3.4 BIOS Details

This section gives a detailed description of the BIOS functions. See Sections 3.5, 3.6, and 3.7 for details of the keyboard, display, and microcassette.

3.4.1 Programming Notes on the Use of BIOS Functions

(1) Save the contents of registers if necessary because the contents of the registers except those for receiving return parameters are not guaranteed.

(2) The entry to each BIOS function is indicated by 1) the offset from WBOOT and 2) an fixed address.

1. When using the first method, obtain the entry address based on the WBOOT entry address that is stored in 0001H and 0002H. The WBOOT entry address varies depending on the size of the RAM disk or the user BIOS.
2. No consideration is required when using the second method. Keep the RAM map in mind, however, when converting application programs intended for execution under standard CP/M into PINE CP/M.

```
                        ;***********************************************
                        ;       BIOS CALL SAMPLE PROGRAM
                        ;***********************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is consist of only
                        ;               BIOS calling routine.
                        ;               So, this program doesn't run by itself.
                        ;
                        ;       <> assemble condition <>
                        ;
                        ;       .Z80
                        ;
                        ;       <> loading address <>
                        ;
                        ;       .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
0000                    ;RBIOS1         EQU     00000H
EB03                    ;RBIOS2         EQU     0EB03H
                        ;
                        ;***********************************************
                        ;       BIOS CALL ROUTINE
                        ;***********************************************
                        ;
                        ;       NOTE :
                        ;               This routine is used for calling BIOS.
                        ;
                        ;       <> entry parameter <>
                        ;               A    : BIOS function number * 3
                        ;               Depending on each BIOS function.
                        ;       <> return parameter <>
                        ;               Depending on each BIOS function.
                        ;       <> preserved registers <>
                        ;               IY is used by calling address.
                        ;
                        ;       CAUTION :
                        ;               If you use resident BIOS, change
                        ;               RBIOS1 to RBIOS2.
                        ;               If Your program is ROM execute program,
                        ;               you must use RBIOS2.
                        ;
0100                    ;BIOS:
0100    E5              ;       PUSH    HL              ; Save registers.
0101    D5              ;       PUSH    DE              ;
0102    2A 0001         ;       LD      HL,(RBIOS1+1)   ; Get WBOOT entry address.
0105    5F              ;       LD      E,A             ; Function code.
0106    16 00           ;       LD      D,00H           ; Get target BIOS function entry addr.
0108    19              ;       ADD     HL,DE           ;
0109    E5              ;       PUSH    HL              ; Set target address to IY register.
010A    FD E1           ;       POP     IY              ;
010C    D1              ;       POP     DE              ; Restore registers.
010D    E1              ;       POP     HL              ;
010E    FD E9           ;       JP      (IY)            ; Jump to target BIOS function.
                        ;
                        ;       END
```

3.4.2 BIOS Entries

Offset from WBOOT	Entry name	Funciton	Page
−03H	BOOT	Performs a CP/M cold boot.	II-99
±00H	WBOOT	Performs a CP/M warm boot.	II-100
+03H	CONST	Checks for entry from the CON: device	II-101
+06H	CONIN	Inputs one character from the CON : device.	II-101
+09H	CONOUT	Outputs one character to the CON : device.	II-104
+0CH	LIST	Outputs one character to the LST : device.	II-105
+0FH	PUNCH	Outputs one character to the PUN : device.	II-108
+12H	READER	Inputs one character from the RDR : device.	II-109
+15H	HOME	Positions the disk head to track 0.	II-109
+18H	SELDSK	Specifies the drive.	II-110
+1BH	SETTRK	Specifies the track.	II-111
+1EH	SETSEC	Specifies the sector.	II-111
+21H	SETDMA	Specifies the DMA address.	II-112
+24H	READ	Reads the specified data.	II-112
+27H	WRITE	Writes the specified data.	II-113
+2AH	LISTST	Returns the status of the LST:device.	II-114
+2DH	SECTRN	Translates a logical sector to a physical sector.	II-114
+30H	PSET	Performs logical operation on the specified data and VRAM data.	II-115
+33H	SCRNDUMP	Dumps the contents pf VRAM.	II-117
+36H	BEEP	Sounds the loudspeaker.	II-119
+39H	(RSOPEN)		
+3CH	(RSCLOSE)		
+3FH	(RSINST)		
+42H	(RSOUTST)		
+45H	(RSIN)		
+48H	(RSOUT)		

