. It must be a string of up to 14 printable ASCII characters (for illegal characters see below).</p> <p>Passwords give the user protection at volume or file level. They may be entered after a volume name, a drive number, or a file name and preceded by a slash. For example</p> <pre>f1 0:myfile/newpass /CR/</pre> <p>However, when assigning a password (using VPASS or FPASS) it is preceded by a comma as it does not yet form part of the identifier parameter. For example</p> <pre>fp 1:FILE111,NEWPASS /CR/</pre> <p>In this case the password "NEWPASS" is assigned to the file "FILE111" resident on the diskette in drive 1</p>

Illegal Characters

The following table indicates the characters that may not be included in a volume name, file name, or password.

= (equals) , (comma) \ (backslash) * (asterisk) /SPACE/	- (minus sign) : (colon) / (slash) ? (question mark)	+ (plus sign) # (hash or pound) ' (single quotation mark) " (double quotation mark)
or any control character		

Table 4-1 Illegal Characters

Note: The asterisk (*) and the question mark (?) may also be used in a file name in certain commands but with a special meaning (see the section "Wild Cards").

WILD CARDS

The M20 supports two "wild card" characters, the asterisk (*), and the question mark (?); which can be used in a file name to specify a group of file names.

An asterisk (*) represents any string of characters of any length (including no characters)

A question mark (?) represents any character. That is, it must match one, and only one, character.

Examples

IF you enter...	THEN...
v*.cmd	all file names starting with "v" and with the extension "cmd" are specified
????.*	all files with a four character file name, with an extension of any length are specified

Wild cards can be used in file identifiers with the following commands:

FCOPY	FDEPASS	FFREE	FKILL
FLIST	FPASS	FUNPROT	FWPROT
VLIST	VQUICK		

NO INTERACTION FLAG

The execution of PCOS commands often involves interaction with the user after the command has been entered (that is, after /CR/ is pressed). In some cases the video displays the result of the command. For example

if you enter

sb /CR/

then the M20 displays the current values of the SBASIC command parameters. That is, no further interaction takes place. But in other cases

interactive messages ask the user whether the process is to continue in one way or another. For example

if you enter

```
fk 1:V* /CR/
```

then the M20 will display all the file names starting with "V" one by one, asking the user whether the file is to be deleted or not. In each case the user must enter "y" for yes or "n" for no.

All this interaction and message display can be suppressed by specifying the "no-interaction" flag (%n) immediately after the command keyword. For example

if you enter

```
fk %n,1:V* /CR/
```

then all the file names beginning with "V" are deleted from the diskette inserted in drive 1, and no messages are displayed.

This facility allows PCOS commands, which normally display interactive messages, to be called and executed from a BASIC program without the need for interaction with the user. It can also be very useful to preserve the screen image while executing various commands.

5. INITIALISATION AND CHANGING ENVIRONMENTS

ABOUT THIS CHAPTER

This chapter describes the operational modes of the M20 and the initialisation process.

CONTENTS

<u>M20 ENVIRONMENTS</u>	5-1	INITIALISATION FOLLOWING A PRUN COMMAND	5-9
PCOS	5-1	<u>THE INITIALISATION FLOWCHART</u>	5-9
BASIC	5-1	<u>BEGINNING AND ENDING A WORKING SESSION</u>	5-12
VIDEO FILE EDITOR	5-1		
CHANGING ENVIRONMENTS	5-2		
<u>MODES OF PCOS</u>	5-5		
COMMAND MODE	5-5		
EXECUTION MODE	5-5		
CHANGING MODES	5-5		
<u>STANDARD INITIALISATION</u>	5-6		
<u>INITIALISATION FILES</u>	5-7		
<u>NON-STANDARD INITIALISATION</u>	5-8		
INITIALISATION FOLLOWING A PSAVE COMMAND	5-9		

M20 ENVIRONMENTS

The M20 can be operated in several distinct environments, such as alternative operating systems, editors and application programs. However, three such environments fall within the scope of this manual:

- PCOS
- BASIC
- Video File Editor

In each environment the M20 responds to the keyboard in a different way.

This section outlines the modes of operation in each of the environments and how you pass from one environment to another. The modes of PCOS are described in detail in the next section as are the techniques for passing from one PCOS mode to another. BASIC modes are fully described in the "BASIC Language Reference Guide". The modes of operation within the Video File Editor are described in Chapter 12.

PCOS

In this environment you can:

- enter PCOS commands
- execute PCOS commands

BASIC

In this environment you can:

- enter and edit BASIC immediate and program lines
- execute BASIC immediate and program lines
- invoke PCOS commands and Assembler subroutines using CALL or EXEC statements

VIDEO FILE EDITOR

In this environment you can:

- create and edit text files (including programs written in Assembly Language or PASCAL)

- create and edit BASIC program files

CHANGING ENVIRONMENTS

The following are the only possible changes of environment:

- from PCOS to BASIC
- from BASIC to PCOS
- from PCOS to the Video File Editor
- from the Video File Editor to PCOS

That is, you cannot enter either the Video File Editor from BASIC, neither can you enter BASIC from the Video File Editor.

The following figure illustrates the possible changes of environment.

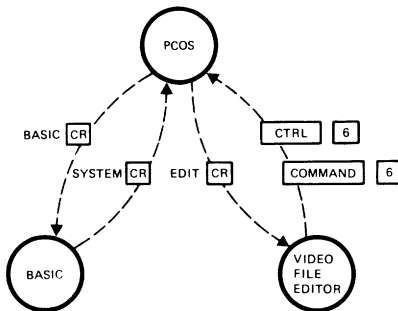


Fig. 5-1 M20 Environments

From PCOS to BASIC and Vice Versa

IF the M20 is in...	AND you enter...	THEN...
PCOS	ba /CR/	the M20 enters the BASIC environment
PCOS	a file name with the extension BAS OR the BASIC command with a file identifier as a parameter	the M20 enters the BASIC environment and executes the specified program
PCOS or BASIC	/CTRL/ /RESET/ simultaneously (logical reset)	the system is re-initialised but without running diagnostics
PCOS or BASIC	physical reset	the system is re-initialised and diagnostic tests are performed
BASIC	SYSTEM /CR/ OR the SYSTEM command is encountered during execution of a BASIC program	the system returns to the PCOS environment

From PCOS to Video File Editor and Vice Versa

IF M20 is in...	AND you enter...	THEN...
PCOS	ed file identifier /CR/	the M20 enters the Video File Editor environment for working on the specified file. If the file already exists on the volume specified by the file identifier (or on the last accessed drive if no volume is specified in the file identifier), then it is loaded into user memory. If not the file is created
Video File Editor	/CTRL/ /6/ simultaneously. Note: /6/ must be entered from the top row alphanumeric section of the keyboard	the M20 saves the current text file and exits the Video File Editor. (See note below.)
Video File Editor	/COMMAND/ /6/ simultaneously. Note: /6/ must be entered from the top row alphanumeric section of the keyboard	if text has been changed the editor will prompt you to confirm the abort. To do this you must press /COMMAND/ /6/ again, upon which the M20 exits the Video File Editor without saving the current text file. (See note below.)

Note: It is possible to work on more than one file without exiting the Video File Editor. If you want to exit the editor using /CTRL/ /6/ or /COMMAND/ /6/, you must first return to the file that was specified when the editor was invoked (by using /CTRL/ /6/ or /COMMAND/ /6/).

MODES OF PCOS

PCOS has two modes

- command mode (for entering PCOS commands)
- execution mode (for executing PCOS commands)

COMMAND MODE

When the M20 enters this mode it displays the PCOS prompt (n> - where n refers to the drive number) and the cursor (█).

In this mode you can enter PCOS commands. For example

```
fc 0:MYFILE,1:YOURFILE /CR/
```

While entering a PCOS command there are two line edit functions available. These can only be performed before /CR/ has been entered. If you

- press /CTRL/ /H/ (simultaneously) then the last character is deleted
- press /CTRL/ /C/ (simultaneously) then the entire line is deleted

Furthermore, subsequent characters can be rendered invisible by pressing /CTRL/ /G/. To return to visual mode you must press /CTRL/ /G/ again or /CR/.

EXECUTION MODE

In this mode PCOS executes the command that you have just entered. The mode is entered when /CR/ is pressed.

The user can interrupt some activities and return to command mode by pressing /CTRL/ /C/ simultaneously. Other activities, however, once started cannot be interrupted; for example, copying volumes. Having interrupted a command in this way it is not possible to resume execution of that command. You must re-enter the command.

On completion of command execution PCOS returns to command mode.

CHANGING MODES

The following table summarises how to pass from command mode to execution mode and vice versa.

IF the M20 is in...	AND...	THEN...
PCOS command mode	you enter a PCOS command (terminating with /CR/)	PCOS passes into execution mode
PCOS execution mode	you enter /CTRL/ /C/	where possible the command being executed is aborted and PCOS returns to command mode
PCOS execution mode	the command execution is completed	PCOS returns to command mode

STANDARD INITIALISATION

Initialisation takes place when you switch the machine on, or perform a physical reset. First of all diagnostics are run. These diagnostic tests check that the hardware is functioning correctly. Any faults are indicated by messages on the screen (see Appendix D). The diagnostics take a few seconds to run. On completion you will hear two 'beeps'. If, during diagnostics you press any of the keys /L/, /D/, /F/, /S/, or /B/ then a non-standard initialisation takes place (see next section). Otherwise, the initialisation is standard and proceeds as described below.

A logical reset (/CTRL/ /RESET/) also causes initialisation, but without diagnostics.

After diagnostics the bootstrap loader searches the drives for a "bootable" file. This file must be the first file on the volume and must also be of a certain format.

The bootstrap loader first checks the hard disk (drive 10 - if fitted) for a bootable file and loads it into RAM if found. If your system does not have a hard disk, or if the search on the hard disk was not successful, an error message is issued and the bootstrap examines the diskette in drive 0 and again tries to load a bootable file into RAM. If still unsuccessful, the search is repeated on the second diskette drive (drive 1 - if fitted). The loaded file is then executed.

If the above procedure has been unsuccessful in finding a bootable file, then the following bootstrap error message is displayed on the screen

Insert system disk and type any key

You must then insert the system diskette into an available drive, hit any key and the first file will be loaded into memory and executed. If it is a standard system diskette then the loaded file is named PCOS.SAV.

Note: Initialisation resets global parameters to their default values (these are the parameters managed by the Set System global Commands). An exception to this, however, is that the system date and time parameters are not reset on logical reset.

INITIALISATION FILES

Once the keyboard is initialised, PCOS starts a search on both drives for an initialisation file. This file can be named INIT.CMD, INIT.SAV or INIT.BAS (in upper or lower case).

The file names are searched for in the following order:

- INIT.CMD
- INIT.SAV
- INIT.BAS

If no initialisation file is found then the system enters PCOS command mode. The effect of an initialisation file, however, is as follows:

- INIT.CMD
This file can contain any program in machine language (for example a PCOS command) which is to be executed at initialisation. It can be created by saving the program in question in a file called INIT.CMD. If it exists in any of the two drives, the file is automatically loaded into RAM by PCOS at initialisation, executed and then purged from RAM. Subsequently, the system remains in the PCOS environment and waits for you to enter a PCOS command at the keyboard
- INIT.SAV
This file has the same characteristics as INIT.CMD. It is only loaded into RAM if INIT.CMD does not exist. Moreover, when it is loaded, the program is executed but not purged. It remains in RAM for the rest of the current session. After INIT.SAV is executed, the system remains in the PCOS environment and waits for you to enter a PCOS command at the keyboard
- INIT.BAS
This file can contain any BASIC program to be run at initialisation. To create an INIT.BAS file you can save the program in question specifying the file name to be INIT.BAS. (This can only be done from the BASIC environment.) If INIT.BAS exists, and neither INIT.CMD nor INIT.SAV exist, then the BASIC Interpreter and INIT.BAS are loaded into RAM by PCOS at initialisation. On encountering the extension BAS, the M20 enters the BASIC environment and the program file

INIT.BAS is run. What happens after INIT.BAS is run depends on the program itself. Note that INIT.BAS must reside on the same volume as BASIC.COM and BASIC.ABS

NON-STANDARD INITIALISATION

The initialisation process can be modified by pressing one of the keys /L/, /D/, /F/, /B/ or /S/ while the power-up diagnostics are still running. Each of these options is described in the following table:

IF, during power-up diagnostics you press	THEN...
/L/	<p>the system loops on diagnostics indefinitely. Its effect can be cancelled by</p> <ul style="list-style-type: none">- pressing one of the keys /D/, /F/, /B/ or /S/, in which case the corresponding non-standard initialisation process is performed- hitting any key other than /L/, /D/, /F/, /B/ or /S/, in which case initialisation continues as if /L/ had not been pressed in the first instance
/D/	<p>the system loops indefinitely on disk drive diagnostics, repeatedly reading track zero, side zero of the first ready drive</p>
/F/	<p>the bootstrap loader examines the diskette drive before the hard disk drive when looking for a bootable file</p>

/B/	the system passes directly into BASIC command mode without attempting to execute any initialisation file
/S/	the system passes directly into PCOS command mode without executing any initialisation file

INITIALISATION FOLLOWING A PSAVE COMMAND

The PSAVE command saves the current operating system then reboots the system as described in the section "Standard Initialisation", but without performing diagnostic tests.

INITIALISATION FOLLOWING A PRUN COMMAND

Following a PRUN command the bootstrap loader searches the drives in the same order as for a standard initialisation, but looks for a file (containing the operating system) specified by the parameter to the PRUN command. This file may be anywhere on diskette or hard disk, that is, it need not be the last PSAVEd file on the volume.

The message

Invalid File Error (xx) on drive (x)

is produced after each drive has been searched and either the specified file has not been found, or the specified file has been found but is not bootable.

No diagnostic tests are performed.

THE INITIALISATION FLOWCHART

The following figure summarises the initialisation process - both standard and non-standard - by means of a flowchart.

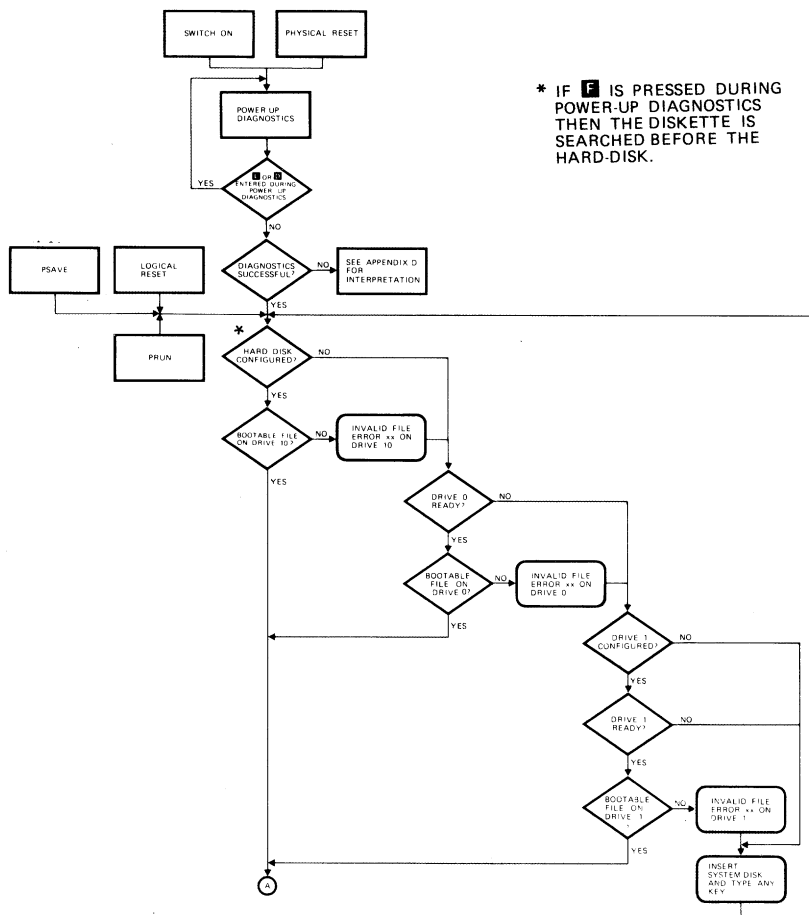


Fig. 5-2 The Initialisation Flowchart

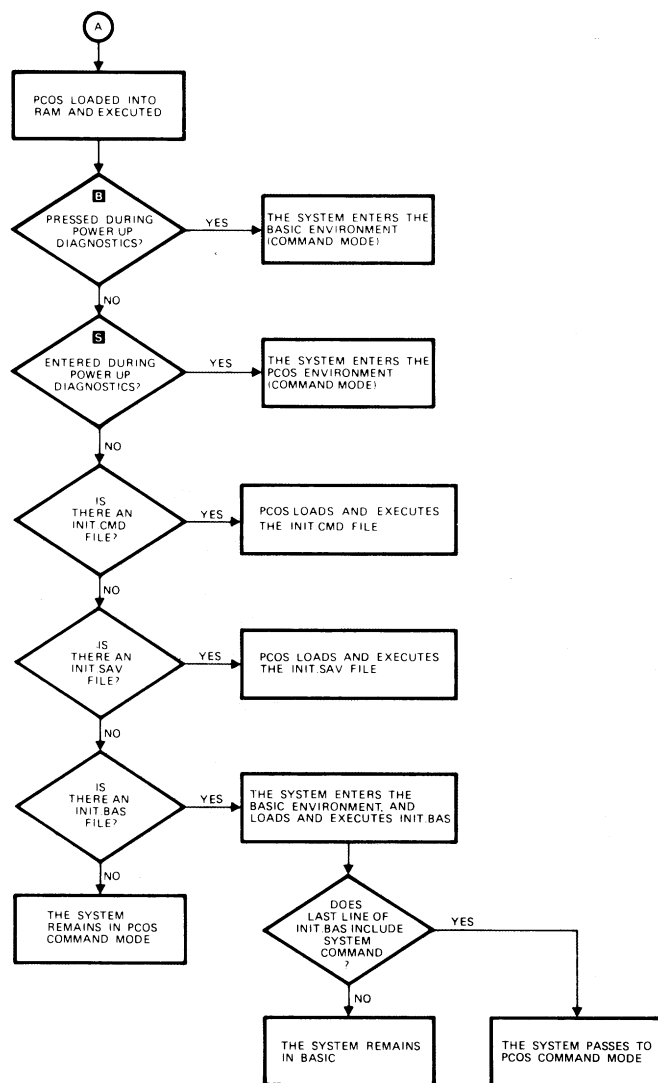


Fig. 5-2 The Initialisation Flowchart

BEGINNING AND ENDING A WORKING SESSION

The following table summarises the ways in which a working session can be started and terminated.

IF the M20 is...	AND you...	THEN...
switched off	switch on the M20 pressing the ON/OFF switch on the back panel	power-up diagnostics and initialisation follow, after which the bootstrap loader attempts to load a bootable file. If successful a new working session is commenced
switched on (in any environment)	perform a physical reset by pressing the physical reset button located down the right hand side of the main unit	the current working session is terminated. Power-up diagnostics and initialisation then follow after which the bootstrap loader attempts to load a bootable file. If successful a new working session is commenced
in the PCOS, BASIC or Video File Editor environment	perform a logical reset by pressing /CTRL/ /RESET/	the current working session is terminated, after which a standard initialisation (but without diagnostics) is performed. If a bootable file can be loaded the M20 passes into the PCOS environment and a new working session is commenced

6. CONFIGURING PCOS

ABOUT THIS CHAPTER

This chapter describes how to modify PCOS to meet your own needs. This is, how to make transient commands resident, how to assign functions to keys and how to change the system global parameters. For further details of the commands mentioned in this chapter refer to Chapter 13.

CONTENTS

<u>INTRODUCTION</u>	6-1	<u>SETTING THE PRINTING ENVIRONMENT</u>	6-15
<u>MAKING TRANSIENT COMMANDS RESIDENT</u>	6-1	<u>RECONFIGURING THE KEY-BOARD LANGUAGE</u>	6-19
<u>ASSIGNING STRINGS TO KEYS</u>	6-5	<u>SAVING A USER-CONFIGURED PCOS</u>	6-21
<u>CHANGING SYSTEM GLOBAL PARAMETERS</u>	6-7		
<u>SETTING THE SYSTEM GLOBAL ENVIRONMENT</u>	6-7		
<u>DISPLAYING AND MODIFYING DEVICE NAMES</u>	6-11		
<u>MODIFYING THE BASIC ENVIRONMENT</u>	6-14		
<u>SETTING THE ENVIRONMENT FOR AN RS-232-C COMMUNICATIONS PORT</u>	6-15		

INTRODUCTION

PCOS contains facilities that enable you to modify the standard PCOS configuration supplied by Olivetti on your system diskette. The possible changes fall into three categories:

- making transient commands resident: that is, transient commands that you expect to use frequently, or that you will require after removing the system diskette. These may be standard PCOS commands, or commands that you have written yourself using the Assembler package. (See the "Assembler Language User Guide")
- assigning functions to keys: for example a frequently input sequence of instructions can be assigned to a key, thus enabling a complex function to be performed at a single key stroke
- changing system global parameters: for example to change the system date and time, increase BASIC memory, accommodate a different type of printer, etc.

In this way you can create an operating system that is tailored to your specific needs.

The changes you make to your operating system become semi-permanent - that is, for the duration of the current working session (until you switch off the system or perform either a logical or physical reset) - unless you choose to make them permanent by saving your newly-configured operating system. How to do this is described in the section "Saving a User-Configured PCOS".

Note: A further way of altering the state of your system at initialisation is by means of an initialisation file. If such a file is present on an enabled volume, it will be loaded and executed at initialisation to automatically perform functions that you always require at initialisation (see Chapter 5).

Note: This chapter first tells you how to make transient commands resident, then how to assign functions to keys, and afterwards how to modify the system global parameters. These may, however, be performed in any order.

MAKING TRANSIENT COMMANDS RESIDENT

Once the standard PCOS is booted, three commands are loaded with PCOS. These are:

- PLOAD - used to load transient commands into memory
- PUNLOAD - used to remove a command from memory; that is, one that has previously been made memory resident by use of the PLOAD command. This can only be used for commands that are unloadable; that is, all except CI, RS232, IEEE, EPRINT,

VMOVE, PDEBUG (not included on the system diskette), PLOAD, PUNLOAD and LTERM

- LTERM - used in BASIC to distinguish between the use of the three line-terminator keys: / \downarrow /, /S1/ and /S2/

The M20's memory now comprises the PCOS Nucleus which includes the resident commands (PLOAD, PUNLOAD and LTERM) and user memory as shown in Figure 6-1.

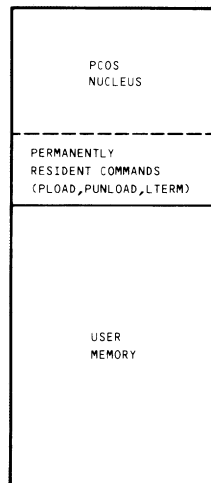


Fig. 6-1 M20 Memory after Initialisation of a Standard PCOS

All commands other than PLOAD, PUNLOAD AND LTERM are transient and fall into two categories:

- those with extension CMD : When you enter such a command it is loaded from the system diskette or hard disk, executed, then removed from memory. Using a transient command in this way takes much longer than using a command that is already in the M20's memory. Furthermore, if PCOS resides on diskette, this must be inserted. It is often necessary to have some such commands memory resident. To do this requires the PLOAD command
- those with the extension SAV : When you enter such a command it is loaded from the system diskette or hard disk into memory, executed, then remains in memory for the remainder of the current working session. Subsequent access will not require the command to be reloaded. Furthermore, once executed the command can be used again after the

system diskette has been removed. It is, therefore, not necessary to use the PLOAD command unless you need to remove the system diskette before the command is first executed

Commands with the SAV extension are:

- BVOLUME - for enabling a BASIC program to use the "Search" and "DiskFree" system calls or to obtain the name of the current volume
- CI - for programming the RS-232-C and current loop interface from BASIC
- EPRINT - for displaying PCOS error messages
- IEEE - for loading the (optional) IEEE-488 package
- PDEBUG - for entering the program debug environment (not included on the system diskette)
- RS232 - for loading the RS-232-C interface package
- VMOVE - for copying diskettes on a single drive system
- KANA - for enabling the Japanese keyboard (special project)

Example

If you enter

```
pl vc,px /CR/
```

then PCOS searches the drives, starting with the last volume accessed, for a command file with the short form "vc". VCOPY.CMD is found, loaded into user memory, and information concerning the command is displayed on the screen. For example

```
Disk file name = vcopy.cmd
Program name   = Volume Copy Rev. 3.x
Operation Mode = Segmented / System
Main entry = <0A>%B88C; Init entry = --None--
Memory allocated:
  Block No. %00; Starting address = <0A>%B88C; Size = %04FA
```

PCOS then looks for the command file with the short form "px". Since this does not exist, the error message

```
Error 92 in parameter 2
```

is displayed.

If you then enter

ep 92 /CR/

PCOS searches the drives for the corresponding command, finds EPRINT.SAV, loads it into memory and executes it. The result is a display of the PCOS error message 92 as follows

ERROR 92 ---- command not found

That is, the command short form "px" does not exist.

Since the EPRINT command has the SAV extension it remains in semi-permanent memory after execution and therefore further reduces user memory.

The effect on memory caused by the addition of these two commands is illustrated in Figure 6-2.

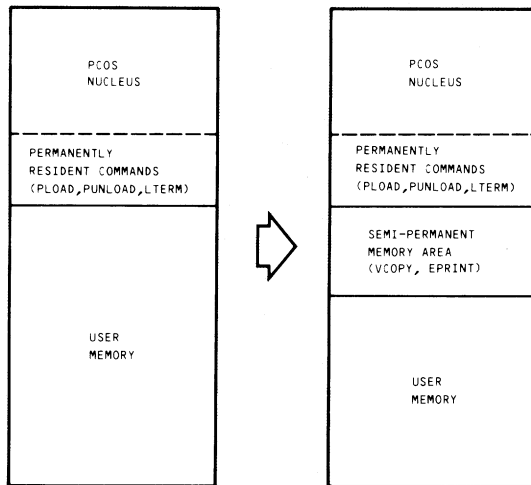


Fig. 6-2 M20 Memory after Making two Transient Commands Semi-Permanent


The exact amount by which user memory is reduced by the addition of the VCOPY and EPRINT commands can be determined using the DCONFIG command.

For each command that is loaded, user memory is reduced by the size of the command file (see the VLIST command) plus 40 bytes.

Making Commands Permanently Resident

Commands that are loaded into semi-permanent memory can be made permanent by means of the PSAVE command as described in the section "Saving a User-Configured PCOS".

ASSIGNING STRINGS TO KEYS

Any key on the keyboard (except /SHIFT/, /CTRL/, /COMMAND/, /RESET/, /S1/, /S2/, AND /) can have its value changed by means of the PKEY command. The new value can simply be another key value, for example, key A becomes key B, or the key can have a string assigned to it which can perform a function that can be entered from the keyboard. The key keeps its new value for the remainder of the current working session.

It is normal practice to use the /CTRL/ or /COMMAND/ key in conjunction with other keys when assigning values. In this way the original function of the key is not destroyed.

The key assignment is recorded in semi-permanent memory area and therefore reduces user memory. Each programmable key defined requires the space for the string (1 byte per character) plus one byte to hold the length of the string. One further byte may also be required if it is necessary to make the string address fall on an even boundary. Furthermore, if more than 26 key assignments are made, a further 156 bytes will be required.

You can specify the key either by enclosing the actual keyboard character in quotation marks, or by specifying the ASCII code (in decimal or hexadecimal) generated by that key. For example, on the USA ASCII keyboard

'B'

66

&42

all refer to the same key.

Similarly, the string(s) to be assigned to the key can be specified as either actual key values enclosed within quotation marks, or the ASCII code of each character, or a combination of the two. For example

'ba',13,&A

is a valid string representing 'ba' followed by a carriage return and a line-feed.

Example

Suppose that you want to enter the BASIC Interpreter and execute the statement FILES simply by pressing key combination /CTRL/ /!1/ (on the USA ASCII keyboard) simultaneously. Do this by entering

```
pk &E1,'ba',13,10,'files',13,10 /CR/
```

Where E1 is the (hexadecimal) ASCII code normally generated by pressing /CTRL/ /!1/.

On subsequently pressing /CTRL/ /!1/, PCOS will display 'ba', execute it by entering a carriage return (13) and a line feed (10), display 'files', then execute the BASIC command FILES by entering another carriage return / line feed pair.

The assignment will require the following memory space:

$1 + 11 = 12$ bytes

and the effect on M20 memory will be as shown in Figure 6-3.

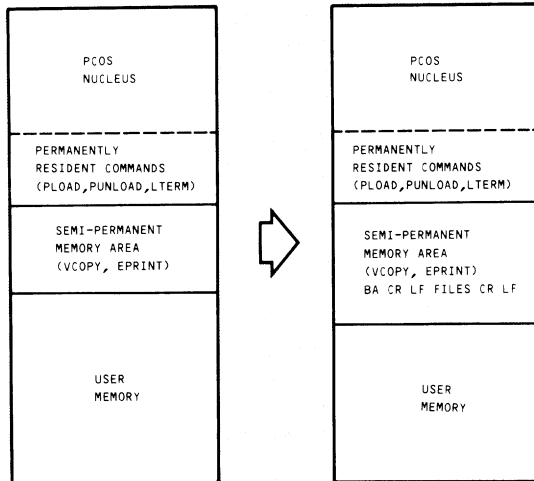


Fig. 6-3 The Effect of a PKEY command on M20 Memory

String assignments made in this manner are valid up to the end of the current working session. To make such assignments a permanent feature of the operating system you must use the PSAVE command as described in the section "Saving a User-Configured PCOS".

The template that fits into the slot above the top row of the keyboard can be used as a memory aid to the keys that you have programmed. Alternatively, you can display a list of programmed keys, along with their string assignments by entering

```
pk /CR/
```

A key assignment may be cancelled by entering the ASCII code as a single parameter to the PKEY command, thereby releasing the memory occupied by the corresponding string. Moreover, all strings assigned using the PKEY command can be cancelled by entering

```
pk %c /CR/
```

CHANGING SYSTEM GLOBAL PARAMETERS

The standard PCOS configuration contains system global parameters with default values already assigned to them. However, the PCOS command library contains a group of commands that enable you to modify these parameters to suit your needs. The commands are:

- SSYS.CMD - for setting the system global environment. For example, setting the system date and time
- SDEVICE.CMD - for displaying or modifying the device name table; that is, a table that lists the current names assigned to the hardware configured devices
- SBASIC.CMD - for setting the BASIC environment
- SFORM.CMD - for specifying the printer type and printing style
- SCOMM.CMD - for specifying the environment for devices connected via the RS-232-C interface
- SLANG.CMD - for simulating the keyboard of another country

SETTING THE SYSTEM GLOBAL ENVIRONMENT

If, after initialisation from a standard PCOS, you enter the SSYS command, the M20 will display the default values of the system global environment parameters. That is, if you enter

```
ss /CR/
```

then the M20 will respond with a display as follows

Date = 01/01/1982 Time = 00:00:01 Disk Verify = 0 (Off)
Extent size = 8 Display = 0 (64 x 16) Disk time = 2

Each parameter is described in turn below.

Date

This parameter specifies the system date. Its value can be any actual date since 1899. It comprises month, day and year separated from each other by a date separator, where:

- month is any integer value in the range 01 to 12
- day is any integer value in the range 01 to 28, 29, 30 or 31 depending on the month
- year is any integer value in the range 1900 (or 00) to 1999 (or 99)
- date separator is any legal name-printable character

(On keyboards other than the USA ASCII and USA ASCII with BASIC the order is day, month, year.)

The default value is 01/01/1982.

The system date is incremented by PCOS whenever the value of the Time parameter is incremented from 23:59:59 to 00:00:00.

The Date parameter is used in date-dependent programs. That is, it can be interrogated from BASIC using the BASIC DATE\$ function. For example:

DISPLAY	COMMENTS
100 IF DATE\$="04/30/82" THEN 3000 . . .	statement 100 checks the date.
500 DATE\$="05:06:82" . . .	statement 500 sets the date and also changes the date separator. Note: it is not necessary to specify the year in full. The last two digits are sufficient

Time

This parameter specifies the system time. Its value is maintained in hours, minutes and seconds separated by a time separator, where:

- hours is any integer value in the range 00 to 23
- minutes is any integer value in the range 00 to 59
- seconds is any integer value in the range 00 to 59
- time separator is any legal name-printable character

The time parameter assumes its default value (00:00:01) at system switch-on and upon physical reset, but not upon logical reset. It is then incremented until either you switch the system off or perform a physical reset.

The time parameter can be interrogated from BASIC using the BASIC TIME\$ function for use within time-dependent programs. For example

DISPLAY	COMMENTS
600 PRINT TIME\$. . .	statement 600 displays the time.
700 TIME\$="07:40:15" . . .	statement 700 sets the time

Disk Verify

This parameter determines whether or not verification is to be performed following a diskette (or hard disk) I/O operation. Its possible values are:

- 1 - verification on
- 0 - verification off

When verification is on, data that is written to a diskette (or hard disk) is subsequently read and checked. If an error is detected then the

message

ERROR 57 --- disk i/o error

is displayed.

The default is "0" - no verification.

Note: With Disk Verify on, the disk access time is slowed down significantly.

Extent Size

This parameter specifies the number of sectors to be allocated to a file when an output operation requires more space. Its value must be an integer in the range 1 to 1087. A lesser value wastes less space at the end of the last extent to be allocated but may require more extents to be allocated. Conversely, a larger value will result in more wasted sectors at the end of the last extent allocated, but will reduce the number of extents that need to be allocated.

The default is "8".

Display

This parameter determines the number of characters per display line and the number of lines per screen. Its value must be one of :

0 - for a 64 character by 16 line display

1 - for an 80 character by 25 line display.

The default is "0" - 64 by 16.

Disk time

This parameter specifies the number of seconds that the diskette drive motor will remain on after the last access to a diskette in that drive. This feature enables you to set this period such that start-up delays are minimised when, for example, an application performs communication or printer access between diskette accesses.

The Disk time parameter must be an integer in the range 1 to 30. The default is "2".

Using the SSYS Command to Modify the System Global Parameters

For example, if you enter

```
ss 12/12/82,09:30:00,,,1 /CR/
```

then the system date is changed to December 12 1982, the system time is started from 9:30 a.m., the Disk Verify and Extent Size parameters remain unaltered, the Display parameter is changed to generate an 80 by 25 display, and the Disk time parameter remains unaltered. M20 responds by displaying the new parameter values as follows

```
Date = 12/12/1982   Time = 09:30:00   Disk Verify = 0 (off)
Extent size = 8     Display = 1 (80 x 25)   Disk time = 2
```

Note that when entering the Date parameter it is not necessary to enter the year in full, the last two digits are sufficient. When examining the parameter value, however, all four digits will always be displayed.

Making the Changed Parameters Permanent

Any changes made to the Disk Verify, Extent Size, Display, and Disk time parameters remain until either they are respecified using another SSYS command, or the current working session is terminated. In the latter case these parameters revert to their default values. The Date and Time parameters are continually incremented until the system is either physically reset or switched off, upon which they too revert to their default values. Note, however, that logical reset has no effect on the Time and Date parameters.

All modified parameters (except Disk time) can be made permanent by means of the PSAVE command as described in the section entitled "Saving a User-Configured PCOS".

DISPLAYING AND MODIFYING DEVICE NAMES

After initialisation from a standard PCOS you can examine the device name default values via the SDEVICE command. That is, if you enter

```
sd /CR/
```

then the device name table will be displayed as follows

Device Name	Type
cons:	R/W
prt:	W
com:	R/W
com1:	R/W
com2:	R/W

The device names are listed along with their device type. The latter is an indication as to whether the device can only be written to, only read from, or whether both read and write operations are possible.

The entries in the table have the following meanings:

cons:

This is the default device name for the PCOS console driver (VDU and keyboard). It is a read and write type of device.

prt:

This is the default device name for the printer driver. It is a write-only type of device.

com:

This is the default device name for the built-in RS-232-C communications port. This value is only displayed if the RS-232-C extension package has already been loaded by means of the RS232 command.

com1:

This is the default device name for the first expansion RS-232-C communications port. This entry will not be displayed unless the RS-232-C extension package has already been loaded by means of the RS232 command.

com2:

This value is as for com1: except that it refers to the second expansion RS-232-C communications port.

Using Device Names

The device names are used for directing output to a specific device or for receiving input from a specific device. For example, when re-routing output from the console to the printer you would type

```
-dcons:.,+dpri: /CR/
```

See Chapter 7 for further details on device re-routing.

Changing Device Names

To change the name of a device you must again use the SDEVICE command but this time specifying as parameters first the old device name, then the new name for the same device. For example, if you enter

```
sd prt:,printer: /CR/
```

then the printer driver will be renamed accordingly. PCOS will respond with the display

Device Name	Type
prt:	W

changed to

Device Name	Type
printer:	W

The new name will remain until the end of the current working session. If you subsequently examine the device name table by again entering

```
sd /CR/
```

then the updated device name table will be displayed.

Making Changed Device Names Permanent

The new device names entered using the SDEVICE command remain in effect until either they are changed again by a subsequent SDEVICE command, or the current working session is terminated. In the latter case the device names revert to their default values. The changed names, however, can be made permanent by means of the PSAVE command as described in the section "Saving a User-Configured PCOS".

MODIFYING THE BASIC ENVIRONMENT

If, on initialising the system from a standard PCOS, you subsequently enter the SBASIC command, M20 will respond with the default values for the BASIC environment parameters. That is, if you enter

```
sb /CR/
```

then PCOS will respond with the default parameters as follows

```
Files = 3  Memory = 36000  Windows = 1  Record size = 256
```

Each parameter is described in turn below.

Files

This parameter indicates the maximum number of files that can be open concurrently under BASIC. Its value must be in the range 0 to 15. The default is 3.

