

**EPSON**

**PX-8**

**USER'S MANUAL**

FEDERAL COMMUNICATIONS COMMISSION  
RADIO FREQUENCY INTERFERENCE  
STATEMENT

---

“This equipment generates and uses radio frequency energy and if not installed and used properly, that is, in strict accordance with the manufacturer’s instructions, may cause interference to radio and television reception. It has been type tested and found to comply with the limits for a Class B computing device in accordance with the specifications in Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference in a residential installation. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- ..... reorient the receiving antenna
- ..... relocate the computer with respect to the receiver
- ..... move the computer away from the receiver
- ..... plug the computer into a different outlet so that computer and receiver are on different branch circuits.

If necessary, the user should consult the dealer or an experienced radio/television technician for additional suggestions.

The user may find the following booklet prepared by the Federal Communications Commission helpful: “How to Identify and Resolve Radio-TV Interference Problems.”

This booklet is available from the US Government Printing Office, Washington, D.C., 20402, Stock No. 004-000-00345-4.”

Also, only peripherals (computer input/output devices, terminals, printers, etc.) certified to comply with the Class B limits may be attached to this computer. Operation with non-certified peripherals is likely to result in interference to radio and TV reception.

**PX-8**  
**USER'S MANUAL**



## Trademark Acknowledgments

CP/M® is a registered trademark of Digital Research™.  
BASIC (Copyright 1977 – 1983 by Microsoft and Epson) is upward compatible with the BASIC-80 specifications of Microsoft, Inc.  
MICROCASSETTE™ is a trademark of OLYMPUS OPTICAL CO., LTD.  
Portable WordStar™ is a trademark of MicroPro International.

### NOTICE

- All rights reserved. Reproduction of any part of this manual in any form whatsoever without EPSON's express written permission is forbidden.
- The contents of this manual are subject to change without notice.
- All efforts have been made to ensure the accuracy of the contents of this manual. However, should any errors be detected, EPSON would greatly appreciate being informed of them.
- The above notwithstanding, EPSON can assume no responsibility for any errors in this manual or their consequences.

© Copyright 1983 by EPSON CORPORATION.  
Nagano, Japan

## TABLE OF CONTENTS

<b>Chapter 1 OVERVIEW</b> .....	1-1
1.1 Unpacking Notes .....	1-1
1.2 Hardware Configuration .....	1-2
1.3 Software Configuration .....	1-10
<b>Chapter 2 STARTING UP AND OPERATING THE PX-8</b> .....	2-1
2.1 Starting Up .....	2-1
2.2 Operating the Computer .....	2-11
<b>Chapter 3 OPERATING THE COMPUTER UNDER CP/M</b> .....	3-1
3.1 What is CP/M? .....	3-1
3.2 Files and File Names .....	3-3
3.3 Starting to Use CP/M .....	3-6
3.4 Using the Keyboard Under CP/M .....	3-10
3.5 Printing When on the CP/M Command Line .....	3-12
3.6 The Concept of Logical and Physical Devices .....	3-13
3.7 The CP/M Built-In Commands .....	3-16
3.8 Using Utilities and Application Program with CP/M .....	3-32
3.9 Communications .....	3-75
3.10 Other CP/M Transient Programs .....	3-91
3.11 Finding Out More About CP/M .....	3-93
<b>Chapter 4 INPUT AND OUTPUT DEVICES AND OPTIONAL ADDITIONS TO THE PX-8</b> .....	4-1
4.1 Storage of Files and Data .....	4-1
4.2 RS-232C Interface .....	4-22
4.3 Serial Interface .....	4-29
4.4 Using Printers with the PX-8 .....	4-31
4.5 Speaker .....	4-34
4.6 Analog Interface .....	4-36
4.7 Bar Code Reader Interface .....	4-37
4.8 The Universal Unit .....	4-38
4.9 System Bus Interface .....	4-39
4.10 Acoustic Coupler .....	4-42

<b>Chapter 5 THE SYSTEM INTERFACE</b> .....	5-1
5.1 The CP/M Configuration .....	5-1
5.2 The IOBYTE .....	5-2
5.3 The File Control Block (FCB).....	5-3
5.4 BDOS Function Calls .....	5-4
5.5 Zero Page Locations .....	5-8
5.6 The BIOS Interface.....	5-9
5.7 The Entry Address of the BIOS Subroutines.....	5-10

**Appendix A PX-8 CONSOLE ESCAPE SEQUENCES**

**Appendix B DRIVE NAME ASSIGNMENTS**

**Appendix C THE BATTERIES**

**Appendix D KEYBOARD LAYOUTS BY COUNTRY**

**Appendix E ASCII CODE TABLE AND INTERNATIONAL CHARACTER SETS**

**Appendix F MEMORY MAPS**

**Appendix G HARDWARE SPECIFICATIONS**

**Appendix H SOME EXAMPLE PROGRAMS**

**Appendix I CP/M ERRORS AND MESSAGES**

**Index**

Chapter 1 OVERVIEW

Chapter 2 STARTING UP AND OPERATING THE PX-8

Chapter 3 OPERATING THE COMPUTER UNDER CP/M

Chapter 4 INPUT AND OUTPUT DEVICES AND OPTIONAL ADDITIONS TO THE PX-8

Chapter 5 THE SYSTEM INTERFACE

Appendices

# *Chapter 1*

## **OVERVIEW**

The PX-8 is a portable computer which has more facilities than many desk top computers. This chapter will give you an idea of the general capabilities of the computer. The next chapter explain the features of the PX-8 so that they can be exploited to the full and you as a user can obtain maximum benefit from them in operating the computer. Some features are necessary for running the PX-8 normally and would be familiar to the user who has handled a microcomputer before, but others have been added to increase the power of the PX-8.

### **1.1 Unpacking Notes**

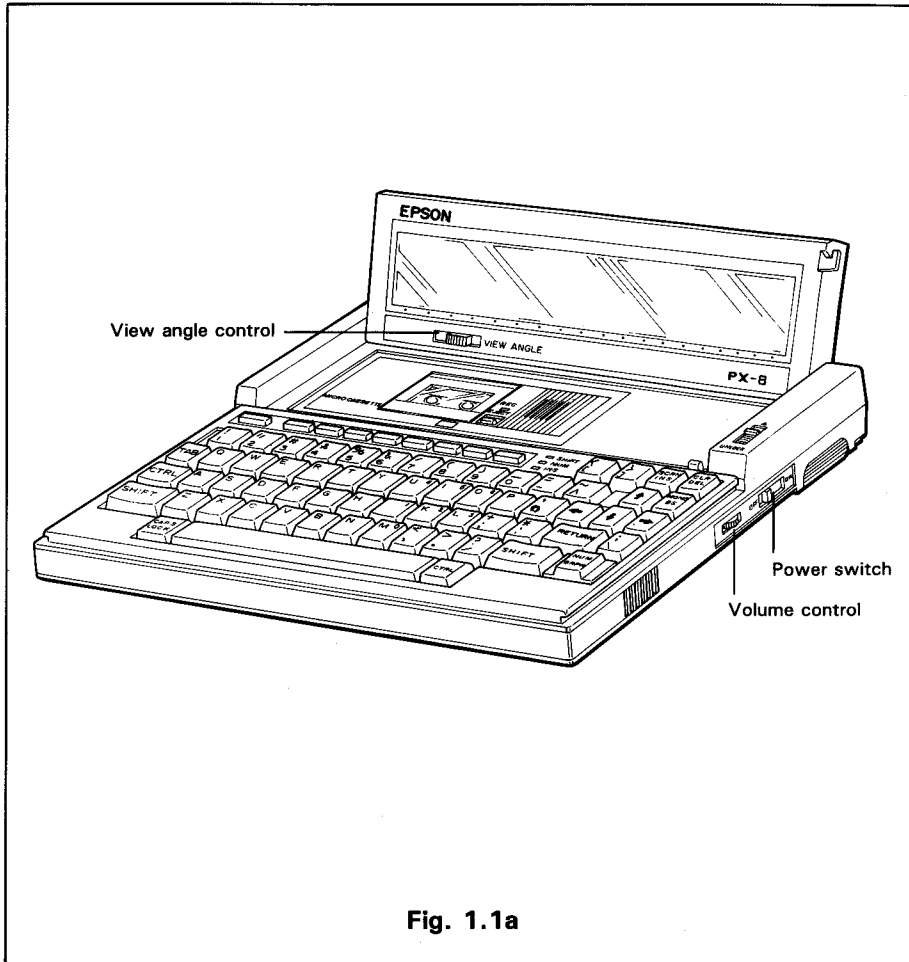
When you unpack your PX-8 package you should find the following items. Confirm that the following are contained in the package. Please keep the packing materials in case the PX-8 needs to be returned for any reason.

- PX-8
- AC adapter
- User's Manual (this manual)
- BASIC Reference Manual
- Command Summary
- End-User License Agreement
- Programmable Function-Key Label

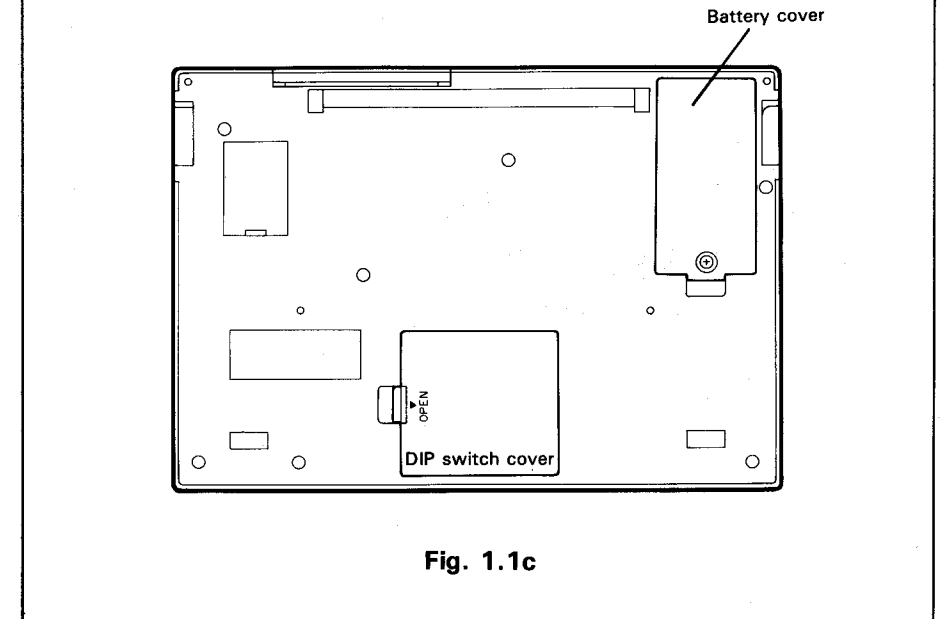
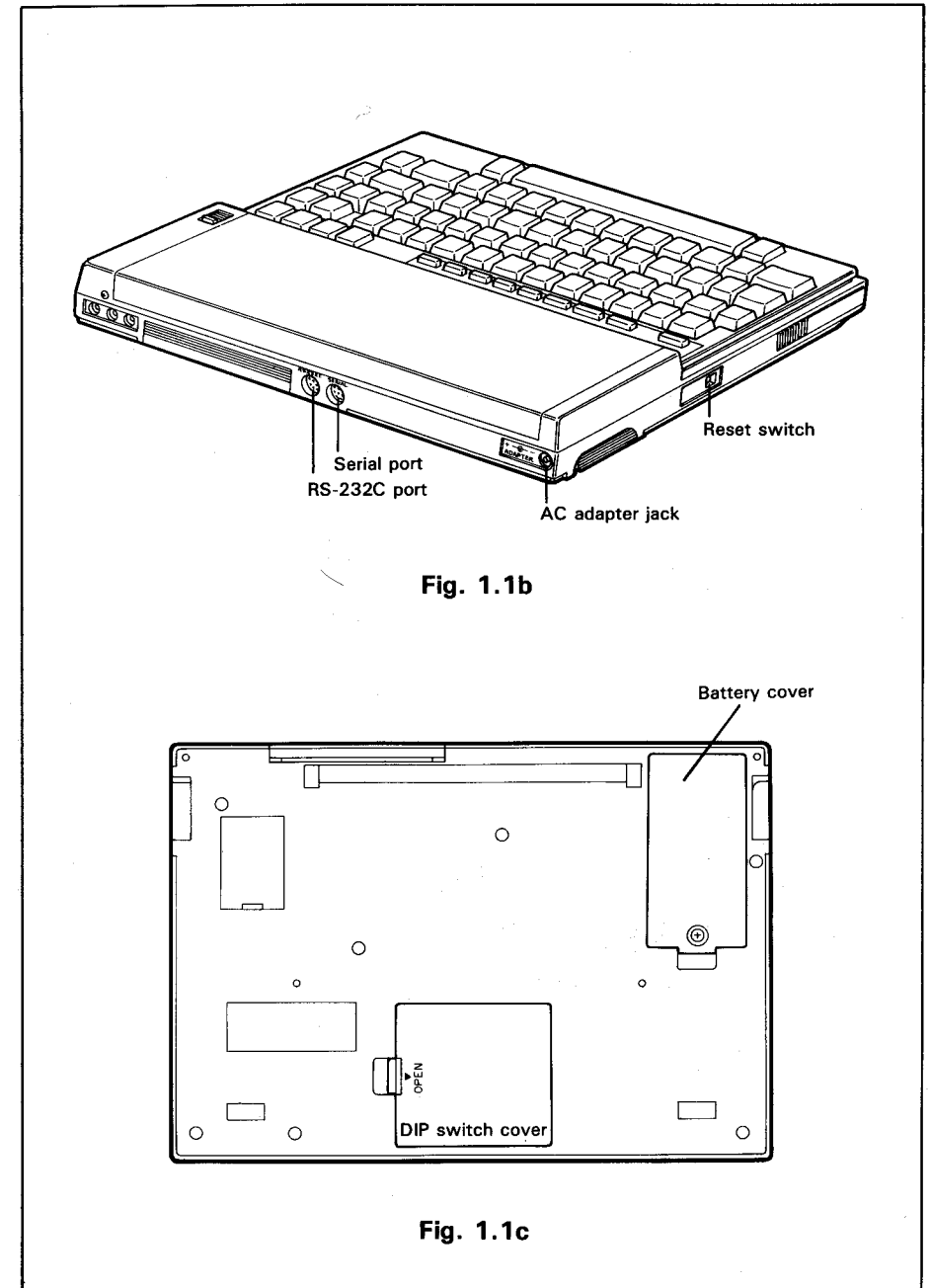
## 1.2 Hardware Configuration

This section describes the hardware components of the PX-8. All the hardware components required for basic operation are built into a single unit. These include the central processor units (CPUs), memory, keyboard, display, microcassette tape deck and interfaces for a printer and serial communication with other devices including disk drives.

Figures 1.1a to 1.1c show the appearance of the PX-8 from several viewpoints.



1-2



1-3

## 1.2.1 Configuration diagram

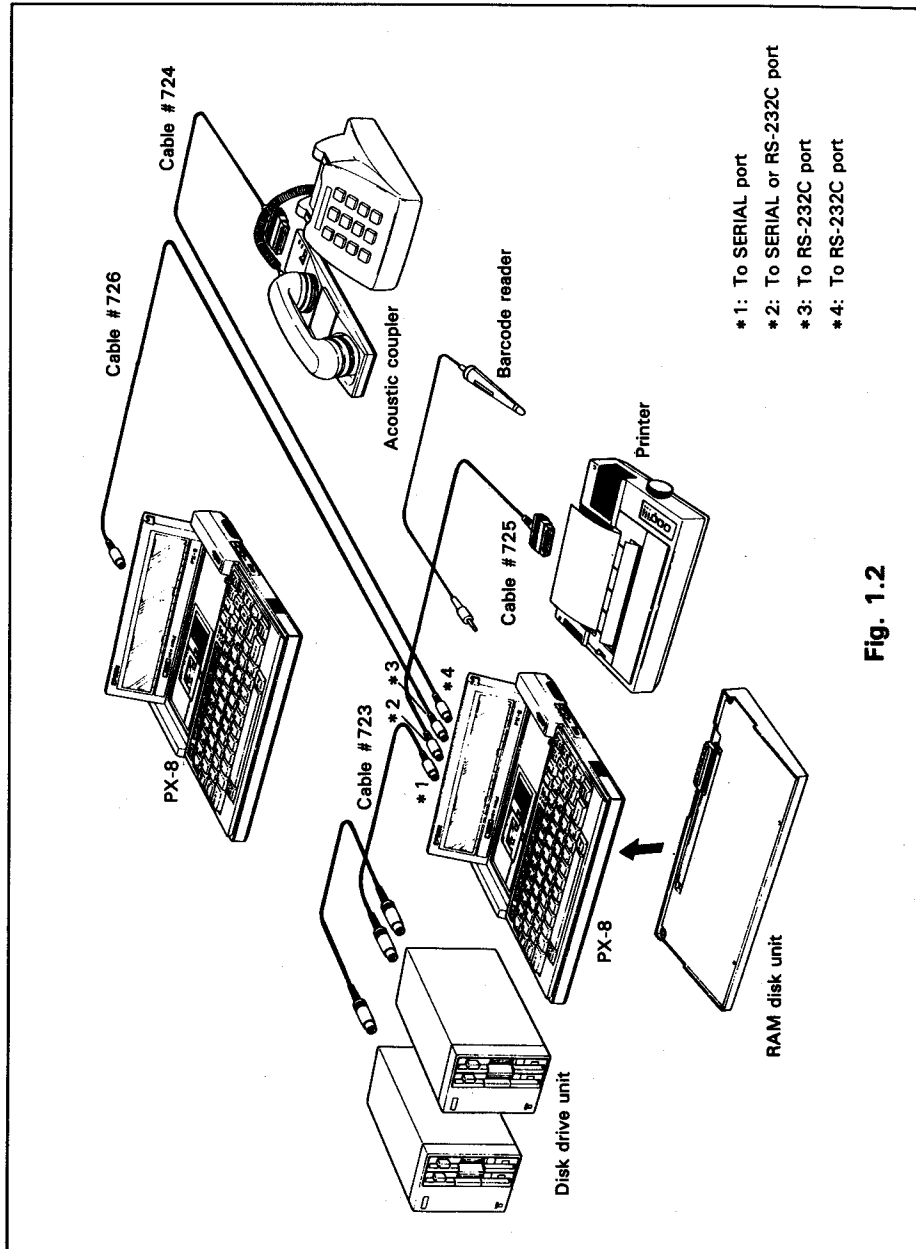


Fig. 1.2

## 1.2.2 The Carrying handle

The handle enables the PX-8 to be carried easily like a briefcase. It is possible to slide the handle up to the main body of the computer when not in use.

To extend the handle, place the index finger of each hand on the serrated portion of the handle, and pull it gently towards you, as shown in the diagram. To retract it, simply push the handle towards the computer using even pressure.

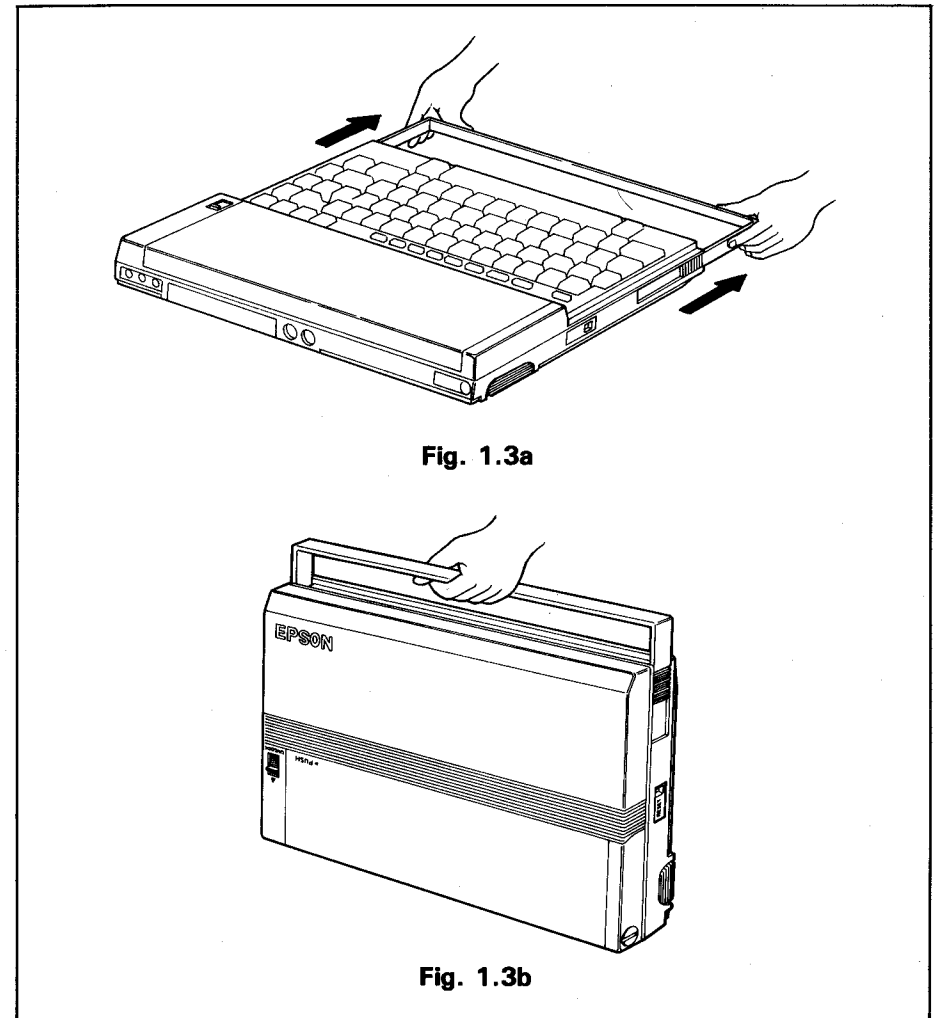


Fig. 1.3a

Fig. 1.3b

### 1.2.3 The keyboard

The keyboard is accessed by removing the protective cover.

To remove the cover, hold the PX-8 with the keyboard cover towards you and uppermost, preferably on a level working surface.

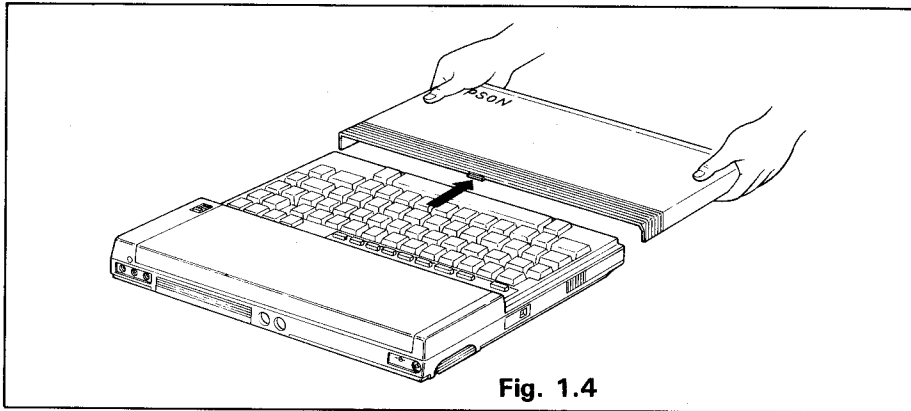


Fig. 1.4

Place the thumb of each hand on the top of the cover, with the other fingers on each side of the cover. Press gently with the index fingers to release the catch in the top centre (see the arrow in the diagram). Now pull the whole cover towards you using the both hands so that it slides off.

To replace the cover, place it centrally over the keyboard, so that the edges of the cover fit evenly on the guides. Push the cover home using the thumbs applying pressure evenly at both ends. Take care that the cover moves along the guides. Do not try to force it vertically down. If the cover is not in the guides, take it off and try again. Make sure the cover is fully up against the LCD screen and the catch has fitted firmly.

The layout of character keys on the PX-8 keyboard varies from country to country. The layouts available are as follows.

ASCII (USA)	Danish
French	Swedish
German	Norwegian
English	Spanish
Italian	

It is possible to change the display to that of any other country both by setting a DIP switch and under software control. In this manual, the ASCII keyboard is used as the standard for explanation.

The PX-8 is equipped with folding legs which makes it possible to change the keyboard angle. Opening these legs will make the keyboard easier to use when working with the PX-8 on a table or desk.

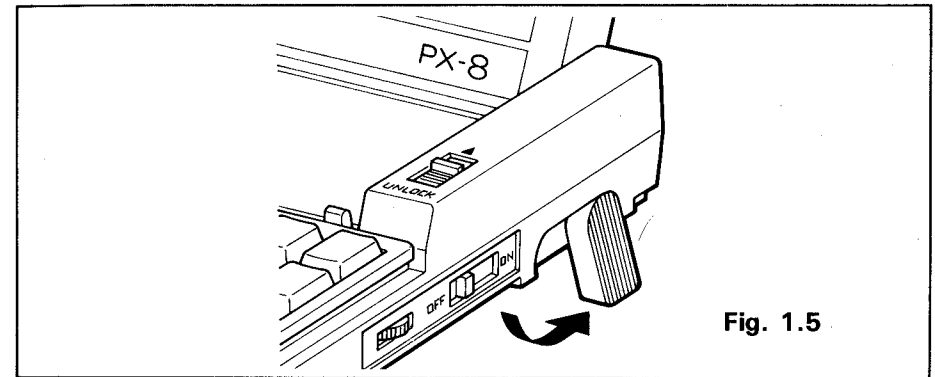


Fig. 1.5



### 1.2.4 Display

The PX-8 uses a large LCD display of 80 columns  $\times$  8 lines, on a hinge which can be folded down to protect the display when not in use.

Unlock the display by pressing the switch on its right toward the rear. When the display pops up slightly, open it by hand.

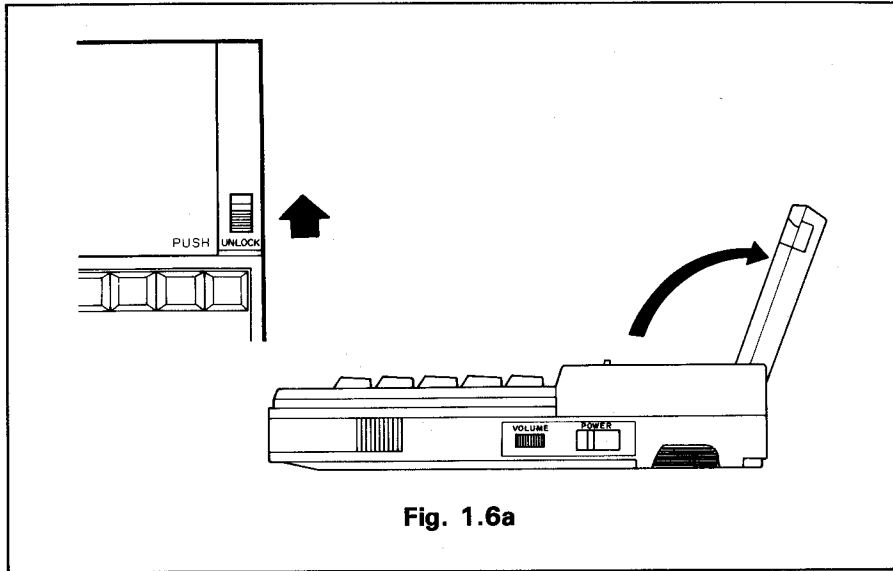


Fig. 1.6a

The view angle control on the front of the display makes it possible to obtain the optimum contrast.

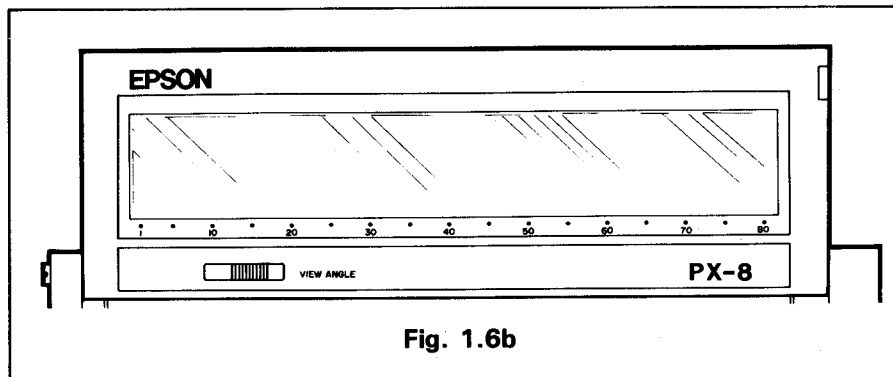


Fig. 1.6b

### 1.2.5 CPU and memory

The PX-8 has three central processor units. The main CPU is a Z80 compatible microprocessor. This controls the programs which are run on the PX-8. A slave CPU 6301 is used for display and Input/Output (I/O) control, i.e. communication with other devices such as printers, disk drives, the internal speaker, cassette drives etc. The sub CPU 7508 controls interfacing with the keyboard and the Analog/Digital (A/D) converter which allows signals from other devices to be read as voltages and converted into numbers the PX-8 can understand.

The main CPU controls 64K bytes of RAM and 32K bytes of ROM. The slave CPU has 6K bytes of RAM (used as video RAM) and 4K bytes of ROM.

All RAM is backed up by built-in batteries so that the contents can be retained when the power switch is turned off.

### 1.2.6 Microcassette tape deck

The PX-8 is equipped with a microcassette tape deck which can be used for saving data and program files sequentially. The microcassette tape deck is controlled manually or by software.

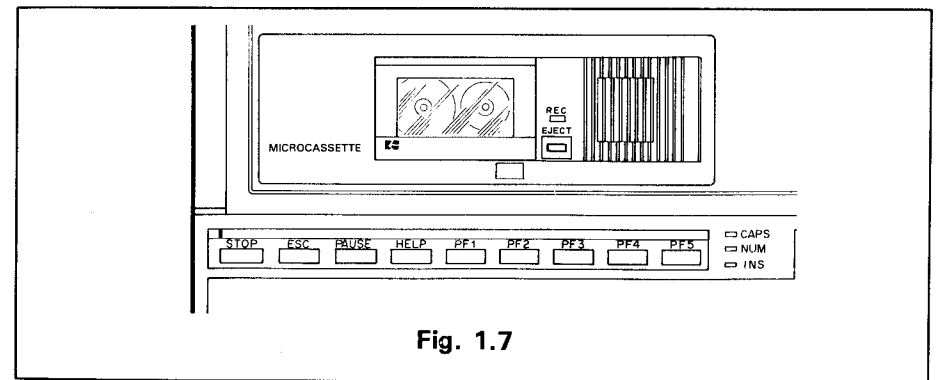


Fig. 1.7

The maximum number of files which can be stored in a microcassette is 12. This restriction is due to the size of the directory on the cassette tape.

## 1.3 Software Configuration

### 1.3.1 Operating systems

The operating system of the PX-8 is an extended version of CP/M which is one of the most popular operating systems for small business computers and personal computers. CP/M is an abbreviation for Control Program for Microprocessors. This operating system was developed by Digital Research and is a very popular operating system on microcomputers because programs developed under CP/M can be used with minimum modification on any computer using the CP/M operating system. This makes a vast library of application programs for this system instantly available commercially, whereas with a new operating system they would have to be written specifically for the operating system. EPSON has added a variety of useful functions to CP/M version 2.2 in developing the PX-8 operating system. Further details of the use of CP/M on the PX-8 are given in Chapter 3.

### 1.3.2 Utilities

The PX-8 has the ability to use programs loaded from the built in microcassette tape or from floppy disk (using the optional disk unit), but it has a more powerful facility to use programs which reside in ROMs which can be plugged into the sockets underneath the computer. This is a more convenient method of storing programs than either tape or disk, since the programs are available almost immediately, whereas with disk and tape it can take a noticeable time to load them into memory.

- **BASIC**

The PX-8 is equipped with a powerful BASIC interpreter which is an extended version of Microsoft BASIC which is a language widely used for personal computers. Programs can be written by the user and are also available commercially. There are also listings of such programs printed in books and magazines which can either be typed in and run directly, or will require a minimum of modification. Extra commands have been added to cater for the added features of the PX-8.

For details of BASIC, refer to the BASIC Reference Manual.

- **Portable WordStar™**

The word processor WordStar™ is one of the most popular wordprocessing programs available on microcomputers. Portable WordStar™ is an enhanced version of the Micropro WordStar™ which been specially modified to take account of the features of the PX-8. Whereas the PX-8 can be used directly as a wordprocessor with the WordStar ROM installed, the files can be transferred to other computers using the built in communications port of the PX-8. This means that the PX-8 can retain its portability where necessary and another computer used for the main processing.

- **Portable Calc™**

Apart from use as wordprocessors, the other main impact on the business world made by microcomputers has been in the area of electronic spreadsheet programs. The Portable Calc™ program is a ROM based program in this category.

- **Portable Scheduler™**

Because of the built in clock facility of the PX-8, the computer is ideal for keeping a diary. The Portable Scheduler™ is a program which keeps a diary which not only allows a record of the users appointments to be kept, but also enables the time to be signalled using the ALARM function of the PX-8.

These utility programs are available as optional ROM capsules. For details of their use refer to the reference manual for each utility program. They represent a small proportion of the software available. Please consult your dealer for the particular application you require.

## *Chapter 2*

# ***STARTING UP AND OPERATING THE PX-8***

The first section of this chapter describes how to set up your PX-8 for the first time. The remainder of the chapter deals with the general functions of the computer.

## **2.1 Starting Up**

### **2.1.1 Precautions**

#### **a) The computer**

The PX-8 is a precise machine and in order to achieve the best performance, you should observe the following points:

- i) Avoid using or leaving the PX-8 in a place exposed to direct sunlight.
- ii) Avoid using or leaving the PX-8 in a damp place.
- iii) Avoid using or leaving the PX-8 in a place exposed to strong vibration.
- iv) Avoid using or leaving the PX-8 in a place exposed to high temperatures.
- v) Do not allow the PX-8 to be subjected to shocks of any kind.

#### **b) The AC adapter**

An AC adapter is provided with the PX-8 to re-charge the built-in batteries. When using this AC adapter:

- i) Avoid plugging the adapter plug into the AC adapter jack of the PX-8 without plugging the adapter to an AC outlet.
- ii) The shape of the power plug differs according to the country for which the PX-8 has been shipped. In certain countries there may not be a power plug attached. Your dealer will be able to advise you on the correct plug in this case.
- iii) The PX-8 will take approximately 8 hours to charge if the PX-8 is switched off. However, if the power switch is ON the time will be greater (at least 11 hours if no access is made to any device including the keyboard).

- iv) If you use the PX-8 while charging it at the same time, you will not be able to use it for as long as you might expect. If you intend to use the PX-8 from the battery alone, the adapter should be removed from the PX-8 and then reinserted in order to ensure it will fully charge. You should then ensure that it is left for at least 8 hours before use in order to achieve the maximum length of use.
- v) The maximum length of use on battery power is 15 hours without Input/Output operation.
- vi) There are special circuits to prevent overcharging of the battery.

**WARNING:**  
 Never use any AC adapter other than that provided.

### 2.1.2 Before switching on

Your PX-8 cannot be used immediately after unpacking. The following procedures must be carried out before turning on the power switch.

Your dealer will probably have carried out the procedures in this section when you receive the computer, in which case you can go to section 2.2 to see how to operate the computer.

#### a) Inserting and charging the battery

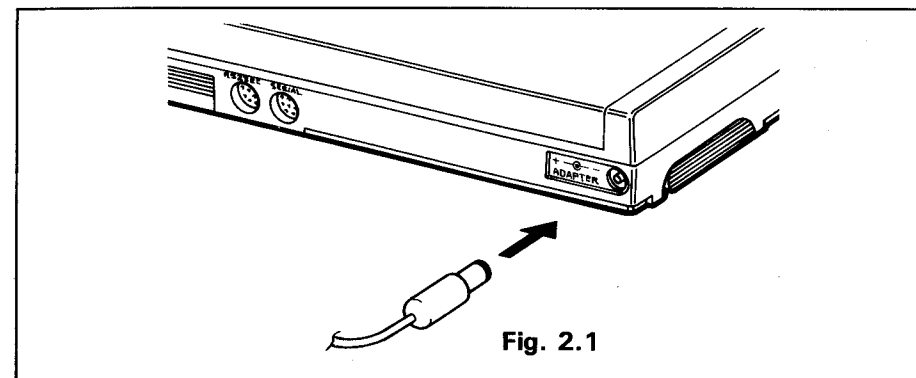
The PX-8 has been designed as a portable computer powered by rechargeable internal batteries. There are two battery units, main and backup. When the main batteries have insufficient power remaining to allow continuous operation, a message appears on the screen for 20 seconds:

### CHARGE BATTERY

then the power is automatically turned off and the backup battery takes over so that the program or work in progress is retained until the batteries can be re-charged. If the computer is not switched on the batteries are used to retain the contents of the memory. It is possible, therefore, for the batteries to become discharged. In order to prevent this the computer will normally be shipped with no main batteries. You should therefore proceed as follows:

- i) Insert the batteries as described in the addition sheet, or as described in Appendix C.

- ii) Plug the AC adaptor power cable into an AC outlet.
- iii) Plug the AC adaptor output cable into the PX-8 adaptor jack.



- iv) About 8 hours (or 11 hours if the power switch is left on) are required to fully charge the battery units.

If when you come to switch on the computer the "CHARGE BATTERY" message comes up on the screen, it is still possible to use the computer while it is charging.

IF NO DISPLAY IS VISIBLE WHEN YOU FIRST SWITCH ON THE COMPUTER the batteries may be discharged. This can occur at any time. Simply plug in the charger, wait 10 seconds and then switch on again.

IF NO DISPLAY IS VISIBLE AT THIS STAGE, the batteries are so low that all the power is being used to charge them and there is no additional power available to perform the normal functions of the computer. Switch the power switch off, wait a few minutes and try again.

#### b) DIP Switch settings

As mentioned in Chapter 1, all of the layouts of character keys shown in table 2.1 are available with your PX-8 regardless of the markings on the key tops. The keyboard layout can be altered by means of the DIP switch (SW 4) which lies under a cover beneath the PX-8 near the keyboard (see fig 2.2). It is also possible to alter the character set by means of software. This is described in Appendix A 'ESC codes' using ESC "C" and Chapter 3 under the CONFIG.

Open the DIP switch cover underneath the PX-8 (fig 1.1c) and check that the DIP switch settings correspond to the characters for the country you require,

as follows:

When you open the DIP switch cover you will see a silver flexible flap. Lift this out of the way and the DIP switch will be seen on the left (if the DIP switch cover is towards you as you are looking at the base of the PX-8). The DIP switch can then be set. The individual switch is in the ON position when it is towards the side which is marked ON.

Only the first four positions of the DIP switch are used for the character sets. The settings for various keyboard layouts are shown in table 2.1. The switches can be changed with the tip of a ball point pen or a matchstick.

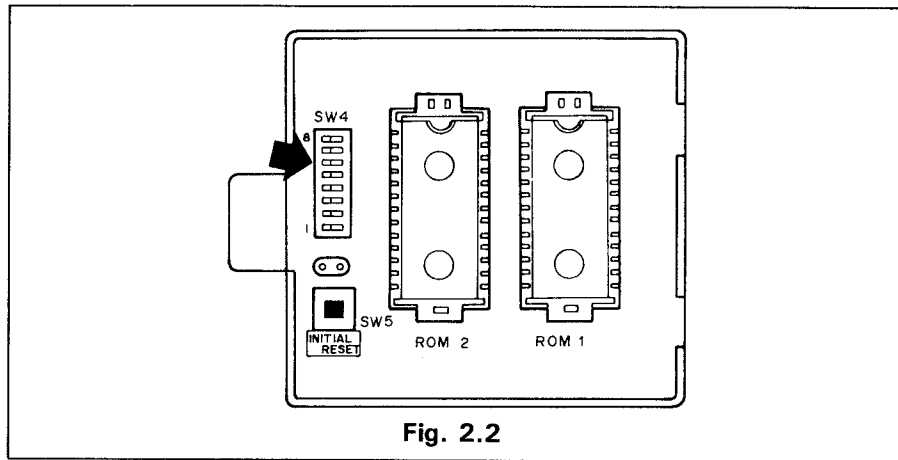


Fig. 2.2

Table 2.1

Keyboard Layout	DIP switch position			
	1	2	3	4
USA (ASCII)	ON	ON	ON	ON
French	OFF	ON	ON	ON
German	ON	OFF	ON	ON
English	OFF	OFF	ON	ON
Danish	ON	ON	OFF	ON
Swedish	OFF	ON	OFF	ON
Norwegian	OFF	ON	ON	OFF
Spanish	OFF	OFF	OFF	ON
Italian	ON	OFF	OFF	ON

**NOTES:**

i) **IMPORTANT:** Altering the settings will not cause any change to occur until the reset procedure outlined in section 2.2.5 is carried out.

- ii) Other DIP switch settings are used for different purposes. The position of switch 5 is used to decide whether a RAM check should be carried out if an Intelligent RAM disk is connected. Switch 6 is concerned with certain characters output to the printer when a screen dump is carried out. Both of these will be explained in Chapter 4.
- iii) Switches 7 and 8 are not used. They should be left in the OFF position as set in the factory.

**2.1.3 Initialization**

There are a number of times when it is necessary to reset the system. There are three different ways of resetting the system. The action of these three reset modes are summarised in section 2.2.4.

A complete reset (resetting the sub-CPU) and initialisation must be carried out if:

- i) The batteries have been changed or inserted for the first time on purchasing.
- ii) The system has become so corrupt as to leave no alternative but to carry out a complete reset. This is a very rare occurrence and there are less drastic ways of resetting the computer. The complete reset procedure should only be attempted as a last resort. See section 2.2.4 for a comparison of the different reset situations.
- iii) The optional expansion unit has been installed. (e.g. RAM Disk Unit)

**a) Resetting the sub-CPU**

The procedure for resetting the 7508 sub-CPU is as follows:

- i) Turn off the power switch.

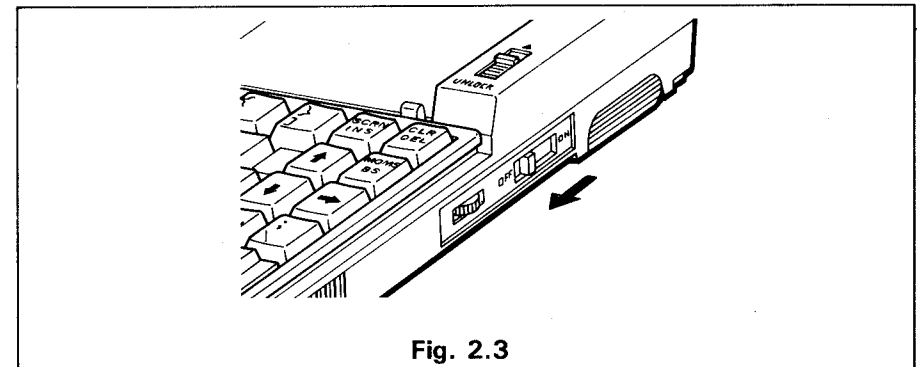


Fig. 2.3

ii) Open the plastic switch cover on the base of the PX-8.

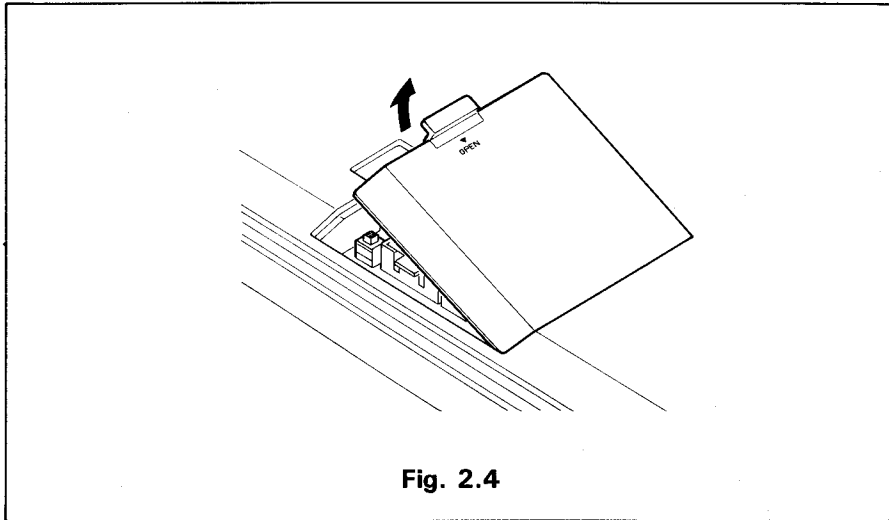


Fig. 2.4

iii) Press the push button reset switch near the DIP switch. The switch is covered to prevent accidental resetting of the sub CPU. It has a hole in the top through which it is possible to push the switch. Do not use a conductive implement to push the switch, or one whose end can break off (e.g. a pencil).

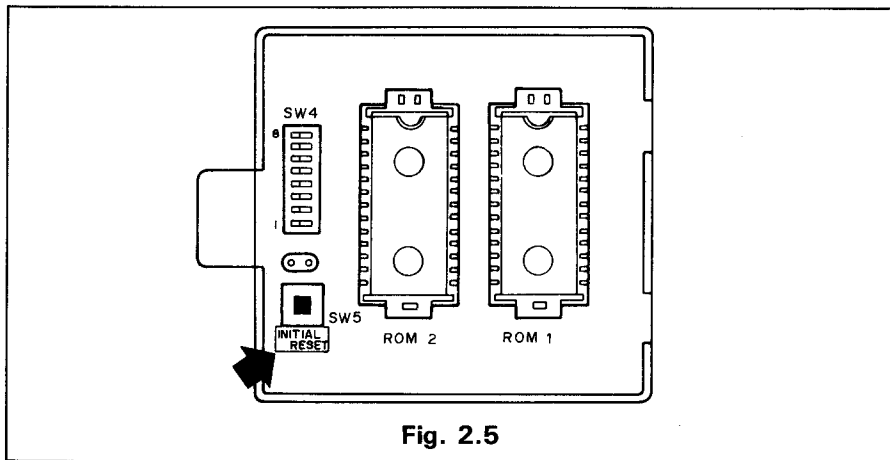


Fig. 2.5

iv) Replace the cover. Turn the PX-8 over and switch the power back on. The following message will then be displayed on the LCD screen:

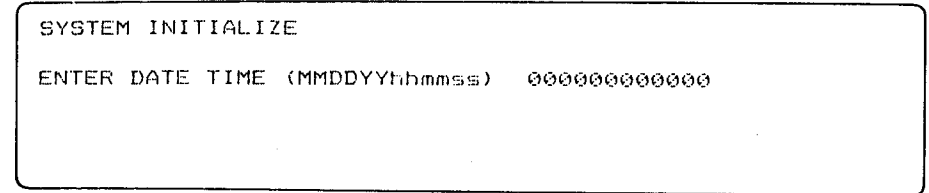


Fig. 2.6

This display image is the same as that displayed after system initialization described in Section 2.2.5b.

The display shows the first of a series of questions. These set the date and time, and the organization of the memory.

The first question the display shows allows the date and time to be entered. The expression:

**MMDDYYhhmmss**

shows the order in which the items have to be entered. The number of items corresponds to the row of zeroes to the right of the message. The first zero is covered by a flashing block. This is the cursor, which is used throughout the operation of the computer to show the position of the next character to be printed. It can sometimes be moved by using the cursor keys which are the keys marked with an arrow on the right of the keyboard. This movement is under the control of either an applications program or the overall controlling program or operating system which on the PX-8 is called CP/M. The use of CP/M is discussed in Chapter 3. In this particular case only two of the cursor keys are active. They are the right and left pointing arrows which allow movement of the cursor within the bounds of the row of zeros. By using these keys the flashing cursor can be placed over any one of the zeros and when you have chosen a particular position the value can be altered by pressing the appropriate numerical key. The cursor then moves to the next position to the right.

The various letters correspond to the date and time as follows:

**MM:** month (e.g. 01 for January and 11 for November)

**DD:** day of the month

**YY:** year (e.g. 84 for 1984)

**hh:** the hour (on a 24-hour clock basis)  
**mm:** minutes past the hour  
**ss:** seconds past the minute

When these figures have been typed in, any mistake can be corrected by moving the cursor to sit over the incorrect number and then retyping that number. The complete values of the time and date are not entered into the memory of the PX-8 until the key marked **RETURN** is pressed. You should check that the values are absolutely correct before pressing the **RETURN** key. Leading zeros must be added if the month etc is less than 10. Failing to do this will result in the information being out of sequence.



**WARNING:**

*The date and time are not actually updated until the day is input. It is possible to update the time and date using either the CONFIG program (see Chapter 3) or using the TIME\$ and DATE\$ commands in BASIC (see the BASIC Reference Manual). These give accurate updating of the time.*

When **RETURN** is pressed, the next question comes up on the screen:

**ENTER DAY (0 to 6)**

The cursor is flashing over a single zero this time, waiting for a number from 0 to 6 to denote the day of the week. The days are represented numerically in the order:

- 0 Sunday
- 1 Monday
- 2 Tuesday
- 3 Wednesday
- 4 Thursday
- 5 Friday
- 6 Saturday

The day is not changed in the memory of the PX-8 until the **RETURN** key is pressed. The day is then stored and the next question is asked.

The third question is

**ENTER RAM DISK SIZE**

The PX-8 allows part of the memory to be reserved to store programs and data. It can be used as if it were a disk drive. Since the memory used is the standard Random Access Memory used to run programs and store working data, the part of memory used as a pseudo-disk drive is known as the RAM disk. In accessing the drive under CP/M the drive A: is assigned to the RAM Disk as described in Chapter 3. It is also possible to have an additional block of memory which gives a further 60 or 120 kilobytes of storage which plugs into the back of the PX-8. Further details are given in Chapter 4. If this extra item, the Intelligent RAM disk, is attached this particular question will not be asked because no internal memory can then be allocated for use as a RAM disk.

The cursor lies over the first digit of the default value of 9 kilobytes (9K) expressed in the form "09". Default means the value the PX-8 thinks you would normally expect to use, and which it will set unless you change this value. Up to 24K of RAM can be reserved for the RAM disk.

SYSTEM INITIALIZE

```
ENTER DATE TIME (MMDDYYhhmmss) 000000000000
ENTER DAY (0 to 6) 0
ENTER RAM DISK SIZE 09
```

Fig. 2.7

The procedure for entering the size of RAM disk you require is the same as with the previous questions. Since the value entered is that displayed, simply press the **RETURN** key if you wish to use the default value of 9K bytes. If you wish to change the size of the RAM disk, alter the value and press the **RETURN** key. The RAM disk size can only be set in 1K byte units and the value 0 and 2 to 20 can be input.

When the RAM disk size has been set the next question is:

**ENTER USER BIOS SIZE 00**

This is a facility for advanced programmers. The cursor will be positioned over the first digit of a pair of zeros. Unless you have a specific need to use the USER BIOS area simply press the **RETURN** key at this stage.

The use of the USER BIOS area is covered in the OS Reference Manual. The location of the USER BIOS area can be seen in the memory map (Appendix F). The number entered in response to this question denotes the number of 256

256 byte blocks. Note that if USER BIOS area is reserved, the total area of the USER BIOS plus the RAM disk cannot exceed 24K bytes. If U is the number entered for the USER BIOS and R is the number entered for the RAM disk, then  $U/4 + R$  must not exceed 24. This is because the RAM disk is assigned in units of 1K and the USER BIOS in units of 1/4 K.

Before the USER BIOS question has been answered the display will show

```

SYSTEM INITIALIZE

ENTER DATE TIME (MMDDYYhhmmss) 000000000000
ENTER DAY (0 to 6) 0
ENTER RAM DISK SIZE 09
ENTER USER BIOS SIZE 00
  
```

Fig. 2.8

On pressing the **RETURN** key it will clear and ask a further question.

### RAM DISK FORMAT (Y/N) ?

Formatting a disk makes it possible for programs and data files to be stored as if the RAM disk were a conventional floppy disk. Until it is formatted data cannot be written to the disk. There are times when it is necessary to go through the initialization procedure when the disk contains files. In this case only it is permissible to say 'N' to this question. The disk will not be formatted, but **there is no guarantee that the files will be intact or that the RAM disk can be read.**

When the Y key is pressed, the RAM disk will be formatted and the display will change to show the MENU screen.

A typical MENU screen would be:

```

*** MENU screen *** 03/01/84 (WED) 04:26:07 54.5k CP/M ver 2.2 PAGE 1/1
C:PIP
C:PIP COM C:STAT COM C:SUBMIT COM C:XSUB COM
C:FILINK COM C:TERM COM C:CONFIG COM B:BASIC COM
  
```

Fig. 2.9

This type of screen makes it easier to run programs and saves typing. The setting up and use of the MENU screen is described in section 2.2.3d.

## 2.2 Operating The Computer

### 2.2.1 The Keyboard

If you are used to using a normal typewriter or another computer, you will see that the keyboard is similar. You will already have used the keyboard if you have set up and initialized the PX-8, but in order to use all of its facilities you will need to study this section to understand the special keys and key combinations. There are also special key combinations which only apply if you are using CP/M, the BASIC language or a particular applications program. Special use of certain keys and key combinations under CP/M are explained in Chapter 3, and under BASIC in the BASIC Reference Manual. Consult the appropriate Manual for any applications software.

#### a) Alphanumeric keys

Most keys on the keyboard generate characters, alphabetic, punctuation and numerals or graphics characters. These keys are called auto-repeat keys. Each auto-repeat key generates a certain character code repeatedly if it is pressed and held on for more than a specified period of time. This normally causes a character to be printed on the screen. Some keys are switch keys which alter the character code output by the character keys, for example to allow one key to output either upper or lower case characters. Other keys are special keys which allow insertion and deletion of characters, for example.

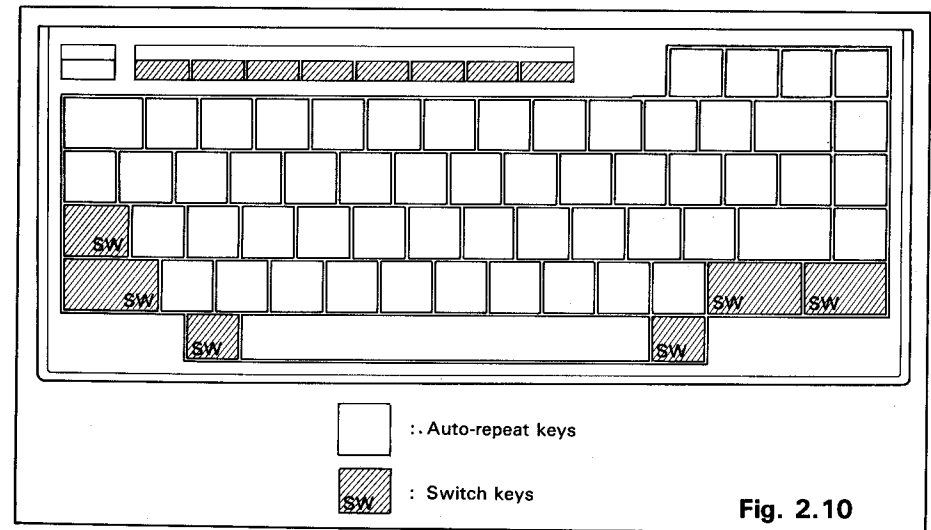


Fig. 2.10

The time before starting repetition and the repetition interval can be set by the



user by means of an ESC code sequence. See Appendix B.

If you are not familiar with a computer with auto repeat keys, you may hold them down too long initially, although you will soon adjust to the response time. If the key is held down past the repetition time, the character will be printed more than once and this could cause a program to respond in a way you might not expect. For example the System Display uses the **ESC** key to move back through the various levels, and also as the exit key. If you hold down the key for too long it is possible to exit when you did not mean to.

Sometimes a key may not appear to respond. This is because a program is processing data. Do not press the key again as the key you have pressed is stored in a buffer and will be retrieved when the program is ready to use your input. Continued pressing of a key may cause action to occur which you did not intend.

### b) Switch keys

Switch keys allow the other keys to perform more than one function. For example, they can allow upper and lower case (capital and small letters) to be obtained from the alphabet keys.

**SHIFT** : Pressing this key together with a key other than a switch key enters the alternative code assigned to that key. The **SHIFT** key should be held down before the other key is pressed. Thus if any numerical key is pressed whilst holding the **SHIFT** key down the alternative character above the numeral on that key will be printed. The **SHIFT** key also allows upper and lower case letters to be obtained from the alphabetical keys.

**CAPS LOCK** : Pressing this key makes the alphabet keys enter upper case letters without pressing the **SHIFT** key. The lock is removed by pressing the key again. When the **CAPS LOCK** is set the LED marked "CAPS" at the top of the keyboard will be lit. If the **SHIFT** key is pressed when the Caps Lock is set, the lower case character is printed instead.

**CTRL** : There are two **CTRL** keys, one to the left of the keyboard above the **SHIFT** key, and one to the right of the space bar. They are equivalent. Pressing either key makes it possible to enter control codes from the keyboard. Control codes are characters which are not printed. Some correspond to characters such as a carriage return, line feed, etc., others are used by applications software packages to perform such functions as clearing the screen or going to the end of the text. Control codes are normally associated with alphabetic keys and are indicated by the word "Control" followed by the associated letter, for example, "Control C". This is often abbreviated with the "CTRL"

joined to the letter by a hyphen. For example pressing the **CTRL** and the **C** key at the same time could be written **CTRL**-**C**. Another common way to show a control character is to place a "^" character in front of the letter. Thus **CTRL** - **C** could also be written as "^C". This will be printed on the screen if a control character is entered on the command line in CP/M. To obtain a control character the **CTRL** key should be held down and then the alphabetic character pressed.

**NUM GRAPH** : This key has two functions. Pressing a character key together with this key enters the graphic character code assigned to the key. Not all keys have graphics characters assigned to them. It is also possible to have user defined characters. Those are defined in the codes indicated in ( ).

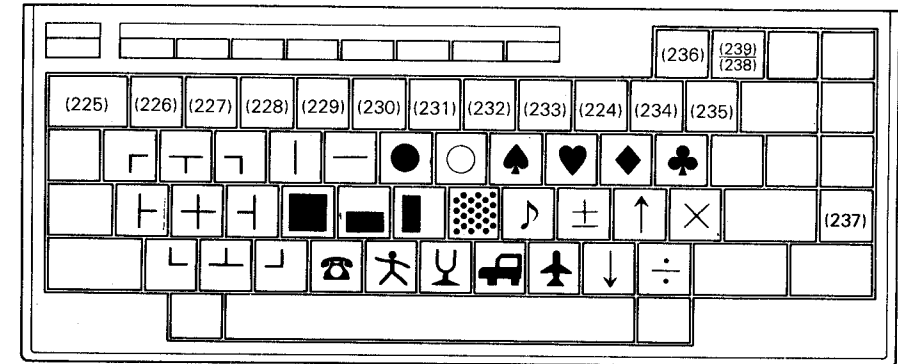


Fig. 2.11a ASCII keyboard

If the **NUM GRAPH** key is pressed while the **SHIFT** key is held down, a block of keys can be used as a numeric key pad. The key layout is as follows:

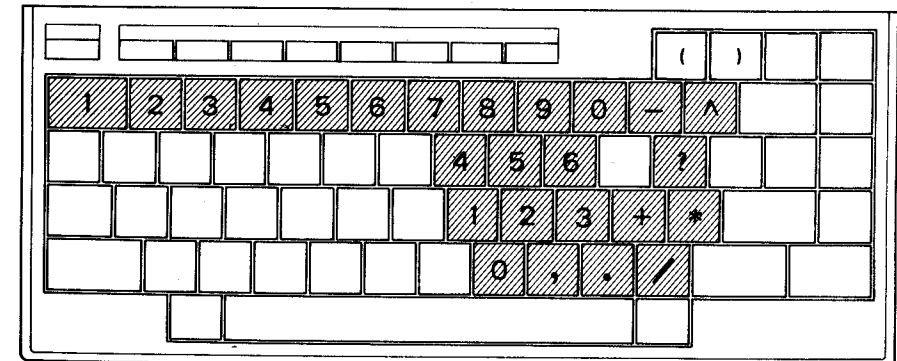


Fig. 2.11b ASCII keyboard

When the PX-8 is in the numeric keypad mode the LED at the top of the keyboard marked "NUM" will be lit. Pressing this key together with the **SHIFT** key when the PX-8 is in this mode returns to the normal state.

### c) Special keys

There are a number of special keys some of which allow control of the cursor and also simplify operation of the PX-8. Some of them also allow interruption of the computer so that, for example, the screen can be dumped to a printer.

The CURSOR KEYS are the keys marked with arrows on the right of the keyboard. They are used to move the cursor on the screen under control of the applications program. As was seen in the initialization, they may not all be active at once and may only allow movement within certain limits. The cursor keys are not supported by CP/M.

**RETURN** : This means carriage return, and is normally used to place the cursor at the beginning of a line. Since this would place the cursor on top of the characters already printed, a line feed is normally added to a carriage return by the software. The **RETURN** key is also used to signal the end of entry when a sequence of characters are being entered into the computer, otherwise the software has no way of knowing when the user has finished. Until the carriage return character is entered, the software will wait for the next character, unless a single character entry is required. The ASCII code for a carriage return is 13 decimal (0D hexadecimal), which corresponds to a **CTRL** - **M** .

**STOP** : In CP/M pressing this key clears the keyboard buffer and makes a warm start of the system. In BASIC it is used to stop a program.

**CTRL** and **STOP**: Pressing these keys simultaneously immediately terminates I/O operation, and makes a warm start of the system.

**ESC** : Pressing this key enters an ESCAPE code (ASCII code 27 decimal). It is used frequently to exit from many of the special function programs of the PX-8.

**PAUSE** : Pressing this key enters a PAUSE code (ASCII code 19 decimal) which can be used to stop printing to the screen. For example if there are too many files on a disk to be displayed on the screen at one time pressing the **CTRL** key can temporarily stop the printing of the filenames. The effect of a pause can be reversed by pressing any other key.

**CTRL** and **ESC** or **PAUSE**: Pressing the **CTRL** and **ESC** or **PAUSE** keys calls a subroutine specified.

**HELP** : Pressing this key enters a **HELP** code (ASCII code 0). This is sometimes used in applications software to show a table of commands etc. On the PX-8 the main use of the key is in combination with the **CTRL** key to enter the System Display.

**CTRL** and **HELP** : Pressing these keys simultaneously turns on the System Display. This is described in full in section 2.2.2.a.

**TAB** : This is a key which moves the cursor to pre-set positions across the screen. These are normally in steps of 8 characters. When the TAB key is pressed it generates an ASCII code 9.

**DEL** : Pressing this key enters a **DEL** character (ASCII code 127 decimal). It is usually used to delete the character under the cursor. However, some application programs use it in different ways. It has a special use on the CP/M command line and this will be explained in Chapter 3.

**CLR** : This is obtained by using the **SHIFT** key with the **DEL** key. Both keys must be pressed even if the **CAPS LOCK** key is active. It enters a - which has the ASCII code 12 decimal. If used in an applications program it normally clears the screen.

**INS** : Pressing this key enters an ASCII code 18 decimal. Normally this is used to allow characters to be inserted. Its use on the CP/M command line is described in Chapter 3, as it has a different function.

**SCRN** : This key is used to change the cursor tracking mode. The PX-8 displays a window on a virtual screen. The cursor moves over the virtual screen. Normally the window follows the cursor. This is called tracking mode. The key (obtained by pressing the **INS** and **SHIFT** key) is used to turn the tracking mode on and off. If the **SCRN** key is pressed while the PX-8 is in tracking mode, the window will be frozen over a particular portion of the virtual screen. This is known as the non-tracking mode. Pressing the **SCRN** key when the PX-8 is in this mode will turn the tracking mode on again and move the window on the virtual screen so that the cursor is displayed in the window.

**BS** : This key causes the cursor to Back-Space and delete the last character. It enters a BACKSPACE character (ASCII code 8). In some applications

software, and on the CP/M command line, it is the only way to delete a character.

**HOME** : The **HOME** key as the name suggests moves the cursor to the top left hand corner of the virtual screen (not the window).

### KEYS **PF1** TO **PF5** : The Programmable Function keys

These are a set of keys which enable the user to enter a string of characters which will be printed when the key is pressed. This is as if they had been typed in by the user at the keyboard. The PX-8 is programmed with a default set of characters which allow the most common CP/M commands to be input with one keystroke. Also when BASIC is entered, a default set is available, which corresponds to the common BASIC commands.