Offset from WBOOT	Entry name	Funciton	Page
+4BH	TIMDAT	Returns time information.	II-124
+4EH	MEMORY	Returns the current bank status.	II-132
+51H	RSIOX	Supports serial communication functions.	II-132
+54H	(LIGHTPEN)		
+57H	MASKI	Mask interrupts.	II-150
+5AH	LOADX	Reads one byte from the specified bank.	II-154
+5DH	STORX	Writes one byte to the specified bank.	II-161
+60H	LDIRX	Transfer data from the specified bank.	II-161
+63H	JUMPX	Jumps to the specified bank.	II-162
+66H	CALLX	Calls the specified bank.	II-162
+69H	GETPFK	Gets a character string defined for a PF key.	II-163
+6CH	PUTPFK	Defines a PF key.	II-166
+6FH	READSW	Read switch settings.	II-168
+72H	(SLAVE)		
+75H	RDVRAM	Reads the contents of the virtual screen.	II-172
+78H	MCMTX	Processes microcassette.	II-174
+7BH	POWEROFF	Turns off system power.	II-175
+7EH	USERBIOS	Gives the entry point to the user BIOS.	II-175
+81H	AUTOST	Specifies an auto start string.	II-176
+84H	RESIDENT	Specifies Resident.	II-179
+87H	CONTINUE	Sets or resets the continue mode.	II-179

Remarks: Functions enclosed in parentheses contain only a RET instruction and do nothing. The USERBIOS function initially contains nothing but a RET instruction.

3.4.3 BIOS Functional Descriptions

This subsection describes individual BIOS functions. The meaning of the items in the description is given below.

Function: Gives a BIOS function outline.

Entry address: The address with which the BIOS function is called. WBOOT denotes the address of the WBOOT routine that is stored in 0001H and 0002H.

Entry parameter: Parameters specified when the function is called.

Return parameter: Parameters returned after the function is executed.

Explanation: Detailed description of the BIOS function.

Note: Programming notes on the BIOS function.

Related functions: Other BIOS functions related to the function.

See also: Sections to be referred to in this manual.

(1) BOOT

Function: Performs a CP/M cold boot.

Entry address: WBOOT - 3H or 0EB00H

Entry parameter: None.

Return parameter: None.

Explanation:

BOOT performs the following:

1. Sets the current drive to A:.
2. Initializes the I/O byte.
3. Displays the CP/M sign-on message.
4. Reads the DIP switches and initializes the keyboard nationality and the character set to be used.
5. Initializes the addresses of the CTRL/HELP (System Display) and CTRL/PF5 (Screen Dump) subroutines.
6. Initializes the data associated with the numerical keypad on the item keyboard.
7. Initializes the pointer to the PF key table.

BOOT performs the following when Resident is not selected:
1. Initializes the data area associated with the PF keys.
2. When no item keyboard is installed, BOOT initializes the item function keys.
3. When an item keyboard is installed, BOOT does the following:
   - Initializes the item keys.
   - Turns off the keyboard auto repeat function.
   - Initializes the drive in which the menu is to be displayed.

BOOT carries out the same steps as WBOOT after the above operations.

Note: BOOT is entered at system initialization, system reset, or 75Ø8 reset. This routine is used not by application programs but by the operating system.

Related function: WBOOT

See also: Section 2.3, "Reset"

(2) WBOOT

Function: Performs a CP/M warm boot.

Entry address: WBOOT +- ØH or ØEBØ3H

Entry parameter: None.

Return parameter: C = Drive No.

Explanation:

WBOOT performs the following:
1. Writes the write data left in the FDD buffer into the floppy disk.
2. Initializes the Micro Cassette parameters.
3. Switches the active screen to the user screen and initializes the cursor.

