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,FØ	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 SCRN 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,43,44l
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 progrm 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,{
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.

$0 \leqq (n-)1 < (R-1)$
$1 \leqq m \leqq R$
$(n-1+m-1) < R$

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:

$0 \leqq (n-1) \leqq (R-1)$
$1 \leqq m < R$
$(n-1+m-1) < P$

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

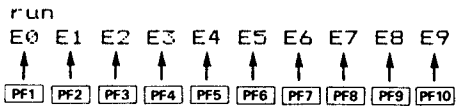
Turns off the NUM LED.

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 EØ 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 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 $0 \leqq n1 \leqq 63$, $0 \leqq n2 \leqq 479$
Hexadecimal $0 \leqq n1 \leqq 3F$, $0 \leqq n2 \leqq 1DF$

ESC 208

Switches the display mode. Mode specification is as follows:

	Dec	Hex
Mode 0		
Byte 1:	27	1B
Byte 2:	208	D0
Byte 3:	2	2
Byte 4:	n1	n1
Byte 5:	n2	n2
Mode 1		
Byte 1:	27	1B
Byte 2:	208	D0
Byte 3:	1	1
Byte 4:	n1	n1

	Dec	Hex
Mode 2		
Byte 1:	27	1B
Byte 2:	208	D0
Byte 3:	2	2
Byte 4:	n1	n1
Byte 5:	m	m
Byte 6:	p	p
Mode 3		
Byte 1:	27	1B
Byte 2:	208	D0
Byte 3:	3	3

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 it his 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 possiblity 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.

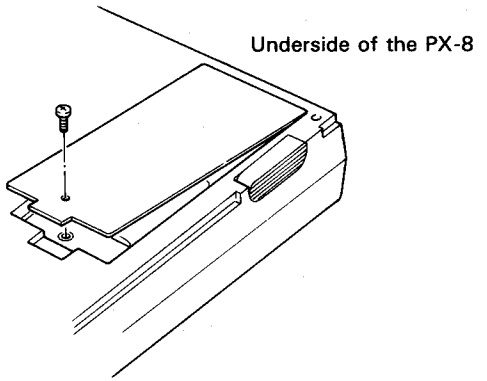


Fig. C.1

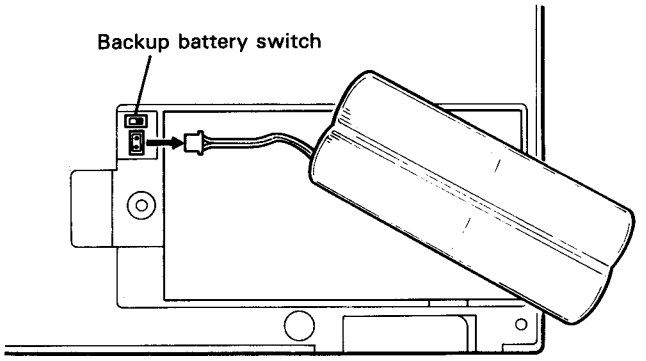


Fig. C.2

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:

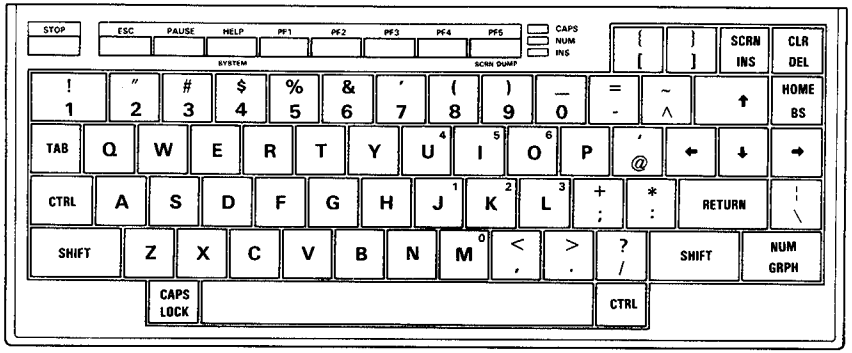
1) SAVE ALL PROGRAMS AND DATA ON TAPE OR DISK. All programs and data in memory, including the RAM disk and Intelligent RAM Disk may be lost when the batteries are changed.
2) Leave the PX-8 switched on until the "CHARGE BATTERY" message comes up, i.e. the main battery becomes discharged.
3) Turn the 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.