The strings are altered by means of the CONFIG program in CP/M, which is described in Chapter 3. The BASIC Reference Manual describes how to alter them in BASIC.

Pressing a programmable function key together with the **SHIFT** key enters a different user defined string. The names PF6 to PF10 are given to the shifted keys **PF1** to **PF5**, i.e. a value of 5 is added to the number of the programmable function key if it is shifted.

Since there are ten function key strings it is sometimes difficult to remember which key has which string associated with it. The strings assigned to each function key can be displayed on the eighth line of the screen. If the eighth line shows the following display:

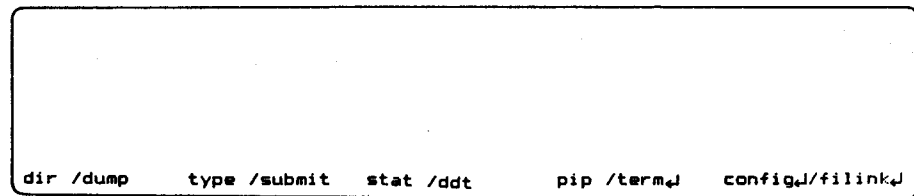


Fig. 2.12

This means function key **PF1** will print the characters "dir" when pressed, and "dump" when pressed with the shift held down (i.e. the **PF6** key is pressed) and so on across the screen. The "␣" character means that the equivalent of pressing the **RETURN** key is also added to the characters printed. A **RETURN** is not added with the characters if it is likely that other commands or parameters

need to be added.

The assignments can be changed using the CONFIG program described in Chapter 3. This also has an option to switch the display of these function key assignments on and off.

Pressing a programmable function key ( **PF1** to **PF4** ) together with the **CTRL** key calls a subroutine specified by the user. If no subroutine is specified, nothing happens.

Pressing the **CTRL** and **PF5** keys simultaneously outputs, the contents of the display screen to the printer.

### 2.2.2 Switching the PX-8 on and off

The PX-8 has a power switch on the right hand side (see fig 2.3) which is the normal way the computer is switched on and off at the beginning and end of a session of using the PX-8. However, the PX-8 is not switched off completely as a mains operated computer would be if the power was turned off. The power switch can be thought of as a means of temporarily halting the operation of the computer, so that the batteries are conserved. As soon as the power is turned on again it is possible to begin exactly where you left off, even if you were in the middle of typing a word or running a program.

The power switch is not the only means by which the power can be switched on and off. The power switch of the PX-8 is controlled by the computer, and can and is switched off under software control. Thus, if it is not used for a length of time (which can be set by the user using the CONFIG program described in Chapter 3), it will switch itself off to conserve power. The user can also program the computer to switch itself off at specific times. Furthermore the PX-8 can switch itself on, run a particular program and then switch itself on again to run the same or another program. This ability to switch itself on and off is a powerful feature of the computer.

To switch the power on when the switch is already in the on position, switch it off and on again. This means, holding the computer with the keyboard nearest to you, moving the switch towards you and then away from you again.

To switch the computer off when it is already in the off position, switch the power switch to the on position and then off again.

Because the computer can switch itself on and off, the power switch position may not reflect the true state of the power. For example if the PX-8 switches itself off, the power switch will still be in the on position, even though the power is off. Similarly, if the PX-8 switches itself on, the power switch will be in the off position even though the power is on. Holding the PX-8 normally with the keyboard towards you, the power switch is on the right hand side of the computer. The ON position is towards the back of the computer. The OFF position is towards the front of the computer.

There are also two ways to switch the power off: **the restart and continue modes**. The selected mode is determined by the conditions prevailing at the time the power is turned off. When the power is switched off in the restart mode, the Menu or CP/M command line display will be activated when the power is turned on again. When the power is switched off in the continue mode, the operation which is being executed when the power is turned off will be continued when the power is turned on again.

i) Conditions under which the restart mode is selected:

- When the power switch is turned off.
- When system initialization is performed.
- When the sub-CPU is reset.
- When the BIOS POWEROFF routine is called. (See Chapter 5)

ii) Conditions under which the continue mode is selected:

- When the power switch is turned off while pressing the **CTRL** key.
- When the auto power off time is reached, because no key has been pressed. (See Chapter 3.)
- When the main battery voltage drops below a certain level (power failure).
- When the BIOS POWEROFF routine is called. (See Chapter 5)

## SWITCHING THE PX-8 ON

As can be seen from the above explanation, when the PX-8 is switched on, a number of routes can be taken by the operating system. If the power was switched off in the continue mode, operation will continue at the point at which it was switched off. If the PX-8 was simply switched off in the restart mode, there are a number of options possible. These also vary with the way the computer is switched on (i.e. manually or under the control of the PX-8). In order to understand these possible situations, an overview of the operating system is necessary. However, to keep the information together for reference, the following summary of what can happen when the PX-8 is switched on is given at this point, even though some of the concepts have not been covered at this stage in the Manual.

- i) If the Password has been set the PX-8 will ask for the Password to be entered. It will not be possible to operate the system any further without entering the correct Password.
- ii) If the PX-8 has been switched off in the continue mode, the program which was being used, will continue at the point it had reached when the PX-8 was switched off. Thus if the program was waiting for input, or was processing information, this would continue.
- iii) If the PX-8 was switched off in the restart mode, either the MENU page or the CP/M command line would be displayed.
- iv) If BASIC was in use when the PX-8 was switched off, it is possible to re-enter BASIC. If the MENU is switched on, this is done simply by pressing the **RETURN** key. If the MENU is switched off, it would appear not to be present. It can be retrieved (see the SAVE command in Chapter 3), but you may not remember that you were using BASIC. It is wise to set the MENU on before switching off when using BASIC, so that it is evident that BASIC was in use, and to prevent any loss of programs stored in the five program areas.

### IF YOU SWITCH THE COMPUTER ON AND NO DISPLAY APPEARS:

First check that the view angle of the LCD is set so that you can see the screen display.

If there is still no display, the battery needs recharging. Plug in the AC charger, wait a few seconds and then switch on again.

## 2.2.3 The PX-8 operating system modules

The operating system of the PX-8 is functionally divided into several units which are referred to as modules. Some of them are part of the CP/M operating system and others supplement it.

- i) **System Display module**  
This module enables the microcassette tape to be operated manually and also to check and reset some of the system parameters. This module can normally be operated while applications software is running. Many of the following modules have parameters which can be changed by using this module.
- ii) **Password module**  
This module makes it possible to stop unauthorised users from using the PX-8 without knowing an entry password.
- iii) **MENU module**  
This module displays program files and data files on the screen and makes it easier for the user to run any program file by selecting it on the screen, using the cursor keys rather than having it to type the full name.
- iv) **Screen dump module**  
This module is used to make a copy of the current screen on a printer.
- v) **Console Command Processor (CCP)**  
This is the part of the CP/M operating system which interprets command strings typed on the keyboard. This module includes the CP/M built-in commands (DIR, TYPE, REN, ERA, SAVE and USER).
- vi) **Basic Disk Operating System (BDOS)**  
This is the part of the CP/M operating system which manages disk files. It also treats the ROM capsule and RAM as disk devices.
- vii) **Microcassette Tape Operating System (MTOS)**  
This module manages microcassette tape files.
- viii) **Basic Input/Output System (BIOS)**  
This part of CP/M which acts as the interface between the operating system and input/output devices e.g. the screen, keyboard and RS232 interface.
- ix) **Microcassette Input/Output System (MIOS)**  
I/O interface between MTOS and microcassette firmware
- x) **Clock module**  
This module controls the alarm and wake functions and updates any time displays.
- xi) **System activator**  
This module controls system activation, deactivation, the auto-start function and initialization.

The best way to understand the operation of the modules which are directly under the control of the user is to check and alter the status of the parameters involved using the System Display module.

#### a) System Display module

The System Display is brought on to the screen by pressing the **CTRL** and **HELP** key simultaneously. The screen is then changed to show a display similar to one of the following:

```
*** SYSTEM DISPLAY ***    03/01/84 (THU) 12:01:34    <MENU> <PASSWORD>
<RAM DISK> 009 kb    <AUTO START>
<USER BIOS> 000 256 b    <MCT MODE> stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA    <MENU FILE> 1 .COM 2 . 3 . 4 .
- Select number or ESC to exit.
  1=password 2=alarm/wake 3=auto start 4=menu 5=MCT
  <<- /    <- /mount    #/dirinit    ->> /erase    000/
```

Fig. 2.13a

```
*** SYSTEM DISPLAY ***    03/01/84 (THU) 12:01:34
<RAM DISK> 009 kb    <AUTO START>
<USER BIOS> 000 256 b    <MCT MODE> stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA    <MENU FILE> 1 .COM 2 . 3 . 4 .
- Select number or ESC to exit.
  1=password 2=alarm/wake 3=auto start 4=menu 5=MCT
  <<- /    <- /mount    #/dirinit    ->> /erase    000/
```

Fig. 2.13b

This screen image is referred to as the System Display. It shows a number of items of information which would be hard to determine except by running a separate program to interrogate the system. Since the System Display can be obtained even in the middle of an applications program it is a fundamental part of the PX-8 which you will be using regularly, so try altering as many of the parameters as possible to see what they do.

Looking around the screen you will see the following:

The first line shows the title "SYSTEM DISPLAY" in the top left hand corner. Followed by the date and time. It may show one or other of the words "<MENU>" and "<PASSWORD>". Figs 2.13 a and b show cases with and without these options displayed. These are visible if the corresponding options have been set.

The second line shows the ALARM or WAKE time together with the appropriate string. If this option has not been specified, the line will be blank.

The third line shows the size of the RAM disk as set on initialization or using the CONFIG program (see Chapter 3). Next to it will be an Autostart string if one has been specified. The title "<AUTO START>" will still be present even if there is no string specified.

The fourth line shows the size of the USER BIOS, in 256 byte pages as set by initialization or the CONFIG program. To the right of the line are a series of parameters associated with the Microcassette drive.

The fifth line is concerned with the MENU. The drives from which the files will be taken are displayed next to the "<MENU DRIVE>" title on the left. The right hand side of the line shows which file extensions are to be chosen for display on the MENU.

The sixth line shows the prompt string and flashing prompt to enable one of the options on the seventh line to be taken.

The eighth line shows the strings associated with the programmable function keys which are used to control the Microcassette drive. If a cassette has been mounted, only the **PF6** (shifted **PF1**) key will be active.

The following items CANNOT be changed by using the System Display:

- |                                 |            |
|---------------------------------|------------|
| 1) Month/Day/Year (Day of week) | (1st line) |
| 2) Hour:Minute:Second           | (1st line) |
| 3) RAM disk size                | (3rd line) |
| 4) Size of USER BIOS area       | (4th line) |

They are changed either by the initialization procedure, or the CONFIG program described in Chapter 3.

The following sections explore the use of each of the modules which can be changed by using the System Display.

**NOTE:**

*System Display cannot be entered during PASSWORD or ALARM/WAKE display.*

## b) Password module

The Password module makes it possible to prevent the computer system from being used by unauthorised people. It should be used with care. It is not possible for the Password to be determined once it has been assigned. If the Password is being used, make sure that all data and programs including those in the RAM disk are saved to disk or Microcassette tape before switching off. If it is necessary to break the Password, all data and programs will be lost.

From the System Display press the **[1]** key and lines 6 and 7 will change to:

```
*** SYSTEM DISPLAY ***      03/01/84 (THU) 12:01:34
<RAM  DISK> 009 kb      <AUTO START>
<USER  BIOS> 000 256 b  <MCT  MODE>      stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA      <MENU FILE> 1 .COM  2 .      3 .      4 .
- Select number or ESC to return.
<PASSWORD> 1=off 2=assign
```

Fig. 2.14

### i) Setting a password

To assign a password, press the **[2]** key and the prompt will change to:

```
*** SYSTEM DISPLAY ***      03/01/84 (THU) 12:01:34
<RAM  DISK> 009 kb      <AUTO START>
<USER  BIOS> 000 256 b  <MCT  MODE>      stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA      <MENU FILE> 1 .COM  2 .      3 .      4 .
- Input password, ESC to cancel.
```

Fig. 2.15

with the cursor on the seventh line awaiting the password. Type in your password (this is limited to 8 characters) and press the **[RETURN]** key when you have typed in all the characters. Remember the password you have typed because there is no way to find out what it is once you have pressed **[RETURN]**.

The System Display screen will show the word "<PASSWORD>" in the top right hand corner as soon as the **[RETURN]** key is pressed after the password has been entered.

Having entered a password, switch off the PX-8, wait a few seconds and switch it on again. The screen will now show:

PASSWORD

Fig. 2.16

The system cannot be used unless the correct password is typed from the keyboard. Type the password and press the **[RETURN]** key. The letters you type will not be displayed on the screen. If the typed word is incorrect, the cursor will be returned to the starting position. When the correct word is entered, the speaker will beep and the display will change to either the MENU page or the CP/M command line display.

### ii) Removing the PASSWORD

Press the **[CTRL]** and **[HELP]** keys to go back to the System Display again. This time after pressing 1, remove the password by pressing 1 again. If you had pressed 1 to change the password status and then decided not to do so, the **[ESC]** key could be used to return to the options choice line. In general, pressing **[ESC]** anywhere in the System Display will successively take you back a level, so that you eventually leave it altogether.

The user will be asked to supply the password under the following conditions, not just when the PX-8 is switched on:

- If power is turned on by the power switch or the wake time is reached with the password mode specified.
- If the alarm or wake mode is activated while the password is still being entered, both will be treated as an alarm string.

#### NOTE:

The PX-8 will only treat the wake string as if it were an alarm string if the Password is activated. The user will then be asked for the password again after pressing **[ESC]** to exit from the alarm/wake message.

- Pressing the reset button will return the user to the password screen.

The password mode is terminated in one of the following cases:

- a) When the password is typed correctly.
- b) When power is turned off.
- c) When the auto shut-off time is reached.
- d) When power failure is detected.

The password mode is interrupted when the alarm/wake time is reached.



**WARNING:**

*The only way to exit from the password if it is not known is to initialize the system by either pressing the reset button with the right-hand **SHIFT** and **NUM GRAPH** keys held down or by doing a full system reset, as when starting the up the system for the first time.*

*In this case all your programs will be lost and also anything stored in the RAM disk area.*

**c) Clock module - the ALARM and WAKE functions**

The clock module manages the software clock and controls the ALARM and WAKE functions which allow the PX-8 to switch on and present a message or start a program running. If the computer is in use, and alarm or wake time is reached, the program being run will be interrupted. The screen will clear and show the message.

In order to familiarise yourself with the possible ways of setting and using the ALARM and WAKE functions press **CTRL** and **HELP** to turn on the System Display and then press option 2 to change the ALARM and WAKE options. Lines 6 and 7 of the screen then change to:

```

*** SYSTEM DISPLAY ***      03/01/84 (THU) 12:01:34      <MENU>

<RAM  DISK> 009 kb      <AUTO START>
<USER  BIOS> 000 256 b  <MCT  MODE>  stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA      <MENU FILE> 1 .COM  2 .      3 .      4 .
- Select number or ESC to return.
  <ALARM/WAKE> 1=off 2=alarm 3=wake 4=message/string

```

**Fig. 2.17**

It is not possible to have both the ALARM and WAKE functions operating together. If option 2 is taken it will cancel any WAKE setting and replace it with an ALARM. Option 1 switches off whichever of the ALARM and WAKE has been set, and the message string (option 4) is used with both the ALARM and WAKE functions in a slightly different way.

*i) SETTING THE ALARM*

Before looking at the various options for setting the ALARM, the following example shows how to set the ALARM simply to switch on the computer and display a message at a specific time and date.

Press the **2** key and the prompt changes as follows.

```

*** SYSTEM DISPLAY ***      03/01/84 (THU) 12:01:34      <MENU>

<RAM  DISK> 009 kb      <AUTO START>
<USER  BIOS> 000 256 b  <MCT  MODE>  stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA      <MENU FILE> 1 .COM  2 .      3 .      4 .
- Set alarm time, ESC to cancel.
  MMDDhhmm

```

**Fig. 2.18**



The cursor is to the right of the prompt string. The PX-8 now expects you to type in the time at which you wish the alarm to sound. The prompt shows the order in which the date and time should be entered:

**M: Month**  
**D: Day**  
**h: Hour**  
**m: Minute**

Care should be taken to ensure that a leading zero is added to a single digit value.

In order to have the alarm sound within a reasonable time, so that more examples can be tried, set the date to that shown on the top line of the System Display, and the time to two minutes later. As an example if the date and time shows "01/18/84 (WED) 11:38:45" enter "01181140" for the ALARM time. This ensures that the alarm will sound at "11:40" on the same day. When you reach the last character, the cursor will flash over this character, to show you that you are in the last position; otherwise the cursor will move to the next position. If you decide you do not want to enter a value, simply press the **ESC** key. If one or more of the characters are wrong, you can move back and forth along the line using the cursor keys. When you press the **RETURN** key, the ALARM time will be entered onto the second line of the System Display.

The label "<ALARM MSG>" will appear to the right of the alarm time, but the rest of the line will be blank until a message string is inserted. As you have been returned to the prompt line shown in fig 2.17, press the **4** key so that a message string can be inserted. Write a message of up to 40 characters, for example "Time to telephone home". When the **RETURN** key is pressed the message string will be entered next to the "<ALARM MSG>" label.

When the alarm time is reached the speaker will sound a warbling note and the screen will clear to show the display of fig 2.19.

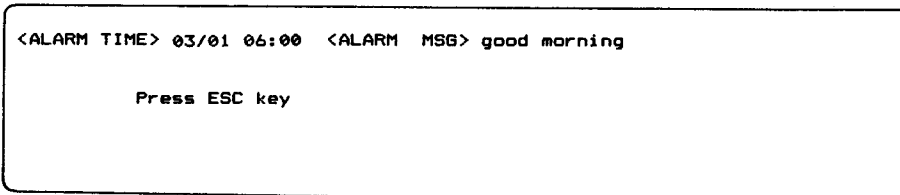


Fig. 2.19

If no message string was entered, the rest of the line after the "<ALARM MSG>" label will be blank.

You can return to whatever was happening when the alarm sounded by pressing the **ESC** key. The alarm might also have occurred when the PX-8 was switched off. In this case the speaker would sound in exactly the same way and the power switched on to display the same screen. As this can happen when you might not be near the computer and the battery could be discharged, the auto power off time will decide when to switch the power off (see section 2.2.2 and the CONFIG program of Chapter 3 for details of how to set this).

The PX-8 will exit from the ALARM screen when:

- The **ESC** key is pressed.
- Fifty seconds has passed.
- The power switch is turned off.
- The battery needs recharging.

Unless the power is switched off the PX-8 will revert to the state it was in immediately before the alarm was sounded.

#### ii) Ways of setting the ALARM

The example above showed how to set the alarm to go off at a specific time on a specific day. There are a number of ways that the alarm can be set, for example to repeat at specific intervals. This involves the use of wildcards. When an ALARM date and time is input, the PX-8 compares the current date and time with the string of characters input character by character. By using an asterisk or question mark as a wildcard in the ALARM string, the comparison will regard the position where the wildcard character was inserted as always matching. For example, type "\*\*\*\*0930" into the string following the "MMDDhhmm" prompt when taking the ALARM option. This will change the alarm time following the "<ALARM TIME>" label to "\*\*/\* 09:30". It will cause the alarm to sound every day at 9.30 am. The wildcard option is not completely flexible. It can only set the alarm to go off at intervals of one minute, ten minutes, one hour, twenty four hours and monthly. It is also possible to set the ALARM to go off at specific intervals in a given period e.g. every ten minutes for an hour. The following table is a summary of how to use the wildcard.

	TYPE IN	DISPLAY READS	NOTES
Every minute:	*****	**/** **:**	sounds every minute
Every 10 minutes:	*****5	**/** **:*5	whenever the 5 matches i.e. at 05, 15, 25 minutes etc
Every hour:	*****21	**/** ** :21	sounds at 21 mins past every hour
Every 24 hours:	****1730	**/** 17:30	at 17:30 every day
Every minute for ten minutes:	****093*	**/** 09:3*	from 9.30 am to 9.39 am
Every minute for an hour:	****14**	**/** 14:**	from 14.00 to 14.59
Every minute for ten minutes ten every hour:	*****3*	**/** **:3*	from 30 minutes to 39 minutes ten past the hour every hour:
Once a month:	**151015	**/15 10:15	On the 15th of the month at 10.15 am
Every day for a month:	06**1415	06/** 14:15	At 14.15 every day in June
A specific time:	11111111	11/11 11:11	On Nov 11 at 11.11

**NOTE:**

Only the asterisk or question mark should be used as a wildcard character.

The time should be checked before exiting from the System Display as the PX-8 carries out a comparison between your string and the string output by the clock. If the alarm does not go off when you expect it to, check the System Display to see that the time you entered was correct.

*iii) Setting the WAKE function*

The wake function makes it possible to automatically turn on the power and/or start a program or execute a command at any desired time.

Note that the WAKE or ALARM cannot be used together.

The procedure for setting the WAKE time is the same as that for setting the ALARM time, except that option 3 is taken instead of option 2 (Fig 2.17). All the wildcard options can be used in the same way.

The message string (option 4 Fig 2.17) is used to name a program which is to be run. It is used in a similar way to the AUTO START string (see the next module). The string input into the WAKE message is used as if the characters were typed from the keyboard when the WAKE function switches the PX-8 on. Thus the message "A:PROG^M" will run the CP/M COM file "PROG" on drive A: when WAKE switches the PX-8 on. The two characters at the end are a CTRL-M written as "^M". This is the carriage return character. Just as nothing will happen if you do not press the **RETURN** key when typing in a file name, so missing the "^M" from the end of a file name in a message string will normally cause the PX-8 to wait until the auto power off time is reached and then switch off.

For the options and rules to be followed in setting up a WAKE string, see the next module (the AUTO START string). Note that the WAKE string is entered as a message under the ALARM/WAKE option and NOT under the AUTO START string. The AUTO START string does not operate if the WAKE is executed.

When you try out the WAKE function, remember that it won't work unless you switch off the PX-8 before the WAKE time is reached. Set a WAKE time for a couple of minutes ahead and then enter a WAKE string. As an example you could use "DIR^M" to display a directory of the current drive. Make sure you have switched off the MENU for this experiment or one of two error conditions will arise. First, if BASIC is resident when you switch off, it will still be there when the PX-8 WAKES up. When the string "DIR^M" is encountered it will not be recognised by BASIC 'Ok' prompt.

Refer to the rules in the next section on setting the AUTO START string for details of how to overcome this.

**IMPORTANT:** In adding the "^M" to the end of the message string in WAKE, the "M" must be upper case. If the message string following the "<WAKE STRG>" label has a "-" character at the end you have entered "m" and not "^M".

The action taken when the WAKE time is reached differs according to the state of the PX-8 when the wake time is reached.

- If the power is turned on:

The WAKE behaves as if it were an ALARM. This is to prevent the program initiated by the WAKE string from interrupting the task in hand when the WAKE

time is reached. The speaker warbles and the display changes in the same way as if an alarm was sounded. Instead of the WAKE string being a command string it is simply printed as a message string. The screen shows that a WAKE time (and not an ALARM time) has been reached when it appears as shown in Fig. 2.20, for example.

```
*** MENU screen *** 00/00/00 (SUN) 00:12:24 54.5k CP/M ver 2.2 PAGE 1/1
B: BASIC Good morning!
B: BASIC COM
```

Fig. 2.20

The appropriate action can be taken, treating the WAKE as an alarm message. The same criteria apply for termination as apply to the ALARM time being reached.

- If the power is turned off:

The action taken will depend on the state of the PX-8 when switched off.

- If the PX-8 is in the continue mode:

The power will be turned on and the program which was interrupted by switching off will be continued. There will be no message, the computer will simply behave as if it had been turned on manually. ANY WAKE STRING WILL BE IGNORED.

- If the PX-8 is in the restart mode:

- The power will be turned on and the WAKE string executed as if it had been typed in at the keyboard.
- If no WAKE string is specified the situation will be equivalent to the power being switched on manually, except that it will have been switched on automatically. Although the speaker will sound, there may not be anyone near the computer to notice that this has occurred. The PX-8 will then switch itself off when the auto-power-off time is reached.
- If no WAKE string is specified but an AUTO START string has been entered, the AUTO START string WILL BE IGNORED.

#### iv) AUTO START string

The AUTO START string is used to set a string which will be entered as if it had been typed from the keyboard when the PX-8 is switched on.

The AUTO START string will be ignored if the PX-8 is switched on in the continue mode. When the computer has been switched off in the continue mode, this is because the computer has been stopped in the middle of a program execution. To start up again with a new program could destroy valuable data.

To set the AUTO START string, press the **[3]** key when the System Display is as shown in Fig. 2.13a or b.

The prompt changes as follows.

```
*** SYSTEM DISPLAY *** 03/01/84 (THU) 12:01:34 <MENU>
<RAM DISK> 009 kb <AUTO START>
<USER BIOS> 000 256 b <MCT MODE> stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA <MENU FILE> 1 .COM 2 . 3 . 4 .
- Select number or ESC to return.
<AUTO START> 1=off 2=assign
```

Fig. 2.21

To assign the string, press the **[2]** key and the prompt changes as follows:

```
*** SYSTEM DISPLAY *** 03/01/84 (THU) 12:01:34 <MENU>
<RAM DISK> 009 kb <AUTO START>
<USER BIOS> 000 256 b <MCT MODE> stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA <MENU FILE> 1 .COM 2 . 3 . 4 .
- Input auto start string, ESC to cancel.
```

Fig. 2.22

As an example of how to use the AUTO START string from the System Display type 3 to select the AUTO START option.

When lines 6 and 7 show the options as in Fig 2.21 take option 2 and then type in the following string: "B: BASIC^M". This assumes the BASIC interpreter ROM is in the socket assigned to drive B: and should be changed to "C: BASIC^M" if it is in the C: drive socket.

Press the **[RETURN]** key when the string has been correctly entered.

Now ensure that the MENU is off by entering option 4 on the System Display and then taking option 1. MENU is described in the next module.

Switch the PX-8 off in restart mode, i.e. simply switch off WITHOUT holding down the **[CTRL]** key.

Wait a few seconds then switch it on again. The screen will clear and place you on the CP/M command line and the characters "B:BASIC" will be printed next to the system prompt. Since the carriage return has been added by the PX-8 from the "^M" characters at the end of the AUTO START string, this will then be executed and about ten seconds later the BASIC program menu will be displayed.

To cancel the auto start string, press the  $\square$  key when the display is as shown in Fig. 2.21, and the display will change as shown in Fig. 2.13b.

To change the AUTO START string, carry out the procedure for assigning a string. The previous string will be displayed and can be edited. It is possible to delete the character to the left of the cursor but not to insert characters. Typing over a character replaces it. In many cases it is simpler to cancel the string and then reassign another one.

v) *Rules for setting up AUTO START and WAKE Strings*

- 1) If no carriage return is entered at the end of a string meant to be executed, the PX-8 will wait for the next character to be typed manually when switched on. This is the case either when the string is a WAKE string or an AUTO START string. A carriage return should be entered as the pair of characters "^M" and not as a lower case "m". If the characters have been entered as "m", the string will show "-" instead of the correct "M".

If the carriage return is not present, the PX-8 will wait until the auto-power-off time is reached and then switch off.

- 2) If the WAKE string switches on the PX-8 the AUTO START string will be ignored even if the WAKE string is blank.
- 3) If the PX-8 is restarted, either manually or by the WAKE function, and the computer is in the continue mode the WAKE or AUTO START strings will be ignored.
- 4) If the PASSWORD is set the PX-8 will not be able to AUTO START or use the WAKE string until the password has been correctly entered.
- 5) If the MENU (next module) is set then the AUTO START or WAKE string will be printed next to whatever is on the command line. When the carriage return is entered the complete string on the command line is executed. Since this includes the file name which is in the top left hand corner of the menu

list, an invalid command will almost certainly be present.

This can be overcome if the AUTO START or WAKE string backspaces over the file name entered onto the command line by the MENU. The control character for a backspace is CTRL-H. Thus if the MENU is likely to be set when the WAKE or AUTO START strings are used, add the following characters to the string:—"H^H^H^H^H^H^H^H^H^H^H". Eleven characters are needed. Thus to run BASIC as in the example above, the WAKE or AUTO START string should be "H^H^H^H^H^H^H^H^H^H^HB:BASIC^M". As with the carriage return the characters should be "H" and not "h".

It will not matter if this string is entered when the MENU is inoperative, and the PX-8 WAKES or AUTO STARTs on the CP/M command line.

The best course of action is to switch off the MENU if a WAKE or AUTO START string is used.

- 6) There is one case when it is not necessarily beneficial to switch off the MENU. This is when BASIC is resident in memory and a number of programs are loaded into the five BASIC program areas. In this case the AUTO START or WAKE string would be set up either to run one of the loaded programs, or to load a program from one of the disk drives. The BASIC Reference Manual gives details of how to set up AUTO START and WAKE strings under the AUTO START and ALARM commands of BASIC. These examples can be used in the same way to enter AUTO START and WAKE strings directly through the System Display.
- 7) Control Codes 00H to 1FH can be set using the combination of "H" and ASCII codes (40H to 5FH).



**WARNING:**

*It BASIC is resident and the AUTOSTART or WAKE string is not a BASIC program name (or even requests BASIC to be loaded), an error will be generated. Thus in setting up WAKE and AUTOSTART strings, note should be taken of the way the system is set up. It is best to set the system up specially for the AUTOSTART or WAKE, so that errors do not occur.*

#### d) MENU module

When switching on the PX-8 it is sometimes difficult to find a particular program among a number of others on a disk device. Many of them may not be relevant. In many cases the same program is used over and over again, and like using the PF keys it is easier to press one key to load the program. The MENU module simplifies the selection of a program by allowing the types of file names which are displayed to be limited, and allows programs from a mixture of drives to be displayed together. It also allows the programs to be selected by movement around the displayed files using the cursor keys, and then running the program simply by pressing the **RETURN** key.

The MENU is controlled by option 4 of the System Display. Press the **CTRL** and **HELP** key to turn on the System Display. Now press the **4** key and lines 6 and 7 change to:

```
*** SYSTEM DISPLAY ***      03/01/84 (THU) 12:01:34
<RAM DISK> 007 kb      <AUTO START>
<USER BIOS> 000 256 b  <MCT MODE>      stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA      <MENU FILE> 1 .COM  2 .      3 .      4 .
- Select number or ESC to return.
<MENU> 1=off 2=on 3=drive 4=ext1 5=ext2 6=ext3 7=ext4
```

Fig. 2.23

Press the **2** key and note how the label "<MENU>" comes up on the top right of the screen on the first line. It can be switched off using option 1 at this level. Before actually displaying the MENU, it is necessary to select which drives the files are to be chosen from and also which types of files are to be chosen from those drives. Chapter 3 explains the types of files. As a first illustration of the use of the MENU, choose option 3 and type "BC" and press the **RETURN** key. Note that these letters appear after the label "<MENU DRIVE>" on the left of the fifth line and mean that files will be chosen from the B: and C: drives, i.e. the ROM drives. In order to select a type of file, take option 4 and type "COM" and press the **RETURN** key. Note how the fifth line has changed so that the first file name extension (next to the label "<MENU FILE>") has changed to "COM".

Now press the **ESC** key and the System Display returns to the first state. The MENU mode is set and the MENU screen will be displayed when the power switch is next turned on, or a warm start is made.

Turn off the power switch and then turn it on again. The speaker beeps and the following screen is displayed:

```
*** MENU screen ***      03/01/84 (THU) 12:00:52  54.5k CP/M  ver 2.2 PAGE 1/1
C:PIP
C:PIP      COM      C:STAT      COM      C:SUBMIT      COM      C:XSUB      COM
C:FILINK   COM      C:TERM      COM      C:CONFIG      COM      B:BASIC      COM
```

Fig. 2.24

The MENU screen is divided into three sections as shown below.

1 line	Header
1 line	Command line
6 lines	File name area

Fig. 2.25

#### Header section

The header section consists of the title "\*\*\* MENU screen \*\*\*", date, day of the week, time, CP/M comment, current menu page and number of menu pages. If the number of files' exceeds 120, "\*" will be displayed at the end of header section.

#### Command line

The command line section displays the currently selected file name, adding any COM file name if the file selected requires a COM file for execution. The System Display is used to decide exactly what is printed on this line for each file, and the method of setting it is explained in the next few pages.

#### File name area

The lower part of the screen shows all the file names set up on the System Display, chosen from the selected drives and the selected file extensions.

#### i) Using the MENU

The command line duplicates the name of the file in the top left of the file area and, to highlight the selected file, the name in the file area flashes. It is possible to change the selected file by using the cursor keys. Pressing the right arrow key moves to the right along a row, and then from left to right on the next

line. When the last filename is reached, the first one of the next line is selected to be displayed on the command line. The left arrow moves to the left in the same way. The up and down cursor keys allow movement in a column, but do not move into another column.

When a file name is on the command line, simply pressing the **RETURN** key will run that particular program. As an example, move the cursor keys until the file STAT is on the command line. As will be seen in Chapter 3, STAT is a program which gives STATistical information about the status of the computer and the disk drives. When STAT.COM (with its appropriate drive name prefix) is flashing, note that the drive name followed by STAT is printed on the command line. Press the **RETURN** key. The screen clears and the STAT program shows information such as the following:

```
A>C:STAT
A: R/W, Space: 8k
C: R/O, Space: 33k
H: R/W, Space: 29k

A>
```

Fig. 2.26

The second line of the screen shows the CP/M prompt and then the command line as would have been typed in had the STAT program been run directly from CP/M. This display shows a situation with the CP/M utilities in drive C: and thus the drive name C: would be displayed before the filename. After a few seconds, when the program has collected the information, the other lines are displayed. These show the space left on the various drives and whether they can be written to (R/W) or are read only (R/O). The CP/M prompt "A>" is then followed by the cursor.

**NOTE:**

*In this case ending the program has returned the user to the CP/M command line. Sometimes when a program ends it will return to the MENU. If the STAT program had returned you to the MENU, all the information would have been erased from the screen before it could have been read.*

Return to the MENU by using either the **STOP** key or **CTRL** - **C** and again place the STAT program on the command line. When the selected file has been placed on the command line, the cursor is placed to the right of the file name and is seen as a flashing underline character. This allows further parameters to be added from the keyboard. As an example of this use of the MENU, with

the STAT program on the command line, type in DEV: and then press the **RETURN** key. As is explained in Chapter 3, this shows the physical devices assigned to the four logical devices. As before the screen will clear and show the command as if it were typed in from the CP/M command line, before displaying the information a few seconds later. A possible display is shown in fig 2.27 below.

```
A>C:STAT DEV:
CON: is CRT:
RDR: is UR1:
PUN: is UP1:
LST: is LPT:

A>
```

Fig. 2.27

*ii) Setting the file extensions on the System Display*

In setting the file extensions on the System Display care has to be taken in setting up files which are to be run as subsidiary files of a main COM file. The most common case of this is the use of BASIC files with the extension ".BAS". Other examples of extensions to filenames which denote subsidiary files are ".OVR" and ".DAT". Attempting to run these files without the appropriate COM file will result in an error. As an example of setting up the MENU to run subsidiary files, the following procedure should be used to set up BASIC files.

Enter the System Display and choose option 4, the MENU option.

Now edit the second extension by choosing option 5.

To specify the subsidiary file type its extension (for BASIC files the extension is ".BAS") followed by the filename of the COM file of which it is a subsidiary files. This is the reverse of normal CP/M conventions, but is necessary to display the extension correctly on line 5 of the System Display. Lines 6 and 7 would thus display:

```

*** SYSTEM DISPLAY ***      03/01/84 (THU) 12:01:34      <MENU>
<RAM DISK> 007 kb      <AUTO START>
<USER BIOS> 000 256 b  <MCT MODE> stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA      <MENU FILE> 1 .COM 2 . 3 . 4 .
- Input extension name 2, ESC to cancel.
  BAS

```

Fig. 2.28

When the **RETURN** key is pressed, the right hand side of the fifth line will show

```
<MENU FILE> 1 .COM 2 .BAS 3 . 4 .
```

However, if a file name is chosen with the extension ".BAS", for example "a:ZZZZ.BAS" the command line will show

```
b:basic a:ZZZZ.BAS
```

and when the **RETURN** key is pressed, BASIC will be loaded first and then the subsidiary file run.

**WARNING:**  
 If the MENU is used to run a program this way, since the BASIC interpreter is loaded as well, any programs which are in the five program areas will be lost.  
 A way to overcome this is described in the following section.

iii) Using BASIC from the MENU

Details of using BASIC on the PX-8 are given in the BASIC Reference Manual. When BASIC has been loaded and the PX-8 is switched off the first file from the disk drives will not be flashing. Instead the file area will show "BASIC (resident)" in the top left corner which will be flashing, and the command line will be blank. Simply pressing **RETURN** at this stage will cause the BASIC Program Menu to be turned on. Any programs in the five program areas can be run as described in the BASIC Reference Manual.

If BASIC is resident, the above method of selecting a BASIC program from the MENU will load the BASIC interpreter and destroy the programs in the five BASIC program areas. The following method will allow a BASIC program to be selected from the MENU and run while BASIC is resident.

- i) Use either the MENU or the CP/M command line to load BASIC.

- ii) Login to one of the program areas and type "system" or use PF8 (shifted PF3) to return to the CP/M command line or the MENU.
- iii) Either on the MENU or CP/M command line type:

SAVE # A:GO.COM

The reason for doing this is explained in Chapter 3 under the SAVE command.

- iv) Enter the System Display and using option 5 on the MENU, alter the extension to read:

BAS A:GO

This uses the COM file "GO" as the main file, with the BAS file as the subsidiary file.

When a file with the "BAS" extension is chosen from the MENU, if AND ONLY if BASIC IS RESIDENT, the MENU command line will show:

A:GO A:ZZZZ.BAS

when the file "A:ZZZZ.BAS" is chosen from the menu, and the program will run directly in BASIC in program area one.

Note however, that there is no need to resort to this method if BASIC is resident and the WAKE or AUTO START strings are used.

The MENU module is activated under one of the following conditions:

- 1) The MENU mode is specified and the AUTO START string is not specified in the System Display, and power is turned on by the power switch or when the wake time is reached.
- 2) The **STOP** or **CTRL** and **C** keys are pressed together when an application program which has been entered from the menu mode is being executed or the "SYSTEM" statement is executed in the BASIC mode.

Operation of the MENU module is interrupted in one of the following cases.

- 1) When the alarm or wake time is reached.
- 2) When the **CTRL** and **HELP** keys are pressed together. (The system module is activated.)
- 3) When the **CTRL** and **PF5** keys are pressed together. (The screen dump module is activated.)

The MENU module is started at the state at which it was interrupted when the MENU mode is returned to from one of the above conditions.

The MENU mode is terminated in one of the following cases:

- 1) When an application program is selected and the **RETURN** key is pressed.
- 2) When the AUTO POWER OFF time is reached.
- 3) When power failure is detected.
- 4) When the power switch is turned off.
- 5) When the **ESC** key is pressed. (The screen clears and the CP/M system prompt appears on the screen.)

#### e) Microcassette mode

The System Display is used to operate the Microcassette drive manually and also to check on the Microcassette mode settings. Full information on the Microcassette drive is given in Chapter 4.

##### i) Setting the Microcassette mode

To specify the Microcassette mode, choose option 5 on the System Display by pressing the **5** key.

Lines 6 and 7 of the screen then change to:

```
*** SYSTEM DISPLAY ***      03/01/84 (THU) 12:01:34      <MENU>
<RAM  DISK> 009 kb      <AUTO START>
<USER  BIOS> 000 256 b  <MCT  MODE> stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA      <MENU FILE> 1 .COM  2 .BAS  3 .  4 .
- Select number or ESC to return.
<MCT> 1=stop 2=nonstop 3=verify 4=nonverify
```

Fig. 2.29

The stop and nonverify modes are automatically set after system initialization. Pressing the **1** key sets the stop mode, pressing the **2** key sets the nonstop mode, pressing the **3** key sets the verify mode and pressing the **4** key sets the nonverify mode. Full details of the significance of these terms are described in Chapter 4 where the operation of the Microcassette drive is covered in depth.

The current settings for the Microcassette operation are shown on the right of the fourth line of the System Display, following the label “<MCT MODE>”. This line also shows the position of the tape counter, following the “<COUNT>” label.

To return to the screen shown in Fig. 2.13b, press the **ESC** key.

##### ii) Manual operation of Microcassette drive

The System Display is also used to allow the Programmable Function keys to operate the Microcassette drive manually. The function assigned to each of these keys is indicated on the 8th line of the system display shown in Fig. 2.13b.

Two of the keys are assigned for fast winding of the tape forward and backwards. These are the **PF1** and **PF4** keys which wind the tape on (<<-) and rewind the tape on (->>) respectively.

The **PF2** key is used to play the tape at normal speed through the speaker.



The **PF7** (shifted **PF2**) key is used to mount a tape, i.e. make it ready to be read from or written to by reading the directory into memory. When the tape is used first, the **PF8** (shifted **PF3**) key can be used to initialize the tape by storing space for a directory. It is equivalent to formatting a conventional disk. The **PF9** (shifted **PF4**) key is used to erase data from a tape. It does this from the current position of the tape.

The **PF5** key is used to reset the tape counter to zero. The current position of the counter can be seen on the right of line 5 of the System Display next to the "<COUNT>" label.

The **PF3** key marked with a ■ character is the key to stop the cassette.

TO STOP THE MICROCASSETTE IN AN EMERGENCY press the **CTRL** and **STOP** keys together.

If a Microcassette has been mounted, the only key which is assigned is **PF6** (shifted **PF1**). This will show the string "remove" which will allow the tape to be removed from the cassette drive. It is worth getting into the habit of checking the status of the cassette tape in the Microcassette drive, by inspecting the System Display BEFORE removing a tape from the drive. If a tape has been mounted, the eighth line will only show the word remove.

If a tape has been mounted, when the **PF6** (shifted **PF1**) key is pressed, the screen will clear and show the message "remove" in the top left hand corner of the screen. If the tape has simply been read this will flash up momentarily, and the System Display will be redisplayed but with the eighth line showing the assignments for an unmounted tape.

If the tape has been written to, the revised directory on the tape will have to be written back on to the tape from memory. The tape will thus wind back to the beginning, and then record the updated directory. When this has been carried out, the System Display will be redisplayed but with the eighth line showing the assignments for an unmounted tape.



**WARNING:**

*Do not change a tape and attempt to write to another one, without first removing the first tape and mounting the new one. Failure to observe this will almost certainly mean that data on both tapes will not be able to be read, and may also remove valuable data from the second unmounted tape.*

**f) The screen dump module**

The screen dump module outputs the contents of display screen to a printer. It is activated by pressing the **PF5** key together with the **CTRL** key or calling the BIOS SCRNDUMP routine in CP/M mode. In screen modes 0,1 and 2 data is sent to the printer in ASCII format, and in screen mode 3 in bit image format.

Printing stops when the **CTRL** and **STOP** keys are pressed simultaneously, the power switch is turned off or power failure is detected.

If the printer is not ready, the module waits until it becomes ready.

**NOTE:**

*If there is no printer attached to the PX-8, this creates a lock-up of the system. If this happens, press **CTRL** and **STOP** keys together. User can resume his session with no loss of data.*

## 2.2.4 Alternative reset sequences

Situations occasionally arise where the PX-8 ceases to respond to any instructions from the keyboard when you expect it would. Even the **STOP** key does not appear to work. The computer is said to be "hanging". This should not be confused with the situation where a program is processing data and takes seconds or even a minute to reach a situation where the user can input data again. When the system appears to have hung, it is sometimes possible to gain control by switching off, waiting a few seconds and switching on again. However, in most cases it is necessary to RESET the computer. There are a number of methods of resetting, which affect the system parameters to a different extent.

### a) SUB-CPU RESET

This is the most drastic reset procedure and is carried out as described in section 2.1.3. It must be carried out if the battery is changed, or if all other reset methods fail.

### b) INITIALIZATION RESET

This reset sequence is achieved by holding down the right-hand **SHIFT** and **NUM GRAPH** keys while pressing the reset switch on the left of the PX-8 using the tip of a ball point pen or other similar implement. See fig. 1.1c for the position of the reset switch.

This method of resetting, is for practical purposes as drastic as the sub-cpu reset and will cause loss of the current program, data and possibly destroy the RAM disk.

When this reset method is used, the initialization procedure outlined in section 2.1.3 from item v) onwards will have to be carried out.

### c) SIMPLE RESET

The least drastic method of resetting the PX-8 is to simply press the reset switch on the side of the computer using the tip of a ball point pen etc. In most cases very little harm will have been done to data, and most if not all variables in a program will probably still be intact. A number of system parameters will be changed and details of these are shown in table 2.3.

This method of resetting should be tried first.

### d) Summary of RESET actions

In resetting the computer it is as well to be aware of the parameters which are altered by the reset action. The following table shows which settings are altered. Since these are normally the default settings of the system, these are shown also.

Table 2.3

RESETPARAMETER	DEFAULT	SIMPLE RESET	NUM/GRPH RESET	SUB-CPU RESET	CHANGE WITH
ALARM/WAKE	off	-	d	d	s,b
AUTO POWER OFF	10 mins	-	d	d	c,b
AUTO START	off	-	d	d	s,b
CHARACTER SET	DIP switch setting	d	d	d	
CURRENT DRIVE	A:	d	d	d	NOTE 8
CURSOR TYPE	flashing block	d	d	d	c,b
DATE/TIME	NOTE 1	-	u	u	c,b
DISK ASSIGNMENTS	NOTE 2	d	d	d	c
PF KEY DISPLAY	off	d	d	d	c,b
PF KEY STRINGS	NOTE 3	d	d	d	c,b
I/O BYTE (03H)	10101001B	d	d	d	NOTE 9
KEYBOARD	NOTE 4	d	d	d	NOTE 4
MENU	on	-	d	d	s
MENU DRIVES	CBA	-	d	d	s
MENU FILE EXT	.COM	-	d	d	s
MICROCASSETTE COUNT		-	destroyed	destroyed	s,b
STATE	unmounted	d	d	d	s,b
STOP MODE	stop	d	d	d	s,b
VERIFY MODE	non-verify	d	d	d	s,b
PASSWORD	off	d	d	d	s
PRINTER	RS232C	d	d	d	c
RAM DISK SIZE	9K	-	u	u	c
RS232C	NOTE 5	d	d	d	c
SCREEN MODE	mode 0	d	d	d	c,b
SERIAL INTERFACE	NOTE 6	d	d	d	c
USER BIOS SIZE	0 pages	-	u	u	c
USER DEFINED CHARS					
E0,E1	del, cr symbols	d	d	d	NOTE 7
E2 - FE	NOTE 7	-	-	-	
WINDOW	tracking mode	d	d	d	s,c,b

#### KEY TO abbreviations:

d = default mode    u = set by user on reset  
s = System Display    c = CONFIG program    b = BASIC

\* Some functions can also be changed using ESCAPE code sequences as described in Appendix A.

**NOTE 1:**

The DATE and time are set to the default 00/00/00 00:00:00 with the day as zero (Sunday), and the user is asked to change them on reset.

**NOTE 2:**

The disk assignments for the default mode are shown in the explanation of the CONFIG program in Chapter 3.

**NOTE 3:**

The default Programmable Function strings are shown in the explanation of the CONFIG program in Chapter 3, and in section 2.2.1 above.

**NOTE 4:**

The default setting of the keyboard has the auto repeat start time set at 650 ms and the repeat interval at 75 ms. The keyboard is normal, in that all CAPS LOCKS etc are switched off. Any jumps to user functions using the **CTRL** keys and the **ESC**, **PAUSE** and **PF1** to **PF4** keys are set to a return.

**NOTE 5:**

The default RS232C settings are shown under the CONFIG program of Chapter 3.

**NOTE 6:**

The default setting of the Serial Interface are given in the explanation of the use of the CONFIG program in Chapter 3.

**NOTE 7:**

The User Defined Characters ASCII codes 224 to 254 (hexadecimal E0 to FE) can be set using control code sequences as described in Appendix A. They can also be set from BASIC, and a program to do this is shown in Appendix H of the BASIC Reference Manual.

The first two characters 224 and 225 (E0H and E1H) have the default symbols for delete "Δ" and a carriage return assigned to them. These characters are also frequently reset to the defaults at other times for example on power up. The character defined into ASCII code 226 (E2H) may also be corrupted on a reset.

**NOTE 8:**

The current drive is set by the applications program or from the CP/M command line.

**NOTE 9:**

The use of the I/O byte is described in Chapters 3 and 5.

**Priority of system reset**

When a lock-up of the system occurs, reset the system according to the following priority.

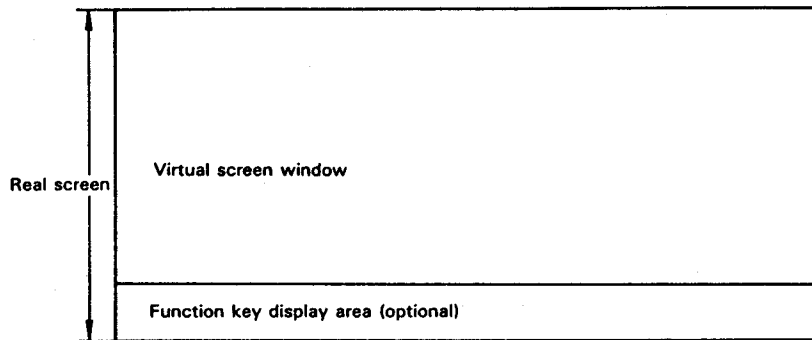
1. Press **STOP** key.
2. Press **CTRL** - **STOP** keys.
3. Press Reset switch on the left hand.
4. Press Reset switch while holding down **SHIFT** - **NUM GRAPH** keys.  
(or press Sub CPU Reset switch)

**2.2.5 Changing parameters**

For a description of how to change the parameters see the CONFIG program in Chapter 3, and section 2.2.3a. Details of which parameters are changed by the CONFIG program and which by the System Display can be found from table 2.2.

## 2.2.6 The Screen of the PX-8

When you view the LCD screen of the PX-8, you are looking at a window on a much larger screen. The screen displayed on the LCD is known as the real screen. This is 8 lines high by 80 columns. The operating system works with a much larger screen, a screen of up to 40 lines and 80 columns. The real screen then displays a window on this larger screen (the virtual screen), which is called the virtual screen window.



There are also four different screen modes (including a graphics mode) which show different types of display. In all but the graphics mode there are two virtual screens.

This section outlines the different screen modes and shows how to use them. Changing between different screen modes can be achieved using the CONFIG program described in Chapter 3. It is also possible to change the screen modes using BASIC commands, and the BASIC Reference Manual contains a practical guide to the screen modes in section 2.14.

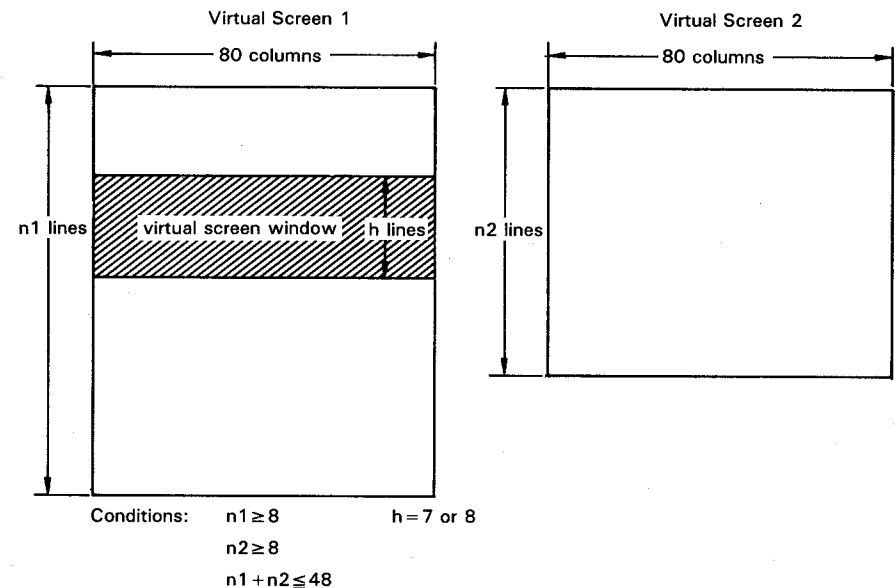
It is unlikely that you will use the different screen modes in the CP/M environment unless a particular applications program has made use of them. However, there are certain benefits to be obtained by learning to use the tracking and non-tracking modes, and to understand the virtual screens. As these would be utilised mainly in screen mode 0, a detailed explanation is given in the following outline of this screen mode.

## i) The Screen Modes

There are four screen modes possible with PX-8. Three of them are text only screens, and the fourth is a mixed text and graphics mode. The difference between the modes is primarily the way the real screen presents the information written on the virtual screens.

### a) Screen mode 0 (the 80 column text screen mode)

This screen mode has two virtual screens each with 80 columns. The number lines on these screens can be set by the user provided that the sum of the number of lines does not exceed 48 and there are at least 8 lines (i.e. one real screenful of lines) on each screen.



The virtual screen window moves over the virtual screen to display a part of the virtual screen on the real screen. This movement is known as scrolling. When the real screen moves over the virtual screen, scrolling with the cursor, this is known as tracking mode. Tracking mode can be switched off by pressing the **SCRN** key (**SHIFT** + **INS**). When this is done the cursor moves over the virtual screen while the real screen stays at the same place on the virtual screen. The real screen becomes locked in a particular position on the virtual screen.

The following illustration will show the effect of scrolling in the tracking and non-tracking modes. It assumes that the virtual screen size of 24 lines on each

screen has been set. The CONFIG program described in Chapter 3 can be used to check this and reset it if necessary.

Go to the System Display and switch off the MENU. Exit from the System Display to display the CP/M command line.

If the screen is not clear, it may be cleared by pressing the **CLR** key (**SHIFT** + **DEL**), or typing CTRL-L by holding down the **CTRL** key while pressing the "**L**" key. In either case the display will show "**^L**" next to the CP/M system prompt. Thus if the currently logged in drive is A:, the following will be displayed:

```
A>^L
```

When the **RETURN** key is pressed, the screen will clear and show a question mark in the top left hand corner. This is because CP/M does not understand **CTRL** - **L**, as a command. After printing a blank line the system prompt is printed again, with the cursor to the right of it.

Now obtain a directory of the utility ROM by typing "DIR C:" if the ROM is in ROM socket 2, or "DIR B:" if it is in ROM socket 1. For the purposes of this illustration it will be assumed that it is in ROM socket 2 as supplied.

The display will show:

```
?
A>dir c:
C: PIP          COM : STAT      COM : SUBMIT  COM : XSUB    COM
C: FILINK      COM : TERM      COM : CONFIG  COM
A>█
```

Because only two lines and a new system prompt have been written on the screen, the screen has not needed to scroll. However if the command is typed again, the "?" will scroll off the top of the real screen, and show the following:

```
A>dir c:
C: PIP          COM : STAT      COM : SUBMIT  COM : XSUB    COM
C: FILINK      COM : TERM      COM : CONFIG  COM
A>dir c:
C: PIP          COM : STAT      COM : SUBMIT  COM : XSUB    COM
C: FILINK      COM : TERM      COM : CONFIG  COM
A>█
```

The "?" is still the first character on the virtual screen. This can be seen in one of two ways. Pressing the **CTRL** and **↑** cursor key will display the first lines of the virtual screen because these keys move the real screen backwards one page (i.e. one real screen) on the virtual screen. Pressing the space bar (or any other key) will return to the original display. Alternatively the screen can be scrolled up by pressing the **SHIFT** and **↓** cursor key. Again pressing the space bar or another key will return the real screen to show the cursor.

To show the effect of this sequence of operations in non-tracking mode, first clear the screen again using **CTRL** - **L** or the **CLR** key. Then press the **SCRN** key (**SHIFT** + **INS**). Nothing will appear to happen. Now obtain two directories of the utility ROM in drive C: by typing "DIR C:" as before. The screen will show:

```
?
A>dir c:
C: PIP          COM : STAT      COM : SUBMIT  COM : XSUB    COM
C: FILINK      COM : TERM      COM : CONFIG  COM
A>dir c:
C: PIP          COM : STAT      COM : SUBMIT  COM : XSUB    COM
C: FILINK      COM : TERM      COM : CONFIG  COM
```

Note how the "?" on the first line remains visible in contrast to the same sequence of operations in the tracking mode. This time the cursor and CP/M system prompt has disappeared off the real screen. They are further down the virtual screen just outside the window. They can be seen by pressing the **SHIFT** + **↓** cursor key to move the screen up. This leaves the real screen window locked on these particular eight lines, so that typing "DIR C:" again will write the directory outside the real screen. Pressing the **SCRN** key again will cause the tracking mode to be restored and the screen will move up to show the cursor. When returning to tracking mode the cursor is positioned in the centre of the screen and so the screen will appear as follows:

```
A>dir c:
C: PIP          COM : STAT      COM : SUBMIT  COM : XSUB    COM
C: FILINK      COM : TERM      COM : CONFIG  COM
A>█
```

Now press the **RETURN** key ten times. This will almost reach the bottom of the virtual screen. Press the **CTRL** and **↑** key to move to the bottom of the virtual screen. Note that there are still two free lines of the virtual screen with nothing printed on them.

Move to the top of the virtual screen by pressing the **CTRL** and **↑** key twice. Note that the first character the “?” is still present. Lock the screen into non-tracking mode by means of the **SCRN** key. Now type “DIR C:” again. This will write the directory of disk C: on the bottom of the virtual screen next to the cursor, and then print a new CP/M system prompt. However, as there are not enough lines on the virtual screen the whole virtual screen must scroll up one line. This causes the window at the top of the screen to change. The line containing the “?” character is pushed off the cap of the virtual screen, and lost forever. The top 8 lines of the virtual screen are still displayed on the real screen, but they are 8 new lines now that scrolling has occurred. Return to tracking mode by pressing the **SCRN** key again and the bottom of the virtual screen, containing the cursor will be shown, with the directory of drive C: and the new CP/M system prompt.

So far only one virtual screen has been used. It is possible to move onto the second virtual screen by using the **CTRL** and **→** key, and back to the first virtual screen using the **CTRL** and **←** key. The two screens can be written to independantly, although there is no guarantee that all applications software will allow you to use the second virtual screen or scroll or go into non-tracking mode.

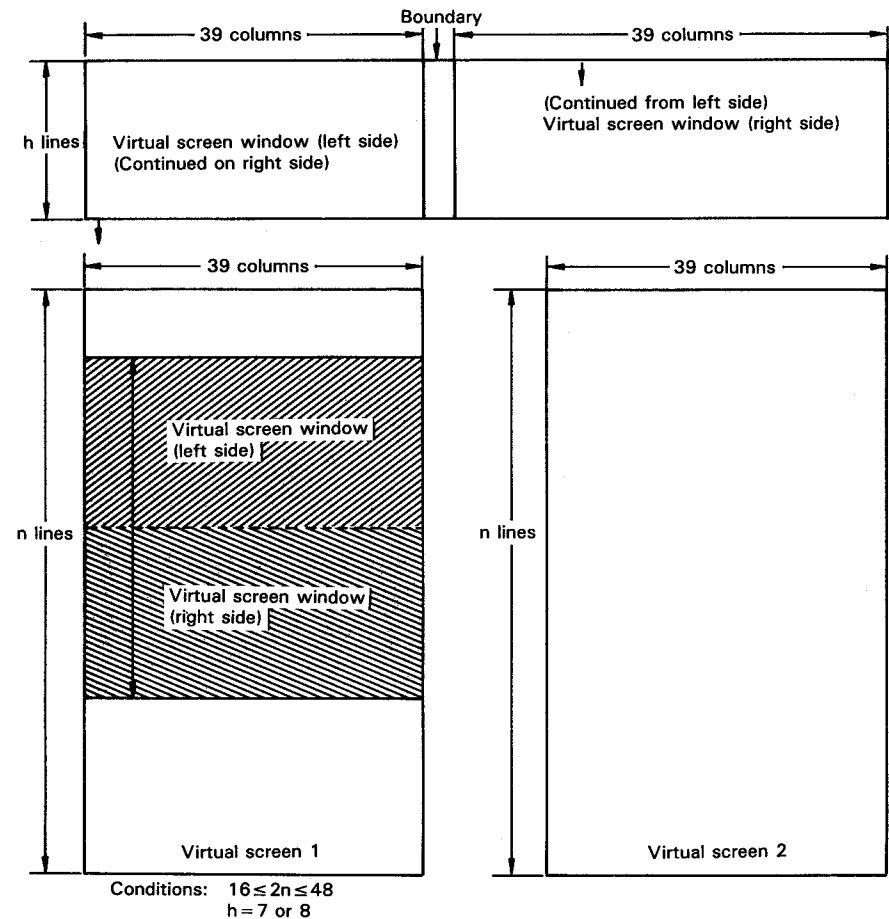
Switch to the second virtual screen using the **CTRL** and **→** keys. The screen should be blank. Type “DIR B:” and contents of the directory of the B: drive will be printed. If you switch back to the first virtual screen using the **CTRL** and **←** keys, the screen will be exactly as you left it. Clear it with a **CTRL** - **L** or by using the **CLR** key and then go back to virtual screen 2 using the **CTRL** and **→** key. Note that the display has not been changed.

One use of the two virtual screens is to store data. For example, frequently one needs to know the contents of the directory of a disk. Go to virtual screen 2, and print the directory of the disk there. If the program will allow it (and the CP/M utilities such as PIP will) you can easily look at the second screen to inspect the directory by using the **CTRL** and **→** keys, and then return to the first virtual screen to continue with the program. In both cases the previously displayed screen, normally where the cursor lies, will appear in the real screen area.

It is not advisable to enter a program command on the second virtual screen if the previous command was executed on the first virtual screen.

**b) Screen Mode 1 (39 column Split screen text mode)**

This screen mode splits the real screen into two halves, each of 39 columns with a boundary of two characters in the centre of the screen. There are two virtual screens but they must both have the same number of lines. The number of lines in the virtual screens must be in the range 16 to 48. Only one virtual screen can be displayed at a time. The real screen can be thought of as a screen of 39 columns, 16 lines long which has been split into two blocks of eight lines which are displayed on the two halves of the real screen. The bottom of the left half of the screen is continued on the top of the right half of the screen.



If the sequence of clearing the screen and printing two directories of the C: drive is performed with screen 1 turned on (it can be chosen by using the CONFIG program described in Chapter 3), then the screen will show the following output.

```

A>dir c:
C: PIP      COM : STAT      COM : SUBMIT
   COM : XSUB      COM
C: FILINK  COM : TERM      COM : CONFIG
   COM
A>dir c:
C: PIP      COM : STAT      COM : SUBMIT
   COM : XSUB      COM
C: FILINK  COM : TERM      COM : CONFIG
   COM
A>
COM : XSUB      COM
C: FILINK  COM : TERM      COM : CONFIG
COM
A>

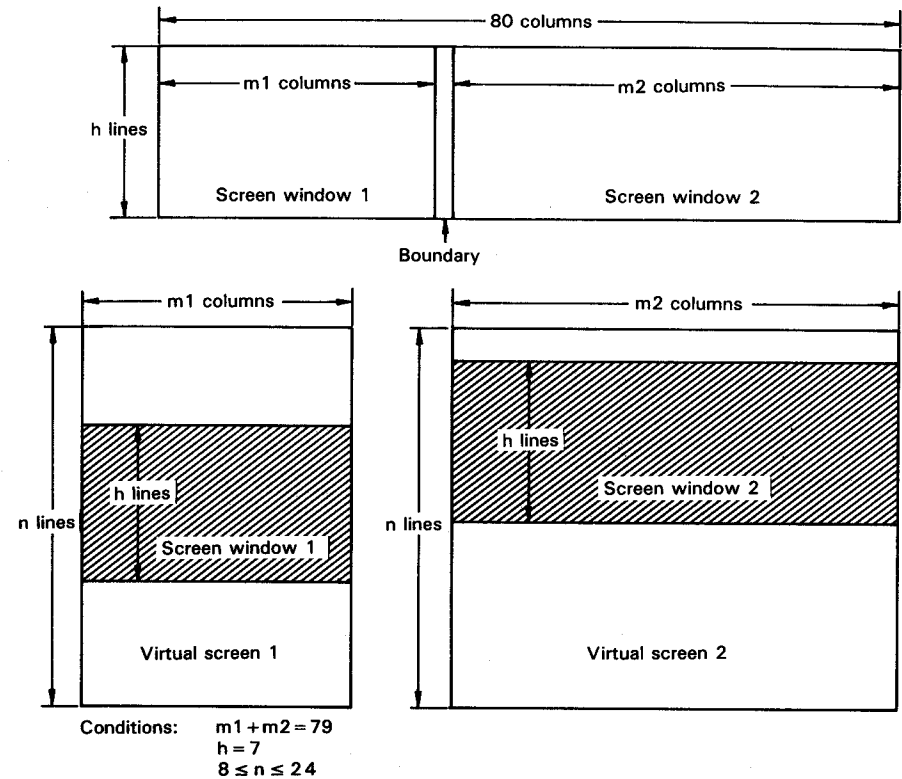
```

Tracking and non-tracking modes can be set with the SCRNL key, and the two virtual screens can be displayed using the **CTRL** and **→** and **CTRL** and **←** keys just as with screen mode 0.