The following steps are common to BOOT and WBOOT:
1. Closes the serial interface.
2. Initializes the WBOOT and BDOS entry addresses.
3. Loads BDOS into RAM.

WBOOT returns control to the menu or CCP after the above steps of operation.

Note: WBOOT is entered when a JP Ø is executed by the application program to terminate or when power is turned on in the restart mode.

See also: Section 2.4, "Power-on"

(3) CONST

Function: Checks whether any entry is made to the currently assigned CON: device (default is the keyboard).

Entry address: WBOOT + 03H or 0EB06H

Entry parameter: None.

Return parameter: A = 00H: Console input buffer is empty.
A = 0FFH: Data is present in console input buffer.

Explanation:
The current CON: device is determined by the I/O byte. If the current CON: device is the keyboard, CONST returns the no entry state when a switch key, a undefined key, or a key returning no code is pressed.

See also: Section 3.5, "Keyboard"
Section 3.9, "I/O byte"

(4) CONIN

Function: Returns one character read from the currently assigned CON: device (default is the keyboard). If no character is ready, CONIN waits until a character is received.

Entry address: WBOOT + 06H or 0EB09H

Entry parameter: None.

Return parameter: A = Input data
C = Check mode flag (valid only for standard keyboards)

Explanation:
The current CON: device is determined by the I/O byte.

When the console is the keyboard:
- The console is in the sleep mode and the auto power off function is enabled when CONIN is waiting for input data.
- CONIN operates in different modes according to the sate of YPFCMFLG (0F017H). If a PF key is pressed when YPFCMFLG = 00H, CONIN returns the string defined for the PF key. CONIN indicates whether a PF key is pressed when YPFCMFLG = 0FFH.
- The cursor movement keys (especially SHIFT/arrow keys or CTRL/arrow keys) perform special functions on the screen, and CONIN returns no key code when one of them is depressed. If the special functions are disabled, however, CONIN can return the corresponding key code (when a standard keyboard is installed). The special functions of the cursor movement keys are enabled or disabled depending on the state of YSFCMFLG (0F018H).

When YPFCMFLG (ØF017H) = ØØH:
C register = Unpredictable
A register = ØØH ØFEH
(When a PF key is depressed, CONIN returns a character defined for the PF key. See the Key Entry Code Chart for key codes.)

When YPFCMFLG = ØFFH:

1. C register = ØØH
   A register = ØØH ØFEH
   (Indicates that a key other than PF keys is pressed. See the Key Entry Code Chart for key codes.)
2. C register = ØFFH
   A register = ØE0H: Indicates that PF1 is pressed.
   = ØE1H: Indicates that PF2 is pressed.
   = ØE2H: Indicates that PF3 is pressed.
   = ØE3H: Indicates that PF4 is pressed.
   = ØE4H: Indicates that PF5 is pressed.
   = ØE5H: Indicates that PF6 is pressed.
   = ØE6H: Indicates that PF7 is pressed.
   = ØE7H: Indicates that PF8 is pressed.
   = ØE8H: Indicates that PF9 is pressed.
   = ØE9H: Indicates that PF10 is pressed.

When YSFCMFLG (ØF018H) = ØØH:
C register = Unpredictable
A register = ØØH ØFEH
(Codes for arrow keys, SHIFT/arrow keys, and CTRL/arrow keys can be altered. See 3.5.3.8.)

When YSFCMFLG = ØFFH:

1. C register = ØØH
   A register = ØØH ØFEH
   (Indicates that a key other than SHIFT/arrow keys, CTRL/arrow keys, SHIFT/INS, and CTRL/INS is pressed. See the Key Entry Code Chart for key codes.)
2. C register = ØFFH
   A register = ØACH: Indicates that CTRL/← is pressed.
   = ØADH: Indicates that CTRL/→ is pressed.
   = ØAEH: Indicates that SHIFT/← is pressed.
   = ØAFH: Indicates that SHIFT/→ is pressed.
   = ØB0H: Indicates that CTRL/↑ is pressed.
   = ØB1H: Indicates that CTRL/↓ is pressed.
   = ØB2H: Indicates that CTRL/INS is pressed.
   = ØB3H: Indicates that SHIFT/↑ is pressed.
   = ØB4H: Indicates that SHIFT/↓ is pressed.
   = ØB5H: Indicates that SHIFT/INS is pressed.