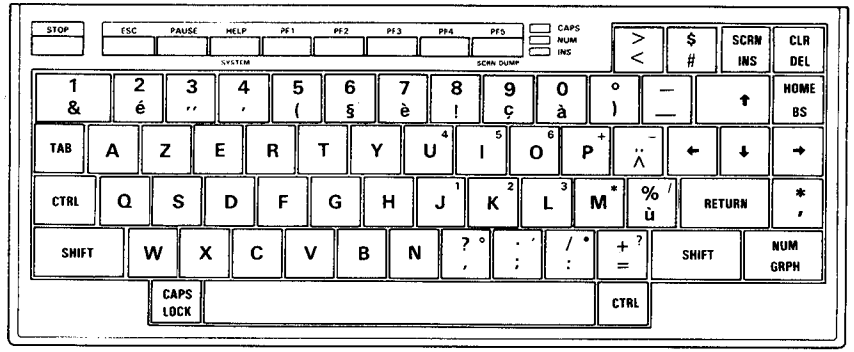
Appendix D

KEYBOARD LAYOUTS BY COUNTRY

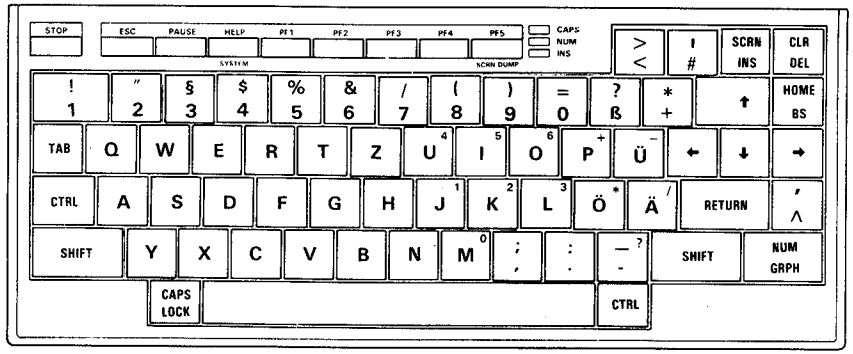
1) ASCII



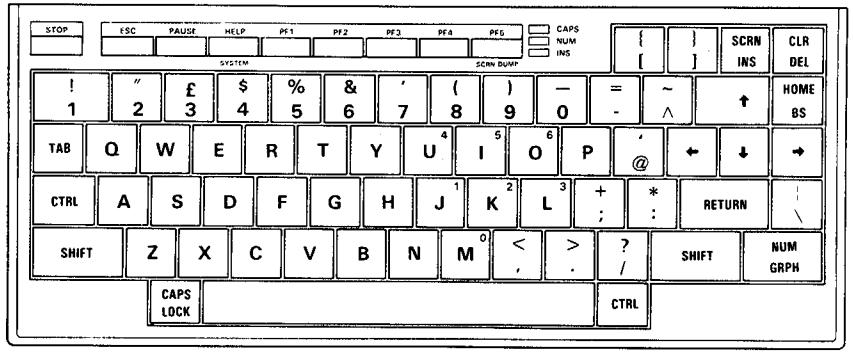
2) France



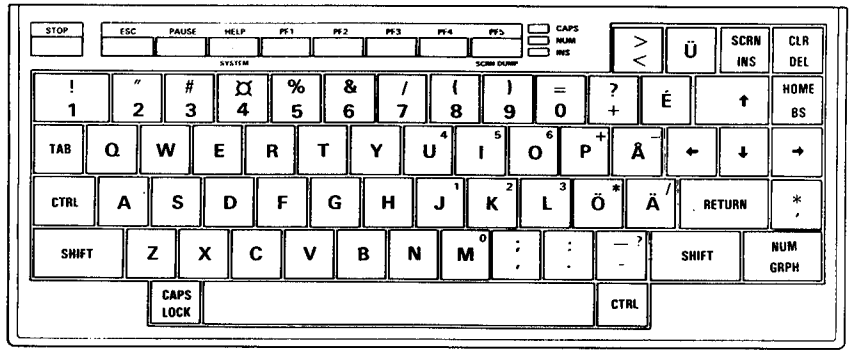
3) Germany



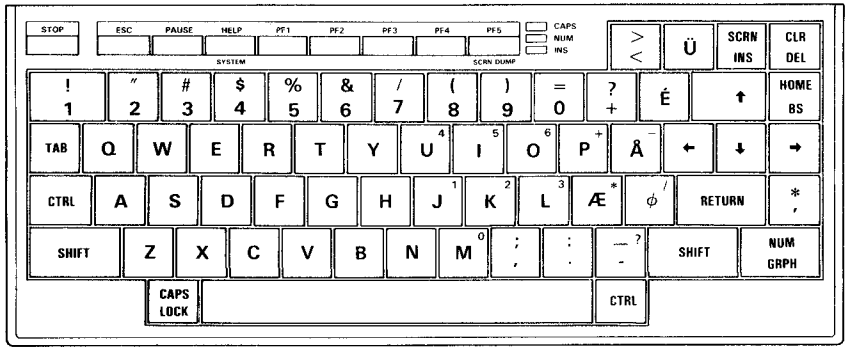
4) England



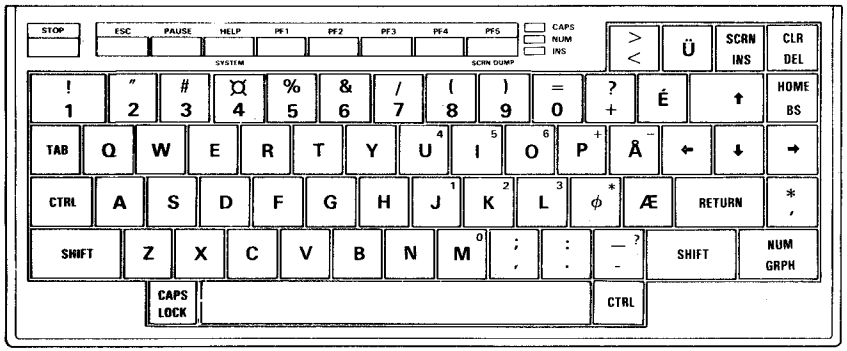
5) Sweden



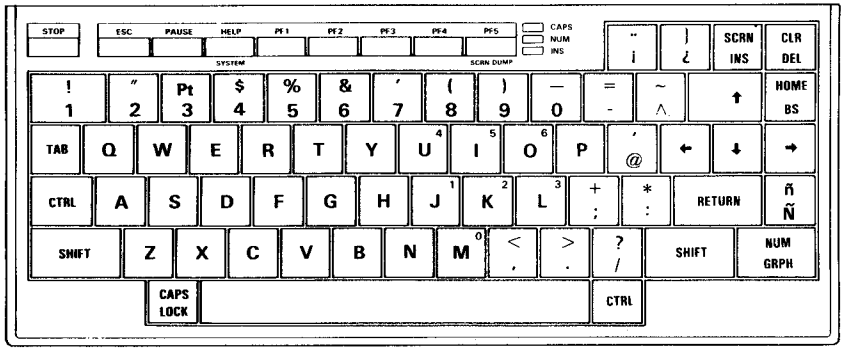