Memory

This parameter specifies how much memory (in bytes) is available to BASIC. All of BASIC memory resides within user memory and the maximum value of this parameter is therefore determined by the size of user memory. The default is 36,000.

Windows

This parameter indicates the number of windows for which BASIC memory is guaranteed. That is, if you open more windows than stated by this parameter you cannot be sure that the necessary space will be available. Its value must be in the range 1 to 16. The default is 1.

Record size

This parameter specifies the maximum record size in bytes. Its value must be in the range 1 to 4096. The default is 256.

The Effect on BASIC User Memory

The actual amount of memory that is available to BASIC programs is dependent upon all the parameters to the SBASIC command. That is, the Record Size and Windows parameters reduce the available memory by the following amount:

$$829 + (578 + R)F \text{ bytes}$$

where R is the value of the Record size parameter and F is the value of the Files parameter.

The Windows parameter reduces the amount of available memory by 108 bytes for each window - except the first - that has space guaranteed.

Example

If you enter

```
sb 4,40000,7,64 /CR/
```

then the maximum number of files that can be open simultaneously is defined as 4, BASIC user memory is set to 40000 bytes, the number of windows for which space is guaranteed is 7, and the maximum record size is defined as 64 bytes.

The amount of memory available to the BASIC program is therefore

$$\begin{aligned} 40000 - [(829 + (578 + 64)4) + 108] \\ = 35955 \text{ bytes} \end{aligned}$$

SETTING THE ENVIRONMENT FOR AN RS-232-C COMMUNICATIONS PORT

This procedure uses the SCOMM command which is described in full in the "I/O with External Peripherals User Guide".

SETTING THE PRINTING ENVIRONMENT

If, upon initialising PCOS, you invoke the SFORM command without specifying any parameters, a display of the default values for the printing environment will result. That is, if you enter

sf /CR/

then the following display will appear.

```
auto .....  off,  ptype ... pr1450, lines ..... 0060,  
spacing ... 0001, compress .. ne, interface ..parallel,  
title .....
```

Each parameter is described below.

auto

This parameter determines the source of the nil parameters for this command. It can have one of two values:

- OFF - indicates that nil parameters will always take their values from the default values specified upon loading a standard PCOS, irrespective of any previous SFORM command
- ON - indicates that the nil parameters will take their values from those set by the last SFORM command of the current working session. Alternatively, if there was no such command, the nil parameters take their values from the parameters that were made permanent during a previous working session (see section on "Saving a User-Configured PCOS")

A value of ON makes this command behave as any other Set System Global command; but note that OFF is the default value, and in this respect the SFORM command behaves differently to the other Set System Global Commands.

On entering the SFORM command immediately after initialisation from a standard PCOS, the nil parameters will take their default values irrespective of the value of the "auto" parameter.

ptype

This parameter specifies the type of printer. Its value must be one of:

- pr1450
- pr1471
- pr1481
- pr2300
- pr2400

- pr2835
- pr-320
- pr-430
- et-121
- et-231
- transp

transp invokes transparent mode. This is a free format facility whereby file contents are printed exactly as specified in the file irrespective of the type of printer. That is, no additional end-of-line characters or form feed characters are added.

Note: The et-121 and et-231 are typewriters but can be connected as printers.

The default value is pr1450.

lines

This parameter specifies the number of lines that are to be printed on each page before automatic form feed. Zero implies that no form feed will be issued.

The default value is 60.

spacing

This parameter specifies the number of inter-line spaces between printed lines. Its value can be :

- 1 - single spacing
- 2 - double spacing
- etc.

The default is 1.

compress

This parameter specifies the style of the character. It is made up of two characters, the first of which must be one of:

w - wide (bold) character

n - normal width

and the second character specifies the width of character and must be one of:

c - compressed; that is, 16.6 characters per inch (at normal width)

e - elite; that is, 12.5 characters per inch (at normal width)

p - pica; that is, 10 characters per inch (at normal width)

The default value is ne.

interface

This parameter specifies whether the printer is to be connected to the serial or parallel interface. Its value must be one of:

se - the RS-232-C (serial) interface

pa - the Centronics-like (parallel) interface

The default value is pa.

title

This parameter defines the title which is to be printed at the top of each page. It can comprise up to 24 characters and must be enclosed in quotation marks. Entering a value of '}' deletes the current title.

The default value is no title.

Using the SFORM command

If you enter

```
sf ,pr1471,30,2,,, 'PCOS USER GUIDE' /CR/
```

then the printing environment parameters will have the following values:

auto - OFF - by default

ptype - pr1471


```

lines -          30
spacing -        2
compress-        ne - by default
interface -      parallel - by default
title -          'PCOS USER GUIDE'

```

All subsequent printer output will be directed to a PR1471 printer connected to the M20 via the parallel interface. The output will be printed 30 lines per page at double spacing in normal/elite type-face. At the top of each page will be the heading "PCOS USER GUIDE".

If, at a later time during the same working session, you enter

```
sf on,,50,1 /CR/
```

then the printing environment parameters will take the following values:

```

auto -          ON
ptype -         pr1471 - set by previous SFORM command
lines -         50
spacing -        1
compress -       ne - set by previous SFORM command
interface -      pa - set by previous SFORM command
title -         'PCOS USER GUIDE' - set by previous SFORM command

```

All subsequent output to the printer is to be printed 50 lines to a page at single spacing.

The above values will remain until either another SFORM command is issued, or until the end of the current working session. The values can be made permanent by use of the PSAVE command and thereby invoked during a future working session by entering an SFORM command with the auto parameter having a value of ON. See "Saving a User-Configured PCOS".

RECONFIGURING THE KEYBOARD LANGUAGE

The SLANG command enables you to reconfigure your keyboard to behave like any one of those defined in Appendix B.

If you enter

```
sl /CR/
```


then M20 displays a menu as follows:

Available Country Configurations:

Italy	0	Yugoslavia	10
West Germany	1	Norway	11
France	2	Greece	12
Great Britain	3	Switzerland/France ...	13
United States	4	Switzerland/Germany ..	14
Spain	5	Germany (Original) ...	15
Portugal	6	Datev	16
Sweden/Finland	7	Delta	17
Denmark	8		

Enter Your Selection by Number (or q to quit) -->

The last line of the display prompts you to enter the number corresponding to the country you require. For example, if you enter

2 /CR/

then the M20 responds with the message

Language requested : French (2)

and the individual keys on your keyboard subsequently correspond to those of a French keyboard.

Alternatively, you can quit the menu without changing the country number by entering

q /CR/

If you already know the number of the keyboard you require it is not necessary to examine the menu. You can simply include the number as a parameter in the command line. For example, if you enter

sl 5 /CR/

then the system responds with the message

Language requested : Spanish (5)

and the Spanish keyboard is invoked.

The new keyboard setting will remain until either another SLANG command is entered, or the current working session is terminated. On rebooting the system the original language is restored, unless the current selection is made permanent by use of the PSAVE command as described in the section "Saving a User-Configured PCOS".

SAVING A USER-CONFIGURED PCOS

All temporarily resident commands, programmed key definitions and modified global parameters, along with any currently active user-defined font (see Chapter 11), and any specified global device re-routing parameters (see Chapter 7) can be made permanent features of the operating system by means of the PSAVE command. A user-configured PCOS can be created using this technique.

A new system diskette can be created by performing a sequence such as the following:

STEP	OPERATION
1.	Insert the system diskette in drive 0, and boot the system
2.	Use the PLOAD command to select all the individual utilities which you require to be permanently loaded
3.	Use the PKEY command to set up all the function key definitions required
4.	Use the global commands (SBASIC, SSYS, SFORM, SCOMM, SDEVICE and SLANG) to provide the system parameter values you require
5.	Insert a newly formatted diskette in drive 1

6.	<p>Enter</p> <p>ps 1: /CR/</p> <p>then the system responds</p> <p>Save system on "1:PCOS.SAV"? (y/n)</p> <p>A "yes" response causes the operating system to be saved on the specified file, after which the system is rebooted from the new operating system which is then ready for use</p>
7.	<p>Label your new system diskette with a suitable title to distinguish it from the standard PCOS. You should also write-protect it using an aluminised label and make a back-up (see Chapter 9)</p>

PCOS can also be PSAVEd on a volume that contains other files. Moreover it may be given any valid file name. For example, entering

```
ps 1:newop /CR/
```

will save the operating system on the diskette inserted in drive 1. If the volume already contains a bootable file, but with a different name to the one you have just PSAVEd, the system will still re-boot from the newly PSAVEd file. Moreover, any future attempt to re-initialise the system from this volume will cause the system to boot itself from the most recently PSAVEd file. To boot any other operating system, on the volume requires the PRUN command. For example

```
pr 1:oldop /CR/
```

will boot the operating system saved in file "oldop".

Remark

If your system has a hard disk, then the user-configured PCOS can be saved either on hard disk, or on a diskette. If you choose to save it on the hard disk, you are advised to write-protect it using the FWPROT command.

7. DEVICE RE - ROUTING

ABOUT THIS CHAPTER

This chapter describes how input and output can be re-routed from/to alternative devices or files.

For further details of the commands mentioned in this chapter refer to Chapter 13.

CONTENTS

<u>INTRODUCTION</u>	7-1
<u>LOCAL DEVICE RE-ROUTING</u>	7-2
<u>GLOBAL DEVICE RE-ROUTING</u>	7-5
<u>DEVICE RE-ROUTING FROM A BASIC PROGRAM</u>	7-7

INTRODUCTION

Normal operation of the M20 system involves entering information via the keyboard and receiving responses on the VDU. In some cases, however, you may wish to use additional or alternative input and/or output devices; for example a printer can be used to get a hard copy of what is displayed on the video. This can be done on the M20 using "device re-routing parameters".

Device re-routing can be specified on two levels:

- local (for the duration of a single command only)
- global (for all subsequent PCOS commands entered during the current working session, or until respecified by another global device re-routing operation)

The input device may be re-specified as:

- the keyboard (standard)
- a hard disk or diskette file
- an RS-232-C port

The output device may be specified as:

- the VDU (standard)
- a disk or diskette file
- a printer
- an RS-232-C port

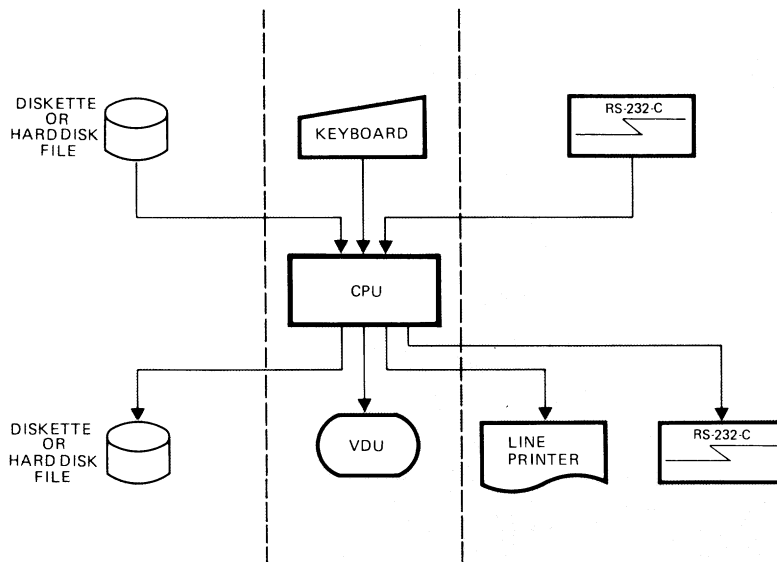


Fig. 7-1 Input/Output Devices

LOCAL DEVICE RE-ROUTING

This facility enables you to change the input and output devices for the current PCOS command, such that input can be received from devices other than the keyboard, and output can be routed to devices other than the VDU. After the command has been executed the re-routing specified is cancelled.

The implementation takes the form of a parameter to the command. This parameter specifies the name of the device whose I/O status is to be changed, preceded by two indicators. The first of these indicators specifies whether the device is to be cancelled as an input/output device (indicated by a "-" sign), or enabled as an input/output device (indicated by a "+" sign). The second indicator specifies whether the device is to supply the input (indicated by "S" for source), or receive

the output (indicated by "D" for destination).

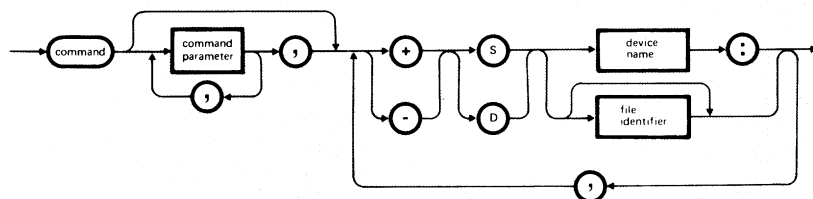


Fig. 7-2 Local Device Re-Routing

Where

SYNTAX ELEMENT	MEANING
command	a keyword of any PCOS command to be executed via the devices specified in the device re-routing parameters
command parameter	a parameter to be passed to the PCOS command in question
+	the device or file specified is to be enabled
-	the device or file specified is to be cancelled
S	the specified device is to be a source (for input to the CPU). It can be entered in upper or lower case

D	the specified device is to be a destination (for output from the CPU). It can be entered in upper or lower case
device name	<p>a string of up to 13 printable ASCII characters, the first of which must not be a digit, denoting the device in question. This can be the device default name or any other name assigned to the device using the SDEVICE command.</p> <p>Note: A device name must be followed by a colon</p>
file identifier	<p>any file identifier. If the file does not exist it will be created on either the specified volume or the volume inserted in the last selected drive.</p> <p>Only one source and one destination file are permitted to be open at one time</p>

Note

- + or -, S or D and device name or file name follow each other without any spaces in between, and constitute one parameter
- Each device re-routing parameter must be followed by a comma to separate it from the next parameter
- The position of the device re-routing parameter in the comand line is arbitrary. It is shown here at the end of the line merely for clarity
- +DPRT: may also be written as +PRT (in upper or lower case letters) see last example below
- If additional devices are enabled for input or output without disabling the currently active device(s) then devices become simultaneously active. The user must therefore exercise caution when re-routing input to avoid data from several devices becoming intermixed

Examples

IF you enter...	THEN...
<pre>vl 0:,-dcons:,+DPRT: /CR/</pre>	<p>the directory of the diskette inserted in drive 0 is printed (because of +DPRT:). However, it is not displayed on the VDU (because of -dcons:)</p>
<pre>ba +SBASIC.CMD /CR/</pre>	<p>the M20 goes into the BASIC environment (see the BASIC command) and will take input from both the BASIC.CMD file and the keyboard. The CPU will no longer take input from the BASIC.CMD file when it goes back into the PCOS environment</p>
<pre>ss +PRT /CR/</pre>	<p>the global system environment parameters are not only displayed on the VDU, but also printed</p>

GLOBAL DEVICE RE-ROUTING

Specification of global device re-routing causes the input and output of all PCOS commands executed thereafter to be re-routed. Any re-routing remains in operation until otherwise specified or until the system is reset. Furthermore, global device re-routing parameters can be made a permanent feature of the operating system by use of the PSAVE command.

Global device re-routing can be put through if device re-routing parameters, as described in the syntax diagram for local device re-routing (see Figure 7-2), are specified by themselves; that is, in the absence of a command keyword.

Note:

- + or -, S or D, and device name follow each other without any spaces in between
- Device re-routing parameters must be separated from each other by a comma
- +DPRT: may also be written as +PRT (in upper or lower case letters)
- If additional devices are enabled for input or output without disabling the currently active device(s) then devices become simultaneously active. The user must therefore exercise caution when re-routing input to avoid data from several devices becoming intermixed
- No more than one file for output and one file for input may be specified at any one time
- The keyboard can be disabled by specifying "-SCONS:". But, if this is done at the global level, control cannot be regained unless an external active device issues a "+SCONS:", or the system is reset

Examples

IF you enter...	THEN...
+scom: ,+dcom: /CR/	the CPU will receive input from both the keyboard and the built-in RS-232-C communications port (assuming the latter has already been initialised). Output will be displayed on the VDU and re-routed to the RS-232-C communications port
-dcom: ,+dprrt: /CR/	the RS-232-C communications port (previously enabled) will be cancelled as a destination (for output) and the printer activated. Any other devices previously allocated as a source or a destination will remain active

<pre>+d1:fileA /CR/</pre>	<p>"fileA" resident on the diskette inserted in drive 1 is enabled for output. If "fileA" does not exist on the volume it is created, and all the output will be both displayed on the screen and re-routed to "fileA"</p>
<pre>+dprt: /CR/ +d1:output /CR/ -dprt: /CR/ -d /CR/</pre>	<p>the first command enables the printer for output. The second enables the file named "output" resident on the diskette inserted in drive 1 to receive output. The third command cancels the printer and the fourth cancels the file.</p> <p>Note: When a file is cancelled the identifier is not checked and can therefore be omitted</p>

DEVICE RE-ROUTING FROM A BASIC PROGRAM

When local device re-routing is specified while in BASIC, I/O redirection will continue until either a command is executed to explicitly turn it off, or BASIC is exited. Conversely, global device re-routing will remain active after BASIC is exited.

Examples

IF you enter...	THEN...
<pre>ba /CR/ EXEC "v1 1:;+D1:OUT" /CR/ . . EXEC "-D1:OUT" /CR/</pre>	<p>the first command enters the BASIC environment. The first EXEC statement performs a VLIST command on the diskette inserted in drive 1 and routes the volume list to the file named "OUT" on the same diskette as well as to the VDU. All subsequent output will also be routed to the file until it is cancelled by the second EXEC statement</p>

<pre>ba /CR/ EXEC "v1 1:,,+dprt:" /CR/ . . . SYSTEM /CR/</pre>	<p>the system enters the BASIC environment. The EXEC statement causes the directory of the diskette inserted in drive 1 to be printed. All subsequent output is also directed to the printer until the printer is cancelled as an output device by the SYSTEM command</p>
<pre>ba /CR/ EXEC "+prt" /CR/ . . . SYSTEM /CR/</pre>	<p>the system enters the BASIC environment. The EXEC statement specifies the printer as an output device, but because "+prt" is specified globally, the following SYSTEM command will not cancel it</p>

8. PROTECTION TOOLS

ABOUT THIS CHAPTER

This chapter describes the mechanisms by which a file or volume can have protection applied. For details about the commands mentioned in this chapter refer to Chapter 13.

CONTENTS

<u>INTRODUCTION</u>	8-1
<u>VOLUME PASSWORDS</u>	8-1
<u>FILE PASSWORDS</u>	8-2
<u>WRITE-PROTECTION</u>	8-4
<u>COPY-PROTECTION</u>	8-4
<u>BASIC PROGRAM SECURITY</u>	8-4

INTRODUCTION

The M20 offers password protection at both volume and file level. Write-protection may be applied to a diskette or a file, but not to the hard disk. Moreover, a BASIC program can be protected so that it cannot be listed, edited or saved again.

The following sections summarise the various protection mechanisms.

VOLUME PASSWORDS

IF you want to...	THEN...
<p>assign a password to a volume</p>	<p>issue a VPASS command, specifying the password. For example</p> <pre>vp MYVOL:,MYPASS /CR/</pre> <p>OR</p> <p>if the volume already has a password this must be specified by the VPASS command, which, in this case, will change the password. For example</p> <pre>vp VOL1/OLDPASS:,NEWPASS /CR/</pre>
<p>access a volume that has a password (or a file saved on a volume that has a password)</p>	<p>enable that volume by specifying the volume password after the volume name or the drive number. This can be done in a BASIC or PCOS command or in an OPEN statement.</p> <p>Note: Once a diskette volume password has been specified it need not be re-specified until the diskette has been removed and another diskette has been referenced in the drive in which the diskette was inserted, or the current working session is terminated. Moreover, once a hard disk volume password has been specified, the hard disk will remain enabled until the end of the current working session</p>

remove a volume password	<p>issue a VDEPASS command, for example</p> <p>vd MYVOL/MYPASS: /CR/</p> <p>Note: You must know the password to use the VDEPASS command</p>
hide a volume password	<p>press /CTRL/ /G/.</p> <p>The cursor will change its shape and blink rate and the display of entered characters is suppressed (Hide mode).</p> <p>To return to normal display mode you must press /CTRL/ /G/ again, or /CR/</p>

FILE PASSWORDS

IF you want to...	THEN...
assign a password to an existing file (that has no password)	<p>issue an FPASS command, specifying the password. For example</p> <p>fp V1:MYFILE,PASS001 /CR/</p>
create a new file and assign a password to it	<p>issue an FNEW command, specifying the password. For example</p> <p>fn 1:newfile/pass002,4 /CR/</p>
change the password to an existing file	<p>issue an FPASS command, specifying the old password within the file identifier, and the new password as the second parameter. For example</p> <p>fp 1:newfile/pass002,pass102 /CR/</p>

assign a password to a group of files	<p>issue an FPASS command, specifying the group using wild card characters. The same password will be assigned to all files in the group. For example</p> <pre>fp 1:my*,mine /CR/</pre>
assign a password to a list of files	<p>issue an FPASS command, specifying a list of files and, as the last parameter, the common password. For example</p> <pre>fp 1:myfile,yourfile,hisfile,herfile,ours /CR/</pre>
assign a password to a program file (that has none)	<p>an FPASS command can be issued or, if in BASIC, the program can be stored on a volume using the SAVE command specifying the password. For example</p> <pre>SAVE "FILEABC/PASSABC" /CR/</pre>
access a file that has a password	<p>specify that password after the file name. For example</p> <pre>fw FILE002/PASS002 /CR/</pre>
remove a file password	<p>issue an FDEPASS command, specifying the password. For example</p> <pre>fd V1:MYFILE/PASS001 /CR/</pre>
hide a file password	<p>press /CTRL/ /G/ simultaneously.</p> <p>The cursor changes its shape and blink rate and the display of entered characters is suppressed (Hide mode).</p> <p>To return to normal display mode you must press /CTRL/ /G/ again, or /CR/</p>

WRITE-PROTECTION

You can apply write-protection to a diskette or a file.

IF you want to...	THEN...
write-protect a diskette (that is, to prevent any writing to that diskette)	cover the write-protect notch with an aluminised label
remove write-protection from a diskette	remove the aluminised label
write-protect a file	issue an FWPROT command, specifying the file identifier. For example fw 1:myfile /CR/
remove write-protection from a file	issue a FUNPROT command, specifying the file identifier. For example fu 1:myfile /CR/

COPY-PROTECTION

Copy-protection can only be assigned by the supplier. It can be assigned at file or volume level. A copy-protected file can be copied only a specified number of times. A copy-protected volume, however, cannot be copied at all.

BASIC PROGRAM SECURITY

Besides password protection and write-protection the M20 offers a further level of security. In the BASIC environment a program can be saved using the SAVE command with the P option. In such a case, the

saved program can no longer be:

- listed
- modified
- saved again

For example

SAVE "1:FILEAPROT", P /CR/

saves the file program FILEAPROT with the P option on the diskette inserted in drive 1.

9. VOLUME HANDLING

ABOUT THIS CHAPTER

This chapter provides an operational guide to the use of the volume directed commands.

Throughout this chapter the availability of commands is always assumed. That is, it is assumed that either a volume containing the command is present in one of the drives, or that the command in question is already resident in memory.

Additional information about the commands mentioned in this chapter can be found in Chapter 13.

CONTENTS

<u>FORMATTING AND INITIALISING NEW VOLUMES</u>	9-1	<u>COPYING VOLUMES</u>	9-6
FORMATTING A DISKETTE OR HARD DISK	9-1	COPYING VOLUMES ON A DUAL-DRIVE SYSTEM (VOLUMES OF EQUAL SIZE)	9-8
INITIALISING A VOLUME	9-3	COPYING DISKETTES USING ONLY ONE DRIVE (VOLUMES OF EQUAL SIZE)	9-9
<u>LISTING A VOLUME</u>	9-4	COPYING VOLUMES (OF DIFFERENT SIZES)	9-9
LISTING A VOLUME USING THE VLIST COMMAND	9-4		
LISTING A VOLUME USING THE VQUICK COMMAND	9-5		

<u>NAMING AND PROTECTING A VOLUME</u>	9-9
WRITE-PROTECTION	9-9
PASSWORD PROTECTION	9-10
NAMING A VOLUME	9-10
<u>ALPHABETISING A VOLUME</u>	9-11

FORMATTING AND INITIALISING NEW VOLUMES

The operations of formatting a volume and initialising a volume are carried out by the VFORMAT and VNEW commands, respectively.

Formatting creates blank sectors on the tracks of a diskette or hard disk and checks for defective blocks on the hard disk only). A new diskette or hard disk must be formatted before use but note that Olivetti-supplied diskettes for the M20 are pre-formatted to a high precision and therefore do not require formatting. In fact to ensure reliability and to ensure that the same diskette can be used on any M20, you are advised not to re-format. Pre-formatted diskettes, however, are not initialised, and therefore require the VNEW command before they can be used.

Diskettes that are not supplied by Olivetti, however, will need to be formatted using the VFORMAT command. Once formatted in this manner, a diskette is initialised, ready for use.

Used M20 diskettes, whether Olivetti-supplied or not, should not be re-formatted. It is sufficient to initialise them using the VNEW command. The same is true for hard disks, except that after performing a VVERIFY destructive test it is necessary to re-format.

FORMATTING A DISKETTE OR A HARD DISK**Formatting New Diskettes**

To format a new diskette you must place the unformatted diskette (which must not be write-protected) in an available drive (for instance drive 1), then enter

```
vf 1: /CR/
```

OR

```
vf %s 1: /CR/ - for the special case of a 160 Kbyte diskette on
a 320 Kbyte drive
```

In either case the following message will be displayed:

```
Warning - VFormat deletes all files. Format disk? (y/n)
```

Respond by entering

```
y /CR/
```

(or n /CR/ to abort)

and formatting begins. The message

Formatting Track 0

appears on the screen, and the track number is subsequently incremented as each track is completed until all tracks are formatted. At this point the message

Formatting Complete

is displayed and the PCOS prompt appears. The diskette is then formatted and ready for use.

Formatting Used Diskettes

Formatting will also work on used diskettes, although you are recommended to use the VNEW command. However, supposing you have a used diskette bearing the name "mydisk", you can reformat it by entering

```
vf mydisk: /CR/
```

OR

```
vf %s mydisk: /CR/ - for the special case of a 160Kbyte  
diskette on a 320 Kbyte drive
```

The M20 will respond with the message

Warning - VFormat deletes all files. Format disk? (y/n)

Respond by entering

```
y /CR/
```

If the diskette is not enabled, that is, it is password protected and has not yet been accessed during the current working session, then the message

Diskette appears password protected. Format disk? (y/n)

is displayed. Respond by entering

```
y /CR/
```

and formatting proceeds as for a new diskette. Once formatting is complete the diskette will be unnamed and have no password. The diskette is then ready for use.

Formatting Hard Disks

Hard disks are formatted in exactly the same way as diskettes except

that the drive number is always 10. That is, you should enter

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

then the procedure continues as for a diskette.

No Interaction

The dialogue for any of the above formatting procedures can be skipped by including the no interaction flag (%n) in the command line. For example, if you enter

```
vf %n mydisk: /CR/
```

then the diskette named "mydisk" is formatted. Formatting is complete when the PCOS prompt is displayed. The diskette is then ready for use.

INITIALISING A VOLUME

To initialise a volume you must enter the VNEW command along with the volume identifier and, if the disk or diskette is not enabled, the volume password. For example

```
vn mydisk/pass: /CR/ - for a diskette having the name "mydisk"
                        and password "pass"
```

OR

```
vn %s mydisk: /CR/ - for a diskette having the name "mydisk"
                    and that is a 160 Kbyte diskette on a
                    320 Kbyte drive
```

OR

```
vn 10: /CR/ - for a hard disk
```

The M20 will respond with the message

```
Warning - vnew deletes all files. Initialise disk? (y/n)
```

Respond by entering

```
y /CR/
```

The initialisation process then begins. Its completion is indicated by the PCOS prompt.

The initialisation process removes any volume name.

Note that the above dialogue can be avoided by use of the no-interaction flag (%n). For example

vn %n mydisk/pass: /CR/

will perform the same function but without dialogue.

If you have forgotten the password of the diskette you will not be able to use the VNEW command.

LISTING A VOLUME

There are two commands available for listing the files contained in a volume. These are VLIST and VQUICK.

LISTING A VOLUME USING THE VLIST COMMAND

If you enter the VLIST command along with an appropriate volume identifier, a list of the files on the volume is generated. Displayed with each file name is information about that file: that is, the number of bytes occupied by the file, the number of sectors used, the number of sectors allocated, the number of extents used, and whether or not the file is write-protected or password protected.

For example, to examine the contents of the system diskette you must insert it in an available drive (for instance drive 1) then enter

vl 1: /CR/

A display such as the one shown in Figure 9-1 will result.

VOLUME: 0		SECTORS			WR-PROT/ PASSWORD
	BYTES	USED	ALLOCATED	EXTENTS	
PCOS.SAV	36912	145	146	1	
basic.abs	37611	147	148	1	
basic.cmd	1600	7	8	1	
bvolume.sav	1610	7	8	1	
cl.sav	1530	6	7	1	
ckey.cmd	806	4	5	1	
dconfig.cmd	2275	9	10	1	
eprint.sav	1573	7	8	1	
fcopy.cmd	4887	20	21	1	
fdepass.cmd	1145	5	6	1	
ffree.cmd	3626	15	16	1	
fkill.cmd	1263	5	6	1	
flist.cmd	2241	9	10	1	
fmove.cmd	2849	12	13	1	
fnew.cmd	1235	5	6	1	
font.all	15888	63	64	1	
fpass.cmd	1353	6	7	1	
frename.cmd	662	3	4	1	
funprot.cmd	1591	7	8	1	
fwprot.cmd	1581	7	8	1	
ieee.sav	2583	11	12	1	
SUBTOTALS		500	521	21	
21 FILES		(HIT ANY KEY TO CONTINUE)			

Fig. 9-1 Typical Volume List of the System Diskette

By repeatedly striking a key you will step through the list one screen at a time until you reach the end of the list. At this point the M20 will display some totals, after which the PCOS prompt will re-appear.

Use of the no-interaction (%n) flag with this command causes the list to scroll continuously until the end of the list.

LISTING A VOLUME USING THE VQUICK COMMAND

Entering the VQUICK command along with a volume identifier also generates a list of the files on the volume but without any additional information about the files. For example, if you want to list the files contained on the system diskette using the VQUICK command, first insert the diskette in one of the drives (for instance drive 0) then enter

```
vq 0: /CR/
```


A display such as that shown in Figure 9-2 will result.

VOLUME 0 yourdisk Free Disk Blocks = 206				
PC05.SAV	basic.abs	basic.cmd	bvolume.sav	ci.sav
ckey.cmd	dconfig.cmd	eprint.sav	fcopy.cmd	fdepass.cmd
ffree.cmd	fkil.cmd	flist.cmd	fmove.cmd	fnew.cmd
font.all	fpass.cmd	frename.cmd	funprot.cmd	fwprot.cmd
ieee.sav	kana.sav	kb.all	label.cmd	lscreen.cmd
pkey.cmd	prun.cmd	psave.cmd	rfont.cmd	rkill.cmd
rs232.sav	sbasic.cmd	scomm.cmd	sdevice.cmd	sform.cmd
slang.cmd	sprint.cmd	ssys.cmd	valpha.cmd	vcopy.cmd
vdepass.cmd	vformat.cmd	vlist.cmd	vmove.sav	vnew.cmd
vpass.cmd	vquick.cmd	vrename.cmd	vverify.cmd	wfont.cmd

Fig. 9-2 Typical Volume Quick list of the system disk

COPYING VOLUMES

There are two commands available for copying diskettes. These are:

VCOPY - for copying a diskette from one drive to another. That is, for copying diskettes on a dual-drive system.

VMOVE - for copying diskettes on a system with only one diskette drive

Note that by using these commands it is not possible to copy a volume of a given capacity onto a volume of a different capacity. Any attempt to do so will cause an error message to be displayed: that is, you can only copy a 160 Kbyte diskette onto another 160 Kbyte diskette, a 320 Kbyte diskette onto another 320 kbyte diskette, or a 640 Kbyte diskette onto another 640 Kbyte diskette. Copy operations between different volume sizes can only be performed using the FCOPY command (or FMOVE command on a single drive system).

CAUTION: IT IS IMPORTANT THAT YOU WRITE-PROTECT YOUR SOURCE DISKETTE BEFORE COPYING IT TO AVOID ACCIDENTALLY OVERWRITING IT.

COPYING VOLUMES ON A DUAL-DRIVE SYSTEM (VOLUMES OF EQUAL SIZE)

Before copying a diskette you must first load the VCOPY command into memory using the PLOAD command (if you are copying the system diskette this is not necessary). That is, insert the system diskette into one of the drives then enter

```
pl vc /CR/
```

Remove the system diskette and insert the write-protected source diskette in one of the drives (for instance drive 0) and the target volume (which must not be write-protected) in the other drive. Then enter

```
vc 0:,1: /CR/
```

and the M20 will respond with

```
Warning- vcopy deletes all files. Copy disk? (y/n)
```

Respond by entering

```
y /CR/
```

A message such as the following will then be displayed

```
Read block 0 to 144
```

which indicates that the M20 is reading blocks 0 to 144 from the source volume. After a time this message changes to

```
Write block 0 to 144
```

which indicates that blocks 0 to 144 are being written to the target diskette. (The number of blocks read and written at once depends on the size of user memory.) When this operation is complete another "Read block" message appears for the next group of blocks. This process continues until all blocks have been copied to the target volume. At this point the message changes to

```
VCopy complete
```

and the PCOS prompt appears.

Note that in the command line the volume identifiers may alternatively be specified by name, in which case the target volume assumes the same name as the source volume once the copy operation is complete.

If the source volume is password protected then the password need not be specified, but it will also be copied to the target volume. If the target volume is password protected then its password need not be specified either.

COPYING DISKETTES USING ONE DRIVE (VOLUMES OF EQUAL SIZE)

If you intend to copy a diskette on a single-drive system you must first insert the system diskette then make the VMOVE command memory resident via the PLOAD command (if you are copying the system diskette this is not necessary). That is, you must enter

```
pl vm /CR/
```

Now remove the system diskette.

The above step is unnecessary on a hard disk system, provided the VMOVE command resides on the hard disk.

Now insert the write-protected source diskette into the drive and enter the VMOVE command. If neither the source volume nor the target volume is password protected it is only necessary to specify the command name. Simply enter

```
vm /CR/
```

If the source volume is password protected then the full source volume identifier - including the password - must be specified. For example, if your source volume is named "mydisk" and has the password "pass" then you must enter

```
vm mydisk/pass: /CR/
```

Furthermore, if the target volume is password protected then both the source and target volumes must be specified in full. For example, if your source volume is named "mydisk" and has the password "pass", and the target volume is named "yourdisk" and has the password "pass1", then it is necessary to enter

```
vm mydisk/pass:,yourdisk/pass1: /CR/
```

Once you have entered the command line the M20 responds with the following message

```
Warning- vmove deletes all files and PCOS. Vmove disk? (y/n)
```

This means that not only does the VMOVE command overwrite everything in the target volume, but in doing so it also overwrites the M20's memory thereby deleting the operating system.

Respond by entering

```
y /CR/
```

At this point the M20 will fill all its memory space with data from the source diskette. When memory is full, a message will be generated asking you to insert the target diskette and hit any key. The M20 will then transfer all the data from memory onto the target diskette.

This process will be repeated a number of times (depending on the

diskette capacity). At the end of the copy operation you can either make further copies or re-boot PCOS.

COPYING VOLUMES (OF DIFFERENT SIZES)

Where the source and destination volumes are of a different size the copy operation can only be performed using a file copy command. If the target volume is to contain only the contents of the source volume it must be blank, and formatted. For example, to copy the entire contents of a 160 Kbyte diskette inserted in drive 0 onto a 320 Kbyte diskette inserted in drive 1 enter

```
fc 0:*,1 /CR/
```

The wild card character '*' specifies all the files on volume 0.

You must take care, however, when copying from a larger volume to a smaller volume that there is enough capacity on the diskette to accommodate all files.

This operation can be performed on a single drive system using the FMOVE command but note that wild card characters may not be used; each file must be specified in a separate FMOVE command.

For further details on the FCOPY and FMOVE commands refer to Chapter 10.

NAMING AND PROTECTING A VOLUME

Volumes can be write-protected and/or password protected. Furthermore, it is often convenient to name a volume such that it can subsequently be identified by name rather than the number of the drive in which it resides.

WRITE-PROTECTION

Diskettes can be write-protected by fixing an aluminised label over the notch in the side of the diskette. Write-protection can be removed by simply removing the aluminised label. Hard disks cannot be write-protected.

PASSWORD PROTECTION

Password protection can be applied to a volume by means of the VPASS command. If you enter

```
vp 1:,pword /CR/
```

then the diskette inserted in drive 1 will be assigned the password "pword".

Similarly the (optional) hard disk can be password protected by specifying either the volume name or the drive number. In the latter case the drive number is always 10. For example, if you enter

```
vp 10:,dpass
```

then the hard disk is assigned password "dpass".

Password protection has no effect on some commands, such as VCOPY, VLIST and VQUICK, but most operations require the diskette to be enabled.

Enabling a diskette implies specifying the diskette password as part of its volume identifier as a parameter to a command. For example, if you terminate the current working session then re-boot the system and insert into drive 0 the diskette that you previously password protected, then enter

```
v1 0/pword: /CR/
```

then the files contained on that volume are listed and the volume is enabled. The volume then remains enabled until either the volume is removed from its drive and another one inserted, or the current working session is terminated.

To remove password protection from a diskette use the VDEPASS command. For example, if you enter

```
vd 0/pword: /CR/
```

then password "pword" is removed from the diskette. But note that you must know the password to be able to delete it.

Do not assign a password to your system diskette since doing so prevents you from accessing any command on that diskette. Moreover, the VDEPASS command will also be disabled and hence you will not be able to enable the volume except from a back-up.