### c) Screen Mode 2 (the Dual screen mode)

In this screen mode, the real screen is divided into two halves. There are two virtual screens, and each of the virtual screens are displayed in the two halves of the real screen. While the contents of the two screens can be seen at the same time, they are also independent of one another, and so can be scrolled separately. They cannot be set in the tracking and non-tracking mode independently.

The number of lines of the two virtual screens must be in the range 8 to 48, and must be the same in both virtual screens. The number of columns and the boundary character can be set by the user. The total number of columns must equal 79, and there must be at least one column in one of the screens. The CONFIG program described in Chapter 3, can be used to set these parameters.

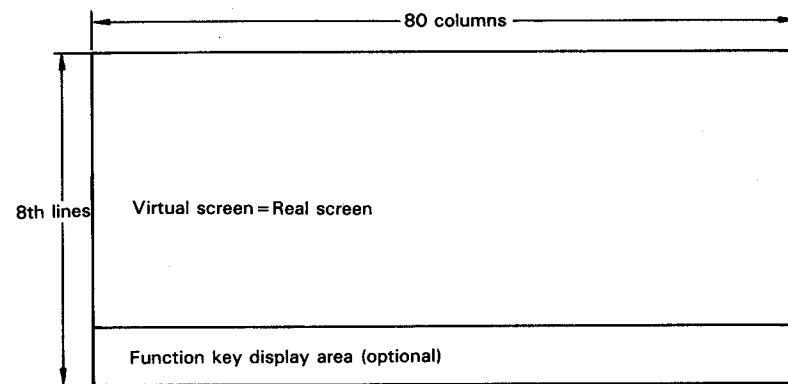


### d) Screen Mode 3 (the graphic screen mode)

This mode enables graphics to be displayed. Text can also be displayed on this screen. There is only one virtual screen whose size is the same as the size of the real screen. Thus scrolling the real screen over the virtual screen and setting the tracking and non-tracking modes do not apply.

## Chapter 3

# OPERATING THE COMPUTER UNDER CP/M



The screen allows individual dots of the screen to be lit (bit image mode). This is only possible by means of software. The most convenient way to understand this mode is to use BASIC. If this mode is used other than in BASIC, it will be with special applications software and the appropriate manual should be consulted for its use.

Chapter 2 discussed the operation of the computer at the simplest level. This chapter deals with the operation on a day to day level. It mainly covers using the operating system CP/M, housekeeping of files and using the utility and applications programs. For a complete guide to the operation of an applications program, see the appropriate manual for that program.

### 3.1 What Is CP/M?

CP/M is the most popular operating system for microcomputers. An operating system is a collection of computer programs which have been assembled to make it easy for the user to run programs, handle saving and loading of data and generally allow easy transfer of information between software and hardware. The popularity of CP/M is due to the fact that it can be used on a large number of machines.

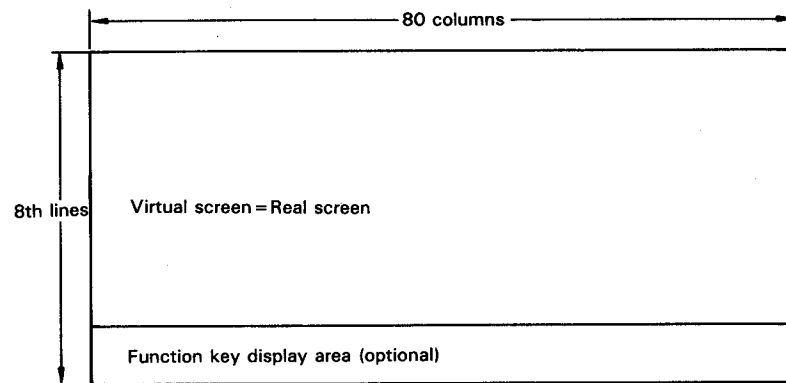
To understand what an operating system does, consider the analogy of a telephone system linked to the switchboard of a large company. If anyone dials into the company the public telephone network handles the connection to the company. The switchboard in the company then directs the caller to the person to which he wishes to speak. Similarly, if a person inside the company wishes to call out, he again goes through the switchboard to reach the public telephone system.

CP/M is very much like this except that there are a number of 'switchboards' for different parts of the hardware, e.g. the disk, Microcassette and devices such as the keyboard and screen. Anyone writing software for a CP/M based machine only has to use the equivalent of the public telephone system in this analogy. This means that having written a program such as a word-processor, it requires little if any further programming to allow the program to be run on a computer other than the one for which it was originally designed, providing both computers use the CP/M operating system. All interfaces between the



## Chapter 3

# OPERATING THE COMPUTER UNDER CP/M



The screen allows individual dots of the screen to be lit (bit image mode). This is only possible by means of software. The most convenient way to understand this mode is to use BASIC. If this mode is used other than in BASIC, it will be with special applications software and the appropriate manual should be consulted for its use.

Chapter 2 discussed the operation of the computer at the simplest level. This chapter deals with the operation on a day to day level. It mainly covers using the operating system CP/M, housekeeping of files and using the utility and applications programs. For a complete guide to the operation of an applications program, see the appropriate manual for that program.

### 3.1 What Is CP/M?

CP/M is the most popular operating system for microcomputers. An operating system is a collection of computer programs which have been assembled to make it easy for the user to run programs, handle saving and loading of data and generally allow easy transfer of information between software and hardware. The popularity of CP/M is due to the fact that it can be used on a large number of machines.

To understand what an operating system does, consider the analogy of a telephone system linked to the switchboard of a large company. If anyone dials into the company the public telephone network handles the connection to the company. The switchboard in the company then directs the caller to the person to which he wishes to speak. Similarly, if a person inside the company wishes to call out, he again goes through the switchboard to reach the public telephone system.

CP/M is very much like this except that there are a number of 'switchboards' for different parts of the hardware, e.g. the disk, Microcassette and devices such as the keyboard and screen. Anyone writing software for a CP/M based machine only has to use the equivalent of the public telephone system in this analogy. This means that having written a program such as a word-processor, it requires little if any further programming to allow the program to be run on a computer other than the one for which it was originally designed, providing both computers use the CP/M operating system. All interfaces between the

software and hardware are handled by the different versions of CP/M written for the different computers. This is the reason why CP/M has been chosen as the operating system for the PX-8. The huge library of CP/M based software can be installed on the PX-8, making it easy for the user to find the particular application he requires. This increases the power and versatility of the computer since it is only a machine to run software - a tool to extend the mind much as a hammer is a tool to extend the body.

If you are a beginner to computing, CP/M can be a little frightening. You will find that the terms used to communicate ideas to the computer are not as near to normal human communication as one would like. There are a number of jargon words which will have to be learnt. This chapter has been written in such a way that as far as possible you only have to think about the current task. If necessary, use the index to direct you to the appropriate part of the chapter to find out about anything you do not understand. Any difficulties you experience will disappear with practice, and you will find using CP/M will soon be second nature to you.

The only way to learn to use CP/M is to try out the examples given. This manual cannot hope to cover the finer points of the system, but fortunately there are a number of books which describe all aspects of CP/M at all levels, from the absolute beginner to the most advanced user. A list of some of these are given at the end of Chapter 3.

## 3.2 Files And File Names

In a standard CP/M system each drive holds floppy disks which are used to store programs. The PX-8 has some drives which do not contain floppy disks. For the purposes of using them under CP/M, the RAM disks, Microcassette drive and the ROM drives behave as if they were standard disk drives. There are a few specific differences, e.g. the ROM drives cannot be used to store information. The use of the Microcassette drive is described in Chapter 4.

The programs and data are stored in files. A file is simply a collection of information which is related as a unit. Files fall into two main categories. Program files are a set of instructions, for example a wordprocessing program. Data files store information which is used by a program, for example text used by the wordprocessor or the information used by an accounts program. Files are stored by name. To make it easier to understand which files are which, the name is divided into two parts. The first part consists of up to eight characters which are used to give a distinctive name to the file.

All of the printable characters can be used except:

< > , . ; : = ? \* [ ] and the space character

The secondary part of the name is used to describe what type of file is stored. Secondary names are frequently referred to as file types or extensions. This part of the name is used in the MENU to select the type of file to be displayed on the MENU in the file name area. It is possible to store a file without an extension, but this can cause problems. Applications programs frequently add the extension to the primary name you have given the file so that you know which files belongs to that program. The extension is separated from the primary filename by a full stop.

Examples of file names with extensions are:

**BASIC.COM**  
**STAT.COM**  
**SAMPLE.ASM**  
**MEMO.DAT**

Note that all filenames are given in upper case letters. Even if you type them in lower case, they are stored on the disk as upper case. If you are using the filename, CP/M will understand if you type the file you want in lower case or even a mixture of upper and lower case. Note also that the separator in front of the extension is printed next to the primary file name. This may not always

be so if the computer prints it out.

Some extensions are standardised and are recognised by CP/M or particular programming languages or applications languages.

Examples of common extensions are:

<b>.ASM</b>	ASseMbly language source file
<b>.BAK</b>	BACkUp file
<b>.BAS</b>	BASIC program source file
<b>.COM</b>	Directly executable COMmand file
<b>.DAT</b>	DATA file
<b>.DOC</b>	DOCument file
<b>.MSG</b>	A MeSsaGe file
<b>.OBJ</b>	OBJect code for machine code program
<b>.SUB</b>	Command file for use with the SUBMIT program
<b>.TXT</b>	TeXT file
<b>.\$\$\$</b>	Temporary file

Some terms used in this list which are commonly met are:

**SOURCE CODE.** This is text which is converted or interpreted by a program to create machine readable codes. It is in a form which is understandable provided you can understand the language it was written in.

**OBJECT CODE.** This is code directly readable by the computer as a set of instructions.

**COMMAND FILE or COM FILE.** This is the most important file you will meet under the CP/M operating system. It is basically a program file which is loaded into memory to execute a set of instructions. There are a number of commands built in to CP/M. A COM file is sometimes known as a TRANSIENT COMMAND since it is temporarily loaded into the TRANSIENT PROGRAM AREA or TPA, i.e. the part of the computer's memory set aside to run programs.

### 3.2.1 Referring to file names - (ambiguous/unambiguous file names and wildcards)

In many cases CP/M or a program requires you to use the complete, exact file name. In this case there is no doubt which file is being referred to. The name is UNAMBIGUOUS as there is only one file of that name. Sometimes you may

wish to refer to several similar file names. For example they may be all the files beginning with the letter "E" or all the files with extension ".COM". In this case the file name is AMBIGUOUS. In references to CP/M you will frequently see these terms abbreviated:

ufn	means	Unambiguous File Name
afn	means	Ambiguous File Name

In using ambiguous file names, you need to tell CP/M some further information, so that it can have a precise idea what you mean. In order to do this you can use the "?" characters to denote "any character" and the "\*" to denote "any group of characters". The "?" and "\*" characters are known as WILDCARD characters.

For example:

<b>*.COM</b>	means any file name with the extension .COM
<b>NAME.*</b>	means all the files with the primary name "NAME" but with any extension, eg. NAME.COM, NAME.BAS, NAME.DAT
<b>E*.COM</b>	means any file name beginning with E which has the extension .COM
<b>E*.*</b>	means all files beginning with E no matter what their extension



**Warning:**  
you cannot use:

**\*P.\*** to mean all the files ending in the letter "P". CP/M will treat this as meaning every single file. This is important to remember if you are erasing files from a disk.

<b>?????????.COM</b>	means any file with the extension .COM because there are eight (i.e. the maximum) wildcard characters.
<b>?????.COM</b>	means any file of not more than four characters in length, which has the extension .COM.
<b>A???.*</b>	means any file beginning with the letter "A" which has not more than four characters.
<b>?S???.BAS</b>	means any file whose second character is "S", which has five or less characters, and has the .BAS extension.
<b>?????ROP.*</b>	means any file ending in the characters "ROP".
<b>*.B??</b>	means all files whose extensions begin with the character "B".

### 3.3 Starting To Use CP/M

Although the MENU is an aid to using CP/M, it is not a replacement. To use the MENU effectively, a working knowledge of CP/M is required. An understanding of how to use CP/M is also required to make effective use of the CP/M utility programs.

#### 3.3.1 Disk drive names

Switch off the MENU from the System Display (section 2.2.3) and the screen will clear to show the system prompt "A>". All drives are given an identification letter. It is possible to change the relationship between the actual physical drives and the identifying letter on some of the drives. This can be done with the CONFIG program described in section 3.8. When you initialize your PX-8 the drives are specified as follows:

Identifying letter	Physical Drive
A:	RAM disk
B:	ROM 1
C:	ROM 2
D:	Floppy Disk Drive 1
E:	Floppy Disk Drive 2
F:	Floppy Disk Drive 3
G:	Floppy Disk Drive 4
H:	MICROCASSETTE DRIVE

The default drive is set to drive A: which is why the system prompt said "A>". Notice that apart from the system prompt the drive name is followed by a colon(:). Since file names and commands consist of letters, CP/M has to have some means of recognising when you are talking about a disk drive identifier. The colon is always used in CP/M to denote a drive. The system prompt always includes the default or currently "logged in" drive as a reminder to you which drive is currently logged in. Unless you specify another drive, all access to drives (saving and loading or simply inspecting the contents of the drive) will be made to that drive. To use the currently logged in drive, you need not type in its identifier.

This can be seen with the following sequence of operations:

Type the command DIR next to the system prompt and press the **RETURN** key. This means "show me all the files stored on the currently logged in drive". If there are no files on the disk the screen will show:

```
A>DIR
NO FILE
A>
```

You can specify another drive by adding the drive name after the DIR command as follows when the screen will show:

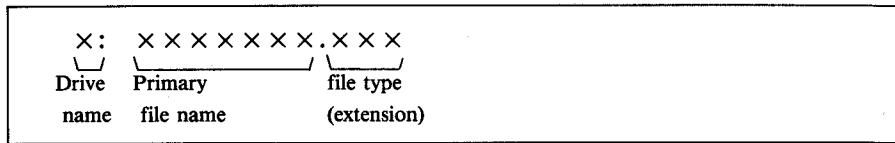
```
A>DIR C:
C: PIP          COM : STAT      COM : SUBMIT  COM : XSUB    COM
C: FILINK      COM : TERM      COM : CONFIG  COM
A>
```

Rather than type the command DIR, you can press the Programmable Function Key **PF1**. This will print the command "dir", including the required space.

To change the logged in drive, type the name of the new drive and press the **RETURN** key. For example to change the logged in drive to drive C: type the two characters "C:" and press **RETURN**. The prompt then changes to "C>". Now simply using the DIR command will give a directory of drive C:, and then return to show drive C: is the currently logged in drive. If you do this the screen will show:

```
A>C:
C>DIR
C: PIP          COM : STAT      COM : SUBMIT  COM : XSUB    COM
C: FILINK      COM : TERM      COM : CONFIG  COM
C>
```

Drive name can be used in combination with the file name described in section 3.2 as follows.



**Example**

B: BASIC.COM

or

A: SAMPLE.BAS

### 3.3.2 Changing disks, tapes and ROMs — warm starts

On a conventional CP/M system the disks in the drives could, and indeed would be changed regularly. If you have additional disk drives you will experience the same problems as on a conventional CP/M system if you do not change the disks in a correct manner. On the PX-8, unless you have additional floppy disk drives, only Microcassette tapes and ROMs will be changed. It is more important to obey the rules in changing tapes, as data you have saved may not be readable at all unless you remove the tape in the correct manner.

When you change disks, you must let CP/M know you have done so. This is because certain disk directory information is stored in memory at all times, which is used to allocate space on the disk. When a disk is changed, this information must be replaced by the information from the new disk. If you change a disk without telling CP/M you have done so, it is possible to lose data from the new disk, and also files may not be closed properly on the old disk making it difficult to read them.

IF YOU CHANGE FLOPPY DISKS always press **CTRL** - **C** or the **STOP** key after you have done so.

ALWAYS EXIT A PROGRAM IN AN ORDERLY MANNER by taking the EXIT option. In some cases there will be a specific command to do so, in others you can use **CTRL** - **C** .

A WARM START restores the internal information which CP/M uses to re-orientate itself. This is normally when disks have been changed or you have exited from a program. If the MENU is switched on, and you exit from a program if it does a warm start, the MENU will be displayed. If a program leaves the CP/M prompt on display and the MENU is switched on you should carry out a warm start by pressing **CTRL** - **C** or the **STOP** key. There may be a reason for the CP/M command line being displayed, most probably because there is information to be read from the screen which would disappear before you could read it, if the MENU came up straight away.

### 3.3.3 Changing ROMs

Details of physically changing over ROMs are given in Chapter 4. Since ROMs behave like write protected disks, when changing ROMs you must also carry out a warm start.

### 3.3.4 Changing microcassette tapes

The Microcassette drive has been designed to operate as a disk drive. However, because it is physically different it cannot be treated quite the same. It should become a habit to change a Microcassette tape through the System Display. Otherwise it is possible to not only make the removed tape unreadable, but also the one onto which you are writing the new data. Full details of using the Microcassette Drive are given in Chapter 4.



**WARNING:**

*A Warm start does not update the directory on a Microcassette tape. Always change tapes through the System Display. This can be done in the middle of an applications program by pressing **CTRL** and the **HELP** key. On completing any changes via the System Display, control will be returned to the applications program at the point where you left off.*

## 3.4 Using The Keyboard Under CP/M

Since CP/M has been designed as a universal operating system, it does not necessarily allow all the special keys on the PX-8 to function as they do in BASIC or in some applications programs. In particular the cursor keys are not supported by CP/M. Some of the special keys behave differently. The following summarises the use of the different functions available from the keyboard when the PX-8 is on the CP/M command line. They do not hold for the MENU command line.

### 3.4.1 Using the programmable function keys

The keys **PF1** to **PF5** and their shifted counterparts can be used to print the common CP/M command words, thus preventing errors and saving time. When setting them up, it is worth ensuring that trailing spaces are included to conform with the requirements of CP/M. The key display is normally absent from the lowest line of the screen but can be switched on using the CONFIG program. Another alternative in deciding which key provides which string is to use the attached Programmable Function key label. The CONFIG program also allows alteration of the strings given by each key.

### 3.4.2 Line termination commands

When you have entered a command, you must tell CP/M that you have finished. There are three key commands to do this:

The **RETURN** key is the simplest way to enter a command.

A **CTRL** - **M** is equivalent to a pressing the **RETURN** key.

A **CTRL** - **J** is equivalent to a line feed. On the CP/M command line it also causes the command to be entered as though you had pressed the **RETURN** key.

### 3.4.3 Character deletion commands

If a mistake is made, it is necessary to correct it. This can be done in two ways, but the CURSOR KEYS CANNOT BE USED.

**CTRL** - **H** or the BS (backspace) characters delete the character to the left of the cursor from the screen and move the cursor one step back.

The **DEL** key causes the key to the left of the cursor to be deleted, but also repeats the deleted character on the screen. If the **BS** or **CTRL** - **H** is used

to delete another character it will place the cursor one place to the left of this deleted character because any characters deleted by the **DEL** key do not exist except on the screen.

### 3.4.4 Line deletion commands

Sometimes it is necessary to delete a whole line when a mistake is made or for other reasons. There are two ways to do this:

**CTRL** - **X** will backspace to the beginning of the line, erasing all of the text.

**CTRL** - **U** will place a “#” character at the end of the line and then place the cursor on the next line ready to start over again. This is useful, because the incorrect command is still present on the screen, although it has been erased.

### 3.4.5 Display control commands

There are a number of control key sequences which can be used to handle the screen display. Again not all the special keys work, although some can be made to perform their function to a limited extent.

The **CLR** key can be used to clear the screen, although it is not supported by CP/M. It is equivalent to a **CTRL** - **L**. Typing either of these commands causes the CP/M command line to show:

A > ^L

Pressing the **RETURN** key will clear the screen, printing a question mark in the top left hand corner. The system prompt will be placed on the third line. This is because if you give CP/M a command it does not understand it will repeat the command followed by a question mark and return the system prompt.

**CTRL** - **E** can be used to overflow on to the next line before the user has pressed the **RETURN** key. This is useful if the screen is split (screen mode 1 or 2), as it enables a new word to be written on to a new line so that it can be read more easily. The “#” character is placed at the end of a line when using the **CTRL** - **U** or **CTRL** - **R** command to show it is a terminated line, to distinguish it from a **CTRL** - **E**.

**CTRL** - **R** is used to repeat a line with the errors removed if you are a bad typist and using the **DEL** key frequently. For example if you type an “R” instead of a “T” in the “STAT” command, and had deleted the “R” using DEL, and pressed **CTRL** - **R** keys after typing “T” again the screen would show:

A>STARRT #  
STAT

The **INS** key performs the same function as **CTRL** - **R** .

The **PAUSE** or **CTRL** - **S** key can be used to temporarily halt the screen scrolling. For example if a DIRectory has too many files to fit in the screen window, pressing either the **PAUSE** or **CTRL** - **S** keys will stop the display until another key is pressed, enabling the text to be read. The shifted cursor keys can also help with this problem as described below.

The SHIFTED CURSOR KEYS ( **↑** and **↓** ) can be used to scroll the real screen window up and down the virtual screen. Thus if a long DIRectory has scrolled off the real screen, these keys can be used to bring it on again.

The **CTRL** and CURSOR KEYS ( **←** and **→** ) can be used to change the displayed virtual screen in modes 0 and 1 and the cursor position in screen mode 2. For example a DIRectory can be stored on one virtual screen while a program is used on another. Care must be taken in using this facility. Full details are given in section 2.2.6.

The **SCRN** key can be used to lock the real screen window on part of the virtual screen. This is described in section 2.2.6.

### 3.5 Printing When On The CP/M Command Line

There are two ways to print when on the CP/M command line.

The **CTRL** and **PF5** key can be used to dump the contents of the screen to the printer.

**CTRL** - **P** can be used to toggle the echoing of text to the printer. The first time **CTRL** - **P** is pressed anything output to the screen is simultaneously sent to the printer. Pressing **CTRL** - **P** a second time only sends input to the screen. This can be used in utility programs, but may not work in applications programs.

## 3.6 The Concept of Logical And Physical Devices

As a universal operating system, CP/M has to have a means of communicating with all the devices associated with a computer in a general way. The software in each computer is then connected to the devices which are unique to that particular computer. This is achieved through the use of the concept of logical (symbolic) device names in CP/M being assigned to physical (actual) devices supported by the computer. Such a concept considerably simplifies the use of the computer.

### NOTE:

*In practice assignments are carried out by the use of the CONFIG program described in section 3.8. This allows various devices to be assigned using common names, rather than following the normal CP/M system. The description outlined below is presented for completeness, and for those familiar with CP/M, who wish to use the STAT program (section 3.8) to change assignments.*

### 3.6.1 The logical devices

There are four logical devices in CP/M which correspond to the types of devices which one would expect with a computer. They are given the symbols CON:, RDR:, PUN:, and LST: from the names associated with their functions.

**CON:** denotes the CONsole device, which is the means of allowing data to be input by and output to the user. In the case of the PX-8 it would correspond to the keyboard and LCD screen.

**RDR:** denotes the ReaDeR device, which receives information from another source. The name originally derives from a paper tape reader, but in the PX-8 is normally the RS-232C device.

**PUN:** denotes a device to output information. Its name derives from a paper tape PUNch. In the PX-8 it is normally assigned to the RS-232C port.

**LST:** denotes the output LiST device, normally a printer.

### 3.6.2 The physical devices

There are many possible physical devices which can be attached to a computer. Examples of such devices are printers, plotters, modems, and video monitors. They all have one thing in common, they involve the transfer of information to or from the computer. The physical devices are given symbols in CP/M which like the logical symbols relate to their function. A computer manufacturer may not support each device, and different manufacturers may give different assignments to the same device. The symbols corresponding to the physical devices are:

**TTY:** denotes a slow speed terminal. Its name is derived from the TeleTypewriter, which has a keyboard and printer as input and output devices for the user to communicate with the computer.

**CRT:** represents a high speed console device. Its name is derived from the commonest type of device a Cathode Ray Tube.

**BAT:** denotes a BATch processing device, which historically relates to devices such as punched card readers.

**UC1:** denotes a User defined Console device, which allows special console equipment to be used.

**PTR:** corresponds to a Paper Tape Reader.

**UR1: and UR2:** correspond to two different User defined Reader devices. They enable a computer manufacturer to provide two different high speed devices for input of data.

**PTP:** denotes a Paper Tape Punch device for output of data to paper-punch equipment.

**UP1: and UP2:** correspond to two different User defined Punch devices. They enable a computer manufacturer to output data to two high speed output devices.

**LPT:** is a Line PrinTer which allows data to be printed for a permanent record.

**UL1:** is a User defined List device. It enables the computer manufacturer to provide an interface for communication to special devices such as plotters.

### 3.6.3 Physical Devices Supported by the PX-8

The PX-8 does not have all the physical devices which could be supported by CP/M. The physical devices which the PX-8 has are:

- 1) The LCD display.
- 2) The keyboard
- 3) High speed serial port, which is used to communicate with floppy disk drives, printers and other special devices.
- 4) RS-232 serial port, which is used to communicate with other computers and also printers.

A computer manufacturer has to modify CP/M so that a particular physical device corresponds to one or more of the names allowed by CP/M. A device can correspond to more than one name, and each of the logical devices can be connected to more than one physical device. The four logical devices can be assigned to the physical devices as follows:

**Table 3.1**

Logical Device	I/O	Physical Device			
LST:	O	TTY: Serial (printer)	CRT: LCD display	LPT: RS-232C	UL1 not implemented
PUN:	O	TTY: not implemented	PTP LCD display	UP1: RS-232C	UP2: not implemented
RDR:	I	TTY: keyboard	PTR: not implemented	UR1: RS-232C	UR2: not implemented
CON: output input	O I	TTY RS-232C keyboard	CRT: LCD keyboard	BAT: RS-232C RS-232C	UC1: RS-232C RS-232C



## 3.7 The CP/M Built-In Commands

Built-in commands are commands which can be executed simply by typing in their name on the command line. They do not show up as files on the directory (as do STAT and PIP, for example) but are part of CP/M itself. They are:

<b>DIR</b>	Display a list of files on a particular disk.
<b>ERA</b>	Erase a specified file or files.
<b>REN</b>	Rename a specified file.
<b>SAVE</b>	Save a portion of memory as a file.
<b>TYPE</b>	Display the contents of a file on the screen.
<b>USER</b>	Assign files to different users

### 3.7.1 DIR (Directory)

This command is used to display a list of file names stored on a disk. There are various ways of specifying the files depending on the results you want. These are achieved using wildcards (section 3.2.1). On a conventional CP/M computer the CP/M operating system is loaded into the computer from disk. It is not displayed on the directory. It is also possible to make other files non-directory files. This can be done in two ways. Files can be set (and reset) to 'SYS' files using the STAT program. These files do not appear on the directory of any user. Files can also be assigned to different users using the USER command, and then only appear on the directory when the particular user to which they belong is using the disk.

Points worth bearing in mind when using the DIR command are:

- There must be a space between DIR and the filename (if there is one).
- No space must be put between the drive name and filename if this is used.
- The files are not sorted into alphabetical order, and are usually displayed in the order in which they were first put on to the disk. If you require an alphabetical list it is best to use the STAT command.

**a) List all files on the currently logged-in drive**

The simplest way to do this is to type DIR on the command line:

```
C>DIR
C : PIP      COM : STAT  COM : SUBMIT  COM : XSUB  COM
C : CONFIG  COM : FILINK COM : TERM   COM
```

The same result can be obtained by being more long-winded and using the drive-name or wildcards:

```
C>DIR C:
C>DIR C:????????.*
```

**b) List all the files on a drive other than that currently logged in**

This can be done simply by adding the required drivename to the command:

```
C>DIR A:
A : PIP      COM
```

```
C>DIR B:
B : BASIC      COM
```

c) List all the files with the same extension on any drive  
Wildcards can be very useful when this is required. For example, to show all the COM files on the current drive:

```
C>DIR *.COM
C : PIP      COM : STAT  COM : SUBMIT  COM : XSUB  COM
C : CONFIG  COM : FILINK COM : TERM   COM
```

```
C>DIR A:*.BAS
NO FILE
```

This message informs you that there are no files conforming to that name on the A: drive. The message will appear whenever you ask for a DIRECTORY of a drive on which no files with the specified name reside.

```
C>DIR D:*.BAS
D : TEST      BAS : DEMO1  BAS : DEMO2  BAS
```

There may be other files on drive D: but as the request was only for those with the extension BAS so they are the only ones listed.

d) Check that a particular file is on a particular drive  
If you ask for a particular file by name it will be displayed on the screen. If it is not present the "NO FILE" message will be printed.

For example:

```
A>DIR B:TEXT1.DAT
B:TEXT1.DAT
A>
```

shows the file is on disk B :

```
A>DIR B:TEXT1.DAT
NO FILE
A>
```

shows the file is not present.

e) List all the files with the same name on any drive  
It may be that there are files with the same name but different extensions on the same drive, for instance assembler program files, so the asterisk wildcard can again be used:

```
D>DIR START.*
D : START    ASM : START    HEX : START    PRN
```

```
C>DIR D:START.*
D : START    ASM : START    HEX : START    PRN
```

f) List all files with similar names on any drive  
This is where you can really start to use the wildcards! For instance:

```
D>DIR S??????.*
```

will list all files on drive D: whose names begin with S of any extension;

```
D>DIR C : *.C??
```

will list all the files on drive C: whose extension begins with C;

```
A>DIR D:DEM??.BAS
```

will list all those files on drive D: whose names consist of five letters beginning with DEM and whose extensions are BAS.

g) Microcassette Tape DIRECTORIES

The Microcassette drive is always named drive H: and is used in the same way as the other drives.

```
H>DIR
H : TEST.BAS
```

If the tape is mounted, the directory will be read from the directory stored in memory, and so will be read immediately.

If the drive is unmounted when an access is made the drive will wind the tape to the beginning and mount the tape. The directory is then stored in memory. It may take a short while for the tape to be mounted.

It is better to keep to the habit of mounting and removing the tape through the System Display. Before carrying out a directory operation on the Microcassette drive, use the System Display to check if the tape is mounted. If it is not mounted, mount it through the System Display.



**WARNING:**

*NEVER try to obtain the directory of a Microcassette tape when another tape is already mounted. The directory of the second tape must be loaded into memory before the directory can be printed. This can only be done by removing the first tape and then mounting the second tape.*

**h) Pausing a DIRectory**

Either the **PAUSE** key or **CTRL** - **S** can be used to pause a DIRectory as it is displayed. If a directory has scrolled off the screen in screen modes 0,1 or 2, it can be seen by using the shift cursor keys ( **↑** and **↓** ).

**i) Printing a DIRectory**

To obtain a hardcopy of a DIRectory, use **CTRL** - **P** before issuing the DIR command or use the screen dump facility ( **CTRL** and the **PF5** key).

**j) Stopping the DIR**

If any key is pressed during execution of this command it is terminated and the CP/M command line reappears.

### 3.7.2 ERA (Erasing files)

Files can be erased at any time using the ERA command. The command is used in a similar way to the DIR command and the rules about spaces apply equally. It is possible to set files to be Read/Only files. Files which have been set to Read/Only cannot be erased. If you really do want to erase them you can use the STAT program to set them to Read/Write. In fact ERA does not really erase the file from the disk - it simply removes the directory entry for it.

Files on the ROM drives cannot be erased. If an attempt is made to erase a file on a drive which is set to Read/Only, one of the following error messages will be displayed:

**BDOS ERROR ON dr:FILE R/O**  
**BDOS ERROR ON dr:R/O**

will occur. The particular drive name "dr:" will be printed as the drive name accessed. If this does occur you should press **CTRL** - **C** , **STOP** or **RETURN** .

**a) Erase a specific file on any drive**

To do this you simply specify the full filename and extension of the file you wish to erase:

```
D>ERA TESTING.COM
D>
```

```
C>ERA A:WASTE.FIL
C>
```

If the file is not present the "NO FILE" message will be printed.

```
C>ERA DUDFILE.COM
NO FILE
C>
```



**WARNING:**  
**USING WILDCARDS WITH THE ERA COMMAND**

*It is very easy to erase files you do not wish to erase from a drive by using wildcards with the ERA command.*

*TO AVOID ACCIDENTS use the DIR command with the same wildcard option you wish to use. The files displayed by the DIRectory will then be the ones ERAsed. This will ensure that you know what your action will be without causing anything drastic to happen.*

*IF YOU ARE USING 'SYS' FILES these will be deleted by the ERA command, even though you did not see them on the directory. To avoid accidents use the STAT program to check the complete list of file names if you are using 'SYS' files.*

Alternatively delete files one at a time.

**b) Erase all files on a drive**

This is achieved by using the asterisk wildcard. Since this is drastic action, it results in a question from the PX-8 asking if you are sure you want to erase all the files:

```
A>ERA *.*  
ALL (Y/N)?
```

If you answer N execution will be terminated, and answering Y will result in all the files in the directory being ERAsed. Any files NOT in the directory will not be erased, that is system files and those with the SYS attribute set.

To erase all files on another drive the drivename must be specified:

```
C>ERA D: *.*  
ALL (Y/N)?
```

**c) Erase all files with the same filename on any drive**

The asterisk wildcard can be used to erase all the files on a particular drive, which have the same name but whose extensions differ.

```
D>ERA START.*
```

will erase all those files on drive D: whose filename is START and whose extension is anything, for instance, START.ASM, START.BAS or START.DAT.

```
C>ERA D:START.*
```

will have exactly the same effect but is executed when C: is the logged-in drive instead of D:.

**d) Erase all files with the same extension on any drive**

To do this you must specify the extension in full but use the asterisk wildcard to allow any filename:

```
C>ERA *.COM
```

will erase all the files on drive C: having the extension COM.

```
C>ERA D:*.BAS
```

will erase all files on drive D: with the extension BAS, that is, all BASIC program files.

**e) Erase all files with similar names or extensions on any drive**

Here you can use both the asterisk and the question mark wildcards. The asterisk replaces an entire filename or extension and the question mark replaces individual characters in the filename or extension:

```
C>ERA S?????.*
```

will erase all files on drive C: with any extension and whose filename consists of six characters beginning with S.

```
C>ERA D:TES?????.BAS
```

will erase all those files on drive D: whose extension is BAS and whose filename begins with TES.

```
D>ERA A:DEMO???.?A
```

will erase all files on drive A: whose filename consists of six letters beginning with DEMO and whose extension ends in A.

**f) Erasing Files on a Microcassette drive.**

On a conventional drive when a file is removed from the directory, the disk is opened up for saving other files. On a Microcassette tape the files are saved sequentially as a single block. This means that the removing a series of files from a tape directory does not free up the tape. It will simply removes the name

from the directory. If the space were freed, it would be possible for a new file to overwrite the start of a current file. If the file is the last file on the tape, the file space will be set free for further use.

To remove a block of files from the middle of a tape, and free the tape for new files,

- o Use the PIP program to transfer the files at the end of the tape into the RAM disk.
- o ERASE all files from the block you want to delete to the end.
- o Use the PIP program to replace the end files back on the tape.



**WARNING:**

*The directory is only updated in memory when disk access is carried out. Only when the tape is removed will the directory be written to tape. It is especially important to remove a tape using the correct procedure if the ERA command has been used on a Microcassette tape.*

### 3.7.3 **REN** (Rename a file)

**Format**    **REN** new filename = old filename

A file can be renamed at any time by giving this command followed by the new filename, an equals sign and the old filename. There must be a space between **REN** and the new filename. although it is optional either side of the equals sign.

```
A>REN NEW.COM = OLD.COM  
A>
```

has changed the filename "OLD.COM" to "NEW.COM".

If the new filename you have chosen already exists on that drive, the message **FILE EXISTS** is displayed and the command line reappears:

```
A>REN TESTING.BAS = DEMO.BAS  
FILE EXISTS
```

This indicates that there is already a file on drive A: called TESTING.BAS. This can also happen if you specify the names the wrong way round, since a test is carried out on the new name before carrying out the renaming.

If a drive name is specified it must be placed before new filename:

```
C>REN A:START.COM = TESTING.COM
```

This example will have the result of renaming a file on drive A: called TESTING.COM to be called START.COM.

Bear in mind that this command only renames the file, so the old filename disappears from the directory and is replaced by the new name while the file itself remains exactly the same. Another important thing to remember is that you can only rename files on one drive, for example:

```
C>REN A:DEMO.COM = B:TESTING.COM
```

is not allowed and will result in the message:

```
B:TESTING.COM?
```

If you want to copy a file from one drive to another and rename it at the same time use the PIP program.

### 3.7.4 **SAVE** (Save an area of memory)

First of all, remember that this is completely different to the SAVE command in BASIC. The only feature they have in common is that they store programs on disk, but there the resemblance ends. It enables blocks of memory to be saved as a file on a disk. Normally only programmers will use this command, but there is one case where it may be useful to even BASIC programmers which is described below.

The format for the SAVE command is very simple.

#### **SAVE n filename**

The filename consists of a drivename (if necessary), filename and extension, in the usual way.

The 'n' parameter denotes the number of "memory pages" to be saved and must have a space either side of it. The value is specified in decimal notation. One page consists of 256 bytes of data, so the basic method of calculating n is to divide the number of bytes you want to save by 256 and round the answer up to the next integer. For instance, if you want to save 300 bytes the number of pages (n) would be 2 because 300 divided by 256 is greater than one but less than two.

A frequent use of SAVE is to write to disk a program which has been patched using the DDT program (see section 3.10). When the program is loaded with DDT the address of the next available byte is shown as NEXT. You can calculate the number of pages to SAVE in the following way:

- 1) Round the value shown as NEXT up to the next highest page. Taking as an example if 2C5A were returned by NEXT in DDT, this would give 2D00 (remembering that you are working in hexadecimal notation).
- 2) Drop the zeros from the end of the rounded value and calculate the value of the remaining digits in decimal. In the example 2D becomes 45 decimal.
- 3) Since DDT loads programs at the beginning of the first page of memory, subtract 1 to allow for this, giving 44 in the example.

If the file we have just patched is called FREDDY.COM and lives on the D:drive, the command given to write it back to disk will be

**C>SAVE 44 D:FREDDY.COM**

#### a) The "SAVE # GO.COM" Trick

There is a special trick you can play on the PX-8 using the SAVE command which can be very helpful in various circumstances. When you exit a program it is still present in memory. For example when exiting BASIC using the 'system' command, the BASIC interpreter and the programs in the five program areas are still existant in memory as far as MENU is on. You may then remember that one of the programs should have been saved, and will be lost if you try to restart BASIC. Also you might load a program such as PIP, then having used it once to transfer a single file done nothing except a DIR or ERA or other minor operation, and then wish to use it again. In either case the program (the BASIC interpreter, PIP or whatever) can be recalled instantly if you know how, instead of having to be reloaded from disk or even tape. In the case of BASIC the five programs stored in memory can be recovered just as if you had not exited to CP/M. First you have to create a file called GO.COM then execute it when you wish to continue with the previous program by typing GO. To create this file all you have to do is type:

```
A>SAVE # GO.COM
```

or

```
A>SAVE # D:GO.COM
```

If you want it on the D: drive, for instance. what you are actually doing with this command is making a file of length 0 bytes and saving it on the specified drive. When you type GO at the prompt the program GO.COM (of length 0 bytes) is loaded into memory and executed. Because the file contains 0 bytes it does not overwrite the program already in memory, so in fact the program already there is executed instead.

### 3.7.5 TYPE (Displaying the contents of a file)

**Format**    **TYPE filename**

This command is used to display a file on the screen. There must be a space between TYPE and the filename. This is usually an ASCII file because any other data format results in a screenful of rubbish and is totally incomprehensible. This is because as the computer reads through the file it tries to convert all the hexadecimal numbers it finds into printable ASCII characters, and the only files which produce anything recognisable are those which are in ASCII format. For example, if you wanted to look at a BASIC program file using TYPE, it would have to have been SAVED in ASCII format using the "A" option in BASIC. Normally, BASIC programs are stored in compressed binary format which will produce some very strange results if it is TYPEd to the screen.

#### a) Displaying the contents of a file to the screen

The command is invoked either to show the contents of a file from the current drive:

```
A>TYPE TEXT.TXT
```

or for example:

```
C>TYPE D:DATAFILE.DOC
```

will display the contents of the file DATAFILE.DOC which is held on drive D:.

The only files which you can be sure will contain data in ASCII format are those with the extensions ASM, PRN and DOC or TXT. Others may be in this format, and it doesn't do any harm to look at them anyway if you want to in order to check. WILDCARDS cannot be used with the TYPE command.

If the file is not on the disk the filename will be repeated on the next line with a question mark after it, for example:

```
C>TYPE B:TEXT1.DOC  
B:TEXT1.DOC?
```

This would also happen if an attempt was made to use wildcard characters.

**b) Pausing the display of a text file.**

When the text is displayed on the screen, it may be going too fast to be read. You can pause the display by pressing the **PAUSE** or **CTRL** - **S** key. If you wish to see some of the text which has scrolled out of the real screen window, but is still present on the virtual screen, the shifted cursor keys ( **↑** and **↓** ) can be used when the display has been paused. To continue the display press any key other than the shifted cursor keys.

**c) Stopping the Display**

Pressing any key other than the **PAUSE** , **CTRL** - **S** or a shifted cursor key will cause the TYPE command to cease displaying the file. The PX-8 will then return to show the CP/M prompt.

**d) Printing a File using TYPE.**

It is possible to echo the file to the printer by typing **CTRL** - **P** before invoking the TYPE command, and so produce a hard copy of the text.

Small passages (i.e. of screen size) can be printed using the screen dump, by pressing **CTRL** and the **PF5** key.

**e) Problems with TYPE using non ASCII files.**

If an attempt is made to view a non text file, the screen may be covered with a garble of non displayable characters. When stopped, the CP/M prompt may not appear. In this case you will have to press the reset button, and in extreme cases initialize the system. See section 2.1.3 if this occurs.

### 3.7.6 **USER** (selecting a different user areas)

**Format**      **USER n**

If more than one person will be using the PX-8 it is a good idea to give each a user number. This has the effect of dividing the drives into different areas, each of which contains files belonging to a different user. Thus two users may label their areas 0 and 1, and they can both have files with the same names because their numbers are 'invisibly' added to the filenames in their respective directories. When the PX-8 is switched on it defaults to user area 0, that is, it assumes you are user number 0 unless you tell it otherwise. If you are user number 2, simply give the command

**C>USER 2**

and you will have immediate access to your own files. There must be a space between USER and the area number. If you want to look at a file in another user area you can either give the USER command again with the relevant number, or use the PIP program using the [G] option.

To find out which user numbers are active on a particular disk, use the STAT program.



## 3.8 Using Utilities and Application Program with CP/M

Computers are normally used to run applications programs, for example to write text using a word processor. These programs are loaded from the disk into the memory of the computer when they are needed. For this reason they are often called transient commands to contrast them with the built-in commands. Such programs are given the file extension .COM meaning a command type of file. They are executed by typing the name of the file without the extension as if they were a command. For example to use the file TERM.COM which is a program to use the PX-8 as a terminal, you would simply type the word TERM after the CP/M prompt.

### 3.8.1 Utilities

The PX-8 is provided with a ROM which contains a number of programs which are used for common day to day management of the computer, for example transferring files and changing parameters. They perform the following functions:

PIP is a program for copying files.

STAT is a program to determine the status of the disks and perform general housekeeping functions.

SUBMIT is used to process a sequence of commands.

XSUB is used to allow input from the keyboard with SUBMIT.

CONFIG is used to change system parameters which are not changed often.

TERM is a program to allow the use of the PX-8 as a terminal.

FILINK is used to transfer files between the PX-8 and other computers.

These programs are discussed in detail in the following pages.

Other programs are often provided with CP/M. These are discussed in section 3.10. Please consult your dealer for further information.

### 3.8.2 Application programs

You will also use programs such as Portable WordStar™ and Portable Calc™. They will have instruction manuals which will give you details of how to use them. Whatever media they are provided on, ROM, Microcassette tape or floppy disk, they will have a file in the directory which is a .COM file to run them. They are executed in the same manner as the utility programs. They are transient commands of a different kind.

### 3.8.3 PIP

The PIP program is used to copy files between peripherals, e.g. from disk to disk, disk to Microcassette tape, from a disk to a printer etc. The name comes from the initial letters of the words Peripheral Interchange Program.

PIP can be used with wildcards (see section 3.2.1) to transfer all types of files. It can also be used to perform other valuable functions such as:

- Remove part of a file either from the beginning, end or the middle.
- Convert all characters into upper case or lower case
- Join a number of files together
- Make a backup of a file under a different name
- Add sequential numbers to each logical line of text
- Reform the page length of a text file

There are two ways PIP can be used.

- 1) When only a single operation is required, PIP can be used by following the command 'PIP' with the command string of the operation you wish to carry out. On completion of the operation the CP/M prompt will be returned or the MENU displayed if it is switched on. When using PIP from the MENU the command string can be typed on the MENU command line, when PIP has been selected.
- 2) When a number of operations are to be performed, the PIP program can be loaded into memory. The CP/M prompt (A>, B> etc ) will be replaced by a PIP prompt indicated by an asterisk. The use of PIP in this manner is essentially the same as when only one operation is required.

When PIP is chosen from the MENU, unless an additional command string is given, the PIP screen will clear and the PIP asterisk prompt will appear.

Pressing the **RETURN** key, the **STOP** key or **CTRL** - **C** will return to the MENU screen or the CP/M system prompt whichever is set.

In using PIP the command is written to perform the operation to a particular drive or device, from a particular drive or device. The location of the file being moved always comes last.

#### 1. COPYING A FILE FROM DRIVE TO DRIVE

Any file with a read/write attribute can be copied from one disk to another. For example,

```
C>PIP H:=A:INFO.DAT
```

This copies the file DEMO.DAT from drive A : to drive H: then returns to the system prompt. The same filename will appear on the new drive. The filename can be changed, for example:

```
C>PIP H:NEWNAME.DAT=A:INFO.DAT
```

By using this option to copy the file to the same drive, it can be used to make a backup of a file under a different name.

Various options can be added to the end of the PIP command string. These are listed starting from item 5. For example specifying the [V] option causes the file to be verified as it is copied:

```
C>PIP H:=A:INFO.DAT[V]
```

#### 2. COPYING A FILE FROM DISK TO PRINTER

This has almost the same effect as pressing **CTRL** - **P** then asking the computer to TYPE a file:

```
C>PIP LST:=A:LETTER.TXT
```

The file will be printed character by character, and words which come at the end of a line are likely to be split in the middle of the word.

This command can also be used to list a BASIC file to a printer if you are not in BASIC and do not have the BASIC ROM installed.

```
C>PIP LST:=A:DEMO.BAS
```

The BASIC program is sent to the printer as if you had LLISTed it while in BASIC. You MUST have saved the BASIC program as an ASCII file using the "A" option in BASIC.

A refinement of this command is to add the option [Pn] where the printer will

execute a form feed every n lines. This can be used to prevent lines being printed on the perforations. For example:

```
C>PIP LST:=A:INFO.DAT[P60]
```

will force a form feed every 60 lines giving the same effect as a one-inch skip-over on paper which takes 60 lines.

Another addition could be to echo the file to the screen at the same time as it is printed. This is achieved using the [E] option:

```
C>PIP LST:=A:INFO.DAT[E]
```

If both options are required, the command would be:

```
C>PIP LST:=A:INFO.DAT[P60E]
```

### 3. COPYING A FILE FROM DISK TO SCREEN

The use of this command will have the same effect as the [E] option when copying from and to any other device. It is also the same as using TYPE command. The format is:

```
C>PIP CON:=A:MEMO.DOC
```

One of the options that can be specified here to good effect is [N] or [N2]. This causes line number to be added at the beginning of each line in the form 01 if [N] is specified and 000001 if [N2] is specified:

```
C>PIP CON:=A:MEMO.DOC[N2]
```

In addition, all lower case characters can be converted to upper case, and vice versa, using the [U] and [L] options:

```
A>PIP CON:=A:MEMO.DOC[N2U]
```

will give

```
C>PIP CON:=A:MEMO.DOC[N2U]
000001 THIS IS A LINE OF TEXT
000002 HAVE A NICE DAY
```

```
C>
```

and

```
A>PIP CON:=A:MEMO.DOC[NL]
```

will give

```
C>PIP CON:=A:MEMO.DOC[NL]
1: this is a line of text
2: have a nice day
```

```
C>
```

### 4. COPYING A FILE TO AN EXTERNAL DEVICE

Files can be copied to external devices such as printers, disk drives and other computers by specifying the relevant output port as the destination. For instance:

```
A>PIP TTY:=A:DEMO.BAS
```

will copy the file to a printer or any other device connected to the high-speed serial output port.

### 5. OPTIONAL PARAMETERS AVAILABLE WITH PIP

There are nineteen options which can be used with the PIP command, including those which have been mentioned already:

#### 1. Block mode transfer

**Format** [B]

This option causes the data to be copied across in blocks, that is, 256 bytes at a time. When the buffer has received a block it sends the data to the destination device before allowing the source device to send it the next block. The actual character it recognises as an end-of-block marker is ASCII code 19, that is, **CTRL** - **S** or X-OFF.

**Example** A>PIP TEST.FIL=CRT:[B]

#### 2. Echo to screen

**Format** [E]

Giving this option will cause the screen to display all the data being transferred so that you can see what is being copied across during the process.

**Example** A>PIP LST:=TEST.FIL[E]

It is advisable to use the [V] entry, i.e. to verify the file when using the [E] option in copying files.

### 3. Form feed insertion

**Format** [Pn]

This makes it possible to force a form feed every n lines. It is particularly useful when sending files to the printer, allowing it to skip the perforations in continuous stationery to improve legibility. If n is omitted or given as 1 PIP assumes a form feed is required every 60 lines (that is, the default setting). If your paper requires 66 lines per page use the parameter as in the following example:

```
A > PIP LST: = REPORT.TXT[P66]
```

If used with the next parameter, [F], the [F] should come first:

```
A > PIP LST: = REPORT.TXT[FP66]
```

### 4. Form feed suppression

**Format** [F]

This is most useful when using PIP with a printer. It suppresses the form feed character (ASCII code 12 or hex 0C) which would otherwise cause the printer to feed a sheet through when it encounters the code. It is as well to remember that this code is also the 'clear screen' code, so it is advisable to take care when using this option.

**Example** A > PIP TTY: = TEXT.DOC[F]

In conjunction with [Pn] it can be used to reform page lengths, by copying a file, removing previously added form feeds (e.g. ones inserted by PORTABLE WORDSTAR) and adding them at a different place.

### 5. Hex format

**Format** [H]

When this option is specified the data is checked to ensure it is in Intel HEX format. If there is a discrepancy it terminates the transfer.

**Example** A > PIP DEMO2.HEX = DEMO1.HEX[H]

### 6. Ignore NULL records

**Format** [I]

This can be used as an alternative to [H] - it causes PIP to ignore NULL records (hex 00) and ensures the data is in Intel HEX format. It is therefore an extension of the [H] option.

**Example** A > PIP DEMO2.HEX = DEMO1.HEX[I]

### 7. Lower case conversion

**Format** [L]

Using this option will convert all upper case characters into lower case as they are transferred.

**Example** A > PIP LST: = C:LITTLE.DOC[L]

### 8. Numbering lines

**Format** [N]  
[N2]

When sending programs to another device it may be useful to have the lines numbered. PIP regards a line as a series of characters terminated by a carriage return (ASCII code 13 or 0DH). Specifying [N] will begin the file at column 9, with a colon(:) at column 7 followed by a space. The number of the line will be placed before the colon, as the following example shows:

If the file TEST.DOC contains the following information:

```
This is line one  
This is line two  
      *  
This is line ten  
      *  
This is line one hundred
```

where the text would also correspond to the numbers of the lines as seen by PIP, and the asterisks refer to further lines of text. If the file is transferred to from drive D: to drive A: using the [N] option, using:

```
C > PIP A: = D: TEST.DOC[N]
```

The file on drive A: will have the lines numbered as follows:

```
1: This is line one
2: This is line two
   *
10: This is line ten
   *
100: This is line one hundred
```

Similarly specifying [N2] fills in all leading spaces with a zero and replaces the colon with a space. For example if the following command is given:

```
C>PIP A:=D: TEST.DOC[N2]
```

the file on drive A: will have the lines numbered as follows:

```
000001 This is line one
000002 This is line two
      *
000010 This is line ten
      *
000100 This is line one hundred
```

If you wish to display the file with numbered lines on the screen, use:

```
C>PIP CON:=D: TEST.DOC[N]
```

or

```
C>PIP CON:=D: TEST.DOC[N2]
```

## 9. Object file transfer

**Format** [O]

Normally PIP can only copy standard ASCII or HEX files, but using this option allows it to transfer other types of files. Its effect is to ignore the physical end-of-file code a CTRL-Z (ASCII 26 or 1A hex) wherever it occurs in the object file, because in this context it will not be signalling the end of the file.

**Example** A>PIP B:=OBJECT.FIL[O]

It is not necessary to use this optional parameter with a COM file, because PIP will automatically add it. However, when using it with other machine code or object files which do not have the file extension COM, the [O] parameter MUST be used.

## 10. Read system files

**Format** [R]

This option makes it possible to read and copy system files, that is, files which do not appear in the directory and those with a filetype of SYS. It automatically sets the [W] option.

**Example** A>PIP B:=A:OSTAB.SYS[R]

## 11. Stop copying at specified string

**Format** [Qstring^Z]

(^Z means type a **CTRL** - **Z** by holding the **CTRL** key down while pressing the Z key.)

If you only want to transfer part of a file, giving this option will cause PIP to copy the file until it finds the specified string. Only text before the string will be copied.

**Example** A>PIP LST:=B:REPORT.DOC[QTHE END^Z]

If you want to search for a string containing lower case characters this can only be used from the PIP \* prompt. This is because CP/M converts everything typed on the command line into upper case including the string you have given and PIP will not be able to find it, because it is searching for the upper case equivalent of your string.

**Example** \*LST:=B:REPORT.DOC[QThe End^Z]

It can be used with the next option to copy a section from the middle of a file.

## 12. Start copying at a specified string

**Format** [Sstring^Z]

This behaves in much the same way as the [Qstring^Z] option, except that it starts copying from the end of the specified string. The same conditions apply

if you want to detect lower case characters.

**Example** A>PIP CHAPTER1.DOC=CHAPTER1.DOC[SIN-  
TRODUCTION^Z]

The two options can be used together if you want to copy a piece of data from the middle of a file.

**Example** \*CHAPTER1.DOC=CHAPTER2.DOC[SIntroduction ^ZQlast.^Z]

### 13. Tab settings

**Format** [Tn]

The tab settings on the copied file can be changed from those of the original using this option. The number given by n puts tabs at columns n, 2n, 3n etc. For instance, [T9] will give tabs at columns 9, 19, 29, 39 and so on. These settings will be used wherever PIP comes across a TAB character in the file it is copying. The TAB character is **CTRL** - **T** (ASCII code 9).

If the wordprocessor, or whatever created the text file, inserts spaces instead of using the TAB character, this option will have no effect.

**Example** A>PIP CON:=B:PROGRAM.ASM[T10]

### 14. Transfer between user areas

**Format** [Gn]

Normally it is not possible to use files from another user area. However, specifying this option will allow transfer of files from user area n to the current area.

**Example** C>PIP A:TEST.DOC=A:DEMO.DOC[G3]

### 15. Truncate lines of data

**Format** [Dn]

This option allows truncation of lines past column n, that is, PIP will delete all characters between column n and the next carriage return.

**Example** A>PIP LST:=PROGRAM.BAS[D80]

### 16. Upper case conversion

**Format** [U]

This works in the opposite sense to the [L] option in that it converts all lower case characters to upper case.

**Example** A>PIP TTY:=BIGTYPE.TXT[U]

### 17. Verify the copy

**Format** [V]

Verifying a file as it is copied acts as a double check on its integrity. When given this option PIP compares the copy it has made with the original as it goes along, ensuring a faithful reproduction containing only those errors that were in the original.

**Example** A>PIP B:=PERFECT.COM[V]

### 18. Write to a read/only file

**Format** [W]

If you have files which have been set to read-only using STAT, you can copy over them if you give this option without the computer asking you if you want the existing file erased and overwritten. As it is not possible to reverse the process if you make a mistake, use with care!

**Example** A>PIP A:=B:SECURE.COM[W]

### 19. Zero parity setting

**Format** [Z]

The leftmost bit of a byte is usually the parity bit, and this option sets all these high-order bits to zero. It results in converting 8-bit ASCII bytes to 7-bit ASCII bytes. It is as well to ensure none of the characters you are sending use this bit because otherwise you could get some unexplained results. For example graphics characters would be changed, and some console ESC codes would be changed.

**Example** A>PIP C:=B:ORDINARY.FIL[Z]

## USING THE PIP \* PROMPT

If you want to transfer a large number of files or transfer files between two data disks, it may be more convenient to give the PIP command on its own, producing the \* prompt:

The command string as described above can then be typed for each operation. For example to copy all the COM files to drive A: from drive C:, the display would show:

```
C>PIP
*E:=C:*.COM

COPYING -
PIP.COM
STAT.COM
SUBMIT.COM
XSUB.COM
FILINK.COM
TERM.COM
CONFIG.COM
*
```

Or to output a file to the screen:

```
C>PIP
*CON:=MEMO.DOC
```

At the \* prompt you can type in any PIP commands you like without having to load PIP on every operation. This can save a great deal of time if there are a number of operations to be carried out.

One you have finished, pressing the **RETURN** key, the **STOP** key or **CTRL - C** will return you to the system prompt on the drive form which you entered PIP (the default drive), or to the MENU if it is switched on.

## 3.8.4 STAT

The STAT program is used to find out the STATistics or STATus of the various disk drives. This makes it the most frequently used of the utility programs. Many CP/M users do not use the full facilities provided by STAT. The following information is therefore provided as a list of operations, with a summary at the end.

### STAT and the Microcassette Drive

Because the Microcassette drive is not a standard CP/M disk device, but is a sequential tape device, the STAT program cannot determine the correct data for space on the Microcassette Drive. Also file name lengths may not be accurate.

## USE OF THE STAT PROGRAM

### 1. CHANGE DEVICE ASSIGNMENTS

**Format** STAT logical: = physical:

The device assignments can be altered with this command. However, it is more likely that the CONFIG program would be used instead since the devices are named in real terms rather than as codes.

For those familiar with CP/M who are used to using the STAT command, the following table shows the correspondence between the physical devices and those implemented:

Logical device	Physical device			
LST:	TTY: Serial (printer)	CRT: LCD display	LPT: RS-232C	UL1 not implemented
PUN:	TTY: not implemented	PTP: LCD display	UP1: RS-232C	UP2: not implemented
RDR:	TTY: Keyboard	PTR: not implemented	UR1: RS-232C	UR2: not implemented
CON: Output Input	TTY: RS-232C Keyboard	CRT: LCD Keyboard	BAT: RS-232C RS-232C	UC1: RS-232C RS-232C

For instance, to tell the computer that the printer is now attached to the serial port instead of the RS232-C port, the command

**C>STAT LST : = TTY :**

is given, after which all output destined for the printer is sent to the serial port instead of the RS-232C port. Then, if the STAT DEV: command is given, the result will be:

```
C>STAT DEV:
CON: 1s CRT:
RDR: 1s UR1:
PUN: 1s UP1:
LST: 1s TTY:

C>
```

Further details of the physical devices assigned to the logical devices are given in section 3.6.

### 2. DISK CHARACTERISTICS

**Format** STAT DSK:

The complete status of a disk is displayed using this command. It shows the status of both the current disk and that of any others that have been accessed during the same session:

This command displays the characteristics of the disks accessed, such as its capacity for example:

**A>C:STAT DSK:**

<b>A:</b> Drive characteristics	(drive name)
<b>72:</b> 128 Byte Record Capacity	(no. of 128 byte records allowed)
<b>9:</b> Kilobyte Drive Capacity	(formatted capacity of drive)
<b>16:</b> 32 Byte Directory Entries	(no. and size of directory entries)
<b>0:</b> Checked Directory Entries	(no. of checked directory entries)
<b>128:</b> Records/Extent	(no. of records per extent)
<b>8:</b> Records/Block	(no. of records per block)
<b>64:</b> Sectors/Track	(no. of sectors per track)
<b>0:</b> Reserved Tracks	(no. of tracks reserved for CP/M)

### 3. HELP

**Format** STAT VAL:

This acts as a sort of HELP command. It shows the formats of the various STAT commands which can be given to obtain information or alter device attributes and assignments:



C>STAT VAL:

Temp R/O Disk: d: = R/O  
Set Indicator: d:filename.typ \$R/O \$R/W \$SYS \$DIR  
Disk Status: DSK: d:DSK:  
User Status: USR:  
I/O byte Assign:  
CON: = TTY: CRT: BAT: UC1:  
RDR: = TTY: PTR: UR1: UR2:  
PUN: = TTY: PTP: UP1: UP2:  
LST: = TTY: CRT: LPT: UL1:

#### 4. DISPLAY DEVICE ASSIGNMENTS

**Format** STAT DEV:

This gives the details showing which logical devices have which physical devices assigned to them. To check the devices by name, use the CONFIG program.

```
C>STAT DEV:
CON: is CRT:
RDR: is UR1:
PUN: is UP1:
LST: is LPT:
C>
```

#### 5. READ/ONLY - PROTECT ALL THE FILES ON A DISK

**Format** STAT drivename: = R/O

An entire disk can be set to read/only with this command. It remains effective until either a warm or a cold start is made:

C>STAT A: = R/O

The Read/Only command protects the files on a disk so that they cannot be erased or written to. If you try to write or erase a file with a R/O attribute set the error message:

```
A>ERA LETTER.DOC
BDOS ERROR ON A:R/O
```

will be displayed.

Pressing either the **RETURN** key or the **STOP** key or **CTRL** - **C** will return

you to the prompt.

It is also possible to protect single files and unprotect them using STAT.

#### 6. READ/ONLY - PROTECT A SPECIFIED FILE

**Format** STAT drivename:filename.filetype \$R/O

Any file can be set to read/only using this format. This will prevent any alteration of the file until it is reset to read/write:

```
C>STAT A:DOCUMENT.TXT $R/O
```

DOCUMENT.TXT set to R/O

#### 7. READ/WRITE - REMOVING PROTECTION

**Format** STAT drivename:filename.filetype \$R/W

This resets a file to allow it to be written to as well as read from.

```
C>STAT A:DOCUMENT.TXT $R/W
```

DOCUMENT.TXT set to R/W

If a file is protected, and subsequently the whole disk is protected, the file will still be protected when a warm or cold boot is made to remove the protection from the disk.