6) Denmark



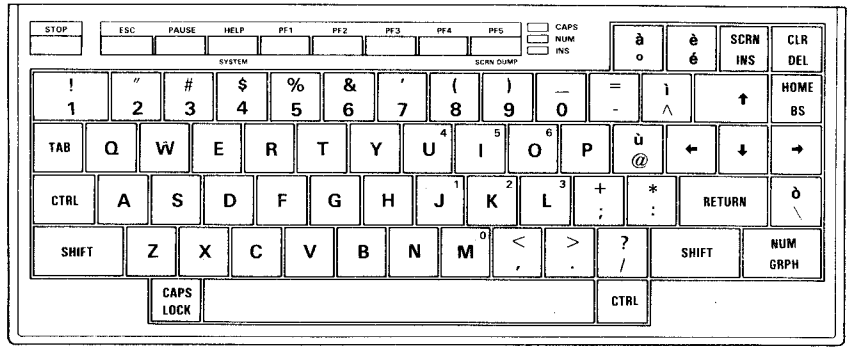
7) Norway



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	Δ ↵ User defined characters	GRPH-0
E1	225		GRPH-1
E2	226		GRPH-2
E3	227		GRPH-3
E4	228		GRPH-4
E5	229		GRPH-5

HEX	DECIMAL	CHAR	KEYSTROKES(ASCII) to obtain the character
E6	230	User defined characters	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		