NAMING A VOLUME

There are three commands that enable you to name a volume. These are VFORMAT, VNEW and VRENAME.

The VFORMAT and VNEW commands enable you to name a volume at the same time as formatting or initialising by specifying the volume name as the second parameter to the command. For example, if you enter

```
vf 1:,datadisk /CR/
```

then the new diskette inserted in drive 1 will be formatted and assigned the name "datadisk".

Alternatively, a used diskette (or a hard disk) can be re-initialised and assigned a name by use of the VNEW command. For example, if you enter

```
vn 1:,newname /CR/
```

then the volume inserted in drive 1 is re-initialised and assigned the name "newname".

The third method of naming a diskette (or hard disk) simply assigns a name to an existing volume. This is done using the VRENAME command. If you enter

```
vr newname:,oldname /CR/
```

then the volume you previously named "newname" will have its name changed to "oldname". Note that it is not necessary to specify an old volume name in this command, it is also valid to specify the diskette (or hard disk) by the drive number. For example, if the last operation was performed with the diskette in drive 1 it would have been equally effective to enter

```
vr 1:,oldname /CR/
```

Note that if a volume is not enabled then its password must be specified if its name is to be changed using the VRENAME command.

ALPHABETISING A VOLUME

PCOS contains a facility that enables you to sort the files contained in a volume into alphabetical order. Any unused directory blocks on the volume are removed, thereby improving access time whenever the directory is scanned. To do this you simply enter the VALPHA command along with the volume identifier. For example, if the volume named "datadisk" resides in one of the drives you can alphabetise it by entering

```
va datadisk: /CR/
```

Note that if the volume is not enabled then you must also specify the password. Furthermore, the volume must not be write-protected.

10. FILE HANDLING

ABOUT THIS CHAPTER

This chapter provides an operational guide for the use of the file directed commands.

Throughout the chapter the availability of commands is always assumed. It is assumed that either a volume containing the command is inserted in one of the drives, or that the command in question is already resident in memory.

For a detailed description of the commands mentioned in this chapter, refer to Chapter 13.

CONTENTS

<u>CREATING FILES</u>	10-1	<u>PROTECTING FILES</u>	10-8
CREATING AN EMPTY FILE	10-1	PASSWORD PROTECTION	10-8
CREATING A FILE BY COPYING	10-2	WRITE-PROTECTION	10-9
<u>COPYING FILES</u>	10-2	FREEING UNUSED FILE BLOCKS	10-10
COPYING FILES ON A DUAL DRIVE SYSTEM	10-2	<u>DELETING AND RECOVERING FILES</u>	10-14
COPYING FILES USING ONE DRIVE	10-5	DELETING FILES	10-14
<u>LISTING FILES</u>	10-6	RECOVERING DELETED FILES	10-15

RENAMING FILES

10-15

CREATING FILES

File creation within the PCOS environment can be performed either by the FNEW command which creates empty files, or simply by copying existing files using the FCOPY or FMOVE command. Text files can be created using the Video File Editor (see Chapter 12). BASIC files can be created via the Video File Editor and also from the BASIC environment (see "BASIC Language Reference Guide").

CREATING AN EMPTY FILE

Creating an empty file requires the use of the FNEW command. Before this command can be executed the volume on which the file is to be created must be enabled and must not be write-protected.

To create an empty file simply enter the FNEW command along with the file identifier and the file size (in terms of the number of blocks to be made available to the file). For example

```
fn 1:myfile,6 /CR/
```

causes an empty file named "myfile" to be created on the diskette inserted in drive 1, and 6 blocks to be allocated to it.

If the file size parameter is specified as zero, or not specified at all, then the file system allocates the number of blocks specified in the "extent size" global parameter. For example, if you enter

```
ss ,,,4 /CR/
```

```
.  
.  
.
```

```
fn 1:newfile,0 /CR/
```

then the "extent size" parameter is set to 4 by the SSYS command, and an empty file named "newfile" is created on the diskette inserted in drive 1 and consequently allocated 4 blocks.

Note that the minimum number of blocks that can be allocated is 2, and that if you specify the file size as 1 then the file system automatically allocates 2.

If there is insufficient space on the diskette to accommodate the new file then a message

```
ERROR 61 ----- disk filled
```

is displayed.

CREATING A FILE BY COPYING

Files are created when using the file copy commands if the target file does not already exist. For example, if you enter

```
fc 0:myfile,1: /CR/
```

on a dual drive system then the file named "myfile" on the diskette inserted in drive 0 will be copied onto the diskette inserted in drive 1. That is, the file named "myfile" will be created there, unless a file of that name already exists, in which case it will be overwritten. Furthermore, if you enter

```
fc 0:myfile,yourfile /CR/
```

then file "yourfile" is created (if it does not already exist) and the contents of "myfile" are copied into it.

On single drive systems the FMOVE command must be used. In this case only one file can be specified in the command line. Assuming the source diskette is already inserted in the drive the file to be copied is written into user memory. You then have to remove the source diskette, insert the target diskette and copy the contents of user memory onto the target diskette, thereby creating a file of the same name. If, however, the file is too large to fit into user memory then the operation requires a number of passes.

Further operational details are provided in the section "Copying Files".

COPYING FILES

The PCOS command library contains two commands for copying files. These are:

FCOPY - for copying files on a dual drive system

FMOVE - for copying files from one diskette to another using a single drive system

COPYING FILES ON A DUAL DRIVE SYSTEM

First of all the volume containing the file(s) to be copied must be inserted in one of the drives. To copy a file you then simply need to enter the FCOPY command specifying the file to be copied (the source file) and the destination (the target volume or target file).

Copying One File At a Time

To copy "myfile" from a volume named "myvol" inserted in drive 1 to a volume named "copyvol" inserted in drive 0 enter

```
fc myvol:myfile,copyvol: /CR/
```

and, provided "myfile" does not already exist on "copyvol" the M20 will respond

```
COPY FILE 1:myfile TO 0:myfile
```

then the file is copied and the new file is given the same name as the source file.

If the source file has a password then this is also assigned to the new file. For example, to copy a file named "datafile" and having password "dpass" from the diskette inserted in drive 0 to the hard disk you would enter

```
fc 0:datafile/dpass,10:
```

then, assuming a file named "datafile" does not already exist on the hard disk, the M20 would display

```
COPY FILE 0:datafile TO 10:datafile
```

after which a copy of "datafile" with password "dpass" would be created on the hard disk.

The FCOPY command can also copy a file to an existing file thereby overwriting the target file. For example, to copy a file named "ifile1" having password "ipass" from volume "myvol" inserted in drive 0 to file "ifile2" with password "ind" on volume "yourvol", enter

```
fc myvol:ifile1/ipass,yourvol:ifile2/ind /CR/
```

and the M20 will respond

```
COPY FILE 0:ifile1 TO 1:ifile2
```

```
File already exists. Do you wish to overwrite (y or n)?
```

If you then respond by entering

```
y /CR/
```

then the target file will end up keeping the file name "ifile2" and password "ind", but its contents will be overwritten with those of "ifile1". The PCOS prompt appears when the copy operation is complete.

It is also possible to copy a file to another file within the same volume. For example, if you enter

```
fc myvol:ifile1/ipass,copyfile/cpass /CR/
```


then, after replying "y" to the subsequent prompt, the contents of "copyfile" are overwritten by those of "ifile1". However, "copyfile" still maintains its name and password.

Note: The dialogue in any of the above examples can be bypassed by using the no-interaction (%n) flag. For example

```
fc %n myvol:newfile,copyvol: /CR/
```

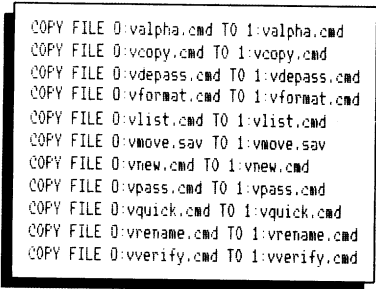
performs the copy operation without any intervening dialogue.

Copying Groups of Files using Wild Cards

A group of related files can be copied from one volume to another using the wild card facility. To do this you must enter the FCOPY command along with the group of files specified using wild cards, and the target volume. For example, you can copy all the volume-directed commands from the system diskette to another volume using one command by inserting the system diskette in one of the drives (for instance, drive 0), having the target volume inserted in the other and entering

```
fc 0:v*,1: /CR/
```

The M20 will then respond with the display shown in Figure 10-1.



```
COPY FILE 0:\alpha.cmd TO 1:\alpha.cmd  
COPY FILE 0:\vcopy.cmd TO 1:\vcopy.cmd  
COPY FILE 0:\vdepass.cmd TO 1:\vdepass.cmd  
COPY FILE 0:\vformat.cmd TO 1:\vformat.cmd  
COPY FILE 0:\vlist.cmd TO 1:\vlist.cmd  
COPY FILE 0:\vmove.sav TO 1:\vmove.sav  
COPY FILE 0:\vnew.cmd TO 1:\vnew.cmd  
COPY FILE 0:\vpass.cmd TO 1:\vpass.cmd  
COPY FILE 0:\vquick.cmd TO 1:\vquick.cmd  
COPY FILE 0:\vrename.cmd TO 1:\vrename.cmd  
COPY FILE 0:\vverify.cmd TO 1:\vverify.cmd
```

Fig. 10-1 Sample Display of a Copy Operation on a Group of Files

The copy operation is then complete. In this case none of the files already existed on the target volume. For each file that had existed on

the target volume the prompt

File already exists. Do you wish to overwrite (y or n)?

would be displayed.

Password protected files will not be copied by this procedure, unless the password is specified in the command line and is common to all files in the group.

Note: The dialogue in all the above examples can be bypassed by entering the no-interaction (%n) flag in the command line. In this case the system simply returns the PCOS prompt when the copy operation is complete.

COPYING FILES USING ONE DRIVE

Copying Files Diskette to Diskette

To copy files from one diskette to another on a single drive or hard disk system you must enter the FMOVE command along with the file name. For example, to copy "myfile" from the diskette named "mydisk" to the diskette named "archive" you would first enter

```
fm myfile /CR/
```

and the M20 will respond with

Please put SOURCE disk in drive,
then press any key (CTRL C to abort)

Respond by inserting the diskette named "mydisk" in the drive and hitting any key. A message such as the following will then be displayed

File transfer will take two pass(s)
Please put DESTINATION disk in drive,
then press any key (CTRL C to abort) (pass 1 of 2):

Respond as instructed and the file will be copied into user memory.

The above message indicates that it will require two passes to transfer the file. To transfer the file the M20 must first copy it into user memory, then ask you to insert the target diskette, and finally transfer the file to the target volume. In this case, however, the file is too large to fit into user memory, therefore the transfer requires two passes.

Now insert the target diskette "archive" and hit any key as instructed. A file named "myfile" will be created on the target diskette (if it does not already exist) and the contents of the user memory copied into it. When this is completed the following message will be displayed

Please put SOURCE disk in drive,
then press any key (CTRL C to abort) (pass 2 of 2):

Respond by removing the diskette, inserting the source diskette and hitting any key, as instructed, then the remainder of the file will be copied into user memory. When this is completed the following message will be displayed

Please put DESTINATION disk in drive,
then press any key (CTRL C to abort) (pass 2 of 2):

Respond as instructed and the remainder of the file will be copied from user memory onto the target diskette and the process is then complete. The M20 responds with the PCOS prompt.

Note that it is not possible to use the wild card facility.

Copying Files on the Same Diskette

This process is the same as for copying files on the same diskette on a dual drive system. For example, to overwrite the contents of a file named "oldfile" with the contents of "myfile" - both being resident on the same diskette - insert the diskette in the drive and enter

```
fc myfile,oldfile /CR/
```

and the M20 responds

```
COPY FILE 0:myfile TO 0:oldfile
File already exists. Do you wish to overwrite (y or n)?
```

Respond by entering

```
y /CR/
```

and the copy process is executed. The target file keeps its original name. Furthermore, a password protected target file will also keep its own password. For example, if you enter

```
fc %n dfile/passa,efile/passb /CR/
```

then "efile" is overwritten with the contents of "dfile", but the target file maintains its name "efile" and password "passb". Use of the no-interaction (%n) flag bypasses the dialogue.

LISTING FILES

To list a text file simply enter the FLIST command followed by the file identifier. For example, to list the file named "workfile" that is resident on the the diskette inserted in drive 1 enter


```
f1 1:workfile /CR/
```

and a display such as the following will be generated.

```
LISTING FILE 1:workfile
10 subname$="getvolname"
20 volname$="12345678901234"
30 call "bv"(subname$,@volname$)
40 print "the current volume is named", volname$
50 end
```

Fig. 10-2 Sample Display of a Text File List

Moreover, any file can be listed in hexadecimal form, 16 bytes per line, by including the %h program flag in the command line. For example, if you enter

```
f1 %h 1:workfile /CR/
```

then the following display will be generated (in 80 by 25 display mode):

```
LISTING FILE 1:workfile
0000: 31 30 20 73 75 62 6E 61 6D 65 24 30 22 67 65 74 10 subname$="get
0010: 76 6F 6C 6E 61 6D 65 22 00 0A 32 30 20 76 6F 6C volname$..20 vol
0020: 6E 61 6D 65 24 30 22 31 32 33 34 35 36 37 38 39 name$="123456789
0030: 30 31 32 33 34 22 00 0A 33 30 20 63 61 6C 6C 20 01234"..30 call
0040: 22 62 76 22 28 73 75 62 6E 61 6D 65 24 2C 40 76 "bv"(subname$,@v
0050: 6F 6C 6E 61 6D 65 24 29 00 0A 34 30 20 70 72 69 olname$)..40 pri
0060: 6E 74 20 22 74 68 65 20 63 75 72 72 65 6E 74 20 nt "the current
0070: 76 6F 6C 75 6D 65 20 69 73 20 6E 61 6D 65 64 22 volume is named"
0080: 38 20 76 6F 6C 6E 61 6D 65 24 00 0A 35 30 20 65 ; volname$..50 e
0090: 6E 64 00 0A 00 0A nd....
```

Fig. 10-3 Sample Display of a Hexadecimal File List

The first four columns specify the byte address relative to the

beginning of the file (in hexadecimal) of the first character in the line. Each such entry is followed by the hexadecimal codes for the 16 bytes subsequent to the displayed address. The right-hand columns show the corresponding ASCII characters, or '.' where the character is non-printable (octal characters 0 to 31 and 127).

The text or hexadecimal contents of password protected files can be displayed, but the password must be specified to do so.

The contents of more than one file can be displayed by specifying a list of files in the command line. Moreover, groups of files can be listed by use of the wild card facility. However, password protected files can be displayed in this manner only if the password is common to all the files in the group.

Remark

A text listing is only useful where the specified file is a text file, but a hexadecimal listing of any file can be obtained.

PROTECTING FILES

PCOS files can be password protected to limit access to only those who know the password. Furthermore, they can be write-protected to prevent accidental damage to the contents of a file.

PASSWORD PROTECTION

Assigning and Removing a Password to a File

To assign a password to a file, enter the FPASS command along with the file identifier and the password to be assigned. For example, to assign the password "pass" to the file named "mine" that resides on the hard disk simply enter

```
fp 10:mine,pass /CR/
```

At some later time, you may wish to remove this password protection. To do so requires the use of the FDEPASS command.

To use the FDEPASS command simply enter the command mnemonic along with the file identifier, including the password. In this case enter


```
fd 10:mime/pass /CR/
```

Note that you must know the password to be able to remove it.

Assigning and Removing a Password to a Group of Files

To assign a password to a group of files is the same as for a single file except that the wild card feature is used. For example, to assign the password "rst" to all files beginning with "data..." on a diskette named "datadisk" you would enter

```
fp datadisk:data*,rst /CR/
```

and the M20 would respond

```
Set Password on 0:data1
```

Then a "y" response will set the password of the file named "data1" and corresponding interactive messages will be displayed for each subsequent file in the group. The PCOS prompt appears when the operation is complete.

To subsequently remove this password you would enter

```
fd datadisk:data*/rst /CR/
```

and an interactive prompt for each file in the group appears thus:

```
Delete Password of 0:data1?
```

A response of "y" deletes the corresponding password and the prompt reappears for the next file in the group. The PCOS prompt appears when the operation is complete.

To remove passwords in this way the password must be common to all the files in the group. Note: The dialogue in the above examples will be suppressed if the no-interaction (%n) flag is used.

WRITE-PROTECTION

Applying and Removing Write-Protection to a File

To apply write-protection to a file enter the FWPROT command along with the file identifier. For example, to write-protect a file named "masterfile" on a the diskette inserted in drive 1 enter

```
fw 1:masterfile /CR/
```


To subsequently remove write-protection requires the FUNPROT command to be entered along with the file identifier. For example, to remove write-protection from "masterfile" enter

```
fu 1:masterfile /CR/
```

Applying and Removing Write-Protection to a Group of Files

To write-protect a group of files, For example, all files beginning with "index" on the diskette inserted in drive 1, enter

```
fw 1:index* /CR/
```

then the M20 displays a prompt for each file in turn in the specified group asking you to confirm or decline write-protection

```
Set WP 1:index1?
```

Respond "y" and write-protection is set for the file named "index1"

```
Set WP 1:index3.5?
```

A "y" response sets write-protection for the file named "index 3.5", etc.

To remove write-protection from the above group of files, enter

```
fu 1:index* /CR/
```

and the M20 responds with a sequence of interactive prompts requesting confirmation for each file in the group. The PCOS prompt is displayed when the operation is complete.

Note: The dialogue in the above examples will be suppressed if the no-interaction flag (%n) is used.

FREEING UNUSED FILE BLOCKS

This section describes how to free blocks that are allocated to files but not used.

To free unused blocks of a file, group of files or an entire volume requires the FFREE command. Before this command can be executed the volume containing the files to be worked on must be present in one of the drives. It must also be enabled and must not be write-protected.

Before performing the FFREE command take a look at the contents of the volume to be worked on. For example, if you enter

vl 1: /CR/

then a volume list of the contents of the diskette inserted in drive 1 will be displayed and will look something like the following:

VOLUME: mydisk		SECTORS			WR-PROT/
	BYTES	USED	ALLOCATED	EXTENTS	PASSWORD
kfile	3589	15	16	1	WP
kfile1	3589	15	19	1	WP
asd	1687	7	14	1	/PW
sdf	1687	7	14	1	
qwe	1687	7	14	1	
basfile	32512	127	150	1	
TOTALS		178	150	6	
6 FILES	FREE SECTORS ON DISK = 861				

Fig. 10-4 Sample Volume List before Freeing Unused Blocks

To free unused blocks in the entire volume perform the following:

Enter

ff 1: /CR/

And the M20 will respond

You may not change disks while FFree in progress. Continue?

Respond by entering

y /CR/

and the display shown in Figure 10-5 will result.


```

kfile . . . . . write-protected
kfile1 . . . . . write-protected
asd . . . . . ERROR 73 --- invalid password

sdf . . . . . 6 block(s) freed
qwe . . . . . 6 block(s) freed
basfile . . . . . 22 block(s) freed

FFree complete -- 34 block(s) freed
1>

```

Fig. 10-5 Example of a Display Produced by an FFREE Command

The unused blocks allocated to "kfile" and "kfile1" are not freed because these files are write-protected. "asd" did not have unused blocks freed because it is password protected and did not have its password specified.

To free the unused blocks from file "asd" another FFREE command is necessary specifying the complete file identifier including the password. That is, if you enter

```
ff 1:asd/pass /CR/
```

M20 will display

You may not change disks while FFree in progress. Continue?

Respond by entering

```
y /CR/
```

then the unused blocks will be freed and the M20 will display

```

asd . . . . . 6 block(s) freed

FFree complete -- 6 block(s) freed
1>

```

To free the unused blocks in files "kfile" and "kfile1" you must first remove the write-protection by entering

```
fu 1:k* /CR/
```

then, after answering "y" to the subsequent confirmation prompts, free

the unused blocks by entering

```
ff 1:k* /CR/
```

M20 will respond

You may not change disks while FFree in progress. Continue?

respond by entering

```
y /CR/
```

and the M20 will display

```
kfile . . . . . 0 block(s) freed
kfile1 . . . . . 3 block(s) freed

FFree complete --      3 block(s) freed
1>
```

The total effect of freeing the unused blocks can be seen by examining a volume list and comparing it to the previous one. That is, if you enter

```
vl 1: /CR/
```

then the following will be displayed:

VOLUME: mydisk		SECTORS			WR-PROT/
	BYTES	USED	ALLOCATED	EXTENTS	PASSWORD
kfile	3589	15	16	1	
kfile1	3589	15	16	1	
asd	1687	7	8	1	/PW
sdf	1687	7	8	1	
qwe	1687	7	8	1	
basfile	32512	127	128	1	
TOTALS		178	184	6	
6 FILES		FREE SECTORS ON DISK = 904			

Fig. 10-6 Sample Volume List after Freeing Unused Blocks

Finally, write-protection should be re-applied to files "kfile" and "kfile1".

DELETING AND RECOVERING FILES

Deleting a file implies freeing the disk space occupied by that file for use by other files. The actual file data, however, is not deleted until the space it occupied is actually overwritten. This makes it possible to recover a deleted file so long as the file has not yet been overwritten and the volume has not been alphabetised.

DELETING FILES

To delete a file or group of files requires the FKILL command. Before this command can be executed the following conditions must apply:

- The volume containing the file or files to be deleted must be inserted in one of the drives. It must also be enabled and not write-protected
- The file or files to be deleted must not be write-protected

To delete a file from a volume requires you to enter the FKILL command along with the file identifier including any necessary volume identifier and/or password. For example, to delete a file that has the name "myfile" and password "mine" from the diskette inserted in drive 1 enter

```
fk 1:myfile/mine /CR/
```

A list of files can be specified for deletion. For example, to delete the (unprotected) files "myfile" and "yourfile" from the diskette inserted in drive 0 enter

```
fk 0:myfile,yourfile /CR/
```

A killed file is not actually deleted from the diskette, but it can no longer be accessed by any command other than the RKILL command. Furthermore, its name will no longer feature in a volume list as it is erased from the directory, and the space it occupied will be available for other files.

Groups of files can be deleted using the wild card feature. For instance to delete from the diskette inserted in drive 0 all the files beginning with "k" you must enter

```
fk 0:k* /CR/
```

and the M20 will respond

```
Delete 0:kfile?
```

which gives you the option whether or not to delete this particular file. If you enter

y

then the file is deleted and the M20 responds

```
Delete 0:kfile1?
```

and so on through the complete list of files beginning with "k", after which the PCOS prompt appears. Note: The dialogue can be by-passed by specifying the no-interaction (%n) flag.

RECOVERING DELETED FILES

To recover a deleted file requires the RKILL command. For this command to be executed successfully the following conditions must apply:

- The volume containing the file to be recovered must be inserted in one of the drives
- The deleted file to be recovered must still be intact. That is, it must not have been fully or partially overwritten
- The volume must not have been alphabetised since the file was killed

To recover a file requires you to enter the RKILL command along with the file identifier. It is not necessary to enter the password. For example, to attempt to recover the deleted file "myfile" on the diskette inserted in drive 1 enter

```
rk 1:myfile /CR/
```

If the recovery is successful then the M20 will respond

```
File Successfully Repaired
```

If the recovery was not successful then the following error message is displayed:

```
ERROR 53 --- file not found in parameter 1
```

Note: It is not possible to use the wild card facility, neither is it possible to specify a list of files.

RENAMING FILES

Renaming a file requires the FRENAME command. Before this command can be executed the file to be renamed must reside on a volume that is currently inserted in one of the drives.

To rename a file requires you to enter the file identifier along with the new file name. The file must not be write-protected.

For example, to change to "oldfile" the name of a file named "newfile"

with password "npass" that is resident on a disk named "datadisk" enter

fr datadisk:newfile/npass,oldfile /CR/

Note: FRENAME has no effect on the password.

11. PCOS GRAPHIC AND CONSOLE - RELATED FACILITIES

ABOUT THIS CHAPTER

This chapter describes the graphics facilities that are available within the PCOS environment. For further details of the commands mentioned in this chapter, refer to Chapter 13.

CONTENTS

<u>INTRODUCTION</u>	11-1	<u>THE FONT MATRIX FILE</u>	11-11
<u>DISPLAYING LABELS</u>	11-1	<u>USING THE WFONT COMMAND</u>	11-14
<u>PRINTING THE SCREEN IMAGE</u>	11-3	<u>DISPLAYING CONTROL CHARACTERS</u>	11-15
<u>USING THE SPRINT AND LSCREEN COMMANDS</u>	11-4	<u>THE LINE TERMINATION KEYS</u>	11-19
<u>ENTERING CHARACTERS AT THE KEYBOARD</u>	11-5		
RAW KEY CODES	11-5		
ASCII TABLES	11-6		
INTERPRETATION OF ASCII TABLE OUTPUT	11-8		
USING THE PKEY COMMAND	11-9		
<u>CREATING USER-DEFINED FONTS</u>	11-10		
<u>CREATING A FONT MATRIX USING THE RFONT COMMAND</u>	11-11		

INTRODUCTION

PCOS allows you to set the display mode to either 256 x 512 pixels (64 columns x 16 lines) or 256 x 480 pixels, (80 columns x 25 lines) by means of the SSYS command, and to allocate space for a certain number of windows using the SBASIC command. Moreover, PCOS contains a number of graphic and console-related facilities that enable you to:

- display a label. That is, to display a string of characters of a specified magnification and orientation at a specified position, and within a specified window (LABEL command)
- print out the entire contents of the video display or a specified window (SPRINT command)
- print out just the textual content of a video display or a specified window (LSCREEN command)
- modify the ASCII code generated on striking a specific key or key combination (CKEY command)
- reconfigure the keyboard to simulate another standard national layout (SLANG command)
- assign a string to a key or key combination (PKEY command)
- create user-defined fonts (RFont and WFont commands)
- display control characters
- distinguish, in BASIC, among the three line termination keys /↵/, /S1/ and /S2/ (LTERM command)

Note: The LABEL, SPRINT, and LSCREEN commands are often called from BASIC by a CALL or EXEC statement, as windows cannot be opened within the PCOS environment.

DISPLAYING LABELS

Labels can be displayed by using the LABEL command. This command enables you to specify the following features of the label to be displayed.

FEATURE	MEANING
the string	the text to be displayed. This can be any string of printable characters, included in quotation marks
position	the x/y co-ordinates of the bottom left-hand corner of the first character of the label string with respect to the bottom left-hand corner of the screen or window. This is measured in pixels
magnification	the number of times that the font character is magnified. This can be up to 16 times
orientation	the number (0, 1 or 2) specifying the direction of the label string; that is, parallel to the x-axis left to right (0), parallel to the y-axis bottom to top (1), or top to bottom (2)
colour	the colour number in the range 0 to 7 for an eight colour display, 0 to 3 for a four colour display, or 0 or 1 for a black and white display. If omitted, the foreground colour is assumed

Using the LABEL command

If you enter

```
1a 'LABEL',100,150,4,2 /CR/
```

then the string "LABEL" is displayed in the foreground colour, at

(100,150) four times the normal size, with the text rotated as follows:

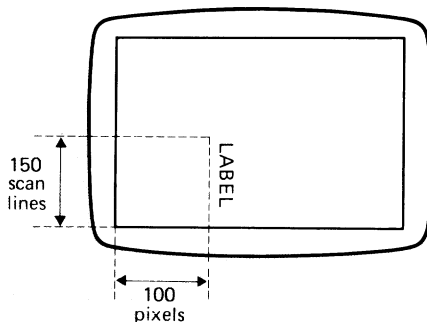


Fig. 11-1 Example of the LABEL command

PRINTING THE SCREEN IMAGE

There are two PCOS commands that perform this type of function. These are:

SPRINT which prints an image of all text and graphics displayed on the screen or within a specified window. With this command you are also able to specify the polarity of the print-out, (that is, n for negative - black on white on print-out for white on black on display, p for positive - white on black on print-out for white on black on display). You can also specify a title to appear at the top of the print-out along with the date and time.

The SRRINT command can, however, only be used with printers that have graphic capabilities.

LSCREEN which prints just the text displayed on the screen or specified window. Graphic elements are ignored.

The LSCREEN command can be used with any PCOS-compatible printer

USING THE SPRINT AND LSCREEN COMMANDS

Supposing, when working in the BASIC environment, you have segmented the screen into five windows as follows:

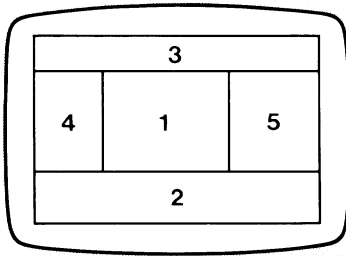


Fig. 11-2 Example of a Windowed display

IF you are in BASIC and you enter ...	THEN...
EXEC "sp 5,p,'SALES', dt" /CR/	the contents of window 5 will be printed with positive polarity beneath the heading "SALES" and the date and time (specified by dt)
EXEC "sp 0" /CR/	prints the contents of the entire screen
EXEC "ls 4" /CR/	prints the text contained in window 4. Any graphic elements are ignored

For details of how to define windows refer to the "BASIC Language Reference Guide".

ENTERING CHARACTERS AT THE KEYBOARD

Figure 11-3 illustrates what happens when you press a key.

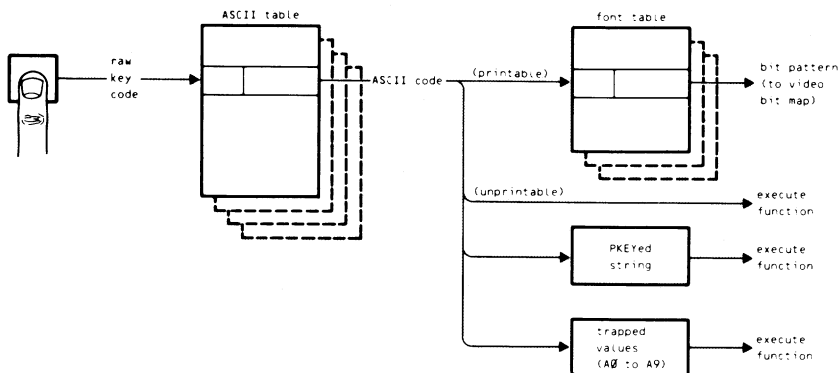


Fig. 11-3 Entering a Character at the keyboard

RAW KEY CODES

When a key is pressed it generates a raw key code (see Figure 11.4). This code depends on the physical position of the key on the keyboard, and on whether the key is pressed on its own, or in conjunction with the /SHIFT/ key, the /CTRL/ key, or the /COMMAND/ key. For example, if you press the 'A' key on the USA ASCII keyboard then raw key code (hexadecimal) 02 will be generated. The same raw key code will be generated by pressing 'Q' on the French keyboard since the code depends on the physical position of the key, not the key inscription. Moreover, pressing this key in conjunction with the /SHIFT/ key will generate raw key code (hexadecimal) 32, with the /CTRL/ key will generate (hexadecimal) 62, and with the /COMMAND/ key will generate (hexadecimal) 92.

and the second the ASCII code to be assigned to it. For example

```
ck &C3,8 /CR/
```

assigns ASCII code 8 (backspace) to the raw key code (hexadecimal) C3, thereby enabling backspace to be performed by striking the S2 key. Note that values may be entered either in decimal (by entering the number on its own, or in hexadecimal) by preceding the number with &. Furthermore, the ASCII code can be specified simply by striking the corresponding key enclosed within quotation marks. For example

```
ck &72,'p' /CR/
```

will cause the ASCII code for 'p' to be generated when /CTRL/ /Q/ are pressed simultaneously.

The ASCII code assigned to a particular raw key code can be examined, again by use of the CKEY command by specifying just one parameter - the raw key code. For example, if you wish to examine the code assigned to the key combination /CTRL/ /C/, enter

```
ck &64 /CR/
```

then PCOS responds

```
KEY = 100
CODE = 162
```

The values displayed are decimal. That is, KEY specifies the raw key code (100 decimal being the equivalent of 64 hexadecimal), and CODE indicates the assigned code.

Using the SLANG Command

The ASCII table can be replaced at will by the ASCII table corresponding to another national keyboard by means of the SLANG command. Doing so also destroys any values assigned using the CKEY command. If you enter

```
sl /CR/
```

then a menu is displayed enabling you to select a national configuration by entering the appropriate number (see Chapter 6 for details). Alternatively, the selection can be made by entering the appropriate number as a parameter to the SLANG command. For example

```
sl 0 /CR/
```

selects the Italy keyboard. The result is that the ASCII table corresponding to the Italy keyboard is loaded from the kb.all file and overwrites the currently active table.

Making the Current ASCII Table Permanent

Any modifications made to the ASCII table by means of the CKEY command, or any ASCII table made active by means of the SLANG command remain active either until the current working session is terminated, or until further modified by a CKEY or SLANG command. However, such changes can be made a permanent feature of the operating system by means of the PSAVE command (see Chapter 6); that is, any CKEYed values will become permanent, as will any ASCII table loaded by means of the SLANG command.

INTERPRETATION OF ASCII TABLE OUTPUT

The output from the ASCII table is treated as follows:

- values A0 to A9 (hexadecimal) are special cases. They are never placed in the keyboard buffer but are always trapped by the keyboard handler to perform the following functions:
 - . A0 - Logical Reset
 - . A1 - (reserved)
 - . A2 - Break facility
 - . A3 - Halt Display
 - . A4 - Cursor lock
 - . A5 - Shift lock
 - . A6 - Two zeros
 - . A7 - End of line (CR in keyboard buffer, zero in LTERM buffer)
 - . A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
 - . A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
 - . AA - Special function for DATEV keyboard
 - . AB - Special function for DATEV keyboard
 - . AC - Special function for DATEV keyboard
 - . AD - (reserved)
 - . AE - (reserved)
 - . AF - No operation
- for values that have a PKEYed string assigned to them the value is placed in the buffer and the corresponding function is subsequently

executed

- for unprintable ASCII values (codes (hexadecimal) 00 to 1F and 7F) the code is placed in the keyboard buffer and the corresponding function is subsequently performed
- for printable ASCII characters other than those that have strings assigned by means of the PKEY command (codes (hexadecimal) 20 to 7E plus any additional CKEYed values) the code is placed in the keyboard buffer and the corresponding entry in the currently active font table is accessed and the bit pattern is written to the video bit map - see the section entitled "Creating User-Defined Fonts"

USING THE PKEY COMMAND

Any code output from the ASCII table (with the exception of the special cases A0 to AF) can have a string assigned to it by means of the PKEY command, thereby ensuring that the original function is not destroyed.

Assignment is made by passing parameters to the PKEY command, the first of which specifies the code as output from the ASCII table, and the second and subsequent parameters define the string to be assigned.

The ASCII code can be specified either as a decimal value on its own, a hexadecimal value preceded by an ampersand (&), or in the case of an ASCII code that already has a corresponding font defined in the currently active font table, the character can be entered directly from the keyboard, but enclosed within quotation marks. For example

'B'

66

&47

all refer to the same key.

Similarly, the strings to be assigned to the ASCII code can be specified as either the actual characters enclosed within quotation marks, or the ASCII code for each character, or a combination of the two. For example

'ba',13

is a valid string representing 'ba' followed by a carriage return.

Suppose that you wish to assign the string 'FILES "1:"',13 to the key combination /COMMAND/ /!1/ (ASCII code hexadecimal ED, decimal 237). Do this by entering

pk 237,'FILES "1:"',13 /CR/

Thus when in the BASIC environment, the key combination /COMMAND/ /!1/ can be used to list the directory on the diskette inserted in drive 1.

String assignments made in this manner are valid up to the end of the current working session. However, such assignments can be made a permanent feature of the operating system by means of the PSAVE command as described in Chapter 6.

The template that fits into the slot above the top row of the keyboard can be used as a memory aid to the keys that you have programmed. Alternatively you can display a list of programmed keys along with the string assignments by entering

```
pk /CR/
```

PCOS will typically respond:

Code	Char	String
35	#	ba, 13, 10, files, 13, 10
237		FILES "1:", 13
(Press any key to exit)		

Fig. 11-5 Sample Display of Programmed Keys

Note that the code is given in decimal.

An individual key assignment may be cancelled by entering the ASCII code as a single parameter to the PKEY command. For example

```
pk 237 /CR/
```

Moreover, all assigned strings may be cancelled by entering

```
pk %c /CR/
```

For details of the effect string assignments have on user memory refer to Chapter 6.

CREATING USER-DEFINED FONTS

The RFONT command creates a graphic representation of the active font set and stores it in a font matrix file. This file can be edited using the Video File Editor (see Chapter 12) to redefine the shape of existing characters and add new characters. Once the editing session is complete you can invoke the WFONT command to cause the system to display character fonts as they appear in the edited file. Thus you can create customised font sets, each of which is stored on diskette (or hard disk) in a

separate file, and can be selected to become the active font set.

CREATING A FONT MATRIX FILE USING THE RFONT COMMAND

The RFONT command is invoked by entering, for example

```
rf 1:myfont /CR/
```

The specified file will be created if it does not exist. If the file already exists, the following prompt will appear:

```
File Already Exists. Do You Wish to Overwrite? (y/n)
```

A "y /CR/" response causes the existing file to be overwritten.

THE FONT MATRIX FILE

A font matrix file must be structured as defined below. A file created by the RFONT command is of this structure.