#### 8. HIDE FROM DIRECTORY - SPECIFIED FILE

**Format** STAT drivename:filename.filetype \$SYS

It is possible to give a file SYStem status using this command. This effectively removes its name from the directory so it cannot be used by anyone who does not know it exists:

```
C>STAT A:DOCUMENT.TXT $SYS
```

DOCUMENT.TXT set to SYS

If a STATus is carried out on the file when it is set to a SYS file it shows with the filename in brackets

C>STAT A:DOCUMENT.TXT

Recs	Bytes	Ext	Acc	
41	6k	1	R/O	A:(STAT.COM)

Bytes Remaining On A: 3k

### 9. REPLACE IN DIRECTORY - SPECIFIED FILE

**Format** STAT drivename:filename.filetype \$DIR

This countermands the previous format, resetting the file so that it shows on the directory:

C>STAT A:DOCUMENT.TXT \$DIR  
DOCUMENT.TXT set to DIR

### 10. SIZE AND ATTRIBUTES - SPECIFIED FILE

**Format** STAT drivename:filename.filetype

The size and attributes of the specified file are displayed using this command. It gives specific information about the number of records (Recs), number of bytes (Bytes), number of extents (Ext) and read/write status (Acc) of each file on the disk, then the total number of unused bytes remaining. A complete file name can be given, or wildcards can be used to specify a number of files:

#### a) Information on a particular file

C>STAT A:TESTING.COM

Recs	Bytes	Ext	Acc	
16	4k	2	R/O	A:TESTING.COM

Bytes Remaining On A: 4k

#### b) Information on all files

C>STAT C:\*. \*

Recs	Bytes	Ext	Acc	
64	8k	1	R/W	C:CONFIG.COM
22	3k	1	R/W	C:FILINK.COM
58	8k	1	R/W	C:PIP.COM
41	6k	1	R/W	C:STAT.COM

10	2k	1	R/W	C:SUBMIT.COM
24	3k	1	R/W	C:TERM.COM
6	1k	1	R/W	C:XSUB.COM

Bytes remaining on C: 33k

#### c) All files with a particular extension

C>STAT A:\*.COM

Recs	Bytes	Ext	Acc	
58	8k	1	R/W	A : PIP.COM

Bytes remaining on A: 0k

#### d) Files containing particular characters

A>STAT D:DEMO??.BAS

Recs	Bytes	Ext	Acc	
20	3k	1	R/W	D:DEMO1.BAS
16	2k	1	R/O	D:DEMO13.BAS
14	2k	1	R/W	D:DEMO1A.BAS

Bytes remaining on D: 258k

### 11. SIZE AND ATTRIBUTES - SPECIFIED FILE

**Format** STAT drivename:filename.filetype \$\$

Using this form of the command will give the same information as without the \$\$ option, with the addition of the size of the file. This value is the same as the number of records for sequential access files, and is generally used to show the amount of space that has been reserved for a random access file. This is because a sequential file simply takes up space as it is added to, and a random access file has an amount of space allocated to it when it is created:

```
C>STAT A:DOCUMENT.* $$
Size Recs Bytes Ext Acc
1 1 1k 1 R/W A:DOCUMENT.BAS
1 1 1k 1 R/W A:DOCUMENT.TXT
Bytes Remaining On A: 6k
C>
```

## 12. SPACE REMAINING ON DISK

**Format** STAT

Giving this command at the prompt will result in a display of the amount of space available on the current drive and on any other drives which have been accessed during the current session. It also shows the read/write attribute:

a) C>STAT

C: R/O, Space: 33k

b) C>STAT

A:R/W, Space: 0k

C:R/O, Space: 33k

H:R/W, Space: 24k

## 13. SPACE REMAINING ON SPECIFIED DISK

**Format** STAT drivename:

This form of the command gives the amount of space remaining on specified drive:

C>STAT A:

Bytes Remaining On A: 0k

## 14. USER STATUS

**Format** STAT USR:

This format is used to display the number of the current user and the numbers of the users who have active files on the disk:

A>STAT USR:

Active User : 0

Active Files : 0 1

This gives the information that the current user is number 0 and that users 0 and 1 both have active files on the disk.

## SUMMARY OF STAT COMMANDS

STAT	SPACE REMAINING ON DISK
STAT DEV:	LOGICAL TO PHYSICAL ASSIGNMENTS
STAT drivename:	SPACE REMAINING ON SPECIFIED DISK
STAT drivename: = R/O	READ/ONLY - SET SPECIFIED DISK
STAT drivename:filename.filetype	SIZE AND ATTRIBUTES - SPECIFIED FILE
STAT drivename:filename.filetype \$DIR	REPLACE IN DIRECTORY - SPECIFIED FILE
STAT drivename:filename.filetype \$R/O	READ-ONLY - SET SPECIFIED FILE
STAT drivename:filename.filetype \$R/W	READ/WRITE - SET SPECIFIED FILE
STAT drivename:filename.filetype \$\$	SIZE AND ATTRIBUTES - SPECIFIED FILE
STAT drivename:filename.filetype \$\$SYS	REMOVE FROM DIRECTORY - SPECIFIED FILE
STAT DSK:	DISK STATUS
STAT logical: = physical:	CHANGE DEVICE ASSIGNMENTS
STAT USR:	USER STATUS
STAT VAL:	HELP

## 3.8.5 SUBMIT/XSUB

Two very useful commands available in CP/M are SUBMIT and XSUB. Suppose you wish to execute the same commands in a certain sequence at various times. Instead of having to type in the commands one by one you can put them all into a special file with the extension .SUB. Then when you want that series of commands executed you can simply type

```
C>SUBMIT filename
```

and they will all be executed one by one. You can use the XSUB command to allow for input from the keyboard when running programs listed in the SUBMIT file by inserting it at the beginning of the file.

The most useful aspect of having a SUBMIT file facility in CP/M on the PX-8 is to enable a whole sequence of files to be run from an AUTOSTART or WAKE string. Section 2.2.3 shows how to set up these strings.

### 1. SUBMIT

First of all you need to make a list of the programs you want to run. These must all be programs which will return to the CP/M system prompt when they end because otherwise the next command in the sequence cannot be executed. It is possible to run other types of programs, or ones which require input from the keyboard, providing you are present to type in at the keyboard when necessary.

The files containing the commands can be created in a variety of ways - any word processing program such as Portable WordStar™, a BASIC program or using the CP/M ED program if you have it.

Here is an example of a simple batch of jobs to be carried out:

1. Display a directory of the disk on the D: drive;
2. Show the sizes of the files;
3. Erase all those with the extension .BAK (that is, back-up files);
4. Move all BASIC files to the disk in the E: drive.

The sequence of commands to do this would be:

```
DIR D:  
STAT D:*.*
```

```
ERA D:*.BAK  
PIP E:=D:*.BAS
```

If this series of commands were held in a file called START.SUB the command to execute them will be:

```
A>SUBMIT START
```

The commands will be executed one by one, and each one will be shown on the screen before it is run, just as if you had typed it in from the keyboard.

If you want to run a BASIC program from SUBMIT, remember that when such a program ends it returns to the BASIC 'Ok' prompt instead of to CP/M, so the next command in the SUBMIT file will not be executed. One way round this is to insert a line in the program:

```
65000 SYSTEM
```

This will ensure a return to CP/M at the end of the program.

If one or more of the programs you want to be executed may vary, it is possible to insert variables into the SUBMIT file which can be substituted when the submit program is run. The variables symbols are inserted in the form "\$1", "\$2", "\$3" etc, where "\$1" is the variable symbol used for the first substitution, "\$2" for the second and so on. The values of the variables are substituted according to the following syntax:

```
C>SUBMIT filename X1 X2 X3.....
```

where X1 will substitute all values of "\$1", X2 all values of "\$2" and so on.

For instance if the SUBMIT file BEGIN.SUB contained

```
DIR D:*.BAS  
B:BASIC $1  
STAT D:*.BAS
```

the command given could be

```
C>SUBMIT BEGIN D:PROG1
```

This has the effect of inserting the filename you have given (D:PROG1) into the variable "\$1" in the SUBMIT file. The result would be to display a directory of the disk in drive B: and then load BASIC from drive B: before running the BASIC program PROG1 on the disk in drive D: and then showing the BASIC files on drive D:.

There can be any number of variables in the SUBMIT file, and you must ensure that the filenames you give are in the correct order to be substituted at the appropriate time. For instance, if you have a SUBMIT file TESTING.SUB containing the following commands:

```
DIR
$1
DIR
$2
STAT
$3
DIR
```

the command you give to execute it with the .COM files TESTA, TESTB and TESTC in that order should be:

```
C>SUBMIT TESTING TESTA TESTB TESTC
```

so that they are substituted in the correct order.

Some CP/M programs (PIP for example) use the "\$" character in their subcommands, and if this will mean they will be interpreted as variables. Consequently care must be taken as follows to overcome this by adding an extra "\$" whenever the subcommand uses a "\$". SUBMIT will then ignore the first one and then treat the second one as normal. For example to insert the command to cause STAT to set all files on disk A: as Read/Only use:

```
STAT A:*. * $$R/O
```



**WARNING:**

*The SUBMIT program writes a temporary file when it runs. This file will be written to drive A: and if the disk is write protected, an error will be generated and you will not be able to proceed further. Similarly if the RAM disk A: is set to zero bytes a "DIRECTORY FULL" error will be generated.*

### 1.1. Creating a SUBMIT file in BASIC

Since the PX-8 is not supplied with the CP/M program ED, and you may not have a wordprocessor program, BASIC may be the only means of creating a SUBMIT file. This can be done very simply by creating a sequential text file and writing the commands to it as data. Chapter 5 of the BASIC Reference Manual gives details of creating text files. For instance, the series of commands given above could be put into the file called START.SUB with the following program:

```
10 OPEN "O", #1, "D:START.SUB"
20 PRINT #1, "DIR D:"
30 PRINT #1, "STAT D:*. * "
40 PRINT #1, "ERA D:*.BAK"
50 PRINT #1, "PIP E:=D:*.BAS"
60 CLOSE
70 END
```

Or, put another way:

```
10 OPEN "O", #1, "D:START.SUB"
20 READ A$
30 IF A$="END" THEN 60
40 PRINT #1, A$
50 GOTO 20
60 CLOSE
70 END
80 DATA "DIR D:","STAT D:*. *","ERA D:*.BAK";PIP E:=D:*.BAS";END"
```

Either of these programs will create a file called START.SUB containing the commands you want which can then be executed from the CP/M prompt by typing:

```
C>SUBMIT d:START
```

You could even write a utility program in BASIC to allow you to create a SUBMIT file at any time by putting in the commands as you go and then storing them in a .SUB file of your choice:

```
10 INPUT "Type in drivename and filename ";F$
20 OPEN "O", #1, F$ + ".SUB"
30 INPUT "Enter command or END ";A$
```

```

40 IF A$ = "END" THEN 70
50 PRINT #1, A$
60 GOTO 30
70 CLOSE
80 END

```

### 1.2 Creating a SUBMIT file with Portable WordStar™

You can create a SUBMIT file with any word processing program, but the example given here deals specifically with Portable WordStar™. In fact the procedure will be almost identical regardless of which word processor you use.

First of all you must create a new document, giving it the name.

Filename.SUB

If you do not give the extension .SUB the program will not allocate the extension .SUB and the file you have created will not work with SUBMIT.

Now all you have to do is type in the commands you want, one on each line and each followed by a carriage return:

```

A:FILENAME.SUB PAGE 1 LINE 5 COL 01
L-----!-----!-----!-----!-----!-----!-----R
DIR D :  <
STAT D:*. *  <
ERA D:*.BAK  <
PIP E:=D:*.BAS  <

```

Once you have saved this back to disk you will be able to execute it in the normal way using the SUBMIT command.

### 1.3 Adding Comments to SUBMIT Files

In many cases SUBMIT files will be short and easily understood. As you use SUBMIT more, you will realise how powerful a program it is and build up more complex files. It is often difficult to understand why they were created in a specific fashion, and also if someone else wishes to understand what you have done, they may not be able to follow your file. This may be overcome by adding comments to the SUBMIT file which are not executed, but can be found by loading the SUBMIT file into Portable WordStar™ or another word processor, or using the TYPE command of CP/M to read the file. Comments can be added

by preceding the comment by a semicolon.

For example:

```

DIR D: ;first show a directory of disk in drive d:
STAT D:*. * ;then show the statistics of the files on the disk
ERA D:*.BAK ;and erase all backup files on drive d:

```

## 2. XSUB

Some CP/M programs expect information to be typed in at the keyboard when the program has been loaded. The CP/M utilities DDT and ED are two such programs. They can only be used in this way. In order to use them with SUBMIT it is necessary to use the XSUB program with SUBMIT.

If the PIP program is to be used to carry out a number of operations, it is normally loaded first and then the individual operations typed from the keyboard after the PIP asterisk prompt (\*). These subcommands can be included in a SUBMIT if the XSUB program is included also. To use XSUB it is typed into the SUBMIT file as the first command. For example, the operations

1. Move the files to be updated from the D: drive to the E: drive;
2. Execute the program A:UPDATE.COM;
3. Move the updated files to the H: drive;
4. Move the original files to the E: drive as back-up files;
5. Erase the original files from the D: drive.

would be executed by a SUBMIT file containing the following commands:

```
XSUB
PIP
E: = D:RECORDS.DAT
^C
A:UPDATE
PIP
H: = E:RECORDS.DAT
E:RECORDS.BAK = D:RECORDS.DAT
^C
ERA D:RECORDS.DAT
```

While this file is executing you would see the PIP \* prompt appear when PIP was being executed, and the names of the files being moved exactly as you entered them in the SUBMIT file, as would be the case if you typed them from the keyboard. The fourth and ninth commands are 'C' which means "CTRL-C" — this is the command given to exit from PIP, and has to be given in the SUBMIT file to allow return to the CP/M prompt. An alternative is 'M' (CTRL-M) which is a carriage return and has the same effect of leaving PIP and going back to CP/M. When PIP has exited to the CP/M command line, the program UPDATE will be executed. Such a program MUST exit normally to the CP/M command line when being used in a SUBMIT file. When the program UPDATE has finished, PIP can carry out a further transfer of records.

## 3.8.6 CONFIG

The CONFIG program is used to set those system parameters which are not changed very often. It is complementary to the System Display, and the current values of some of the parameters changed by the CONFIG program are shown on the System Display. The CONFIG program can also be used to check the settings of parameters not shown on the System Display.



### WARNING:

*Do not switch off the PX-8 (either manually or by allowing the auto power off to be activated) after changing the RAM disk or USER BIOS size without exiting from the CONFIG program. The full configuration can only be carried out by CP/M. If the PX-8 is switched off, the RAM disk could be destroyed and it may also be necessary to re-initialise the system. You will see repeated message of "RAM disk format (Y/N)" until you press Y(es) key.*

When you have entered CONFIG, either from the MENU or CP/M command line, the screen will show the following:

```
*** MAIN MENU ***                CONFIG V1.0
      Select alphanumeric or ESC to exit.
1=auto power off                 7=RAM disk
2=CP/M function key             8=RS-232C
3=cursor & function key display 9=screen mode
4=date & time                   A=serial
5=disk drives                   B=user BIOS
6=printer                       C=country
```

The CONFIG program is used in a similar way to the System Display, but with 12 options. As with the System Display the ESC key is used to move back to the main menu and to exit.

### 1. AUTO POWER OFF FUNCTION

Press the 1 key from the main CONFIG menu to change the auto power off time. The screen will clear and change to:

```
*** AUTO POWER OFF ***
      Set time(1 to 255) in minutes or ESC to return (0 disables auto power of
      Auto power off time ?

auto power off      : 10
```

The value shown against the semi-colon on the fifth line is the current setting in minutes for the auto power off time. This is the time for which the PX-8 will wait before switching itself off in continue mode if no key is pressed. It is advisable to keep the time short, otherwise the battery will run down. The reason for having such a system is to conserve the battery.

As can be seen from the display, the time can be set in intervals of one minute (up to 255 minutes) or switched off altogether using the value 0. When you have entered the time you require, press RETURN. The chosen time will be displayed on the fifth line, and you can use the ESC key to return to the main CONFIG menu.

## 2. The CP/M FUNCTION KEY ASSIGNMENTS

When the 2 key is pressed on the main CONFIG menu, the display changes to:

```
*** CP/M FUNCTION KEY ***
Select function key number (PF10=0) or ESC to return.

PF1  dir
PF2  type
PF3  stat
PF4  pip
PF5  config^M

PF6  dump
PF7  submit
PF8  ddt
PF9  term^M
PF10 filink^M
```

These are the strings assigned to the programmable function keys on the top of the keyboard. A description of the use of the keys is given in Chapter 2 section 2.2.1c. The screen above shows the default setting, i.e. the strings which will be assigned when the system is initialized or a reset is performed.

Note that when shown on the eighth line of the CP/M screen, some of the strings terminate in the ↵ character. In setting them up with the CONFIG program, this carriage return is shown as “^M”. These strings have a carriage return (CTRL-M) added to them because there is no possibility that any more characters need to be typed after the string. For example the PF5 key can be used to run the CONFIG program as follows. On the CP/M command line type the drive name where the CONFIG program is located (e.g. C:) and then press the PF5 key. All the letters of the word CONFIG will appear, and the carriage return will be ‘typed’ also. In a few seconds the CONFIG main menu will appear on the screen.

The CTRL-M is not added to all commands since some of them might possibly require extending. For example with DIR you might want to add the name of

a drive other than the current one to find out the files on that particular drive instead of changing the logged in drive first. When the [PF1] key is pressed, the letters “DIR” are printed, and the PX-8 will wait for any further input. If you simply press the [RETURN] key, the directory of the current drive will be printed. If you press the space bar and then the two keys “C:” followed by the [RETURN] key, the directory of the C: drive will be printed on the screen.

To change the assigned string, press the number corresponding to the PF key (i.e. 1 for PF1, 2 for PF2 etc, remembering that 0 is used for PF10). The second line of the screen will display the message:

### Terminate function key string with HELP

and the third line of the screen will then display the name of the key whose function is to be changed, with the cursor to the right of it awaiting input. Up to 15 characters can be inserted into each PF string. If a control key is to be added, e.g. a carriage return (CTRL-M), this MUST be added by pressing the [CTRL] and the appropriate alphabetic key. However, in the special case of the carriage return simply pressing the [RETURN] key will add the characters “^M” to denote the CTRL-M for the carriage return. This counts as one character of the string, although it is displayed as two (“^” and “M”) in this section of the CONFIG program. If a control character has been added correctly, the backspace ([BS]) key will remove it as a pair if the [BS] key is pressed once. When a carriage return is inserted correctly it will be seen on the function key display line as a ↵ character. If the characters “^M” appear on the function key display line, they are represented as those characters and DO NOT REPRESENT A CARRIAGE RETURN. For example if the function key display line shows “DIR^M” for one of the keys when the corresponding key is pressed, all five characters will be displayed and the cursor will be placed at the right of the last one awaiting further input.

Although the string can consist of up to 15 characters, if more than seven are assigned to the string, only the first seven will be seen on the function key display line. However, if the function key is pressed the complete number of characters will be printed to the screen.

If a wrong character is input, the backspace key ([BS]) can be used to erase the previous character.

Because the [RETURN] key can be used to enter the carriage return as a “^M”, the [HELP] key is used to terminate the string. The [ESC] key is used to return to the CONFIG main menu.



### 3. SETTING THE CURSOR AND THE FUNCTION KEY DISPLAY

If key 3 is pressed from the CONFIG main menu, the display changes to:

```
*** CURSOR & FUNCTION KEY DISPLAY ***
  Select alphanumeric or ESC to return.

cursor tracking   : on           1=on           2=off
cursor display   : on           3=on           4=off
cursor type      : block,blink  5=block,blink  6=block,nonblink
function key display: off       7=underline,blink 8=underline,nonblink
                                          9=on           A=off
```

The left of the screen shows the current status of the various parameters which can be altered by this section of the CONFIG program. The right part of the screen shows which keys will change them. The parameter setting is altered simply by pressing the key. For example if the 9 key is pressed, the left hand side of the screen will change to show the function key display to be 'on' and whenever the CP/M command line is displayed, the assignments to the function keys will be shown on the base of the screen. The display can be switched off using the A key on this option of the CONFIG program.

This option also allows the type of cursor displayed to be changed, by typing any of the keys 5 to 8. These keys do not affect the cursor on the command line of the MENU and System Display.

Keys 3 and 4 switch the cursor on and off. This affects both the cursor on the CP/M command line and the cursor on the MENU command line.

Keys 1 and 2 switch the tracking mode on and off. The tracking mode means that the cursor follows the window as the real screen moves over the virtual screen. The non-tracking mode means that the window is locked on a particular part of the virtual screen and the cursor (where input is made, or the PX-8 prints the next character) will move along the virtual screen, and once it moves outside the window will not be visible. The tracking and non-tracking modes can also be set using the **SCRN** key. Setting the non tracking mode with the CONFIG program will set the window at the top of the screen. The **SCRN** key can be used to set the window anywhere on the virtual screen.

Futher details of using the tracking and non tracking modes are given in section 2.2.6.

#### NOTE:

*In the screen mode 3, the type of cursor is always set to "underline, nonblink" and cannot be changed with the CONFIG program.*

### 4. SETTING THE DATE AND TIME

Setting the date and time can be achieved by taking option 4 of the CONFIG main menu. The time can be set more accurately than in the initialization question when using either the sub-CPU or initialization reset. The time is entered the moment the **RETURN** key is pressed. When the date is input calculation of the day of the week is carried out automatically.

When option 4 is taken from the CONFIG main menu, the display changes to:

```
*** DATE & TIME ***
  Set date and time or ESC to return.
  Date as MM/DD/YY ?

Date      : 00/00/00 (SUN)
Time     : 18:13:19
```

Although the PX-8 is waiting for the date to be input, simply pressing the **RETURN** key will switch to asking for the time to be input. As soon as the time option is selected, the time is no longer updated on the sixth line. Change the time by inputting the hours, minutes and seconds separated by a colon. If any of the data is a single digit number, it does not have to be preceded by a zero; however, data must be input for the hours minutes and seconds or an error will be detected and the input line will be cleared. When the correct time as required has been entered press the **RETURN** key to enter it into the memory of the computer. The time will be updated when the **RETURN** key is pressed. If an error is made the **BS** key can be used to erase the characters to the left of the cursor.

When the **RETURN** key is pressed after setting the time, or directly on selecting option 4 from the CONFIG main menu, the date can be input as month and day followed by the last two digits of the year, each item being separated by a backslash "/" character. The date will then be entered when the **RETURN** key is pressed. The input will then be entered into the memory of the PX-8 to update the date. If an error is made, such as trying to enter a day of the month greater then the number of days in that month, the input line clears and awaits a correct input.

When a date has been entered on the screen the display changes to input a time. Neither the time or date is updated at this stage. To see if the correct date has been entered, press **RETURN** once more to change the input line to date input.

The date will be updated on the screen when the time is first updated as the seconds change.

Pressing the **[ESC]** key at any time returns to the CONFIG main menu.

## 5. ALLOCATION OF DISK DRIVES

Selecting option 5 on the CONFIG main menu will cause the screen to change to:

```
*** DISK DRIVES ***      Select number or ESC to return.
disk drives : 1          1= A: RAM disk   2= A: FDD1       3= A: FDD1
                        B: ROM1       B: RAM disk   B: FDD2
                        C: ROM2       C: ROM1       C: RAM disk
                        D: FDD1       D: ROM2       D: ROM1
                        E: FDD2       E: FDD2       E: ROM2
                        F: FDD3       F: FDD3       F: FDD3
                        G: FDD4       G: FDD4       G: FDD4
```

This menu makes it possible to allocate the logical disk drives to the physical disk drives in one of three ways. The three tables on the right show the ways in which the allocations are made. The default drives are set as in table 1. The number to the right of the first colon on line two shows which table is selected. Pressing the 1, 2 or 3 keys will change the selected table, and the selected set will then be shown on line 2. The ESC key is used to return to the CONFIG main menu, and keys other than 1,2 3 and ESC are ignored.

It is only possible to reallocate the drives in these three ways. The Microcassette drive is always drive H: and so is not shown on this menu. The terms FDD1, FDD2, FDD3 and FDD4 refer to additional Floppy Disk Drives which are connected to the PX-8 via the serial interface. The ROM drives are shown in section 4.1.2.

## 6. PRINTER INTERFACING

It is possible to drive a serial printer from either of the interface ports. Selecting option 6 on the CONFIG main menu changes the display to:

```
*** PRINTER ***
Select number or ESC to return.

printer I/F      :RS-232C      1=RS-232C      2=serial
```

The fourth line shows which option is selected. The 1 and 2 keys can be used to change the selection. Details of using a printer with the PX-8 are given in Chapter 4.

As with the other sub-menus, after checking which interface is selected, or changing the selection, the **[ESC]** key is used to return to the CONFIG main menu.

## 7. RAM DISK

This option can be used to change the RAM disk size if part of the memory is being used for file storage.

If option 7 is taken from the CONFIG main menu the screen display changes to:

```
*** RAM DISK ***
Set RAM disk size or ESC to return.
RAM disk size ?

RAM disk size      : 9 kb      Max. RAM disk size is 24 kb.
```

The current RAM disk size is shown on the fifth line, and can be changed by typing in the new size. If the size is decreased any files already on the RAM disk will be destroyed. If the RAM disk size is increased the files will remain. The value of the RAM disk size is entered into the memory of the PX-8 by pressing **[RETURN]**. If the value is greater than allowed, the input line will be cleared. If the value input is less than the current value the fourth line will display the message:

**RAM disk files will be destroyed (Y/N) ?**

followed by a flashing cursor. If the Y key is pressed, the new size RAM disk will be set up, destroying the files currently saved there. If the N key is pressed, the fourth line will clear and place the cursor next to the RAM size input message.

The maximum size the RAM disk can be extended to is shown on the right of the fifth line. The above display shows the maximum possible (24 kb) which occurs when no RAM is reserved for USER BIOS.

If a USER BIOS area has been reserved, the value of the maximum value will reflect this. The total USER BIOS and RAM disk must not exceed 24 kb.

If an Intelligent RAM disk is connected it is not possible to use part of the memory of the PX-8 to extend the size of this RAM disk. For example, if the RAM Disk Unit 120 is connected, when option 7 is chosen from the CONFIG main menu, the screen changes to show:

```
*** RAM DISK ***
An external RAM disk is connected. ESC to return.

RAM disk size      : 120 kb
```

The ESC key can then be used to return to the CONFIG main menu. Further details of the Intelligent RAM disk can be found in Chapter 4.



**WARNING:**

*Do not switch off the PX-8 (either manually or by allowing the auto power off to be activated) after changing the RAM disk size without exiting from the CONFIG program. The full configuration can only be carried out by CP/M. If the PX-8 is switched off, the RAM disk could be destroyed and it may also be necessary to re-initialise the system.*

## 8. CHANGING THE RS-232C INTERFACE PARAMETERS

Option 8 on the CONFIG main menu allows the settings of the RS-232C interface to be set.

The RS-232C interface is used to transmit data to and from the PX-8. For example if a text file has been written on the PX-8 and the data needs to be used on a desk top computer such as the EPSON QX-10, the file can be sent to the other computer using the TERM or FILINK program in the CP/M UTILITY ROM. This transmission can be by cable directly to the other computer, or over a telephone line using an acoustic coupler.

The RS-232C interface or the serial interface can be used to connect to a printer.

The printer must have a serial interface in order to receive the information. This allows program listings as well as letters/reports or other text data from applications programs to be printed for a permanent record.

In both these cases the parameters of the interface have to be matched. This option on the menu allows the parameters to be set. In general they should be set or checked with this option before using the communications program (e.g. TERM or FILINK). Further details of the RS-232C interface are given in Chapter 4.

When option 8 is chosen from the CONFIG main menu the screen changes to display:

```
*** RS-232C ***
Select alphanumeric or ESC to return.

bit rate : 4800      1=19200    2=9600    3=4800    4=2400    5=1200    6=600
              7=300      8=150     9=110     A=75/1200 B=1200/75 (Tx/Rx)
data bits : 8
parity  : none      E=none    F=odd     G=even
stop bits : 2       H=1       I=2
```

The bit rate (bits per second or baud rate) currently set is shown on the fourth line. It can be changed to various settings using keys 1 to 9 and keys A and B. The settings produced by keys 1 to 9 set transmission rates in both directions (transmit and receive). The A and B keys set a rate which is different for transmitting and receiving. The A key sets 75 baud transmit (Tx) and 1200 receive (Rx) with the reverse for key B.

The number of data bits is shown on the sixth line and can be changed using the C and D keys.

The parity is shown on the seventh line and can be changed using the E, F and G keys.

The number of stop bits is shown on the eighth line and can be changed using the H and I keys.

When the parameters have been set or checked, the ESC key returns the PX-8 to the main menu.

## 9. Screen Mode

This option can be used to change the screen configuration. Details of the types

of screen and its use are given in Chapter 2 (section 2.2.7). There is also a detailed detailed practical guide to using the different screen modes in the BASIC Reference Manual (Chapter 2 section 2.14).

When option 9 is chosen from the main CONFIG menu, the screen changes to:

```
*** SCREEN ***
  Set screen configuration or ESC to return.
  Screen mode (0,1,2,3) ?
screen mode      : 0
virtual screen 1 : 80 x 25
virtual screen 2 : 80 x 23
selected screen  : 1
separation character:
```

The cursor is located next to the prompt on the third line of the screen. The rest of the screen shows the current configuration of the screen.

Since different screen modes show and require different configurations they will be treated individually as follows:

#### Screen Mode 0

Pressing the 0 key followed by **RETURN**, will cause the third line of the screen to change to:

##### Number of lines of virtual screen 1 ?

You can then enter the number of lines of the screen you require for this screen in the range 8 to 40. If you attempt to enter value outside this range, the line will clear after the prompt, and you can then enter a correct value. When the **RETURN** key is pressed the prompt line alters to read:

##### Number of lines of virtual screen 2 ?

Again the number of lines of the screen is in the range 8 to 40, but the total of the number of lines on the two virtual screens **MUST** be the less than 48. Also each screen must have a minimum of eight lines, so the maximum lines on a screen is 40 if the minimum is used on either screen. When entering a value for the second screen if the total of the two screens would be greater than 48 with the value you have entered for the second screen, the prompt line will clear so that you can enter a correct value. When **RETURN** is pressed and a valid entry has been made, the prompt line will display:

##### Select virtual screen (1,2) ?

You can then choose which screen will be displayed.

If you wish to change only one of the parameters, simply pressing **RETURN** will enter the current value. Thus if you only wanted to change the virtual screen, the **RETURN** key would be pressed until the required prompt string was displayed.

When the selected screen is entered, the display changes to show the values selected.

Since screen mode 0 can only display one virtual screen at a time, the separation character parameter will not be shown.

#### Screen mode 1

In screen mode 1, both virtual screens have the same number of lines. Thus in answer to the prompt:

##### Number of lines of virtual screen 1 ?

you are in fact choosing the number of lines on both virtual screens. Since also each the virtual screen when displayed is split into two columns of 39 characters with a separation character, a minimum of 16 lines is required on each virtual screens. Also the maximum number of lines on each screen is 48. If a value outside the range 16 to 48 to entered the line will clear from the prompt in order that a correct value can be inserted.

When the **RETURN** key has been pressed and a correct value has been entered, the prompt line changes to:

##### Select virtual screen (1,2) ?

One of the two virtual screens can then be chosen to be displayed as split on either side of the separator. Since the separator is fixed, the screen changes to show the changed values.

#### Screen mode 2

In screen mode 2 both virtual screens are displayed at once. It is also possible

to alter the number of columns of the display of each screen. The number of lines on each screen must be the same.

When the number of lines in the virtual screens have been chosen correctly in the range 8 to 48, the prompt line changes to show:

#### Number of columns of virtual screen 1 ?

It is thus possible to alter the width of the two halves of the display. The number of columns must be in the range 1 to 79. When the width of the left hand side of the screen is chosen the width of the right hand side is set to a value of 79 minus the size of the left hand side.

When the screen width has been set the prompt line changes to:

#### Select virtual screen (1,2) ?

and it is possible to set which screen the cursor lies in. When this has been chosen, the final input required is the separator character. If the **RETURN** key is pressed, the default character is used; this is the same character as is used as separator for the screen mode 1.

On pressing **RETURN** when the separator character has been chosen, the screen shows all the parameters chosen, and it is then possible to use the **ESC** key to return to the CONFIG main menu.

#### Screen mode 3

There is only one setting for the parameters of screen mode 3. The screen has only one virtual screen of 8 lines and 80 columns. Thus entering screen mode 3 and pressing **RETURN** causes the prompt line to revert to the same prompt line, awaiting either the screen to be changed again or the **ESC** key to be pressed to return to the main CONFIG menu.

### 10. THE SERIAL INTERFACE

The Serial interface is used for communication with other devices (in particular printers and floppy disk drives) but not for general communication in the way that the RS-232C interface is. If an EPSON disk drive is connected, the transmission rate is set by the PX-8. If a printer is connected, the transmission rate can be adjusted using CONFIG command.

Selecting option A from the CONFIG main menu shows the following display:

```
*** SERIAL ***
Select number or ESC to return.

bit rate (printer) : 4800          1=4800    2=600    3=150
bit rate (FDD)    : 38400
```

If a printer is being used check that the bit rate setting is the same otherwise unintelligible data will be printed on the printer. Three settings are provided, and they can be changed using the 1, 2 and 3 keys. If the printer you are using does not have a serial port which can be set to any of these values, you will only be able to print using the RS-232C interface of the PX-8, as this has the standard settings from 19200 to 110 baud. Other protocols are set to 8-bit, non-parity, 1 stop-bit.

The Floppy Disk Drive bit rate is shown on the sixth line for reference.

When the parameters have been set or checked, the ESC key returns the PX-8 to the main menu.

### 11. USER BIOS SETTING

The USER BIOS is used by some applications programs. It can also be used by advanced programmers. If you are such a user, please refer to the OS Reference Manual.

An applications program will normally change the USER BIOS size automatically. The manual which describes the program may tell you to alter the USER BIOS manually in which case follow the instructions given.

Normally the setting of the USER BIOS will be 0 pages. If you find it is not zero it will probably have been set by an application program. For example Portable Scheduler™ uses the USER BIOS area to store the alarm times. If you change the size of the USER BIOS area in this case, the values set up will no longer be actioned.



#### WARNING:

*Do not switch off the PX-8 (either manually or by allowing the auto power off to be activated) after changing the USER BIOS size without exiting from the CONFIG program. The full configuration can only be carried out by CP/M. If the PX-8 is switched off, the RAM disk could be destroyed and it may also be necessary to re-initialize the system.*

## 12. CHANGING THE CHARACTER SET BY COUNTRY

The DIP switch is normally used to set the keyboard layout. Occasionally, it is useful to be able to temporarily change the characters but not the keyboard layout so that, for example, a word processed file written in French can be read on the screen with the correct characters. Taking the last option by pressing the key marked C, will enable the character set for the different countries to be displayed. The menu shows:

```

*** COUNTRY ***
Select number or ESC to return.

country          : ASCII          1=ASCII          6=Italy
                                     2=Denmark         7=Norway
                                     3=England        8=Spain
                                     4=France         9=Sweden
                                     5=Germany
    
```

The county whose characters are displayed is changed by pressing the key corresponding to the country in the table on the right of the screen. The currently chosen country is displayed on the fourth row on the left hand half of the screen.

International Character Sets

Country Dec. Code	United States	France	Germany	England	Denmark	Sweden	Italy	Spain	Norway
35	#	#	#	£	#	#	#	£	#
36	\$	\$	\$	£	\$	¤	\$	\$	¤
64	@	@	§	@	£	¤	@	@	¤
91	[	°	ä	[	£	ä	°	ı	£
92	\	¢	ö	\	¢	ö	\	¢	¢
93	]	§	ü	]	¤	¤	£	ı	¤
94	^	^	^	^	ü	ü	^	^	ü
96	'	'	'	'	£	£	ü	'	£
123	<	£	ä	<	¤	ä	ä	ı	¤
124	ı	ü	ö	ı	¢	ö	ö	¤	¢
125	>	£	ü	>	ä	ä	£	>	ä
126	~	ı	ß	~	ü	ü	ı	~	ü

## 3.9 Communications

The PX-8 can be used to communicate with other computers, either directly or using telephone lines. This means that program and data files can be sent from one computer to another directly along a simple pair of wires - the cables and telephone lines connecting the computers are really no more than this. Connecting a printer to a computer is a very simple form of communication, as is the addition of disk drives, so it is not really much more complicated to connect another computer. The length of the pair of wires between the computers is irrelevant so long as the correct equipment is used. For instance, if you want to connect your PX-8 at home to your QX-10 desktop microcomputer at the office, all you would need is an acoustic coupler at each end into which to plug the telephone handset and the appropriate sending and receiving software to run on each computer. It wouldn't matter if one computer was in New York and the other in London - the system would still work.

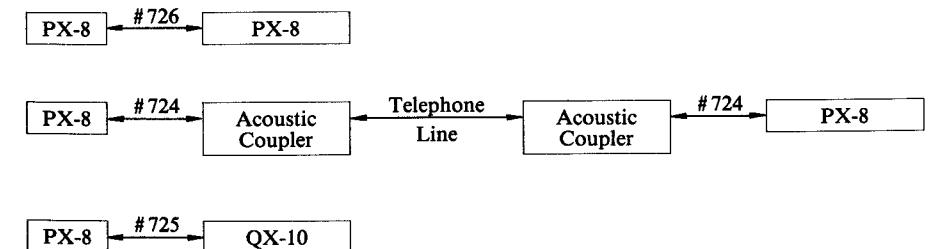
The examples given here relate mainly to communication with the EPSON QX-10 desktop microcomputer, the EPSON acoustic coupler and EPSON printers, although any other computers and printers may be used given the appropriate cables and utility software.

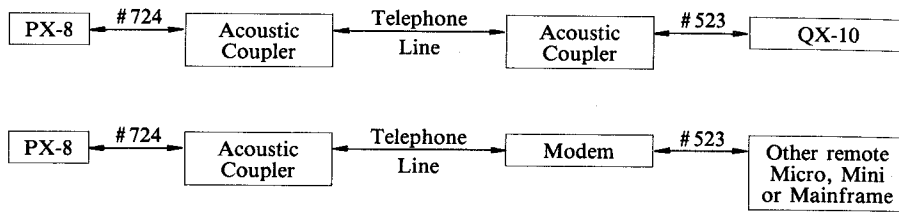
### HARDWARE

When the PX-8 is required to 'talk' to other hardware, the following cables are necessary to connect via the RS232C I/O ports:

	Cable Type
PX-8 to PX-8	# 726
QX-10	# 725
Acoustic Coupler	# 724

There are various configurations possible of which the following are a few:





For an explanation of RS232C communication see Chapter 4.

## SOFTWARE

The two utility programs available on the PX-8 are TERM and FILINK. Various other communications programs are available which work in roughly the same way, and you should consult their respective manuals for instructions as to their use.

### 3.9.1. TERM

The TERM utility is used when it is necessary to 'talk' to a remote computer, using the PX-8 as a terminal, sending single files in either direction or simply using both as 'dumb' messengers. In the following descriptions the second computer is assumed to be an EPSON QX-10 (MultiFont CP/M version) for the sake of clarity, but equivalent software may be used on other machines for this type of communication.

The first thing to do is make sure the PX-8 is connected to the required external devices using the relevant cables, then load the TERM program from the appropriate drive.

When TERM is run the screen shows the following:

```

The RS-232C status is :
bit rate = 4800  data bits = 8  stop bits = 2  parity = NONE
Use CONFIG.COM program to change the RS-232C status.
Modes of TERM
1 = Normal
2 = Delete LF after CR (send)/ Insert LF after CR (receive)
3 = Insert ETX and Delete LF after CR (send)
Select a mode 1
  
```

The top line of the screen shows the default RS232C settings (they may not be the same as the ones shown here). These can be changed using the CONFIG program and should be set before running the TERM program. Make sure the receiving computer has the same settings, and change one or other machine's settings so that they agree.

The three options given on the screen refer to how data is sent and received.

- i) The default is option 1 which merely requires you to press the key, and sets the computer to send and receive data without any modification. This is the most commonly used setting.
- ii) Some computers automatically add a line feed to any carriage return in the data they receive, so in this case you would select Option 2 (delete LF). This will have the effect of deleting line feed codes (ASCII code 10 decimal; hexadecimal 0A) following carriage returns (hex 0D) during transmission, that is, when sending data. It also adds a line feed code immediately after any carriage return code in received data. Remember that a carriage return merely moves the cursor to the beginning of a line - it is the line feed code that moves it to the next line down.

iii) Option 3 (add ETX and delete LF) has the same effect on line feeds as Option 2, but it also adds an 'end of text' code (ASCII code 03 decimal; hexadecimal 03) to each transmitted carriage return after deleting LF code there. When receiving, data is processed as is. This allows communication with computers which require the ETX code instead of the LF code.

When you have selected which mode you wish to use on the PX-8 by pressing **RETURN** key, the screen will display:

```

Modes of TERM
1 = Normal
2 = Delete LF after CR (send)/ Insert LF after CR (receive)
3 = Insert ETX and Delete LF after CR (send)
Select a mode 1

display/      print/      send/      receive/      /exit
  
```

The bottom line of the screen shows the functions assigned to the **PF** keys. When these functions are inactive they appear in lower case characters, and when activated by pressing the relevant key they appear in upper case. These have the following effect:

**PF1 - DISPLAY ON/OFF**

This switches the display ON or OFF while transmission is in progress. Normally, transmitted data does not show on the PX-8's screen, so switching the display on allows you to see what is transmitted as it is sent. If the display is switched on while data is being received (either from the keyboard or from an external source) the data will show twice on the screen. The effect is that anything sent from an outside source instead of appearing on the screen as 'Hello' will appear as 'HHeellloo'. This would also be the case if the display were switched on while using the PX-8 as a terminal.

**PF2 - PRINTER ON/OFF**

Printer output can be switched on or off during reception or transmission to produce a hard copy of the data to be transferred. Thus, whatever appears on the screen will also appear on the printer. Remember that because the TERM program uses the RS232C port the printer must be connected to the serial port. The **PF3** and **PF4** assignments at the bottom of the screen disappear when PRINTER is ON.

**PF3 - SEND A FILE/TERMINATE TRANSMISSION**

A file can be sent to the external device and its transmission ended using this key. This procedure is fully described in 1.4 below.

**PF4 - RECEIVE A FILE/TERMINATE RECEPTION**

Pressing the **PF4** key will make the PX-8 prompt for the name of the file to be received. A full description of this can be found in 1.3 below.

**PF10 - EXIT**

Pressing **PF10** and will cause the PX-8 to exit from the TERM program at any time.

**HELP - OUTPUT BREAK SIGNAL**

Pressing the **HELP** key outputs the BREAK signal to the receiving computer.

*Note:*

*When using TERM only the function displayed on the 8th line can be selected.*

**1.1 SENDING AND RECEIVING MESSAGES**

When transmitting messages between computers it is necessary to have a program to handle the data being transmitted. Since there are many such programs and computers, the EPSON QX-10 is used as an example with the QX-10 TERM program as the executing software.

First of all, execute the TERM programs on both the PX-8 and the QX-10. The QX-10 screen will show:

```

QX-10 TERMinal or Remote ver 1.8

rs232c : bit rate = 300 parity = no stop bit = 1 data char = 8bit

Terminal mode..... 1
Remote mode..... 2
Normal mode..... 3
change rs232c..... X

select 1-3 or X?
  
```

Comparison of the RS232C parameters with the default parameters of TERM on the PX-8 will show that the handshaking does not match. The quickest way to make an alteration is to change the QX-10 parameters by selecting 'X'. When the screen changes to that of fig. below press 'B' and '5' to change the bit rate to 4800 and the stop bits to 2:



### QX-10 TERMinal or Remote ver 1.8

rs232c : bit rate = 300 parity = no stop bit = 1 data char = 8bit  
select A-9 or RETURN ?

bit rate	parity	stop bit	data char
9600 ..... A	no ..... 0	1 ..... 3	0-5bit ..... 6
4800 ..... B	yes even ..... 1	1.5 ..... 4	6bit ..... 7
2400 ..... C	yes odd ..... 2	2 ..... 5	7bit ..... 8
1800 ..... D			8bit ..... 9
1200 ..... E			
900 ..... F			
600 ..... G			
400 ..... H			
300 ..... I			
200 ..... J			
150 ..... K			
135 ..... L			
110 ..... M			
75 ..... N			
50 ..... O			

The parameters given on the PX-8 screen are the default settings, as are those for the QX-10. Either computer's settings can be changed using the CONFIG command, but only the QX-10's can be changed from within the TERM program. However, the QX-10 parameters are only temporarily affected by the TERM program and the QX-10 CONFIG program must be used to alter permanently the QX-10 default parameters. If it is necessary to change the parameters on the PX-8 press **RETURN** then **PF10** (shifted **PF5**) to exit from TERM. Now you can use CONFIG to alter the parameters.

Now select Option 1 (default) on the PX-8 and Option 1 on the QX-10. The QX-10's screen will clear and the message

### QX-10 Terminal Mode

will appear at the top with the flashing cursor beneath it. The PX-8's cursor will move to the beginning of the line above the function key assignment display. Typing a message on the PX-8 will result in the message appearing on the screen of the QX-10. At the end of a sentence or paragraph the **RETURN** key can be pressed which returns the cursor to the beginning of the same line and a line feed can be sent by pressing the **↓** key. Exactly the same thing will happen in reverse if you type in a message from the keyboard of the QX-10. The characters will appear on the screen of the PX-8 - pressing the **RETURN** key will move the cursor back to the beginning of the current line and pressing the **↓** key will move the cursor down a line. Pressing **PF1** on the PX-8 will turn the display on and result in characters input from the PX-8's keyboard showing on both screens instead of just the QX-10 screen. Remember that both computers are completely 'dumb' in this mode and cannot perform any processing. All they do is display on their screens whatever has been typed in on the keyboard of the other computer.

If the printer is switched on using the **PF2** key all output from the QX-10 keyboard will be echoed to the printer.



#### WARNING

*If the printer is not switched on, off line or wrongly connected, the function key display of 'PRINT' will flash.*

*To exit from this mode press **PF10** (shifted **PF5**) on the PX-8 and the **BREAK** key on the QX-10.*

## 1.2 THE PX-8 AS A TERMINAL

This is a very useful function which can be used, among other things, for remote data processing. One instance would be if the QX-10 were in the office and you were in another part of the country. You could leave the QX-10 set up in Remote Mode, plug your PX-8 into an acoustic coupler and thence into the public telephone system, then use the QX-10 with the PX-8 behaving as the keyboard and screen. In this way you could enter data, retrieve it and process it from anywhere you happen to be, at any time!

First of all, make sure the RS232C settings match (see previous Section). then run the TERM program on both computers. On the QX-10 choose Option 2 to enter the Remote Mode. Wait a few seconds and the A> prompt will appear on the screen of the PX-8. This means that the PX-8 is now acting as a 'dumb' terminal to the host QX-10 computer. You can now use the PX-8 exactly as if it were really the QX-10 and perform file manipulation, input and output, programming or any other functions just as though you were using the QX-10's own keyboard and screen. The only constraints are that the screens are a different size (the PX-8 screen behaves like a window on the QX-10 screen) and the keys do not behave in quite the same way - the **PF** key of the PX-8 retain their assignments as part of the TERM program, but the other keys behave as you would expect on the QX-10. In fact, the first draft of this paragraph was typed on a PX-8 acting as a terminal to a QX-10 which was running a powerful word processing program!

If the printer is switched on with the **PF2** key, all output from the host computer will be printed as hard copy as though you had pressed **CTRL - P**.

In order to terminate this host-terminal condition the following operations on the PX-8 must be carried out:

- a) Return to the A> system prompt given by the QX-10;
- b) Make sure the disk containing the TERM.COM program is on the A: drive of the QX-10;
- c) Type TERM to re-enter the Terminal program;
- d) Select Option 3 on the menu - the QX-10 will now return to normal mode;
- e) Press **PF10** to exit from the TERM program on the PX-8.

## 1.3 RECEIVING FILES

Program and data files can be downloaded from an external device to the PX-8 by using the RECEIVE facility. Enter the TERM program in the normal manner, then press **PF4** to select RECEIVE. You will then be prompted for a filename. This should be the filename you wish the file to have on the PX-8, for instance, to send it to the RAM disk call it A:filename.extension or to send it to the microcassette drive call it H:filename.extension. The computer will then wait for input from the RS232C port and direct the file to wherever you have specified.

After all the file was received, press **PF4** key so that the received file may be closed.

It is possible to print the file being received on the printer connected to the serial port, but not at the same time as it is being transferred. If PRINTER ON is selected the printing will not begin until the file has been fully transferred to the PX-8.

If you are using the QX-10 or another CP/M based computer for output, an easy way of demonstrating this is to use the PIP command, for instance

```
A>PIP PUN: = A:FRED.BAS
```

This command tells the QX-10 to send the file A:FRED.BAS to the device called PUN: which is the default output device name for the RS-232C port. The **RETURN** key must not be pressed until the PX-8 is ready and waiting for input. If you have told it to expect a file called H:SALLY.BAS it will receive the file A:FRED.BAS sent by the QX-10 and store it on microcassette tape as H:SALLY.BAS. The contents of the file will appear on the screen of the PX-8 as it is transmitted, and transmission can be terminated by pressing the HELP key. This outputs a Break signal to the transmitting device and tells it to stop sending.

## 1.4 SENDING FILES

The PX-8 can send files to an external device by working the opposite way round to the procedure for receiving a file.

Enter the TERM program in the normal way, then press **PF3** to enter the transmission mode. First the display of assignments for keys **PF4** and **PF2** are suppressed, then the prompt

### Enter file name

appears. Type in the complete filename of the file you wish to send, e.g. H:CHARLIE.BAS.

Next comes a series of three questions relating to time delays to be inserted between blocks of data, as follows:

```
Enter file name      A:DOCUMENT.TXT
Set transmission delay time in 10 ms (Max. 255)
After each character 0
After CR.LF         0
After 128 bytes     0
```

The defaults for these delays are 0 milliseconds, i.e. no delay, and this is the most commonly used setting. Consult the manual for the receiving device and software before deciding if these defaults should be altered. If you find characters are lost, you can try empirically adding a delay, but in normal circumstances you should not need to change the default values. At the receiving end you can use the same method as described in 1.3 above except that this time the RS232C port should be prepared for input. The command

**A> PIP B:KATE.BAS = RDR:**

will read in the PX-8 file H:CHARLIE.BAS through the RS232C port of the other computer and write it to disk drive B: on that computer as B:KATE.BAS.

Another possibility is that you may wish to look at a large screenful of data. To do this you can use the command

**A> PIP CRT: = RDR:**

which will direct the input to the screen.

When data transfer is completed the TERM program on the PX-8 can be exited in the normal way by pressing the **PF10** key.

## Error messages

All errors (except 'Printer not ready', 'File not found' and 'File already exists') which require action will necessitate the TERM program being restarted after the error has been dealt with.

- a) **PRINT** ( **PF2** key display) **blinks** - if you have selected the printer option make sure that:
  - i) It is connected to the serial port;
  - ii) The printer is switched on and on-line.
- b) **RS-232C is not ready** - make sure that RS-232C cable is connected correctly.
- c) **File not found** - the file you have specified for transmission cannot be found. You are given the opportunity to specify the file again - there was probably a typing mistake the first time.
- d) **No file name specified** - specify the file name
- e) **Bad File descriptor** - specify the correct drive and file name.
- f) **Drive select error** - you have specified the drive name other than A: to H:.
- g) **Overwrite (Y/N)?** - This means that the file you have specified already exists but gives you the chance to either overwrite it (by answering Y to the question) or to specify a different filename (by answering N).

- h) **Communication error** - if the RS232C ports of the two computers are not configured with the same parameters the PX-8 will not receive recognisable characters. When data is received from the transmitting computer the screen will show this error message and the prompt:

**Press ESC to restart, STOP to exit from TERM**

You can then take the appropriate action.

### **NOTE:**

*The QX-10 TERM program will cause meaningless characters to appear on the screen if an attempt is made to send to it with non-matching RS232C parameters.*

- i) **Directory full** - this means that the directory on the device to which you want the file written is full, that is, contains the maximum number of entries. You can delete a file, change the disk or tape or specify a different device, and then start TERM again.
- j) **Disk full** - this lets you know that the device to which you are writing the file has no more space on it. In the case of a disk or a tape change it for another one with more room on it. In the case of the RAM disk you will either have to redirect the file to another device or risk losing what is already on it by expanding it with CONFIG. In any case you will have to start the TERM program again.

## 3.9.2 FILINK

FILINK is a more specialised program than TERM in that it supports specific protocols between machines and is used solely for sending and receiving files. It can be used to communicate with the following machines using the specified software:

PX-8 FILINK.COM  
Portable WordStar™ using the 'T' and 'C' commands

QX-10 FILINK.COM

File names can be specified in their entirety or using wildcard characters (\* and ?), unlike TERM which will only support full filenames.

Refer to the OS Reference Manual for details of the communications protocol.

The cables used for communication are the same as those specified in Section 3.9.1 for the TERM command.

### Using FILINK

When the program is run the following screen appears:

```
A file transfer program via RS-232C port.

The RS-232C status is :
bit rate = 4800  data bits = 8  stop bits = 2  parity = NONE
Use CONFIG.COM program to change the RS-232C status.

Press ESC to restart, STOP to exit from FILINK or CTRL/STOP to abort.
Send or Receive (S/R) ?
```

The second line of the screen gives the RS232C settings. These can be changed using the CONFIG program. The fourth and fifth lines give information on special key usage:

**ESC** Pressing the **ESC** key while the program is waiting for key input restarts the program from the beginning.

**STOP** Pressing **STOP** while the program is waiting for key input stops the program and returns control to CP/M.

**CTRL** - Pressing **CTRL** and **STOP** together at any time results in termination of the program and return of control to CP/M.

The sixth line describes the wildcard characters allowed when specifying filenames, and the seventh line asks whether you wish to send or receive. When you have typed in 'S' or 'R' you will be prompted for a filename. This can be typed in as a complete name or using wildcard characters (\* and ?). When receiving files you will have to specify a full filename because in this case you are actually specifying the file into which the data being received is to be put. When sending files, a whole family of files can be sent using the wildcard options, for instance

**D:\*.BAS**

will ensure that all files on drive D: having the extension BAS are sent down the line.

### Error messages

- a) **Directory full** - this means that the directory on the device to which you want the file written is full, that is, contains the maximum number of entries. You can delete a file, change the disk or tape or specify a different device, and then start FILINK again.
- b) **Disk full** - this lets you know that the device to which you are writing the file has no more space on it. In the case of a disk or a tape change it for another one with more room on it. In the case of the RAM disk you will either have to redirect the file to another device or risk losing what is already on it by expanding it with CONFIG. In any case you will have to start the FILINK program again.
- c) **Overwrite (Y/N)?** - This means that the file you have specified already exists but gives you the chance to either overwrite it (by answering Y to the question) or to specify a different filename (by answering N).
- d) **File not found** - the file you have specified for transmission cannot be found. You are given the opportunity to specify the file again - there was probably a typing mistake the first time.
- e) **Receiver is not ready** - make sure you have connected the PX-8 correctly to the external device and that the receiving computer is ready to receive the file.
- f) **No file name specified** - Specify the file name.

- g) **Bad file descriptor** - Specify the correct file name.
- h) **Drive select error** - Specify the drive name from A: to H:.
- i) **Communication error** - if the RS-232C ports of the two computers are not configured with the same parameters the PX-8 will not receive recognisable characters. When data is received from the transmitting computer the screen will show this error message and the prompt:

**Press ESC to restart, STOP to exit from FILINK**

You can then take the appropriate action.

**NOTE:**

*The QX-10 TERM program will cause meaningless characters to appear on the screen if an attempt is made to send to it with non-matching RS-232C parameters.*

- j) **Close error** - replace the new disk and restart from the beginning.
- k) **Sender is not ready** - make sure that the sending computer is ready to send the file.
- l) **RS-232C is not ready** - make sure that RS-232C cable is connected correctly.

## 3.10 Other CP/M Transient Programs

Apart from the utility programs or commands in CP/M which have already been described, there are some others which will be available to you as soon as you connect a disk drive to your PX-8. Some are disk utility programs and some allow you to carry out all sorts of operations on the computer which are either difficult or impossible with the machine on its own. Please consult your dealer for information on obtaining these programs.

### 1. Disk utilities

#### a) **FORMAT** (Preparing a floppy disk for use)

This program allows you to **FORMAT** a floppy disk in a specified drive. **FORMATting** is a process whereby the disk is divided up into sections which help the computer to find its way around. It also has the effect of erasing all the data already on the disk. This function may be included in **COPYDISK**.

#### b) **COPYDISK** (Making an exact copy of a floppy disk)

Instead of using **PIP** you can use **COPYDISK** to copy the entire contents of one disk on to another. This is particularly useful when making back-up copies of program and data disks. This program incorporates a **VERIFY** facility so that it compares what is on the disk from which you are copying with what is being written to the disk on which the copy is being made, thus ensuring a perfect copy.

### 2. Program utilities

#### a) **ED** (Editing a file)

This is an **EDitor** incorporating a powerful set of subcommands which can be used for on-screen creation and editing of files. It is most useful when creating **SUBMIT** files or when typing in or correcting assembler programs.

#### b) **DDT** (Dynamic Debugging Tool)

Programs written in Intel 8080 in assembler or machine code can be quickly debugged or altered with **DDT**. It contains a number of subcommands which can be used to look at, alter and assemble programs in memory. Files changed in this way can be written back to disk using the CP/M **SAVE** command.

c) **ASM** (8080 Assembler)

When an assembler program has been written and given the filename extension .ASM, running this program will assemble it and produce object and list files with the filename extension .HEX and .PRN which are automatically written to disk.

d) **LOAD** (Producing COM files)

When an assembler program has been assembled using the ASM program, the object code file (type .HEX) can be converted into an executable machine code file with the filename extension .COM. You can use this to create your own commands on disk.

e) **DUMP** (Hexadecimal file dump)

Any file on a disk can be shown on the screen in hexadecimal form using this program. Each line contains a 16-byte section of the file with the starting address given at the left hand end.

### 3.11 Finding Out More About CP/M

This manual cannot cover all aspects of using CP/M particularly the finer points. The following books are among many available from your computer dealer or bookshop.

**CP/M and the Personal Computer** by Thos A. Dwyer and Margot Critchfield. This covers CP/M from the beginners point of view as well as containing much information on the internal workings of CP/M and advanced uses. (Published by Addison Wesley. ISBN 0-201-10355-9)

**CP/M Handbook** by Rodney Zaks. This is a general guide to using CP/M. (Published by Sybex. ISBN 089588-048-2)

**Osborne CP/M User Guide** by Thom Hogan. Another general guide. (Published by Osborne/McGraw Hill)

**CP/M BIBLE** - The Authorative Reference Guide to CP/M by Mitchell Waite and John Anglermeyer. As the title suggests, a reference guide to the use of CP/M. (Published by Howard Sams. ISBN 0-672-22015-6)

**Soul of CP/M** by Mitchell Waite and Robert Lafore. This is a guide for the advanced user and programmer. (Published by Howard Sams. ISBN 0-672-22030-X)

**CP/M The Software Bus - A Programmers Companion** by A.Clarke, J.M.Eaton and D.Powys-Lybbe. A further manual for advanced programmers. (Published by Sigma Technical Press. ISBN 0905104-18-8)

## *Chapter 4*

# ***INPUT AND OUTPUT DEVICES AND OPTIONAL ADDITIONS TO THE PX-8***

## **INTRODUCTION**

This chapter deals with the aspects of the PX-8 involved with interfacing the computer to other devices. Some of the devices are an integral part of the PX-8, others can be purchased separately. Epson is continually developing further products and so this manual cannot cover every device it is possible to use with your PX-8. This manual does not mention specific products as in most cases these items vary from country to country. You should consult your dealer for the products available in your country and for details of new optional items.

## **4.1 Storage of Files and Data**

The PX-8 has many means of storing files. There are also ROM capsules on which manufacturers of applications programs can provide their programs. These can be read but not used for storing files. The other important method of storing data is the Microcassette drive. It is also possible to add on floppy disk drives.

### **4.1.1 Microcassette Drive**

The PX-8 is equipped with a built in Microcassette drive. This section describes operation of the drive.

#### **a) Drive name**

The microcassette drive can be used in the same manner as other disk devices. It is assigned drive name H:. It is not possible to assign drive name H: to any other drive, nor to assign any other drive name to the Microcassette unit. Because a cassette tape functions differently from a floppy disk, a special operat-



ing system has been added to the PX-8 CP/M. This is known as MTOS (Microcassette Tape Operating System). It means that CP/M can use the microcassette drive as the equivalent of a disk drive. Because of the physical differences in operation the use of a microcassette is slightly different from using a floppy disk or RAM disk. This will become obvious as you study the operation of the Microcassette drive.

#### b) Microcassette selection and use

Consult your Epson dealer for supplies of microcassette tapes.

You should ensure that the tapes you buy are of the correct type as there are similar size cassettes which are not compatible with the PX-8 Microcassette drive.

Microcassettes are available in sixty-minute lengths (thirty minutes per side) and thirty-minute lengths (fifteen minutes per side). These are marked MC-60 and MC-30 respectively. Because a thirty-minute tape will store 10K to 50K bytes of information per side, it is usually adequate for storage of your most frequently used data and programs. It is recommended that you use the shorter tapes because it is often faster to change tapes (or flip one over) than to wind to the end of a long one. In addition, shorter tapes are less likely to jam. The microcassette directory can only hold 12 file names, so 12 short data files or programs will leave a long length of unusable blank tape.

Some of your most important data will be saved on microcassettes, so it is important to care for them properly. To keep the tape clean, always keep it in its protective case when it is not in use. Never let your fingers touch the tape itself; the oil from your skin can contaminate it. Be sure to keep your tapes away from magnetic surfaces. Proximity to a magnetic field can erase your tapes. Magnetic fields are found in television sets and other devices that have transformers or speakers. Dirty recording heads in your microcassette drive can cause errors in reading and writing and can even damage your tapes. You should clean them periodically, using a head-cleaning cassette, a cleaning kit (available where you purchase your cassettes) or a cotton swab and isopropyl alcohol.

#### c) Tape Insertion

To insert the Microcassette tape proceed as follows:

- 1) Press the EJECT key to open the microcassette compartment cover.

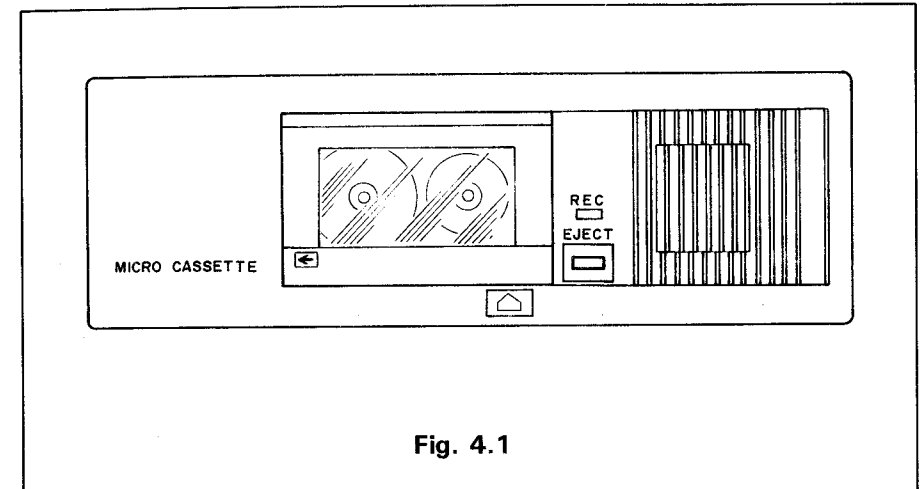


Fig. 4.1

- 2) Insert a microcassette with the side to be used uppermost. This will have the majority of the tape as seen through the window on the right hand side.

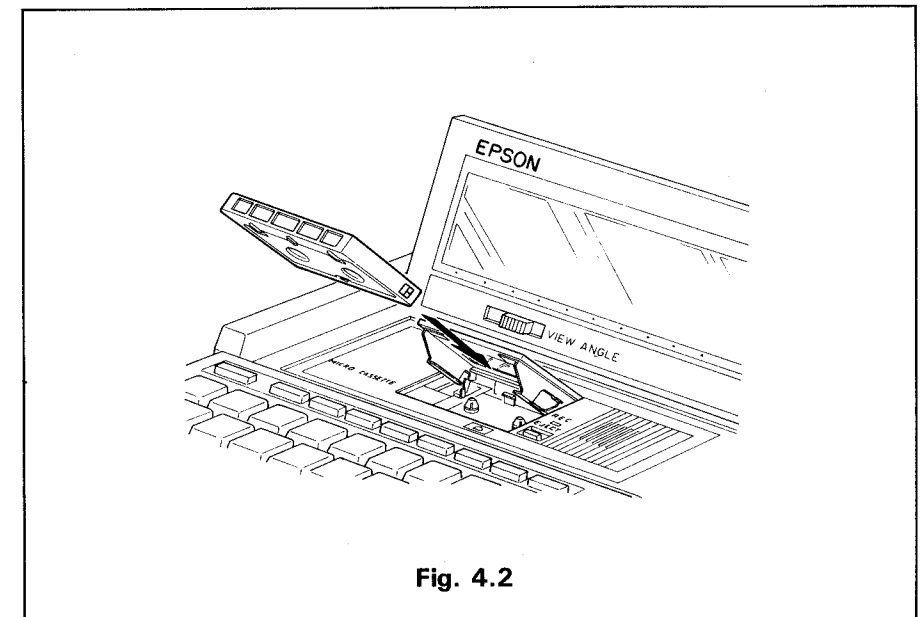


Fig. 4.2

- 3) Close the microcassette compartment cover.