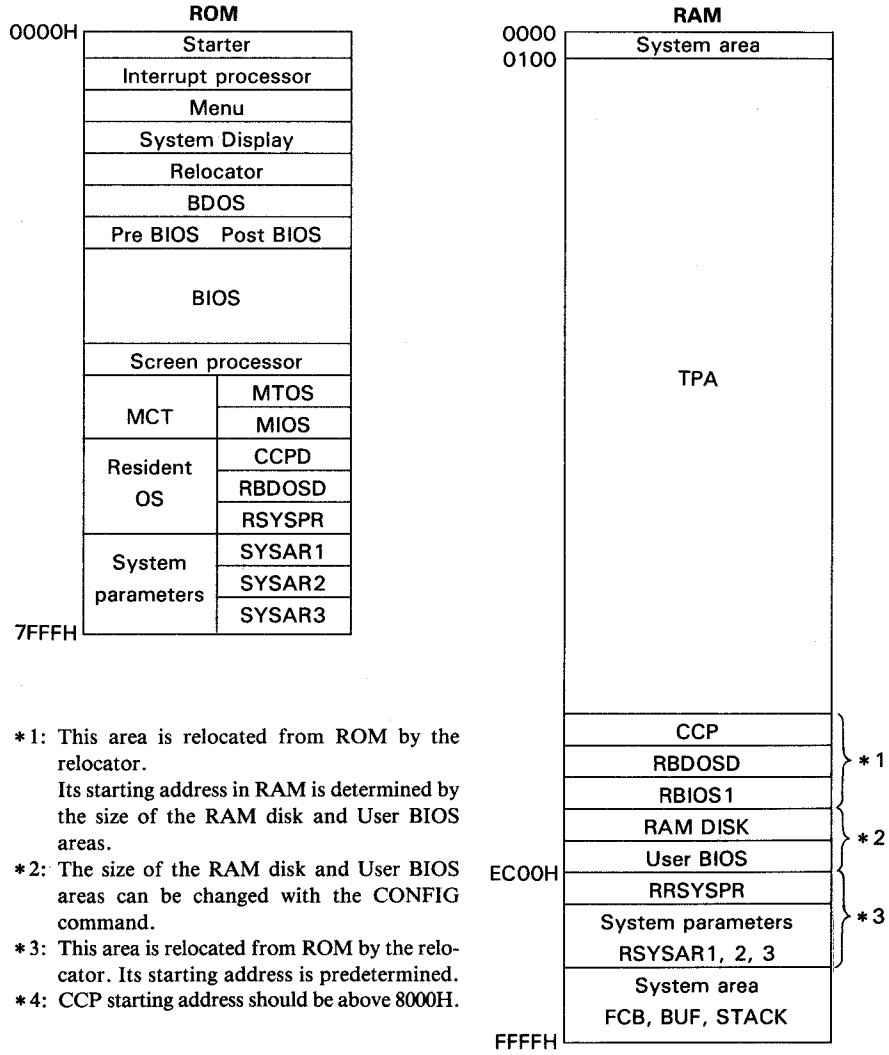
E-5 INTERNATIONAL CHARACTER SETS

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	@	à	§	@	É	É	@	@	É
91	[	°	Ä	[	Æ	Ä	°	¡	Æ
92	\	ç	Ö	\	Ø	Ö	\	Ñ	Ø
93	]	§	Ü	]	Å	Å	é	¿	Å
94	^	^	^	^	Ü	Ü	^	^	Ü
96	`	`	`	`	é	é	ù	`	é
123	{	é	ä	{	æ	ä	à	¨	æ
124	|	ù	ö	|	ø	ö	ò	ñ	ø
125	}	è	ü	}	à	à	è	}	à
126	~	¨	ß	~	ü	ü	ì	~	ü

Appendix F

MEMORY MAPS



*1: This area is relocated from ROM by the relocator.
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 relocator. Its starting address is predetermined.

*4: CCP starting address should be above 8000H.