At the beginning of a font matrix file is a four line header. All four lines must be present, although only the fourth line is actually read by the WFONT command. The header is defined as follows:

- line 1: Any text that describes the file (for example, the national keyboard that the file corresponds to). It is for your reference and is ignored by the WFONT command
- line 2: The country number that was active when the file was created by the RFONT command. This number corresponds to the particular national keyboard. For details refer to the SLANG command. The content is ignored by the WFONT command
- line 3: The height (in lines) of a valid font matrix. This must always be 10
- line 4: The character count followed by at least one other word (for example, "characters"). The count must match the number of font matrices that follow. The minimum is 95 characters and the maximum is 190

Example:

```
USA
country 4
matrix height = 10
95 characters
```

Each matrix is defined as follows:

- line 1: A decimal code representing the character defined in the matrix. (For standard fonts this will be the ASCII code.) Its value is for your reference and is not read by the WFONT command, but it must be present
- lines 2 to 11: A matrix, ten lines down and eight characters wide, made up of ' - ' and (upper case) 'X' characters

Example:

```
50
-----
----XXX-
---X---X
-----X
-----X-
-----X--
-----X---
---XXXXX
-----
-----
```

The correspondence between a font matrix and the ASCII code generated by a particular key on the keyboard depends on the position of the font matrix within the file. That is, the first font matrix will correspond to /SPACE/ - the first printable ASCII character (ASCII code 32) - the second to /!/ (ASCII code 33) and so on up to the 95th entry which will correspond to /±/ - the last printable ASCII character (ASCII code 126).

Redrawing Existing Characters

To redraw a character, invoke the Video File Editor and place 'X's so that they show the intended appearance of the character.

In 64 x 16 display mode, the left most three columns are generally reserved for spacing between characters and should not be used except in special cases where regular spacing between characters is not desired. In 80 x 25 display mode, the left most two columns are not displayed at all, and the third column should be left blank unless joined characters are desired.

Once the edited file is saved and the Video File Editor exited, the new

font can be made active by means of the WFONT command.

Defining Additional Characters

Characters can be added to a font matrix file by incrementing the character count (line 4 of the header) and adding matrices to the end of the file. However, matrices beyond the 95th, (that is, those corresponding to codes 127 to 222) do not necessarily have a corresponding key-generated ASCII code. You must therefore define a key combination for each additional matrix using the CKEY command.

Example:

To add a font matrix, defining the character '0', to the standard font matrix file and assign key combination /COMMAND/ /Q/ to it:

1. Using the Video File Editor update the character count (line 4 of the header) to 96
2. Using the Video File Editor add a font matrix to the end of the file, thus:

```

127
-----
----XXX-
---X---X
---X--XX
---X-X-X
---XX--X
---X---X
----XXX-
-----
-----

```

3. Assign ASCII code 127 to the key combination /COMMAND/ /Q/ using the CKEY command:

```
ck &A2, 127 /CR/
```

This assigns code 127 to raw key code (hexadecimal) A2, which is the raw key code generated when /COMMAND/ /Q/ are pressed simultaneously

Once the edited font has been made active by means of the WFONT command, the key combination /COMMAND/ /Q/ will display '0'.

Note: It is possible to remove or insert font matrices within the first 95 characters, but this would offset the matrix/key correspondence following the first matrix to be removed/inserted.

USING THE WFONT COMMAND

The WFONT command takes one parameter which is the name of a font matrix file. After execution, the font defined by that file becomes active.

Invoke the WFONT command by entering, for example

```
wf 1:myfont /CR/
```

Assuming the specified font matrix file is on an active volume, and that enough memory is available, the font matrices will be read, converted to binary, and written into memory. Once execution is completed, the new fonts will replace those currently known to the system. The system will return to the fonts known at initialisation when re-initialisation occurs, or when the WFONT command is invoked with no parameter.

The WFONT command allocates user memory each time it is invoked with a valid font file. This memory is released either by re-initialising PCOS, or by invoking the WFONT command with no parameter. In order to save memory, it is advisable to release space allocated by the WFONT command before activating another user-defined font. The effect on memory can be determined using the DCONFIG command.

User-defined fonts can be made permanent by means of the PSAVE command.

Example

The following example assumes the existence of two user-defined font matrix files named "myfont" and "yourfont." Both files are located on the diskette inserted in drive 1.

If you enter...	THEN...
wf 1:myfont /CR/ . . .	the font defined by the font matrix file named "myfont" is made active
wf /CR/ . . .	"myfont" is removed from memory and the font that was active at initialisation is again made active

<pre>wf 1:yourfont /CR/ . . .</pre>	the font defined by the font matrix file named "yourfont" is made active
<pre>wf 1:myfont /CR/ . . .</pre>	"myfont" is again made active, but data for "yourfont" is not removed from memory

The effect on user memory is illustrated in Figure 11-6

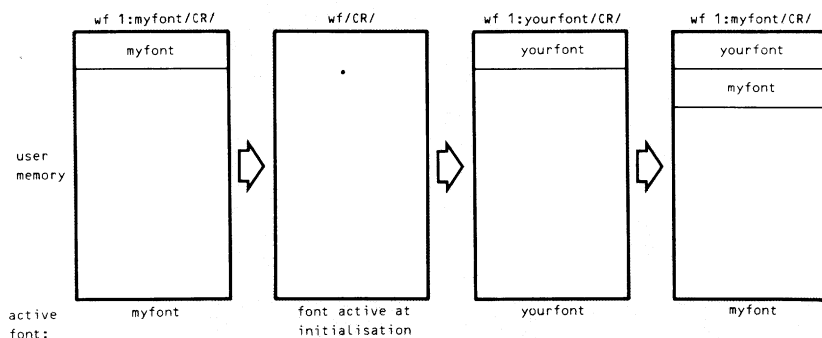


Fig. 11-6 Effect of the WFONT command on user memory

DISPLAYING CONTROL CHARACTERS

When displaying files or displaying data received from a communications line, you may wish to have a visual representation of the ASCII control characters (hexadecimal 00 to 0F). PCOS contains a facility whereby such characters can be represented by the character font definitions illustrated in Figure 11-7. Above each matrix in the figure is the corresponding hexadecimal ASCII code in parentheses, and the control function.

(00) NUL	(01) SOH	(02) STX	(03) ETX	(04) EOT	(05) ENQ
-----	-----	-----	-----	-----	-----
---XXXX	---XXXX	---X---	---X---	---X---	---XXXX
---XXXX	---X---	---X---	---X---	---X---	---X---
---XXXX	---X---	---X---	---X---	---X---	---XX-XX
---XXXX	---X---	---X---	---X---	---X---	---X-X-X
---XXXX	---X---	---X---	---X---	---X---	---XX-XX
---XXXX	---X---	---X---	---X---	---X---	---X-X-X
---XXXX	---X---	---XXXX	---XXXX	---X---	---XXXX
-----	-----	-----	-----	-----	-----
-----	-----	-----	-----	-----	-----
(06) ACK	(07) BEL	(08) BS	(09) HT	(0A) LF	(0B) VT
-----	-----	-----	-----	-----	-----
-----X	-----	---XXX-	---X---	---XXXX	-----X-
-----	-----	---XX-	---X---	-----	-----X-
-----X-	-----	---X-X-	---X---	-----	-----X-
-----	---XXX-	---X-	---XXXX	---XXXX	-----X-
---X-X-	---X-X	-----X	---X---	-----	---X-X-X
---X---	---X-X	-----X	---X---	-----	---XXX-
---X---	---XXXX	-----X	---X---	---XXXX	-----X-
-----	-----	-----	-----	-----	-----
-----	-----	-----	-----	-----	-----
(0C) FF	(0D) CR	(0E) SO	(0F) SI	(10) DLE	(11) DC1
-----	-----	-----	-----	-----	-----
---X-	---X-	---XXX-	---XXX-	---XXXX	---XXX-
---X-X-X	---X-	---X-X-X	---X-X-X	---X-X-X	---X-X-X
---XXX-	---X-	---XX-XX	---X-X-X	---X-X-X	---X-X-X
---X-	---XXXX	---X-X-X	---X-X-X	---XXXX	---X-XXX
---X-X-X	---X-	---XX-XX	---X-X-X	---X-X-X	---X-X-X
---XXX-	---X-	---X-X-X	---X-X-X	---X-X-X	---X-X-X
---X-	---X-	---XXX-	---XXX-	---XXXX	---XXX-
-----	-----	-----	-----	-----	-----
-----	-----	-----	-----	-----	-----
(12) DC2	(13) DC3	(14) DC4	(15) NAK	(16) SYN	(17) ETB
-----	-----	-----	-----	-----	-----
---XXX-	---XXX-	---XXX-	-----X	---XXX-	-----
---X-X-X	---X-X-X	---X-X-X	-----	---X-X-	-----X
---X-X-X	---X-X-X	---X-X-X	---XXX	---X-X-	-----X
---X-XXX	---XXX-X	---XXX-X	-----	---X-X-	---XXX-
---X-X-X	---X-X-X	---X-X-X	---X-X-	---X-X-	-----X
---X-X-X	---X-X-X	---X-X-X	-----	---X-X-	-----X
---XXX-	---XXX-	---XXX-	---X-	---XX-XX	-----
-----	-----	-----	-----	-----	-----
-----	-----	-----	-----	-----	-----

(18) CAN	(19) EM	(1A) SUB	(1B) ESC	(1C) FS	(1D) GS
-----X-----	-----X-----	-----XXX-----	-----XXX-----	-----XXXXX-----	-----XXXXX-----
---X---X	---X---X	---X---X	---X---X	---X-X-X	---X---X
---X-X-	---X-X-	---X-X-	---X-X-X	---X-X-X	---X---X
---X-	---XXX-	---X-	---XXXXX	---XXX-X	---XXX-X
---X-X-	---X-	---X-	---X-X-X	---X-X-X	---X-X-X
---X---X	---X-	---	---X-X-X	---X-X-X	---X-X-X
---XXXXX	---X-	---X-	---XXX-	---XXXXX	---XXXXX
-----	-----	-----	-----	-----	-----
-----	-----	-----	-----	-----	-----
		(1E) RS	(1F) US		
		-----	-----		
		---XXXXX	---XXXXX		
		---X---X	---X-X-X		
		---X---X	---X-X-X		
		---X-XXX	---X-XXX		
		---X-X-X	---X---X		
		---X-X-X	---X---X		
		---XXXXX	---XXXXX		
		-----	-----		
		-----	-----		

Fig. 11-7 Control Character Font Matrices

Control characters that produce a visible/audible effect require special explanation. These are described in the following table:

CONTROL CHARACTER	COMMENTS
TAB (09) LF (0A) FF (0C) CR (0D)	Any of these characters will simply appear as a single character on the screen without performing the corresponding visible effect. That is, characters will 'wrap around' the screen as a continuous line
BEL (07)	The bell will sound and the character will appear on the screen

Table 10-1 Control Characters that Produce a Visible/Audible Effect

Control character display can be specified:

- locally as a parameter to a PCOS command for the duration of that command
- globally as a directive that will remain active until cancelled, or until the current working session is terminated

In either case control character display is activated by specifying "+cc", or cancelled by specifying "-cc".

By default control characters are not displayed, but control character display can be made a permanent feature of the operating system if it is active when the PSAVE command is executed.

Examples

IF you enter...	THEN...
<code>flist +cc 1:myfile /CR/</code>	a text listing of "myfile" is displayed including any control characters. On completion, control character display is cancelled
<code>+cc /CR/</code>	control character display is activated globally; that is, until either re-specified, or until the end of the current working session
<code>vq -cc /CR/</code>	<ol style="list-style-type: none">1. If control character display is set globally, it will be cancelled for the duration of the VQUICK command. On completion of the command, control character display will be re-activated2. If control character display is not set globally, "-cc" will have no effect
<code>-cc /CR/</code>	control character display is cancelled until respecified

THE LINE TERMINATION KEYS

For most operations the three line termination keys perform the identical function of placing the ASCII code for carriage return (08) in the keyboard buffer, irrespective as to whether the key is struck on its own or in conjunction with one of the keys /SHIFT/, /CTRL/ or /COMMAND/. Alternative functions may be assigned by means of the CKEY command by assigning different ASCII codes to the raw key codes generated; but note that the PKEY command can only assign a string to the ASCII code placed in the keyboard buffer and therefore cannot assign unique strings to each of the keys (unless unique ASCII codes have first been assigned using the CKEY command).

Regardless of the CKEY command, PCOS contains a facility whereby the three line termination keys can be distinguished among the three from within a BASIC program. In addition to placing the ASCII code in the keyboard buffer, striking one of these keys also places a unique code in a special (LTERM) buffer (0, 1 or 2 for /↵/, /S1/ or /S2/, respectively). This buffer can then be interrogated from BASIC using the LTERM command to distinguish which of the three keys was last pressed. This can be useful in a situation where a BASIC program prompts the user for an entry of some sort. The LTERM command can subsequently be CALLED to return the current value of the LTERM buffer, in order to process the entry in one of three ways depending on which of the three line termination keys was used to terminate the entry.

12. VIDEO FILE EDITOR

ABOUT THIS CHAPTER

This chapter describes how files containing text can be edited using the Video File Editor.

CONTENTS

<u>INTRODUCTION</u>	12-1
THE DISPLAY	12-1
THE KEYBOARD	12-2
<u>HOW TO INVOKE THE VIDEO FILE EDITOR</u>	12-4
EDIT.CMD	12-4
<u>GENERAL EDITING FUNCTION KEYS</u>	12-6
<u>WINDOW MOVING FUNCTION KEYS</u>	12-12
<u>EXITING AND SAVING FUNCTION KEYS</u>	12-14
<u>COMMANDS AND SEARCHING</u>	12-15
STRING SEARCHES	12-15
COMMANDS	12-16

INTRODUCTION

The Video File Editor enables you to create and edit files of text. A text file is a file of records containing printable ASCII characters, and each record is separated from the next either by a carriage-return/line-feed pair or by the record separator character RS.

The Video File Editor displays a 21-line "window" within which you can perform editing functions via the keyboard. A subset of these functions enables you to move the window to access any part of the file.

In addition to the functions mentioned above the Video File Editor can also perform an extensive set of line editing and cursor moving functions and can operate in overstrike, insert text or command mode. The latter enables a subset of high level commands.

Each text line can contain up to 80 characters.

THE DISPLAY

Once the Video File Editor has been invoked the M20 produces a display such as the one shown in Figure 12-1.

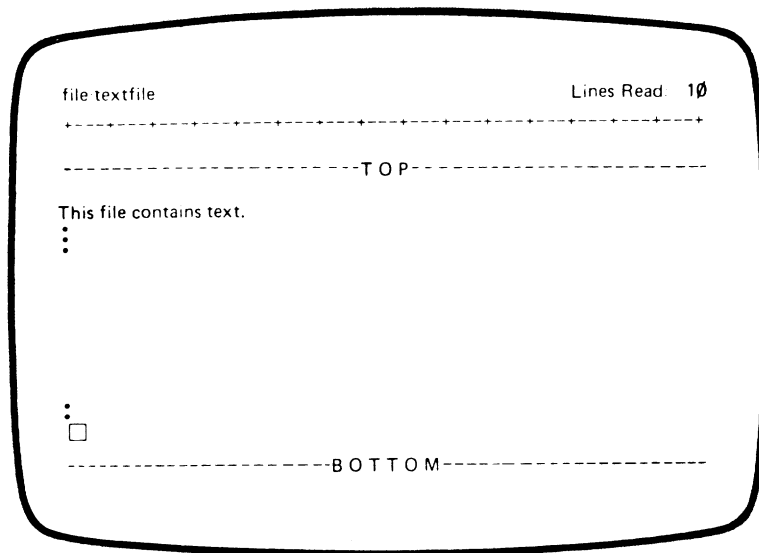


Fig. 12-1 Video File Editor Screen Layout

Line 1 indicates the file name and the current message.

Line 2 is used for high level commands and search strings and is therefore only used when in command mode. Refer to the section entitled "Commands and Searching" for details.

Line 3 shows the tab stop settings (4 character positions per tab).

Lines 4 to 24 contain the text window.

Line 25 is not used.

On entering the Video File Editor the beginning and end of the file are marked by two display lines containing the words TOP and BOTTOM, respectively. The former, known as the TOP bar, always appears immediately before the first line of text in the file. And the BOTTOM bar always appears immediately after the last line of text. They are not actual lines of text and are there merely as markers. The cursor is positioned on the TOP bar.

The cursor changes shape when switching between certain modes of editing. It is represented here as underline.

THE KEYBOARD

The keyboard functions in a different manner once the Video File Editor has been invoked. This provides the means by which the required editing functions are entered. The name of each function key is shown below.

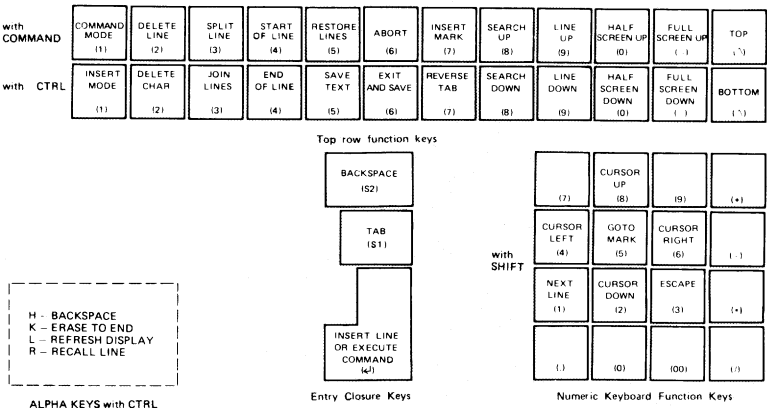


Fig. 12-2 Location of Video File Editor Function Keys

Note: The values in parentheses shown in Figure 12-2 represent the appropriate key on the USA ASCII keyboard. Refer to Appendix B for other keyboard layouts.

The function keys are divided into four areas:

The Numeric Keypad

Pressing the SHIFT key in conjunction with keys on the numeric keypad provides a set of functions primarily for cursor motion, as inscribed on the upper half of the keys themselves. Keys 1, 3 and 5 have other functions and are described later. Note that the numeric keypad can be locked into shift mode by means of the /CTRL/ /!// keys. Subsequently, the numeric values can be entered by means of the /SHIFT/ key. Return to unshifted mode can be made by entering /CTRL/ /!//.

The Top Row

Twelve of the top row of keys are used in conjunction with the /COMMAND/ and /CTRL/ keys to provide 24 functions. These perform or enable most of the major editing operations such as moving the text window, saving text, inserting text and switching between different modes of editing.

Alphabetic Keys

Some of the alphabetic keys when used with the /CTRL/ key provide additional functions.

Entry Closure Keys

These three keys are used for the most frequently used editing functions. They require no shift key.

Programmed Function Keys

You are free to re-arrange function keys at will (while in the PCOS environment) using either the CKEY or PKEY command. For example, if you prefer to have the INSERT MARK function assigned to the /CTRL/ /|/ key (bottom left on the USA ASCII keyboard) enter

pk. &7F,&F3 /CR/

Moreover, frequently entered text strings can be assigned to a single key value by means of the PKEY command. However, YOU MUST TAKE CARE NOT TO DISABLE FUNCTIONS THAT YOU WILL REQUIRE AFTER ENTERING THE VIDEO FILE EDITOR. For example, if you assign some value to the key combination /CTRL/ /6/, then the corresponding edit function (EXIT AND SAVE) will be disabled.

HOW TO INVOKE THE VIDEO FILE EDITOR

The EDIT command is used to enter the Video File Editor.

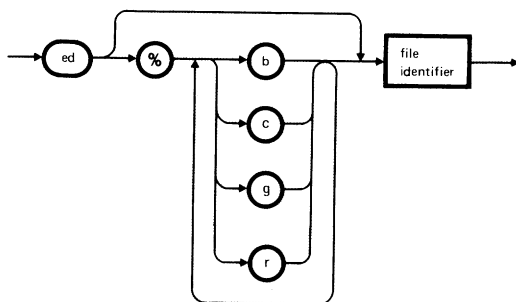


Fig. 12-3 EDIT

Where

SYNTAX ELEMENT	MEANING
b	a backup of the file is to be made when the Video File Editor is entered. This back-up is named filename.bak where filename is the same as that specified in the command line

c	<p>the option which allows you to edit BASIC program files. This is necessary because BASIC files have a different format from standard text files. That is, they contain no tab characters (even if you enter TAB or REVERSE TAB) and have carriage-return/line-feed pairs between lines. Conversely, standard PCOS text files do have tab characters, moreover they delimit lines by means of record separator characters (RS = hexadecimal 1E).</p> <p>Note: if the BASIC program file already exists it must be an ASCII file. Moreover, if a line of such a file contains more than 80 characters, characters beyond the 80th will be lost</p>
g	a special flag that must be included when invoking the Video File Editor from the Greece keyboard
r	the read-only option and is used when you only wish to examine the contents of the file. This protects the file from accidental damage while examining it
file identifier	the name of the file to be edited plus any necessary password or volume identifier

Characteristics

If the file does not already exist the prompt "OK to Create?" appears on the screen to which you must type "y" to create the file.

The Video File Editor is initially in "overstrike" mode. That is, you can enter text and overwrite whatever is already written on the file. The methods of entry into other modes of operation are described later.

The four optional flags (b, c, g and r) can be specified in any order.

Note: For the special project Delta keyboard, %h must be entered in the EDIT command line.

GENERAL EDITING FUNCTION KEYS

The keys whose functions are described below perform general editing functions such as moving the cursor and inserting and deleting text.

CLASS	FUNCTION KEY	MEANING
to move the cursor	/SHIFT/ /8/ (CURSOR UP)	moves the cursor one line up the screen but keeps the same position within the line. If the cursor was on the second line of the window then the window is moved one line up the file and the cursor remains on the second line
	/SHIFT/ /2/ (CURSOR DOWN)	moves the cursor one line down the screen but keeps the same position within the line. If the cursor was on the penultimate line of the window, it stays there and the window is moved down one line
	/SHIFT/ /4/ (CURSOR LEFT)	moves the cursor one character position to the left within the same line
	/SHIFT/ /6/ (CURSOR RIGHT)	moves the cursor one character position to the right within the same line
	/S1/ (TAB)	moves the cursor one tab position (four characters) to the right
	/CTRL/ /7/ (REVERSE TAB)	moves the cursor one tab position (four characters) to the left
	/COMMAND/ /4/ (START OF LINE)	moves the cursor to the start of the current line
	/CTRL/ /4/ (END OF LINE)	moves the cursor to the character position immediately following the last non-space character in the current line

to insert text	/CTRL/ /I/ (INSERT MODE)	is entered from overstrike mode. The cursor changes its shape to show that a new mode has been entered. Any character which is subsequently entered is inserted immediately before the cursor position, and the remainder of the text in the line and the cursor are moved one character position to the right. Any character that was in the last character position in the line is discarded. Striking the INSERT MODE key a second time returns the Video File Editor to overstrike mode and the original cursor is restored
	/↵/ (INSERT LINE)	<p>inserts a blank line immediately after the current line and places the cursor at the beginning of that line. Subsequent text is pushed one line down the screen.</p> <p>If the cursor was already on the bottom line of the screen then the window is moved one line down the file and the blank line is inserted on the last line of the window</p>
to delete text	/S2/ or /CTRL/ /H/ (BACKSPACE)	<p>moves the cursor one character position to the left and deletes the character under the cursor. Subsequent characters in the line do not move. The deleted characters are replaced with spaces.</p> <p>This function is usually used for correcting typing errors when entering new text</p>
	/CTRL/ /2/ (DELETE CHAR)	deletes the character under the cursor and shifts the subsequent characters in the line one position to the left

	/CTRL/ /K/ (ERASE TO END)	deletes the contents of the current line from the current cursor position to the end of the line
	/COMMAND/ /2/ (DELETE LINE)	deletes the current line and moves subsequent text one line up the screen. The position of the cursor is not changed, it remains in the same column position. The deleted line of text is placed in a holding area called the restore buffer. This action overwrites the previous contents of the restore buffer except where DELETE LINE functions immediately follow each other, in which case subsequent deleted lines are appended to the buffer. This enables you to move a block of text from the file into the buffer, from where it can be re-inserted into the same or another file using the RESTORE LINES function
to restore text	/CTRL/ /R/ (RECALL LINE)	restores the contents of the current line to its original state. The contents restored are those that existed before the cursor was moved to this line. Once the cursor is moved off a particular line the old contents of that line cannot be recalled using this function
	/COMMAND/ /5/ (RESTORE LINES)	inserts the contents of the restore buffer into the file starting at the the line below the current cursor position. The cursor is moved to the start of the inserted line(s). The restore buffer itself is not changed. This function is used in conjunction with the DELETE LINE function to move and/or copy blocks of text

to split and join lines of text	/COMMAND/ /3/ (SPLIT LINE)	divides the current line into two by moving all text under and to the right of the cursor onto the next line. The cursor does not move. Text on subsequent lines is shifted one line down the screen
	/CTRL/ /3/ (JOIN LINES)	combines two lines into one. The text on the subsequent line is placed immediately after the last non-space character on the current line. The cursor does not move. If the current line cannot accommodate the entire text of the next line then only that amount which fits is moved and the remaining text stays on the same line but is moved to the left hand edge of the screen
to insert a marker	/COMMAND/ /7/	causes a marker to be inserted in the text immediately following the current line. The marker is a line of reverse video spaces containing the text "MARK". If the MARK line was previously located somewhere else in the text it is moved from where it was to the new position. Note that this is not an actual line of text and will never be written to the file. Its placement is therefore only significant during the current editing session. It is used in conjunction with the GOTO MARK function as a place marker (for details see the section entitled "Window Moving Function Keys"), and in conjunction with the high-level command DELETE (see the section entitled "Commands and Searching")

to enter control characters	/SHIFT/ /3/ (ESCAPE)	<p>inserts the Escape ASCII character (ESC, hexadecimal 1B).</p> <p>The Video File Editor allows you to enter only the printable ASCII character set (hexadecimal codes 20 to 7E). To force the generation of "control" codes (hexadecimal 00 to 1F and 7F) the ESCAPE character must be used. When you type the ESCAPE key a special character (a reverse video pound Sterling symbol) is placed on the screen. This is treated like any any other character except that the following character becomes a control character. This means that only the lower five bits of code are written to the file thereby generating a code in the range 00 to 1F. To generate a code of 7F you must enter /ESCAPE/ /?/</p>
-----------------------------	----------------------	---

Examples

The following table shows some examples of how text can be modified using the functions discussed above.

STEP	IF you enter...	M20 displays...
		The purpose of this text is to act as an example of how to use the editing functions of the Video File Editor
1	DELETE LINE	as an example of how to use the editing functions of the Video File Editor

2	CURSOR UP	as an example of how to use the editing functions of the Video File Editor
3	INSERT LINE INSERT MODE /T/ /h/ /i/ /s/ /SPACE/ /i/ /s/ /SPACE/	This is _ as an example of how to use the editing functions of the Video File Editor
4	JOIN LINES	This is as an example of how to use the editing functions of the Video File Editor
5	DELETE CHAR DELETE CHAR DELETE CHAR	This is an example of how to use the editing functions of the Video File Editor
6	NEXT LINE	This is an example of how to use the editing functions of the Video File Editor
7	DELETE LINE	This is an example of how to use the Video File Editor
8	RESTORE LINES NEXT LINE	This is an example of how to use the Video File Editor the editing functions of
9	INSERT MODE /T/	This is an example of how to use the Video File Editor The editing function of

10	END OF LINE	This is an example of how to use the Video File Editor The editing functions of _
11	BACKSPACE BACKSPACE	This is an example of how to use the Video File Editor The editing functions _
12	RECALL LINE	This is an example of how to use the Video File Editor the editing functions of
13	SPLIT LINE	This is an example of how to use the Video File Editor the editing functions _ of
14	CURSOR UP	This is an example of how to use the Video File Editor _ the editing functions of
15	INSERT LINE	This an example of how to use the Video File Editor the editing functions of

Note: To delete a character in the 80th column you should move the cursor to that position in overstrike mode and enter /SPACE/.

WINDOW MOVING FUNCTION KEYS

The function keys described in the following table enable you to move the window up and down the file.

FUNCTION KEY	MEANING
/COMMAND/ /\/ (TOP)	moves the window to the top of the text file. The cursor is placed on the top bar of the file
/CTRL/ /\/ (BOTTOM)	moves the window to the end of the file. The cursor is placed on the last line of text
/COMMAND/ /-/ (FULL SCREEN UP)	causes the window to be moved up the file by 20 lines. This allows one line of overlap between the old and new displays. The cursor remains on the same screen line
/CTRL/ /-/ (FULL SCREEN DOWN)	causes the window to be moved 20 lines down the file. This allows one line of overlap between the old and new displays. The cursor remains on the same screen line
/COMMAND/ /0/ (HALF SCREEN UP)	causes the window to be moved half a screen (10 lines) up the file. The cursor remains on the same screen line
/CTRL/ /0/ (HALF SCREEN DOWN)	causes the window to be moved half a screen (10 lines) down the file. The cursor remains on the same screen line
/COMMAND/ /9/ (LINE UP)	causes the window to be moved one line up the file. The cursor remains on the same screen line
/CTRL/ /9/ (LINE DOWN)	causes the window to be moved one line down the file. The cursor remains on the same screen line

/SHIFT/ /1/ (NEXT LINE)	moves the window one line down the file and places the cursor at the start of the next text line
/SHIFT/ /5/ (GO TO MARK)	moves the window up or down the file such that the cursor lies on the MARK line. The cursor remains on the same screen line

EXITING AND SAVING FUNCTION KEYS

The function keys described in the following table enable you to exit from the Video File Editor and/or save the file you have been working on.

FUNCTION KEY	MEANING
EXIT AND SAVE	causes the revised text to be written back to the file and the Video File Editor to be terminated. The screen is erased and control is returned to PCOS
SAVE TEXT	causes the revised text to be written to the file. The Video File Editor does not terminate
ABORT	causes the Video File Editor to terminate without writing the revised text to the file. If text has been altered or added since starting the editor you are asked to "Abort?". Strike the ABORT key again to confirm. Any other action causes the Video File Editor to ignore the ABORT. Control is returned to PCOS

COMMANDS AND SEARCHING

The second line of the screen (above the scale line) is called the editor command line and is used for entering high level commands and search strings.

To enter text on the editor command line you must first press the COMMAND MODE function key. This moves the cursor to the second line. You can now enter text there. All line editing operations - such as INSERT MODE, BACKSPACE and DELETE CHAR - now apply to the editor command line. The RECALL LINE function when used in command mode restores the editor command line to its previous contents. The /↵/ key functions instead as EXECUTE COMMAND when used in this mode.

Repeating the COMMAND MODE key returns the cursor to the text window without performing any command operation. The RECALL LINE function, when used in command mode, restores the command line to its previous contents.

STRING SEARCHES

This feature enables you to search the file for a particular combination of characters. Before searching for a text string you must enter command mode by striking the COMMAND MODE function key. Then type in the text to be searched for followed by the appropriate function key, as described in the following table:

FUNCTION KEY	MEANING
/CTRL/ /8/ (SEARCH DOWN)	searches for the text string starting from the the current cursor position and moving down the file until the first occurrence of the string is encountered. If found, the window and cursor are moved to it
/COMMAND/ /8/ (SEARCH UP)	searches for the text string starting from the cursor position and moving up the file. If the string is found then the window and cursor are moved to it

Examples

The following table exemplifies the use of the searching functions.

If you enter on the editor command line	Then strike function key...	M20 displays...
		This is an example of how to use the search function keys of the Video File Editor to find a particular combination of characters
/f/ /u/ /n/ /c/	SEARCH DOWN	This is an example of how to use the search function keys of the Video File Editor to find a particular combination of characters
/e/ /SPACE/ /o/ /f/	SEARCH UP	This is an example of how to use the search function keys of the Video File Editor to find a particular combination of characters

COMMANDS

The Video File Editor commands are a set of special commands that enable you to perform a number of high level functions. Before entering a command you must strike the COMMAND MODE function key. You can then type in the command which is subsequently displayed on the editor command line. To execute the command you must then strike the EXECUTE COMMAND key.

This command enables you to move the window to a specific line number in the file.

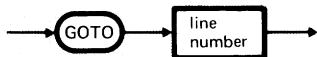


Fig. 12-4 GOTO

Where

SYNTAX ELEMENT	MEANING
line number	a decimal integer and is the desired line number in the file. If this number is greater than the number of lines in the file then the window is moved to the end of the file

Characteristics

Each line of the text file is automatically numbered. That is, the first line of the file is line 1, the TOP bar is line 0 and the MARK bar does not count.

This command removes all text between the current line and the MARK line and places the removed text in the restore buffer from where it can be re-inserted at will. If the MARK line does not exist an error message is given.

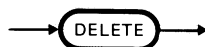


Fig. 12-5 DELETE

The FILE command allows you to suspend processing of the current file and invoke the editor on another file. When editing of the new file is terminated by a SAVE AND EXIT or ABORT function, the old file is recalled at the point at which it was exited.

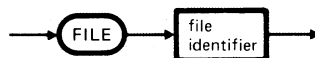


Fig. 12-6 FILE

Where

SYNTAX ELEMENT	MEANING
file identifier	the name of the new file to be edited including any necessary password or volume identifier

Characteristics

The command line option flags (b, c, g or r) used by the old file remain the same for the new file.

Editing of each file is kept entirely independent except for the restore buffer, which enables the transfer of lines of text from one file to another.

Further files can be entered and edited from the new file using the FILE command. There is no limit to the number of levels that can be created in this way except that the text of all the files invoked must fit into user memory.

Files cannot be called recursively.

PART II

13. PCOS COMMANDS

ABOUT THIS CHAPTER

This chapter provides a reference of all PCOS commands. Each command is described in terms of its purpose, a syntax diagram, a description of each syntax element, characteristics of the command, and examples.

CONTENTS

BASIC.CMD	13-1	FKILL.CMD	13-30
BKEYBOARD.BAS	13-2	FLIST.CMD	13-31
BVOLUME.SAV	13-3	FMOVE.CMD	13-33
CI.SAV	13-8	FNEW.CMD	13-35
CKEY.CMD	13-9	FPASS.CMD	13-37
COMMANDS.BAS	13-14	FRENAME.CMD	13-39
DCONFIG.CMD	13-15	FUNPROT.CMD	13-40
EDIT.CMD	13-18	FWPROT.CMD	13-42
EPRINT.SAV	13-18	HELP.BAS	13-44
ERROR.BAS	13-20	IEEE.SAV	13-44
FCOPY.CMD	13-20	LABEL.CMD	13-45
FDEPASS.CMD	13-26	LSCREEN.CMD	13-50
FFREE.CMD	13-28	LTERM (ALWAYS RESIDENT)	13-52

PKEY.CMD	13-53	VQUICK.CMD	13-102
PLOAD (ALWAYS RESIDENT)	13-57	VRENAME.CMD	13-104
PRUN.CMD	13-59	VVERIFY.CMD	13-106
PSAVE.CMD	13-61	WFONT.CMD	13-109
PUNLOAD (ALWAYS RESIDENT)	13-63		
RFONT.CMD	13-65		
RKILL.CMD	13-67		
RS232.SAV	13-69		
SBASIC.CMD	13-69		
SCOMM.CMD	13-72		
SDEVICE.CMD	13-73		
SFORM.CMD	13-75		
SLANG.CMD	13-78		
SPRINT.CMD	13-80		
SSYS.CMD	13-82		
VALPHA.CMD	13-86		
VCOPY.CMD	13-88		
VDEPASS.CMD	13-90		
VFORMAT.CMD	13-91		
VLIST.CMD	13-94		
VMOVE.SAV	13-96		
VNEW.CMD	13-98		
VPASS.CMD	13-100		

Loads the M20 BASIC interpreter into memory and optionally runs a specified BASIC program.

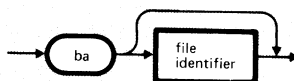


Fig. 13-1 BASIC

Where

SYNTAX ELEMENT	MEANING
file identifier	the file identifier of the BASIC program to be run

Characteristics

If the file is specified, the BASIC Interpreter is loaded, which in turn loads and executes the program stored under that file name. Following program execution the system remains in BASIC command mode.

The no-interaction (%n) flag cannot be used with this command.

Examples

IF you enter...	THEN...
ba my.program /CR/	the BASIC interpreter is loaded into memory, and the file "my.program" is run
ba /CR/	the BASIC Interpreter is loaded into memory and the M20 enters the BASIC environment, command mode

Enables BASIC verbs to be entered at the keyboard by striking the /COMMAND/ key in conjunction with alphabetic keys.



Fig. 13-2 BKEYBOARD

Characteristics

This command is designed for use with the Great Britain and USA ASCII with BASIC keyboards (see Appendix B).

After this command has been executed the BASIC verbs inscribed on the front of the alphabetic keys can be entered by pressing the key in combination with the /COMMAND/ key.

BASIC verbs can be made a permanent feature of your keyboard by using the PSAVE command.

Enables a BASIC program to use the "Search" and "DiskFree" system calls as well as to obtain the name of the current volume.

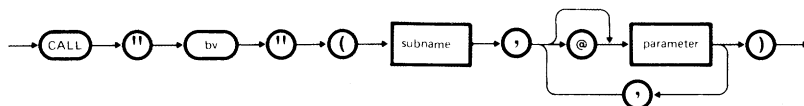


Fig. 13-3 BVOLUME

Where

SYNTAX ELEMENT	MEANING
subname	<p>the name of the function to be used. This value may be one of</p> <ul style="list-style-type: none"> - search to search the volume directory for a specified file name string - diskfree to return the number of free blocks remaining on the specified volume - getvolname to return the name of the current volume
parameter	<p>a parameter to be passed to the command. This depends on the function. See below for a description of the corresponding function</p>

Characteristics

This command can only be CALLED from BASIC. It cannot be executed directly from the PCOS environment.