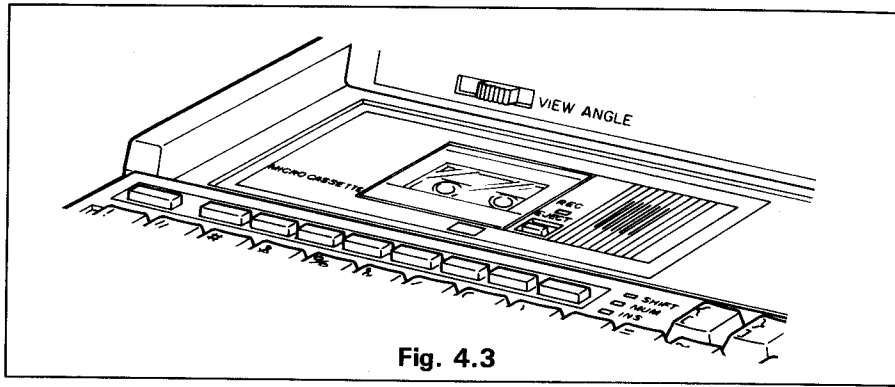


Fig. 4.3

#### d) Manual operation

The microcassette drive can be operated manually by using the System Display.

- 3) Place the PX-8 in the System Display mode by pressing the **HELP** key together with the **CTRL** key.

```

*** SYSTEM DISPLAY ***    00/00/00 (SUN) 19:58:35    <MENU>
<RAM DISK> 009 kb    <AUTO START>
<USER BIOS> 000 256 b    <MCT MODE> stop, nonverify <COUNT> 65535
<MENU DRIVE> CBA    <MENU FILE> 1 .COM 2 .BAS 3 . 4 .
- Select number or ESC to exit.
1=password 2=alarm/wake 3=auto start 4=menu 5=MCT
<<- /    <- /mount    * /dirinit    ->> /erase    000/

```

Fig. 4.4

The bottom line of the System Display can show two states depending on whether a tape has already been set up or 'MOUNTed' in the drive. Section 4.1.1(f) describes this in detail, but for now it is sufficient to say that because a cassette tape accesses data sequentially along the tape and the directory is stored at the beginning, it is not practical to write the new directory to tape each time a program is saved. Consequently it might be possible to change tapes without updating the directory, and the MTOS would not be able to find a file or program. There is a safeguard built in to prevent this. It is not possible to move along a tape if the MTOS is in control of the tape. If a tape is MOUNTed then the bottom line of the system display will show:

remove/


REMOVal and MOUNTing of the tape is described in section 4.1.5f. If the display shows REMOVE and you wish to continue with this explanation, press the SHIFT key and function key PF1, then wait until the display shows the bottom line:

```

<<- /    <- /mount    * /dirinit    ->> /erase    000/

```

The bottom line shows the microcassette drive functions assigned to the programmable function keys.

- PF1** : Fastforward
- PF2** : Play
- PF7** : ( **SHIFT** and **PF2** ) mount
- PF3** : Stop denoted by the  character
- PF8** : ( **SHIFT** and **PF3** ) dirinit
- PF4** : Rewind
- PF9** : ( **SHIFT** and **PF4** ) erase
- PF5** : Counter reset

These are best understood by following an actual operation.

- 1) Insert a blank microcassette. Look at the tape counter setting in the [COUNT] field on the 4th line of the System Display. The counter may be 00000. If not, press the **PF5** key to reset it.
- 2) Press the **PF1** key and the tape will move forward rapidly. Press the **PF3** key and the tape will stop. Look at the tape counter. The value will indicate that the tape has moved.
- 3) Press the **PF4** key and the tape will be rewound. The tape automatically stops when the end of tape is reached. The count should then be 00000.
- 4) Press the **PF2** key and the tape will move forward slowly. If something is recorded on the tape, sound is produced from the speaker. You will not hear this unless the volume is turned up. It is not possible to record sound using the PX-8 but it can play back audio tapes. The tape count increases as the tape moves.
- 5) If the tape contains files, they can be erased by pressing the **PF4** and **SHIFT** keys simultaneously. This is primarily intended for erasing audible sound.

- 6) Rewind the tape and press the EJECT button to remove the microcassette from the drive.

When you have familiarised yourself with the controls you can now use the other controls on the function keys to prepare a tape on which to store programs.

#### e) Directory initialization (DIRINIT)

The tape directory is a tape block which contains various information required for the microcassette tape operating system to manage tape files. This information includes the number of files in the tape, the tape location of each file (the tape count at the beginning and end of the file), the name of each file, etc. The tape directory is always placed at the beginning of a microcassette tape.

Although the operating system manages the tape directory, you must initialize the directory block on to the tape before using a new tape. The procedure to do this is as follows:

- 1) Load a new microcassette.
- 2) Place the PX-8 in the System Display mode by pressing the **HELP** key together with the **CTRL** key.

```

*** SYSTEM DISPLAY ***      00/00/00 (SUN) 19:58:35      <MENU>

<RAM  DISK> 009  kb      <AUTO START>
<USER  BIOS> 000 256 b    <MCT  MODE>  stop, nonverify <COUNT> 65535
<MENU  DRIVE> CBA      <MENU  FILE> 1 .COM  2 .BAS  3 .  4 .
- Select number or ESC to exit.
  1=password 2=alarm/wake 3=auto start 4=menu 5=MCT
  <<- /      <- /mount      # /dirinit  ->> /erase 000/

```

Fig. 4.5

- 3) Press the **PF3** key together with the **SHIFT** key. The screen will clear and the message "dirinit" will appear on the display. The tape will start and the REC lamp light.
- 4) When the directory space has been written to the tape, the REC lamp goes off and the display changes back to the System Display, but now the bottom line is different:

```

*** SYSTEM DISPLAY ***      00/00/00 (SUN) 23:37:47      <MENU>

<RAM  DISK> 009  kb      <AUTO START>
<USER  BIOS> 000 256 b    <MCT  MODE>  stop, nonverify <COUNT> 00189
<MENU  DRIVE> CBA      <MENU  FILE> 1 .COM  2 .BAS  3 .  4 .
- Select number or ESC to exit.
  1=password 2=alarm/wake 3=auto start 4=menu 5=MCT
  /remove      /      /

```

Fig. 4.6

- 5) The tape can be used immediately, that is, it is already MOUNTed ready for use. The function key assignments have been changed and you can only REMOVE the tape. This prevents another tape being MOUNTed. MOUNTing is explained in the next section.

If you do not wish to use the tape, press the **ESC** key to exit to the state in which you were before you pressed the **CTRL** and **HELP** keys.

#### f) Preparing Microcassette tapes for use — MOUNT and REMOVE

Since the tape behaves as if it were a disk, the directory is written to the tape in the same way as on a disk. However, if the tape directory were read each time a tape file was accessed, it would take a relatively long time to rewind the tape to the tape directory and then rewind to the file location. To overcome this, the contents of the tape directory are stored in RAM. The process of loading the directory into RAM ready for saving and loading from the Microcassette tape is known as MOUNTing the tape. This can be achieved in a number of ways:

#### Manually

- i) By pressing the **PF2** and **SHIFT** keys simultaneously in the System Display
- ii) By executing the MOUNT statement in BASIC, in direct mode.

#### Automatically

- i) By using the DIR H: command in CP/M, if the tape has not been mounted.
- ii) By logging into drive H: from the CP/M command line if the tape has not been mounted.
- iii) By saving a file to a tape which has not been mounted.
- iv) By executing a command which loads a file or data from the Microcassette if the tape has not been mounted.
- v) By exiting from a program when the tape has been REMOVED, if the currently logged in drive is drive H:.
- vi) By using the MOUNT command in a BASIC program line.

The contents of the directory stored in RAM (RAM directory) are accessed each time a tape file is read from or written to, that is the directory is updated in RAM. This means that the contents of the tape directory must be replaced with the contents of the RAM directory before removing the tape, otherwise it would not be possible to access the files.

The tape can be REMOVED by one of the following methods:

- i) By pressing the **PF1** and **SHIFT** keys simultaneously in the System Display mode .
- ii) By executing the REMOVE statement in BASIC.

Because of the danger of replacing tapes or mounting them when another is already mounted, the System Display will not allow any operation other than removing a tape, once a tape has been mounted.

**IMPORTANT:**

*It is best to get into the habit of checking whether a tape is mounted or not. ALWAYS press CTRL and HELP to check the status of the tape before REMOVAL or INSERTION of a new tape. This is especially important since the tape could have been MOUNTED without you remembering having done so, for instance by a DIR access.*



**WARNING:**

*If you change a tape without executing the REMOVE statement in BASIC or using the REMOVE on the System Display, you will not only destroy data on the new tape, but you will also destroy data on the old tape.*

When you REMOVE a tape either in BASIC or through the System Display, the following will occur:

- 1) If the tape has only been read, or the directory simply loaded into RAM:
  - i) If the REMOVE was executed from the System Display, the screen will clear for a moment and the word "remove" displayed briefly. The display will then change back to the System Display.
  - ii) If the REMOVE was executed from the BASIC command line, the cursor will simply return after printing "Ok".
  - iii) If the REMOVE was executed from a BASIC program, it will continue from the next line.

- iv) The EJECT button can then be pressed and the tape removed or replaced. The directory will not be written to the tape, in any of these cases.
- 2) If a file has been written to the tape, deleted or renamed, or the file new attribute is set, the directory has to be rewritten.
  - i) The tape will rewind.
  - ii) If REMOVE was executed from the System Display, the screen will clear and then display the word "remove" to show that the tape is being removed. If the REMOVE was executed from BASIC the cursor will disappear (if the command was issued from the command line) or the program will wait for the directory to be rewritten.
  - iii) The REC light will go on and the directory in RAM will be written onto the tape.
  - iv) When the directory has been written back to the tape, the REC light will be extinguished. The System Display will return if that is where the REMOVE was initiated. If the BASIC command 'REMOVE' was used, either the cursor will return if REMOVE was executed from the command line, or it will continue with the next line of the program.
  - v) The EJECT button can then be pressed and the tape removed or replaced.

**g) Checking if the tape is mounted.**

If you need to know if the tape is mounted, press the **CTRL** key together with the **HELP** key to show the System Display. Look at the bottom line. If it says:

/remove

this indicates that the tape directory has been mounted. You can use **ESC** to return to whatever else you were doing, or press the **SHIFT** and **PF1** key to REMOVE the tape.

**h) Using software with the Microcassette drive**

The microcassette drive can be used in the same manner as a disk device. The drive name H: is assigned to the microcassette drive.

*1) Obtaining a directory of the Microcassette tape*

Type DIR H: in the CP/M operating mode (that is, when the system prompt A> is displayed). The tape directory is automatically accessed and the names of all files recorded on the tape are listed on the screen as shown below.

```
A>DIR H:
H: SAMPLE1  BAS : SAMPLE2  BAS
A>
```

Fig. 4.7

ii) *Executing a program from the Microcassette Drive.*

The microcassette drive is used in exactly the same way as any other drive. Since the Microcassette drive is drive H:, to run the program "PROG.COM" from the Microcassette drive type the following on the CP/M command line:

**A>H:PROG**

iii) *BASIC and the Microcassette Drive.*

The BASIC statements and functions concerned with the microcassette drive are as follows.

<b>EOF</b>	<b>OPEN</b>
<b>GET</b>	<b>POS</b>
<b>INPUT #</b>	<b>PRINT #</b>
<b>INPUT\$</b>	<b>PRINT USING #</b>
<b>LINE INPUT #</b>	<b>PUT</b>
<b>LIST</b>	<b>REMOVE</b>
<b>LOAD</b>	<b>SAVE</b>
<b>LOF</b>	<b>TAPCNT</b>
<b>LOC</b>	<b>WIND</b>
<b>LSET/RSET</b>	<b>WRITE</b>
<b>MOUNT</b>	

For details of these statements and functions, refer to the PX-8 BASIC Reference Manual.

Note that the DSKF function does not return meaningful values with the Microcassette drive.

**i) The Stop and Non-Stop Mode and Verification**

Tape storage is different from a conventional floppy disk, in that the data is saved sequentially. On a floppy disk if there is a read error it is possible to go

back to the area on the disk which was mis-read and try again. On a tape this is not possible unless the tape is rewound, which is not a practical proposition. The data is thus stored twice on the tape, so that if an error in reading occurs there is chance for another attempt. Another problem with tape access which does not occur on a floppy disk, is that frequently the tape can go past a block of data because the tape did not stop in time when the previous block was read.

These problems can be overcome by frequent stopping and starting of the tape, both when reading and writing. The System Display allows two different modes of reading and writing the data, the Stop and Non-Stop modes. See section 2.2.3.e. for how to set the Stop and Non-Stop modes. By stopping the Microcassette drive frequently the data can be placed on the tape more accurately when writing and will allow more accurate reading also.

When data is written to the tape, the following sequence occurs. A short length of the tape is erased then a header block is written. This contains information such as the name of the file and its type, and various data dealing with the length of the file, the tape counter etc. Then the data is written in blocks of 256 bytes each block of data being duplicated. Finally the data is followed by an end of file marker. If the Stop mode is set, the microcassette drive stops between each block as it is written. If Non-Stop mode is used when the data is written, the data is written without stopping.

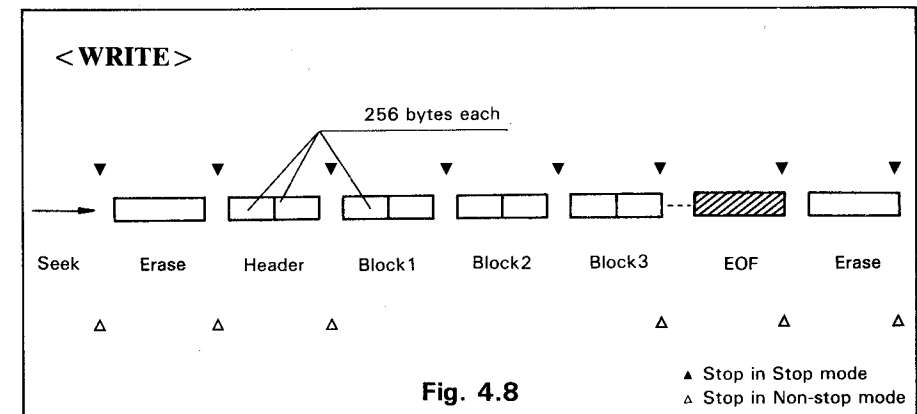
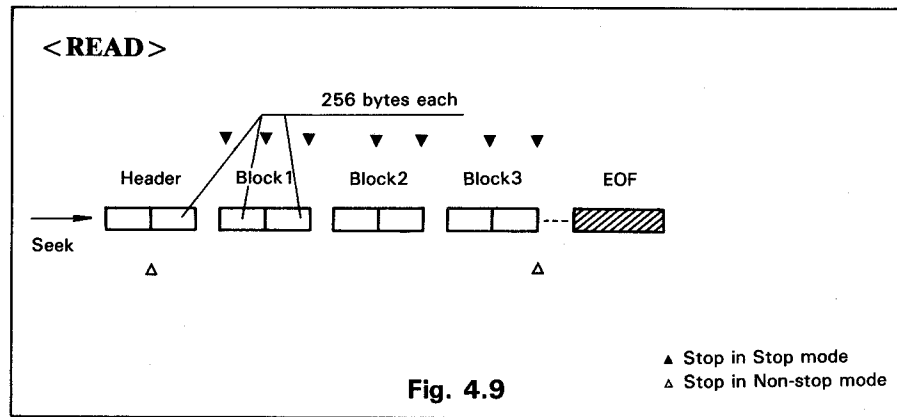


Fig. 4.8

When data is read from the tape if Stop mode is set the tape again stops between each block of data before the next block is read. If Non-Stop mode is set, the data is read consecutively. Unless otherwise set, the data is read in the same mode it was written in.



In reading and writing files using the STOP mode this has the following effects:

- i) More tape is used when the Stop mode is used to write a file.
- ii) The file takes longer to write.
- iii) When reading a tape written in the Stop mode, more accurate reading is possible.
- iv) In extreme cases errors can occur in reading files written in Non-Stop mode. This is most likely to happen when reading single records from a file as this requires the Microcassette drive to be stopped and started by the program. When the tape stops under program control it cannot always stop in the correct position in front of the next record. Thus it can miss the record or part of it and generate an error. This can be overcome by reading the record in Stop mode. As further security against this error occurring read and write such files in Stop mode, by setting the STOP mode from the System Display before running the program.

#### j) The Verify and Non-Verify Mode

In order to ensure accurate recording of data on a Microcassette tape, it is possible to have the data checked by specifying the Verify mode when the data is written. The Verify mode is switched on and off from the System Display, (see section 2.2.3.e).

In the verify mode, each 256-byte block is verified after the entire file has been written to the tape.

If an error is detected, one of the following messages is displayed.

**BDOS ERROR ON H: BAD SECTOR** (under CP/M)  
**Disk Read error** (under BASIC)

In the Non-verify mode, files written to tape are not verified.

It is recommended that the verify mode be used when writing files to cassette tape.

When a file is written, the Microcassette drive will carry out the following procedure:

- i) The tape will be wound to the next available position. Sometimes it may rewind to the beginning of the tape and then wind on so that it can be sure to be at the correct position.
- ii) The REC light will come on and the file will then be written to the tape. The tape may stop and start if the Stop mode is set.
- iii) The REC light will go out.
- iv) If Verify is set, the tape will be rewound to the start of the file and the file will be checked.
- v) The tape will stop and the EJECT button will be unlocked.

Verify does not operate when reading tapes.

#### k) Rules for day to day use of the Microcassette Drive.

- 1) Always check that a tape has been REMOVED before taking it out of the drive, otherwise the directory will not be written back to the tape, and could be written to another tape.
- 2) Never use DIRINIT without checking the directory of a tape if it is not a new one.
- 3) Never switch off while a file is being recorded or read. The file will not be appear on the directory if the power is switched off manually while recording is taking place.

## SUMMARY OF MICROCASSETTE DRIVE OPERATION

System Display mode		Execution in BASIC	Function
Display	Operation		
mount	SHIFT + PF2	MOUNT	Reads the tape directory
remove	SHIFT + PF1	REMOVE	Writes the tape directory
dirinit	SHIFT + PF3		Initializes the tape directory
<<-	PF1	WINDn	Tape Fast forwrd
->>	PF4	WINDn	Rewinds the tape
<-	PF2		Plays the tape through the speaker
erase	SHIFT + PF4		Erases the contents of the tape
⌘	PF3		Stops the tape
000	PF5	TAPCNT = 0	Ressets the tape counter to zero

### 4.1.2 ROM capsules

Many computers are able to plug in programs which are then instantly able to be run. The PX-8 has a slight variation on this concept. The programs are supplied in a ROM which is similar to the program cartridge, but is used by the PX-8 operating system as if it were a disk drive. There are two ROM sockets in the base of the PX-8 which contain the BASIC language and a number of CP/M Utility programs, when you purchase the PX-8. If you have been supplied with other programs in ROM you can change them as follows.

#### a) Changing ROM capsules

- 1) First check with the applications program manual that no special instructions are given for changing ROMs. This is only likely to occur if the program requires more than one ROM for operation. An example of this is given below.
- 2) Carry out whatever operations are necessary to return to the MENU or to the CP/M command line.
- 3) Switch the power off using the power switch.

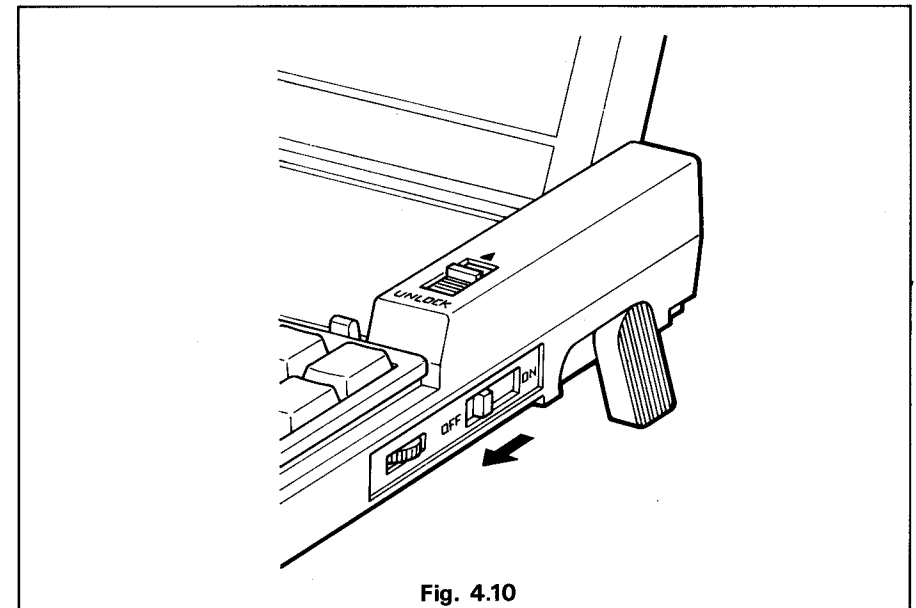


Fig. 4.10

- 4) Remove the ROM compartment cover as shown below and lift the silver coloured flap. The two ROMs lie side by side and are held in a carrier.

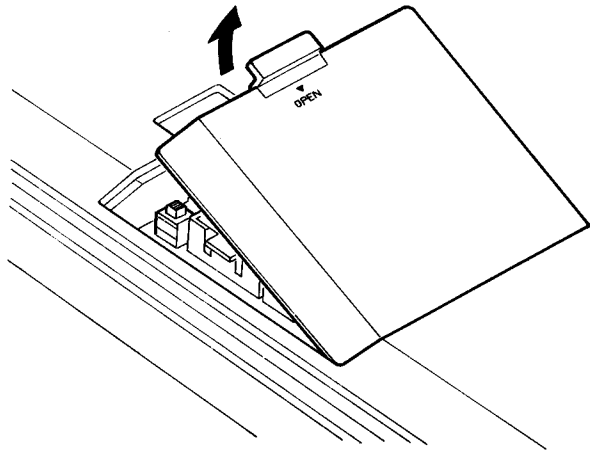


Fig. 4.11a

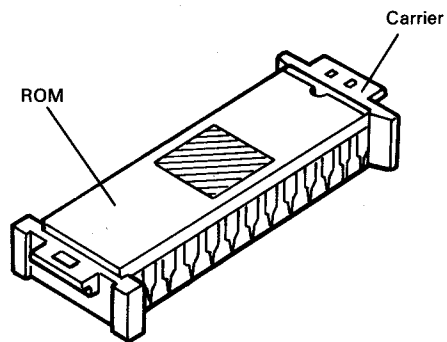


Fig. 4.11b

- 5) Remove one of the the ROM capsules indicated by the arrows by lifting the tabs at the top and base of the ROM. Do not use excessive force

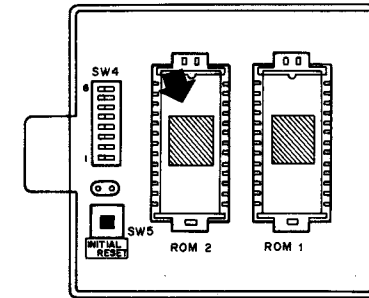


Fig. 4.12a

- 6) Now take the ROM capsule you wish to insert into the PX-8. The plastic carrier has two tabs. Hold it so that the tab with two holes is towards the back of the PX-8. Then place the carrier over the ROM socket so that it sits loosely in the socket.

The carrier and socket are made like a key and a lock. It is only possible to fit the carrier into the socket the correct way.

Now gently push the ROM and plastic carrier so that they fit level with the top of the socket. Use two fingers, one at the top and one at the bottom, to ensure even pressure.

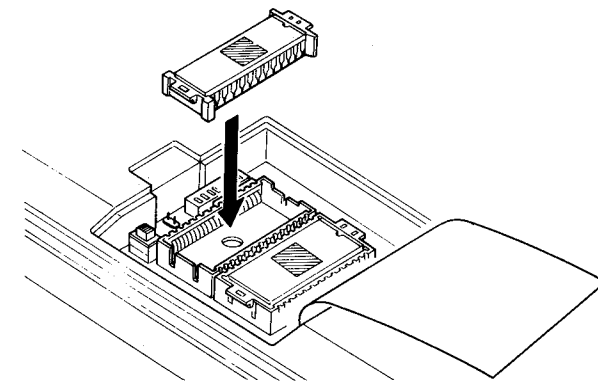
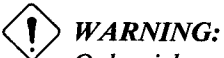


Fig. 4.12b



- 7) Finally return the silver coloured flap to its original position and replace the outer plastic cover.
- 8) Turn the power back on.
- 9) If you are on the MENU page, press the **[ESC]** key to go to the CP/M command line. From the CP/M command line carry out a warm boot by pressing **[CTRL] - [C]** or the **[STOP]** key.
- 10) The programs can be loaded from drive B: if the ROM has been placed in ROM socket 1, and from drive C: if placed into ROM socket 2. The assignment of the sockets to different drives, can be changed using the CONFIG program described in Chapter 3.



**WARNING:**

*Only pick up the ROM by the plastic carrier. Do not touch the metal pins of the ROM with your fingers. This can cause the pins to corrode, and could also destroy the program contained in the ROM.*

**b) Using the ROM as a disk to contain data or BASIC programs.**

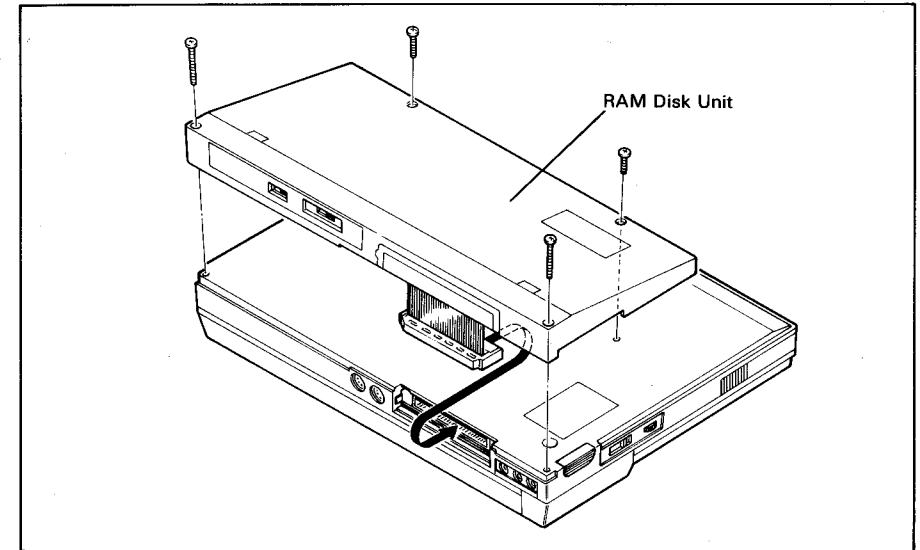
Some application programs may use data which is provided in ROM form. The manual which comes with the program will describe how to use this data. The program will have a command or specifically prompt you to change the ROM. Alternatively the manual may tell you to change the ROM in order to run the BASIC program. If the program is a BASIC program you can use it from ROM as follows:

- 1) When you have loaded BASIC, switch the computer off in the continue mode by switching the power off while pressing the **[CTRL]** key.
- 2) Remove the BASIC ROM, and insert the program ROM as in steps 4 to 7 of the description of changing the ROM.
- 3) Switch the power back on.
- 4) From the BASIC command line type the RESET command followed by the **[RETURN]** key.
- 5) You can now load the program or data from the ROM, treating it as any other drive.

### 4.1.3 RAM Disk Unit — additional RAM disks

Previous chapters have described the use of part of the memory area to store programs. This is referred to as the RAM disk. The size of RAM disk in the main memory is at most 24K bytes. If your applications require more memory space, an optional RAM disk unit of larger capacity can be connected to the system bus connector. This is known as an Intelligent RAM disk

Two types of optional RAM disk units are available: one is of 60K bytes (Model H102A) and the other is of 120K bytes (Model H103A). (These units actually contain 64K and 128K of RAM. However, some RAM is used by the firmware so is not all available to the user.)



When an Intelligent RAM disk unit is connected, the internal RAM disk cannot be used. When fitting an additional RAM Disk Unit the PX-8 must be initialized, and the disk formatted in the same way as the RAM disk which uses part of the internal memory. When initializing either pressing a 7508 (sub-CPU) reset switch or using the **[SHIFT] - [NUM GRAPH]** keys while pressing the reset button, the PX-8 will know that an external disk is connected. There will be no request for the size of the RAM disk, but it will still be necessary to format the disk.

If you try to write to the RAM disk when the write protection switch is on, a BDOS error message will be displayed on the screen.

CP/M manages the organisation of saving and loading programs to the RAM disk, just like any other disk, with the following additional features:

- i) Input and output speed is much higher than other disk devices even ordinary floppy disk drives.
- ii) The PX-8 is very little larger in size and weight than without the disk added.
- iii) The Intelligent RAM disk can be write protected with a switch and so permanently protected.

Apart from having more available memory, the Intelligent RAM Disk Unit is used in exactly the same way as the normal RAM disk. It cannot be changed in size, nor can any of the PX-8 RAM be used if the Intelligent RAM disk is attached. Thus when initializing with an Intelligent RAM Disk connected, no message will come up asking for the size of the RAM disk. You will be asked whether it should be formatted however.

When an Intelligent RAM Disk Unit is fitted, a check of the contents is made each time on power up, to ensure the data is intact. This only occurs if DIP switch 4-5 is set to ON. If the data is corrupted, the screen will clear and a message will appear to ask if you wish the RAM Disk to be formatted. The check takes eight seconds. If DIP switch 4-5 is set to the OFF position this check will not be performed. Since it is unlikely that the data will be corrupted, it is often more convenient to set the switch to OFF. It is necessary to perform a reset in order to activate this DIP switch. Take care not to alter the country settings.

When setting the size of the RAM disk using the CONFIG program, if the Intelligent RAM Disk is connected, the size will be displayed but there will be no option to change the size.

The Intelligent RAM Disk unit has its own battery to preserve the contents of the memory when the PX-8 is not in use. This backup switch should be set to ON when the unit is installed. However, if the unit is disconnected from the PX-8 for long periods of time, the switch should be turned off so that the battery does not completely run down.

The battery does not allow the Intelligent RAM Disk unit to be disconnected and then used on a different PX-8 while still keeping the contents of memory intact.

#### 4.1.4 Floppy disk drives

Additional Floppy Disk Drives are provided for use with the PX-8.

As the drives are a separate item, they have a their own manual to show how to use them. There are also separate programs for copying and formatting disks plus other software. Use of these programs is covered in the manual supplied with the disks.

Please consult your EPSON dealer for further details of the Floppy Disk Drives.

## 4.2 RS-232C interface

The PX-8 is equipped with an RS-232C interface and the operating system supports a communication speed of up to 19200 bps. The RS-232C interface is used to communicate with other computers, to modems acoustic couplers and printers and other devices which support this communication standard.

The operation mode of the RS-232C interface can be specified with the CONFIG program in Chapter 3 or by using the OPEN command in BASIC.

The SHIFT-IN/SHIFT-OUT and XON/XOFF functions are also supported by the operating system, but can only be set by BASIC.

The SHIFT-IN/SHIFT-OUT protocol allows the full ASCII code from 0 to 255 decimal, but only using seven bits. It only operates during seven bit transmission. When a code greater than 127 is required, the SHIFT-OUT (SO) code is sent to tell the receiving device to set the high bit, i.e. to add 128 to the code received. Thus if the code 65 is sent when the SO has been sent the receiving computer will interpret it as code 193 (65 + 128). The SHIFT-IN code reverses an SO condition. If a code 193 is sent at when an SI code has been sent the code will be interpreted as code 65. The SI/SO codes act as a switch. For example all characters will have the high bit set until the SI code is sent once an SO code has been sent.

When communication occurs with two devices using XON/XOFF protocol, the transmitting device can be told by the receiving device to wait until the receiving device is ready to handle further information. This is important if the receiving device is processing the information and needs time to carry out some operations before receiving more data.

The codes for SI/SO, XON/XOFF are:—

	Decimal	Hexadecimal
SO	14	0E
SI	15	0F
XON	17	11
XOFF	19	13

A warm boot will set the default setting of SI/SO and XON/XOFF to off, and thus to ensure they are not active, perform a warm boot before using the CONFIG program to set the parameters.

### NOTE:

The CONFIG program does not allow the XON/XOFF or SI/SO protocol to be set. This can only be done from BASIC, or a machine code BIOS call.



### WARNING:

When sending data which could contain these characters for example a machine code program, the SHIFT-IN/SHIFT-OUT and XON/XOFF options should be set to off. If this is not done, when the receiving device is sent any of the above characters it will be interpreted as such and the appropriate action will be taken. Consequently the data received will have the bit setting changed in the case of SI/SO. Also the particular character being sent as control data (SI/SO/XON/XOFF) will not form part of the data received as it will have been interpreted as a control character.

Before using TERM or FILINK execute a warm boot to set the RS-232 parameters to the default, and then change them using the CONFIG program if necessary. This ensures the SI/SO and XON/XOFF parameters are set to off.

The RS-232C interface can also be used in BASIC programs. For details of use, refer to the PX-8 BASIC Reference Manual.

The PX-8 uses the RS-232C interface to connect a printer unless the system configuration is changed with the CONFIG command.

### Examples of use

The following command transmits the contents of file SAMPLE.TXT to the RS-232 interface.

```
PIP PUN:=A:SAMPLE.TXT
```

The following command receives data from the RS-232C interface and saves it on the disk in drive A: under the file name SAMPLE2.DAT.

```
PIP A:SAMPLE2.DAT=RDR:
```

The default settings of the RS-232C interface operation mode are as follows.

Communication speed:	4800 bps	Parity check:	No
Character length:	8 bits	SI/SO control:	No
Number of stop bits:	2	XON/XOFF control:	No

**NOTE:**  
A warm boot will always return the RS-232C settings to these defaults.

The RS-232C interface connector is provided on the rear panel.

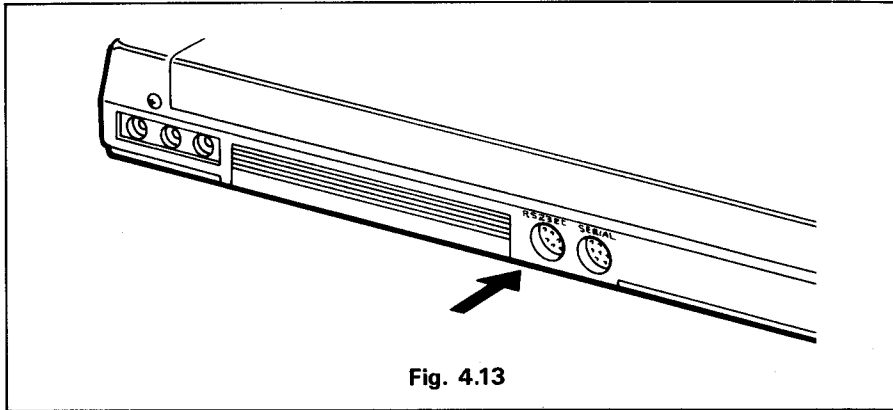


Fig. 4.13

The pin assignments and signal descriptions are as follows.

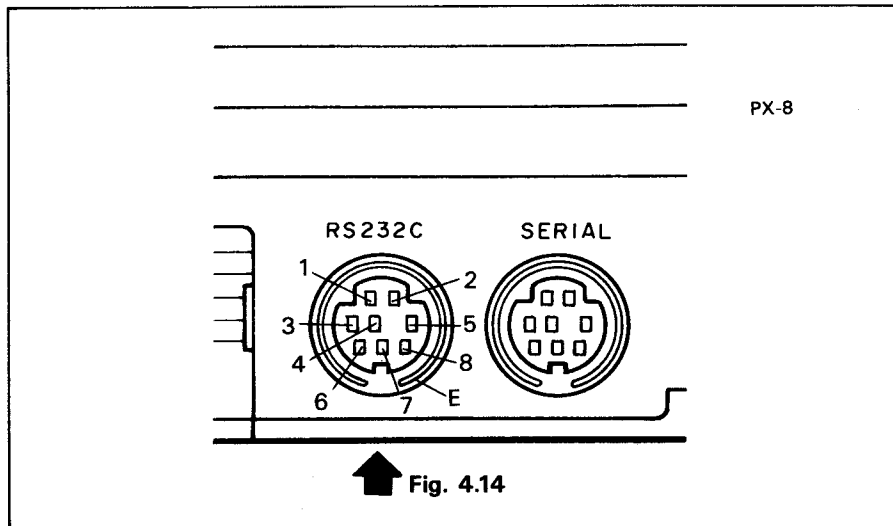


Fig. 4.14

The standard RS-232C interface is connected to external devices through a miniature connector. Pin assignments of this connector are as follows.

Pin No.	Signal Symbol	Signal Direction	Description of Signal
1	GND	—	Ground
2	TxD	OUT	Transmitted data
3	RxD	IN	Receive data
4	RTS	OUT	Request to send
5	CTS	IN	Clear to send
6	DSR	IN	Data set ready
7	DTR	OUT	Data terminal ready
8	DCD	IN	Data carrier detect
E	FG	—	Frame ground

Handshake!  
Handshake!

**NOTE:**  
The direction of signal is as viewed from the PX-8.

The meanings of the various signals are as follows.

**GND (Ground)**

This terminal is used as the return line for the following signals.

**TxD (Send data)**

TxD is the signal used from transmitting data from the PX-8 to the device (acoustic coupler, etc.) with which the PX-8 is connected. This is possible when the Clear to send signal is on.

**RxD (Receive data)**

RxD is the data signal from the acoustic coupler or other RS-232C compatible device to the PX-8.

**RTS (Request to send)**

RTS is the signal which controls the communication function of the device (acoustic coupler, etc.) connected to the PX-8. The connected device becomes ready to send when this signal is ON.

**CTS (Clear to send)**

CTS is the signal which indicates whether the connected device is ready to ac-

cept data transmissions. Transmission is enabled when this signal is ON and disabled when it is OFF.

**DSR (Data set ready)**

DSR is the signal which indicates whether the connected device is ready for operation. When this signal is ON, the applicable device is connected to the interface cable and is ready to accept data transmission/reception control signals.

**DTR (Data terminal ready)**

DTR is the signal output by the PX-8 to the connected device to indicate that it is ready to receive data.

**DCD (Data carrier detect)**

The DCD terminal is used for detecting the carrier signal from the connected device.

**FG (Frame ground)**

This terminal is connected to the chassis of the PX-8; ordinarily, it is also connected via the external cable to the corresponding terminal on the other device.

**RS-232C cables**

Please consult your Epson dealer for the correct type of cable to connect to the other equipment. The cables available from EPSON are as follows:—

1) Cable # 724

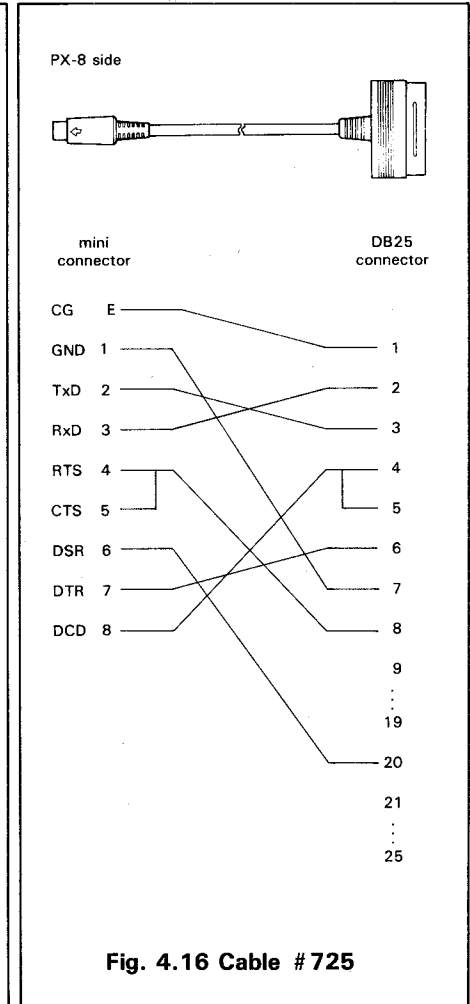
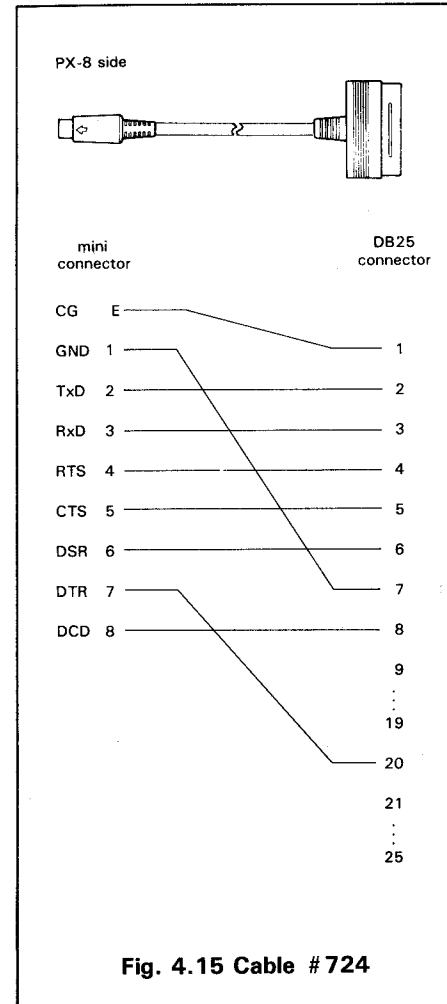
This cable is used to connect the PX-8 to a modem or acoustic coupler.

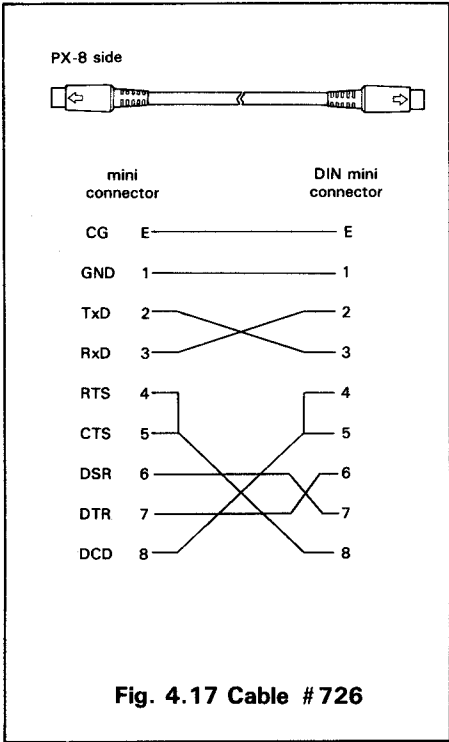
2) Cable # 725

This cable is used to connect the PX-8 to a computer which is equipped with an RS-232C interface and a DB25 connector. It is also used to connect the PX-8 with a serial printer or other I/O device which is equipped with an RS-232C interface.

3) Cable # 726

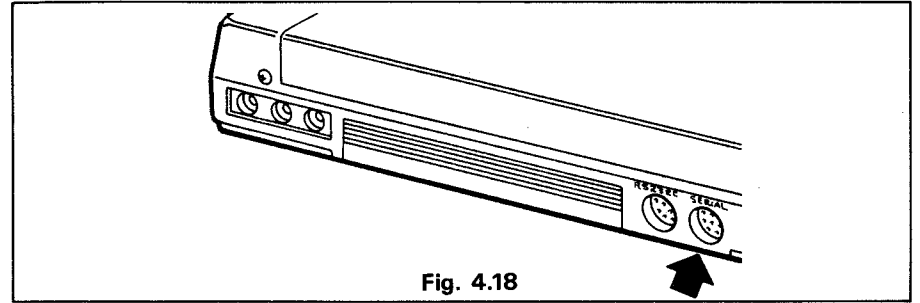
This cable is used to connect two PX-8s through the RS-232C interfaces. This is also referred to as the null modem cable.





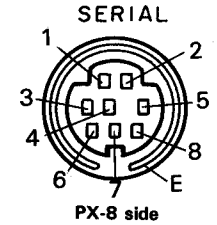
### 4.3 Serial Interface

The serial interface is used to connect optional disk drives (drive D:, E:, F: and G: in the default assignments) or a serial printer. The location of the interface connector is shown below.



The serial interface is connected to external devices through a miniature connector. Pin assignments of this connector are as follows.

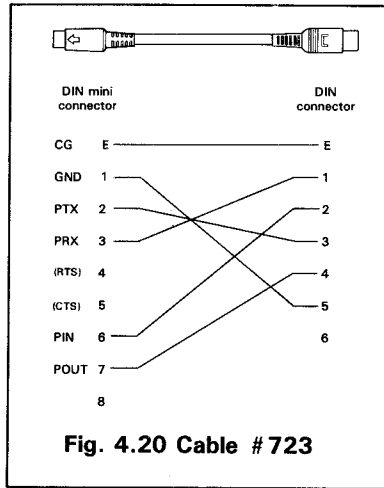
Pin No.	Signal Symbol	Signal Direction	Description of Signal
1	GND	—	Ground
2	PTX	OUT	Transmitted data
3	PRX	IN	Receive data
4	(RTS)	OUT	Request to send
5	(CTS)	IN	Clear to send
6	PIN	IN	Status ready
7	POUT	OUT	Control signal
8			
E	FG	—	Frame ground



**NOTE:**  
The direction of signal is as viewed from the PX-8.

The communication speed is 38400 bps when disk drives are connected and 4800, 600 or 150 bps when a printer is connected. When the serial interface is chosen as the printer interface, the default setting is 4800 bps. Which device is connected is determined by the CONFIG program.

Cable # 723 (Fig. 4.20) is used to connect disk drives and cable # 725 (Fig. 4.16) is used to connect a printer.



The PX-8 can also detect that an EPSON Floppy Disk Drive is attached and will automatically set the baud rate to 38,400 baud without the need to use CONFIG.

When a printer is selected, logical device LST: is assigned to the serial interface. Therefore,

**PIP LST: = A:SAMPLE.TXT**

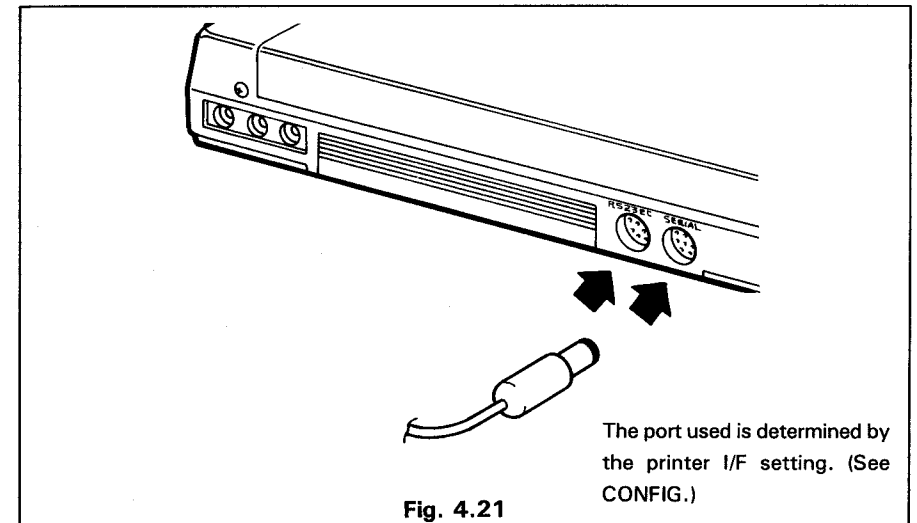
outputs the contents of file SAMPLE.TXT to the printer connected to the serial interface.

## 4.4 Using Printers with the PX-8

Many applications of the PX-8 will require the use of a printer, e.g. documents entered using Portable WordStar™. It is also convenient to have a hardcopy (i.e. a copy printed on paper) of BASIC program listings, disk directories and many other day to day transactions.

### 4.4.1 Connecting a printer

To connect an EPSON printer to the PX-8 an additional serial interface is required. The printer must be connected to the interface connector (RS-232C or serial) which has been selected by the CONFIG command. The default setting is to the RS-232 interface. The data transfer rate should be set to 4800 bps on both the printer and the PX-8. This is the default setting on the PX-8, but may have been altered for other communication. The current setting can be found using the CONFIG program.



If the PX-8 and the printer have different transmission rate settings the result will be that the output to the printer will be unintelligible. For example:

! " # \$ % & ' ( ) \* + , - . / : ; < = > ? @ [ \ ] ^ \_ ` { | } ~

**Fig. 4.22**

The MX, RX and FX series printer have a DB25 connector on their serial interface and require cable # 725. If you have problems in connecting your EPSON printers or any other printer please consult your EPSON dealer.

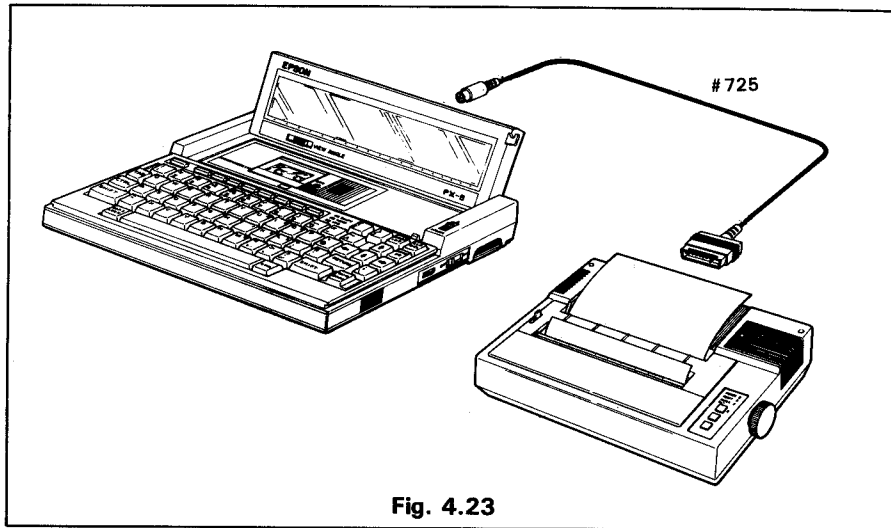


Fig. 4.23

For details of printer specifications and control codes, refer to the operating manual provided with the printer.

#### 4.4.2 Using a printer with the PX-8

The printer is controlled automatically when you use Portable WordStar™ or other programs outputting data to the printer.

A printer connected to the interface selected by the CONFIG command is assigned to logical device LST:. Therefore, any file data can be output to the printer using the PIP command as shown below.

**PIP LST: = filename.ext**

Any data output to the screen by CP/M, can be echoed to the printer if the **CTRL - P** key is pressed. This acts as a switch, turning the output on the first time it is pressed, and off the second time. Thus DIRectories of disks and output from utilities such as STAT can be obtained by pressing **CTRL - P** before giving the appropriate command.

The contents of the real screen can be output to the printer by pressing the **PF5** key together with the **CTRL** key (screen dump). If the screen mode is 0, 1, or 2 then the code will be sent to the printer in ASCII format. If the screen is in mode 3 (the graphics screen), all data including alphanumeric characters will be output in bit image mode (i.e. as dots). All EPSON printers can print such data, some other printers may not be able to. When data is sent as ASCII codes, graphics characters can only be printed if the printer contains those characters in its character set. Some EPSON printers have the characters included. For upgrading of your printer, please consult your dealer. If you do not have the graphics characters on your printer, change the screen mode to mode 3 and use the **CTRL - PF5** key combination to give a graphics dump of the screen.

**NOTE:**

*The ASCII code for the graphics characters may produce unwanted characters to be printed where a printer does not support them. It is possible to have the graphics characters printed as a space by setting DIP switch 4-6. This will only occur when using the screen dump program (pressing CTRL and PF5).*

If the switch is set to OFF the following characters will be printed as a space. This means the graphics characters will be printed.

Codes decimal:	00 to 31, 127 and 255
hexadecimal:	00 to 1F, 7F and FF

If the switch is set to ON the graphic characters will be printed as spaces and so the following characters will be printed as a space:

codes decimal:	00 to 31, and from 127 to 255
hexadecimal:	00 to 1F, and from 7F to FF

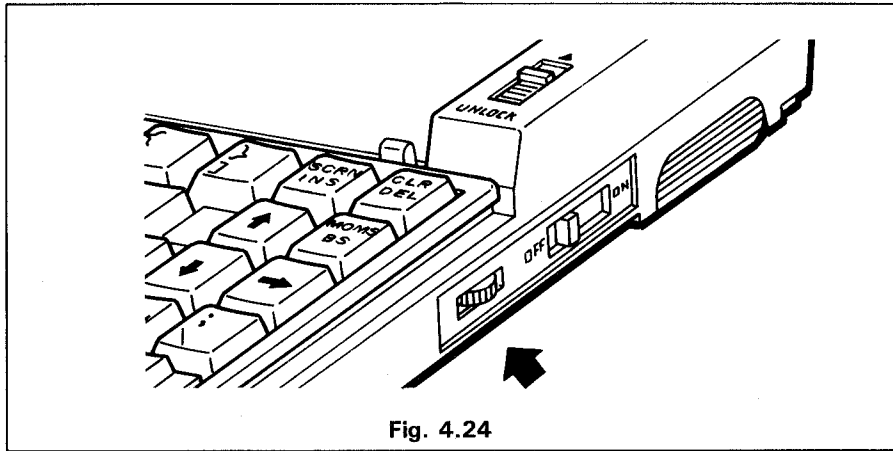
In changing the DIP switch, the reset button must be pressed to make the change effective.

When the BASIC interpreter is used, all printer-related statements and control codes can be used for the printer connected to the interface selected by the CONFIG command. For details of operation in the BASIC mode refer to the PX-8 BASIC Reference Manual.



## 4.5 Speaker

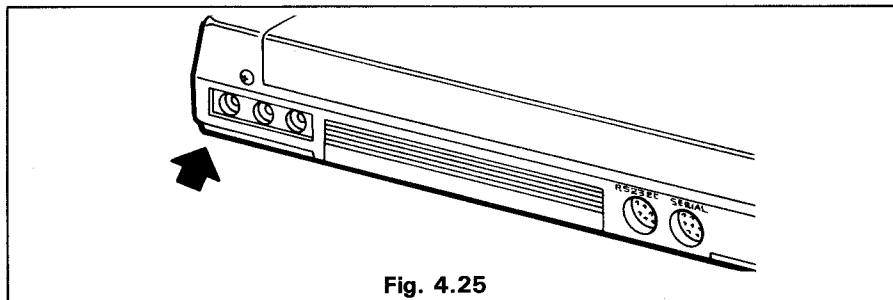
The PX-8 is equipped with a built-in dynamic speaker. The sound volume control is provided on the right hand side panel.



This speaker can be used to generate music composed by software. Sounds can also be output from the speaker in the following ways:

- i) By the BEEP and SOUND commands of BASIC.
- ii) By calling the BIOS BEEP routine (see Chapter 5).
- iii) From the Microcassette Tape.
- iv) From the System Bus (see Section 4.9).

An external speaker jack is provided on the rear panel. It is marked SP OUT.



- External speaker (3-pin jack)

No.	Name	Note	Level	I/O
1	EXSPG	GND		
2	EXSP	External speaker output	0~6V	O
3	EXSP	External speaker output	0~6V	O

Power : 200 to 300 mmW

Impedance :  $8\Omega$

When an external speaker is connected to this jack, the built-in speaker is disconnected.

**NOTE:**

*Avoid using an external speaker at high volume for an extended period of time.*

## 4.6 Analog interface

The PX-8 has a built-in analog interface, which allows analog signals (voltages) to be converted into numerical data for handling by software in the PX-8. It is often referred to as an Analog to Digital converter (abbreviated to A/D). The interface is located on the rear panel marked A/D IN.

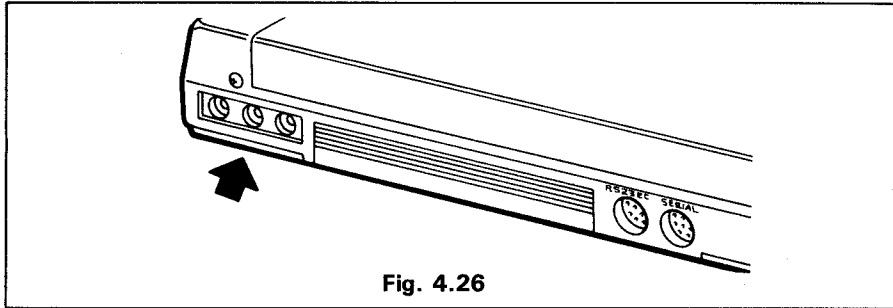


Fig. 4.26

- External speaker (3-pin jack)

No.	Name	Note	Level	I/O
1	GND			
2	ANIN	Analog input	0~2V	I
3	TRIG	Trigger signal	TTL	I

### NOTE:

The analog interface is not supported by BASIC but can be used with a short machine code routine and a BIOS call. An example of such a use is given in Appendix H.

The specifications of the A/D converter are as follows:

Input level: 0 to 2.0 V  
 Resolution: 6 bits (32 mv)  
 Conversion time: 140  $\mu$ s  
 Maximum input level: 4.5 V

## 4.7 Bar Code Reader Interface

The PX-8 is equipped with a Bar Code reader interface.

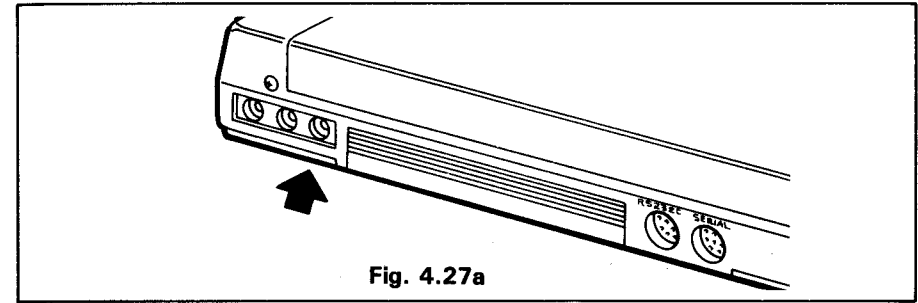


Fig. 4.27a

Bar Code data is a specially printed bar-code which can be read with a special Wand, which is plugged into the BRC socket on the back panel of the PX-8. Such codes are common on food packages and many other products throughout the world. Fig. 4.27b shows an example of a Bar Code, and the Wand used to read it.

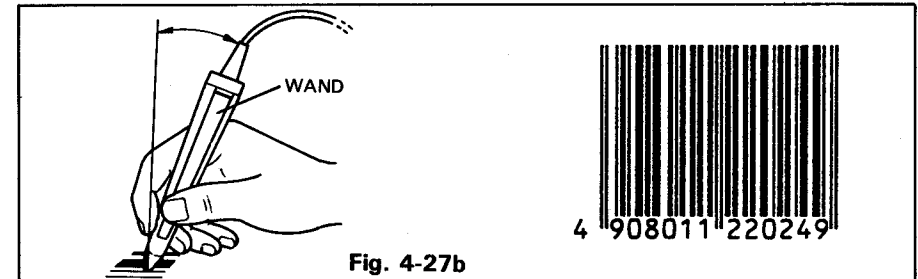


Fig. 4-27b

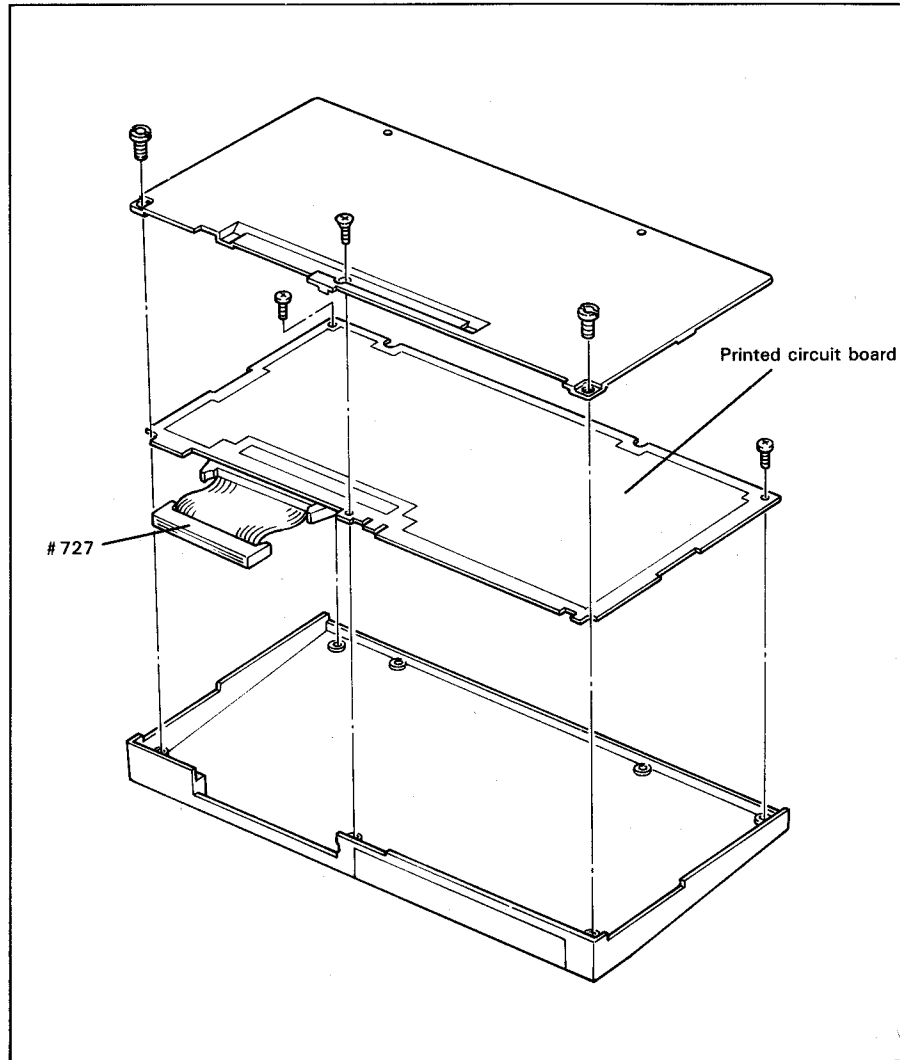
- Bar code reader (3-pin jack)

No.	Name	Note	Level	I/O
1	GND	Ground		
2	BRDT	Bar code read data	TTL	I
3	+5	Turned on and off by program	4.5~6V	

The use of a Bar Code reader requires special applications software. It is not supported by BASIC or the PX-8 operating system. Please consult your EPSON dealer for further information on its availability.

## 4.8 The Universal Unit

Many users and hardware manufacturers may wish to make prototype versions of optional products for the PX-8. Also construction details of add units are often published in books and magazines. The Universal Unit is a printed circuit board in a suitable case which makes it easy for such items to be constructed.



Universal Unit

4-38

## 4.9 System Bus Interface

The system bus interface connector is provided on the rear panel. This connector is used to connect an external RAM disk unit, the Universal Unit (section) or other special dedicated hardware.

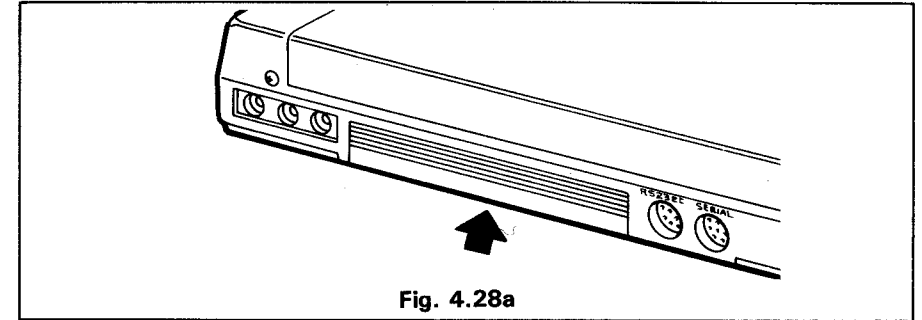


Fig. 4.28a

The system bus cover can be removed with a screwdriver as shown below.