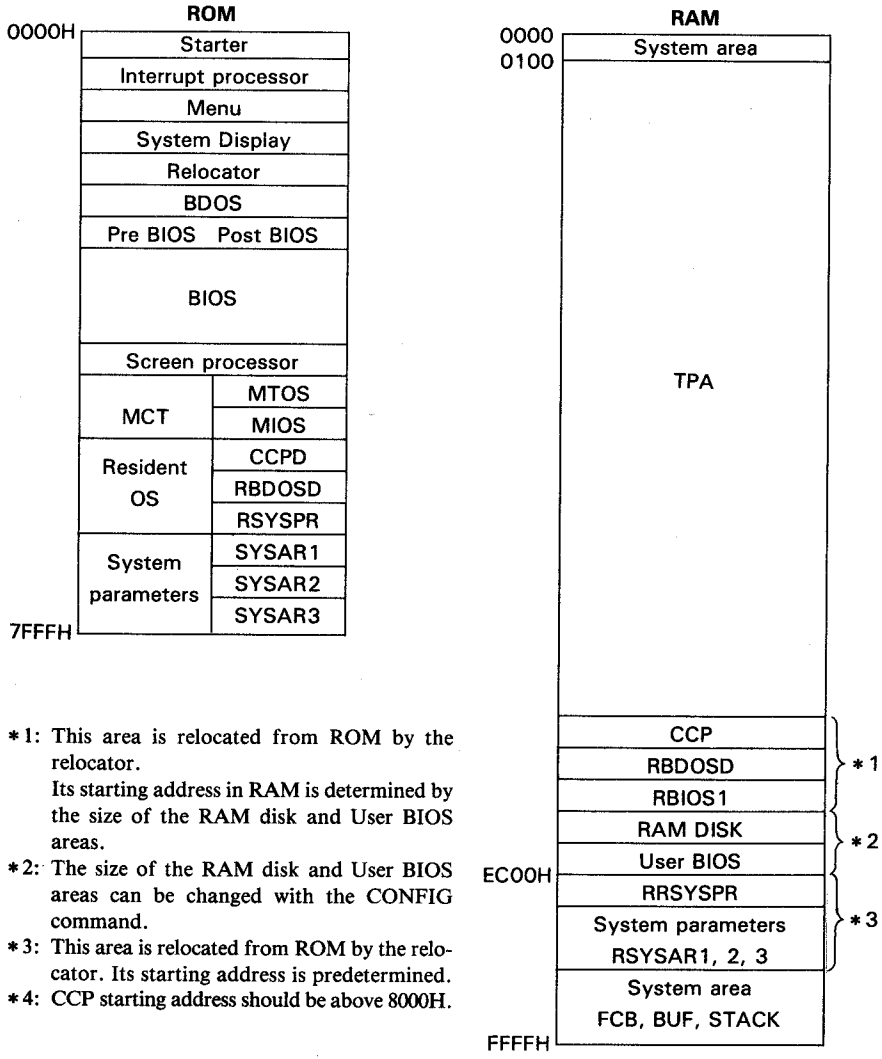
E-5 INTERNATIONAL CHARACTER SETS

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	@	à	§	@	É	É	@	@	É
91	[	°	Ä	[	Æ	Ä	°	¡	Æ
92	\	ç	Ö	\	Ø	Ö	\	Ñ	Ø
93	]	§	Ü	]	Å	Å	é	¿	Å
94	^	^	^	^	Ü	Ü	^	^	Ü
96	`	`	`	`	é	é	ù	`	é
123	{	é	ä	{	æ	ä	à	¨	æ
124	¦	ù	ö	¦	ø	ö	ò	Ñ	ø