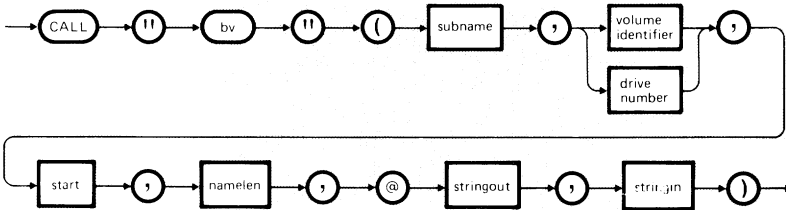


Fig. 13-4 BVOLUME search

Where

SYNTAX ELEMENT	MEANING
subname	the function name. In this case 'search'
drive number	the drive number in which to search, or -1 if the last selected drive is desired
volume identifier	the volume in which to search
start	a flag that determines where the search is to start from. It has two possible values:

	<p>1 - start searching from the start of the directory</p> <p>0 - continue searching from the position established by the last BVOLUME call</p>
namelen	the length of the file name string to be searched for. This value can be set to zero if all names are to be returned
stringout	a dummy string variable in which the file name searched for will be returned. It must be initialised prior to calling BVOLUME by setting it to at least 14 characters
stringin	the name of the file to be searched for. A group of file names may be specified using wild cards

Example

DISPLAY	COMMENTS
<pre> 10 SUBNAME\$="search" 20 DRIVENUM%=0 30 NAMELEN%=5 40 STRINGIN\$="*.cmd" 50 STRINGOUT\$="12345678901234" 60 START%=1T\$="?Err.cmd " 70 CALL "BV"(SUBNAME\$, DRIVENUM%,START%,NAMELEN%, @STRINGOUT\$,STRINGIN\$) 80 PRINT"returned file name is: ";STRINGOUT\$ 90 START%=0 100 GOTO 70 110 END </pre>	<p>Lines 10 to 60 set the parameters for a BVOLUME 'search' CALL on the volume inserted in drive 0. Line 30 sets NAMELEN% to the length of the string to be searched, while line 60 specifies the string; in this case it contains a wild card character. Line 50 sets the length of the output string to 14 characters.</p> <p>Line 80 CALLs the BVOLUME command and line 90 displays the returned file name. Line 90 sets the START% parameter to search for the next occurrence of a file whose name ends with ".cmd"</p>

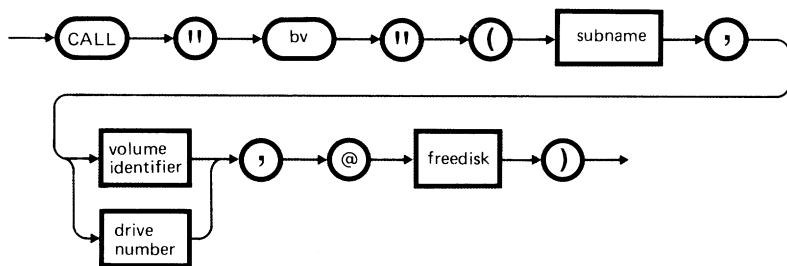


Fig. 13-5 BVOLUME diskfree

Where

SYNTAX ELEMENT	MEANING
subname	the function name. In this case 'diskfree'
drive number	the drive number containing the volume whose number of free blocks is required. If a value of -1 is entered then the last drive accessed is assumed
volume identifier	the volume whose number of free blocks is to be returned
freedisk	an integer variable in which the number of free blocks will be returned

Example

DISPLAY	COMMENTS
<pre> 10 DEFINT F 20 SUBNAME\$="diskfree" 30 V_ID\$="myvol:" 40 FREEDISK=0 50 CALL "bvolume" (SUBNAME\$,V_ID\$, @FREEDISK) 60 PRINT "free blocks ..."; FREEDISK 70 END </pre>	<p>Lines 20 to 40 set the parameters for a BVOLUME "diskfree" CALL on the volume inserted in drive 0.</p> <p>Line 50 CALLs the BVOLUME command.</p> <p>Line 60 displays the number of free blocks</p>

Remark

BASIC always uses signed arithmetic with integer variables. Therefore, if the "diskfree" option of BVOLUME is used on the hard disk, the returned value must be converted to double-precision before being displayed or used in calculations.



Fig. 13-6 BVOLUME getvolname

Where

SYNTAX ELEMENT	MEANING
subname	the function name. In this case 'getvolname'
volume name	a dummy string variable in which the name of the last volume accessed will be returned

Example

DISPLAY	COMMENTS
10 SUBNAME\$="getvolname" 20 VOLNAME\$="12345678901234" 30 CALL "BV"(SUBNAME\$, @VOLNAME\$) 40 PRINT "the current volume is named"; VOLNAME\$ 50 END	Line 10 specifies 'getvolname' as the subname parameter. Line 20 defines the string variable into which the volume name will be returned. Line 30 CALLs the BVOLUME command to perform the required function. Line 40 displays the returned volume name

Allows a BASIC program to interface with the RS-232-C driver. It can only be used from BASIC.

For further details refer to "I/O with External Peripherals User Guide".

Enables the user to change the character codes assigned to the raw key codes generated at the keyboard, or to set the shift lock for the alphanumeric and/or numeric keypads.

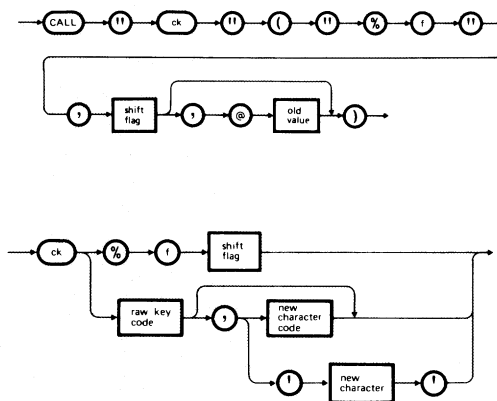


Fig. 13-7 CKEY

Where

SYNTAX ELEMENT	MEANING
f	when present indicates that the numeric value that follows specifies the shift flag value

shift flag	<p>an integer in the range 0 to 3 that determines the setting of the two shift flags - shift lock and cursor lock, where:</p> <p>shift lock - when set 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 at the same time as holding down the SHIFT key will generate a lower case character</p> <p>cursor lock - when set, 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. Likewise, if such a key is pressed in conjunction with the /SHIFT/ key, it will generate the code normally generated by striking the key on its own.</p> <p>The possible values are:</p> <ul style="list-style-type: none"> 0 - both flags cleared 1 - shift lock set, cursor lock cleared 2 - cursor lock set, shift lock cleared 3 - both flags set
old value	<p>a previously defined BASIC integer variable to which the command assigns a value which indicates the previous setting of the shift flag. This facility can only be used when the command is invoked from the BASIC environment using the BASIC CALL statement</p>
raw key code	<p>the code that is immediately generated by striking a particular key of the keyboard. This code is dependent only on the physical position of the key. That is, it is independent of the national keyboard and of any string assignments made by the PKEY command</p>

new character code	the code to be generated when the key or key combination specified by the raw key code is struck. The code may be specified as a decimal integer on its own, or as a hexadecimal integer preceded by &
new character	any character that can be entered from the keyboard

Characteristics

Figure 13-7 shows the raw key codes that are generated for every key whether struck on its own, in conjunction with the SHIFT key, with the CTRL key, or with the COMMAND key. The keys shown correspond to the physical position of the keys on the keyboard.

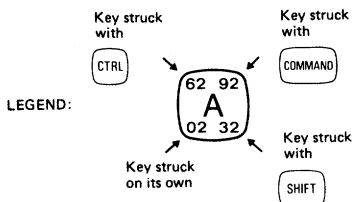
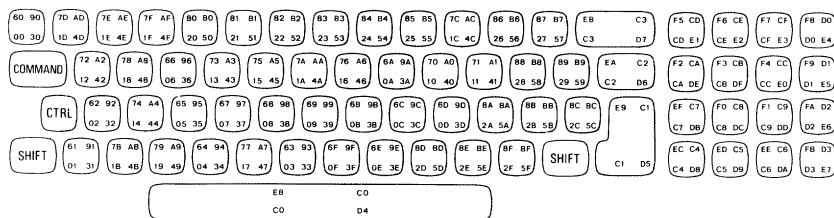


Fig. 13-8 Raw Key Codes

Each raw key code points directly into a table which translates the raw key code into the ASCII code for the keyboard character, or a function for the particular national keyboard. The translated raw key codes correspond to the standard ASCII codes (hexadecimal 0 to 80), while the remainder specify other keyboard functions. But note the special cases shown in the following table:

CODE (hexadecimal)	CORRESPONDING KEY COMBINATION	DESCRIPTION
A0	/CTRL/ /RESET/	Logical reset
A1	-	(reserved)
A2	/CTRL/ /C/	Break facility
A3	/CTRL/ /S/	Halt display
A4	/CTRL/ ///	Cursor lock
A5	/COMMAND/ ///	Shift lock
A6	/00/	Two zeros
A7	/↵/	End of line (CR in keyboard buffer, zero in LTERM buffer)
A8	/S1/	End of line (CR in keyboard buffer, '1' in LTERM buffer)
A9	/S2/	End of line (CR in keyboard buffer, '2' in LTERM buffer)
AA	/↵/ (DATEV keyboard)	End of line (CR in keyboard buffer, zero in LTERM buffer)
AB	/S1/ (DATEV keyboard)	End of line (CR in keyboard buffer, '1' in LTERM buffer)
AC	/S2/ (DATEV keyboard)	End of line (CR in keyboard buffer, '2' in LTERM buffer)
AD	-	(reserved)
AE	-	(reserved)
AF	-	No operation

The raw key codes are shown in hexadecimal in Figure 13-7. They may, however, be entered either as hexadecimal values preceded by an & (ampersand), or on their own as their decimal equivalents.

The user may enter the new character code either as a decimal integer, or as a hexadecimal integer preceded by an &, or the ASCII value can be entered simply by striking the corresponding key.

If, when entering a CKEY command, the raw key code is specified without also specifying a character code, then the current code corresponding to the raw key code is displayed.

The character codes assigned to the raw key codes remain until either changed by another CKEY command, or until the entire conversion table is replaced by means of the SLANG command, or the current working session is terminated. The changed values can alternatively be made permanent by use of the PSAVE command.

The no-interaction (%n) flag cannot be used with this command.

ANY CODE ASSIGNED TO A KEY USING THE CKEY COMMAND WILL BE EFFECTIVE EVEN AFTER ENTERING ANOTHER ENVIRONMENT OR APPLICATION PROGRAM. THIS ENABLES THE USER TO, FOR INSTANCE, RE-ARRANGE FUNCTION KEYS AT WILL, BUT THE USER MUST TAKE CARE NOT TO DISABLE FUNCTIONS THAT WILL BE REQUIRED ON ENTERING THE NEW ENVIRONMENT/APPLICATION PROGRAM.

Examples

IF you enter...	THEN...
ck &C3,8 /CR/	the key S2 becomes backspace. That is, the decimal code for backspace (8) is assigned to the raw key code generated when S2 is struck (hexadecimal C3)

<pre>ck &C3 /CR/</pre>	<p>the raw key code and the corresponding ASCII code are displayed as follows:</p> <pre>KEY = 195 (raw key code) CODE = 8 (ASCII code)</pre> <p>(Note that values are displayed in decimal)</p>
<pre>ck &64,&AF /CR/ . .</pre>	<p>a no operation code (hexadecimal AF) is assigned to the raw key code generated when /CTRL/ /C/ is struck (hexadecimal 64), thereby disabling the break facility.</p>
<pre>ck &64, &A2 /CR/</pre>	<p>the break facility is re-enabled</p>
<pre>ck &72,'w' /CR/</pre>	<p>the ASCII code for "w" is assigned to the raw key code 72 (hexadecimal). Entering /CTRL/ /Q/ will subsequently be the same as "w"</p>
<pre>ba /CR/ OLDVALUE%=0 CALL "ck" ("%f", SHIFTFLAG%,@OLDVALUE%) /CR/</pre>	<p>the M20 enters the BASIC environment from where the CKEY command is called, which in turn reads the shift value from integer variable SHIFTFLAG% and returns the old shift flag value in integer variable OLDVALUE%</p>

Displays a list of all PCOS commands.

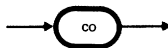


Fig. 13-9 COMMANDS

Characteristics

The PCOS commands are listed, together with a brief description of their use and purpose. Further information about any particular PCOS command can be selected from a menu provided.

Remark

The COMMANDS utility resides on one of the Help diskettes.

Displays the hardware and/or memory configuration.

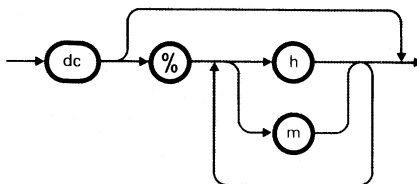


Fig. 13-10 DCONFIG

Where

SYNTAX ELEMENT	MEANING
h	the hardware configuration is to be displayed

m	the memory configuration is to be displayed
---	---

Characteristics

A typical hardware configuration would be displayed as follows :

Memory Configuration:	7.
Floppy Disk Drive(s):	320 KByte.
Drive 0 Diskette:	320 KByte.
Drive 1 Diskette:	320 KByte.
Hard Disk Drive:	Not Present.
IEEE Board:	Not Present.
RS-232 Board:	Not Present.
8086 CPU Board:	Not Present.
Display Type:	Black and White.

Note that on a 320 Kbyte drive, a 320 Kbyte diskette will be indicated even if there is non present. Furthermore, DCONFIG will not recognise a diskette change until the new diskette has been accessed.

The "Memory Configuration" value can be one of the following:

- 0 - eight-colour system with one or more 128K memory expansion boards
- 1 - four-colour system with one or more 128K memory expansion boards
- 2 - eight-colour system with one or more 32K memory expansion boards
- 3 - four-colour system with one or more 32K memory expansion boards
- 5 - black and white system with no memory expansion
- 6 - black and white system with one or more 128K memory expansion boards
- 7 - black and white system with one or more 32K memory expansion boards

Typically, the memory configuration (for a system containing two 32K memory expansion boards) is displayed as follows :

Memory Configuration:

```

Total memory size:    192 KBytes
Free memory size:    141936 Bytes
Basic memory size:   55000 Bytes

```

Address (Hex)	Block Size	Filename
Segment Offset	Hex Decimal	(Owner)
00 0004	3FFC 16380	PCOS.SAV

Address (Hex)	Block Size	Filename
Segment Offset	Hex Decimal	(Owner)
06 0004	7FFC 32764	FREE BLOCK
06 8004	3FFC 16380	PCOS.SAV

Address (Hex)	Block Size	Filename
Segment Offset	Hex Decimal	(Owner)
02 0084	0178 376	PCOS.SAV
02 0200	1480 5348	FREE BLOCK
02 1684	109A 4250	PCOS.SAV
02 2722	D6D8 55000	FREE BLOCK

Address (Hex)	Block Size	Filename
Segment Offset	Hex Decimal	(Owner)
0A 8004	7620 30240	FREE BLOCK
0A F628	0788 1928	dconfig.cmd
0A FDB4	004C 76	FREE BLOCK
0A FE04	0028 40	dconfig.cmd
0A FE30	012C 300	FREE BLOCK
0A FF60	004E 78	PCOS.SAV
0A FFB2	004E 78	PCOS.SAV

Address (Hex)	Block Size	Filename
Segment Offset	Hex Decimal	(Owner)
09 8004	3FFC 16380	FREE BLOCK

If both the %h and %m flags are specified then both the hardware and memory configurations are displayed.

If no flag is specified then a display such as the following is shown:

```
L1.M20 System Configuration.
Total memory size: 192 KBytes.
Free memory size: 141936 Bytes.
Basic memory size: 55000 Bytes.
Display Type: Black and White.
Disk Drive(s): 2 ready.
```

The no-interaction (%n) flag cannot be used with this command.

Example

IF you enter..	THEN...
dc %hm /CR/	the hardware and memory configurations are displayed

This command invokes the Video File Editor for editing a specified file.

Within the Video File Editor it is possible to perform a range of text editing functions. These are described in the Chapter 12.

Lists any error, or errors specified by their number with a brief, one line, description of each.

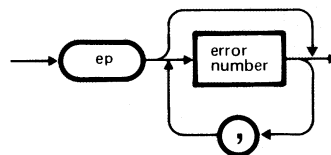


Fig. 13-11 EPRINT

Where

SYNTAX ELEMENT	MEANING
error number	any existing PCOS error number. If no error number is specified then no error is described, but the EPRINT command is loaded into memory

Characteristics

If an unprintable error or a BASIC error number is entered, no description is given.

Once the EPRINT command is resident in memory (via execution, PSAVE or PLOAD), then any errors returned from PCOS will not only be displayed with the error number, but will also have the associated descriptive label with it. Note that the EPRINT command has a SAV extension and therefore becomes resident on execution.

Example

IF you enter...	THEN...
ep 58,59 /CR/	the M20 will display the two errors as shown below: ERROR 58... file already exists ERROR 59... disk type mismatch

Displays a series of display frames that enable a user to display error messages in functional groups.

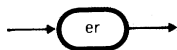


Fig. 13-12 ERROR

Remark

This command resides on one of the Help diskettes. The same information can be obtained via the HELP command.

- (1) - Copies the contents of one file into another.
- (2) - Copies one or more files (specified using wild cards) from one volume onto another

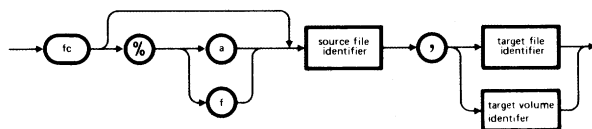


Fig. 13-13 FCOPY - (1)

Where

SYNTAX ELEMENT	MEANING
a	<p>the append flag.</p> <p>If the target file already exists and %a is specified, then the confirmation prompt will be suppressed and the contents of the source file will be appended to the end of the target file. If the target file does not exist, %a is ignored</p>
f	<p>the force copy flag.</p> <p>If the target file already exists and %f is specified, then the confirmation prompt will be suppressed and the contents of the target file will be overwritten with the contents of the source file. If the target file does not exist, %f is ignored</p>
source file identifier	the file name, complete with any necessary password and volume identifier of the file to be copied
target file identifier	the name of a file on an unprotected volume, complete with any necessary password and volume identifier. If the file does not exist, it will be created and will have the same password as the source file
target volume identifier	<p>the volume name or the drive number in which the target volume resides.</p> <p>If the volume is not enabled, then the password has to be specified</p>

Characteristics

If the target file exists and no flag is specified, FCOPY will prompt the user for confirmation before overwriting it.

File already exists. Do you wish to overwrite? (y/n)

If %n is specified in the command line then overwrite is automatically assumed.

If the target volume is specified the newly created file will have the same name and password as the source file.

Examples

IF you enter...	THEN...
<code>fc %a dk1/dkpw1:FILE1, dk2:FILE2 /CR/</code>	the contents of the file called "FILE1", on diskette "dk1", with password "dkpw1" are appended to the file called "FILE2", on diskette "dk2"
<code>fc dk1:myfile, dk1:yourfile /CR/</code>	a file may still be copied within the same volume. This example shows "myfile" is to be copied into "yourfile", both of which are on "dk1"
<code>fc mydisk:*,10:</code>	the entire contents of "mydisk" are copied onto the hard disk

Remarks

If there is sufficient space, a file can be copied or appended to another file on the same volume.

If the target file does not exist it will be created and given the the same password (if any) as the source file.

If the target file already exists, it will maintain its password.

The target file must not be write-protected.

The copying process tidies-up the target file by gathering all the scattered data into one contiguous file - assuming there is enough space to do this. This saves on I/O operation time and is therefore worth doing on files that have become extensively fragmented.

At the end of an attempted FCOPY operation, error message 61 will result if there is insufficient space ("disk filled").

A file cannot be appended to itself.

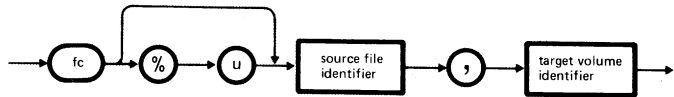


Fig. 13-14 FCOPY (2)

Where

SYNTAX ELEMENT	MEANING
u	<p>the unprotected copy flag.</p> <p>If "%u" is specified, then the copy process will skip any copy-protected files specified by the source file identifier parameter (see below). If not specified, then any copy-protected files will still be allocated file space on the target volume even though the contents of the files are not copied</p>
source file identifier	<p>the volume identifier of the volume containing the files to be copied followed by a string of characters, including at least one wild card, that specifies one or more file names complete with any password, if it is common to all files specified</p> <p>Note: If the volume identifier is not specified, the M20 will search for the files specified on the volume inserted in the last drive accessed</p>
target volume identifier	<p>the volume name or the drive number in which the target volume is inserted.</p> <p>If the volume is not enabled, then the password has to be specified</p>

Characteristics

The newly created files will have the same names and password as the source files.

If a file specified within the group already exists on the target file, then the prompt

File already exists. Do you wish to overwrite? (y or n)

is displayed. If %n is specified in the command line, then overwrite is assumed.

Example

IF you enter...	THEN...
<code>fc %u 0:*.cmd,1: /CR/</code>	all the files which have the extension CMD resident on the volume inserted in drive 0 are copied onto the volume inserted in drive 1. Any copy-protected files are skipped

Remarks

Any target files that do not exist will be created. Each newly created file will have the same password as the corresponding source file (if the source file has one). On the other hand, if the target file already exists, it will maintain its password.

A target file must not be write-protected.

The copying process tidies up the target file by gathering all the scattered data into one contiguous file, if there is enough space. This saves on I/O operation time and is therefore worth doing on files that have become extensively fragmented.

At the end of an attempted FCOPY operation, error message 61 will result if there is insufficient space ("disk filled").

Deletes a previously assigned file password on an unprotected and enabled volume.



Fig. 13-15 FDEPASS

Where

SYNTAX ELEMENT	MEANING
volume identifier	the volume name and/or drive number, plus any necessary volume password
file name	EITHER the name of an existing file OR a string of characters, including wild cards, that specifies a group of file names
old file password	the existing file password which is to be deleted

Examples

IF you enter...	THEN...
fd 0:*/secret /CR/	the user can optionally remove the password "secret" from any file on the diskette inserted in drive 0 by answering "y" (yes) or "n" (no) to the messages displayed
fd 0: myfile/newpass /CR/	the password "newpass" to the file called "myfile" on the diskette inserted in drive 0 is deleted
fd 1:FILX/PASSX /CR/	the password called "PASSX" on the file called "FILX" on the diskette inserted in drive 1 is deleted

Remarks

Since the password must be known before it can be deleted, this command cannot be used to gain access to a file for which the password has been forgotten.

Wild card characters cannot be used in the password portion of a file identifier. Thus, even though more than one file can be specified, you can only specify one password, which in turn can be the same for more than one file.

Frees any unused file sectors from a specified file or a group of files or an entire volume. The freed sectors are made available to the system for future use.

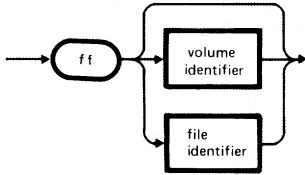


Fig. 13-16 FFREE

Where

SYNTAX ELEMENT	MEANING
volume identifier	the volume name or the drive number in which the volume resides, plus any necessary volume password
file identifier	<p>the name of the file plus any necessary password and volume identifier.</p> <p>The file name can be specified using wild card characters to specify a group of files.</p> <p>If the file identifier is left out, then FFREE will be executed on all the files resident on selected volume</p>

Characteristics

After entering the command the following prompt appears:

You may not change disks while FFREE in progress. Continue?

Replying "y /CR/" the operation continues. "n /CR/" aborts the operation.

If FFREE is used on the hard disk, any bad blocks that are allocated will not be freed.

Examples

IF you enter...	THEN...
ff 1:myfile /CR/	after replying "y" to the subsequent prompt, the file "myfile" resident on the diskette inserted in drive 1 is cleaned up of any unused sectors
ff myvol: /CR/	after replying "y" to the subsequent prompt, all the files on the volume "myvol" are cleaned-up of any unused sectors. That is, the entire volume is cleaned-up

Remarks

FFREE will not be executed if either the volume or the file is write-protected.

Deletes a specified unprotected file or group of files from an unprotected or enabled volume

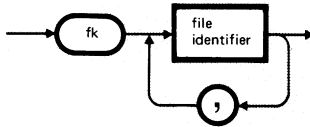


Fig. 13-17 FKILL

Where

SYNTAX ELEMENT	MEANING
file identifier	<p>the name of the file to be deleted, plus any necessary password and volume identifier.</p> <p>The file name can be expressed using wild cards to specify a group of files</p>

Characteristics

If the FKILL command is used on the hard disk, any bad blocks currently allocated to the specified file(s) will not be freed.

Examples

If you enter...	THEN...
fk myvol: myfile/mypass /CR/	the file "myfile" with password "mypass" resident on the volume "myvol" is deleted
fk 1:myfile, yourfile /CR/	files "myfile" and "yourfile" are deleted from the diskette inserted in drive 1
fk mydisk:* /CR/	you can optionally delete any file resident on "mydisk" by answering "y" (yes) or "n" (no) to the interactive messages displayed

Remarks

FKILL will not function if either the volume or the file is write-protected.

Lists the contents of one or more files. The listing can be in either ASCII or hexadecimal form.

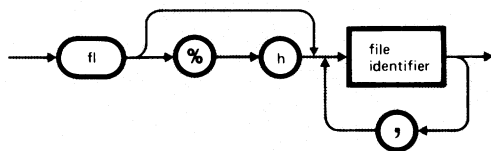


Fig. 13-18 FLIST

Where

SYNTAX ELEMENT	MEANING
file identifier	the name of the file, plus any necessary password and volume identifier. The file name can be specified using wild card characters to specify a group of files
h	the specified files are to be listed in hexadecimal form

Example

IF you enter...	THEN...
fl dk2/diskpwd: myfile/flpwd /CR/	the file "myfile" which has the password "flpwd", and resides on volume "dk2", which in turn has the password "diskpwd", is listed
fl 0:F*/pass,1:G.1 /CR/	all files beginning with the letter F and residing on the diskette inserted in drive 0 will be listed provided they all have the password "pass" File G.1 residing on the diskette inserted in drive 1 will also be listed
fl %h 1:yourfile /CR/	the file "yourfile" resident on the diskette inserted in drive 1 is listed in hexadecimal form

Copies a file from one diskette to another using only one drive.

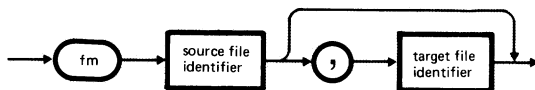


Fig. 13-19 FMOVE

Where

SYNTAX ELEMENT	MEANING
source file identifier	the name of the file to be copied plus any necessary password
target file identifier	<p>the name of the file plus any necessary password.</p> <p>If the file does not exist it will be created.</p> <p>If the target file identifier is not specified then the source file is copied onto the target diskette into a newly created file of the same name as the source file</p>

Characteristics

When FMOVE is called the M20 displays a message asking the user to insert the source diskette in the drive. On hitting any key, the first part of the file is read into memory and consequently a message specifying the number of passes necessary to make the copy is displayed. At

this point the user is asked to insert the target diskette in the drive. On hitting any key the data that was read into memory will be transferred to the target diskette. If the copy operation takes n passes, the M20 will display messages asking the user to repeat the process described above n times. It is only after the last pass is completed that the PCOS prompt appears meaning that other commands can be entered.

FMOVE is only useful on M20 systems that have only one diskette drive. On dual diskette drive systems FCOPY would normally be used.

If the destination file already exists then the message

File already exists: overwrite, append or quit (o/a/q)

will be displayed.

If you enter "o", then the contents of the target file will be overwritten with the contents of the source file.

If you enter "a", then the contents of the source file will be appended to the target file.

If you enter "q", then the command is aborted, and the PCOS prompt appears on the screen.

Example

IF you enter...	THEN...
fm myfile,myfile /CR/	the FMOVE process is started, and if a diskette which has a file called "myfile" resident on it is inserted, then this file is read and stored in memory. If another diskette is now inserted, then the file "myfile" will be written on it

Remark

FMOVE can do all that FCOPY can do except:

- FMOVE does not allow wild cards whereas FCOPY does

- FCOPY deallocates blocks that are not used by the target file, while FMOVE does not. The number of blocks allocated to the target file after an FMOVE operation is always the same as that of the source file.

Creates an empty file on an unprotected or enabled volume, assigns a name to the file and optionally a password. The file system allocates only one extent (a series of contiguous blocks) to the file.

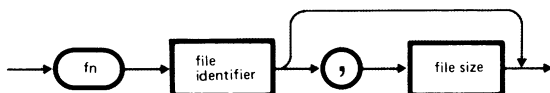


Fig. 13-20 FNEW

Where

SYNTAX ELEMENT	MEANING
file identifier	<p>the file name, optionally with a password and a volume identifier.</p> <p>If specified, the password is assigned to the file</p>

file size	<p>the number of blocks to be made available to the file. The minimum value is 2 and the maximum value depends on the number of contiguous free blocks available on the volume.</p> <p>There are two exceptions to the above:</p> <p>(i) if this parameter is specified as 0 (or not specified at all) then the file system allocates the number of blocks specified in the "extent size" parameter (see the SSYS command), which by default is 8</p> <p>(ii) if specified as 1 then the file system allocates two blocks to the file</p>
-----------	---

Example

IF you enter...	THEN...
fn newfile/pass,12 /CR/	a 12 block file called "newfile" is created on the volume in the last drive to be selected and assigned the password "pass"

Remarks

If the volume has write-protection, the aluminised label must be removed before this utility can be used.

If there is not enough contiguous volume space to honour the request, a message

ERROR 61 ---- disk filled

will be displayed.

Assigns a password to a file, or a number of files, on an unprotected volume, or changes an already existing file password to a new one.

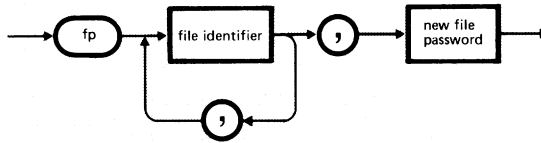


Fig. 13-21 FPASS

Where

SYNTAX ELEMENT	MEANING
file identifier	EITHER the file name, plus any necessary password and volume identifier OR a volume identifier (if necessary) followed by a string of characters, including wild cards, that specifies one or more file names plus any necessary password, if it already exists and is common to all the files specified
new file password	is an alphanumeric string to be assigned to the file(s) as a password

Examples

IF you enter...	THEN...
<pre>flist 0:myfile/pass /CR/ . . .</pre>	file "myfile" is listed
<pre>fpass myfile/pass, newpass /CR/ . . .</pre>	the password "pass" of the file called "myfile" is changed to newpass
<pre>flist myfile /CR/ . . .</pre>	<p>an error message</p> <p>ERROR 73 --- invalid password</p> <p>is displayed</p>
<pre>flist myfile/newpass /CR/</pre>	file "myfile" is again listed
<pre>fp 1:myfile,yourfile, ours /CR/</pre>	the password "ours" is assigned to both "myfile" and "yourfile"
<pre>fp 0:*,secret /CR/</pre>	all the file names on the diskette inserted in drive 0 are displayed one by one and an interactive message asks whether the password "secret" is to be set on the file. Entering "y" (yes) or "n" (no) for each file assigns or does not assign the password, respectively

Remarks

If the file is on a write-protected diskette, the write-protect label must be removed before issuing the command.

Like VPASS, the command FPASS can assign a new password directly to a file which already has one, but the user must specify the old password.

Once assigned, a password must be specified in all future references until it is deleted or changed. Otherwise, the file system will display error code 73 corresponding to the message "invalid password".

File (and volume) security is a user responsibility. It is important to remember passwords as neither PCOS nor BASIC will allow the user access to a file whose password has been forgotten.

To safeguard files from those who have access to them, use volume and file write-protection.

Changes the name of an existing file on an enabled or unprotected volume.

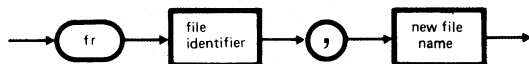


Fig. 13-22 FRENAME

Where

SYNTAX ELEMENT	MEANING
file identifier	the file name, plus any necessary password and volume identifier
new file name	any file name not currently used on the volume where the file being renamed resides

Example

IF you enter...	THEN...
fr 1:OLDname, NEWname /CR/	the file name "OLDname" of the file resident on the diskette inserted in drive 1 is changed to "NEWname"

Remark

The FRENAME command has no effect on the file password.

Removes write-protection from a file, or a number of files, on an unprotected or enabled volume.

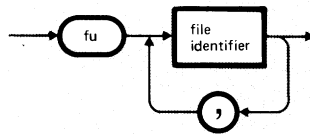


Fig. 13-23 FUNPROT

Where

SYNTAX ELEMENT	MEANING
file identifier	<p>EITHER</p> <p>the name of the file, including any necessary extension, password and volume identifier</p> <p>OR</p> <p>a volume identifier (if necessary) followed by a string of characters, including wild cards, that specifies one or more file names complete with any necessary password, if one exists and is common to all the files specified</p>

Examples

IF you enter...	THEN...
fu mydisk:myfile /CR/	the write-protection is removed from "myfile" which resides on the volume "mydisk"
fu DK1:* /CR/	you can optionally remove write-protection from any file resident on the volume "DK1", by answering "y" (yes) or "n" (no) to the interactive messages displayed

Remarks

If the file is on a write-protected diskette, the write-protect label must be removed before, and replaced after this operation.

Once write-protection is removed, the file can be deleted, overwritten, appended to, renamed, or in any other way modified (providing there is no write-protect label on the diskette).

If the file has no write-protection when this command is issued, this command will change nothing.

Write-protects a file, or a number of files, on an unprotected or enabled volume.

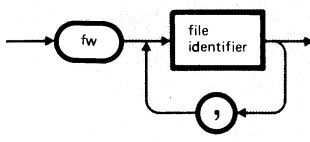


Fig. 13-24 FWPROT

Where

SYNTAX ELEMENT	MEANING
file identifier	<div>EITHER</div> <div>the name of the file, plus any necessary password and volume identifier</div> <div>OR</div> <div>a volume identifier (if necessary) followed by a string of characters, including wild cards, that specifies one or more file names complete with any necessary password, if one exists and is common to all the files specified</div>

Examples

IF you enter...	THEN...
fw myfile/disks: myfile /CR/	the file "myfile" residing on volume "mydisk", which has a password of "disks", is write-protected
fw 1:fileA, fileB /CR/	files "fileA" and "fileB" residing on the diskette inserted in drive 1 are both write-protected
fw 1:prog.??? /CR/	you can optionally apply write-protection to any file that has the file name "prog", with a three letter extension and which is resident on the diskette inserted in drive 1, by answering "y" (yes) or "n" (no) to the messages displayed

Remarks

If the file is on a write-protected diskette, the write-protect label must be removed before this operation and replaced afterwards.

Once write-protected, the file may not be modified until write-protection is removed with a FUNPROT command.

If the file already has write-protection assigned, this command will change nothing.

A write-protected file can be listed or copied.

Write-protection can be assigned to any file, whether it has a password or not.

If all the files of a particular diskette are only meant to be read, then the diskette should be write-protected.

Calls a series of display frames to guide the user about how to use PCOS and BASIC on the M20 system.

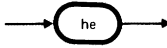


Fig. 13-25 HELP

Characteristics

The HELP command is an interactive utility that displays information about commands or errors.

Remarks

The information provided by the command and error sections is the same as that provided by the COMMANDS and ERROR commands.

Note: This utility is not supplied on the system diskette.

Loads and initialises the IEEE-488 extension package.

For more details about the IEEE-488 package see the "I/O with External Peripherals User Guide".

Displays a string with a specified colour, magnification and orientation at a given position, either within the entire screen or current window.

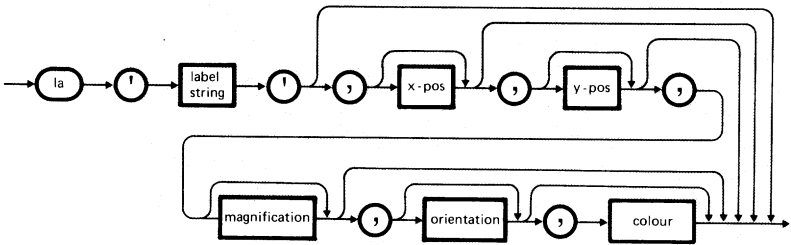


Fig. 13-26 LABEL

Where

SYNTAX ELEMENT	MEANING	RANGE	
		MIN	MAX
label string	<p>a valid string of printable characters that will be displayed at the specified position, magnification and orientation.</p> <p>The string must be entered with leading/trailing quotation marks</p>		

x - pos	<p>a legal value in pixels, used by the LABEL command to determine the distance of the left side of the first string character from the left hand edge of the screen or current window.</p> <p>The user can quote either:</p> <ul style="list-style-type: none"> - the pixel value, if you are in PCOS or BASIC and the hardware co-ordinate system has been assumed, or - the corresponding SCALEX function if you are in BASIC and a user co-ordinate system has been assumed (see "Remarks") 	0	window width, less the character width, less 1 (all measurements in pixels)
y - pos	<p>a legal value in pixels, used by the LABEL command to determine the distance of the string base from the bottom of the screen or current window.</p> <p>The user can quote either:</p> <ul style="list-style-type: none"> - the pixel value if you are in PCOS or BASIC and the hardware co-ordinate system has been assumed, or 	0	window height, less the character height, less 1 (all measurements in pixels)

	- the corresponding SCALEY function if you are in BASIC and a user co-ordinate system has been assumed (see "Remarks")		
magnification	a whole number n, where n is n times the normal character size	1	16
orientation	<p>a value indicating the straight line aspect of the label string</p> <p>0 - parallel to x-axis output left to right</p> <p>1 - parallel to y-axis output bottom to top (+ 90 degrees)</p> <p>2 - parallel to y-axis output top to bottom (- 90 degrees)</p>	0	2
colour	the colour number in the range 0 to 7 for eight colour displays, 0 to 3 for four colour displays, or 0 or 1 for black and white displays. If omitted the foreground colour is assumed	0	7

Defaults

SYNTAX ELEMENT	DEFAULT
x-pos	0
y-pos	0
magnification	1
orientation	0
colour	current foreground colour set from BASIC


The parameters x-pos and y-pos assume their default values if they are not specified. If the command is PLOAded and x-pos and y-pos are not specified they will assume their default values if the LABEL command is being entered for the first time after PLOADing.

For successive LABEL commands these unspecified parameters will assume the value corresponding to the character position following the last character of the preceding label string displayed.

Orientation

The following table illustrates orientation.