Reference:

YPFCMFLG (ØF017H) 1 byte
- PF key check mode flag
  = ØØH: PF key check mode is off (initial value).
  = ØFFH: PF key check mode is on.
- This flag is initialized at reset time.
- The functions of this flag is described above.

YSFCMFLG (ØF018H) 1 byte
- Special key check mode flag
  = ØØH: Special key check mode is off (initial value).
  = ØFFH: Special key check mode is on.
- This flag is initialized at reset time.
- The functions of this flag is described above.

See also: Section 3.5, "Keyboard"
section 3.9, "I/O Byte"

(5) CONOUT

Function: Outputs one character to the currently assigned
CON: device (default is the LCD).

Entry address: WBOOT + 09H or 0EB0CH

Entry parameter: C = Output data

Return parameter: None.

Explanation:
The current CON: device is determined by the I/O byte.

The following descriptions apply when the device is the LCD:
- When a code 20H to 0FFH is given, CONOUT displays the corresponding character on the LCD.
- CONOUT handles control codes from 00H to 1FH to control the screen.
- Using the CONOUT routine, the user can instruct the PINE to perform various functions using combinations of ESC (1BH) and subsequent parameters. The CONOUT routine must be called as many times as the number of parameter bytes.
- CONOUT controls not only the LCD but also the keyboard, LED, and buzzer.

See also: Section 3.6, "LCD Display"
Section 3.9, "I/O Byte"

(6) LIST

Function: Outputs one character to the currently assigned LST: device.

Entry address: WBOOT + ØCH or ØEBØFH

Entry parameter: C = Output data

Return parameter: None.

Explanation:

The current LST: is determined by the I/O byte.
If the device is not ready, LIST waits until it is ready.
LIST terminates processing, if the CTRL/STOP key is pressed while waiting for the device to get ready.
If power is turned off or an alarm is generated while LIST is waiting for the device to get ready, LIST terminates processing and performs power-off or alarm processing.
The default value of LST: is determined by DIP switch settings.

Notes:

LIST outputs the command ESC + 'R' + x to establish the printer version when LIST is used for the first time after a WBOOT, when the correspondence between the character font and printer versions is altered, or when the I/O byte is changed.

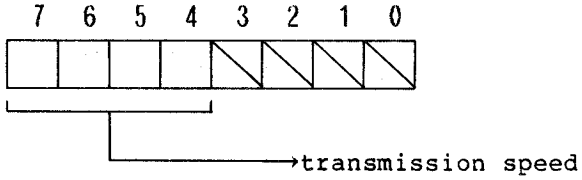
When LIST is directed to the serial port, the communication modes are initialized to 4800 bps, 8 bits, and no parity.

Operating system versions support the cartridge printer. See Section 3.9, "I/O Byte" for details.

See also: Section 3.9, "I/O Byte"
Section 5.8, "Printer"

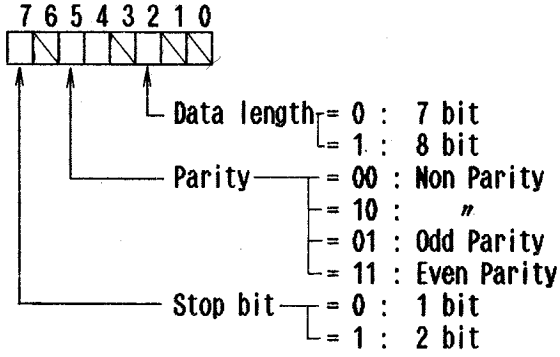
Reference:
The communication modes are specified in the following system areas:

SYSCTLR1 (ØF279H) 1 byte
- Specifies the speed at which data is sent to the printer.