125	}	è	ü	}	å	å	è	}	å
126	~	¨	ß	~	ü	ü	ì	~	ü

Appendix F

MEMORY MAPS



*1: This area is relocated from ROM by the relocator.
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.

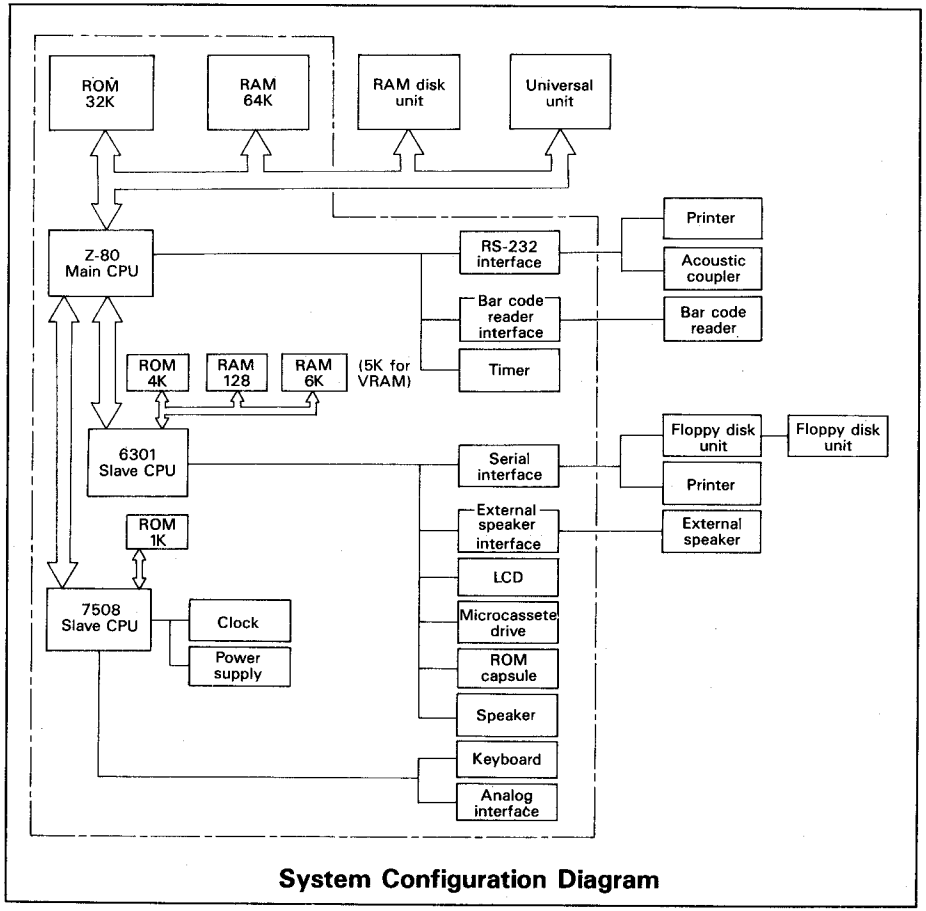
*3: This area is relocated from ROM by the relocator. Its starting address is predetermined.