ORIENTATION	DISPLAY
0	ABC



1	QWERTY
2	ASDFGH

Examples

IF you enter...	THEN...
la 'x-axis',,,8 /CR/ (in PCOS)	if the utility has not been PLOADED so that the given defaults apply, the "x axis" will be displayed starting at 0,0, with a magnification of 8 and orientation 0
SCALE 0,1,0,1 /CR/ CALL "la" ("title", SCALEX(.25), 1,0) /CR/ (in BASIC)	the label string "title" is displayed at magnification one and orientation 0 starting at a point on the screen one quarter of the way along the x-axis (0.25) and three quarters of the way along the y-axis (0.75) Note: In BASIC, strings must always be enclosed in double quotation marks
la 'name',,,,1 /CR/	the string "name" will be displayed with normal orientation at co-ordinate (0,0) and without magnification. The last parameter indicates the colour in which the string is to be displayed

Remarks

LABEL is the only way that standard characters can be printed in graphics mode. It allows a string to be printed in various orientations and magnifications.

The string parameter must be specified.

If the parameters of a LABEL command are such that the label string does not entirely fit on the screen or within the specified window, the command will still be executed, and the label string will be displayed with the part that falls outside the screen or window clipped.

Windows cannot be opened under PCOS.

The hardware co-ordinate system is set only if one of the following conditions applies:

- the system is in the PCOS environment
- the system is in the BASIC environment and
 - . the video has not been split into windows
 - . 512 x 256 display mode is in use

Prints just the text displayed on the screen or specified window. Graphic elements are ignored.

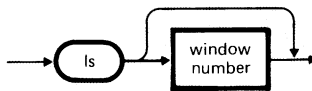


Fig. 13-27 LSCREEN.CMD

Where

SYNTAX ELEMENT	MEANING
window number	the number of the window to be printed. If this parameter is omitted then the current window will be printed

Characteristics

The specified window may contain text with any spacing. Any screen data within the normal 5 by 10 character dot matrix that is not recognisable as text will not be printed.

On colour systems, only screen plane 0 will be read. This means that if the background colour is even then the foreground colour must be odd, and vice versa.

The command does not distinguish highlighted displays on the screen, but both normal and inverse video characters will be recognised and printed.

Example

IF you enter...	THEN...
EXEC "ls 3" /CR/ (in BASIC)	the text included within window number 3 will be printed

Remark

Windows cannot be opened in PCOS.

Returns an integer (0, 1 or 2) corresponding to which of the three carriage returns (/↵/, /S1/ or /S2/) was last used.

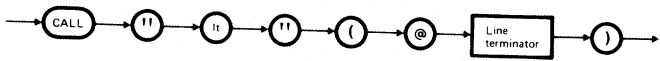


Fig. 13-28 LTERM

Where

SYNTAX ELEMENT	MEANING
line terminator	an integer variable which stores the value 0, 1 or 2, depending on whether the last line of input was terminated by /↵/, /S1/, or /S2/, respectively

Characteristics

The command is only usable in BASIC, using the CALL statement.

In a situation where a BASIC program prompts the user for an entry of some sort, the LTERM command can subsequently be CALLED to process the entry in one of three ways depending on which of the three line terminator keys was used to close the entry.

Where

SYNTAX ELEMENT	MEANING
character code	the decimal ASCII code, from 0 to 255, or hexadecimal code, from 0 to FF preceded by & corresponding to a key or combination of keys which are to be replaced
keyboard character	any of the characters on the keyboard, provided it has an ASCII equivalent (for example, A,a) excluding /RESET/, /CTRL/, /COMMAND/, /SHIFT/, /S1/, /S2/ and /↵/ to be assigned new representation
replacement character code	the ASCII code complying with the conditions described for "character code" which is to replace the original character
string	<p>any string of characters the user finds useful to have available at a single key-stroke. The string can be up to 250 characters long</p> <p>The string must be entered with leading and trailing quotation marks</p>
c	all strings previously assigned using the PKEY command are cancelled

Characteristics

Any PKEY command is cancelled by entering the character code parameter only. Since the entry of a character that has had a PKEY operation carried out on it causes the PKEYed string to be displayed, pk 'keyboard character' will not cancel the command. The character code itself must be used.

All assigned strings can be cancelled by specifying the %c program flag in the command line.

A key that already has a string assigned to it can be redefined using another PKEY command specifying the character code and the new string. It is not necessary to cancel the first string.

If the PKEY command is entered without a parameter then the defined keys are displayed along with their assigned strings. ANY VALUE ASSIGNED TO A KEY USING THE PKEY COMMAND WILL BE EFFECTIVE EVEN AFTER ENTERING ANOTHER ENVIRONMENT OR APPLICATION PROGRAM. THIS ENABLES THE USER TO, FOR INSTANCE, RE-ARRANGE FUNCTION KEYS AT WILL, BUT THE USER MUST TAKE CARE NOT TO DISABLE FUNCTIONS THAT WILL BE REQUIRED ON ENTERING THE NEW ENVIRONMENT/APPLICATION PROGRAM.

Examples

IF you enter...	THEN...
pk &41,&42 /CR/	each time "A" is pressed "B" appears. The ASCII character "A" is represented as hexadecimal &41, and "B" as &42

<p>pk '#','ba',13,10, 'files',13,10 /CR/</p>	<p>this time the keyboard character '#' is used, and is therefore enclosed within quotation marks. 13 is the decimal code for "carriage-return", 10 is the code for "line-feed". The command thus reads; when # is entered, display "ba", do a carriage-return/line-feed, display "files" and execute a further carriage-return/line-feed. The result is that the BASIC interpreter is entered and the command FILES is then executed on the volume inserted in the last selected drive</p>
<p>pk 237,'FILES "1:"', 13,10, /CR/</p>	<p>this command assigns the key combination /COMMAND/ /!1/ the string 'FILES "1:"',13,10</p> <p>Thus, when in the BASIC environment, the key combination /COMMAND/ /!1/ can be used to list the directory on the diskette inserted in drive 1</p>
<p>pk &41 /CR/</p>	<p>the effect set up in the first example above is cancelled</p>
<p>pk &23 /CR/</p>	<p>the effect set up in the second example above is cancelled.</p> <p>(# is represented by hex code &23)</p>
<p>pk /CR/</p>	<p>all defined keys are displayed along with their assigned strings</p>
<p>pk %c /CR/</p>	<p>all defined keys are cleared</p>

Loads a diskette-based or hard disk-based utility into memory.

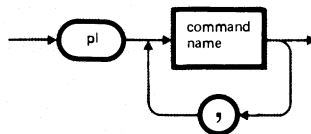


Fig. 13-30 PLOAD

Where

SYNTAX ELEMENT	MEANING
command name	the name of a transient command to be PLOADED

Characteristics

When a utility is PLOADED, it remains loaded in memory until either it is removed by the PUNLOAD command, or the current working session is terminated.

Once a utility has been PLOADED, the M20 displays the disk file name of the PLOADED utility (for example VCOPY.CMD), the program name (for example Volume Copy Rev.3.x), the operation mode (for example segmented/system), the entry points and memory allocated.

Example

IF you enter...	THEN...
pl vc /CR/	the VCOPY command is loaded into memory for the duration of the current working session.

Remarks

The user may wish to PLOAD a utility for the following reasons:

- to use the utility after the diskette on which it resides has been removed
- to save the time lost in having to load the utility from diskette (or hard disk) every time it is used

However, PLOADing utilities has the adverse effect of reducing user memory.

None of the utilities have to be PLOADed in order to be executed. If they are invoked without being PLOADed, they will be automatically sought, loaded and executed. Utilities having the CMD extension will then be removed from memory. But those having the SAV extension will remain until the end of the current working session.

The PLOAD command only works for transient commands. PLOAD, PUNLOAD and LTERM are all resident and thus cannot be PLOADed.

Loads and initialises alternative operating systems.



Fig. 13-31 PRUN

Where

SYNTAX ELEMENT	MEANING
file identifier	specifies the file containing the operating system to be loaded

Example

IF you enter...	THEN...
pr 0:alt.1 /CR/	the operating system previously saved in file "alt.1" on the diskette inserted in drive 0 is loaded and initialised

Characteristics

The command is used in conjunction with the PSAVE command. That is, an operating system must have been previously configured and then saved

using the PSAVE command before it can be reloaded and initialised by the PRUN command.

This command enables the user to configure different operating systems for different functions, then to alternate between the different operating systems. The following table shows an example of this feature:

STEP	OPERATION
1	Define an alternative operating system by setting global parameters, assigning functions to keys, and PLOADing selected commands
2	Save the new operating system in, for instance, file "alt.1" on the diskette inserted in, for instance, drive 1 by entering ps 1:alt.1 /CR/
3	Define another alternative operating system by setting global parameters, assigning functions to keys and PLOADing selected commands
4	Save it, for instance, on the same diskette by entering ps 1:alt.2 /CR/
5	Enter pr 1:alt.1 /CR/ to reload and initialise the operating system saved in "alt.1"
6	Enter pr 1:alt.2 /CR/ to reload and initialise the operating system saved in "alt.2"

Saves the current PCOS on an unprotected or enabled volume.

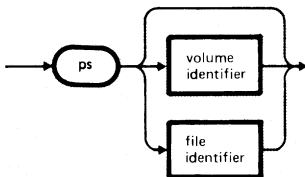


Fig. 13-32 PSAVE

Where

SYNTAX ELEMENT	MEANING
volume identifier	<p>a volume name or a drive number specifying the volume on which the current PCOS configuration is to be saved.</p> <p>If this parameter is omitted the PSAVE command will select drive 0 (even if a hard disk unit is configured)</p>
file identifier	<p>the name of the file which is to contain the current PCOS configuration, possibly preceded by a volume identifier.</p> <p>If the file identifier is omitted, the PSAVE command will assign the name PCOS.SAV</p>

Characteristics

The PSAVEd file will include any utilities which have been PLOADed, any programmed key definitions in effect, and the state of the system as set by the global commands (see Chapter 6). Moreover, any currently active user-defined font, and any globally specified device re-routing will also be included, as will control character display, if currently specified.

After entering a PSAVE command, the user is asked to confirm the intention to save PCOS on the specified file.

Save system on file drive number:file name ? (y / any other key)

A "y /CR/" response will complete the operation assuming the corresponding drive is ready, and that the specified volume is either unprotected or enabled. Entering any other key will abort the operation without error.

A PSAVEd file will follow other files that may already exist on the volume in question, but it is normal practice to PSAVE onto an empty diskette.

If a file with the same name as the PSAVEd file already exists on the specified volume, it will be overwritten. In any case, once the file is PSAVEd, the system is immediately re-booted via a standard boot file search (see Chapter 5). Furthermore, any future attempt to boot the system from this volume will cause the last PSAVEd file to be booted (unless the PRUN command is used).

Both write-protection and password protection can be applied to a PSAVEd file to protect it from being overwritten or modified by unauthorised users.

The no-interaction flag (%n) has no effect with the PSAVE command.

Examples

IF you enter...	THEN...
ps /CR/	the current PCOS configuration is saved on the diskette inserted in drive 0 in a file called PCOS.SAV as no file is specified
ps 1:alt1 /CR/	the current PCOS configuration is saved on the diskette inserted in drive 1 in a file called "alt1"

Remarks

You may find it is useful to use PSAVE in order to set the value of global parameters, for example, the default value for memory size, which may be set too low for some activities in BASIC.

Removes PLOADED commands from memory.

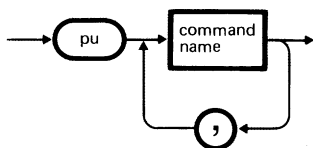


Fig. 13-33 PUNLOAD

Where

SYNTAX ELEMENT	MEANING
command name	the name of a PLOAded command to be made transient. Wild cards are not permitted

Characteristics

This command can be used to remove from memory commands that have either been PLOAded during the current working session or PLOAded and PSAVED during a previous working session. However, some commands change PCOS tables and therefore cannot be removed from memory. That is, the PUNLOAD command will not work on the following commands:

- CI
- RS232
- IEEE
- EPRINT
- VMOVE
- PDEBUG (not included on the system diskette)

Furthermore, the permanently resident commands PLOAD, PUNLOAD and LTERM cannot be removed from memory.

Example

IF you enter...	THEN...
<pre>pl vc,ps /CR/ . . . pu vc /CR/</pre>	<p>the transient commands VCOPY and PSAVE are loaded into memory</p> <p>the M20 responds</p> <p style="padding-left: 40px;">PUnloading vcopy.cmd</p> <p>and the VCOPY command is removed from memory and becomes transient once again</p>

Converts the system font data into a text file that describes each font character as a series of upper-case 'X's on a 10 by 8 matrix. This is the first step in creating a user-defined font.

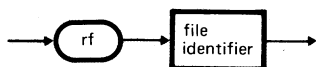


Fig. 13-34 RFONT

Where

SYNTAX ELEMENT	MEANING
file identifier	the name of the file in which the font matrices are to be created, plus any necessary password and volume identifier. If the file does not already exist it is created. If the file does already exist a prompt appears asking the user to confirm the intention. A "y" response overwrites the file

Characteristics

The file generated by the RFONT command comprises four lines of header information :

- the country identifier (USA, ITALY, etc.)
- the country number (see SLANG command)
- the font height (in scan lines). This must always be 10
- the number (decimal) of characters defined in the file (95 to 190)

This is followed by the font matrices. Each character is described in 11 lines, the first of which is the decimal code of the character, the remaining 10 describe the character, for example:

```
50
-----
----XXX-
---X---X
-----X
-----X-
-----X--
----X---
---XXXXX
-----
-----
```

where "50" is the decimal code of the number "2".

To change the character, invoke the Video File Editor on the file and place X's so that they show the intended appearance of the character. It is advised that the first two or three columns appear blank so that

characters do not run together. Once the changes are saved and the editor is exited, the WFONT command can then be entered to direct the system to display characters as they appear in the matrices.

The range of modified characters can be extended to twice the standard 95 character printable range. By adding 10 line font matrices (each preceded by the appropriate decimal code) to the end of the text font file, and by updating the "number of characters" that appears on the third line of the file, the desired number of characters can be defined. Confusing results may occur if matrix entries are not numbered consecutively from the decimal code 32, associated with "space" - the first printable ASCII character. Furthermore, the matrices must always be ordered in ascending ASCII sequence, otherwise the key/character correspondence will be destroyed. If the number of matrices exceeds the number of characters specified in the third line of the file, the extra fonts will not be defined.

Example

IF you enter...	THEN...
rf 1:italicfont /CR/	the system font data is converted into an ASCII font matrix file and saved on the disk inserted in drive 1 in the file named "italicfont"

Recovers (whenever possible) a recently deleted file.

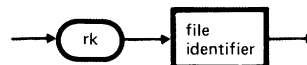


Fig. 13-35 RKILL

Where

SYNTAX ELEMENT	MEANING
file identifier	the name of the file to be recovered. It is not necessary to specify the file password

Characteristics

RKILL will only recover files that have not yet been overwritten. Once partly overwritten, a recently deleted file (either FKILLED from PCOS or KILLED from BASIC) cannot be recovered. Moreover, a file cannot be recovered on a volume that has been alphabetised since the file was deleted.

Examples

IF you enter...	THEN...
fn 1:myfile,20 /CR/	the file "myfile" is created on the diskette inserted in drive 1
vl 1: /CR/	a list of files resident on the diskette in drive 1 is displayed. "myfile" will feature at the end of this list
fk 1:myfile /CR/	the file "myfile" is deleted from the diskette in drive 1. (This can be checked using the VLIST command)
rk 1:myfile /CR/	the file "myfile" is recovered, and it will again feature in a volume list

Remarks

RKILL will not be executed if the volume is write-protected.

Loads the RS-232-C extension package.

When called this utility allows the user to communicate, via the RS-232-C interface, with devices that are compatible with this interface. The transmission environment is set using the SCOMM command. However it is necessary to execute the RS232 command before the SCOMM command.

For more details about the RS-232-C package refer to the "I/O with External Peripherals User Guide".

Sets the BASIC environment for programming.

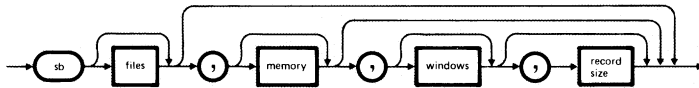


Fig. 13-36 SBASIC

Where

SYNTAX ELEMENT	MEANING	RANGE	
		MIN	MAX
files	the maximum number of files that can be opened concurrently under BASIC	0	15

memory	maximum number of bytes available for user memory in BASIC	0	The actual value can be determined using the DCONFIG command. (It can never be greater than 57K)
windows	the number of windows for which memory space is preallocated. Additional windows can always be opened if there is enough space	1	16
record size	maximum record size (in bytes) for random files	1	4096

Default Values

At initialisation from a standard PCOS these parameters have the default values shown in the table below. Alternatively, the user can establish other values using either an initialisation program (for example INIT.BAS) or by loading a user-configured PCOS (obtained using PSAVE).

In each case the value of these parameters can be changed at any moment using the SBASIC command. If this is done the values only become effective when the system goes into BASIC. If the SBASIC command is called from a BASIC program using an EXEC or a CALL statement, the newly set values will not be taken into account in the current program (otherwise the current program could be destroyed). These will become operative in subsequent programs.

A parameter, once changed, maintains its last assigned value until it is again modified or until the system is reinitialised (by switching the system ON, or by a reset). Thus nil parameters in an SBASIC command keep their last specified values in preceeding commands (or default values).

The following table gives the standard default values.

PARAMETER	DEFAULT VALUE	MEANING
files	3	a maximum of 3 files can be opened concurrently
memory	36000 bytes	PCOS has allocated 36000 bytes for BASIC programs, data which is being processed by BASIC programs, open file tables, etc.
windows	1	the first window allocated requires 38 bytes. Any additional windows for which space is allocated will require a further 108 bytes of user memory each
record size	256 bytes	the maximum record size for random files is 256 bytes

Table 13-1 BASIC Environment Default Parameters

Examples

IF you enter...	THEN...
sb /CR/	<p>the current value of the parameters will be displayed (as no parameters were specified in the command).</p> <p>For example</p> <p>files = 3 memory = 40000 windows = 3 record size = 128</p>

sb 5,38000,,80 /CR/

the command will establish that 5 files can be opened concurrently, that user memory is 38000 bytes and that the maximum record size for random files is 80 bytes. The omitted parameter (windows) will revert to the value specified in the preceeding SBASIC command since the last initialisation ,or the last SBASIC command followed by a PSAVE (if a user-configured PCOS is being used), or the default value (precedence in the order stated)

BASIC Memory

When the SBASIC command is used, and you enter BASIC, user memory is reduced by the following amount in bytes:

- $829 + (578 + R)F$

where F is the number of files that can be opened, and R is the record size

- 108 bytes for each window (except the first) that space is allocated for

It must be kept in mind that utilities, auxiliary commands and functions assigned to keys reduce user memory and therefore reduce the maximum value of the "memory" parameter.

Sets the transmission environment of an RS-232-C communications port.

The RS-232-C communications driver must be loaded prior to the execution of SCOMM.

For more details refer to the "I/O with External Peripherals User Guide".

Displays the device name table and optionally renames a device.

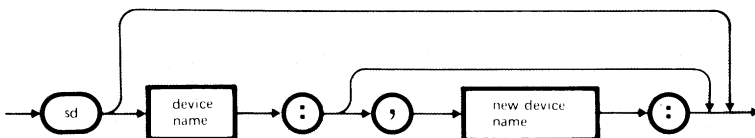


Fig. 13-37 SDEVICE

Where

SYNTAX ELEMENT	MEANING
device name	the current name of any device that exists on the device table consisting of at most 13 printable characters (see illegal characters in Chapter 4). This can be either the name assigned in the last SDEVICE command (in the current working session) or, in the absence of a preceeding SDEVICE command, the default device name. Alternatively, if a user-configured PCOS is being used, the PSAVE device name
new device name	the name to be assigned to the device in question. The name must be at most 13 characters long (for illegal characters see Chapter 4)

Characteristics

The colon (:) must follow immediately after the device name.

Default Device Names

prt: - PCOS Printer Driver
cons: - PCOS Console Driver (video and keyboard)
com: - Standard RS-232-C communication port
com1: - First RS-232-C communication port on Twin Board
com2: - Second RS-232-C communication port on Twin Board

Examples

IF you enter...	THEN...
sd /CR/	the device table will be displayed on the VDU. The information displayed gives the device name and its type; W for a write-only type of device, R for a read-only type and R/W for a device that can be read from and written to
sd prt: /CR/	the M20 will display the device name "prt:" and "W" meaning that the printer is a write-only device
sd cons:, kboard.video: /CR/	the name "cons:" of the console is changed to "kboard.video:". The M20 will display the old name and device type first, followed by the message "changed to" and the new name and device type

This command specifies the type of printer which is being used and the printing format.

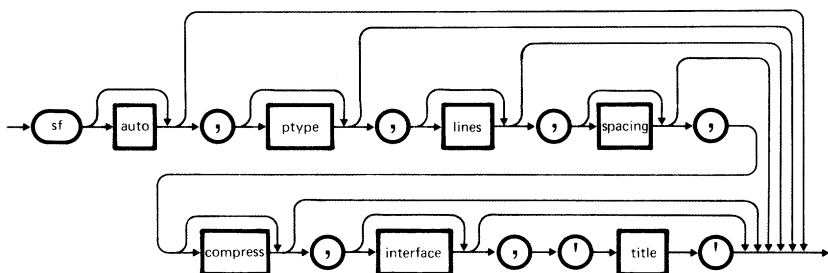


Fig. 13-38 SFORM

Where

SYNTAX ELEMENT	MEANING	DEFAULT VALUE
auto	<p>the status of the nil parameters</p> <p>'off' indicates they are to have their default values</p> <p>'on' indicates they are to have the values set by the most recent SFORM command since switch-on or, failing that, a previous SFORM command that was followed by a PSAVE command</p>	off

ptype	<p>the type of printer connected: PR1450, PR1471, PR1481, PR2400, PR2300, PR2835, ET-121, ET-231, PR-320, PR-430, transp.</p> <p>"transp" sets transparent mode. That is, files are printed in free format, irrespective of the type of printer attached (The printer types must be entered exactly as shown)</p>	PR1450
lines	the number of printed lines per page; 0 implies that no form feed will be issued	60
spacing	the number of interline spaces between printed lines. Single spacing = 1, double-spacing = 2, etc.	1
compress	<p>a two letter parameter where the first letter indicates if the characters are bold (w) or normal (n), and the second letter the type of character. This can be:</p> <ul style="list-style-type: none"> - c (compressed) = 16.6 chars/in. - e (elite) = 12 chars/in. - p (pica) = 10 chars/in. 	ne (normal, elite)

interface	<p>whether the interface through which the M20 controls the printer is serial or parallel:</p> <p>se RS-232-C (serial)</p> <p>pa Centronics type (parallel)</p>	pa
title	<p>the title to be printed at the top of each page. It must not exceed 24 characters</p>	no title

Remarks

To print a title at the top of the first page of a listing, the user must issue a form feed from BASIC; for example, "lprint chr\$(12)".

For the title format, a string of "}" deletes the previous title, without replacement.

Example

IF you enter...	THEN...
sf on,,50 /CR/	the command specifies that the nil parameters are to take the values specified in either the last SFORM command followed by a PSAVE command or the last SFORM command since switch on. The third parameter specifies the page length as 50 lines

Reconfigures the keyboard to that of another country.

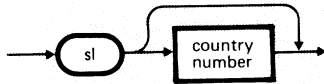


Fig. 13-39 SLANG

Where

SYNTAX ELEMENT	MEANING
country number	a number corresponding to the keyboard configuration of a particular country. See menu below

Characteristics

For the SLANG command to be executed the files "font.all" and "kb.all" must both be on an enabled volume.

The SLANG command can be executed in one of two ways:

- by entering the command without specifying the country number, in which case the following menu is displayed

Available Country Configurations :

Italy	0	Yugoslavia	10
West Germany	1	Norway	11
France	2	Greece	12
Great Britian	3	Switzerland/France ..	13
United States	4	Switzerland/Germany ..	14
Spain	5	German (original)	15
Portugal	6	Datev	16
Sweden/Finland	7	Delta	17
Denmark	8		

Enter Your Selection by Number (or q to quit) -->

The user must then select the country by entering the corresponding number followed by /CR/.

If the user enters a number that does not correspond to a supported language then the prompt re-appears.

Entering "q /CR/" quits the menu without changing the country number

- by entering the country number as a parameter in the command line. In this case an invalid country number is indicated by an error message

ERROR 76 --- error in parameter

The new keyboard configuration remains operative until either it is changed by another SLANG command, or the current working session is terminated.

The new keyboard configuration can be made permanent by means of the PSAVE command.

The keyboard layouts and the codes generated by the keys are shown for all supported national standards in Appendix B.

Examples

IF you enter...	THEN...
sl /CR/	the SLANG command displays the country number menu.
1 /CR/	The West Germany keyboard is simulated

sl 0 /CR/

the Italy keyboard is simulated

Loads and initialises the SPRINT utility to print the image of either the screen or a specified window.

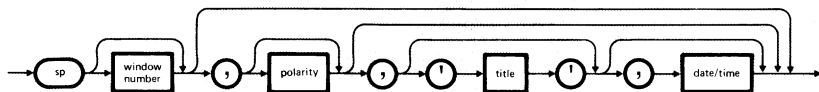


Fig. 13-40 SPRINT

Where

SYNTAX ELEMENT	MEANING	DEFAULT
window number	an integer ≥ 0 and ≤ 16 representing the window to be printed. 0 indicates the entire screen, independent as to whether or not the screen is divided into windows	0 (whole screen)

polarity	n or p, describing the paper image in monochromatic terms. Negative (n) gives black on paper for white on screen, while positive (p) gives black on paper for black on screen. For colour systems, all colours that are not the background colour are considered to be foreground	n (negative)
title	an alphanumeric title string, beginning with a non-numeric character, and no longer than 24 characters: to be printed above the graphic output	no title
date/time	dt or no determining whether (dt) or not (no) the current date and time are to be printed above the graphics output	no (not required)

Example

IF you enter...	THEN...
EXEC "sp 1,,'BAR CHART',dt" /CR/	the contents of window 1 are printed beneath the heading "BAR CHART" and the date and time

Remarks

Graphic elements can only be drawn on the PR 1450, PR 2300 and PR 2400 printers using the SPRINT command.

Windows cannot be opened in PCOS.

Sets system global parameters.

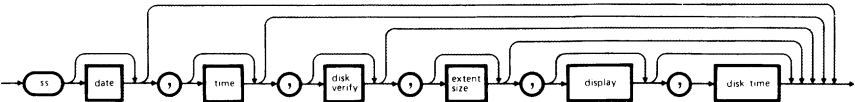


Fig. 13-41 SSYS

Where

SYNTAX ELEMENT	MEANING	RANGE	
		MIN	MAX
Date	any legal date after 1899. Date consists of month, day and year, (for USA ASCII and USA ASCII + BASIC keyboards) distinguished one from the other by a date separator. The order is day, month, year on other keyboard versions		

Time	month: is any number, representing January as 01... December as 12	01	12
	day: is any legal day according to the month of the year	01	31
	year: is any valid value in the range 1900 to 1999. (Note that only the last two digits need to be specified)	1900	1999
	date separator: is any legal name-printable character excluding digits		
	time consists of hour, minute and second, distinguished one from the other by a time separator, where:		
	hour: is any valid value	00	23
	minute: is any valid value	00	59
	second: is any valid value	00	59
	time separator: is any legal legal name-printable character excluding digits		

Disk Verify	whether or not verification is performed after each hard disk or diskette input/output operation. When verification is on, data that is written to the hard disk or diskette is subsequently read and checked. If an error is detected, "ERROR 57 --- disk i/o error" is displayed	0 (off)	1 (on)
Extent size	the number of sectors to be allocated to a file when an output operation requires additional space	1	1087
Display	whether the 64 x 16, or 80 x 25 display mode is in effect	0 (64x16)	1 (80x25)
Disk time	the number of seconds that the diskette drive motor will remain on after accessing that drive	1	30

Default Values

At initialisation from a standard PCOS, parameters attain default values shown in the following table. However, the user can assign these parameters different values at initialisation by using a user-configured PCOS, or by the use of an initialisation program (for example, INIT.BAS) which sets the parameters by calling the SSYS command.

In each case the parameter values can be changed by the user at any subsequent moment by using the SSYS command. Once modified, a parameter maintains the last value assigned to it until it is again modified by another SSYS command, or until the system is reinitialised. In an SSYS command, if a nil parameter is entered it will revert to the current value before the command was entered.

The following table gives the standard default values:

PARAMETER	DEFAULT VALUE	MEANING
Date	01/01/82	January 1 1982, and the date separator is a slash (/)
Time	00:00:01	system time starts from 1 second past midnight and the time separator is a colon (:)
Disk Verify	0 (off)	no verification after any disk or diskette read/write operation
Extent size	8	eight logically contiguous sectors will be allocated to a file when an output operation requires additional space
Display	0	16 rows of 64 characters each
Disk time	2	the diskette drive motor remains on for two seconds after the last access

Table 13-2 Set System Global Parameters Default Values

Example

IF you enter...	THEN...
ss 01/31/83, 00:00:01,0 /CR/	the command sets the date to January 31 1983, the time to one second after midnight and prescribes no verification after a read/write operation

The remaining unspecified parameters (extent size, display and disk time) will either take their default values or the values specified in the last SSYS command that was followed by a PSAVE command (if a user-configured PCOS is being used) or the last SSYS command entered during the current working session

Remark

The Disk time parameter cannot be PSAVEd.

Sorts a volume directory into alphabetic order and frees any unused directory blocks, thus improving access time whenever the directory is scanned

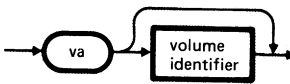


Fig. 13-42 VALPHA

Where

SYNTAX ELEMENT	MEANING
volume identifier	the volume name, or the drive number in which the volume is inserted and, if the volume is not enabled, its password. If the volume identifier is not specified the volume in the last selected drive is assumed

Example

IF you enter...	THEN...
va 1: /CR/	the directory of the diskette inserted in drive 1 will be reordered and all the files will be listed in alphabetical order

Remarks

Any volume write-protect label present must be removed before the VALPHA command can function.

The VALPHA command orders the directory such that files specified in upper case appear before those specified in lower case. On a standard system diskette the PCOS.SAV file will therefore always be placed first on the volume directory. But note that any file that starts with an upper case letter alphabetically higher than "P" will appear before the PCOS.SAV file after using the VALPHA command.

Copies a diskette from drive to drive

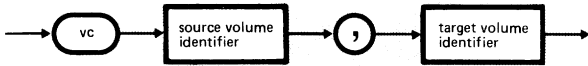


Fig. 13-43 VCOPIY

Where

SYNTAX ELEMENT	MEANING
source volume identifier	the volume name, or the drive number in which the diskette to be copied is inserted. It is not necessary to specify the password
target volume identifier	the volume name or the drive number in which the target volume is inserted. It is not necessary to specify the password. The target volume must not be write-protected

Characteristics

Before copying a diskette it is advisable to write-protect the source diskette to avoid accidentally overwriting it.

You must ensure that the copy process is being carried out in the correct direction - VCOPIY will overwrite your intended source volume with the target volume, if you instruct it to do so.

This command copies the first onto the second volume specified as long as the two volume identifiers do not select the same drive and the source volume is not of different capacity than the target volume. That is, it is only possible to use the VCOPY command when copying a 640 Kbyte diskette onto another 640 Kbyte diskette, a 320 Kbyte diskette to another 320 Kbyte diskette, or a 160 Kbyte diskette to another 160 Kbyte diskette. VCOPY does not work with the hard disk unit.

When the command is called, the screen will display the following message

Warning - VCopy deletes all files. Copy disk? (y/n)

Entering

y /CR/

starts the process.

After VCOPY, the two diskettes will be identical (except for the diskette's creation data which never changes throughout the life of a diskette).

It is not possible to copy a copy-protected diskette.

Examples

IF you enter...	THEN...
vc 1:,0: /CR/	the volume on drive 1 is copied onto the volume on drive 0
vc source/pass:, target: /CR/	the volume named "source" which has the password "pass" is copied to the volume named "target". Subsequently, they will both have the same name "source" and the same password "pass". It is not necessary, however, to specify the password "pass". If the volumes are to have different names the VRENAME command must be used. If the password is not required then VDEPASS must be used first

Deletes the password from a non write-protected volume. In order to do this the password must be known to the user.

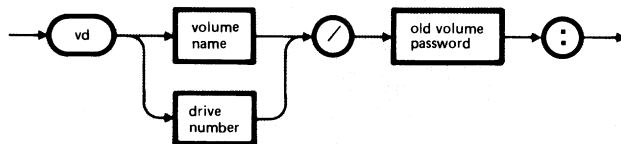


Fig. 13-44 VDEPASS

Where

SYNTAX ELEMENT	MEANING
volume name	the name of the volume whose password is to be deleted
drive number	0, 1 or 10 depending on the drive in which the diskette or hard disk resides
old volume password	the password that is to be deleted. The password must be known if it is to be deleted

Example

IF you enter...	THEN...
vd 0/mypassword: /CR/	the password "mypassword" is removed from the diskette inserted in drive 0

Remarks

VDEPASS will not operate if a volume write-protect label is present. This should be removed and replaced again after the operation.

If the password is not specified correctly, error code 72 "volume not enabled" will be displayed.

Once a volume password has been removed, accessing the volume will be the same as for a volume that never had a password.

Formats an unprotected volume and creates a blank file system on a diskette or hard disk, with the option to name the volume.

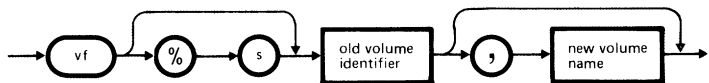


Fig. 13-45 VFORMAT

Where

SYNTAX ELEMENT	MEANING
s	the diskette is to be formatted 160 Kbytes double-density single-sided on a 320 Kbyte drive
old volume identifier	the volume name, or the drive number. It is not necessary to specify the password
new volume name	the new name which is to be assigned to the volume

Characteristics

This command is used to format 640 Kbyte diskettes, 320 Kbyte diskettes, 160 Kbyte diskettes, or the hard disk.

The formatting process puts an empty directory on the diskette or hard disk and places empty sectors on the tracks. Once a used volume has been formatted, any volume name that it had will no longer apply.

If a write-protect label is present on the diskette, it must be removed before VFORMAT can function.

When the command is called, the screen will display the following message

Warning VFormat deletes all files. Format disk? (y/n)

Entering

y /CR/

starts formatting and file system initialisation immediately if the volume has no password. If the volume has a password, and "y" is entered, the following message is displayed

Diskette appears password protected. Format disk? (y/n)

Entering

y /CR/

the diskette is formatted, and the password is deleted. Once begun, the VFORMAT process cannot be aborted.

If the diskette is a new OPE formatted one, then another message is displayed before formatting can start

Diskette is OPE formatted. Format disk anyway? (y/n)

Entering y /CR/

the diskette is formatted.

(OPE formatting is the Olivetti standard for diskettes and is common with other Olivetti machines.)

Examples

IF you enter...	THEN...
vf mydisk: /CR/	the volume called "mydisk" is formatted
vf 1:,newdisk /CR/	the diskette inserted in drive 1 is formatted and assigned the name "newdisk"
vf 10: /CR/	the hard disk (if present) is formatted
vf %s 1: /CR/	one side of the diskette inserted in drive 1 is formatted. That is, the diskette is formatted 160 Kbytes. The drive must, however, be a 320 Kbyte drive

Remarks

All M20 hard disks or diskettes must be formatted before files can be created on them. As for used M20 diskettes (or hard disk), the contents of which are no longer of interest, it is enough to re-initialise them via the VNEW command in order to use them again.

Olivetti-supplied diskettes are pre-formatted to a high precision but are not initialised. They therefore do not require the VFORMAT command, but must be initialised using the VNEW command. In this way, diskettes are guaranteed to be interchangeable between different M20s.

If the %n flag is specified the diskette (or hard disk) is formatted without any prompt being issued.

Lists the directory (or a specified part thereof) of a specified volume.

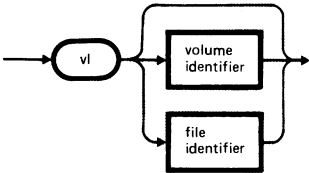


Fig. 13-46 VLIST

Where

SYNTAX ELEMENT	MEANING
volume identifier	the name of the volume to be listed or the drive number. It is not necessary to specify the volume password. If no volume identifier is specified the last selected drive is assumed

file identifier	<p>the name of a file, including any necessary volume identifier.</p> <p>Groups of files can be specified using the wild card facility, or a list of files may be specified. It is not necessary to specify the file password</p>
-----------------	---

Characteristics

In a volume list, the M20 displays the following information:

- the drive in which the volume is inserted
- the name of the volume (if it exists)
- the requested file list, specifying for each file:
 - . its name
 - . its length (in bytes)
 - . blocks used
 - . blocks allocated
 - . the number of extents
 - . whether write-protected or not (shown as WP)
 - . whether a password exists or not (shown as /PW)
- the total number of files, the total number of blocks allocated and used, and the total number of extents used by the files listed
- the total free space (in blocks) available on the volume.

If the number of files is too great to fit them all onto one screen of data then the list is displayed one screen at a time. A message is then displayed to prompt you to strike any key to examine the next screenful.

If the no-interaction (%n) flag is used then the list is scrolled until the last screen of data is displayed.

If the file identifier is not specified, then all the files resident on the volume in question are listed.

For a typical example of a volume list refer to Chapter 9.

Examples

IF you enter...	THEN...
vl /CR/	the M20 displays the directory of the volume inserted in the last drive selected
vl 1:b* /CR/	the M20 will display a volume list of all the files with a name that starts with "b", and are resident on the diskette inserted in drive 1
vl 10: /CR/	the M20 displays a volume list of all files on the hard disk

Copies a diskette using only one drive.