Bit 7	Bit 6	Bit 5	Bit 4	Baud rate (Send)	Baud rate (Receive)
0	0	0	0	110	110
0	0	0	1	150	150
0	0	1	0	300	300
0	0	1	1	600	600
0	1	0	0	1200	1200
0	1	0	1	2400	2400
0	1	1	0	4800	4800
0	1	1	1	9600	9600
1	0	1	0	19200	19200
1	0	1	1	38400	38400
1	0	0	0	1200	75
1	0	0	1	75	1200
1	1			200	200

SYSARTHR (ØF27AH) 1 byte
- Specifies the mode in which data is to be sent to the printer.



PRTFLG (ØF339H) 1 byte
- Printer flag that indicates whether a command specifying the printer language version is to be issued when the logical OR of YLCOUNTRY and the two highest bits of RIOBYTE does not match the value of PRTFLG.

RIOBYTE (ØF529H) 1 byte
- I/O byte save area.
The contents of location ØØØ3H in RAM is copied into this area when a BIOS function is called.

YLCOUNTRY (ØF777H) 1 byte
- Country ID area
Initially loaded with bits 4-1 of the DIP switches. This area is rewritten when the printer language version is altered by BIOS CONOUT.

(7) PUNCH

Function: Outputs one character to the currently assigned PUN: device (default device is the RS-232C interface).

Entry address: WBOOT + ØFH or ØEB12H

Entry parameter: C = Output data

Return parameter: None.

Explanation:
The current PUN: device is determined by the I/O byte.

If the device is not ready, PUNCH waits until it gets ready. PUNCH terminates processing if the CTRL/STOP key is pressed while it is waiting for the device to get ready. If power is turned off or an alarm is generated while waiting for a device ready condition, PUNCH terminates processing and performs power-off or alarm processing.

Notes:
The default serial communication modes are 4800 bps, 8 bits, and no parity.

If the RS-232C interface has been already used by the user (opened with RSIOX), the PUNCH routine takes the parameters that were established when the RS-232C was opened.

If the SIO or cartridge has already been used by the user (opened with RSIOX), the PUNCH routine returns control to the calling program without doing anything.

Reference:
Serial communication modes can be changed by modifying the pertinent parameters that are stored in the following areas.

SRSADR (ØEF31H)
- 2 bytes: Receive buffer address
  Default value: CONBUF (ØFC9BH)
- 2 bytes: Receive buffer size
  Default value: ØØFØH

SRSPAK (ØEF35H)
- 1 byte: Transmission speed
  Default value: ØDH --- 4800 bps
- 1 byte: Bit length
  Default value: Ø3H --- 8 bits
- 1 byte: Parity
  Default value: ØØH --- Non parity
- 1 byte: Stop bit
  Default value: Ø3H --- 2 stop bits.
- 1 byte: Special parameter
  Default value: ØFFH

The above parameters take the same format as the those specified in RSIOX.

Related function: RSIOX

See also: Section 3.9, "I/O Byte"

(8) READER

Function: Inputs one character from the currently assigned RDR: device (default device is the RS-232C interface).

Entry address: WBOOT + 12H or ØEB15H

Entry parameter: None.

Return parameter: A = Input data

Explanation:

The current RDR: device is determined by the I/O byte.

When no input data is present, READER waits until data is received. If power is turned off or an alarm is generated while waiting for input data, READER terminates the current processing and performs power off or alarm processing.

READER always returns 1AH (EOF) if PTR or UR2 (I/O byte items not supported by the PINE) is selected as the RDR: device.

Notes:

The parameter returned when the RS-232C interface is selected as the RDR: device is the same as that returned by RSIOX RSGET.

When RS-232C is specified, READER inputs data in the same modes as PUNCH outputs data.

Related functions: PUNCH, RSIOX

See also: Section 3.9, "I/O Byte"

(9) HOME

Function: Sets the disk seek track to Ø.

Entry address: WBOOT + 15H or ØEB18H

Entry parameter: None.

Return parameter: None.

Explanation:

HOME causes the write data left in the FDD buffer to be written onto the floppy disk. This routine does not actually move the disk head to track Ø.

(10) SELDSK

Function: Specifies the drive to be selected.