*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	: μPD7508
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)



System Configuration Diagram

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 be written to locate itself into the correct position before BASIC is loaded.

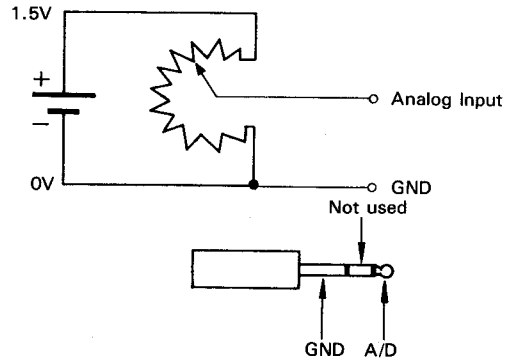
H-1. A simple game illustrating use of the A/D converter.

```
SCORE                HIGH SCORE 0
                                0


                                                    Bat
                                                     ▼
                                                    ▬▬
```

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 "O"; : 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(CH%,ADD%) : ADD% = ADD% * 74/BMAX + 2 : IF ADD%>75 THEN ADD% =
 75
210 PX = ADD%
220 LOCATE PX1,8 : 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 : CH% = 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*78) + 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 18,1 : 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 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 retreiving 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 B400                   LD      DE,DEST         ; Set Destination address
0006' 01 014E                   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 !!                                  *
                        ;*                                                                   *
                        ;*********************************************************************
000E                    SOURCE:
                            .PHASE  0B400H
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      0FFH            ; 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,0FFH          ; 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      0FFH            ; 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.  *
                        ;*                                                                   *
                        ;*********************************************************************
B437                    READVRAM:
B437  3A B54D               LD      A,(COUNT)       ; 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 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 B49B               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.                         *
                        ;*                                                                   *
                        ;*********************************************************************
B44F                    CREATE:
B44F  3E 00                 LD      A,00            ; Zero the first byte after the file type
B451  32 B534               LD      (FCBADR+12),A   ;  in the FCB for safety's sake
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 B0                 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 B4A8               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         *
                        ;*                                                                   *
                        ;*********************************************************************
B49B                    PACKET:                 ; Slave CPU's communications packet
B49B    B4A3                    DW      SNDPKT          ; Address of the packet that will be sent
                                                        ;  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
                                                        ;  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                    DMABUF: 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 4D             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    S

Macros:

Symbols:
0005    BDOS            0001    BIOS            B481    CLOSE
B54D    COUNT           B44F    CREATE          B492    DELETE
B400    DEST            B4A8    DMABUF          B4BA    ERASE
B436    EXIT            B528    FCBADR          014E    LEN
B418    LOOP            B49B    PACKET          B4A7    RCVPKT
B437    READVRAM        F2C9    SCRMODE         F358    SLVFLG
B4A3    SNDPKT          000E    SOURCE          B478    WRITE
B4A4    XCOORD          B4A5    YCOORD
No Fatal error(s)
```

```
MACRO-80 3.44  09-Dec-81      PAGE    1


                        ;*********************************************************************
                        ;*                                                                   *
                        ;*      Program         Load the graphic screen from the currently   *
                        ;*                       logged in drive from 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 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 !!                                  *
                        ;*                                                                   *
                        ;*********************************************************************
000E                    SOURCE:
                                .PHASE  0B200H
B200                    DEST:
B200  21 F2C9                   LD      HL,SCRMODE      ; Get the pointer to the screen mode
B203  7E                        LD      A,(HL)          ;  and find out the screen mode
B204  FE 03                     CP      3               ; Is the screen in graphic mode?
B206  C2 B236                   JP      NZ,EXIT         ; No, so exit from this program
                        ;
                        ;       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      0FFH            ; 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
                        ;
```