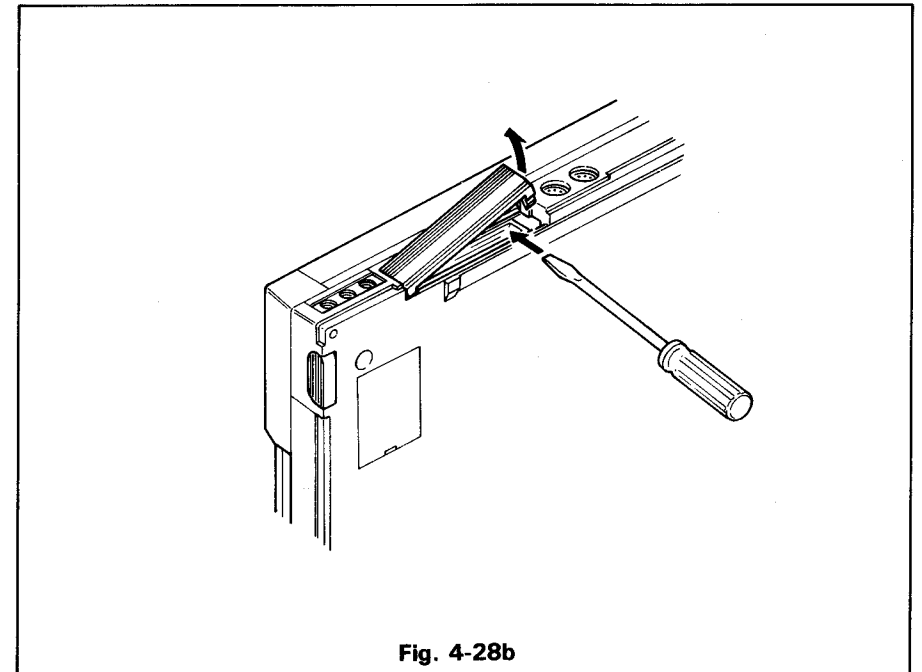
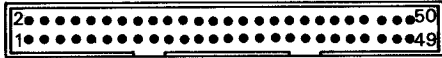


Fig. 4-28b

4-39

The pin assignments of the system bus connector are as follows:

50 pin flat cable connector



(Viewed from rear)

Signal name	Pin No.	I/O	Remarks	Signal level
AB15	7	O	Z-80 address bus (no buffer)	TTL
14	3			
13	4			
12	1			
11	2			
10	13			
9	16			
8	15			
7	14			
6	11			
5	12	I/O	Z-80 address bus (no buffer)	TTL
4	9			
3	10			
2	6			
1	5			
0	8			
BB 7	24	I/O	Z-80 Data bus (no buffer)	TTL
6	23			
5	22			
4	21			
3	20			
2	19			
1	18			
0	17			

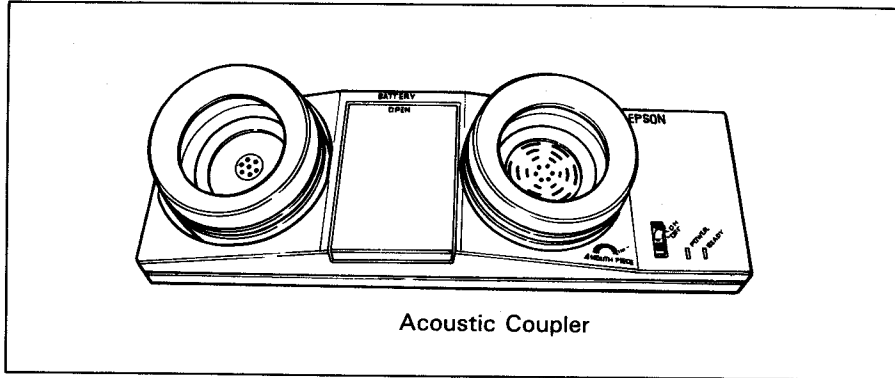
Signal name	Pin No.	I/O	Remarks	Signal level
$\overline{\text{WAIT}}$	28	I	Z-80 wait signal	TTL
$\overline{\text{BURQ}}$	25	I	Z-80 bus request	TTL
$\overline{\text{BUAK}}$	26	O	Z-80 bus acknowledge	TTL
$\overline{\text{MRQ}}$	36	O	Z-80 memory request	TTL
$\overline{\text{RD}}$	35	O	Z-80 data read	TTL
$\overline{\text{WR}}$	37	O	Z-80 data write	TTL
$\overline{\text{IORQ}}$	40	O	Z-80 I/O request	TTL
$\overline{\text{MI}}$	27	O	Z-80 M1 (machine cycle 1) signal	TTL
$\overline{\text{CLK}}$	38	O	Z-80 main clock, 2.45 MHz (1 buffer stage)	TTL
$\overline{\text{RS}}$	33	O	System reset signal (backup)	TTL
$\overline{\text{INTEX}}$	43	I	External interrupt signal	TTL
SPI	34	I	External speaker input (D-MDM)	0 to 5V
$\overline{\text{BK2}}$	48	I	Bank 2 switching signal BK2=0:BANK2	TTL
DW	42	O	RF control signal for expansion DRAM	TTL
DCAS	41	O	RF control signal for expansion DRAM	TTL
OFF	44	O	Logic power supply (+5V) ON signal; logic power turned on when PON=0	TTL
TXD	46	O	82C51 TXD output signal	TTL
$\overline{\text{RXD}}$	45	I	RXD signal to 82C51 from external option	TTL
VBI	47	P/S	Battery power supply; not affected by SW on board	4 to 6V
$\overline{\text{HLTA}}$	30	O	Battery power supply; turned on/off by SW on board	TTL
VCH	39	P/S	Power supply for battery charging (AC adapter output)	6 to 8V
VL	29	P/S	Logic power supply	4.5 to 6V
GND	31,32		Signal ground	
CG	49,50		Chassis ground	

**NOTE:**

The direction of signal is viewed from the PX-8

## 4.10 Acoustic Coupler

With an acoustic coupler, you can connect the PX-8 to a computer (such as another PX-8, Epson QX-10 or a large computer) in another place using the telephone line. Data is transferred through the RS-232C interface.



The TERM and FILINK utility programs described in Chapter 3, enable communications to be handled with a minimum of effort. More information is given under these programs.

Such communication can also be achieved in BASIC.

Cable # 724 is required to attach to an acoustic coupler with a DB25 connector.

The acoustic coupler models available differ from country to country. See your Epson dealer for detailed information.

## Chapter 5

# THE SYSTEM INTERFACE

The operating system of the PX-8 is an expanded version of the CP/M operating system, version 2.2. This chapter deals with the more advanced aspects of using the PX-8 and will be of more use to programmers than any other users. It contains information on subroutines which can handle peripheral devices and files. Familiarity with assembler programming is a prerequisite for using these subroutines and functions. You must purchase an assembler such as MACRO-80 and other program development tools from your dealer for using these BDOS and BIOS subroutines.

Users who are not interested in assembler programming may skip this chapter.

Further information on the operating system will be published as a separate manual.

### 5.1 The CP/M Configuration

The configuration of the expanded CP/M is as follows:

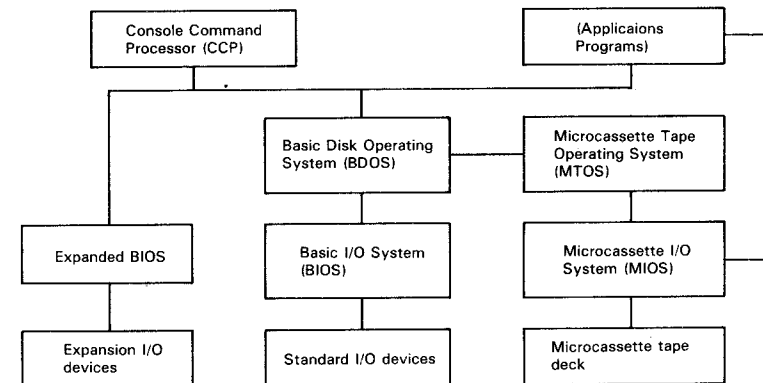
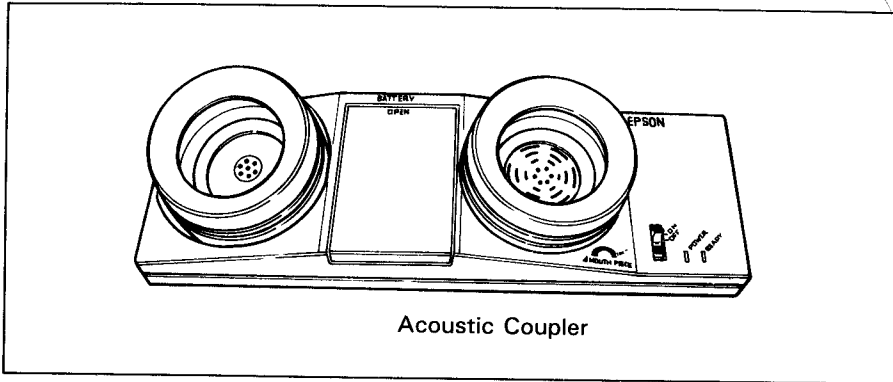


Fig. 5-1

## 4.10 Acoustic Coupler

With an acoustic coupler, you can connect the PX-8 to a computer (such as another PX-8, Epson QX-10 or a large computer) in another place using the telephone line. Data is transferred through the RS-232C interface.



The TERM and FILINK utility programs described in Chapter 3, enable communications to be handled with a minimum of effort. More information is given under these programs.

Such communication can also be achieved in BASIC.

Cable # 724 is required to attach to an acoustic coupler with a DB25 connector.

The acoustic coupler models available differ from country to country. See your Epson dealer for detailed information.

## Chapter 5

# THE SYSTEM INTERFACE

The operating system of the PX-8 is an expanded version of the CP/M operating system, version 2.2. This chapter deals with the more advanced aspects of using the PX-8 and will be of more use to programmers than any other users. It contains information on subroutines which can handle peripheral devices and files. Familiarity with assembler programming is a prerequisite for using these subroutines and functions. You must purchase an assembler such as MACRO-80 and other program development tools from your dealer for using these BDOS and BIOS subroutines.

Users who are not interested in assembler programming may skip this chapter.

Further information on the operating system will be published as a separate manual.

## 5.1 The CP/M Configuration

The configuration of the expanded CP/M is as follows:

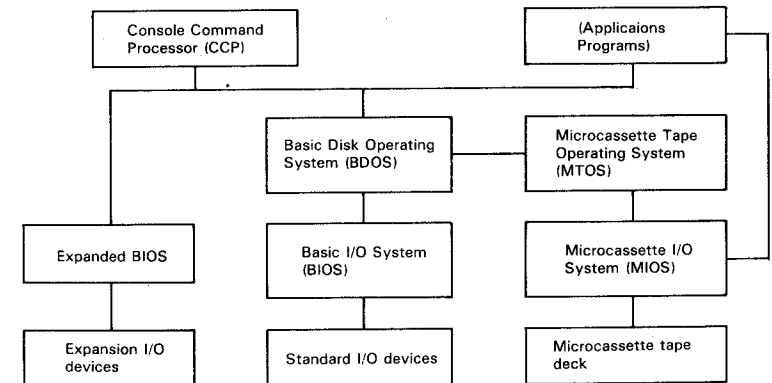


Fig. 5-1

## 5.2 The IOBYTE

The PX-8 operating system supports 4 logical I/O devices, CON:, RDR:, PUN: and LST: as described in Chapter 3. In order to make it easy for the user, the CP/M Utilities STAT enable devices to be assigned using these names.

In software terms the logical and physical devices are assigned using the address 0003H in the main memory bank which contains a byte whose contents decide the logical to physical device assignments. This byte is known as the IO-BYTE. The contents of the IOBYTE assigns the logical to physical devices as shown in the following table.

Table 5.1

Logical device	LST:	PUN:	RDR:	CON:	
Bit position	7 or 6	5 or 4	3 or 2	1 or 0	
Direction Bit pair value (Binary)	Output	Output	Input	Output	Input
00	Serial (printer)	—	Keyboard	RS-232C	Keyboard
01	LCD	LCD	—	*LCD	*Keyboard
10	*RS-232C	*RS-232C	*RS-232C	LCD	RS-232C
11	—	—	—	RS-232C	RS-232C

The default setting of the IOBYTE is 10101001. (Hexadecimal A9) This gives the default assignments as shown by the asterisks in the table.

Further information on the assignments can be found in Chapter 3, sections 3.6 and 3.8.

## 5.3 The File Control Block (FCB)

During file operations, the operating system obtains file information from a table called the file control block (FCB). The information included in the FCB is as shown below.

Dr	F1	F2	/	/	F8	T1	T2	T3	Ex	S1	S2	Rc	D0	/	/	D15	Cr	R0	R1	R2	
00	01	02	...	08	09	10	11	12	13	14	15	16	...	31	32	33	34	35			

Dr: Decimal Drive code

0: use logged-in drive

1: use Drive A:

2: use Drive B:

3: use Drive C:

4: use Drive D:

5: use Drive E:

6: use Drive F:

7: use Drive G:

8: use Drive H: the microcassette drive

F1 — F8: contain the file name in upper case ASCII

T1 — T3: contain the file type in upper case ASCII

Ex: Current extent number

S1: Reserved for system

S2: Reserved for system

Rc: Record count

D0 — D15: Assigned by operating system.

Cr: Record counter

R0 — R2: Random access record counter

Fig. 5.2

The CCP provides a FCB in the system work area which is called a TFCB (temporary FCB); this is used as the default FCB.

## 5.4 BDOS Function Calls

BDOS includes many utility subroutines which input or output data to and from peripherals and handle files. These subroutines are referred to as BDOS functions, and can easily be used in your own programs as shown below. There are 39 BDOS functions which can be utilised; any of these can be used by calling address 0005H with a function number in register C and a parameter in register pair DE. Single byte values will be returned in register A while double byte values are returned in register pair HL.

The BDOS functions are listed below. In this list, the entry parameters are those which are passed to BDOS from the user program which calls the function, and the return parameters are those which are passed to the user program from the function called.

Table 5.2

Function No.	Explanation	Entry Parameter	Returned value
0:	This function returns control to the CP/M command level.	C: 00H	
1	This function reads a character from CON:	C: 01H	A: Read character
2	This function outputs a character to CON:	C: 02H E: Character to be output	
3	This function reads a character from RDR:	C: 03H	A: Read character
4	This function outputs a character to PUN:	C: 04H E: Character to be output	
5	This function outputs a character to LST:	C: 05H E: Character to be output	
6	This function provides direct console input and output operation. It bypasses all CP/M normal control character functions such as "CTRL" + "P."	C: 06H E: For input, 0FFH For output, the character to be output.	A: For input, the character read 00H is returned when CON: is not ready.
7	This function returns the current contents of IOBYTE.	C: 07H	A: Contents of IOBYTE
8	This function sets a new value in IOBYTE.	C: 08H E: Value to be set in IOBYTE	
9	This function outputs the character string starting at the specified address and ending with "\$" to CON:	C: 09H DE: Starting address of the memory area in which the character string is stored	
10	This function reads a character string from CON: into the buffer area starting at the specified address.	C: 0AH DE: Starting address of the buffer area	Buffer: Character string read from CON:
11	This function reads the status of CON:	C: 0BH	A: CON: status FFH — CON: ready 00H — CON: not ready
12	This function returns the version number of the CP/M system currently in use.	C: 0CH	H: CP/M or MP/M L: Version number
13	This function sets all disks to read/write, selects drive A and sets the default DMA address to 0080H.	C: 0DH	
14	This function selects the specified disk drive.	C: 0EH E: Name of drive to be selected	

(Continued)



Function No.	Explanation	Entry Parameter	Returned value
15	This function opens a file.	C: 0FH DE: FCB address Positions 1 through 14 must contain the name of the file to be opened.	A: Directory code OFFH when the file cannot be found. This function fills the FCB in the main memory with the contents of the corresponding FCB in the directory stored on the disk.
16	This function closes a file.	C: 10H DE: FCB address	A: Directory code OFFH when the file cannot be found. This function writes the contents of the FCB in the main memory to the directory on the disk.
17	This function searches for a file.	C: 11H DE: FCB address	A: Directory code OFFH when the file cannot be found.
18	This function is used following function 17 to find the file whose name matches that specified.	C: 12H	A: Directory code OFFH when the file cannot be found.
19	This function deletes the specified file.	C: 13H DE: FCB address	A: Directory code OFFH when the file cannot be found.
20	This function reads the next record from the file into memory at the current DMA address.	C: 14H DE: FCB address	A: 00H when the read operation is completed. Other than 00H when the next record contains no data.
21	This function writes 128 bytes of data at the current DMA address to a record of the file specified by the FCB.	C: 15H DE: FCB address	A: 00H when the write operation is completed. Other than 00H when the disk is full.
22	This function generates a new file and catalogs it in the directory.	C: 16H DE: FCB address	A: Directory code OFFH when the directory is full.
23	This function changes the file name.	C: 17H DE: FCB	A: Directory code OFFH when the file cannot be found
24	This function returns the log-in vector which indicates drives which are currently on line.	C: 18H	HL: Log-in vector The least significant bit of L corresponds to drive A and the most significant bit of H corresponds to drive P. A "1" bit indicates that the corresponding drive is on line. A: Contains the same value as register L.
25	This function returns the currently logged-in drive.	C: 19H	A: Currently logged-in drive 01H - Drive A 02H - Drive B 05H - Drive E 06H - Drive F

(Continued)

Function No.	Explanation	Entry Parameter	Return Parameter
26	This function changes the DMA address.	C: IAH DE: DMA address	
27	This function returns the base address of the allocation vector.	C: IBH	HL: ALLOC address
28	This function sets the read only attribute for the currently logged-in drive.	C: 1CH	
29	This function returns the R/O vector which indicates drives which are set to read only.	C: IDH	HL: R/O vector
30	This function sets the file attributes. The R/O and system attributes can be set or reset with this function.	C: IEH DE: FCB address	A: Directory code A: FFH (no file)
31	This function returns the BIOS resident disk parameter block (DPB).	C: 1FH	HL: DPB address
32	This function sets or gets the user number.	C: 20H E: OFFH for get E: User code for set	A: User number (GET)
33	This function is similar to function 20; however, a particular record is read according to the contents of positions R0 through R2 in the FCB.	C: 21H DE: FCB address	A: Return code 00H - Normal completion Non-zero - Abnormal completion
34	This function is initiated similarly to function 33. However, the data at the DMA address is written to the disk.	C: 22H DE: FCB address	A: Return code 00H - Normal completion Non-zero - Abnormal completion
35	This function returns the virtual file size to the random record bytes (R0 to R2) of the FCB.	C: 23H DE: FCB address	
36	This function returns the random record position to the random record bytes of the FCB after a series of sequential reads or writes.	C: 24H DE: FCB address	
37	This function resets the specified drives according to the 16-bit drive vector indicating drives to be reset; the least significant bit of the vector corresponds to drive A, and so forth.	C: 25H DE: Drive vector	A: 00H
40	This function is similar to function 34. However, the data written is all 00H.	C: 28H DE: FCB address	A: Return code 00H - Normal completion

## 5.5 Zero Page Locations

CP/M holds a number of parameters in page zero of the main memory bank. The IOBYTE (section 5.2) is held in location 3H. The following are important locations in page zero. Further information is included in the books referred to in Chapter 3, and other such books.

Location in hex		Contents
from	to	
0000	0002	A jump instruction followed by the warm start entry address. Thus locations 0001 and 0002 contain WBOOT.
0003		The IOBYTE. See section 5.2 for further details.
0004		The current default drive number (0 = A:, 1 = B: etc.)
0005	0007	A jump instruction followed by the location of BDOS. Thus locations 0006 and 0007 contain the lowest address used by CP/M.

## 5.6 The BIOS Interface

The Basic Input/Output System (the BIOS) of the operating system for the PX-8 includes many useful subroutines which can be used by calling their entry addresses from user programs after setting parameters (if required) in applicable registers. Care must be taken to ensure that the entry addresses are correctly specified when calling these subroutines.

## 5.7 The Entry Address of the BIOS Subroutines

Table 5.3 lists the entry addresses of BIOS subroutines.

Since the BIOS routines are relocatable, the entry addresses of BIOS subroutines are indicated relative to the warm boot routine (WBOOT) address. The WBOOT address can be found from locations 0001 and 0002 in page zero.

Table 5.3

ADDRESS	ENTRY NAME	ADDRESS	ENTRY NAME
WBOOT - 03H	BOOT	+ 45H	RSIN
WBOOT	WBOOT	+ 48H	RSOUT
WBOOT + 03H	CONST	+ 4BH	TIMDAT
+ 06H	CONIN	+ 4EH	MEMORY
		+ 51H	RSIOX
+ 09H	CONOUT	+ 54H	LIGHTPEN
+ 0CH	LIST	+ 57H	MASKI
+ 0FH	PUNCH	+ 5AH	LOADX
+ 12H	READER	+ 5DH	STORX
+ 15H	HOME	+ 60H	LDIRX
+ 18H	SELDSK	+ 63H	JUMPX
+ 1BH	SETTRK	+ 66H	CALLX
+ 1EH	SETSEC	+ 69H	GETPFK
+ 21H	SETDMA	+ 6CH	PUTPFK
+ 24H	READ	+ 6FH	ADCVRT
+ 27H	WRITE	+ 72H	SLAVE
+ 2AH	LISTST	+ 75H	RDVRAM
+ 2DH	SECTAN	+ 78H	MCMTX
+ 30H	PSET	+ 7BH	POWEROFF
+ 33H	SCRNDUMP	+ 7EH	USERBIOS
+ 36H	BEEP		
+ 39H	RSOPEN		
+ 3CH	RSCLOSE		
+ 3FH	RSINST		
+ 42H	RSOUTST		

### 5.7.1 Functions of BIOS Subroutines

In the following explanations, entry parameters are those which must be assigned by user programs calling the BIOS routines. The contents of registers other than those to which the system returns parameters may be changed unless otherwise specified.

### 5.7.2 Subroutines Concerned With Power-Up and Initialization

#### BOOT

Entry Point: WBOOT - 03H

BOOT is the entry point for the cold start loader, which runs only when system initialization is made or the 7508 sub-CPU is reset or **SHIFT** + **NUM GRAPH** keys are pressed together with RESET switch. This routine is not used by the user.

#### WBOOT

Entry Point: See locations 0001H and 0002H in page zero.

The WBOOT location is used as the point from which all other BIOS routines are given in the following sections. WBOOT can alter if the configuration of the system changes, e.g. the RAM Disk size alters or the USER BIOS size changes.

WBOOT is the entry point for the warm start bootstrap; this routine loads CCP and BDOS into memory. The MENU or CCP module is activated after a warm start.

Return Parameters:

The currently selected drive is returned in register C.

#### POWEROFF

Entry Point: WBOOT + 7BH

POWEROFF is the entry point for the subroutine which turns off the main system power supply, after saving the current status.

Entry parameters

Register C = 00H Sets continue mode when switching on.

Register C = 01H Sets restart mode upon power-up.

### 5.7.3 BIOS Subroutines for use with the console

#### CONST

Entry Point: WBOOT + 03H

CONST is the entry point for the subroutine which reads the console status and sets it in register A.

Return Parameter

Register A = 00H Indicates that the console input buffer is empty.

Register A = FFH Indicates that the console input buffer contains characters.

## CONIN

Entry Point: WBOOT + 06H

CONIN is the entry point for the subroutine which enables a character to be input from the keyboard. It loops indefinitely until a character is entered when the buffer is empty. Because it is also possible to press the Programmable Function keys, special provision is made for them as follows.

Return Parameters

Register A contains the ASCII code of the input character if a key other than a PF key is pressed.

The FUNKFLG (address F108H) setting determines the contents of register A when a PF key is pressed.

If FUNKFLG is set to FFH, registers A and C contain the following data:

	Register C	Register A
PF key pressed	FFH	Code as in table of right
OTHER KEY pressed	00H	ASCII code

Code returned by the PF keys			
PF1	E0H	PF6	E5H
PF2	E1H	PF7	E6H
PF3	E2H	PF8	E7H
PF4	E3H	PF9	E8H
PF5	E4H	PF10	E9H

If FUNCFLG is not set to FFH (normally it will be 0), then register A will contain the ASCII code of the key pressed if a key other than a PF key is pressed, and will contain the first character of the PF key string if a PF key is pressed. Continuous polling of the keyboard will cause the other characters to be entered from the PF key string as if they were typed from the keyboard.

## CONOUT

Entry Point: WBOOT + 09H

CONOUT is the entry point for the subroutine which outputs a character to the console from register C. Note that a number of functions can be obtained by using the control codes in Appendix E and also the ESC control sequences in Appendix A.

Entry parameter

Register C = ASCII code of character to be output to the console

## 5.7.4 BIOS subroutines for use with devices

### LIST

Entry point: WBOOT + 0CH

LIST is the entry point for the subroutine which outputs a character to the logical device LST:. It will be output if both DSR and TxRDY are "1", otherwise will wait until these conditions are satisfied.

Entry parameter

Register C = ASCII code of character to be output to LST:

### PUNCH

Entry Point: WBOOT + 0FH

PUNCH is the entry point for the subroutine which outputs a character to the logical device PUN:.

Entry parameter

Register C = ASCII code of character to be output

### READER

Entry Point: WBOOT + 12H

READER is the entry point for the subroutine which reads a character from the logical device RDR:. This subroutine loops until a character is input.

Return Parameter

Register A = ASCII code of character input from RDR:

### LISTST

Entry Point: WBOOT + 2AH

LISTST is the entry point for the subroutine which reads the status of the logical device LST:.

#### Return Parameter

Register A = 00H Indicates that LST: is busy.  
Register A = FFH Indicates that LST: is ready.

This subroutine returns FFH when DSR = 1; otherwise, it returns 00H.

### GETPFK

GETPFK is the entry point for the subroutine which accesses the character strings assigned to the PF keys.

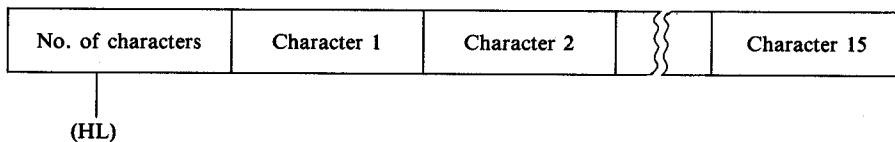
Entry Point: WBOOT + 69H

#### Entry parameters

Register C = PF key number -1 (i.e. 0 to 9 for keys 1 to 10)  
Register HL = Starting address of the character string buffer

#### Result

The format of the character string buffer is as follows.



Whether or not any of the PF keys has been pressed can be determined using the CONST and CONIN routines. The maximum capacity of the character string buffer is 15 characters. The contents of register pair HL are not affected by execution of this routine.

### PUTPFK

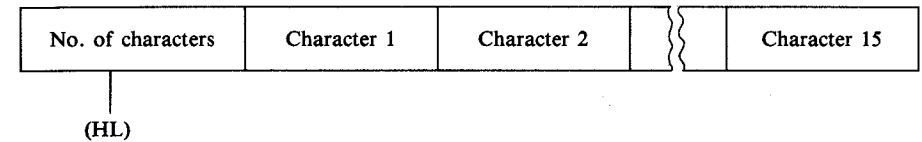
PUTPFK is the entry point for the subroutine which assigns a character string to the PF keys.

Entry Point: WBOOT + 6CH

#### Entry parameters

Register C = PF key number -1 (i.e. 0 to 9 for keys 1 to 10)  
Register HL = Starting address of the character string buffer

The format of the character string buffer is as follows.



A maximum of 15 characters can be assigned to one PF key. The contents of register pair HL are not affected by execution of this routine.

### PSET

Entry Point: WBOOT + 30H

PSET is the entry point for the subroutine which replaces graphic screen data at a specified address in accordance with a specified logical operation. The graphics screen is sequential in memory, with each dot corresponding to one bit. The graphic screen memory starts at 8380H and has a length of F00H bytes. Each byte thus corresponds to eight dots across the screen. Since there are 480 dots, the second row of the screen starts at byte 83BC. Logical operation is required if individual bytes need to be set.

#### Entry parameters

Register B = Data  
Register C = 1 (AND)  
          C = 2 (OR)  
          C = 3 (XOR)  
          C = Others (NOP)  
Register HL = graphic screen address

#### Return Parameters

Register A = 00H — Normal completion  
Register A = FFH — Character screen mode

Register A = other than 00H and FFH — Indicates that the address in HL is not within the VRAM area.  
The contents of register B and register pair HL are not changed.

### SCRNDUMP

Entry Point: WBOOT + 33H

SCRNDUMP is the entry point for the subroutine which prints a hard copy of the LCD screen image on the printer. It is terminated by pressing the CTRL-STOP key.

#### Return Parameters

F67EH (LSTERR) = 00H — normal operation  
F67EH (LSTERR) = FFH — Terminated by CTRL-STOP

## 5.7.5 BIOS Subroutines related to the Disk operation

### HOME

Entry Point: WBOOT + 15H

HOME is the entry point for the subroutine which sets the track to zero.

### SELDSK

Entry Point: WBOOT + 18H

SELDSK is the entry point for the subroutine which specifies a disk drive.

#### Entry parameter

Register C = 00H Drive A:  
Register C = 01H Drive B:  
Register C = 02H Drive C:  
Register C = 03H Drive D:  
Register C = 04H Drive E:  
Register C = 05H Drive F:  
Register C = 06H Drive G:  
Register C = 07H Drive H:

HL register includes the disk parameter header address.

#### Return Parameters

Register HL = 00H Indicates that a parameter error has occurred.  
Register HL is not 00H Normal operation

### SETTRK

Entry Point: WBOOT + 1BH

SETTRK is the entry point for the subroutine which selects the track to be read from or written to.

#### Entry parameter

Register BC = Track number

The track number varies according to the drive as shown below. It can be any number from zero to that shown in the following table:

Drive	Default Track number	Default Device
A:	0 to 1 (Max. 2) 0 to 6 0 to 13	Internal RAM disk 60K RAM Disk Unit 120K RAM Disk Unit
B: C:	0 to 3	ROM 1 (32K ROM) ROM 2 (32K ROM)
D: E: F: G:	0 to 39	FDD 1 FDD 2 FDD 2 FDD 4
H:	0 to 4	Microcassette drive

If a value which is not within the above range is specified, an error occurs when a read or write is performed. These values assume that the default settings for the drive assignments hold. See section 3.8.6 for further details of drive assignments.

#### NOTE:

Maximum track number changes in accordance with the sign of RAM disk set by CONFIG.

### SETSEC

Entry Point: WBOOT + 1EH

SETSEC is the entry point for the subroutine which sets the sector number to be read from or written to.

Entry parameter

Register BC = Sector number to be accessed (00H — 3FH)

If a value other than 00H to 3FH is specified, an error will occur when a read or write is performed.

#### SETDMA

Entry Point: WBOOT + 21H

SETDMA is the entry point for the subroutine which sets the starting address of the 128 byte DMA buffer area used for disk access.

Entry parameter

Register BC = DMA address

#### READ

Entry Point: WBOOT + 24H

READ is the entry point for the subroutine which reads data from a disk drive into the DMA buffer according to parameters set by the SELDSK, SETTRK, SETSEC and SETDMA subroutines.

Return Parameter

Register A = 0 Normal completion

Register A = Other than 0 Abnormal completion

#### WRITE

Entry Point: WBOOT + 27H

WRITE is the entry point for the subroutine which writes data to a disk drive according to parameters set by the SELDSK, SETTRK, SETSEC and SETDMA subroutines.

Entry parameter

Register C = 00H Write standard format data.

Register C = 01H Write unblocked data.

Register C = 02H Write sequential file.

Return Parameter

Register A = 00H Normal completion

Register A = Other than 00H Abnormal completion

#### SECTRAN

Entry Point: WBOOT + 2DH

SECTRAN is the entry point for the subroutine which converts a logical sector number into the corresponding physical sector number.

Entry parameters

Register BC = Logical sector number

Register HL = Physical sector number

#### DISKTBL AND DISKROV — CHANGING DRIVES

The CONFIG program enables the logical and physical drives to be assigned in three ways. It is not possible to have the Microcassette Drive assigned to any drive other than drive H:. The logical and physical drives are assigned in a table of 7 bytes in length beginning at address F1D2H (DISKTBL). The table at default contains the following data:

Address	Logical Drive	Code	Physical Drive
F1D2H	A:	00	RAM Disk
F1D3H	B:	01	ROM Capsule 1
F1D4H	C:	02	ROM Capsule 2
F1D5H	D:	03	Floppy Disk Drive 1
F1D6H	E:	04	Floppy Disk Drive 2
F1D7H	F:	05	Floppy Disk Drive 3
F1D8H	G:	06	Floppy Disk Drive 4

If the code 07H to FFH is contained in this table, the drive cannot be selected. If the table is changed, the two bytes at address F1DAH (DISKROV) must also be changed. This is the vector table which sets the READ/WRITE status of the drives. If the corresponding bit is "1" the drive is set to R/O, and if to R/W it is set to "0". F1DAH (DISKROV) includes the drive in the following order.

bit 7	6	5	4	3	2	1	0
H:	G:	F:	E:	D:	C:	B:	A:

**NOTE:**

Please not that device assignments changed by this address may not be reflected in CONFIG main menu 5. Do not set the same value as the code.

**5.7.6 Bios Subroutine for the Speaker**

**BEEP**

Entry Point: WBOOT + 36H

BEEP is the entry point for the subroutine which sounds the speaker.

Entry parameters

Register C = 0 turns the speaker off.  
C = 1 to FF The speaker sounds for an interval of [C] × 0.1 secs

Register DE sets the frequency of the sound according to the expression:

$$(DE) = \frac{10^6}{3.2 \times \text{Frequency (Hz)}}$$

**5.7.7 BIOS Subroutines for the RS232 Port**

The following routines can be used to operate the RS-232C interface. Details of the RS-232C interface are given in Chapter 4.

The interface supports all commonly used transmission rates, up to 19,200 bps. In general it is not possible to transmit and receive at different rates. Since some databases (e.g. the European Videotext databases) use 75 bps transmit and 1200 bps receive this option is available together with the reverse 1200 bps transmit/75 bps receive.

The interface is initialized using the RSIOX routine, but can also be initialized by BASIC and partially by the CONFIG program. The operating system uses a 261 byte buffer. Characters overflowing this buffer are lost. The size of the buffer can be increased by the RSIOX routine.

The interface should be opened before attempting to send data.

The interface also supports XON/XOFF and SI/SO in communication. These are described in Chapter 4.

The SI/SO function only applies in 7 bit transmission.

When  $\overline{SI/SO} = 1$ , no action is taken.

When  $\overline{SI/SO} = 0$ , if a SI character (0FH) is received, subsequent characters will have a "0" added as the MSB to complete the byte. When  $\overline{SI/SO} = 0$ , if a SO character (0EH) is received, subsequent characters in the range 20H to 7EH will have a "1" added as the MSB to complete the byte. Characters 00H to 1FH and character 7FH will remain unchanged.

The  $\overline{XON/XOFF}$  protocol allows the receiving device to tell the sender to wait while it catches up on processing.

When  $\overline{XON/XOFF} = 1$  no action is taken.

When  $\overline{XON/XOFF} = 0$  a transmission off character CTRL-S (13H) is sent when the buffer becomes more than three-quarters full, so that processing can catch up. A transmission on character CTRL-Q (011H) is sent when the buffer becomes less than one-quarter full, so that transmission can begin again. This makes it possible to prevent the buffer overflowing and the loss of characters.

**RSOPEN**

Entry Point: WBOOT + 39H

RSOPEN is the entry point for the subroutine which initializes the RS232C interface for communication according to the conditions set with the CONFIG program. This subroutine turns on the power supply to the RS-232C interface (and the serial interface) and performs processing to prevent noise from being output over the line. When this routine is called, the former RS-232C conditions are reset; therefore, any data remaining in the receive buffer at the time of the call is lost.

**RSCLOSE**

Entry Point: WBOOT + 3CH

RSCLOSE is the entry point for the subroutine which disables communication through the RS232C interface.

**RSINST**

Entry Point: WBOOT + 3FH



RSINST is the entry point for the subroutine which checks whether any characters have been received.

**Return Parameters**

Register A = 00H No data received

Register A = FFH Data received

**RSOUTST**

Entry Point: WBOOT + 42H

RSOUTST is the entry point for the subroutine which checks whether transmission is enabled.

**Return Parameters**

Register A = 00H Transmission disabled

Register A = FFH Transmission enabled

**RSIN**

Entry Point: WBOOT + 45H

RSIN is the entry point for the subroutine which receives one character. When the receive buffer is empty, it waits until data is received.

**Return Parameter**

Register A = Character received

Z Flag = 01H Normal completion

Z Flag = 00H CTRL-STOP was used to terminate transmission.

**RSOUT**

Entry Point: WBOOT + 48H

RSOUT is the entry point for the subroutine which outputs a character via the RS232C interface.

**Entry parameter**

Register C = ASCII code of character to be sent.

Z Flag = 01H Normal completion

Z Flag = 00H CTRL - STOP was used to terminate transmission.

Buffer address and size are as specified by the user with the RSOPEN routine.

**NOTE:**

RSOPEN must be called to open the RS232C port before RSIN or RSOUT is called.

**RSIOX**

Entry Point: WBOOT + 51H

RSIOX is the entry point for the subroutine which determines the I/O configuration via the RS-232C interface and opens or closes the interface. The functions of this subroutine are explained below.

**RSIOX (OPEN)**

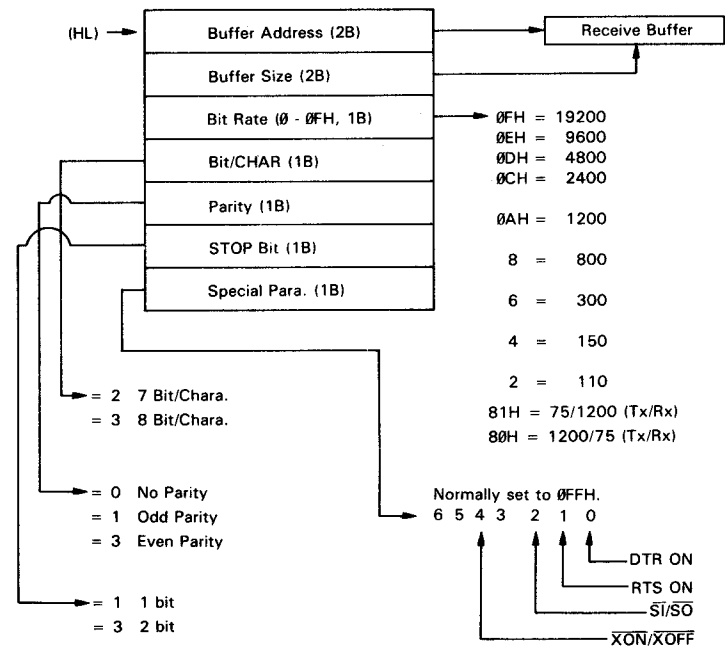
Function: Opens the RS232C interface

**Entry parameters**

Register B = 10H

Register HL= Parameter block address

The contents of the parameter block are shown below.



**Return parameters**

Register A = 00H Normal open

A = 02H Busy (i.e., used by another program)

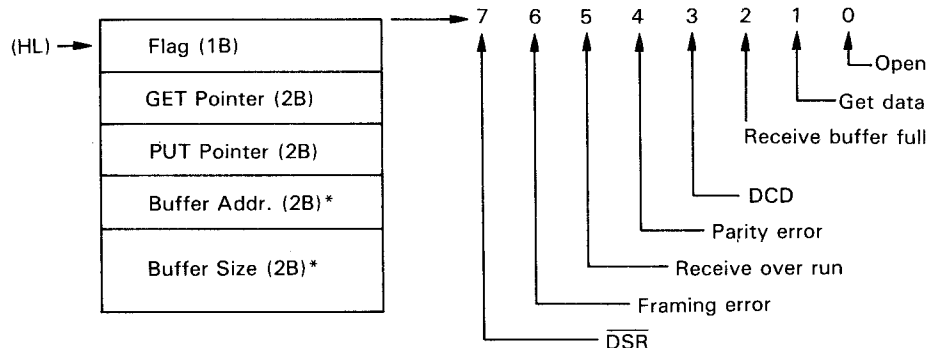
A = 04H Parameter error

Register HL contents are not changed

Z flag = 1

Z flag = 0

Z flag = 0



**RSIOX (CLOSE)**

Function: Closes the RS232C interface.

Entry parameter

Register B = 20H

**RSIOX (INSTS)**

Function: Checks whether there is any data in the receive buffer.

Entry parameters

Register B = 30H

Register HL = 9 byte block address which is used to store return information.

Return parameters

Z flag = 01H Normal completion

Register A = 00H No data in the receive buffer

A = FFH Data has been received

Register BC = LOC — Number of bytes of received data

Register HL = Return information (See Open)

**NOTE:**

The meaning of LOC relates to the value of PUT(Pointer) and GET(Pointer) (see diagram under RSIOX (OPEN)).

If  $PUTP \geq GETP$  then  $LOC = PUTP - GETP$

If  $PUTP < GETP$  then  $LOC = PUTP - GETP + \text{Buffer size}$

Z flag = 0 Abnormal end

Register A = 03H — Interface not open

Register BC Unknown

Register HL Unknown

**RSIOX (OUTST)**

Function: Checks whether output is enabled.

Entry parameters

Register B = 40H

Register HL = Address of the block which is used to store return information

Return parameters

Z flag = 1 Normal completion

Register A = 00H Output disabled

Register A = FFH Output enabled

Z flag = 0 Abnormal end

Register A = 03H Interface not open

Register HL = Not changed (For the HL contents, refer to the OPEN function.)

**RSIOX (GET)**

Function: Reads in one byte of data from the receive buffer.

Entry parameters

Register B = 50H

Register HL = Starting address of the block which is used to store return information

Return parameters

Z flag = 1 Normal completion

Register A = Receive data

Register HL = Refer to the OPEN function.

Z flag = 0 Abnormal end

Register A = 03H Interface not open

A = 00H CTRL-STOP was pressed

Register HL = not changed

### RSIOX (PUT)

Function: Sends one byte of data.

Entry parameters

Register B = 60H

Register C = Send data

Register HL= Address of the block which is used to store return information

Return parameters

Z flag = 1 Normal completion

Register HL= Refer to the OPEN function.

Z flag = 0 Abnormal end

Register A = 03H Interface not open

A = 00H CTRL-STOP was pressed

The contents of register HL is not changed.

### RSIOX (CTLIN)

Function: Reads the status of the control line.

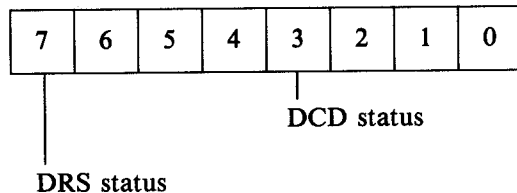
Entry parameter

Register B = 70H

Return parameters

Z flag = 1 Normal completion

Register A shows the conditions of the control line flags as follows:



Z flag = 00H Abnormal end

Register A = 03H Interface not open

### RSIOX (SETCTL)

Function: Sets the control lines.

Entry parameters

Register B = 80H

Register C = Data to be set

BIT 0 = DTR (H for "1" and L for "0")

BIT 1 = RTS (H for "1" and L for "0")

Return parameters

Z flag = 1 Normal completion

Z flag = 0 Abnormal end

Register A = 3 Interface not open

### RSIOX (ERSTS)

Function: Checks the error status and clears the error flags.

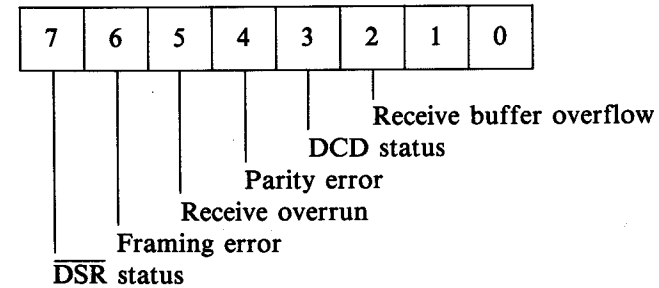
Entry parameter

Register B = 90H

Return parameters

Z flag = 1 Normal completion

Register A contains the conditions of the control line flags



Z flag = 00H Abnormal end

Register A = 03H Interface not open

### RSIOX (SENS)

Function: Checks the status of the RS232C interface.

Entry parameter  
Register B = F0H

Return parameters  
Z flag = 1 The RS232C interface can be opened.  
Z flag = 0 Busy (i.e., the port is being used by another program.)  
Register A = 02H

### 5.7.8 BIOS Subroutines related to the Clock

#### **TIMDAT**

Entry Point: WBOOT + 4BH

TIMDAT is the entry point for the subroutine which has 6 functions relating to the clock (time and date), according to the contents of the C register. The TIMDAT routine also uses a series of bytes called the Time Descriptor to read or write the description of the time. There are a total of 11 bytes in the Time Descriptor, although not always all of them may be used by or for each function. The date and time are coded in BCD. The order of the bytes are as follows.

- Year — 1 byte, 2 BCD digits
- Month — 1 byte, 2 BCD digits
- Day — 1 byte, 2 BCD digits
- Hour — 1 byte, 2 BCD digits
- Minute — 1 byte, 2 BCD digits
- Second — 1 byte, 2 BCD digits
  
- Day of week — 1 byte (0=Sunday, 6=Saturday)
- Type — 1 byte (0=none, 1=alarm, 2=wake, 3=wake subroutine).
- Address — 2 bytes (message address for alarm if type=1, wake string address if type=2 or subroutine address if type=3).
- Alarm — 1 byte (0=alarm not sounded, 1=alarm sounded).

It is important to make sure that the number of bytes of the Time Descriptor required by the function are in fact present. It is also important that the order described above is adhered to.

#### **TIMDAT (READ TIME)**

Function: Reads the time and sets data in time descriptor address.

Entry parameters  
Register C = 00H — Time read function  
Register DE = Starting address for 7 bytes of Time Descriptor information

Return Parameters  
The contents of register pair DE are not changed.  
The Year Month Day Weekday and time are placed in memory starting at the address placed in the DE register.

#### **TIMDAT (SET TIME)**

Function: Allows the date, weekday and time to be set. If any bytes of the Time Descriptor are set to FFH they will not be updated. It is also up to the programmer to check the contents of the Time Descriptor before calling this routine as no checks are made.

Entry Parameters  
Register C = FFH — set time  
Register DE = starting address of the 7 bytes of the Time Descriptor

#### **TIMDAT (ALARM ENABLE)**

Function: Enables the alarm/wake function.

Entry parameter  
Register C = 80H — Alarm enable

#### **TIMDAT (ALARM DISABLE)**

Function: Disables the alarm/wake function.

Entry parameter  
Register C = 81H — Alarm disable

#### **TIMDAT (SET ALARM)**

Function: Sets the alarm/wake time, whether it is an alarm or a wake function and the address of the location of the message/wake string. The time can be set in increments of 10 seconds. If the byte 0FH is specified, applicable digits will be regarded as wild-card positions. For example, setting FH for the year, month, and day will cause the alarm/wake function to operate at the specified hour, minute, and second every day.

#### Entry parameters

Register C = 82H — Alarm set

Register DE = Time descriptor address 10 bytes including message string address.

#### Return Parameters

The contents of register pair DE are not changed.

### TIMDAT (READ ALARM)

**Function:** Reads the Time Descriptor into the address required. Note that the year byte (byte 1) will contain FFH and the seconds byte (byte 6) contains FH in the lowest nibble (lowest 4 bits). This is because neither the year nor the 1s place of the seconds (i.e. it can only be set in 10 second intervals) is relevant for alarm setting.

#### Entry parameters

Register C = 84H

Register DE = Starting address to load the 11 bytes of the Time Descriptor

### 5.7.9 BIOS Subroutines related to the Serial Port

The Serial Port is not supported by the BIOS.

### 5.7.10 BIOS Subroutines related to memory

The memory of the PX-8 consists of 64K bytes of RAM and 32K of ROM. By means of bank switching it is possible to read addresses 0000H to 7FFFH from either RAM or ROM.

In the following routines when references are made to system bank and user bank they correspond to addresses as follows:

System bank: 0000H to 7FFF (ROM)  
8000H to FFFF (RAM)

User bank: 0000H to FFFF (RAM)

### LOADX

LOADX is the entry point for the subroutine which reads 1 byte of data from the specified memory bank.

Entry address: WBOOT + 5AH

#### Entry parameters

Register C = -1 — System bank

C = 0 — User bank

Register HL = Data address

#### Return Parameter

Register A = Data read

This routine does not change the contents of the other registers.

### STORX

(Ordinarily, this entry is not used by the user.)

STORX is the entry point for the subroutine which writes 1 byte of data into the user bank.

Entry Point: WBOOT + 5DH

#### Entry parameters

Register C = 0 — User bank

Register A = Data

Register HL = Data address into which data is to be written.

#### Return Parameters

This routine will not change the contents of the other registers.

### LDIRX

(Ordinarily, this entry is not used by the user.)

LDIRX is the entry point for the subroutine which transfers data from the specified memory bank to the user bank.

Entry Point: WBOOT + 60H

#### Entry parameters

Register HL = Starting address of the memory area from which data is transferred

Register DE = Starting address of the memory area to which data is transferred

Register BC = Number of bytes of data to be transferred

Register A = 0 — Data is transferred to the user bank.

The operating system ROM cannot be changed or modified using this routine.

### JUMPX

JUMPX is the entry point for the subroutine which jumps to the specified address in the specified memory bank.

Entry Point: WBOOT + 63H

Entry parameters

Register IX = Jump address

If Address F539H (DISBNK) = -1 System bank

If Address F539H (DISBNK) = 0 User bank

#### NOTE:

*When a stack is to be used at the destination address, a new stack must be established.*

### CALLX

CALLX is the entry point for the subroutine which calls the specified address in the specified bank.

Entry Point: WBOOT + 66H

Entry parameters

Register IX = Call address

If Address F539H (DISBNK) = -1 — System bank

If Address F539H (DISBNK) = 0 — User bank

#### NOTE:

*CALLX uses one level in the user stack. Therefore, the stack pointer must be used in the common area in RAM (8000H to 0FFFFH), and there must be at least one free level of stack space.*

### SLAVE

SLAVE is the entry point for the subroutine which sends commands and data to the 6301 slave CPU and returns results and data from the slave. The use of this command is beyond the scope of this manual. Full details are shown in the OS Reference Manual and Technical Reference Manual.

### RDVRAM

RDVRAM is the entry point for the subroutine which reads the contents of the virtual character screen.

Entry Point: WBOOT + 75H

Entry parameters

Register B = Column at which the read is to start (1 to 80)

Register C = Line at which the read is to start (1 to virtual screen line size)

Register DE = Number of characters to be read

Register HL = Starting address of the area in which data read is to be stored.

Return Parameters

Register A = 00H — Normal completion

A = 01H — Read error

A = FFH — Parameter error or graphic screen read attempted

If the end of the screen is encountered before the specified number of characters has been read, remaining addresses in the storage area are padded with spaces and 1 is returned in register A.

### 5.7.11 Miscellaneous Subroutines

#### MEMORY

Entry Point: WBOOT + 4EH

This is not supported. It will simply RETURN.

### LIGHTPEN

Entry Point: WBOOT + 54H

This is not supported. It will simply RETURN.

### MASKI

MASKI is the entry point for the subroutine which sets and resets interrupt masks.

Entry Point: WBOOT + 57H

#### Entry parameters

- Register B = 00H Interrupt disable
- B = 01H Interrupt enable
- B ≥ 2 Interrupt enable register (IER) read
- Register C Bit 0 — 7508
- C Bit 1 — 8251
- C Bit 2 — DCD (RS-232C carrier detect)
- C Bit 3 — ICF (Input capture flag)
- C Bit 4 — OVF (Timer overflow)
- C Bit 5 — EXT (Interrupt from system bus)

#### Return Parameters

Register A = Contents of IER register (I/O port 4)

### ADCVRT

ADCVRT is the entry point for the subroutine which reads the A/D converter input, Bar Code reader, DIP switch settings, battery voltage, or power switch status.

Entry Point: WBOOT + 6F

#### Entry parameters

- RegisterC=00H — A/D channel 1 (input from the analog jack)
- RegisterC=01H — A/D channel 2 (input from the bar code reader connector)
- RegisterC=02H — DIP SW1
- RegisterC=03H — Battery voltage
- RegisterC=04H — Power switch status (TRIG status of the analog connector)

When any other value is set in register C, the routine returns without doing anything.

#### Return Parameters

When Register C contains 00H or 01H on calling ADCVRT

Register A contains A/D converter data in the 6 MSB as follows:

Bit	7	6	5	4	3	2	1	0
Data	MSB					LSB		

Bits 2 to 7 will all be 1 if the input voltage is greater than 2.0 V. If the input voltage is negative, bits 2 to 7 will all be 0.

When Register C contains 2 on calling ADCVRT

Register A contains the Switch information as follows.

Bit	7	6	5	4	3	2	1	0
SW	8	7	6	5	4	3	2	1

When Register C contains 3 on calling ADCVRT

Register A contains data from the A/D converter

Full scale=5.7 V

When register C contains 4 on calling ADCVRT

- Register A, bit 0=1 — Power switch ON
- Register A, bit 0=0 — Power switch OFF
- Register A, bit 1=1 — TRIG ON
- Register A, bit 1=0 — TRIG OFF

### 5.7.12 Subroutines for using the MICROCASSETTE DRIVE

#### **MCMTX**

MCMTX is the entry point for the subroutine which calls the MTOS functions.

Entry point: WBOOT + 78H

The use of this routine is similar to using a BDOS function call. Full details are given in the OS reference manual.

### 5.7.13 Subroutines for the USER BIOS

The BIOS can be extended to include routines which are not supported by CP/M.

#### **USERBIOS**

Entry Point: WBOOT + 7EH

USERBIOS is the routine which allows USER BIOS entry points to be changed. The table of entry points is reinitialized if a reset is carried out or the system is initialized. Thus it is necessary to reload the USER BIOS entry points if any type of reset is carried out.

The use of the USER BIOS is explained in the OS Reference Manual.

## *Appendix A*

### ***PX-8 CONSOLE ESCAPE SEQUENCES***

The PX-8 uses a separate CPU to handle the console. Rather than have many calls to specific routines for particular console functions, the functions are executed by sending a sequence of characters to the console. This appendix deals with the use of these command sequences. It is also possible to use certain ASCII control codes. These are described in Appendix E.

The sequences involve the ESCAPE character ASCII code 27 decimal (1B in hexadecimal), followed by one or more characters, the values of which determine the command to be carried out. In the remainder of this appendix the ESCAPE character is denoted by the letters "ESC". The ESC code would normally be used as part of a machine code routine, by using the CONOUT routine described in section 5.7.3, although they could also be input directly from the keyboard on the CP/M command line in many cases. Since it may be necessary to enter them from the keyboard, the key sequence is given where this is possible. The character sequence can also be executed as part of a BASIC PRINT statement. The BASIC MANUAL contains further information on using the sequences in BASIC programs. Not all the commands are supported in BASIC because they interact with the screen editor.



### 5.7.12 Subroutines for using the MICROCASSETTE DRIVE

#### **MCMTX**

MCMTX is the entry point for the subroutine which calls the MTOS functions.

Entry point: WBOOT + 78H

The use of this routine is similar to using a BDOS function call. Full details are given in the OS reference manual.

### 5.7.13 Subroutines for the USER BIOS

The BIOS can be extended to include routines which are not supported by CP/M.

#### **USERBIOS**

Entry Point: WBOOT + 7EH

USERBIOS is the routine which allows USER BIOS entry points to be changed. The table of entry points is reinitialized if a reset is carried out or the system is initialized. Thus it is necessary to reload the USER BIOS entry points if any type of reset is carried out.

The use of the USER BIOS is explained in the OS Reference Manual.

## *Appendix A*

# *PX-8 CONSOLE ESCAPE SEQUENCES*

The PX-8 uses a separate CPU to handle the console. Rather than have many calls to specific routines for particular console functions, the functions are executed by sending a sequence of characters to the console. This appendix deals with the use of these command sequences. It is also possible to use certain ASCII control codes. These are described in Appendix E.

The sequences involve the ESCAPE character ASCII code 27 decimal (1B in hexadecimal), followed by one or more characters, the values of which determine the command to be carried out. In the remainder of this appendix the ESCAPE character is denoted by the letters "ESC". The ESC code would normally be used as part of a machine code routine, by using the CONOUT routine described in section 5.7.3, although they could also be input directly from the keyboard on the CP/M command line in many cases. Since it may be necessary to enter them from the keyboard, the key sequence is given where this is possible. The character sequence can also be executed as part of a BASIC PRINT statement. The BASIC MANUAL contains further information on using the sequences in BASIC programs. Not all the commands are supported in BASIC because they interact with the screen editor.

## A-1 Alphabetic Table of sequences

The following table shows an alphabetical list to enable the character sequence for each command to be found. Notes on the use of the commands and parameters are given in numerical order following the table. The numerical values are given in decimal and hexadecimal notation in the table and headings.

Control Code	Decimal	Hexadecimal	Function
ESC “%”	27,37	1B,25	Access CGROM directly
ESC 243	27,243	1B,F3	Arrow key code
ESC 246	27,246	1B,F6	Buffer clear key
ESC 163	27,163	1B,A3	CAPS LED off
ESC 162	27,162	1B,A2	CAPS LED on
ESC “C”	27,67	1B,43	Character table
ESC 246	27,246	1B,F6	Clear key buffer
ESC “*”	27,42	1B,2A	Clear screen
ESC 245	27,245	1B,F5	CTRL key code
ESC 215	27,215	1B,D7	Cursor find
ESC 243	27,243	1B,F3	Cursor key code
ESC “2”	27,50	1B,32	Cursor off
ESC “3”	27,51	1B,33	Cursor on
ESC “=”	27,61	1B,3D	Cursor position set
ESC 214	27,214	1B,D6	Cursor type select
ESC “P”	27,80	1B,50	Dump screen
ESC “T”	27,84	1B,54	Erase to end of line
ESC “Y”	27,89	1B,59	Erase to end of screen
ESC 210	27,210	1B,D2	Display characters on real screen
ESC 208	27,208	1B,D0	Display mode set
ESC 198	27,198	1B,C6	Dot line write
ESC 224	27,224	1B,E0	Download user defined character
ESC 213	27,213	1B,D5	End locate
ESC 215	27,215	1B,D7	Find cursor
ESC 176	27,176	1B,B0	Function key code returned
ESC 177	27,177	1B,B1	Function key string returned
ESC 211	27,211	1B,D3	Function key display select
ESC “C”	27,67	1B,43	International character sets
ESC 161	27,161	1B,A1	INS LED off
ESC 160	27,160	1B,A0	INS LED on

Control Code	Decimal	Hexadecimal	Function
ESC 242	27,242	1B,F2	Key repeat interval time
ESC 240	27,240	1B,F0	Key repeat on/off
ESC 241	27,241	1B,F1	Key repeat start time
ESC 244	27,244	1B,F4	Key code scroll
ESC 247	27,247	1B,F7	Key shift set
ESC “T”	27,84	1B,54	Line erase
ESC 198	27,198	1B,C6	Line dot draw
ESC 213	27,213	1B,D5	Locate end of screen
ESC 212	27,212	1B,D4	Locate top of screen
ESC 125	27,125	1B,7D	Non secret
ESC 165	27,165	1B,A5	NUM LED off
ESC 164	27,164	1B,A4	NUM LED on
ESC 199	27,199	1B,C7	PSET/PRESET
ESC 242	27,242	1B,F2	Repeat key interval time
ESC 240	27,240	1B,F0	Repeat on/off for keys
ESC 241	27,241	1B,F1	Repeat start time for keys
ESC “*”	27,42	1B,2A	Screen clear
ESC 151	27,151	1B,97	Screen down
ESC 209	27,209	1B,D1	Screen display select
ESC “P”	27,80	1B,50	Screen dump
ESC 213	27,213	1B,D5	Screen window end
ESC “Y”	27,89	1B,59	Screen erase
ESC 212	27,212	1B,D4	Screen window top
ESC 145	27,145	1B,91	Scroll down
ESC 244	27,244	1B,F4	Scroll key code
ESC 148	27,148	1B,94	Scroll step
ESC 149	27,149	1B,95	Scroll mode
ESC 144	27,144	1B,90	Scroll up
ESC 151	27,151	1B,97	Screen down n lines
ESC 150	27,150	1B,96	Screen up n lines
ESC 123	27,123	1B,7B	Secret mode
ESC 125	27,125	1B,7D	Secret mode cancel
ESC 214	27,124	1B,D6	Select cursor type
ESC 209	27,209	1B,D1	Select display screen
ESC 211	27,211	1B,D3	Select function key display
ESC 247	27,247	1B,F7	Shift key set
ESC 149	27,149	1B,95	Tracking mode
ESC 212	27,212	1B,D4	Top locate
ESC 224	27,224	1B,E0	User defined characters

## A-2 Use of the ESCAPE Code Control Sequences

Where numeric codes must be given from the keyboard (shown as n or m in the escape sequence — these are ASCII codes) the key or key combination giving that code can be found in the table in Appendix E. Some sequences cannot be obtained direct from the keyboard. Also some control codes (e.g. CTRL-M which is equivalent to pressing the RETURN key) will function normally and so terminate the input.

### Example 1 ESC control sequence from the CP/M command line

The following example shows how to turn the cursor off and then back on again on the CP/M command line.

Press the ESC key followed by the number “2” then press the RETURN key.

The display will show:

```
A>^[2 RETURN
?
A>
```

with no cursor after the CP/M prompt. To make the cursor display return, type the ESC key followed by the number “3”. The question mark in either case means CP/M does not understand the command.

### Example 2 The Use of the CONOUT routine

The CONOUT routine (section 5.7.3) be used to execute the console ESC sequences, for example to change the cursor to non flashing.

```
LD C.1BH
CALL CONOUT
LD C. D6H
CALL CONOUT
LD C. 01H
CALL CONOUT
```

## A-3 The ESC Sequences

### ESC “%”

Reads the character corresponding to the specified code from the character generator ROM and displays it at the present cursor position in the currently selected screen (in the virtual screen for modes 0, 1, and 2, and in the real screen for mode 3). The sequence is as follows:

Keyboard	ESC,%n
Decimal	27,37,n
Hexadecimal	1B,25,n

The value of n is the ASCII code corresponding to the character to be displayed. This code cannot be obtained directly from the keyboard.

### ESC “\*”

Clears the currently selected screen and moves the cursor to the home position. The **CLR** key or **CTRL** - **L** performs the same function.

Keyboard	ESC,*
Decimal	27,42
Hexadecimal	1B,2A

### ESC “2”

Turns off display of the cursor. However, the cursor is still present even though it cannot actually be seen.

Keyboard	ESC,2
Decimal	27,50
Hexadecimal	1B,32

The cursor can also be turned on and off by using the CONFIG program.

### ESC “3”

Turns on the cursor.

Keyboard	ESC,3
Decimal	27,51
Hexadecimal	1B,53

## ESC “=”

Moves the cursor to the specified position on the current screen. In the tracking mode, the screen window is moved so that the cursor is positioned at the screen center if the position specified is outside the screen window. The tracking mode is turned on and off by pressing the shift and SCRNLK keys together, or by using the sequence ESC 149. The sequence for moving the cursor is as follows:

Keyboard ESC, =, (m + 31), (n + 31)  
Decimal 27,61,m + 31,n + 31  
Hexadecimal 1B,3D,m + 1F,n + 1F

Here, m specifies the vertical cursor position and n specifies the horizontal position. The value of n should be greater than 1 and less than the screen width in the particular screen mode being used. The value of m should be greater than 1 and less than the number of lines in the virtual screen.

## ESC “C” <character>

Used to select one of the nine international character sets as follows:

The <character> is a letter which corresponds to the character sets of one of the following countries.

	Keyboard	Decimal	Hexadecimal
US ASCII	ESC,C,U	27,67,85	1B,43,55
France	ESC,C,F	27,67,70	1B,43,46
Germany	ESC,C,G	27,67,71	1B,43,47
England	ESC,C,E	27,67,69	1B,43,45
Denmark	ESC,C,D	27,67,68	1B,43,44
Sweden	ESC,C,W	27,67,87	1B,43,57
Italy	ESC,C,I	27,67,73	1B,43,49
Spain	ESC,C,S	27,67,83	1B,43,53
Norway	ESC,C,N	27,67,78	1B,43,4E

This code sequence is equivalent to the BASIC OPTION COUNTRY command. The “country” option of the CONFIG program can also be used to change the full character set.