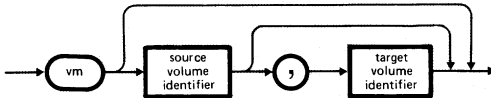


Fig. 13-47 VMOVE

Where

SYNTAX ELEMENT	MEANING
source volume identifier	the volume name or drive number (must be drive 0) of the diskette to be copied, plus any necessary volume password
target volume identifier	the volume name or drive number (must be drive 0) of the diskette to which the source volume is to be copied, plus any necessary volume password

Characteristics

Before copying a diskette it is advisable to write-protect the source diskette to prevent accidentally overwriting it.

As VMOVE can do all that VCOPY does using only one drive, VMOVE is therefore only useful on single drive systems, or for copying diskettes on hard disk systems. It cannot be used to copy the hard disk.

If neither the source nor target volume is password protected then the command name alone is sufficient. If the source volume is password protected then the full source volume identifier - including the password - must be specified in the command line. If the target volume is password protected, then both the source and target volume identifiers must be specified.

When VMOVE is called the M20 displays a message warning the user that VMOVE deletes all files and PCOS, and asks the user to confirm the intention. Following a "y /CR/" (yes) reply, the M20 displays another message asking the user to insert the source diskette in drive 0, and to hit any key.

At this point the M20 will fill all its memory space with data from the source diskette. When the memory is full, a message will be displayed asking the user to insert the target diskette in drive 0 and to hit any key. The M20 will then transfer all data from its memory onto the target diskette.

This process will be repeated (typically) two more times. At the end of the copy operation you can either continue to make more copies, or re-boot PCOS.

Note: After VMOVE, PCOS has to be re-booted because VMOVE uses all memory space as a buffer thus overwriting PCOS.

The target diskette must not have a write-protection label applied.

Any name and/or password assigned to the source volume will be copied to the target volume.

Initialises an unprotected or enabled volume with the option to name the volume.

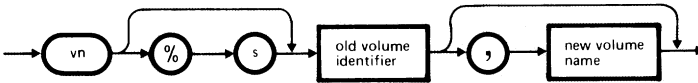


Fig. 13-48 VNEW

Where

SYNTAX ELEMENT	MEANING
s	a 160 Kbyte double-density single-sided diskette is to be initialised in a 320 Kbyte drive
old volume identifier	the volume name, or the drive number, and, if the volume is not enabled, its password
new volume name	is the new name which is to be assigned to the volume. If omitted, the new volume will have no name

Characteristics

This command can be used to initialise any diskette or the hard disk.

If a volume write-protect label is present on the diskette, it must be removed before VNEW can function.

When this command is called, the screen will display the following message:

Warning -vnew deletes all files. Initialize disk? y/n

Entering

y /CR/

starts the file system initialisation immediately. Once started, the VNEW process cannot be aborted.

Once initialised, any old volume name will no longer apply.

Examples

IF you enter...	THEN...
vn mydisk:,newdisk /CR/	the diskette called "mydisk" is initialised and renamed "newdisk"
vn 10: /CR/	the hard disk is initialised
vn %s 0: /CR/	the 160 Kbyte diskette inserted in 320 Kbyte drive 0 is initialised

Remarks

A volume can only be initialised using VNEW if it has previously been formatted. Since VFORMAT performs everything VNEW does, VNEW is normally used to re-initialise volumes which have unwanted files on them.

If the volume is not already formatted, or if there is some error at initialisation, the M20 will display an error message.

If the volume password has been forgotten VNEW cannot be used. Use VFORMAT instead.

After performing a VVERIFY destructive or non-destructive test (%d option) VNEW cannot be used. Use VFORMAT instead.



Assigns a password to an unprotected volume or changes the password of a volume that already has one.

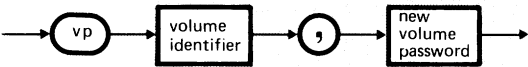


Fig. 13-49 VPASS

Where

SYNTAX ELEMENT	MEANING
volume identifier	a volume name, or the number of the drive in which the volume resides, and, if the volume has one, its password
new volume password	the password to be assigned to the volume. If the volume has a password this will be replaced by the new one

Characteristics

If a volume write-protection label is present on the diskette, it must be removed before VPASS can operate.

When a volume has a password assigned, the user will not be able to use the volume without specifying the password as part of the volume identifier. Once this has been done a first time, it need not be specified again until the volume is removed and another one is referenced on the drive unit on which the volume was inserted. This point is illustrated in the following sequence.

STEP	OPERATION
1	Insert a diskette having the password "mypass" into drive 0 and rename it by entering vr 0/mypass:,newname /CR/
2	Rename it again by entering vr 0:,testdisk /CR/ Note that the password need not be specified as the volume is already enabled
3	Remove "testdisk" from drive 0 and insert another diskette that has a password
4	Attempt to rename it by entering vr 0:,newdisk /CR/ The message ERROR 72 volume not enabled is displayed because the new diskette has not had its password specified

5	Remove "newdisk" and reinsert "testdisk"
6	<p>Attempt to rename "testdisk" by entering</p> <pre>vr testdisk:,mydisk /CR/</pre> <p>and ERROR 72 is again displayed because another diskette (see step 4) has been referred to in drive zero. The original diskette must have its password restated to re-enable it</p>

Example

IF you enter...	THEN...
vp myvol:,psswd /CR/	the volume "myvol" is given the password "psswd"

Lists all the file names, or a desired subset of the file names resident on a volume , and displays the number of free blocks on the volume.

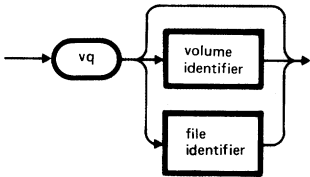


Fig. 13-50 VQUICK

Where

SYNTAX ELEMENT	MEANING
volume identifier	<p>the volume name or the drive number. If omitted, the last selected drive is assumed.</p> <p>It is not necessary to specify the volume password</p>
file identifier	<p>the name of the file plus any necessary volume identifier.</p> <p>The file name can be specified using wild card characters to specify a group of files.</p> <p>If the file identifier is not specified, then all the file names resident on the volume in question are displayed.</p> <p>It is not necessary to specify the file password</p>

Characteristics

In a volume quick list the M20 displays the drive number in which the volume in question is inserted, the volume name (if it exists) and the number of free blocks on the volume. This information is followed by a list of requested file names in four columns (for a 64 x 16 display) or five columns (for an 80 x 25 display).

The no-interaction flag (%n), if used, is ignored.

A typical volume quick list is given in Chapter 9.

Examples

IF you enter...	THEN...
vq /CR/	the M20 will display a volume quick list. The last selected drive is assumed
vq 0:*.cmd /CR/	only the files with the file name extension CMD and resident on the diskette inserted in drive 0 will be displayed in the list
vq 1:?s* /CR/	the only files listed are those resident on the diskette inserted in drive 1 and which have "s" as the second letter of their file name

Names or renames an unprotected or enabled volume.

If the new volume name is not specified, the old name will be deleted.

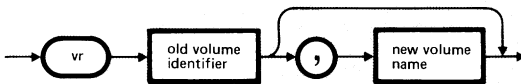


Fig. 13-51 VRENAME

Where

SYNTAX ELEMENT	MEANING
old volume identifier	the old volume name, or the drive number and, if the volume is not enabled, its password
new volume name	the name the volume is to be given. Leaving this parameter out results in the volume being given no name and it can then only be referenced by the drive number (and its password if it has one)

Examples

IF you enter...	THEN...
vr 0: /CR/	the name is removed from the diskette in drive 0
vr 0:,mydisk /CR/	the name "mydisk" is assigned to the diskette inserted in drive 0
vr mydisk:,dl /CR/	the name is changed from "mydisk" to "dl"

Remarks

Any volume write-protect label present must be removed before VRENAME can function.

A volume can be named (again, or for the first time) by referencing either the drive number, or by specifying a new name in the second parameter.

VRENAME changes only the volume name. The password is unaffected.

If a new volume name is not specified, the old one will be deleted and in future it will only be possible to refer to the volume by the drive number (followed by its password if it exists).



Checks the hard disk for faulty blocks.

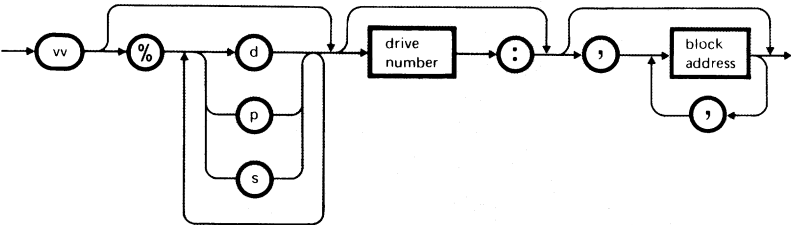


Fig. 13-52 VVERIFY

Where

SYNTAX ELEMENT	MEANING
d	destructive test. The hard disk is checked for faulty blocks. Because the destructive test destroys all the information held on the hard disk, the user is prompted to confirm the intention. Throughout the test the user is kept informed of the test's progress by a series of messages that tell the user which side of which cylinder is currently being tested. On completion of the test, a further message displays the number of bad blocks that there are on the hard disk, but not

	<p>their addresses.</p> <p>Following a destructive test, the hard disk cannot be re-used until it has been re-formatted using the VFORMAT command. Re-formatting also prevents listed bad blocks from being re-allocated</p>
p	<p>displays on the VDU the physical addresses of the faulty blocks. No test is performed. Each block address is displayed as two words, each comprising four hexadecimal digits as follows:</p> <div style="text-align: center;"> <p>word 1 word 2</p> <p>XXXX XXXX</p> <p>~~~~~ ~~~~~</p> </div> <p>cylinder number _____</p> <p>side number _____</p> <p>physical block number _____</p>
s	<p>block(s) specified by the block address parameter(s) are defined as being faulty. This is particularly useful if the user suspects some bad block(s) as it allows the block(s) to be added to the bad block list without performing the test. The specified blocks, however, will not be deallocated until the hard disk is reformatted</p>
drive number	<p>the drive number of the hard disk is always 10. If this parameter is omitted then the last selected drive is assumed</p>
block address	<p>the physical block address on the disk. It must be entered as an eight-digit hexadecimal number preceded by the letter "h". The format is as follows:</p> <div style="text-align: center;"> <p>hXXXXXXXX</p> <p>-----</p> </div> <p>cylinder number _____</p> <p>side number _____</p> <p>block number _____</p>

Characteristics

If VVERIFY is entered without parameters (other than the optional drive number parameter), then a non-destructive test is performed. Faulty blocks are listed on cylinder 0 side 0 block 1, but the information held on the hard disk remains intact. The non-destructive test, however, takes longer than the destructive test and is less exhaustive. Throughout the test, the cylinder and side number currently being tested are displayed on the VDU, and the final message displays the total number of bad blocks found. Following the test, bad blocks will continue to be allocated by the system until the VFORMAT command is used to re-format the hard disk. Before re-formatting, it is the user's responsibility to copy onto diskette the files that need to be saved.

If the contents of the hard disk are no longer required, the %d option may be specified, thereby reducing execution time. Furthermore, if the user already suspects which blocks are faulty, these blocks may be added to the bad block list using the %s option, thereby not performing any test at all.

The following are the possible program flag combinations:

- %dp - performs the destructive test then displays a list of the updated bad block list
- %pd - displays the bad block list then performs the destructive test
- %sp - adds the specified block addresses to the bad block list, then displays the updated list
- %ps - displays the bad block list then adds the specified block addresses to it

The no-interaction flag (%n) can be used to suppress interactive messages and the display of the command title. It does not suppress the display of the bad block list with the %p option, neither will it suppress the display of error messages.

The VVERIFY command can only be used on a formatted drive. Any attempt to do otherwise will result in the message

ERROR 111 --- invalid device

Examples

IF you enter...	THEN ...
vv 10: /CR/	the hard disk is checked for faulty blocks. The bad block list on side 0, cylinder 0, block 1 is updated
vv %p 10: /CR/	the bad block list held on cylinder 0, side 0, block 1 is displayed
vv %np 10: /CR/	the bad block list is displayed, but display of the command name is suppressed
vv %nd 10: /CR/	the hard disk is checked for faulty blocks. The entire contents of the hard disk are destroyed. Only error messages are displayed
vv %s h01010001 /CR/	cylinder 1, side 1, block 1 is added to the bad block list

Makes an active font from an ASCII file which was first created by the RFONT command, then subsequently modified using the Video File Editor.

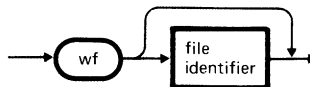


Fig. 13-53 WFONT

Where

SYNTAX ELEMENT	MEANING
file identifier	the file containing the character font matrices. This must be complete with any necessary volume identifier and file password. The file must be structured like the output from an RFONT command. If the volume is not enabled the volume password must also be specified. If the file identifier is omitted the system will return to the font that was active at initialisation

Characteristics

At the beginning of the file is a four line header. All but the character count line can be modified without affecting the program.

- line 1: Any text that describes the file. (The content is ignored by the WFONT command. It is there for reference only.)
- line 2: The country number that was active when the file was created. (The content is ignored by the WFONT command. It is for reference only.)
- line 3: The height (in lines) of a valid font matrix - 10 lines. (The content is ignored by the WFONT command.)
- line 4: the character count followed by at least one other word (like "characters"). The count must match the number of font matrices that follow. The minimum is 95 characters and the maximum is 190 characters

Example

```
USA
4
10
95 ch
```


Each matrix is defined as follows:

- line 1: the character number. This is not read by the WFONT command. Changing this number will have no effect, as long as the line is not deleted altogether
- line 2 - 11: Any combination of X's (upper case only) and -'s are acceptable

Example of a Matrix

```

50
-----
----XXX-
---X---X
-----X
-----X-
-----X--
-----X
----XXXXX
-----
-----

```

When the command is entered, assuming the specified font matrix file is on an active volume and that enough memory is available, the character code will be read, converted to binary, and written into memory.

Once execution is completed, the new fonts will replace those known to the system from initialisation. The system will return to the font that was active at initialisation when re-initialisation occurs, or when the WFONT command is invoked with no parameter.

User-defined fonts can be made permanent by means of the PSAVE command.

Examples

IF you enter...	THEN...
wf italicfont /CR/	the font matrix file named "italicfont" is converted to binary and made active
wf /CR/	the original system font is re-activated

Remarks

1. In 64 x 16 display mode, the leftmost 3 columns are generally reserved for spacing between characters and should not be used except in special cases where regular spacing between characters is not desired
2. In 80 x 25 display mode, the leftmost two columns are not displayed at all. The third column should be left empty for spacing, unless connected characters are desired
3. For fonts that will work well in either display mode, the user is urged to confine X's to the right-most 5 columns
4. If a numeric character is placed after the 8th character but not by means of the editor's "insert mode", it is ignored and no error will be issued. The same is not true however, for non-numeric characters
5. More font matrices can be added to a font file by adding a new font matrix at the end of the file (with a character number) and by updating the character count (4th header line). If a font matrix is inserted between two existing matrices, this serves only to increment the character numbers of all matrices beyond the one inserted after Write Font is invoked with this file. Any character numbers that appear before each matrix are not interpreted and are only for reference. The added characters are read by position. That is, the character following the one referenced to ASCII code 127, will correspond to code 128, whatever the character number is
6. Within the range of character codes 127 through 222, many character codes do not have keys that correspond to them. If it is desired to have a key or a key combination that corresponds to one of the character codes in this range, use the CKEY command

Memory Usage

The WFONT command allocates user memory each time it is invoked with a valid font file. This memory space is released either by re-initialising PCOS, or by invoking the WFONT command with no parameter. In order to save memory, it is advisable to release space allocated by the WFONT command before activating another user-defined font set.

Example

IF you enter...	THEN...
wf 1:script.font /CR/	script is made active
.	
.	
wf /CR/	memory for script is freed
.	
.	
wf 1:bold.font /CR/	bold print is made active
.	
.	
wf 1:script.font /CR/	script is made active, but data for bold print cannot be released from memory without re-initialising PCOS

A. ASCII CODE

ABOUT THIS APPENDIX

This appendix provides a table of ASCII codes.

CONTENTS

ASCII CODE

A-1

HBACKUP.CMD

The HBACKUP command allows the user to back up the PCOS partition of the hard disk (or the entire disk if it is all partitioned for PCOS) onto one or more diskettes. Moreover, it is the only way in which a file larger than the capacity of a diskette can be backed up.

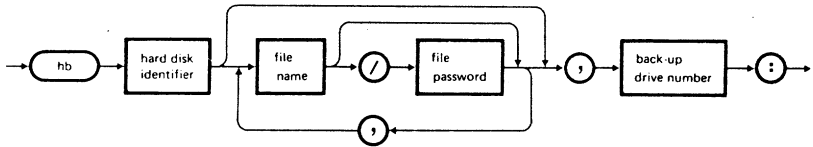


Fig. 14-23 HBACKUP

Where

SYNTAX ELEMENT	MEANING
hard disk identifier	The volume name or drive number to be backed up
file name	EITHER the name of a file to be backed up OR a file name containing wild cards to specify a group of files to be backed up
back-up drive number	The number of the diskette drive in which the backup will be made

Characteristics

If the specified diskette drive contains a blank, formatted diskette, then the backup commences immediately after the user enters the command line correctly.

If you specify less than two parameters in the command line, then the message

parameter missing

is displayed and the command terminates.

If the specified target volume identifier is not configured, then the message

destination volume missing

is displayed, and the command terminates.

If the user has entered the command line correctly but the specified drive does not contain a blank, formatted diskette, then the message

Bad disk in drive 0:

Insert a blank formatted disk and press RETURN

is displayed. Backing up will commence only after the user has inserted a blank, formatted diskette and strikes /CR/.

Once the backup has started, the HBACKUP command creates a file called "Backup.000" on the backup diskette. This is a composite file that will contain

- A table of all the names of the files on the hard disk to be backed up.
- As much of the data to be backed up as will fit onto the remainder of the diskette. This will comprise any number of the specified files, and/or a partial file. The data is held in a compressed form to save space and hence reduce the number of copies.

If all the specified files fit onto one diskette, then the backup operation terminates and control is returned to PCOS. If additional diskettes are required, however, the following prompt is displayed:

disk full - insert new disk and press RETURN

After the user inserts another blank, formatted diskette and strikes /CR/, the operation continues. The composite file on this diskette is automatically named "Backup.001." The user must then repeat this process until the backup is complete. Each subsequent diskette will contain one composite file automatically named in sequence "Backup.002," "Backup.003," etc., where each such file contains as many hard disk files and/or partial files as will fit on the diskette. Note that only the first diskette contains the table of file names.

Password-protected files may be backed up, but the password must be given. Write-protected files may be backed up, but will not be restored if a file of the same name exists on the hard disk and is still write-protected at restoration time. Copy-protected files will not be backed up by the HBACKUP command.

If the user specifies a file that does not exist on the hard disk, the message

file not found

is displayed, followed by the file name, and the backup continues.

Since the backup is saved in a format different from that of the original files on hard disk, a special command -- the HRESTORE command -- must be used to restore the backup to hard disk; that is, it is not possible to restore the backup using the FCOPY command. The backup can be copied (using the VCOPY command) **in the same format** to another set of diskettes, but cannot actually be used until it has been restored.

Examples

IF you enter...	THEN...
hb 10:,0: /CR/	The entire contents of the PCOS partition of the hard disk are backed up onto a series of diskettes in drive 0
hb 10:*.cmd,0: /CR/	All files with the file name extension ".cmd" on the PCOS partition of the hard disk will be backed up onto a series of diskettes in drive 0

Remarks

The FCOPY command can be used to back up hard disk files, provided no file to be copied is greater than the capacity of the target diskette.

If the hard disk is partitioned, only the PCOS partition can be backed up using the HBACKUP command. Other partitions must be backed up under their respective operating systems.

HDISK.CMD

Partitions the hard disk.

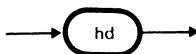


Fig. 14-24 HDISK

Characteristics

The HDISK command creates the PCOS partition on the hard disk. For operational details, refer to Chapter 12.

HRESTORE.CMD

Restores to hard disk all or part of a backup that was made using the HBACKUP command.

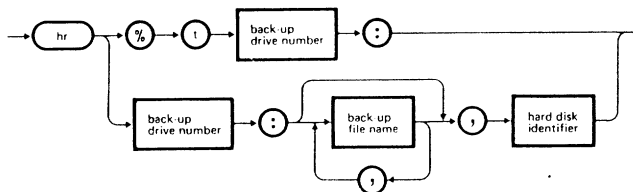


Fig. 14-25 HRESTORE

Where

SYNTAX ELEMENT	MEANING
t	A list of all files saved by the corresponding backup is displayed. No restore is performed. Only the first diskette of the corresponding backup operation is required for this operation
back-up drive number	The drive number of the diskette drive from which the backup copy is to be restored
back-up file name	<p>EITHER the name of a backed up file to be restored</p> <p>OR a group of files specified using wild card characters.</p> <p>If this parameter is omitted, then all files from the corresponding backup are restored</p>
hard disk identifier	The drive number of the hard disk to which the backup is to be restored

Characteristics

If the target volume is not specified and %t is not entered, the message
destination volume missing

is displayed and the command terminates.

If the source volume is not specified, then the message
backup volume missing

is displayed, and the command terminates.

If the drive specified by the source drive number contains the first disk of a backup session, then the restore operation begins as soon as the command line is entered. If not, then the message

please insert first back-up disk and press RETURN

will be displayed. In this case, the restore operation will begin after inserting the appropriate diskette and striking /CR/. The backup diskettes must be restored in the order in which they were saved; that is, the one containing backup file "Backup.000", followed by "Backup.001", and so on.

If the backup was made on more than one diskette, then the user is prompted to insert each diskette in turn.

If the no-interaction (%n) flag is used with this command, then files that already exist on the target volume will be overwritten without prompting.



























If a requested file is not on the backup, the message

file not found

followed by the file name is displayed, and the restore operation continues.

Examples

IF you enter...	THEN...
hr 0:,10: /CR/	The hard disk will be restored from one or more backup diskettes inserted in series in drive 0
hr 0:*.cmd,10: /CR/	All files with the file name extension ".cmd" on a series of one or more backup diskettes inserted in drive 0 are restored to the hard disk
hr %t 0: /CR/	The files contained in the backup are displayed

a	b	c	d	a	b	c	d	a	b	c	a	b	c
0	00	00000000	NUL	64	40	01000000		128	80	10000000	192	C0	11000000
1	01	00000001	SOH	65	41	01000001		129	81	10000001	193	C1	11000001
2	02	00000010	STX	66	42	01000010		130	82	10000010	194	C2	11000010
3	03	00000011	ETX	67	43	01000011		131	83	10000011	195	C3	11000011
4	04	00000100	EQT	68	44	01000100		132	84	10000100	196	C4	11000100
5	05	00000101	ENQ	69	45	01000101		133	85	10000101	197	C5	11000101
6	06	00000110	ACK	70	46	01000110		134	86	10000110	198	C6	11000110
7	07	00000111	BEL	71	47	01000111		135	87	10000111	199	C7	11000111
8	08	00001000	BS	72	48	01001000		136	88	10001000	200	C8	11000100
9	09	00001001	HT	73	49	01001001		137	89	10001001	201	C9	11000101
10	0A	00001010	LF	74	4A	01001010		138	8A	10001010	202	CA	11000110
11	0B	00001011	VT	75	4B	01001011		139	8B	10001011	203	CB	11000111
12	0C	00001100	CR	76	4C	01001100		140	8C	10001100	204	CC	11000100
13	0D	00001101	CR	77	4D	01001101		141	8D	10001101	205	CD	11000101
14	0E	00001110	SO	78	4E	01001110		142	8E	10001110	206	CE	11000110
15	0F	00001111	SI	79	4F	01001111		143	8F	10001111	207	CF	11000111
16	10	00010000	DLE	80	50	01010000		144	90	10010000	208	0D	11010000
17	11	00010001	DC1	81	51	01010001		145	91	10010001	209	D1	11010001
18	12	00010010	DC2	82	52	01010010		146	92	10010010	210	D2	11010010
19	13	00010011	DC3	83	53	01010011		147	93	10010011	211	D3	11010011
20	14	00010100	DC4	84	54	01010100		148	94	10010100	212	DA	11010100
21	15	00010101	NAK	85	55	01010101		149	95	10010101	213	D5	11010101
22	16	00010110	SYN	86	56	01010110		150	96	10010110	214	DE	11010110
23	17	00010111	ETB	87	57	01010111		151	97	10010111	215	DF	11010111
24	18	00011000	CAN	88	58	01011000		152	98	10011000	216	D8	11010100
25	19	00011001	EM	89	59	01011001		153	99	10011001	217	D9	11010101
26	1A	00011010	SUB	90									

B. NATIONAL KEYBOARDS

ABOUT THIS APPENDIX

This appendix describes all the national keyboards available with the M20.

CONTENTS

<u>ASCII CHARACTER EQUIVALENCES</u>	B-1	SWEDEN/FINLAND KEYBOARD	B-33
<u>NATIONAL KEYBOARD LAYOUTS AND CODES</u>	B-1	SWITZERLAND FRENCH KEYBOARD	B-36
DENMARK KEYBOARD	B-3	SWITZERLAND GERMAN KEYBOARD	B-39
FRANCE KEYBOARD	B-6	USA ASCII KEYBOARD	B-42
GERMANY (ORIGINAL) KEYBOARD	B-9	USA ASCII + BASIC KEYBOARD	B-45
GERMANY (WEST) KEYBOARD	B-12	YUGOSLAVIA KEYBOARD	B-48
GREAT BRITAIN KEYBOARD	B-15		
GREECE KEYBOARD	B-18		
ITALY KEYBOARD	B-21		
NORWAY KEYBOARD	B-24		
PORTUGAL KEYBOARD	B-27		
SPAIN KEYBOARD	B-30		

ASCII CHARACTER EQUIVALENCES

The following table shows the national equivalents for those ASCII characters which are displayed in various national guises.

ASCII VALUE		NATIONAL EQUIVALENT														
DECIMAL	HEXADECIMAL	USA	ITALY	FRANCE	GREAT BRITAIN	GERMANY (ORIGINAL)	GERMANY (WEST)	SPAIN	PORTUGAL	DENMARK	SWEDEN FINLAND	NORWAY	SWITZERLAND FRENCH	SWITZERLAND GERMAN	GREECE	YUGOSLAVIA
35	23	#	£	£	£	#	#	£	#	£	#	£	£	£	£	#
36	24	\$	\$	\$	\$	\$	\$	\$	\$	\$	¤	\$	\$	\$	\$	¤
64	40	@	§	à	@	§	§	§	§	'	@	'	§	§	@	§
91	5B	[o	o	[Å	Å	i	Ã	Æ	Å	Æ	à	à	[Ø
92	5C	\	ç	ç	\	Ö	Ö	Ñ	ç	Ø	Ö	Ø	ç	ç	\	Ç
93	5D]	é	§]	Ü	Ü	¿	Ö	Å	Å	Å	é	é]	Ž
96	60	~	û	~	~	~	~	~	~	~	~	~	~	~	~	ž
123	7B	{	à	é	{	ä	ä	°	ã	æ	ä	æ	ä	ä	{	đ
124	7C		ô	û		ö	ö	ñ	ç	ø	ö	ø	ö	ö		č
125	7D	}	è	è	}	ü	ü	ç	ö	ä	ä	ä	ü	ü	}	ž
126	7E	~	ì	~	~	ß	ß	~	°	~	~	~	é	é	~	č

* Encircled characters are used for functions in BASIC.

Table B-1 ASCII Character Equivalences

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 hexadecimal ASCII codes generated for each key whether struck on its own, or in conjunction with the /SHIFT/, /CTRL/, or /COMMAND/ key.

The keyboards are each toured in the same physical sequence, in ascending order of 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/ /!//.

Note that the cursor moving keys (↑, ↓, ←, →, and HOME) are not operational in the PCOS environment

DENMARK KEYBOARD

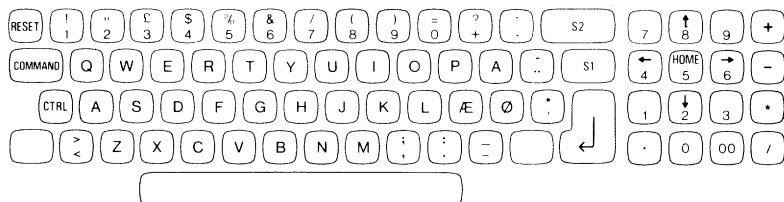


Fig. B-1 Denmark Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	Df
01	<	3C	3E	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

19	x	78	58	18	97
1A	y	79	59	19	98
1B	z	7A	5A	1A	99
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	+	2B	3F	EA	F6
27	/	40	60	EB	F7
28	8	7D	5D	00	13
29	..	7E	5E	FB	1C
2A	æ	7B	5B	1E	FC
2B	ø	7C	5C	1F	FD
2C	,	27	2A	1D	9F
2D	,	2C	3B	FE	F9
2E	.	2E	3A	FF	FA
2F	-	2D	5F	A4	A5
C0	SPACE	20	20	20	20
C1	←	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

FRANCE KEYBOARD

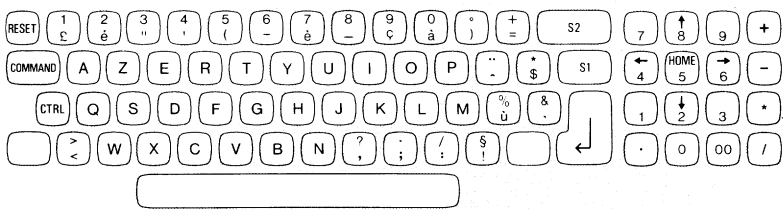


Fig. B-2 France Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	<	3C	3E	7F	F8
02	q	71	51	11	90
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	,	2C	3F	1E	FC
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	a	61	41	01	80
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	z	7A	5A	1A	99

NATIONAL KEYBOARDS

19	x	78	58	18	97
1A	y	79	59	19	98
1B	w	77	57	17	96
1C	à	40	30	E0	EC
1D	é	23	31	E1	ED
1E	è	7B	32	E2	EE
1F	"	22	33	E3	EF
20	'	27	34	E4	F0
21	(28	35	E5	F1
22	-	2D	36	E6	F2
23	è	7D	37	E7	F3
24		5F	38	E8	F4
25	ç	5C	39	E9	F5
26)	29	5B	EA	F6
27	=	3D	2B	EB	F7
28	,	5E	7E	00	13
29	\$	24	2A	FB	1C
2A	m	6D	4D	0D	8C
2B	û	7C	25	1F	FD
2C	\	60	26	1D	9F
2D	;	3B	2E	FE	F9
2E	:	3A	2F	FF	FA
2F	!	21	5D	A4	A5
C0	SPACE	20	20	20	20
C1	←	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

NATIONAL KEYBOARDS

GERMANY (ORIGINAL) KEYBOARD

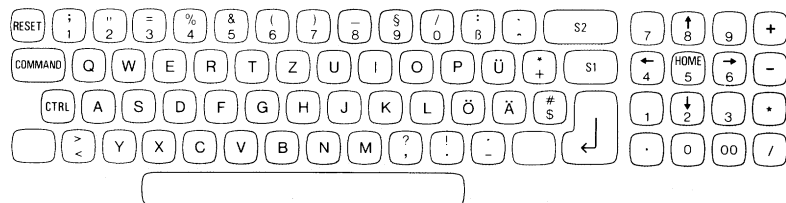


Fig. B-3 Germany (Original) Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	<	3C	3E	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

19	x	78	58	18	97
1A	z	7A	5A	1A	99
1B	y	79	59	19	98
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	ß	7E	3A	EA	F6
27	^	5E	60	EB	F7
28	ü	7D	5D	00	13
29	+	2B	2A	FB	1C
2A	ö	7C	5C	1E	FC
2B	ä	7B	5B	1F	FD
2C	\$	24	23	1D	9F
2D	,	2C	3F	FE	F9
2E	.	2E	21	FF	FA
2F	-	2D	27	A4	A5
C0	SPACE	20	20	20	20
C1	↵	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

—



→

NATIONAL KEYBOARDS

19	x	78	58	18	97
1A	z	7A	5A	1A	99
1B	y	79	59	19	98
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	ß	7E	3F	EA	F6
27	,	27	60	EB	F7
28	ü	7D	5D	00	13
29	+	2B	2A	FB	1C
2A	ö	7C	5C	1E	FC
2B	ä	7B	5B	1F	FD
2C	£	23	5E	1D	9F
2D	,	2C	3B	FE	F9
2E	.	2E	3A	FF	FA
2F	-	2D	5F	A4	A5
C0	SPACE	20	20	20	20
C1	← S1	A7	A7	A7	A7
C2		A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

GREAT BRITAIN KEYBOARD

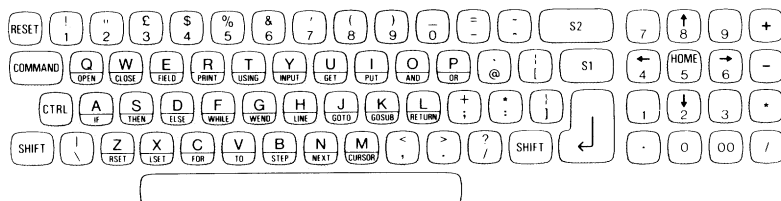


Fig. 0-5 Great Britain Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01		5C	7C	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

19	x	78	58	18	97
1A	y	79	59	19	98
1B	z	7A	5A	1A	99
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	-	2D	3D	EA	F6
27	~	58	7E	EB	F7
28	@	40	60	00	13
29	[5B	7B	FB	1C
2A	;	3B	2B	1E	FC
2B	:	3A	2A	1F	FD
2C]	5D	7D	1D	9F
2D	,	2C	3C	FE	F9
2E	.	2E	3E	FF	FA
2F	/	2F	3F	A4	A5
C0	SPACE	20	20	20	20
C1	↵	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

GREECE KEYBOARD

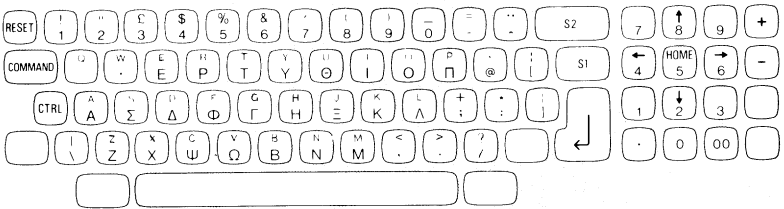


Fig. B-6 Greece Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	\	5C	7C	7F	F8
02	A	C0	41	01	80
03	B	C1	42	A1	81
04	Ψ	C2	43	A2	82
05	Δ	C3	44	04	83
06	E	C4	45	05	84
07	Φ	C5	46	06	85
08	Γ	C6	47	07	86
09	H	C7	48	08	87
0A	I	C8	49	09	88
0B	≡	C9	4A	0A	89
0C	K	CA	4B	0B	8A
0D	Λ	CB	4C	0C	8B
0E	M	CC	4D	0D	8C
0F	N	CD	4E	0E	8D
10	O	CE	4F	0F	8E
11	Π	CF	50	10	8F
12		AF	51	11	90
13	P	D0	52	12	91
14	T	D1	53	A3	92
15		D2	54	14	93
16	⊖	D3	55	15	94
17	.	D4	56	16	95
18		D5	57	17	96

NATIONAL KEYBOARDS

19	X	D6	58	18	97
1A	Y	D7	59	19	98
1B	Z	D8	5A	1A	99
1C	0	30	5F	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	27	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	~	2D	3D	EA	F6
27	^	5E	D9	EB	F7
28	@	40	60	00	13
29	[5B	7B	FB	1C
2A	;	3B	2B	1E	FC
2B	:	3A	2A	1F	FD
2C]	5D	7D	1D	9F
2D	,	2C	3C	FE	F9
2E	.	2E	3E	FF	FA
2F	/	2F	3F	A4	A5
C0	SPACE	20	20	20	20
C1	↵	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

ITALY KEYBOARD

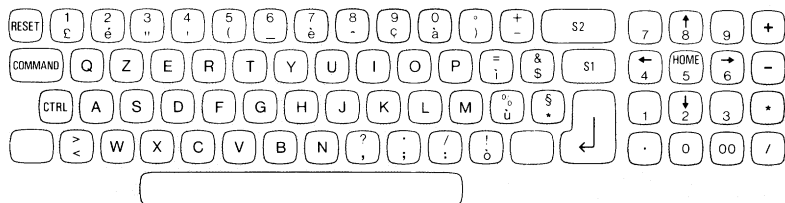


Fig. B-7 Italy Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	<	3C	3E	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	,	2C	3F	1E	FC
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	z	7A	5A	1A	99

19	x	78	58	18	97
1A	y	79	59	19	98
1B	w	77	57	17	96
1C	a	7B	30	E0	EC
1D	f	23	31	E1	ED
1E	e	5D	32	E2	EE
1F	"	22	33	E3	EF
20	'	27	34	E4	F0
21	(28	35	E5	F1
22		5F	36	E6	F2
23	~	7D	37	E7	F3
24	e	5E	38	E8	F4
25	ç	5C	39	E9	F5
26)	29	5B	EA	F6
27	-	2D	2B	EB	F7
28	i	7E	3D	00	13
29	\$	24	26	FB	1C
2A	m	6D	4D	0D	8C
2B	u	60	25	1F	FD
2C	*	2A	40	1D	9F
2D	;	3B	2E	FE	F9
2E	:	3A	2F	FF	FA
2F	ö	7C	21	A4	A5
C0	SPACE	20	20	20	20
C1	↵	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

NORWAY KEYBOARD

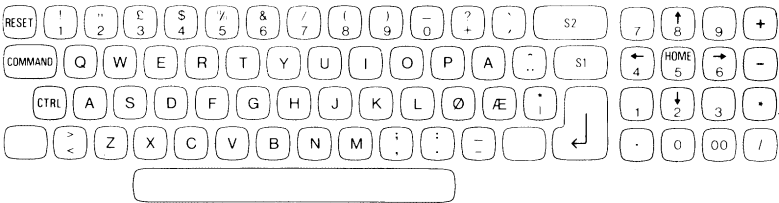


Fig. B-8 Norway Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	<	3C	3E	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