```
MACRO-80 3.44  09-Dec-81      PAGE    1-1


B211  3E FF                     LD      A,0FFH          ; 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:
B218  32 B33B                   LD      (COUNT),A
B21B  CD B275                   CALL    READ            ; Get the next block of data from the disk
B21E  FE FF                     CP      0FFH            ; Have we tried to get data that doesn't exist
B220  CA B22E                   JP      Z,EOF           ; Yes, so just exit because there is nothing
                                                        ;  else we can do !
B223  CD B237                   CALL    SETVRAM         ; We've got the next  block so write it to VRAM
B226  3A B33B                   LD      A,(COUNT)       ; 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:
B22E  3E 00                     LD      A,00H           ; Disable communication with the slave CPU
B230  32 F358                   LD      (SLVFLG),A      ;  so that commands & data are ignored
B233  CD B27E                   CALL    CLOSE           ; Close the file.
                        ;
                        ;       It is now safe to exit.
                        ;
B236                    EXIT:
B236  C9                        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.   *
                        ;*                                                                   *
                        ;*********************************************************************
B237                    SETVRAM:
B237  3A B33B                   LD      A,(COUNT)       ; 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
B23A  CB 07                     RLC     A               ; Calculate the Y coordinate of the screen
B23C  32 B291                   LD      (YCOORD),A      ;  block (= data block number *2)
B23F  3E 00                     LD      A,00H           ; Set the X coordinate to 0 i.e. always
B241  32 B290                   LD      (XCOORD),A      ;  start at the left edge of the screen
B244  11 B287                   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 WBOOT
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 0D                 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 up 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:                     ; Slave CPU's communications packet
B287  B28F                  DW    SNDPKT        ; 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
B28F  25                    DB    025H          ; Slave command code that sets the
                                                ;  graphic screens contents.
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             ; Nueber of bytes to be set
                                                ;  in the Y direction
B294  00                    DB    0
B295                DMABUF: DS    128           ; 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.
B315  00                    DB    00H           ; Return code that indicates the success
                                                ;  of the execution of the command
B316                FCBADR:                     ; File control block (FCB) work area
B316  00                    DB    00H           ; Drive code in this case the currently
```

```
MACRO-80 3.44   09-Dec-81      PAGE    1-4


                                        ;  logged in disk
B317   56 52 41 4D                DB      'VRAMDATA'      ; 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..O....6.>
0120 FF 32 5B F3 3E 00 32 4D B5 CD 37 B4 CD 7B B4 FE .2X.>.2M..7..x..
0130 FF CA 8A B4 3A 4D B5 3C FE 20 20 EA 3E 00 32 5B ....: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 9B 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.!4..5.........
0170 05 00 0E 1A 11 A8 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 5B F3 CD 81 B4 ..(.....>.2X....
01A0 0E 13 11 2B 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 2B AF 32 F2 3C 7A B7 C2 +>.2.<..(.2.<z..
01D0 9D 04 3A 11 3E FE 20 C0 7B 3D F8 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 C8 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 C8 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..O....6.>
0120 FF 32 5B 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 5B ..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 87 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 ".!"..#.........
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 8F 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 2B AF 32 F2 3C 7A B7 C2 +>.2.<..(.2.<z..
01D0 9D 04 3A 11 3E FE 20 C0 7B 3D F8 CA 9D 04 FE 10 ..:.>. .{=......
01E0 D2 9D 04 3C 32 F1 3C AF 67 6B C3 34 1B CD C4 28 ...<2.<.gk.4...(
01F0 7A B7 C2 9D 04 3A EC 3C B7 C8 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 dislayed 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 approprite 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 overlayed.
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
	U Underlined label: label used does not exist. V Value error: improperly formed operand encountered in an expression.
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	"x" is one of the symbols described below and c is the command letter being executed when the error occurred. # Search failure. ED cannot find the string specified in an F, S, or N command. ? 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. O The file specified in an R command cannot be found.

Message	Meaning
	> 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. E Command aborted. A keystroke at the console aborted command execution. 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.
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 soom 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 ↵, 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

B