## ESC “P”

In screen modes 0, 1, and 2 this escape sequence outputs the contents of the screen window currently being displayed to a printer in ASCII format. In mode 3 it outputs the contents of the entire physical screen in bit image format. It duplicates the COPY command of BASIC or screen dump function obtained by pressing the CTRL and PF5 key.

Keyboard ESC,P  
Decimal 27,80  
Hexadecimal 1B,50

## ESC “T”

Clears the line currently containing the cursor from its present position to the end of that logical line.

Keyboard ESC,T  
Decimal 27,84  
Hexadecimal 1B,54

## ESC “Y”

Clears the screen from the current position of the cursor to the end of the screen.

Keyboard ESC,Y  
Decimal 27,89  
Hexadecimal 1B,59

## ESC 123

Causes all characters to be displayed on the screen as blanks (the secret mode). The secret mode is not active in the System Display.

Keyboard ESC,{  
Decimal 27,123  
Hexadecimal 1B,7B



### WARNING:

*You should make sure that a program returns the user to normal non-secret mode, for example with an error handling routine. If the user is placed in immediate mode and the secret mode is still active, it is impossible to know what is happening. Also the reset button on the left of the*

*PX-8 must be pressed in order to see any printed output except for the clock on the MENU page and the System Display.*

#### ESC 125

Terminates the secret mode.

Keyboard	ESC, <sub>f</sub>
Decimal	27,125
Hexadecimal	1B,7D

#### ESC 144

Scrolls (m - 1) lines starting at line (n + 1) up one line so that line (n + m - 1) becomes blank. This is done as follows:

Keyboard	ESC,GRPH-U,(n - 1),(m)
Decimal	27,144,n - 1,m
Hexadecimal	1B,90,n - 1,m

Numbers specified for n and m must satisfy all of the following conditions.

$$\begin{aligned}0 &\leq (n - 1) < (R - 1) \\1 &\leq m \leq R \\(n - 1 + m - 1) &< R\end{aligned}$$

Here, R is the number of virtual screen lines in mode 0, 1, or 2 and is the number of screen window lines in mode 3.

#### ESC 145

Scrolls the (m - 1) lines starting at line n down one line so that line n becomes blank. This is done as follows:

Keyboard	ESC,GRPH-I,(n - 1),(m)
Decimal	27,145,n - 1,m
Hexadecimal	1B,91,n - 1,m

Numbers specified for n and m must satisfy all of the following conditions:

$$\begin{aligned}0 &\leq (n - 1) \leq (R - 1) \\1 &\leq m < R \\(n - 1 + m - 1) &< P\end{aligned}$$

Here, R is the number of virtual screen lines in mode 0, 1, or 2 and is the number of screen window lines in mode 3.

#### ESC 148

In modes 0, 1, and 2 this escape sequence sets the number of lines n which are moved by one scrolling operation. The actual scrolling is carried out by printing an ESC 150 sequence. The number of lines are set up using the following sequence:

Keyboard	ESC,GRPH-@,(n)
Decimal	27,148,n
Hexadecimal	1B,94,n

The number specified for n must be greater than 1 and less than the number of lines in the screen window.

This escape sequence does nothing in mode 3.

#### ESC 149

In modes 0, 1, and 2 this escape sequence determines whether scrolling is performed automatically. The automatic scrolling mode is referred to as the tracking mode, and the mode in which automatic scrolling is not performed is referred to as the non-tracking mode. The tracking mode is used unless otherwise specified. The escape sequence for determining the tracking mode is as follows:

Keyboard	ESC,GRPH-K,CTRL-@ for tracking mode ESC,GRPH-K,CTRL-A for non-tracking mode
Decimal	27,149,0 or 1
Hexadecimal	1B,95,0 or 1

In this sequence, <mode> is specified as either 0 or 1. The tracking mode is selected when 0 is specified, and the non-tracking mode is selected when 1 is specified.

Normally the **SCRN** key or CONFIG command can be used to switch between the tracking and non-tracking modes.

### ESC 150

In modes 0, 1, and 2 this escape sequence displays the contents of the virtual screen containing the cursor after moving the screen window up n lines where n is the value specified by ESC 148, or 1 if ESC 148 has not been executed. If scrolling the screen up n lines would move the screen window beyond the home position, the virtual screen is displayed starting at the home position. The cursor remains in its original position in the virtual screen.

Keyboard ESC,GRPH-V  
Decimal 27,150  
Hexadecimal 1B,96

### ESC 151

In modes 0, 1, and 2 this escape sequence displays the contents of the virtual screen containing the cursor after moving the screen window down n lines, where n is the value specified by ESC 148, or 1 if ESC 148 has not been executed. If scrolling the screen down n lines would move the screen window beyond the end of the virtual screen, the screen window is positioned so that the virtual screen's last line is displayed in the last line of the screen window. The cursor remains in its original position in the virtual screen.

Keyboard ESC,GRPH-,(comma)  
Decimal 27,151  
Hexadecimal 1B,97

### ESC 160

Lights the INS LED. It does not put the user in insert mode.

Decimal 27,160  
Hexadecimal 1B,A0

### ESC 161

Turns off the INS LED.

Decimal 27,161  
Hexadecimal 1B,A1

### ESC 162

Lights the CAPS LED. It does not set the caps lock key to the on position.

Decimal 27,162  
Hexadecimal 1B,A2

### ESC 163

Turns off the CAPS LED.

Decimal 27,163  
Hexadecimal 1B,A3

### ESC 164

Lights the NUM LED, but does not set the numeric keypad.

Decimal 27,164  
Hexadecimal 1B,A4

### ESC 165

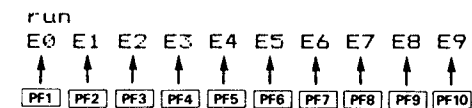
Turns off the NUM LED.

Decimal 27,165  
Hexadecimal 1B,A5

### ESC 176

Returns the key codes of the programmable function keys, and disable the string printed by them. (from E0 to E9)

Decimal 27,176  
Hexadecimal 1B,B0



### ESC CHR\$(177)

This ESC code re-enables the programmable function keys so that a string is printed when they are pressed.

### ESC 198

In mode 3, this escape sequence draws a line on the graphic screen using the

dot pattern specified by the user. No operation is performed when this sequence is executed in modes 0, 1, or 2. The elements of the sequence are as follows:

- Byte 1: Decimal 27     Hexadecimal 1B
- Byte 2: Decimal 198     Hexadecimal C6
- Byte 3: High byte of horizontal starting position
- Byte 4: Low byte of horizontal starting position
- Byte 5: High byte of vertical starting position
- Byte 6: Low byte of vertical starting position
- Byte 7: High byte of horizontal ending position
- Byte 8: Low byte of horizontal ending position
- Byte 9: High byte of vertical ending position
- Byte 10: Low byte of vertical ending position
- Byte 11: First byte of mask pattern
- Byte 12: Second byte of mask pattern
- Byte 13: Function

The starting and ending positions are specified as two-byte hexadecimal numbers which indicate co-ordinates in the graphic screen. For example, starting co-ordinates of 400,20 (&H0190,&H0014) would be specified as follows:

- Byte 3:     1 (&H01)
- Byte 4:    144 (&H90)
- Byte 5:     0 (&H00)
- Byte 6:    20 (&H14)

The mask pattern used for drawing the line is specified in bit image format as described in the explanation of the LINE statement in Chapter 4 of the PX-8 BASIC Manual. Calculations for diagonal lines are performed automatically. Function is specified as a number from 1 to 3 with the following meanings:

- 1: OFF
- 2: ON
- 3: Complement

Dot positions corresponding to "1" bits in the mask pattern are re-set (turned off) when 1 is specified for the function and are set (turned on) when 2 is specified. When 3 is specified, the complements of dots corresponding to "1" bits are displayed (ON dots corresponding to "1" bits are turned off, and OFF dots are turned on).

An example of specification of this sequence as follows draws a line from point (400,18) of the screen to point (18,18):

Decimal            27,198,1,144,0,18,0,18,0,18,170,170,2  
 Hexadecimal     1B,C6,1,90,0,12,0,12,0,12,AA,AA,2

This command duplicates the LINE command of BASIC, but also allows the dots to be inverted (i.e. switch them on if they are off and vice versa), which LINE does not.

### ESC 199

This escape sequence sets or re-sets the specified points of the graphic screen. No operation is performed if this sequence is executed in modes 0, 1, or 2. The sequence consists of six bytes as follows:

- Byte 1: Decimal 27     Hexadecimal 1B
- Byte 2: Decimal 199     Hexadecimal C7
- Byte 3: Function code (1: PSET, 0: PRESET)
- Byte 4: Vertical dot position — n1
- Byte 5: High byte of horizontal dot position — n2
- Byte 6: Low byte of horizontal dot position

Numbers specified for n1 and n2 must be in the following ranges:

Decimal             $0 \leq n1 \leq 63, 0 \leq n2 \leq 479$   
 Hexadecimal      $0 \leq n1 \leq 3F, 0 \leq n2 \leq 1DF$

### ESC 208

Switches the display mode. Mode specification is as follows:

	Dec	Hex		Dec	Hex
Mode 0			Mode 2		
Byte 1:	27	1B	Byte 1:	27	1B
Byte 2:	208	D0	Byte 2:	208	D0
Byte 3:	2	2	Byte 3:	2	2
Byte 4:	n1	n1	Byte 4:	n1	n1
Byte 5:	n2	n2	Byte 5:	m	m
			Byte 6:	p	p
Mode 1			Mode 3		
Byte 1:	27	1B	Byte 1:	27	1B
Byte 2:	208	D0	Byte 2:	208	D0
Byte 3:	1	1	Byte 3:	3	3
Byte 4:	n1	n1			

The meanings of n1, n2, m, and p are as follows:

- n1 Number of lines in virtual screen 1
- n2 Number of lines in virtual screen 2
- m Number of columns in virtual screen 1
- p ASCII code corresponding to desired boundary character

The following sequence selects screen mode 2, sets the number of lines in virtual screen 1 to 10, the number of columns to 20 and “#” as the boundary character.

Decimal 27,208,2,10,20,35  
Hexadecimal 1B,D0,2,A,14,23

#### ESC 209

In modes 0, 1, or 2 this escape sequence specifies which of the two virtual screens is to be displayed. The operation is performed if this sequence is executed in mode 3. This is done as follows:

Decimal 27,209,n  
Hexadecimal 1B,D1,n

The first virtual screen is selected when 0 is specified for n, and the second virtual screen is selected when 1 is specified for n. If the third byte is not specified the default is 0.

#### ESC 210

Displays the specified character in the specified position on the real screen. This is done as follows:

Decimal 27,210,x,y,p  
Hexadecimal 1B,D2,x,y,p

The meanings of x, y and p are as follows:

- x Vertical position (1 to 8)
- y Horizontal position (1 to 80 decimal, 1 to 50 hexadecimal)
- p ASCII character code of the Character Generator ROM

This sequence makes it possible to output characters to any location in the real

screen, regardless of the position of the cursor or number of lines in the screen window.

#### ESC 211

Turns on or off display of function key definitions. This is done as follows:

Decimal 27,211,n  
Hexadecimal 1B,D3,n

Function key definitions are displayed when 0 is specified for n, and are not displayed when 1 is specified. The default value is 1.

#### ESC 212

In modes 0, 1, and 2 this escape sequence moves the screen window to the top of the virtual screen containing the cursor. No operation is performed if this sequence is executed in mode 3. The position of the cursor remains unchanged.

Decimal 27,212  
Hexadecimal 1B,D4

#### ESC 213

In modes 0, 1, and 2 this escape sequence moves the screen window to the end of the virtual screen containing the cursor. No operation is performed if this sequence is executed in mode 3. The position of the cursor remains unchanged.

Decimal 27,213  
Hexadecimal 1B,D5

#### ESC 214

In modes 0, 1, and 2 this escape sequence selects the type of cursor to be displayed. In mode 3 always cursor type 3 is selected. The sequence consists of three bytes as follows:

Byte 1 :	Decimal 27	Hexadecimal 1B
Byte 2:	Decimal 214	Hexadecimal D6
Byte 3:	n	

Here, n specifies the type of cursor displayed as follows:



- 0 Block cursor, blink
- 1 Block cursor, non-blinking
- 2 Underline cursor, blink
- 3 Underline cursor, non-blinking

### ESC 215

In modes 0, 1, and 2 this escape sequence moves the screen window to the position occupied by the cursor. This sequence does nothing if executed in mode 3. The screen window is positioned so that the cursor is located near its centre.

Decimal 27,215  
Hexadecimal 1B,D7

### ESC 224

This escape sequence defines those characters corresponding to ASCII codes 224 (&HE0) to 254 (&HFE). This sequence consists of eleven bytes as follows:

Keyboard	Decimal	Hexadecimal
Byte 1: ESC	27	1B
Byte 2: GRPH-0	224	E0
Byte 3: Character code		
Byte 4: Pattern for dot row 1		
Byte 5: Pattern for dot row 2		
Byte 6: Pattern for dot row 3		
Byte 7: Pattern for dot row 4		
Byte 8: Pattern for dot row 5		
Byte 9: Pattern for dot row 6		
Byte 10: Pattern for dot row 7		
Byte 11: Pattern for dot row 8		

The pattern making up each dot row is specified as the ASCII code equivalent of the binary number whose "1" bits correspond to dots which are turned on, and whose "0" bits correspond to dots which are turned off. For example, specifying 63 (where 63 is the decimal equivalent of 00111111B) for byte 4 causes all dots in dot row one to be turned on when the character code specified in byte 3 is displayed; conversely, specifying 0 (00000000) causes all dots in the applicable row to be turned off.

A sample definition for character code 230 is shown below:

Keyboard	ESC,GRPH-0,GRPH-6,CTRL-L,CTRL-L,CTRL-^,?,CTRL-L, CTRL-R, CTRL-@, CTRL-@
Decimal	27,224,230,12,12,30,63,12,18,0,0
Hexadecimal	1B,E0,E6,C,C,1E,3F,C,12,0,0

The altered code can be seen by pressing GRPH-6.

### ESC 240

Controls the key repeat function. This sequence consists of three bytes as follows:

	Keyboard	Decimal	Hexadecimal
Byte 1:	ESC	27	1B
Byte 2:	CTRL-0	240	F0
Byte 3:	(n)	n	n

If 0 is specified for n, the repeat function is turned off. If 1 is specified, it is turned on.

### ESC 241

Sets the starting time for the key repeat function. The sequence consists of three bytes as follows:

	Keyboard	Decimal	Hexadecimal
Byte 1:	ESC	27	1B
Byte 2:	CTRL-1	241	F1
Byte 3:	(n)	n	n

The repeat function starting time is equal to n/64 seconds where n is a number from 1 to 127 (decimal) or 1 to 7F (hexadecimal).

### ESC 242

Sets the duration of the key repeat interval. This sequence consists of three bytes as follows:

	Keyboard	Decimal	Hexadecimal
Byte 1:	ESC	27	1B
Byte 2:	CTRL-2	242	F2
Byte 3:	(n)	n	n

The key repeat interval is equal to n/256 seconds, where n is a number from 1 to 127 (decimal) or 1 to 7F (hexadecimal).

### ESC 243

Sets the arrow key codes. This sequence consists of six bytes as follows:

	Keyboard	Decimal	Hexadecimal
Byte 1:	ESC	27	1B
Byte 2:	CTRL-3	243	F3
Byte 3:	Code for →		
Byte 4:	Code for ←		
Byte 5:	Code for ↑		
Byte 6:	Code for ↓		

### ESC 244

Sets the SHIFT + arrow key codes. This sequence consists of six bytes as follows:

	Keyboard	Decimal	Hexadecimal
Byte 1:	ESC	27	1B
Byte 2:	CTRL-4	244	F4
Byte 3:	Code for SHIFT + →		
Byte 4:	Code for SHIFT + ←		
Byte 5:	Code for SHIFT + ↑		
Byte 6:	Code for SHIFT + ↓		

### ESC 245

Sets the CTRL + arrow key codes. This sequence consists of six bytes as follows:

	Keyboard	Decimal	Hexadecimal
Byte 1:	ESC	27	1B
Byte 2:	CTRL-5	245	F5
Byte 3:	Code for CTRL + →		
Byte 4:	Code for CTRL + ←		
Byte 5:	Code for CTRL + ↑		
Byte 6:	Code for CTRL + ↓		

For further details of the arrow code setting see the OS Reference Manual.

### ESC 246

Clears the keyboard buffer of all unprocessed input characters.

Keyboard	ESC,CTRL-6
Decimal	27,246
Hexadecimal	1B,F6

### ESC 247

The ESC 247 code allows the programmer to switch the various shift keys on and off. Thus the numeric key pad can be set on, or the shift key 'held down'. The key state is set to normal by the user pressing the appropriate key, so it is advisable to program with the possibility in mind that the key may be reset outside program control.

The sequence of characters is as follows:

	Keyboard	Decimal	Hexadecimal
Byte 1:	ESC	27	1B
Byte 2:	CTRL-7	247	F7
Byte 3:	(n)	n	n

Numbers which may be specified for n and their meanings are as follows:

Keyboard	Decimal	Hexadecimal	Shift state
CTRL-@	0	0	Normal
CTRL-B	2	2	SHIFT
CTRL-D	4	4	CAPS LOCK
CTRL-F	6	6	CAPS LOCK SHIFT
CTRL-P	16	10	NUM
CTRL-R	18	12	Numeric SHIFT
SPACE	32	20	GRPH
"	34	22	GRPH SHIFT
@	64	40	CTRL
B	66	42	CTRL SHIFT

This sequence does nothing if numbers other than those above are specified for n.

## *Appendix B*

# **DRIVE NAME ASSIGNMENTS**

The default drive names assigned to the physical drives are as follows:

DRIVE NAME	PHYSICAL DRIVE
A:	RAM disk
B:	ROM 1
C:	ROM 2
D:	FDD 1
E:	FDD 2
F:	FDD 3
G:	FDD 4
H:	Microcassette drive

The drive assignments can be changed using the CONFIG program described in Chapter 3.

It is not possible to assign any drive other than the Microcassette Drive to drive H:, nor is it possible to have the Microcassette drive assigned to any other drive.

## *Appendix C*

# **THE BATTERIES**

The PX-8 uses two sets of batteries. The main batteries are used to provide power for running the PX-8. The back-up batteries hold programs in memory when the main battery voltage falls below a useable value.

This appendix covers points related to charging, switching off and removing the batteries, and replacing them.

Some precautions on using the batteries are given in section 2.1.2. To prevent the batteries running down during transportation or storage, the batteries are disconnected during shipment. Your dealer will normally connect them and initialize the PX-8. Otherwise follow the procedures outlined in Chapter 2.

### **C-1 Charging the Batteries**

When the main battery voltage falls below a useable value, the message:

#### **CHARGE BATTERY**

will appear on the screen. The PX-8 will then use the backup batteries to preserve the programs and data in memory. If it is convenient to use the AC adapter, the PX-8 can be used while the batteries are being charged, although they will take longer to charge.

IF NO DISPLAY IS VISIBLE WHEN THE COMPUTER IS FIRST SWITCHED ON, the batteries may be discharged. The other possibility is that the view angle of the LCD screen may be set incorrectly. When the AC adaptor is plugged in, the batteries may be so low that not enough power is available to allow the PX-8 to operate and to charge the batteries. In this case no display will appear. Wait 10-15 seconds and switch the main power on again with the AC adapter inserted.

The batteries will take about 8 hours to charge if the PX-8 is not in use. To prevent overcharging the batteries will trickle charge after 8 hours. If the batteries are being charged while the computer is in use, the batteries may not become fully charged. Thus charging should be repeated for at least 8 hours after a session in which the PX-8 has been used with the AC adapter attached. This charging should be carried out as follows:

- 1) Switch the main power switch to the OFF position.
- 2) Remove the AC adapter plug.
- 3) Replace the AC adapter plug.
- 4) Charge for 8 hours.

## C-2 Problems with Recharging.

If the PX-8 has been left for an extended period of time, the backup batteries will continue to be charged from the main batteries. At some stage the voltage of the backup batteries will reach a level below which it is not possible to restart normal operation of the PX-8. In this case proceed as follows:

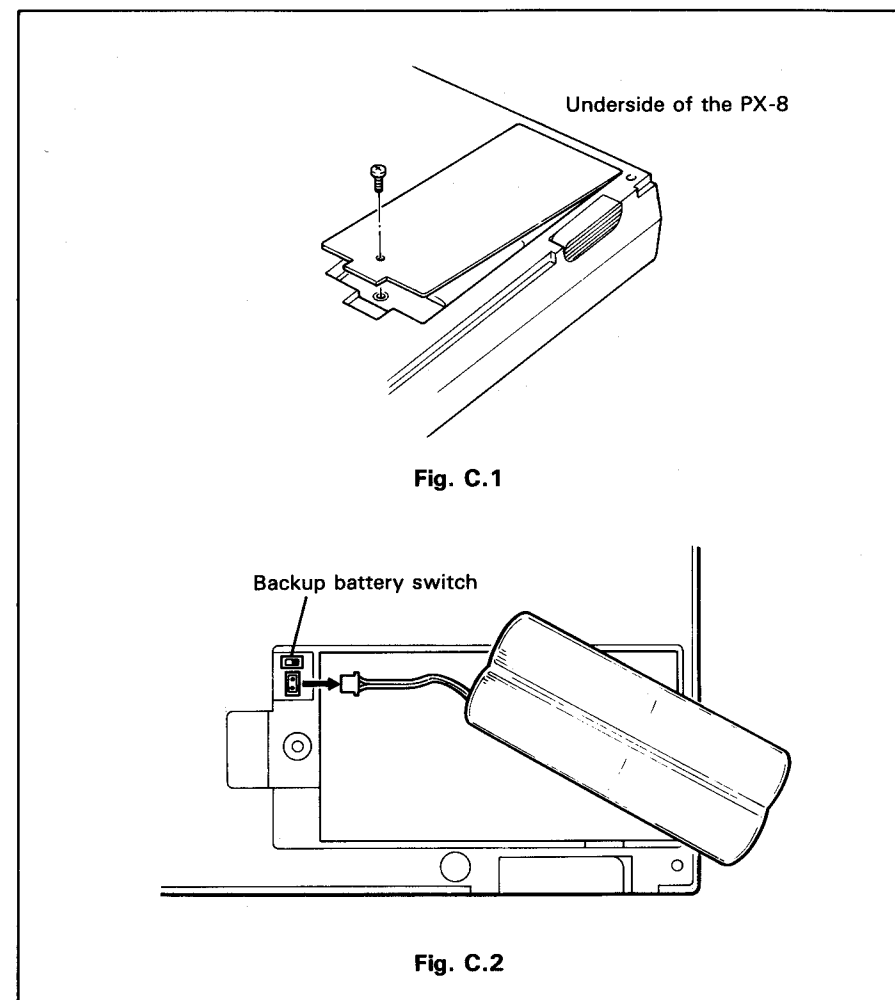
- 1) Charge the batteries with the AC adapter for 10—15 seconds.
- 2) Turn the main power switch on the side of the PX-8 to the ON position.
- 3) If normal operation does not occur, open the cover to expose the DIP switch and 7508 sub-CPU reset switch.
- 4) Press the sub-CPU reset switch, and the initialization message should appear on the screen as described in section 2.1.3.
- 5) If nothing happens at this stage, consult your EPSON dealer.

## C-3 Switching off the Batteries during Extended Storage.

If the PX-8 is not used for an extended period of time the battery charge will gradually be reduced as the power is used to charge the backup batteries. If this continues for too long, the battery will become overdischarged. This can cause deterioration of the battery. In extreme cases, after a very long time, it can result in the batteries leaking and damaging the computer.

If the PX-8 is to be left unused for a period of sixth months or more, the batteries should be switched off and disconnected as follows:

- 1) **SAVE ALL PROGRAMS AND DATA ON TAPE OR DISK.** All programs and data in memory, including the RAM disk and Intelligent RAM Disk will be lost when the batteries are switched off.
- 2) Switch the backup battery switch to off.
- 3) Open the battery cover (Fig. C.1) and unplug the main battery lead.
- 4) Replace the cover.



## **C-4 Connecting the Batteries after Extended Storage, or on Purchase.**

Connection Sequence:

- 1) Check that the power switch is set to the OFF position.
- 2) Turn the PX-8 upside down and find the battery cover position as in fig C.1
- 3) Remove the battery retaining screw and battery cover.
- 4) Check the positions of the main battery socket and switch for the backup battery. (Fig C.2)
- 5) Insert the battery lead connector into the socket making sure it fits in the correct way.
- 6) Press the center of the connector with an insulated implement, such as a match, to make the connection firm. **DO NOT USE** an conductive material as this could short the battery.
- 7) Set the slide switch to the ON position to connect the backup battery. Again use an insulated instrument to do this.
- 8) Install the battery cover and tighten the retaining screw, taking care not to trap the battery leads.
- 9) Initialize the PX-8 as outlined in Chapter 2 section 2.1.3

## **C-5 Replacing the Main Battery**

The main battery is expected to have a life of 3-4 years, depending on the type and frequency of use. If the "CHARGE BATTERY" message comes up frequently at this time, the main battery needs replacing. Consult your EPSON dealer for a replacement battery. If you are unsure about any part of the following procedure, please ask your dealer to change the battery for you.

To change the battery proceed as follows:

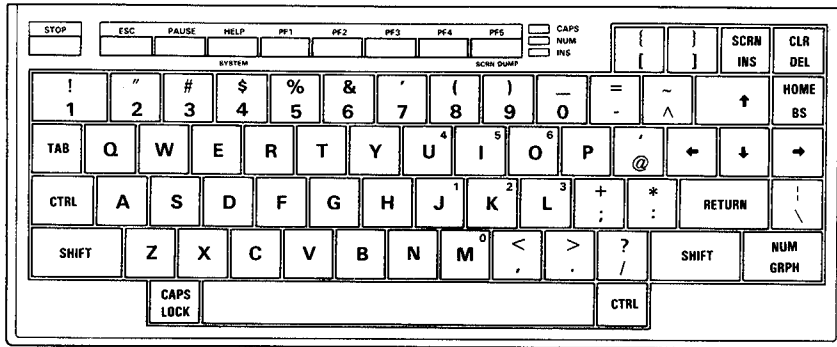
- 1) **SAVE ALL PROGRAMS AND DATA ON TAPE OR DISK.** All programs and data in memory, including the RAM disk and Intelligent RAM Disk may be lost when the batteries are changed.
- 2) Leave the PX-8 switched on until the "CHARGE BATTERY" message comes up, i.e. the main battery becomes discharged.
- 3) Turn the main power switch on the side of the PX-8 to the OFF position.
- 4) Open the battery cover. Remove the lead to the main battery, and dispose of the battery in a responsible way. Do not puncture the battery or throw it onto a fire or into water. Do not attempt to take the battery apart.
- 5) Do **NOT** switch the backup battery switch to OFF.

- 6) Plug in the lead from the new battery.
- 7) Press the reset switch on the left side of the PX-8 while switching the main power switch on. Then release the reset switch. This should be done on replacing the battery and not left until it is necessary to use the PX-8.
- 8) The supplied battery will only be partially charged and may become completely discharged if it is not charged immediately after replacement. Therefore, charge the battery following the procedure in section C-1, even though the "CHARGE BATTERY" message may not be displayed.

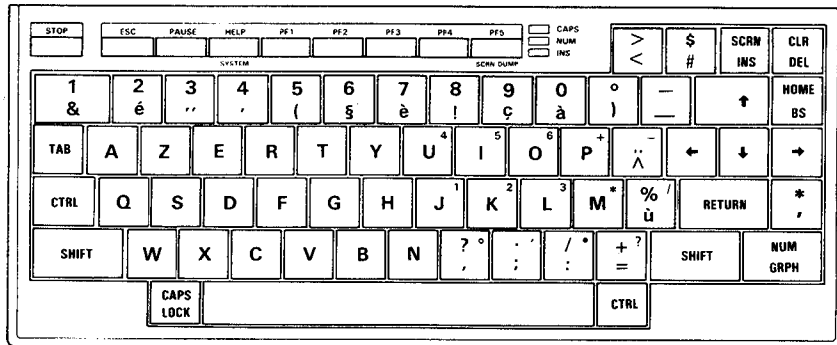
# Appendix D

## KEYBOARD LAYOUTS BY COUNTRY

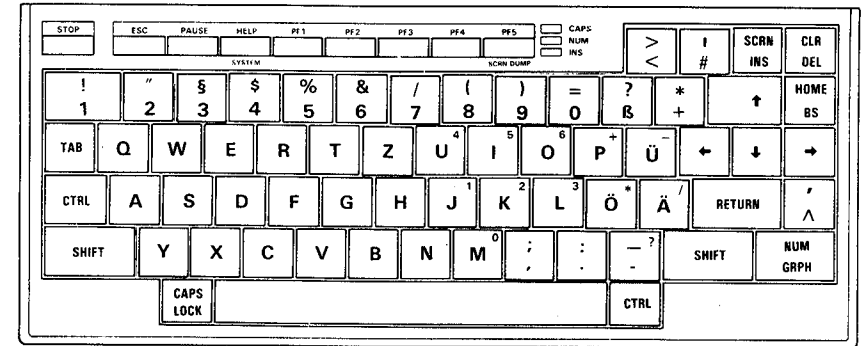
### 1) ASCII



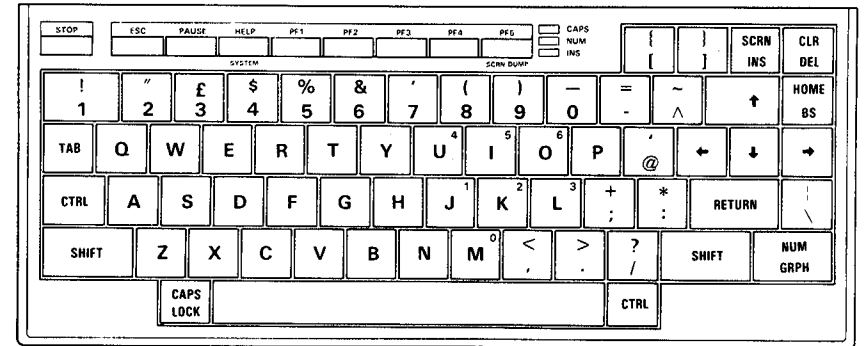
### 2) France



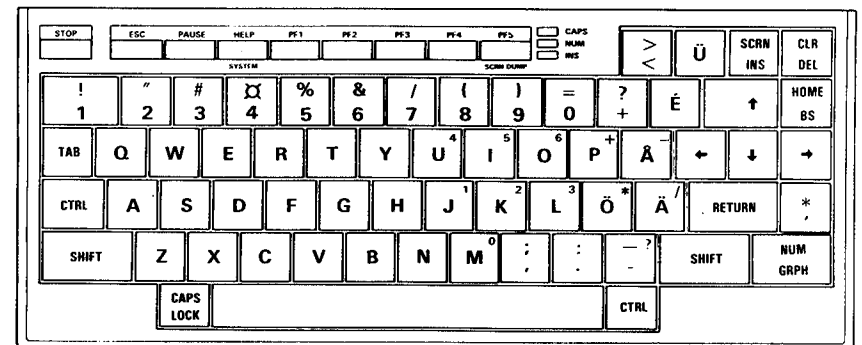
### 3) Germany



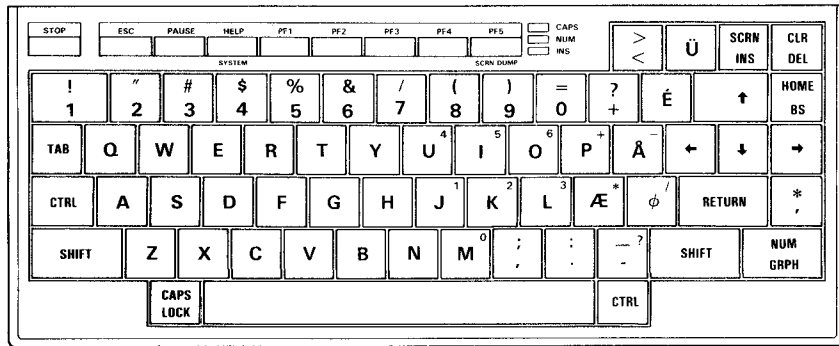
### 4) England



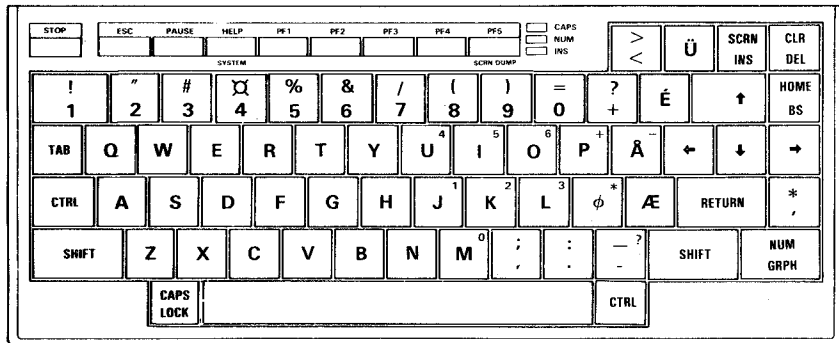
### 5) Sweden



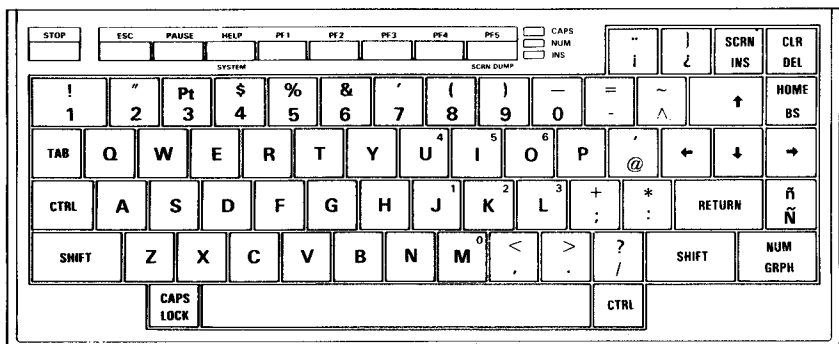
6) Denmark



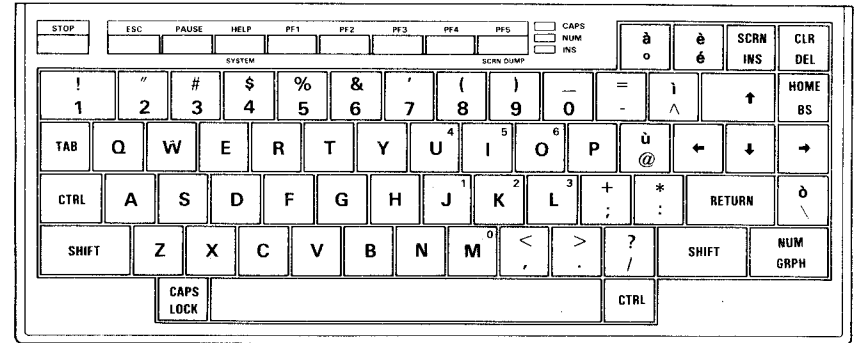
7) Norway



8) Spain



9) Italy



## Appendix E

# ASCII CODE TABLE AND INTERNATIONAL CHARACTER SETS

### E-1 ASCII codes and setting international character sets

Since a computer and its associated peripheral devices are essentially numerical machines, in order to produce “human readable” output, either on a screen or printer, there has to be a code relating the alphanumeric characters to the numbers the computer uses. This code has been standardised so that different computer, printers etc., produce the correct output. The standard is called the ASCII code. The letters ASCII (pronounced ASKEE) stand for American Standard Code for Information Interchange. There are of necessity slight differences when dealing with country to country variations. The table in this appendix shows the standard ASCII character set for the PX-8. This can be changed to display characters from a number of different countries in two ways.

- i) The keyboard layout can be changed using the DIP switch 4 as shown in section 2.1.2. If the layout is changed in this way, the legend on the key-top may not correspond to the character obtained when the key is pressed. For example if the French keyboard has been set up using the DIP switch when the keyboard legends are ASCII, numbers can be obtained by pressing the SHIFT key as well as the numerical keys. The unshifted keys will give a different output. The layouts for the different keyboards are shown in Appendix D.
- ii) The characters corresponding to certain ASCII codes can be changed by using either the CONFIG program, or the ESC code sequence ESC C <character>, where the <character> used decides the country. The codes whose characters are changed is shown at the end of this appendix. When this method is chosen to change the character set, the keyboard layout does not change. It is also possible to obtain Italian and Spanish characters by this method. It is not possible for the keyboard layout to be changed to that either of these countries using DIP switch 4.

### E-2 Other information in the ASCII code table

The table in section E-4 contains the complete ASCII codes. Some of these are control codes which are used in controlling other devices, or the console. They have names which reflect their historical use. Some other manuals may refer to them under these names.

The table also contains information on how to obtain some of the non-printable characters and also the graphics characters. THIS TABLE HAS BEEN MADE FOR THE ASCII KEYBOARD AND MAY NOT BE COMPLETELY CORRECT IF THE LEGENDS ON THE KEYS ARE SET FOR ANOTHER LAYOUT. Consult Appendix D to see the physical position of the graphics characters on the keyboard. The graphics keys do not change if the layout changes for alphanumeric keys.

### E-3 ASCII codes with printers and other communication

In communicating with another computer or a printer, ASCII codes are sent. The characters displayed depend on the interpretation of the two computers. For example suppose the TERM program is being used with two PX-8 computers one of which is set to display standard ASCII and the other the English character set. If the ASCII code 35 is sent from the first (standard ASCII) to the second computer (English), the character “#” would be displayed on the screen of the first computer but a “£” character on the screen of the second.

Similarly a printer has a means of deciding which character set should be printed as the ASCII code can be interpreted in different ways. If the character sets are different, the output on the printer will differ from that displayed on the screen.

### E-4 Graphics and user defined characters

In addition to the standard ASCII character set, the PX-8 has a number of graphics characters. It is also possible to define some codes to correspond to characters which can be designed by the user. An explanation of these User-Defined characters is given in Appendix H of the BASIC Reference Manual.

In order to obtain a printout using these characters, the printer has to contain the characters in its character set. Some EPSON printers have the PX-8 graphics characters included in their character set, or can be modified by an EPSON dealer to include them. It is also possible for some these characters and the screen dump must be carried out from screen mode 3, so that a bit-image of the characters are obtained.



HEX	DECIMAL	CHAR	KEYSTROKES(ASCII) to obtain the character
00	000	NUL	CTRL-@
01	001	SOH	CTRL-A
02	002	STX	CTRL-B
03	003	ETX	CTRL-C: STOP
04	004	EOT	CTRL-D
05	005	ENQ	CTRL-E
06	006	ACK	CTRL-F
07	007	BEL	CTRL-G
08	008	BS	CTRL-H; BS
09	009	HT	CTRL-I; TAB
0A	010	LF	CTRL-J
0B	011	VT	CTRL-K; HOME
0C	012	FF	CTRL-L; CLR
0D	013	CR	CTRL-M; RETURN
0E	014	SO	CTRL-N
0F	015	SI	CTRL-O
10	016	DLE	CTRL-P
11	017	DC1	CTRL-Q
12	018	DC2	CTRL-R; INS
13	019	DC3	CTRL-S; PAUSE
14	020	DC4	CTRL-T
15	021	NAK	CTRL-U
16	022	SYN	CTRL-V
17	023	ETB	CTRL-W
18	024	CAN	CTRL-X
19	025	EM	CTRL-Y
1A	026	SUB	CTRL-Z
1B	027	ESC	CTRL-[; ESC
1C	028	FS	CTRL-\; →
1D	029	GS	CTRL-]; ←
1E	030	RS	CTRL-^; ↑
1F	031	US	CTRL-_; ↓
20	032	SPACE	
21	033	!	
22	034	''	
23	035	#	
24	036	\$	
25	037	%	
26	038	&	
27	039	.	
28	040	(	
29	041	)	
2A	042	*	
2B	043	+	
2C	044	,	
2D	045	-	

HEX	DECIMAL	CHAR	KEYSTROKES(ASCII) to obtain the character
2E	046	.	
2F	047	/	
30	048	0	NUM-M
31	049	1	NUM-J
32	050	2	NUM-K
33	051	3	NUM-L
34	052	4	NUM-U
35	053	5	NUM-I
36	054	6	NUM-O
37	055	7	
38	056	8	
39	057	9	
3A	058	:	
3B	059	;	
3C	060	<	
3D	061	=	
3E	062	>	
3F	063	?	
40	064	@	
41	065	A	
42	066	B	
43	067	C	
44	068	D	
45	069	E	
46	070	F	
47	071	G	
48	072	H	
49	073	I	
4A	074	J	
4B	075	K	
4C	076	L	
4D	077	M	
4E	078	N	
4F	079	O	
50	080	P	
51	081	Q	
52	082	R	
53	083	S	
54	084	T	
55	085	U	
56	086	V	
57	087	W	
58	088	X	
59	089	Y	
5A	090	Z	
5B	091	[	

HEX	DECIMAL	CHAR	KEYSTROKES(ASCII) to obtain the character
5C	092	\	
5D	093	]	
5E	094	^	
5F	095	~	
60	096	`	
61	097	a	
62	098	b	
63	099	c	
64	100	d	
65	101	e	
66	102	f	
67	103	g	
68	104	h	
69	105	i	
6A	106	j	
6B	107	k	
6C	108	l	
6D	109	m	
6E	110	n	
6F	111	o	
70	112	p	
71	113	q	
72	114	r	
73	115	s	
74	116	t	
75	117	u	
76	118	v	
77	119	w	
78	120	x	
79	121	y	
7A	122	z	
7B	123	{	
7C	124		
7D	125	}	
7E	126	~	
7F	127	DEL Δ	
80	128	+	GRPH-S
81	129	±	GRPH-X
82	130	⊞	GRPH-W
83	131	⊠	GRPH-D
84	132	⊡	GRPH-A
85	133	⊣	GRPH-T
86	134	⊤	GRPH R
87	135	⊥	GRPH-Q
88	136	⊦	GRPH-E
89	137	⊧	GRPH-Z

HEX	DECIMAL	CHAR	KEYSTROKES(ASCII) to obtain the character
8A	138	⊨	GRPH-C
8B	139	⊩	GRPH-J
8C	140	⊪	GRPH-F
8D	141	⊫	GRPH-G
8E	142	⊬	GRPH-H
8F	143	⊭	GRPH-Y
90	144	⊮	GRPH-U
91	145	⊯	GRPH-I
92	146	⊰	GRPH-O
93	147	⊱	GRPH-P
94	148	⊲	GRPH-@
95	149	⊳	GRPH-K
96	150	⊴	GRPH-V
97	151	⊵	GRPH-, (comma)
98	152	⊶	GRPH-M
99	153	⊷	GRPH-N
9A	154	⊸	GRPH-B
9B	155	⊹	GRPH-; (semi-colon)
9C	156	⊺	GRPH- (full stop)
9D	157	⊻	GRPH-: (colon)
9E	158	⊼	GRPH-/
9F	159	⊽	GRPH-L
A0	160		
A1	161		
A2	162		
A3	163		
A4	164		
A5	165		
A6	166		
A7	167		
A8	168		
A9	169		
AA	170		
AB	171		
AC	172		
AD	173		
AE	174		
AF	175		
B0	176		
B1	177		
B2	178		
B3	179		
B4	180		
B5	181		
B6	182		
B7	183		

HEX	DECIMAL	CHAR	KEYSTROKES(ASCII) to obtain the character
B8	184		
B9	185		
BA	186		
BB	187		
BC	188		
BD	189		
BE	190		
BF	191		
C0	192		
C1	193		
C2	194		
C3	195		
C4	196		
C5	197		
C6	198		
C7	199		
C8	200		
C9	201		
CA	202		
CB	203		
CC	204		
CD	205		
CE	206		
CF	207		
D0	208		
D1	209		
D2	210		
D3	211		
D4	212		
D5	213		
D6	214		
D7	215		
D8	216		
D9	217		
DA	218		
DB	219		
DC	220		
DD	221		
DE	222		
DF	223		
E0	224		GRPH-0
E1	225		GRPH-1
E2	226		GRPH-2
E3	227		GRPH-3
E4	228		GRPH-4
E5	229		GRPH-5

Δ↑  
 User  
 defined  
 characters  
 ↓

HEX	DECIMAL	CHAR	KEYSTROKES(ASCII) to obtain the character
E6	230		GRPH-6
E7	231		GRPH-7
E8	232		GRPH-8
E9	233		GRPH-9
EA	234		GRPH--(hyphen)
EB	235		GRPH-^
EC	236		GRPH-[
ED	237		GRPH-\
EE	238		GRPH-]
EF	239		GRPH + SHIFT -]
FO	240		CTRL-0
F1	241		CTRL-1
F2	242		CTRL-2
F3	243		CTRL-3
F4	244		CTRL-4
F5	245		CTRL-5
F6	246		CTRL-6
F7	247		CTRL-7
F8	248		CTRL-8
F9	249		CTRL-9
FA	250		CTRL--(hyphen)
FB	251		CTRL-;(semicolon)
FC	252		CTRL-:(colon)
FD	253		CTRL-,(comma)
FE	254		CTRL-.(full stop)
FF	255		

User  
 defined  
 characters  
 ↓

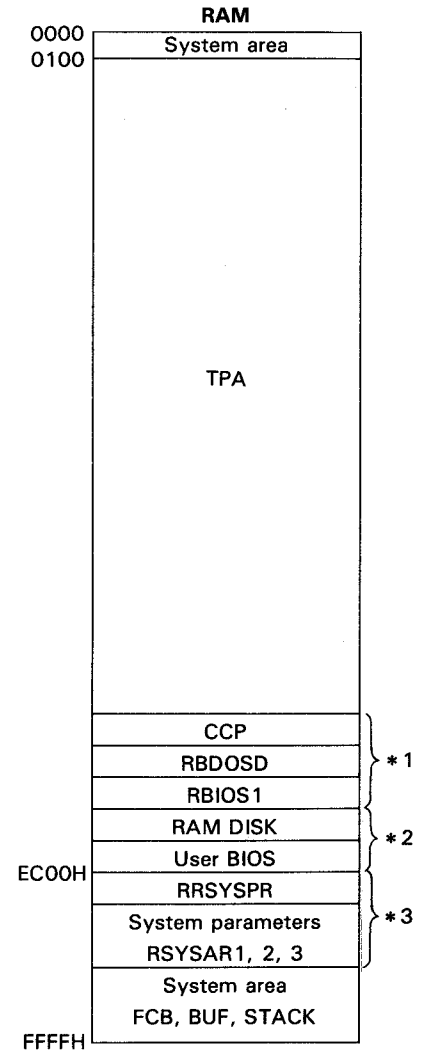
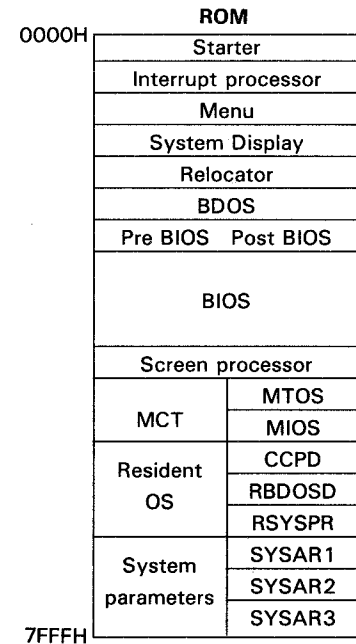
## E-5 INTERNATIONAL CHARACTER SETS

Differences between the USASCII character set and the character sets of other countries are as shown below.

Country Dec. Code	United States	France	Germany	England	Denmark	Sweden	Italy	Spain	Norway
35	#	#	#	£	#	#	#	℞	#
36	\$	\$	\$	£	\$	¤	\$	\$	¤
64	@	@	@	@	£	£	@	@	£
91	[	°	ä	[	℞	ä	°	i	℞
92	\	µ	ö	/	ø	ö	/	z	ø
93	]	®	ü	]	ä	ä	®	ç	ä
94	^	>	>	^	ü	ü	>	>	ü
96	·	·	·	·	®	®	ü	·	®
123	ç	®	®	ç	®	®	®	·	®
124		ü	ö		ø	ö	ö	z	ø
125	>	®	ü	>	®	®	®	ç	®
126	~	·	ß	~	ü	ü	·	ç	ü

## Appendix F

## MEMORY MAPS



\*1: This area is relocated from ROM by the relocater.

Its starting address in RAM is determined by the size of the RAM disk and User BIOS areas.

\*2: The size of the RAM disk and User BIOS areas can be changed with the CONFIG command.

\*3: This area is relocated from ROM by the relocater. Its starting address is predetermined.

\*4: CCP starting address should be above 8000H.

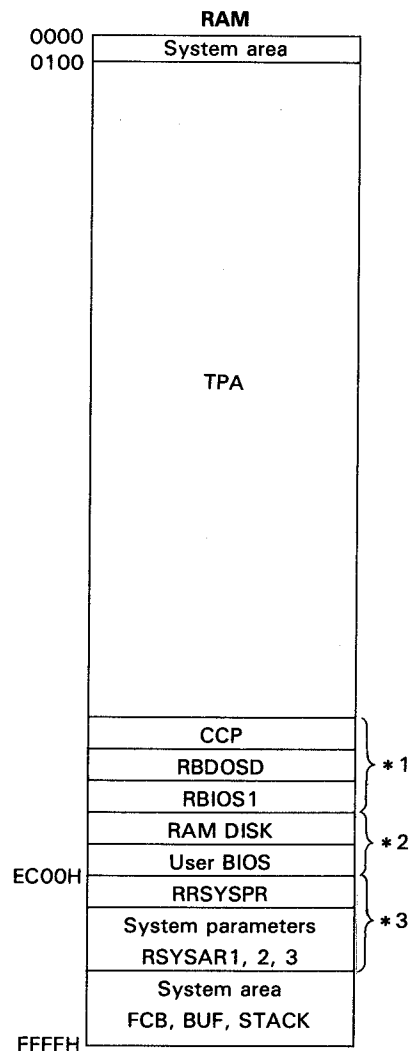
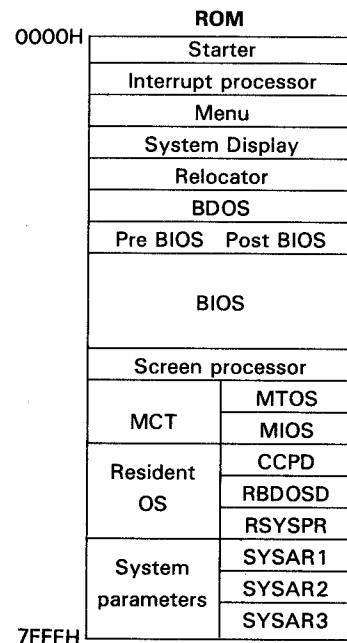
## E-5 INTERNATIONAL CHARACTER SETS

Differences between the USASCII character set and the character sets of other countries are as shown below.

Country Dec. Code	United States	France	Germany	England	Denmark	Sweden	Italy	Spain	Norway
35	#	#	#	£	#	#	#	¤	#
36	#	#	#	#	#	¤	#	#	¤
64	©	©	©	©	©	©	©	©	©
91	£	¤	¤	£	¤	¤	¤	£	¤
92	/	√	ö	/	ö	ö	/	ö	ö
93	J	ö	ü	J	ö	ö	ö	ö	ö
94	>	>	>	>	ü	ü	>	>	ü
96	•	•	•	•	ö	ö	ü	•	ö
123	€	ö	ö	€	ö	ö	ö	•	ö
124	!	ü	ö	!	ö	ö	ö	ö	ö
125	ö	ö	ü	ö	ö	ö	ö	ö	ö
126	ö	•	ö	ö	ü	ü	ö	ö	ü

## Appendix F

### MEMORY MAPS



- \*1: This area is relocated from ROM by the relocater. Its starting address in RAM is determined by the size of the RAM disk and User BIOS areas.
- \*2: The size of the RAM disk and User BIOS areas can be changed with the CONFIG command.
- \*3: This area is relocated from ROM by the relocater. Its starting address is predetermined.
- \*4: CCP starting address should be above 8000H.

# Appendix G

## HARDWARE SPECIFICATIONS

### CPUs

Main CPU : Z-80 compatible C-MOS CPU  
 Slave CPU : 6301  
 Sub-CPU :  $\mu$ PD7508

### Main memory

RAM : 64KB  
 ROM : 32KB

### Memory for slave CPU

RAM : 6KB (external) for Video RAM  
 128 bytes (internal)

ROM : 4KB (internal)

Display : LCD display (480 × 64 dots)

Speaker : Dynamic speaker

Interfaces : RS-232C, serial, bar code reader, analog input

Keyboard : ASCII, France, Germany, etc.

Power supply : NiCd battery pack  
 AC adaptor

Dimensions : 297(W) × 216(D) × 48(H) mm

Weight : Approx. 2.3 kg

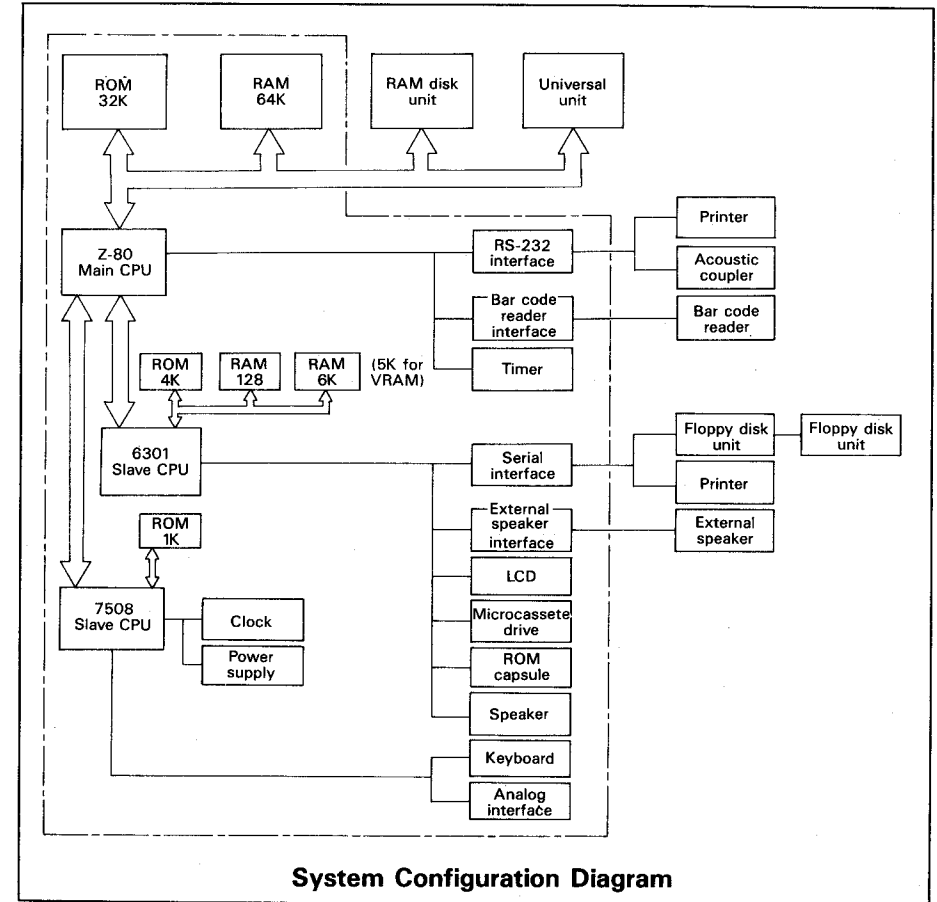
### Environmental conditions

Temperature : 5 to 35°C (operation)  
 -20 to 60°C (storage)

Humidity : 10 to 80% (operation, no condensation)  
 10 to 80% (storage, no condensation)

Resistance to shock : Max. 1 G, 1 msec (operation)

Resistance to vibration : Max. 0.25 G, 55 Hz (operation)

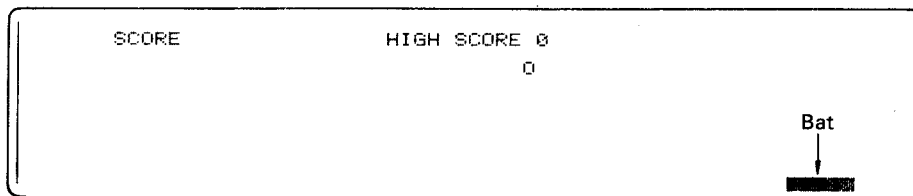


# Appendix H

## SOME EXAMPLE PROGRAMS

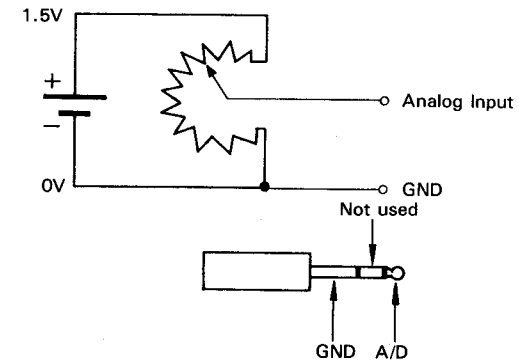
Two examples of programs are given in this appendix. They show different uses of function calls. One covers the use of the A/D Converter, which can only be used with a BIOS call. This is then linked to BASIC to show how the two can be combined. The second shows how the screen data can be saved to disk, and so covers examples of sending and receiving data to the 6301 slave processor. It also covers disk handling and a certain amount of error checking. Two programs are given, to save and read the screen data. Again they are linked to BASIC, but as they are longer programs this needs to be done in a different way. They could be poked in, but the code has to be written to locate itself into the correct position before BASIC is loaded.

### H-1. A simple game illustrating use of the A/D converter.



The following program shows the use of the ADCVRT BIOS routine to enable the voltage across a variable resistor to be read. By rotating the shaft of the resistor the voltage read by the A/D converter will vary in proportion to the amount of rotation. This can then be used as a means of altering the position of a "bat" on the screen to play a simple game. This game shows how the BIOS routine can be used and how it can be linked to a BASIC program through the CALL statement of BASIC.

The circuit for the game is as follows:



The circuit comprises a battery to provide the voltage, a variable resistor and the necessary connecting wires. The connections to the A/D interface are shown in section 4. 6.

The program listing is as follows:

```

10 * A/D CONVERTER IN A GAME *
20 *
30 CLEAR ,%HC000: GOSUB 4000: GOTO 300
40 *SUBROUTINES
50 LOCATE X,Y : PRINT "0"; : X1 = X : Y1 = Y : RETURN
60 LOCATE X1,Y1 : PRINT " "; : RETURN
70 X = X + DX
80 IF X<2 THEN X = 4-X : DX = -DX
90 IF X>79 THEN X = 158-X : DX = -DX
100 RETURN
110 *
200 CALL ADC(CHX,ADD%): ADD% = ADD% * 74/BMAX + 2 : IF ADD%>75 THEN ADD% =
  75
210 PX = ADD%
220 LOCATE PX,8 : PRINT "      "; CHR%(&H1E)
230 LOCATE PX,8 : PRINT STRING$(5,140); CHR%(&H1E) : PX1 = PX : RETURN
240 *
250 *
290 *MAIN PROGRAM
300 *
310 SCREEN 3,,0 : CLS : CHX = 0 : PX = 2 : PX1 = 2 : X = 2 : X1 = 2 : SCORE
  = 0 : N = 0 : BALL = 1
320 LINE (5,0) - (5,63) : LINE (475,0) - (475,63)
330 LOCATE 10,1 : PRINT "SCORE" : LOCATE 30,1 : PRINT "BALL NUMBER": BALL
340 X = INT(RND*79) + 2 : DX = FIX ((RND*2-1)*(N/5+1))
360 Y = 2 : GOSUB 200 : GOSUB 50
370 FOR Y = 3 TO 7
380 GOSUB 70 : GOSUB 200 : GOSUB 60 : GOSUB 50 : GOSUB 200
390 NEXT Y
400 Y = 8 : GOSUB 70 : GOSUB 200
410 IF X = PX OR X = PX+4 THEN SOUND 1200,20 : DSCORE = 1 : GOTO 800
420 IF X = PX+1 OR X = PX+3 THEN SOUND 1800,20 : DSCORE = 2 : GOTO 800
430 IF X = PX+2 THEN SOUND 2400,20 : DSCORE = 3 : GOTO 800
440 DSCORE = 0: GOSUB 60 : GOSUB 50 : SOUND 600,50
800 SCORE = SCORE + DSCORE : LOCATE 10,1 : PRINT SCORE:
810 BALL = BALL + 1 : IF BALL > 20 THEN 1000
820 GOSUB 60
830 N = N+1 : GOTO 330

```

```

1000 CLS :LOCATE 20,1 : PRINT "HIGH SCORE"; HSCORE
1010 LOCATE 20,3 : PRINT "Your score was"; SCORE
1020 LOCATE 20,5 : PRINT "Play again(y/n)"; Y$=INPUT$(1)
1030 IF SCORE > HSCORE THEN HSCORE = SCORE
1040 IF Y$ <> "Y" AND Y$ <> "y" THEN CLS : END ELSE 310
1050 ?
1060 ?
4000 ? SUBROUTINE FOR LOADING MACHINE CODE
4010 CLS
4020 AD=&HC000
4030 READ DAT$
4040 IF DAT$="end" THEN 4240
4050 IF DAT$ = "LB" THEN POKE AD, PEEK(1) + &H6F : AD= AD + 1 : GOTO 4030
4060 IF DAT$ = "HB" THEN POKE AD, PEEK(2) : AD= AD + 1 : GOTO 4030
4070 POKE AD,VAL("&h"+DAT$)
4080 AD=AD+1:GOTO 4030
4090 DATA D5                : *PUSH DE      PUSH ADD% POINTER
4100 DATA 4E                : *LD C, (HL)  SET ADC CHANNEL
4110 DATA CD, LB, HB        : *CALL ADCVRT CALL A/D CONVERTER
4120 DATA D1                : *POP DE      POP ADD% POINTER
4130 DATA E6, FC           : *AND 0FCH   MASK BITS 0 AND 1
4140 DATA 1F               : *RRA       ROTATE RIGHT
4150 DATA 1F               : *RRA       ROTATE RIGHT
4160 DATA 12               : *LD (DE),A  ADD% = DATA RETURNED
4170 DATA 13               : *INC DE    ADD% MSB
4180 DATA AF               : *XOR A     ZERO A REGISTER
4190 DATA 12               : *LD (DE),A  ADD% MSB = 0
4200 DATA C9               : *RET      RETURN TO BASIC
4210 DATA end              : *end
4220 ?
4230 ?
4240 ADC=&HC000 : *ADDRESS TO CALL MACHINE CODE SUBROUTINE
4250 ?
4260 ? DETERMINE MAX RESPONSE TO BATTERY
4270 ?
4280 LOCATE 15,2 : PRINT "TURN RESISTOR TO GIVE MAXIMUM VALUE"
4290 LOCATE 15,3 : PRINT "THEN PRESS SPACE BAR"
4300 LOCATE 20,5 : PRINT "RESPONSE = "
4310 CALL ADC(CH%,ADD%) : LOCATE 40,5,0 : PRINT ADD%
4320 IF INKEY$ = " " THEN BMAX = ADD% : RETURN ELSE 4300

```

The main part of the program consists of a series of subroutines to move the "ball" on the screen, and move the "bat" in response to changes in the position of the resistor. These subroutines are at the beginning of the program (lines 50 to 230).

Line 200 reads the value of the voltage from the A/D converter.

Lines 310 to 1040 form the main part of the program which handles the ball.

Lines 4000 to 4190 load the machine code routine.

Lines 4260 to 4300 use the machine code routine to read the A/D converter. They also allow setting of the maximum value of the voltage returned to be used in line 200, so that the maximum sensitivity of movement can be obtained for the battery being used.

Since the object of this appendix is to show the use of the BIOS call, only the machine code routine and the BASIC CALL will be described in detail.

The BASIC CALL statement is used with a list of parameters. This allows the channel of the A/D converter to be varied as well as returning a value of the voltage into a named variable. The channel is passed to the machine code routine by means of the variable CH%, and the voltage is read back into the variable ADD%. Further details of using the CALL statement are given in the BASIC Reference Manual (Appendix D). When two parameters are passed to the machine language subroutine, the location in memory of the first variable (in this case the channel CH%) will be placed in register pair HL. The address of the second variable (in this case ADD%) will be placed in register pair DE.