NATIONAL KEYBOARDS

19	x	78	58	18	97
1A	y	79	59	19	98
1B	z	7A	5A	1A	99
1C	0	7B	5B	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	+	2B	3F	EA	F6
27	,	40	60	EB	F7
28	&	7D	5D	00	13
29	..	7E	5E	FB	1C
2A	ø	7C	5C	1F	FD
2B	æ	7B	5B	1E	FC
2C	'	27	2A	1D	9F
2D	,	2C	3B	FE	F9
2E	.	2E	3A	FF	FA
2F	-	2D	5F	A4	A5
C0	SPACE	20	20	20	20
C1		A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

PORTUGAL KEYBOARD

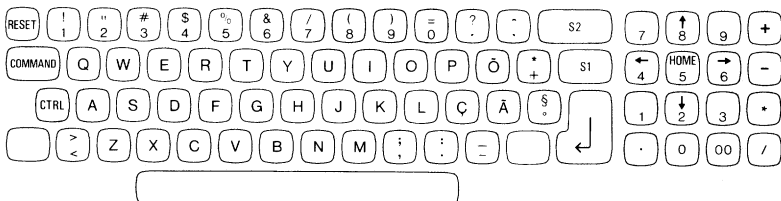


Fig. B-9 Portugal Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	<	3C	3E	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

19	x	78	58	18	97
1A	y	79	59	19	98
1B	z	7A	5A	1A	99
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	/	27	3F	EA	F6
27	\	60	5E	EB	F7
28	õ	7D	5D	00	13
29	+	2B	2A	FB	1C
2A	ç	7C	5C	1E	FC
2B	ã	7B	5B	1F	FD
2C	o	7E	40	1D	9F
2D	,	2C	3B	FE	F9
2E	.	2E	3A	FF	FA
2F	-	2D	5F	A4	A5
C0	SPACE	20	20	20	20
C1	↵	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

SPAIN KEYBOARD

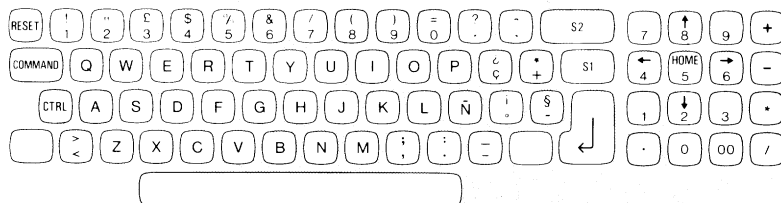


Fig. B-10 Spain Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	<	3C	3E	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

NATIONAL KEYBOARDS

19	x	78	58	18	97
1A	y	79	59	19	98
1B	z	7A	5A	1A	99
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	/	27	3F	EA	F6
27	\	60	5E	EB	F7
28	ç	7D	5D	00	13
29	+	2B	2A	FB	1C
2A	ñ	7C	5C	1E	FC
2B	ó	7B	5B	1F	FD
2C	~	7E	40	1D	9F
2D	,	2C	3B	FE	F9
2E	.	2E	3A	FF	FA
2F	-	2D	5F	A4	A5
C0	SPACE	20	20	20	20
C1		A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	Bf	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

SWEDEN/FINLAND KEYBOARD

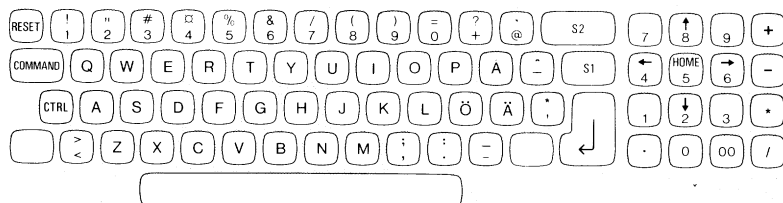


Fig. B-11 Sweden/Finland Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	<	3C	3E	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

19	x	78	58	18	97
1A	y	79	59	19	98
1B	z	7A	5A	1A	99
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	+	2B	3F	EA	F6
27	@	40	60	EB	F7
28	§	7D	5D	00	13
29	-	7E	5E	FB	1C
2A	ö	7C	5C	1E	FC
2B	ä	7B	5B	1F	FD
2C	,	27	2A	1D	9F
2D	,	2C	3B	FE	F9
2E	.	2E	3A	FF	FA
2F	-	2D	5F	A4	A5
C0	SPACE	20	20	20	20
C1	↵	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

SWITZERLAND FRENCH KEYBOARD

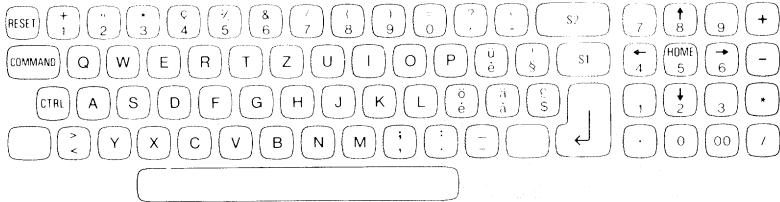


Fig. B-12 Switzerland French Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	<	3C	3E	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

NATIONAL KEYBOARDS LAYOUTS AND CODES

19	x	78	58	18	97
1A	z	7A	5A	1A	99
1B	y	79	59	19	98
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	/	27	3F	EA	F6
27	^	5E	60	EB	F7
28	é	5D	7D	00	13
29	§	40	21	FB	1C
2A	è	7E	7C	1E	FC
2B	à	5B	7B	1F	FD
2C	\$	24	23	1D	9F
2D	;	2C	3B	FE	F9
2E	.	2E	3A	FF	FA
2F	-	2D	5F	A4	A5
C0	SPACE	20	20	20	20
C1	↵	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

SWITZERLAND GERMAN KEYBOARD

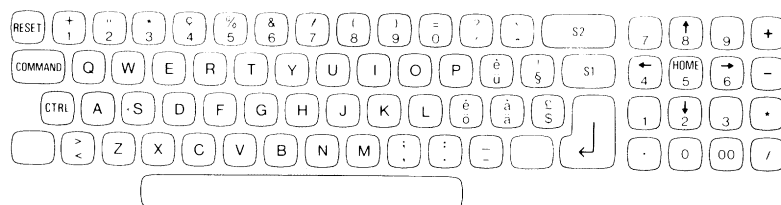


Fig. B-13 Switzerland German Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	<	3C	3E	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

19	x	78	58	18	97
1A	z	7A	5A	1A	99
1B	y	79	59	19	98
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	'	27	3F	EA	F6
27	^	5E	60	EB	F7
28	ü	7D	5D	00	13
29	§	40	21	FB	1C
2A	ö	7C	7E	1E	FC
2B	ä	7B	5B	1F	FD
2C	§	24	23	1D	9F
2D	,	2C	3B	FE	F9
2E	.	2E	3A	FF	FA
2F	-	2D	5F	A4	A5
C0	SPACE	20	20	20	20
C1	↵	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

USA ASCII KEYBOARD

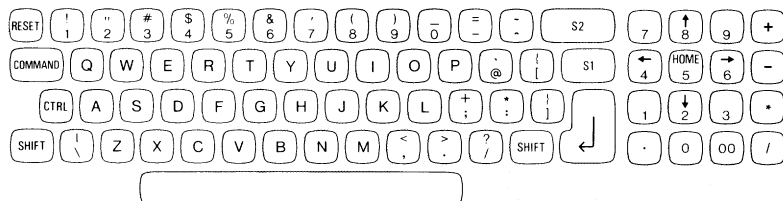


Fig. B-14 USA ASCII Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	\	5C	7C	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

NATIONAL KEYBOARDS

19	x	78	58	18	97
1A	y	79	59	19	98
1B	z	7A	5A	1A	99
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	-	2D	3D	EA	F6
27	^	5E	7E	EB	F7
28	@	40	60	00	13
29	[5B	7B	FB	1C
2A	;	3B	2B	1E	FC
2B	:	3A	2A	1F	FD
2C]	5D	7D	1D	9F
2D	,	2C	3C	FE	F9
2E	.	2E	3E	FF	FA
2F	/	2F	3F	A4	A5
C0	/ SPACE	20	20	20	20
C1	↵	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

USA ASCII + BASIC KEYBOARD

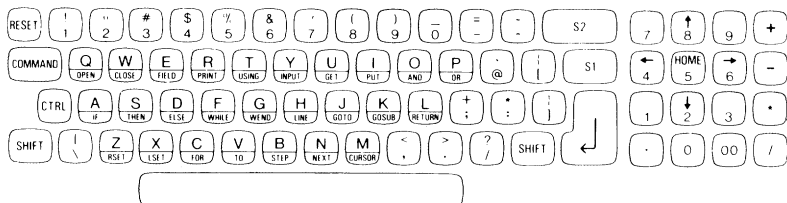


Fig. B-15 USA ASCII + BASIC Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	\	5C	7C	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

19	x	78	58	18	97
1A	y	79	59	19	98
1B	z	7A	5A	1A	99
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	~	2D	3D	EA	F6
27	^	5E	7E	EB	F7
28	@	40	60	00	13
29	[5B	7B	FB	1C
2A	;	3B	2B	1E	FC
2B	:	3A	2A	1F	FD
2C]	5D	7D	1D	9F
2D	,	2C	3C	FE	F9
2E	.	2E	3E	FF	FA
2F	/	2F	3F	A4	A5
C0	SPACE	20	20	20	20
C1	↵	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

YUGOSLAVIA KEYBOARD

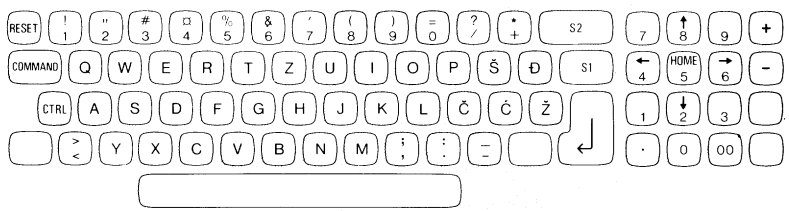


Fig. B-16 Yugoslavia Keyboard

Alphanumeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
00	RESET	DD	DE	A0	DF
01	<	3C	3E	7F	F8
02	a	61	41	01	80
03	b	62	42	A1	81
04	c	63	43	A2	82
05	d	64	44	04	83
06	e	65	45	05	84
07	f	66	46	06	85
08	g	67	47	07	86
09	h	68	48	08	87
0A	i	69	49	09	88
0B	j	6A	4A	0A	89
0C	k	6B	4B	0B	8A
0D	l	6C	4C	0C	8B
0E	m	6D	4D	0D	8C
0F	n	6E	4E	0E	8D
10	o	6F	4F	0F	8E
11	p	70	50	10	8F
12	q	71	51	11	90
13	r	72	52	12	91
14	s	73	53	A3	92
15	t	74	54	14	93
16	u	75	55	15	94
17	v	76	56	16	95
18	w	77	57	17	96

NATIONAL KEYBOARDS

19	x	78	58	18	97
1A	y	79	59	19	98
1B	z	7A	5A	1A	99
1C	0	30	3D	E0	EC
1D	1	31	21	E1	ED
1E	2	32	22	E2	EE
1F	3	33	23	E3	EF
20	4	34	24	E4	F0
21	5	35	25	E5	F1
22	6	36	26	E6	F2
23	7	37	2F	E7	F3
24	8	38	28	E8	F4
25	9	39	29	E9	F5
26	/	2F	3F	EA	F6
27	+	2B	2A	EB	F7
28	~	60	40	00	13
29	-@	7B	5B	FB	1C
2A	~	7E	5E	1E	FC
2B	~	7C	5C	1F	FD
2C	~	7D	5D	1D	9F
2D	,	2C	3B	FE	F9
2E	.	2E	3A	FF	FA
2F	-	2D	5F	A4	A5
C0	SPACE	20	20	20	20
C1	↙	A7	A7	A7	A7
C2	S1	A8	A8	A8	A8
C3	S2	A9	A9	A9	A9

Numeric Section

RAW KEY CODE	KEYTOP	ALONE	with SHIFT	with CTRL	with COMMAND
C4	.	2E	2E	B0	2E
C5	0	30	30	B1	30
C6	00	A6	A6	B2	A6
C7	1	31	1C	B3	31
C8	2	32	9A	B4	32
C9	3	33	1D	B5	33
CA	4	34	9B	B6	34
CB	5	35	9C	1B	35
CC	6	36	9D	B8	36
CD	7	37	1E	B9	37
CE	8	38	9E	BA	38
CF	9	39	1F	BB	39
D0	+	2B	2B	BC	2B
D1	-	2D	2D	BD	2D
D2	*	2A	2A	BE	2A
D3	/	2F	2F	BF	2F

Notes:

Codes A0 through AF are special cases that are trapped by the keyboard driver. These codes are not placed in the buffer. The special cases are defined as follows:

- A0 - logical reset
- A1 - (reserved)
- A2 - break facility (clears buffer, then puts 03 in first location)
- A3 - halt display
- A4 - cursor lock
- A5 - shift lock
- A6 - two zeros
- A7 - End of line (CR in keyboard buffer, '0' in LTERM buffer)
- A8 - End of line (CR in keyboard buffer, '1' in LTERM buffer)
- A9 - End of line (CR in keyboard buffer, '2' in LTERM buffer)
- AA - End of line (CR in keyboard buffer, '0' in LTERM buffer - for DATEV keyboard)
- AB - End of line (CR in keyboard buffer, '1' in LTERM buffer - for DATEV keyboard)
- AC - End of line (CR in keyboard buffer, '2' in LTERM buffer - for DATEV keyboard)
- AD - (reserved)
- AE - (reserved)
- AF - no operation

C. HARD DISK AND DISKETTE CHARACTERISTICS

ABOUT THIS APPENDIX

This appendix provides a summary of the characteristics of the hard disk and the three types of diskette that can be used on the M20.

CONTENTS

<u>THE HARD DISK UNIT</u>	C-1
CHARACTERISTICS	C-1
<u>DISKETTES</u>	C-3
CHARACTERISTICS	C-4

THE HARD DISK UNIT

The hard disk is a Winchester disk unit incorporating three disk platters providing six recording surfaces. The read/write heads fly four microns above the disk surface allowing data to be stored at a very high density. To ensure reliability with this level of precision the disk is installed in a sealed casing and cannot be physically accessed by the user.

Although only 5 1/4in. in diameter the use of Winchester technology and high quality engineering allows the hard disk unit to store over 11 Mbytes of data (unformatted capacity) and transfer data at speeds up to 5 Mbits per second.

The hard disk unit is a high precision instrument and care must be taken whenever the M20 is moved to ensure that the unit is not damaged. The M20 should never be moved while the hard disk is being accessed.

CHARACTERISTICS

The main features of the hard disk unit are listed below.

Disk diameter 5 1/4in.

Unformatted capacity 11.25 Mbytes

Formatted capacity 8.85 Mbytes

Number of read/write heads 6 (one per surface)

Track density 254 tracks per inch (10 tracks per mm)

Tracks per surface 180

Number of cylinders 180

Access times:

- track to track 1.1 ms
- average 66 ms
- maximum 198 ms

Rotation speed 3,600 revs per minute (+ or - 1%)

Average latency time 8.33 ms

Transfer rate 5 Mbits per second

Formatted Capacity

256 bytes per block

32 blocks per track

8.192 Kbytes per track

1.475 Mbytes per surface

DISKETTES

Each 320 Kbyte and 640 Kbyte diskette drive has two read/write heads, and can read from, and write to, both surfaces of the diskette. 160 Kbyte diskette drives have one read/write head to read from and write to single-sided diskettes.

The diskettes that the drives use are standard 5 1/4in. magnetic disks, each one containing 35 tracks per surface.

Recording can be carried out on double-density diskettes, either single-sided or double-sided, thereby giving capacities of 160 Kbytes and 320 Kbytes, respectively; or on double-sided, quadruple-density diskettes thereby offering a capacity of 640 Kbytes.

A 640 Kbyte drive can perform both read and write operations on a 640 Kbyte diskette, but can only read from 320 or 160 Kbyte diskettes. A 320 Kbyte drive can perform both read and write operations on either 320 or 160 Kbyte diskettes, but can neither read from nor write to 640 Kbyte diskettes. Moreover, a 160 Kbyte can perform read and write operations only on 160 Kbyte diskettes.

CHARACTERISTICS

The main features of the diskettes are listed below.

	Diskette Type (Capacity)		
	640 Kbytes	320 Kbytes	160 Kbytes
diskette diameter	5 1/4 in.	5 1/4 in.	5 1/4 in.
average access time	303 ms	303 ms	303 ms
transfer rate	250 Kbits per second	250 Kbits per second	250 Kbits per second
tracks per surface	80	40 (35 used)	40 (35 used)
blocks per track	16	16	16
bytes per block	256 (except track 0 side 0 which is 128)	256 (except track 0 side 0 which is 128)	256 (except track 0 which is 128)

Note: Side 0 track 0 is not used by the system. It is written with standard Olivetti control data when the disk is formatted.

D. DIAGNOSTIC/BOOTSTRAP ERROR MESSAGES

ABOUT THIS APPENDIX

This appendix provides an explanation of the error messages that might occur during the diagnostic/bootstrap process.

CONTENTS

<u>DIAGNOSTIC ERROR MESSAGES</u>	D-1
<u>BOOTSTRAP ERROR MESSAGES</u>	D-3

DIAGNOSTIC ERROR MESSAGES

Diagnostic error symbols and messages appear in the extream top left corner of the screen.

DISPLAYED SYMBOL	MESSAGE TYPE	DESCRIPTION
TRIANGLE	CPU diagnostic	A malfunction has been detected during the CPU diagnostic test. Correct operation of all registers, addressing modes, and instruction classes is verified
SQUARE	ROM diagnostic	<p>A failure has been detected during the ROM diagnostic test. An error message is sent to the parallel printer port with the following format</p> <p style="padding-left: 40px;">E(xn)</p> <p style="padding-left: 40px;">where</p> <p style="padding-left: 40px;">x= H or L - indicating high or low byte</p> <p style="padding-left: 40px;">n= ROM number - a value in the range 0 to 4</p>
DIAMOND	RAM diagnostic	<p>A malfunction has been detected during the RAM diagnostic test. An error message is sent to the parallel printer port with the following format:</p> <p style="padding-left: 40px;">EM(c) (bb) (ssss) (www)</p> <p style="padding-left: 40px;">where</p> <p style="padding-left: 80px;">c = RAM configuration number</p> <p style="padding-left: 80px;">bb = bank failure number</p> <p style="padding-left: 80px;">ssss = the expected data</p> <p style="padding-left: 80px;">www = the actual data</p> <p>The RAM configuration number is a single digit octal number which specifies the hardware configuration of the particular M20 system</p>

		The bank number is a two-digit hexadecimal number that specifies the particular 16 Kbyte bank of memory (RAM or ROM) located either on a RAM expansion board or on the M20 mother board
FOUR BAR	Trap and Interrupt diagnostic	An illegal trap or interrupt has been detected.
E C0	IC error	Error detected in the 8255 parallel Centronics-like port IC
E C1	IC error	Error detected in the 6845 video controller IC
E C2	IC error	Error detected in the 1797 disk drive controller IC
E C3	IC error	Error detected in the 8253 real-time IC
E C4	IC error	Error detected in the 8251 (keyboard) USART IC
E C5	IC error	Error detected in the 8251 RS-232-C USART IC
E C6	IC error	Error detected in the 8259 interrupt controller IC

DIAGNOSTIC/BOOTSTRAP ERROR MESSAGES

E K0	Keyboard error	No response from the keyboard
E K1	Keyboard error	Self-test failure
E I0	Interrupt error	Non-vectored interrupt
E I1	Interrupt error	Vectored interrupt
E D0	Disk error	Error detected in diskette drive 0
E D1	Disk error	Error detected in diskette drive 1
E D10	Disk error	Error detected in hard disk drive 10

BOOTSTRAP ERROR MESSAGES

The messages listed in the following table can occur during the bootstrap procedure.

MESSAGE	MEANING
Insert system disk and type any key	Neither of the diskette drives is READY. Insert the system diskette and strike any key

Invalid File Error
(xx) on drive (n)

where (xx) is

- 00 - invalid extent count for the file descriptor block
- 01 - invalid file type
- 02 - invalid block count
- 03 - end of file error
- 04 - parameter out of range for the diskette drive

Disk Error: (xx)

A diskette or hard disk error has occurred where xx is a two-digit hexadecimal number indicating the diskette or hard disk driver return code. This value must be decoded into an 8-bit number, each bit of which represents an error condition which for a diskette drive is as follows:

- bit 7 - drive not ready
 - (most significant bit)
- bit 6 - write-protect error
- bit 5 - write fault error
- bit 4 - record not found error
- bit 3 - data transfer error
- bit 2 - seek error
- bit 1 - after restore, not track zero
- bit 0 - illegal parameters
 - (least significant bit)

or for the hard disk drive:

- bit 7 - bad block error
 - or
 - drive not ready error
 - (most significant bit)
- bit 6 - cyclic redundancy check error on data
- bit 5 - cyclic redundancy check error on block identifier
- bit 4 - record (block) not found error
- bit 3 - data transfer error
- bit 2 - abort error
- bit 1 - after restore, not track zero
- bit 0 - illegal parameters
 - (least significant bit)

Any number of these bits can be set signifying more than one error.

E. PCOS AND BASIC ERROR MESSAGES

INTRODUCTION

Errors returned from the BASIC Interpreter are not displayed with their error number (only the description is displayed). Errors returned from PCOS are always displayed with the error number; the descriptive label will only be displayed if the EPRINT command is present in memory (via execution, PLOAD or PSAVE), for example:

```
v1 %n novol /CR/
```

```
ERROR 69 --- volume name not found
```

The PCOS and BASIC error codes are listed in the table below. For each code the displayed message is given along with a descriptive comment. The table also indicates which codes apply to BASIC and which to PCOS.

PCOS AND BASIC ERRORS

ER- ROR CODE	MESSAGE	PCOS OR BASIC	COMMENT
1	NEXT without FOR	BASIC	A NEXT statement has been encountered without a matching FOR
2	syntax error	BASIC	A line has been encountered which includes an incorrect sequence of characters (misspelled keyword, incorrect punctuation etc.)
3	RETURN without GOSUB	BASIC	A RETURN has been encountered for which there is no previous unmatched GOSUB statement
4	out of data	BASIC	A READ statement has been executed when there are no DATA statements with unread data remaining in the program

5	illegal function call	BASIC	<p>A parameter that is out of range has been passed to a numeric or a string function.</p> <p>Such an error may occur when:</p> <ol style="list-style-type: none"> An array subscript is either negative or too big A log function is assigned a negative or a null argument The SQR function is assigned a negative value A negative number has an exponent which is not an integer An incorrect argument has been made in one of the following functions: MID\$, LEFT\$, RIGHT\$, TAB, SPC, STRING\$, SPACE\$, INSTR, or ON...GOTO
6	overflow	BASIC	<p>The result of a calculation is too large to be represented in BASIC's number format.</p> <p>Note: With underflow, the result is taken as zero, and execution continues without indication of an error</p>
7	out of memory	PCOS OR BASIC	<p>A program is too big; or has too many loops, GOSUBS, variables; or has expressions too complicated to evaluate, or a command or Assembler routine has been called that cannot be accommodated in the current memory available</p>
8	undefined line number	BASIC	<p>A line reference is to a non-existent line from a GOTO, GOSUB, IF..THEN..ELSE or DELETE</p>

9	out of range	BASIC	An array element has been referred to either with a subscript that is outside the dimensions of the array or with the wrong number of subscripts
10	duplicate definition	BASIC	Two DIM statements have been given for the same array, or a DIM statement has been applied to an array after the default dimension of 10 was previously established for that array
11	division by zero	BASIC	A division by zero has been encountered or the value zero has been raised to a negative power. In the former case the result is machine infinity (with the appropriate sign) and in the latter case the result is positive machine infinity
12	illegal direct	BASIC	A statement which is invalid in immediate mode has been entered as an immediate command
13	type mismatch	PCOS OR BASIC	A string value has been entered when a numeric value is required or vice versa
14	out of string space	BASIC	String variables have caused BASIC to exceed the amount of free user memory remaining. (BASIC will allocate space dynamically until it runs out of memory)
15	string too long	BASIC	An attempt has been made to create a string of more than 255 characters
16	string formula too complex	BASIC	A string expression is too long or too complex to be processed. It should be broken into smaller expressions

17	can't continue	BASIC	An attempt has been made to continue a program that is non-continuable: one that was halted due to an error, was modified during a break in execution, or does not exist in user memory
18	undefined function	BASIC	A function has been called that has not been previously defined
19	no RESUME	BASIC	An error-trapping routine has been entered that contains no RESUME statement
20	RESUME without error	BASIC	A RESUME statement has been encountered before an error-trapping routine is entered
21	unprintable error	BASIC	An error message is not printable. That is, it corresponds to an error with an undefined error code
22	missing operand	BASIC	An expression contains an operator but no following operand
23	buffer overflow	BASIC	An attempt has been made to enter a line with more than 255 characters
26	FOR without NEXT	BASIC	A FOR has been encountered without a matching NEXT
29	WHILE without WEND	BASIC	A WHILE has been encountered without a matching WEND
30	WEND without WHILE	BASIC	A WEND has been encountered without a matching WHILE
31	IEEE: invalid talker/ listener address	BASIC	An invalid talker/listener address has been used
32	IEEE: talker = listener address	BASIC	An attempt has been made to talk to a talker, or listen to a listener
33	IEEE: unprintable error	BASIC	An IEEE error message is not printable. That is, it corresponds to an error with an undefined error code

34	IEEE: board not present	BASIC	An attempt has been made to use IEEE on a machine which does not have the optional IEEE interface
35	window not open	PCOS OR BASIC	An attempt has been made to use a window which is not at present open
36	unable to create window	PCOS OR BASIC	The window to be created is too big or too small for its mode (graphics or text). (This error can be returned while executing an Assembly Language program)
37	invalid action verb	BASIC	An action verb has been incorrectly spelled or used
38	parameter out of range	BASIC	One or more parameters have exceeded the limits set for their range
39	too many dimensions	BASIC	An attempt has been made to use an array of more than one dimension, in graphics mode
50	field overflow	BASIC	A FIELD statement has attempted to allocate more bytes than were specified for the record length of a random file
51	internal error	BASIC	An internal malfunction has occurred. Report the conditions under which the error occurred to your support organisation
52	bad file number	BASIC	A statement or command refers to: - a file that does not have a file number within the range specified at initialisation - a file that is not open
53	file not found	PCOS OR BASIC	A LOAD, KILL or OPEN statement or a PCOS command refers to a file that does not exist on an enabled volume

54	bad file mode	PCOS OR BASIC	An attempt has been made to use random file operations (GET or PUT) with a sequential file; or to use the sequential operation LOAD with a random file; or to use an illegal file mode with OPEN, that is, not A, I, O, or R
55	file already open	PCOS OR BASIC	A sequential OPEN, O has been issued for a file that is already open, or a KILL has been applied to a file that is open
57	disk I/O error	PCOS OR BASIC	An input/output error has occurred during a disk I/O operation. It is a termination error from which PCOS/BASIC cannot recover - apply a RESET
58	file already exists	PCOS OR BASIC	The file name specified in a NAME statement, or the file name you are attempting to assign in a PCOS command is identical to a file name already in use on the volume
59	disk type mismatch	PCOS	A volume has been specified that is an invalid size for the operation
60	disk not initialized	PCOS	An attempt has been made to access a diskette or hard disk that has not been initialised
61	disk filled	PCOS OR BASIC	All disk storage space available is in use
62	end of file	PCOS OR BASIC	An INPUT statement has been executed: after all the data has been assigned, or for an empty (null) file. Note: the EOF function can be used to detect end of file
63	invalid record number	PCOS OR BASIC	The record number used with a GET or PUT statement exceeds range. That is, it is 0 or greater than 32767

PCOS AND BASIC ERROR MESSAGES

64	invalid file name	PCOS OR BASIC	An invalid form of filename has been used with KILL, LOAD, OPEN, SAVE or a PCOS command. For example - too long - includes illegal characters such as space or hyphen
66	direct statement in file	BASIC	An immediate statement has been encountered when loading an ASCII format file. The LOAD operation is terminated
67	too many files	BASIC	An attempt has been made to create a new file (using SAVE or OPEN) when the present directory is already full
68	internal error	PCOS OR BASIC	An internal malfunction has occurred. Report the conditions under which the error occurred to your support organization
69	volume name not found	PCOS OR BASIC	The volume name referred to does not match any diskette or hard disk currently present
70	rename error	PCOS OR BASIC	An attempt has been made to rename a volume with an invalid name
71	invalid volume number	PCOS OR BASIC	The specified volume number is invalid
72	volume not enabled	PCOS OR BASIC	The specified volume has not been enabled
73	invalid password	PCOS OR BASIC	The specified password does not match that of the file
74	illegal disk change	PCOS OR BASIC	The diskette has been changed since the last file was last used
75	write protected file	PCOS OR BASIC	An attempt has been made to write to a write-protected file

76	error in parameter	PCOS OR BASIC	One or more of the quoted parameters contains an unacceptable value
77	invalid number of parameters	PCOS OR BASIC	More than the required number of parameters have been specified
78	file not open	PCOS OR BASIC	An attempt has been made to access a file that is not open
79	printer error	PCOS OR BASIC	A printer error has been returned indicating that some operator response is required, such as out of ribbon
80	copy protected file	PCOS	An attempt has been made to copy a file that is copy-protected
81	paper empty	PCOS OR BASIC	The printer has run out of paper
82	printer fault	PCOS OR BASIC	The printer has a hardware fault
92	command not found	PCOS	An invalid keyword has been entered
99	bad load file	PCOS	The program file specified is not compatible with the PCOS version being used
101	error in time or date	PCOS	An invalid time or date has been entered
108	call user error	PCOS	An error has been encountered in a call to an Assembly Language routine or a PCOS command
110	time out	PCOS	A time-out error has occurred
111	invalid device	PCOS	The specified device name does not exist

F. GET/PUT CONVERSION
— PCOS 1-3 TO PCOS 2-0/3-0

ABOUT THIS APPENDIX

This appendix provides details on how to convert BASIC files created under PCOS 1.3 to a format compatible with PCOS 2.0 or 3.0.

CONTENTS

<u>GET/PUT CONVERSION</u>	F-1
GETCONV.BAS	F-1

GET/PUT CONVERSION

GETCONV.BAS

Converts BASIC files containing video images created using GET/PUT statements under release 1-3 of PCOS, into a format compatible with releases 2-0 and 3-0 of PCOS.

The procedure is outlined below:

STEP	OPERATION
1	Boot PCOS 3.0
2	Execute the GETCONV.BAS utility by entering ge /CR/ OR by entering BASIC and entering run "getconv.bas"
3	Following the subsequent prompt, enter the identifier of the file to be converted
4	Following the subsequent prompt, enter the identifier of the output file
5	Following the subsequent prompt, enter "s /CR/" if the file to be converted is sequential, or "r /CR/" if it is a random file
6	Information about the conversion is then displayed. The process is complete when the "Conversion Complete" message is displayed

G. GLOSSARY OF TERMS

ABOUT THIS APPENDIX

This appendix defines the terms used in this manual that vary from general EDP terminology.

CONTENTS

GLOSSARY OF TERMS

G-1

GLOSSARY OF TERMS

The following table defines the non-standard terminology in this manual and also provides a page reference.

TERM	MEANING	REFERENCE
bootable file	a file of a specific format that the bootstrap loader can load into memory to initialise the system	5-6
device re-routing	a facility that enables input to be accepted from devices or files other than the keyboard, and output to be directed to devices and files other than the video	7-1/8
diskette	a single or double-sided 5 1/4in. floppy disk	C-3/4
drive number	an integer referring to the diskette drive or hard disk drive in question. It may be 10 - hard disk 0 - first diskette drive 1 - second diskette drive	4-8
environment	an operational environment in which the M20 responds to keyboard input in a specific way. Three distinct environments are mentioned in this manual PCOS BASIC Video File Editor	5-1/4

global command	a PCOS command that allows the user to change global parameters	6-7/20
global parameter	a parameter that defines a feature of the system environment	6-7/20
hard disk	a 5 1/4in. Winchester disk unit	C-1/2
initialisation file	<p>a file written in either Assembly Language or BASIC that is automatically loaded and executed on system initialisation. It may have one of the following names:</p> <ul style="list-style-type: none"> - INIT.CMD - INIT.SAV - INIT.BAS 	5-7/8
logical reset	a reset of all global parameters (except those controlled by the real-time clock) and re-initialisation of the system (without performing diagnostic tests). It is caused by pressing /CTRL/ /RESET/, simultaneously	2-6
nil parameter	a parameter to a command where the parameter in question is not specified in the command line. The parameter therefore assumes either a default value, or the last specified value (in the case of global commands)	4-4
non-standard initialisation	a system initialisation where /L/, /D/, /F/, /B/, or /S/ is pressed during power-up diagnostics, or following a PRUN command	5-8/9

GLOSSARY OF TERMS

PCOS nucleus	that part of the operating system that is loaded into memory on initialising the system, and remains there until the working session is terminated	6-1/2
permanent memory area	that part of memory occupied by the PCOS nucleus, and those commands, assembler programs, programmed key definitions and user defined fonts made permanent by a PSAVE command	6-1/6
physical reset	a system re-initialisation caused by pressing the physical reset button. The subsequent initialisation includes diagnostic tests and a reset of all global parameters (including those controlled by the real-time clock	2-7
programmed key	a key that has either had its associated ASCII code changed by means of a CKEY command, or had a string assigned to it by means of the PKEY command	11-6/10
raw key code	the immediate code generated by a key (or key combination) corresponding to the physical position of the key on the keyboard, independent of system tables	11-5/6
semi-permanent memory area	that part of memory occupied by loaded commands and assembler programs, PKEYed strings and user-defined fonts that will be released on termination of the current working session	6-1/6

standard initial- isation	initialisation following switch-on, physical reset, or logical reset; not having /L/, /D/, /F/, /B/ or /S/ pressed during power-up diagnostics	5-6/8
standard PCOS	the PCOS configuration supplied by Olivetti on the system diskette	6-1/2
text file	an ASCII file whose records are sep- arated either by CR/LF, or by record separator (RS) characters	12-1
transient command	a command that is not loaded into memory at initialisation. This includes commands that are loaded and purged (those with CMD extens- ion), and those that are loaded, but remain in memory after execution (those with SAV extension)	3-4
volume	the entire contents of a diskette or hard disk	4-6/8
wild card character	a special symbol used to represent any single character (?), or any string of characters (*)	4-10
working session	the time between booting PCOS and the next boot of PCOS or switch-off	5-12

H. COMMAND INDEX

ABOUT THIS APPENDIX

This appendix provides an alphabetic index of PCOS commands.

CONTENTS

COMMAND KEYWORD INDEX

H-1

COMMAND KEYWORD INDEX

COMMAND KEYWORD	REFERENCE	
	PART I	PART II
BASIC.CMD	3-7, 7-7/8 8	13-1
BKEYBOARD.BAS		13-2
BVOLUME.SAV		13-3
CI.SAV	3-9	13-8
CKEY.CMD	11-6/7	13-9
COMMANDS.BAS		13-14
DCONFIG.CMD		13-15
EDIT.CMD	5-4, 11-10/13, 12-1/19	13-18
EPRINT.SAV	6-3/4	13-18
ERROR.BAS		13-20
FCOPY.CMD	9-9, 10-2/4, 10-6	13-20
FDEPASS.CMD	8-3, 10-8/9	13-26
FFREE.CMD	10-10/13	13-28
FKILL.CMD	10-14/15	13-30
FLIST.CMD	10-6/8	13-31
FMOVE.CMD	10-5/6	13-33
FNEW.CMD	8-2, 10-1	13-35
FPASS.CMD	8-2/3, 10-8/9	13-37
FRENAME.CMD	10-15/16	13-39
FUNPROT.CMD	8-4, 10-9/10	13-40
FWPROT.CMD	8-4, 10-9/10	13-42

HELP.BAS		13-44
IEEE.SAV	3-9	13-44
LABEL.CMD	11-1/3	13-45
LSCREEN.CMD	11-3/4	13-50
LTERM		13-52
PKEY.CMD	6-5, 6-6, 11-9	13-53
PLOAD	6-1/4	13-57
PRUN.CMD	5-9, 6-22	13-59
PSAVE.CMD	5-9, 6-21	13-61
PUNLOAD	6-1/2	13-63
RFONT.CMD	11-10	13-65
RKILL.CMD	10-15	13-67
RS232.SAV	3-9	13-69
SBASIC.CMD	6-14/15	13-69
SCOMM.CMD	3-9, 6-15	13-72
SDEVICE.CMD	6-11/13	13-73
SFORM.CMD	6-15/18	13-75
SLANG.CMD	6-19/20	13-78
SPRINT.CMD	11-3/4	13-80
SSYS.CMD	6-7/11	13-82
VALPHA.CMD	9-11	13-86
VCOPY.CMD	9-6/7	13-88
VDEPASS.CMD	8-2, 9-10	13-90
VFORMAT.CMD	9-1/3, 9-10	13-91
VLIST.CMD	9-4/5	13-94
VMOVE.SAV	9-8/9	13-96

COMMAND INDEX

VNEW.CMD	9-3/4, 9-10	13-98
VPASS.CMD	8-1, 9-10	13-100
VQUICK.CMD	9-5/6	13-102
VRENAME.CMD	9-11	13-104
VVERIFY.CMD		13-106
WFONT.CMD	11-14/15	13-109

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 notwithstanding, 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.

The enclosed programs are protected by Copyright and may be used only by the Customer. Copying for use by third parties without the express written consent of Olivetti is prohibited.

GU Code 3985280 D (0)
Printed in Italy

olivetti

GU Code 3985280 D (0)
Printed in Italy

olivetti