The machine code subroutine first stores the address of the variable ADD% on the stack (data in line 4090). It then loads the channel number into register C (data in line 4100). The ADCVRT routine is then called (data in line 4110). In order to specify the address of ADCVRT, BASIC lines 4050 and 4060 are used. The low byte is determined by taking the low byte of WBOOT from location 0001 in the main memory bank, and adding the offset for the ADCVRT routine (6FH). The high byte is then obtained as the high byte of WBOOT obtained from location 0002 in the main memory bank. On return from the ADCVRT routine, register A contains the value of the voltage, in the top six bytes. After retrieving the location of the variable ADD% from the stack (data in line 4120), the two lowest bits of the returned voltage are made zero by an AND operation with the value FCH, so that any values in the lowest two bytes do not affect the result. (Data in line 4130.) If no further change was made the byte would give number in the range 4 to 255, since the least significant bits are always zero. Thus two rotations are made (data in lines 4140 and 4150) to make the most significant bits always zero. The contents of register A are then placed in the first byte of the variable ADD% by loading register A into the address pointed to by register pair DE (data in line 4160). By incrementing the value of register pair DE, and making register A zero performing an XOR operation on register A with itself, the next byte of the variable ADD% is made zero. This ensures the variable ADD% is always a one byte variable. The data for these operations are to be found in lines 4170 to 4190. Finally the routine returns to BASIC.

The address for the machine code subroutine is assigned to the variable ADC in line 4240. The BASIC CALL statement can only call to a variable.



As an illustration of the use of the routine, and because the battery could vary in voltage, lines 4280 to 4320 will show the variation of the voltage as the resistor spindle is turned, from zero up to the maximum value. A value of 63 will denote 2.0 volts. Since the battery voltage in the diagram is 1.5 volts, the maximum value obtained will be in either 46 or 47.

## H-2. Saving and loading the graphics screen from disk.

The following two listings show heavily commented programs written with the MACRO-80 assembler in Z-80 code. The comments should be sufficient for an assembler programmer to follow, although they may require the OS Reference Manual or the Technical Reference Manual for further details of the use of the slave processor.

For those users wishing to enter the programs, use either the MACRO-80 assembler, or the dumps at the end of the listings together with the pertinent instructions. A BASIC listing is given as an example to show how such programs can be used with BASIC.

```

MACRO-80 3.44 09-Dec-81 PAGE 1

;*****
;*
;* Program Save the graphic screen to the currently
;* logged in drive to file 'VRAMDATA.DAT'.
;*
;* System PX-8
;* Configuration 9 Kb RAM disk & 0 page USERBIOS size
;* Language Zilog Z80 mnemonic code
;* Ref DCE/YM
;* Date JAN-84
;*
;*****

.Z80
;*****
;*
;* Standard CP/M & PX-8 equates.
;*
;*****
0001 BIOS EQU 0001H ; Pointer to WBOOT address
0005 BDOS EQU 0005H ; BDOS function dispatch jump vector
F2C9 SCRMODE EQU 0F2C9H ; Pointer to the address that holds the
; current screen mode.
F358 SLVFLG EQU 0F358H ; Slave communications enable flag.
;*****
;*
;* Relocator to move the main program up to 0B400H
;*
;*****
0000' 21 000E LD HL,SOURCE ; Set up the origin address
0003' 11 0400 LD DE,DEST ; Set Destination address
0006' 01 014E LD BC,LEN ; Set up the length of the program
0009' ED 80 LDIR ; And move the program.
000B' C3 0000 JP 0000H ; Exit back to CP/M

```

```

;*****
;*
;* Main program starts here !!
;*
;*****
SOURCE:
.PHASE DB400H

B400 DEST:
B400 21 F2C9 LD HL,SCRMODE ; Get the pointer to the screen mode
B403 7E LD A,(HL) ; and find out the screen mode
B404 FE 03 CP 3 ; Is the screen in graphic mode?
B406 C2 B436 JP NZ,EXIT ; No, so exit from this program
;
; Since screen mode is graphic save the screen to disk
;
B409 CD B44F CALL CREATE ; Create a new file into which to
; save the screen.
B40C FE FF CP OFFH ; Is the disk directory full?
B40E CA B436 JP Z,EXIT ; Yes, so exit from this program
;
; The file is now open so the data can be saved
;
MACRO-80 3.44 09-Dec-81 PAGE 1-1

B411 3E FF LD A,OFFH ; Enable slave communications so that
B413 32 F358 LD (SLVFLG),A ; commands & data can be sent to the
; slave CPU. If this flag is not set
; then all slave communication is ignored.
;
; The VRAM can now be read and written to disk
;
B416 3E 00 LD A,00H ; Initialize the count for the number of
; blocks of data we have to read from the
; VRAM in order to save the complete screen

B418 LOOP:
B418 32 B54D LD (COUNT),A
B41B CD B437 CALL READVRAM ; Get the next block of data from the VRAM
B41E CD B478 CALL WRITE ; Write the data we've just read to the file
B421 FE FF CP OFFH ; Is the disk full?
B423 CA B48A JP Z,ERASE ; Yes, so erase the file so that only part
; of the screen is saved
B426 3A B54D LD A,(COUNT) ; Increment the block counter
B429 3C INC A
B42A FE 20 CP 32 ; Have we written all of the screen?
B42C 20 EA JR NZ,LOOP ; No, so go back and process the next block
;
; All of the screen has been written to disk, so
; close the file.
;
B42E 3E 00 LD A,00H ; Disable communication with the slave CPU
B430 32 F358 LD (SLVFLG),A ; so that commands & data are ignored
B433 CD B481 CALL CLOSE ; Close the file we're saving to
;
; It is now safe to exit.
;
B436 EXIT:
B436 C9 RET ; We've done it!!

```

```

;*****
;*
;* Subroutine Read the VRAM data in a block of 60 x 2 bytes *
;* this actually reads 2 lines of 480 pixels. *
;*
;*****
READVRAM:
B437 LD A,(COUNT) ; Get the block number of the data to be
B437 3A B54D ; read in, in order to calculate the
; X,Y position to start on the screen
; which we will be reading.
B43A CB 07 RLC A ; Calculate the Y coordinate of the screen
B43C 32 B4A5 LD (YCOORD),A ; block (= data block number *2)
B43F 3E 00 LD A,00H ; Set the X coordinate to 0 i.e. always
B441 32 B4A4 LD (XCOORD),A ; start at the left edge of the screen
B444 11 B498 LD DE,PACKET ; Set up the packet address for the BIOS
; slave call
B447 2A 0001 LD HL,(BIOS) ; Get the BIOS call address i.e. WBOOT
; address so we can calculate the address
; of the SLAVE BIOS call.
;
MACRO-80 3.44 09-Dec-81 PAGE 1-2

B44A 3E 72 LD A,072H ; Get the SLAVE BIOS call offset from WBOOT
B44C 85 ADD A,L ; And calculate the SLAVE call's actual
; address
B44D 6F LD L,A ; Move the calculated address into the correct
; to execute the call
B44E E9 JP (HL) ; Execute the SLAVE BIOS call & return from
; this subroutine.
;*****
;*
;* Subroutine Create a file (called VRAMDATA.DAT) into *
;* which to save the screen data on disk. *
;* It takes up 3.75Kb. *
;*
;*****
CREATE:
B44F LD A,00 ; Zero the first byte after the file type
B44F 3E 00 ; in the FCB for safety's sake
B451 32 B534 LD (FCBADR+12),A
B454 21 B534 LD HL,FCBADR+12 ; Set up the source location to zero the
; the rest of the FCB
B457 11 B535 LD DE,FCBADR+13 ; Set up the destination address to zero
; the rest of the FCB
B45A 01 0019 LD BC,25 ; Set up the number of bytes we need to zero
B45D ED 80 LDIR ; And set them to zero using a ripple effect
B45F 0E 0D LD C,13 ; Set up the BDOS function code that performs
B461 CD 0005 CALL BDOS ; the 'RESET' of the disks. i.e. sets all of
; the disks to a R/W status and sets the DMA
; default address to 0080H & execute it.
B464 0E 1A LD C,26 ; Set up the BDOS function code that sets the
; DMA address.
B466 11 B4AB LD DE,DMABUF ; Set up the DMA buffer address
B469 CD 0005 CALL BDOS ; And go and set the DMA address
B46C CD B492 CALL DELETE ; Erase the file if it exists, but don't
; worry if it doesn't!!
B46F 0E 16 LD C,22 ; Set up the BDOS function code that will
; create a new file.
B471 11 B528 LD DE,FCBADR ; Set up the FCB address
B474 CD 0005 CALL BDOS ; Go and try to create the new file.
B477 C9 RET ; And return to the main program which will
; deal with any errors that may occur.

```

```

*****
;*
;* Subroutine Write the data block which has been read from the
;* screen to the file called VRAMDATA.DAT.
;*
*****
B478 WRITE:
B478 0E 15 LD C,21 ; Set up the BDOS function code that will
; write the data in the current DMA buffer
; to the next record of the currently open
MACRO-80 3.44 09-Dec-81 PAGE 1-3
; file, in this case to 'VRAMDATA.DAT'
B47A 11 B528 LD DE,FCBADR ; Set up the FCB address
B47D CD 0005 CALL BDOS ; Go and write the data
B480 C9 RET ; And return to the main program
*****
;*
;* Subroutine Close the file 'VRAMDATA.DAT', now the screen
;* contents have been written to it.
;*
*****
B481 CLOSE:
B481 0E 10 LD C,16 ; Set the BDOS function code that closes
; a file.
B483 11 B528 LD DE,FCBADR ; Set up the FCB address
B486 CD 0005 CALL BDOS ; Go and close the file 'VRAMDATA.DAT'
B489 C9 RET ; And return to the main program.
*****
;*
;* Subroutine Erase the file we've written the screen
;* contents to because it is more than likely
;* that it is incomplete and therefore totally
;* useless to anybody.
;*
*****
B48A ERASE:
B48A 3E 00 LD A,00H ; Disable slave communication now because
B48C 32 F358 LD (SLVFLG),A ; we no longer need it.
B48F CD B481 CALL CLOSE ; Close the file we've created
; We needn't bother about the error that
; could be returned because we know it
; can't happen.
B492 DELETE:
B492 0E 13 LD C,19 ; Set up the BDOS function code that will
; delete the file (if it exists)
B494 11 B528 LD DE,FCBADR ; Set up the FCB address
B497 CD 0005 CALL BDOS ; Execute function and ignore any error code.
; It would indicate that the file didn't
; exist which is OK because
; a new one must be created.
B49A C9 RET

```

```

*****
;*
;* Data storage for all variables needed by the program
;*
*****
PACKET: ; Slave CPU's communications packet
B49B ; Address of the packet that will be sent
B49B B4A3 DW SNDPKT ; to the slave CPU
MACRO-80 3.44 09-Dec-81 PAGE 1-4
B49D 0004 DW 4 ; Size of the packet that will be sent
; to the slave CPU
B49F B4A7 DW RCVPKT ; Address of the packet that is expected
; back from the slave CPU
B4A1 0079 DW 121 ; Size of the packet expected to be
; returned from the slave CPU
B4A3 SNDPKT: ; Actual packet to be sent to the slave
; CPU. It includes both the command
; and any parameters that are required
B4A3 24 DB 024H ; Slave command code that reads the
; graphic screens contents.
B4A4 00 XCOORD: DB 00H ; Starting X coordinate from which
; to read the data
B4A5 00 YCOORD: DB 00H ; Starting Y coordinate from which to
; to read the data
B4A6 78 DB 120 ; Number of data bytes to read from the
; X,Y coordinate specified above.
B4A7 RCVPKT: ; Packet returned by the slave CPU
; once the command has been executed.
B4A7 00 DB 00H ; Return code that indicates the success
; of the execution of the command
B4A8 DNABUF: DS 128 ; Space allocated for the receipt of the
; screen data and for the DMA buffer.
B528 FCBADR: ; File control block (FCB) work area
B528 00 DB 00H ; Drive code in this case the currently
; logged in disk
B529 56 52 41 40 DB 'VRAMDATA' ; The file name to be used
B52D 44 41 54 41 ;
B531 44 41 54 DB 'DAT' ; The file extension or file type
B534 DS 25 ; The rest of the FCB is used by the system
; and we don't have to worry about it at all
B54D 00 COUNT: DB 00H ; Count used for number of blocks to
; read from the screens VRAM
014E LEN EQU $-DEST
END
MACRO-80 3.44 09-Dec-81 PAGE 5

```

Macros:

Symbols:

B005	BDOS	B001	BIDS	B481	CLOSE
B54D	COUNT	B44F	CREATE	B492	DELETE
B400	DEST	B4A8	DNABUF	B48A	ERASE
B436	EXIT	B528	FCBADR	014E	LEN
B418	LDDP	B49B	PACKET	B4A7	RCVPKT
B437	READVRAM	F2C9	SCRMODE	F358	SLVFLG
B4A3	SNDPKT	000E	SOURCE	B478	WRITE
B4A4	XCOORD	B4A5	YCOORD		

No Fatal errors!

```

;*****
;*
;* Program Load the graphic screen from the currently
;* logged in drive from file 'VRAMDATA.DAT'.
;* System PX-B
;* Configuration 9 Kb RAM disk & 0 page USERBIOS size
;* Language Zilog Z80 mnemonic code
;* Ref DCE/YM
;* Date JAN-84
;*
;*****

.Z80
;*****
;* Standard CP/M & PX-B equates.
;*
;*****
0001 BIOS EQU 0001H ; Pointer to WBOOT address
0005 BDOS EQU 0005H ; BDOS function dispatch jump vector
F2C9 SCRMOBE EQU 0F2C9H ; Pointer to the address that holds the
; current screen mode.
F358 SLVFLG EQU 0F358H ; Slave communications enable flag.
;*****
;* Relocator to move the main program up to 0B200H
;*
;*****
0000' 21 000E' LD HL,SOURCE ; Set up the origin address
0003' 11 B200 LD DE,DEST ; Set Destination address
0006' 01 013C LD BC,LEN ; Set up the length of the program
0009' ED B0 LDIR ; And move the program.
000B' C3 0000 JP 0000H ; Exit back to CP/M
;*****
;* Main program starts here !!
;*
;*****
SOURCE:
.PHASE 0B200H
DEST:
B200 LD HL,SCRMOBE ; Get the pointer to the screen mode
B200 21 F2C9 LD A,(HL) ; and find out the screen mode
B203 7E CP 3 ; Is the screen in graphic mode?
B204 FE 03 JP NZ,EXIT ; No, so exit from this program
B206 C2 B236
;
; Since screen mode is graphic, load the screen from disk
;
B209 CD B24F CALL OPEN ; Open the file from which to
; load the screen.
B20C FE FF CP OFFH ; Does the file exist?
B20E CA B236 JP Z,EXIT ; No, so exit from this program
;
; The file is now open so the data can be loaded
;

```

```

B211 3E FF LD A,OFFH ; Enable slave communications so that
B213 32 F358 LD (SLVFLG),A ; commands & data can be sent to the
; slave CPU. If this flag is not set
; then all slave communication is ignored.
;
; Data can now be read from disk and written to VRAM
;
B216 3E 00 LD A,00H ; Initialize the count for the number of
; blocks of data to be read from the
; disk in order to load the complete screen

B218 LOOP: LD (CBUNT),A
B218 32 B33B CALL READ ; Get the next block of data from the disk
B21B CB B275 CP OFFH ; Have we tried to get data that doesn't exist
B21E FE FF JP Z,EOF ; Yes, so just exit because there is nothing
; else we can do !
B223 CB B237 CALL SETVRAM ; We've got the next block so write it to VRAM
B226 3A B33B LD A,(CBUNT) ; Increment the block counter
B229 3C INC A
B22A FE 20 CP 32 ; Has all the screen been loaded from disk?
B22C 20 EA JR NZ,LOOP ; No, so go back and process the next block
;
; All of the screen has been loaded from disk, so
; close the file.
;
B22E EOF: LD A,00H ; Disable communication with the slave CPU
B22E 3E 00 LD (SLVFLG),A ; so that commands & data are ignored
B230 32 F358 CALL CLOSE ; Close the file.
B233 CB B27E
;
; It is now safe to exit.
;
EXIT: RET ; We've done it!!

;*****
;* Subroutine Set the VRAM data in a block of 60 x 2 bytes
;* this actually sets 2 lines of 480 pixels.
;*
;*****
SETVRAM:
LD A,(CBUNT) ; Get the block number of the data to be
; read in, in order to calculate the
; X,Y position to start on the screen
; which will have to be set
RLC A ; Calculate the Y coordinate of the screen
; block (= data block number *2)
LD (YCOORD),A
LD A,00H ; Set the X coordinate to 0 i.e. always
LD (XCOORD),A ; start at the left edge of the screen
LD DE,PACKET ; Set up the packet address for the BIOS
; slave call
B247 2A 0001 LD HL,(BIOS) ; Get the BIOS call address i.e. WBOOT
; address so that the SLAVE BIOS call

```

```

; address can be calculated
B24A 3E 72          LD    A,072H    ; Get the SLAVE BIOS call's offset from WROOT
B24C 85            ADD    A,L      ; And calculate the SLAVE calls' actual
; address
B24D 6F           LD    L,A      ; Move the calculated address
; to execute the call
B24E E9           JP    (HL)    ; Execute the SLAVE BIOS call & return from
; this subroutine.
    
```

```

;*****
;*                               *
;* Subroutine   Open the file VRAMDATA.DAT from which *
;*             to load the screen data from the disk. *
;*             It takes up 3.75Kb.                    *
;*                               *
;*****
    
```

```

B24F          OPEN:
B24F 3E 00          LD    A,00      ; Zero the first byte after the file type
B251 32 B322        LD    (FCBADR+12),A ; in the FCB for safety's sake
B254 21 B322        LD    HL,FCBADR+12  ; Set up the source location to zero the
; the rest of the FCB
B257 11 B323        LD    DE,FCBADR+13 ; Set up the destination address to zero
; the rest of the FCB
B25A 01 0019        LD    BC,25      ; Set up the number of bytes to be zero
B25D ED B0          LDIR      ; And set them to zero using a ripple effect
B25F 0E 0B          LD    C,13      ; Set up the BDOS function code that performs
B261 CD 0005        CALL   BDOS     ; the 'RESET' of the disks. i.e. sets all of
; the disks to a R/W status and sets the DMA
; default address to 0080H & execute it.
B264 0E 1A          LD    C,26      ; Set up the BDOS function code that sets the
; DMA address.
B266 11 B295        LD    DE,DMABUF  ; Set up the DMA buffer address
B269 CD 0005        CALL   BDOS     ; And go and set the DMA address
B26C 0E 0F          LD    C,15      ; Set up the BDOS function code that will
; open the file 'VRAMDATA.DAT'.
B26E 11 B316        LD    DE,FCBADR  ; Set up the FCB address
B271 CD 0005        CALL   BDOS     ; Go and try to open the file.
B274 C9           RET      ; And return to the main program which will
; deal with any errors that may occur.
    
```

```

;*****
;*                               *
;* Subroutine   Read the data block which is going to be written *
;*             to the screen from the file called VRAMDATA.DAT. *
;*                               *
;*****
    
```

```

B275          READ:
B275 0E 14          LD    C,20      ; Set up the BDOS function code that will
; read the data from the next record of
; the file into the DMA buffer.
B277 11 B316        LD    DE,FCBADR  ; Set up the FCB address
    
```

```

B27A CD 0005        CALL   BDOS     ; Go and read the data
B27D C9           RET      ; And return to the main program
    
```

```

;*****
;*                               *
;* Subroutine   Close the file VRAMDATA.DAT which *
;*             contains the screen data.            *
;*                               *
;*****
    
```

```

B27E          CLOSE:
B27E 0E 10          LD    C,16      ; Set the BDOS function code that closes
; a file.
B280 11 B316        LD    DE,FCBADR  ; Set up the FCB address
B283 CD 0005        CALL   BDOS     ; Go and close the file 'VRAMDATA.DAT'
B286 C9           RET      ; And return to the main program.
    
```

```

;*****
;*                               *
;* Data storage for all variables needed by the program *
;*                               *
;*****
    
```

```

B287          PACKET:
B287 B28F          DW    SNDPKT  ; Slave CPU's communications packet
; Address of the packet that will be sent
; to the slave CPU
B289 007E          DW    126    ; Size of the packet that will be sent
; to the slave CPU
B28B B315          DW    RCVPKT  ; Address of the packet expected
; back from the slave CPU
B28D 0001          DW    1      ; Size of the packet expected to
; be returned from the slave CPU
B28F          SNDPKT:
; Actual packet to be sent to the slave
; CPU. It includes both the command
; and any parameters that are required
; Slave command code that sets the
; graphic screens contents.
B28F 25           DB    025H
B290 00           XCOORD: DB    00H ; Starting X coordinate from which
; to read the data
B291 00           YCOORD: DB    00H ; Starting Y coordinate from which
; to read the data
B292 3C           DB    60      ; Number of bytes to be set
; in the X direction
B293 02           DB    2      ; Number of bytes to be set
; in the Y direction
B294 00           DB    0
B295          DMABUF: DS    12B ; Space allocated for the receipt of the
; screen data and for the DMA buffer.
B315          RCVPKT:
; Packet returned from the slave CPU
; once the command has been executed.
; Return code that indicates the success
; of the execution of the command
B315 00           DB    00H
B316          FCBADR:
; File control block (FCB) work area
; Drive code in this case the currently
B316 00           DB    00H
    
```

```

B317 56 52 41 4D          DB      'VRANDATA'      ; logged in disk
                                           ; The file name to be used
B31B 44 41 54 41
B31F 44 41 54          DB      'DAT'          ; The file extension or file type
B322                                DS      25          ; The rest of the FCB is used by the system
                                           ; and we don't have to worry about it at all
B33B 00                                COUNT: DB      00H      ; Count used for number of blocks to
                                           ; read from the screens VRAM
013C                                LEN      EQU      $-DEST
                                           END

```

The two following listings have been obtained using the DDT program.

```

0100 21 0E 01 11 00 B4 01 4E 01 ED B0 C3 00 00 21 C9 !.....N.....!
0110 F2 7E FE 03 C2 36 B4 CD 4F B4 FE FF CA 36 B4 3E ^.....6..0.....6.>
0120 FF 32 58 F3 3E 00 32 4D B5 CD 37 B4 CD 78 B4 FE .2X.>.2M..7..x..
0130 FF CA 8A B4 3A 4D B5 3C FE 20 20 EA 3E 00 32 58 .....:M.<..>.2X
0140 F3 CD 81 B4 C9 3A 4D B5 CB 07 32 A5 B4 3E 00 32 .....:M.<..2..>.2
0150 A4 B4 11 98 B4 2A 01 00 3E 72 85 6F E9 3E 00 32 .....*:..>r.o.>.2
0160 34 B5 21 34 B5 11 35 B5 01 19 00 ED B0 0E 0D CD 4.14..5.....
0170 05 00 0E 1A 11 AB B4 CD 05 00 CD 92 B4 0E 16 11 .....
0180 28 B5 CD 05 00 C9 0E 15 11 28 B5 CD 05 00 C9 0E (.....(.....
0190 10 11 28 B5 CD 05 00 C9 3E 00 32 58 F3 CD 81 B4 ..(.....>.2X...
01A0 0E 13 11 28 B5 CD 05 00 C9 A3 B4 04 00 A7 B4 79 ... (.....>.y
01B0 00 24 00 00 78 00 3C C3 59 2B AF 32 EA 3C C3 59 $.x.<.Y+.2.<.Y
01C0 2B 3E FF 32 F2 3C CD C4 28 AF 32 F2 3C 7A B7 C2 +>.2.<..(2.<z..
01D0 9D 04 3A 11 3E FE 20 C0 7B 3D FB CA 9D 04 FE 10 ...>..(=.....
01E0 D2 9D 04 3C 32 F1 3C AF 67 6B C3 34 1B CD C4 2B ...<2.<.gk.4...<
01F0 7A B7 C2 9D 04 3A EC 3C B7 CB 7B B7 CA E1 2B FE z.....<..(.....+
0200 0A DC 9D 04 3A 11 3E FE 20 C2 E1 2B 7B 32 32 3D .....>..+{22=
0210 CD 2B 19 CD 3F 04 C3 70 1A CD 55 0B C2 9D 04 F5 +.?.p..U.....
0220 3A E9 3D FE 08 DA FE 2B CD CD 04 3E 07 21 E3 3D !.....+.....>!.=
0230 77 23 22 3F 3D 3A 00 56 52 41 4D 44 41 54 41 44 w?="?.VRAMDATAD
0240 41 54 EA 2B C9 3A 2C 3D B7 C2 C1 04 CD CB 0B 2A AT

```

```

0100 21 0E 01 11 00 B2 01 3C 01 ED B0 C3 00 00 21 C9 !.....<.....!
0110 F2 7E FE 03 C2 36 B2 CD 4F B2 FE FF CA 36 B2 3E ^.....6..0.....6.>
0120 FF 32 58 F3 3E 00 32 3B B3 CD 75 B2 FE FF CA 2E .2X.>.2;..u.....
0130 B2 CD 37 B2 3A 3B B3 3C FE 20 20 EA 3E 00 32 58 ..7.;;<..>.2X
0140 F3 CD 7E B2 C9 3A 3B B3 CB 07 32 91 B2 3E 00 32 .....:..2..>.2
0150 90 B2 11 B7 B2 2A 01 00 3E 72 85 6F E9 3E 00 32 .....*:..>r.o.>.2
0160 22 B3 21 22 B3 11 23 B3 01 19 00 ED B0 0E 0D CD ".!"..f.....
0170 05 00 0E 1A 11 95 B2 CD 05 00 0E 0F 11 16 B3 CD .....
0180 05 00 C9 0E 14 11 16 B3 CD 05 00 C9 0E 10 11 16 .....
0190 B3 CD 05 00 C9 BF B2 7E 00 15 B3 01 00 25 00 00 .....%..
01A0 3C 02 00 3C B7 CA 59 2B 7D 32 8D 40 32 2A 3D C3 <<.<..Y+>2.@2*..
01B0 59 2B 3E 01 32 EA 3C C3 59 2B AF 32 EA 3C C3 59 Y+>.2.<.Y+.2.<.Y
01C0 2B 3E FF 32 F2 3C CD C4 28 AF 32 F2 3C 7A B7 C2 +>.2.<..(2.<z..
01D0 9D 04 3A 11 3E FE 20 C0 7B 3D FB CA 9D 04 FE 10 ...>..(=.....
01E0 D2 9D 04 3C 32 F1 3C AF 67 6B C3 34 1B CD C4 2B ...<2.<.gk.4...<
01F0 7A B7 C2 9D 04 3A EC 3C B7 CB 7B B7 CA E1 2B FE z.....<..(.....+
0200 0A DC 9D 04 3A 11 3E FE 20 C2 E1 2B 7B 32 32 3D .....>..+{22=
0210 CD 2B 19 CD 3F 04 C3 70 1A CD 55 0B C2 9D 04 F5 +.?.p..U.....
0220 3A E9 3D 00 00 56 52 41 4D 44 41 54 41 44 41 54 !.....VRAMDATADAT

```

Having typed them in and saved them under the file names SSAVE.COM and SLOAD.COM, proceed as follows to use them with BASIC:

- i) Execute SSAVE on the CP/M command line. The SSAVE program will relocate the appropriate part of the program to the memory locations beginning with B400H.
- ii) Execute SLOAD on the CP/M command line. The SLOAD program will relocate the appropriate part of the program to the memory locations beginning with B200H.
- iii) Enter BASIC using the command:

BASIC /M: &HB200

This will load BASIC but with the upper memory limit for variables set to B200H, thus protecting the two loaded programs.

- iv) Load and RUN the following program as an example:

```

10 SCREEN 3,0,0 :CLS
20 LINE (0,0) - (479,63),,B
30 LINE (0,0) - (479,63)
40 LINE (479,0) - (0,63)
50 S = &HB400:L = &HB200:REM start addresses to Save and Load
60 CALL S
70 CLS: LOCATE 27,1 : PRINT "The screen has been saved."
80 LOCATE 24,5 : PRINT "Press any key to load the screen."
90 IF INKEY$ = "" THEN 80
100 CLS
110 CALL L
120 LOCATE 24,5 : PRINT "Press any key to exit"
130 IF INKEY$ = "" THEN 130
140 SCREEN 0
    
```

## Appendix I

### CP/M ERRORS AND MESSAGES

When using CP/M and the associated utilities, many possible errors can occur. Messages can come from different sources. They can be displayed when there are errors in calls to the Basic Disk Operating System (BDOS). CP/M also displays errors when there are errors in command lines. The following list of error messages and sources of error covers errors in CP/M and the standard utilities. Some of these utilities may only be supplied on disk, but the error messages are presented as a single table to cover all these cases. Other application programs and the TERM and FILINK utility programs have their own error messages. Please consult the sections in this manual appropriate to these utilities or the manual provided with the application program, when using such programs.

Message	Meaning
?	This message has four possible meanings: 1) DDT does not understand the assembly language instruction. 2) The file cannot be opened. 3) A checksum error occurred in a HEX file. 4) The assembler/disassembler was overlaid.
ABORTED	You stopped a PIP operation by pressing a key.
ASM Error Messages	D Data error: data statement element cannot be placed in specified data area. E Expression error: expression cannot be evaluated during assembly. L Label error: label cannot appear in this context (might be duplicate label). N Not implemented: unimplemented features, such as macros, are trapped. O Overflow: expression is too complex to evaluate. P Phase error: label value changes on two passes through assembly. R Register error: the value specified as a register is incompatible with the code. S Syntax error: improperly formed expression.

Message	Meaning
	<p>U Underlined label: label used does not exist.</p> <p>V Value error: improperly formed operand encountered in an expression.</p>
BAD DELIMITER	Check command line for typing errors.
Bad Load	CCP error message, or SAVE error message.
Bdos Err On d:	Basic Disk Operating System Error on the designated drive: CP/M replaces d: with the drive specification of the drive where the error occurred. This message is followed by one of the four phrases in the situations described below.
Bdos Err On d: Bad Sector	This message appears when CP/M finds no disk in the drive, when the disk is improperly formatted, when the drive latch is open, or when power to the drive is off. Check for one of these situations and try again. This could also indicate a hardware problem or a worn or improperly formatted disk. Press CTRL-C to terminate the program and return to CP/M, or press the return key to ignore the error.
Bdos Err On d: File R/O	You tried to erase, rename, or set file attributes on a Read-Only file. The file should first be set to Ready-Write (RW) with the command: "STAT filespec \$R/W."
Bdos Err On d: R/O	Drive has been assigned Read Only status with a STAT command, or the disk in the drive has been changed without being initialized with a CTRL-C. CP/M terminates the current program as soon as you press any key.
Bdos Err on d: Select	CP/M received a command line specifying a nonexistent drive. CP/M terminates the current program as soon as you press any key. Press return key or CTRL-C to recover.
Break "x" at c	<p>"x" is one of the symbols described below and c is the command letter being executed when the error occurred.</p> <p># Search failure. ED cannot find the string specified in an F, S, or N command.</p> <p>? Unrecognized command letter c. ED does not recognize the indicated command letter, or an E, H, Q, or O command is not alone on its command line.</p> <p>O The file specified in an R command cannot be found.</p>

Message	Meaning
	<p>&gt; Buffer full. ED cannot put any more characters in the memory buffer, or the string specified in an F, N, or S command is too long.</p> <p>E Command aborted. A keystroke at the console aborted command execution.</p> <p>F Disk or directory full. This error is followed by either the disk or directory full message. Refer to the recovery procedures listed under these messages.</p>
CANNOT CLOSE DESTINATION FILE- {filespec}	An output file cannot be closed. You should take appropriate action after checking to see if the correct disk is in the drive and that the disk is not write-protected.
Cannot close, R/O CANNOT CLOSE FILES	CP/M cannot write to the file. This usually occurs because the disk is write-protected.
	An output file cannot be closed. This is a fatal error that terminates ASM execution. Check to see that the disk is in the drive, and that the disk is not write-protected.
	The disk file written by a W command cannot be closed. This is a fatal error that terminates DDT execution. Check if the correct disk is in the drive and that the disk is not write-protected.
	This error can occur during SUBMIT file processing. Check if the correct system disk is in the A drive and that the disk is not write-protected. The SUBMIT job can be restarted after rebooting CP/M.
CANNOT READ	PIP cannot read the specified source. Reader may not be implemented.
CANNOT WRITE	The destination specified in the PIP command is illegal. You probably specified an input device as a destination.
Checksum error	A hex record checksum error was encountered. The hex record that produced the error must be corrected, probably by recreating the hex file.
CHECKSUM ERROR LOAD ADDRESS hhhh ERROR ADDRESS hhhh BYTES READ: hhhh:	File contains incorrect data. Regenerate hex file from the source.
Command Buffer Overflow	The SUBMIT buffer allows up to 2048 characters in the input file.



Message	Meaning
Command too long	A command in the SUBMIT file cannot exceed 125 characters.
CORRECT ERROR, TYPE RETURN OR CTRL-Z	A hex record checksum was encountered during the transfer of a hex file. The hex file with the checksum error should be corrected, probably by recreating the hex file.
DESTINATION IS R/O, DELETE (Y/N)?	The destination file specified in a PIP command already exists and it is Read Only. If you type Y, the destination file is deleted before the file copy is done.
Directory full	There is not enough directory space for file being written to the destination disk. You can use the OX filespec command to erase any unnecessary files on the disk without leaving the editor.  There is not enough directory space to write the \$\$\$SUB file used for processing SUBMITs. Erase some files or select a new disk and retry.
Disk full	There is not enough disk space for the output file. This error can occur on the W, E, H, or X commands. If it occurs with X command, you can repeat the command prefixing the filename with a different drive.
DISK READ ERROR-{filespec}	The input disk file specified in a PIP command cannot be read properly. This is usually the result of an unexpected end-of-file. Correct the problem in your file.
DISK WRITE ERROR-{filespec}	A disk write operation cannot be successfully performed during a W command, probably due to a full disk. You should either erase some unnecessary files or get another disk with more space.  A disk write operation cannot be successfully performed during a PIP command, probably due to a full disk. You should either erase some unnecessary files or get another disk with more space and execute PIP again.  The SUBMIT program cannot write the \$\$\$SUB file to the disk. Erase some files, or select a new disk and try again.
ERROR: BAD PARAMETER	You entered an illegal parameter in a PIP command. Retype the entry correctly.

Message	Meaning
ERROR: CANNOT OPEN SOURCE, LOAD ADDRESS hhhh	Displayed if LOAD cannot find the specified file or if no filename is specified.
ERROR: CANNOT CLOSE FILE, LOAD ADDRESS hhhh	Caused by an error code returned by a BDOS function call. Disk may be write protected.
ERROR: CANNOT OPEN SOURCE, LOAD ADDRESS hhhh	Cannot find source file. Check disk directory.
ERROR: DISK READ, LOAD ADDRESS hhhh	Caused by an error code returned by a BDOS function call.
ERROR: DISK WRITE, LOAD ADDRESS hhhh	Destination Disk is full.
ERROR: INVERTED LOAD ADDRESS, LOAD ADDRESS hhhh	The address of a record was too far from the address of the previously-processed record. This is an internal limitation of LOAD, but it can be circumvented. Use DDT to read the hexfile into memory, then use a SAVE command to store the memory image file on disk.
ERROR: NO MORE DIRECTORY SPACE, LOAD ADDRESS hhhh	Disk directory is full.
Error on line nnn message	The SUBMIT program displays its messages in the format shown above, where nnn represents the line number of the SUBMIT file. Refer to the message following the line number.
FILE ERROR	Disk or directory is full, and ED cannot write anything more on the disk. This is a fatal error, so make sure there is enough space on the disk to hold a second copy of the file before invoking ED.
FILE EXISTS	You have asked CP/M to create or rename a file using a file specification that is already assigned to another file. Either delete the existing file or use another file specification.  The new name specified is the name of a file that already exist. You cannot rename a file with the name of an existing file. If you want to replace an existing file with a newer version of the same file, either rename or erase the existing file, or use the PIP utility.

Message	Meaning
File exists, erase it	The destination filename already exists when you are placing the destination file on a different disk than the source. It should be erased or another disk selected to receive the output file.
**FILE IS READ/ONLY**	The file specified in the command to invoke ED has the Read Only attribute. ED can read the file so that the user can examine it, but ED cannot change a Read Only file.
File Not Found	CP/M cannot find the specified file. Check that you have entered the correct drive specification or that you have the correct disk in the drive.  ED cannot find the specified file. Check that you have entered the correct drive specification or that you have the correct disk in the drive.  STAT cannot find the specified file. The message might appear if you omit the drive specification. Check to see if the correct disk is in the drive.
FILE NOT FOUND- {filespec}	An input file that you have specified does not exist.
Filename required	You typed the ED command without a filename. Reenter the ED command followed by the name of the file you want to edit or create.
hhhh?? = dd	The ?? indicates DDT does not know how to represent the hexadecimal value dd encountered at address hhhh in 8080 assembly language. dd is not an 8080 machine instruction opcode.
Insufficient memory	There is not enough memory to load the file specified in an R or E command.
Invalid Assignment	You specified an invalid drive or file assignment, or misspelled a device name. This error message might be followed by a list of the valid file assignments that can follow a filename. If an invalid drive assignment was attempted the message "Use: d: = RO" is displayed, showing the proper syntax for drive assignments.
Invalid control character	The only valid control characters in the SUBMIT files of type SUB are ^A through ^Z. Note that in a SUBMIT file the control character is represented by typing the cir-

Message	Meaning
	cumflex, ^, not by pressing the control key.
INVALID DIGIT- {filespec}	An invalid hex digit has been encountered while reading a hex file. The hex file with the invalid hex digit should be corrected, probably by recreating the hex file.
Invalid Disk Assignment	Might appear if you follow the drive specification with anything except = R/O.
INVALID DISK SELECT	CP/M received a command line specifying a nonexistent drive, or the disk in the drive is improperly formatted. CP/M terminates the current program as soon as you press any key.
INVALID DRIVE NAME (Use A, B, C, or D)	SYSGEN recognizes only drives A, B, C and D as valid destinations for system generation.
Invalid File Indicator	Appears if you do not specify RO, RW, DIR, or SYS.
INVALID FORMAT	The format of your PIP command is illegal. See the description of the PIP command.
INVALID HEX DIGIT LOAD ADDRESS hhhh ERROR ADDRESS hhhh BYTES READ: hhhh	File contains incorrect hex digit.
INVALID MEMORY SIZE	Specify a value less than 64K or your computer's actual memory size.
INVALID SEPARATOR	You have placed an invalid character for a separator between two input filenames.
INVALID USER NUMBER	You have specified a user number greater than 15. User numbers are in the range 0 to 15.
n?	You specified a number greater than fifteen for a user area number. For example, if you type USER <input type="text" value="18"/> , the screen displays 18?.
NO DIRECTORY SPACE	The disk directory is full. Erase some files to make room for PRN and HEX files. The directory can usually hold only 64 filenames.
NO DIRECTORY SPACE- {filespec}	There is not enough directory space for the output file. You should either erase some unnecessary files or get

Message	Meaning
	another disk with more directory space and execute PIP again.
NO FILE-{filespec}	CP/M cannot find the specified file, or no files exist.  The indicated source or include file cannot be found on the indicated drive.  The file specified in an R or E command cannot be found on the disk.
NO INPUT FILE PRESENT ON DISK	The file you requested does not exist.
No memory	There is not enough (buffer?) memory available for loading the program specified.
NO SOURCE FILE ON DISK	SYSGEN cannot find CP/M either in CPMxx.com form or on the system tracks of the source disk.
NO SOURCE FILE PRESENT	The assembler cannot find the file you specified. Either you mistyped the filespecification in you command line, or the file is not type ASM.
NO SPACE	Too many files are already on the disk, or no room is left on the disk to save the information.
No SUB file present	For SUBMIT to operate properly, you must create a file with filetype of SUB. The SUB file contains usual CP/M commands. Use one command per line.
NOT A CHARACTER SOURCE	The source specified in your PIP commands is illegal. You have probably specified an output device as a source.
**NOT DELETED**	PIP did not delete the file, which may have had the R/O attribute.
NOT FOUND	PIP cannot find the specified file.
OUTPUT FILE WRITE ERROR	You specified a write-protected diskette as the destination for the PRN and HEX files, or the diskette has no space left. Correct the program before assembling your program.
Parameter error	Within the SUBMIT file of type sub, valid parameters are \$0 through \$9.

Message	Meaning
PARAMETER ERROR, TYPE RETURN TO IGNORE	If you press return, SYSGEN proceeds without processing the invalid parameter.
QUIT NOT FOUND	The string argument to a Q parameter was not found in you input file.
Read error	An error occurred when reading the file specified in the type command. Check the disk and try again. The STAT filespec command can diagnose trouble.
READER STOPPING	Reader operation interrupted.
Record Too Long	PIP cannot process a record longer than 128 bytes.
START NOT FOUND	The string argument to an S parameter cannot be found in the source file.
SOURCE FILE INCOMPLETE	SYSGEN cannot use your CP/M source file.
SOURCE FILE NAME ERROR	When you assemble a file, you cannot use the wildcard characters * and ? in the filename. Only one file can be assembled at a time.
SOURCE FILE READ ERROR	The assembler cannot understand the information in the file containing the assembly language program. Portions of another file might have been written over your assembly language file, or information was not properly saved on the diskette. Use the TYPE command to locate the error. Assembly language files contain the letters, symbols, and numbers that appear on your keyboard. If your screen displays unrecognizable output or behaves strangely, you have found where computer instructions have crept into your file.
SYNCHRONIZATION ERROR	The MOVCPM utility is being used with the wrong CP/M system.
"SYSTEM" FILE NOT ACCESSIBLE	You tried to access a file set to SYS with the STAT command.
**TOO MANY FILES**	There is not enough memory for STAT to sort the files specified, or more than 512 files were specified.
UNEXPECTED END OF HEX FILE-{filespec}	An end-of-file was encountered prior to a termination hex record. The hex file without a termination record should be corrected, probably by recreating the hex file.

Message	Meaning
Unrecognized Destination	Check command line for valid destination.
Use: STAT d: = RO	An invalid STAT drive command was given. The only valid drive assignment in STAT is STAT d: = RO.
VERIFY ERROR: {filespec}	When copying with the V option, PIP found a difference when rereading the data just written and comparing it to the data in its memory buffer. Usually this indicates a failure of either the destination disk or drive.
XSUB ACTIVE	XSUB has been invoked.
XSUB ALREADY PRESENT	XSUB is already active in memory.
Your input?	If CP/M cannot find the command you specified, it returns the command name you entered followed by a question mark. Check that you have typed the command line correctly, or that the command you requested exists as a .COM file on the default or specified disk.

# Index

## A

- ABORTED I-1
- AC adapter 2-1
- Acoustic coupler 3-76, 4-26, 4-42  
connection 4-42
- A/D converter see Analog/Digital converter ADCVRT subroutine 5-34  
example of use H-1
- Afn 3-5
- ALARM 2-22, 2-27, 2-47  
setting 2-27, 2-29
- <ALARM MSG> 2-28
- ALARM string 2-27
- Ambiguous file names 3-5
- Analog/Digital converter 1-9, 4-37  
example of use H-1  
location 4-37
- Arrow keys 2-14
- Alphanumeric keys 2-11
- ASCII codes E-1  
printers and E-2  
table E-3, E-4, E-5, E-6, E-7, E-8
- ASM (8080 assembler) 3-92  
error messages I-1
- Assembly language programming 3-92, 5-1
- AUTO POWER OFF time 2-42, 2-47, 3-61
- Auto repeat keys 2-11  
controlling using ESC sequence A-17
- AUTOSTART 2-23, 2-31, 2-32, 2-47
- AUTOSTART string 2-32  
rules for setting up 2-34

## B

- Backspacing 2-15
- BAD DELIMITER I-2
- Bad load I-2
- Bar code reader 4-37
- BASIC 1-10, 2-20  
CALL and BIOS calls H-4

Message	Meaning
Unrecognized Destination	Check command line for valid destination.
Use: STAT d: = RO	An invalid STAT drive command was given. The only valid drive assignment in STAT is STAT d: = RO.
VERIFY ERROR: -{filespec}	When copying with the V option, PIP found a difference when rereading the data just written and comparing it to the data in its memory buffer. Usually this indicates a failure of either the destination disk or drive.
XSUB ACTIVE	XSUB has been invoked.
XSUB ALREADY PRESENT	XSUB is already active in memory.
Your input?	If CP/M cannot find the command you specified, it returns the command name you entered followed by a question mark. Check that you have typed the command line correctly, or that the command you requested exists as a .COM file on the default or specified disk.

# Index

## A

ABORTED I-1  
 AC adapter 2-1  
 Acoustic coupler 3-76, 4-26, 4-42  
     connection 4-42  
 A/D converter see Analog/Digital converter ADCVRT subroutine 5-34  
     example of use H-1  
 Afn 3-5  
 ALARM 2-22, 2-27, 2-47  
     setting 2-27, 2-29  
 <ALARM MSG> 2-28  
 ALARM string 2-27  
 Ambiguous file names 3-5  
 Analog/Digital converter 1-9, 4-37  
     example of use H-1  
     location 4-37  
 Arrow keys 2-14  
 Alphanumeric keys 2-11  
 ASCII codes E-1  
     printers and E-2  
     table E-3, E-4, E-5, E-6, E-7, E-8  
 ASM (8080 assembler) 3-92  
     error messages I-1  
 Assembly language programming 3-92, 5-1  
 AUTO POWER OFF time 2-42, 2-47, 3-61  
 Auto repeat keys 2-11  
     controlling using ESC sequence A-17  
 AUTOSTART 2-23, 2-31, 2-32, 2-47  
 AUTOSTART string 2-32  
     rules for setting up 2-34

## B

Backspacing 2-15  
 BAD DELIMITER I-2  
 Bad load I-2  
 Bar code reader 4-37  
 BASIC 1-10, 2-20  
     CALL and BIOS calls H-4

- from the MENU 2-40
- Microcassette and 4-10
- programs and the MENU 2-40, 2-41
- resident 2-35
- restarting from CP/M command line 3-28
- screen modes in 2-50
- setting parameters from 2-48
- Submit files from 3-57
- Basic Disk Operating System (BDOS) 2-21
- BAT: 3-14, 5-2
- Battery C-1
  - charging 2-2, C-1
  - charge problems C-2
  - charge Time 2-3
  - insertion 2-2
  - length of use on a charge 2-2
  - replacing C-4
  - Switching off during extended storage C-2
- BDOS 2-21, 5-1
  - function calls 5-4
  - location of 5-8
- Bdos error on d: I-2
- Bdos error on d: bad sector I-2
- Bdos error on d: file R/O I-2
- Bdos error on d: select I-2
- BEEP subroutine 5-20
- BIOS 5-1
  - example programs Appendix H
  - interface 5-9
  - subroutine functions 5-10
  - table of entry addresses 5-10
- Blank screen 2-3, 2-20
- Books about CP/M 3-93
- BOOT subroutine 5-11
- Break "x" at c I-2
- BS key 2-15
- Buffer (keyboard) 2-12

## C

- CALLX subroutine 5-32
- Cannot close I-3
- Cannot read I-3
- CAPS LOCK 2-12
- CAPS LED 2-12
- Carriage return 2-14
- Carriage handle 1-5

- CCP 2-21, 5-1
- Central processor units 1-8
- Change device assignments 3-46
- Changing character sets 3-74
- Changing disks 3-8
- Changing parameters (see System Display and CONFIG) 2-49
- Changing Roms 3-9
- Changing Tapes 3-9
- Character deletion commands 3-10
- Character generator ROM A-5
- Character printing using ESC sequence A-14
- Character set (see also DIP switch setting) 2-47, 3-74, A-6
- Charging battery 2-2, C-1
- CHARGE BATTERY C-1
- Charge time 2-3
- Checksum error I-3
- Clear line from cursor to end A-7
- Clearing
  - the screen 2-15, 2-52, A-5
  - from cursor to end A-7
- Clock module 2-21, 2-27
- CLR key 2-15
- COM file 3-4
- Command file 3-4
- Command buffer overflow I-3
- Command too long I-4
- Communication using PIP 3-38
- Communications 3-75
  - cable for PX-8 to PX-8 4-26
  - with QX-10 computer 3-79
- CON: 3-13, 5-2
- CONFIG program 2-8, 2-16, 2-18, 2-23, 2-29, 2-50, 3-62
  - parameters changed by 2-47, 3-62
- CONIN subroutine 5-12
- CONOUT subroutine 5-12
- Console Command Processor(CCP) 2-21
- Console ESCAPE sequences A-1
- CONST subroutine 5-11
- Control codes 2-12
- Continue mode 2-19, 2-20, 2-32
- COPYDISK 3-91
- Copying files (see PIP) 3-35
- Copying Floppy Disks 3-91
- Correct error I-4
- Counter of tape 4-5
- CP/M 1-10, Chapter 3
  - built in commands 3-16
  - configuration 5-1
  - disk utilities 3-91

- finding out more 3-93
- function keys assignments 3-62
- function key display 3-64
- program utilities 3-91
- CP/M command line 2-20
  - and character 2-13
  - and cursor keys 2-14
  - and MENU 2-38
  - keyboard characters accepted by 3-10
  - printing from 3-12
  - turning to from MENU 2-42
- CPUs 1-9
- CRT: 3-15
- CTRL keys 2-12
- CTRL key used to switch off in continue mode 2-19
- CTRL-C 2-12
  - and warm starts 3-9
  - CTRL-HELP 2-15
  - CTRL-L 2-15, 2-52
  - CTRL-M 2-14
  - CTRL-P to toggle printer 3-12
  - CTRL-STOP 2-14
- Current drive 2-47
- Cursor changing 3-64
- Cursor keys and CP/M 2-14
- Cursor keys 2-7, 2-14
- Cursor
  - move to position on screen A-6
  - redefining using ESC sequence A-15
  - switching off A-5
  - switching on 3-64, A-5
- Cursor type 2-47, 3-64
  - changing with ESC sequence A-15

## D

- Date setting 2-7, 2-47, 3-65
- Day setting 2-8, 2-47
- DDT 3-91
- DEL key 2-15
- Default settings 2-47
- Deleting characters 2-15, 3-10
  - lines 3-11
- Destination is R/O I-4
- Device assignments 3-46, 3-48
- DIP switch 2-3
- DIR 3-6, 3-17

- Directory full I-4
- Directory of disk 3-6, 3-17
  - pausing 3-20
  - printing 3-20
  - stopping 3-20
- Disk assignments 2-47, 3-66
- Disk characteristics 3-47
- Disk drive allocation 3-66
- Disk drive connection 4-29
- Disk full I-4
- Disk read error I-4
- Disk write error I-4
- DISKROV 5-19
- DISKTBL 5-19
- Displaying contents of a file using TYPE 3-30
- Display control commands 3-11
- Display does not appear 2-3
- Display screen 1-8
- DRINIT 4-6
- Drive, current 2-47
- Drive name assignments Appendix B
- Drive names 3-6
- Dual screen 2-56
- DUMP 3-92

## E

- ED 3-91
- ERA 3-16, 3-21
- Erasing files from directory 3-21
  - of microcassette tape 3-23
- Error bad parameter I-4
- Error cannot open source I-5
- Error cannot close file I-5
- Error cannot open source I-5
- Error disk read I-5
- Error disk write I-5
- Error inverted load address I-5
- Error no more directory space I-5
- Error on line nn I-5
- ESC key 2-12, 2-14
- ESCAPE 2-14
  - sequences 2-47, A-1
  - sequences examples of use A-4
- Executing a series of commands 3-54
- Extension to files 3-3
  - examples of 3-4

## F

- FCB 5-3
- Files and Filenames 3-3
  - characters allowed 3-3
  - control block (FCB) 5-3
  - size and attributes 3-50, 3-51
  - storage devices 4-1
- File exists 1-5
- File is read only 1-6
- File not found 1-6
- Filename required 1-6
- File types on MENU 2-36, 2-39, 2-47
- FILINK 3-32, 3-88
- Floppy Disk
  - drives 4-21
  - connection 4-29
- Folding legs 1-7
- Forgetting Password 2-26
- Format RAM disk 2-10
- FORMAT for disks 3-91
- Freezing screen window 2-15
- FUNCFLG 5-12

## G

- GETPFK subroutine 5-14
- GO 2-41
- GRPH key 2-13
- Graphics
  - characters 2-13, E-2
  - and printer 4-33
- Graphics screen 2-57

## H

- hhhh? = dd error 1-6
- Halting display temporarily 2-14
- Halting the computer temporarily 2-18
- HELP 2-14
- Hide files on directory 3-49
- HOME key 2-16
- HOME disk subroutine in BIOS 5-16
- How to use the manuals v

## I

- Initialization 2-19
  - to overcome Password 2-26
- Initialization questions 2-7
  - reset 2-46
- INS 2-15
- Inserting characters 2-15
- Inserting battery 2-2
- Insufficient memory 1-6
- Intelligent RAM disk 2-5, 2-9
- Interfacing to other devices 4-1
- International characters table E-9
- Invalid error messages 1-6, 1-7
- IOBYTE 2-47, 5-2
- Italian characters 1-7, 3-74

## J

- JUMPX subroutine 5-32

## K

- Key response 2-12
- Keyboard and CP/M 3-10
  - buffer clearing with ESC sequences A-18
  - cover 1-6
  - layouts 1-6
  - layouts altering 2-3
  - operation 2-11

## L

- Legs positioning 1-7
- LCD (Liquid Crystal Display) 1-8, 2-50
- LCD opening 1-8
- LDIRX subroutine 5-31
- LED A-10, A-11
- LIGHTPEN subroutine 5-33
- Line termination commands 3-11
- Line drawing using ESC sequence A-12
- LIST subroutine 5-13
- LISTST subroutine 5-14
- LOADX subroutine 5-31



Logical and physical devices 3-14  
tables 3-16, 5-2  
Lower case 2-12  
LPT: 3-15, 5-2  
LST: 3-14, 5-2

## M

Main processor 1-9  
MASKI subroutine 5-34  
<MCT MODE> 2-43  
MCMTX subroutines 5-36  
Memory Map Appendix F  
MENU  
screen 2-10, 2-20, 2-21, 2-22, 2-23, 2-36, 2-47  
activation conditions 2-42  
command line 2-37  
default settings 2-47  
header 2-37  
interruption off 2-42  
reverting to CP/M command line 2-42  
setting files types to be displayed 2-36, 2-39  
termination of 2-42  
using 2-37  
using WAKE and AUTOSTART strings with 2-34  
Microcassette 4-1  
BASIC and 4-10  
care of 4-2  
changing tapes 2-44, 3-9  
checking if tape is mounted 4-9  
counter 2-43, 2-44  
deck 1-9  
directory 2-44, 3-19, 3-23, 4-4, 4-9  
directory initialization 4-6  
drive name 4-1  
EJECT button 4-6  
erasing files 3-23, 4-5  
executing a program from 4-10  
insertion of tape 4-2  
loading 4-7  
maximum files 1-9, 4-2  
manual setting 2-43, 4-4  
mode 2-43  
mounting 2-44, 4-4, 4-7  
non-stop and stop modes 2-43, 4-11  
removing 2-44, 4-5, 4-7, 4-8, 4-9  
selection and use 4-2

stop 2-44  
STOP in an emergency 2-44  
summary of operation 4-14  
verify and non verify mode 2-43  
Microcassette I/O System (MIOS) 2-21  
Microcassette Tape Operating System (MTOS) 2-21, 4-1  
MIOS 2-21, 5-1  
Modem 4-26  
MOUNT see under Microcassette  
MTOS 2-21, 5-1

## N

NO directory space I-7  
No display 2-3  
NO FILE 3-7, I-8  
No space I-8  
No sub file present I-8  
Non-stop mode of microcassette 2-43, 4-11  
Not deleted I-8  
Not found I-8  
NUM 2-13  
Numerical keypad 2-13

## O

Object Code 3-4  
Operating systems 1-10  
Operating System modules 2-21  
Operating the Computer Chapter 2

## P

Password 2-20, 2-21, 2-22, 2-24, 2-34, 2-47  
Password removal 2-25  
Password forgetting 2-26  
PAUSE 2-14  
PF keys 2-16  
PIP 3-32, 3-34  
optional parameters 3-37  
with printer 4-32  
Point of graphics screen set using ESC sequence A-13  
Portable Calc 1-11, 3-33  
Portable Scheduler 1-11  
Portable Wordstar 1-11, 3-33

- and SUBMIT files 3-58
- Power failure 2-19
- POWEROFF routine 2-19, 5-11
- Power on automatically, see WAKE
- Power BIOS subroutines 5-11
- Power switch 2-18
- Precautions on using the AC adapter 2-1
- Precautions on using the Px-8 2-1
- Printer 4-32
  - ASCII codes E-2
  - cable 4-26
  - examples of use 4-32
  - output port 2-47, 3-67, 3-73
  - unintelligible output 4-31
- Printing
  - from CP/M command line 3-12
  - contents of a file using PIP 3-35
  - contents of a file using TYPE 3-30
- Programmable function keys 2-16
  - and CP/M 3-10, 3-62
  - disabling and enabling A-11
  - displaying 3-64
  - display switching using ESC sequence A-15
  - and Microcassette Tape 4-5
- Protecting files
  - from other users 3-31
  - make Read/Only 3-48
  - removing protection 3-49
- PSET subroutine 5-15
- PTP: 3-14, 5-2
- PTR: 3-14, 5-2
- PUN: 3-13, 5-2
- PUNCH subroutine 5-13
- PUTPFK subroutine 5-14

## Q

- Quit not found I-9
- QX-10 Terminal mode 3-79, 3-81
- QX-10 communication 3-79

## R

- RAM DISK 2-8, 2-23, 3-66
  - additional 4-19
  - default 2-9, 2-47
  - format 2-10

- see also Intelligent RAM disk
- RDR: 3-13, 5-2
- RDVRAM subroutine 5-33
- READ subroutine 5-18
- READER subroutine 5-13
- Real screen 2-50
- Receiving files 3-83, 3-89
- Record too long I-9
- Removing Password 2-25
- REN 3-16, 3-25
- Renaming files 3-25
- Resetting the system 2-5, 2-46
- Reset altering keyboard 2-5
- Reset summary table 2-47
- Reset CPU 2-5
- Reset complete CPU 2-5
- Reset simple 2-46
- Reset sub CPU 2-5, 2-19, 2-46
- Response of keys 2-12
- Restart mode 2-19, 2-20, 2-32
- Restarting programs 3-28
- RETURN key 2-14
- RETURN ASCII code 2-14
- ROMs and ROM sockets 4-17
  - changing 3-9, 4-15
  - containing data or BASIC programs 4-18
- RS-232C 2-47, 3-67, 4-22
  - BIOS routines 5-20
  - cables 4-26
  - changing parameters 3-68
  - communications with 3-75, 4-22
  - connection 4-24
  - default settings 4-23
    - and printer 4-23
    - pin assignments 4-25
- RSCLOSE subroutine 5-21
- RSIN subroutine 5-22
- RSINST subroutine 5-21
- RSIOX subroutines 5-23
  - RSIOX(CLOSE) 5-24
  - RSIOX(CTLIN) 5-26
  - RSIOX(ERSTS) 5-27
  - RSIOX(GET) 5-25
  - RSIOX(INSTS) 5-24
  - RSIOX(OPEN) 5-23
  - RSIOX(OUTST) 5-25
  - RSIOX(PUT) 5-26
  - RSIOX(SENS) 5-27
  - RSIOX(SETCTL) 5-27

RSOPEN subroutine 5-21  
RSOUTST subroutine 5-22  
RSOUT subroutine 5-22

## S

SAVE 2-20, 3-16, 3-27  
Save a block of memory 3-27  
SAVE GO.COM 2-41, 3-28  
Screen guide to using 2-50  
Screen blank on switching on 2-3, 2-19  
Screen saving to disk H-6  
Screen Dump 2-17, 2-21, 2-45  
    and DIP switch setting 2-4, 4-33  
    using ESC sequence A-7  
    and graphics characters 4-33  
    halting 2-45  
Screen mode 3-69  
    changing 3-70  
    changing with ESC sequence A-13  
    default 2-47  
Screen mode 0 2-51  
Screen mode 1 2-55  
Screen mode 2 2-56  
Screen mode 3 2-57  
Screen printing - halting 2-14  
SCRN key 2-15, 3-12  
SCRNDUMP subroutine 5-16  
Scrolling screen 2-52  
Scrolling screen using ESC sequences A-8, A-9, A-10  
Secret mode A-7, A-8  
SECTAN subroutine 5-19  
SELDSK subroutine 5-16  
Sending files 3-84, 3-89  
Serial interface 2-47, 3-72, 4-29  
    connector 4-29  
    pin assignments 4-30  
Series of commands 3-55  
SETDMA subroutine 5-18  
SETSEC subroutine 5-17  
Setting the ALARM 2-27  
SETTRK subroutine 5-17  
Shift key setting using ESC sequence A-19  
SHIFT IN/SHIFT OUT 4-22, 5-21  
SLAVE  
    subroutine 5-33  
    example of use H-6

Slave processor 1-9  
Source Code 3-4  
Space on disk 3-52  
Spanish characters 1-7, 3-74  
Speaker 4-34  
    BIOS subroutine 5-20  
    external jack location 4-34  
    tape through 2-43  
Special keys 2-14  
Split screen 2-55  
Start not found I-9  
Starting up the Px-8 2-1  
STAT program 2-38, 3-32, 3-45  
    summary of commands 3-53  
STOP key 2-14  
Stop mode of microcassette 2-43, 4-11  
Storage C-2  
Storing on second virtual screen 2-54  
STORX subroutine 5-31  
Sub CPU 1-9  
SUBMIT 3-32, 3-54  
    comments in 3-58  
    files from BASIC 3-57  
    files from Wordstar 3-58  
    keyboard input see XSUB  
Switch keys 2-12  
Switching the PX-8 on and off 2-18  
System activator 2-21  
System Bank 5-30  
System Bus Interface 4-39  
System Display 2-15, 2-21, 2-22  
    parameters changed by 2-47  
System file not accessible I-9  
System Interface 5-1  
System prompt 3-6

## T

TAB 2-15  
TERM 3-32, 3-77  
Terminal mode 3-82  
TIMEDAT  
    subroutines for clock 5-28  
TIMEDAT(ALARM DISABLE) 5-29  
TIMEDAT(ALARM ENABLE) 5-29  
TIMEDAT(READ ALARM) 5-30  
TIMEDAT(READ TIME) 5-29

TIMEDAT(SET ALARM) 5-29  
TIMEDAT(SET TIME) 5-29  
Time setting 2-7, 3-65  
Too many files I-9  
Tracking mode 2-15, 2-50, 2-51  
    setting using ESC sequences A-9  
TTY: 3-15, 5-2  
TYPE 3-16, 3-27

## U

UC1: 3-15, 5-2  
Ufn 3-5  
UL1: 3-14, 5-2  
Unambiguous file names 3-4  
Universal Unit 4-38  
Unpacking notes 1-1  
UP1: and UP2: 3-14, 5-2  
Upper case 2-12  
UR1: and UR2: 3-15, 5-2  
Use: STAT error I-10  
User bank 5-30  
USER BIOS 2-9, 2-23, 2-47, 3-73  
USERBIOS subroutine 5-36  
User defined characters 2-13, 2-47  
    changed by reset 2-48  
    setting with ESC sequence A-16  
USER 3-16, 3-31  
    status 3-52  
Utilities 1-10, 3-32

## V

Verify and non-verify modes of microcassette 2-43, 4-11, 4-13  
Verify error I-10  
Virtual screens 2-50  
    changing between 2-54, 3-12  
    changing with ESC sequence A-14  
    description of using 2-51  
    lines in screen mode 0, 3-70  
    lines in screen mode 1, 3-71  
    lines in screen mode 2, 3-71  
    lines in screen mode 3, 3-72

## W

WAKE 2-22, 2-27  
    rules for setting 2-34  
    setting 2-30  
Wake treated as an Alarm 2-25, 2-32  
Warm start 2-14, 3-9, 4-24  
WBOOT 5-11  
    how to find 5-8  
Wildcards  
    and ALARM setting 2-29  
    and CP/M files 3-5  
    problems if used with ERA 3-22  
Winding tape 2-43  
Window on virtual screen  
    default setting 2-47  
    freezing 2-15  
    moving with ESC sequence A-15, A-16  
WRITE subroutine 5-18

## X

XON/XOFF 4-22, 5-21  
XSUB 3-32, 3-54, 3-60  
    for keyboard input in SUBMIT files 3-60  
XSUB active I-10

## Z

Zero page locations 5-8

^ meaning CTRL 2-13, 2-31  
↵ : meaning RETURN 2-16  
? as error message in CP/M I-1, 2-52