Entry address: WBOOT + 18H or 0EB1BH

Entry parameter:
C = Logical drive number.
00H: Drive A Internal/external RAM disk
01H: Drive B ROM capsule 1
02H: Drive C ROM capsule 2
03H: Drive D Floppy disk
04H: Drive E Floppy disk
05H: Drive F Floppy disk
06H: Drive G Floppy disk
07H: Drive H Microcassette
08H: Drive I RAM cartridge
09H: Drive J ROM cartridge 1
0AH: Drive K ROM cartridge 2
E = Access information
Bit 0 = 0: The first disk access after WBOOT.
= 1: Second or subsequent disk access after WBOOT.

Return parameter:
HL = 00H: Parameter error
= Nonzero: Disk parameter block starting address

Explanation:
A parameter error is signaled if this routine is called when no disk drive is connected or installed.

Reference:
PINE OS permits the user to alter the correspondence between logical and physical drives. See Section 3.8, "Disk Storage" for details.

See also: 3.8, "Disk Storage"

(11) SETTRK

Function: Specifies the track for read or write.

Entry address: WBOOT + 1BH or ØEB1EH

Entry parameter: BC = Track number

Return parameter: None.

Explanation:
Since SETTRK makes no parameter check, it reports no error if a truck number beyond the valid range is specified. The error is reported when an actual read or write operation is performed.

Note:
The legal track number ranges for PINE I/O drives are listed below.

Physical drive	Logical drive	Range	Remarks
RAM disk	A	0≤BC≤15	
ROM capsule	B、C	0≤BC≤8	
Floppy disk drive	D、E、F、G	0≤BC≤39	
Micro cassette drive	H	0≤BC≤4	
RAM cartridge	I	0≤BC≤7	
ROM cartridge	J、K	0≤BC≤8	

Maximum value variable

See also: Section 3.8, "Disk Storage"

(12) SETSEC

Function: Specifies the sector for subsequent read or write.

Entry address: WBOOT + 1EH or ØEB21H

Entry parameter: C = sector No.

Return parameter: None.

Explanation:
Valid sector numbers are Ø through 63. Although SETSEC does not check the entry parameter, an error will be signaled when an actual read or write is performed if a sector number beyond the legal range is specified.

See also: Section 3.8, "Disk Storage"

(13) SETDMA

Function: Specifies the starting address of the 128-byte data area for read or write.

Entry address: WBOOT + 21H or ØEB24H

Entry parameter: BC = DMA starting address

Return parameter: None.

Explanation:
SETDMA specifies the starting address of the DMA buffer. The DMA buffer is used for holding input or output data for read or write.

(14) READ

Function: Reads data in 128 byte units.

Entry address: WBOOT + 24H or ØEB27H

Entry parameter: None.

Return parameter: A = Return information
= ØØH: Normal termination.
= Nonzero: Abnormal termination.

Explanation:
READ reads data in 128 byte units based on the parameters specified by SELDSK, SETTRK, SETSEC or SETDMA.

Reference:
One of the following codes is returned if an error occurs during a read or write:

A = ØFAH: Read error
ØFBH: Write error
ØFCH: Select error
ØFDH: Read only disk
ØFEH: Read only file
ØFFH: Read/write error

The following area can be referred to for information about BIOS errors occurring during a read or write:

BIOSERROR (ØF52BH) 1 byte
- BIOS return code
  = ØØ: Normal termination
  = Ø1: Read error
  = Ø2: Write error
  = Ø3: Write protect error
  = Ø4: Time over error
  = Ø5: Seek error (Microcassette drive)
  = Ø6: Break error (Microcassette drive)
  = Ø7: Power off error (Microcassette drive)
  = ØFE: Other errors

Note:
An error is generated if a READ is executed for Microcassette drive (H:). Use MIOS for Microcassette drive operations. (Control will be returned to the calling program with A register loaded with ØFFH.)

See also: Section 3.7, "MTOS/MIOS Operations"
Section 3.8, "Disk Storage"

(15) WRITE

Function: Writes 128-byte data to the disk.

Entry address: WBOOT + 27H or ØEB2AH

Entry parameter: C = Specifies the mode to write.
= ØØH: Standard write (write after blocking).
= Ø1H: Direct write (write immediately without blocking).
= Ø2H: Write to a sequential file.
Return parameter: A = Return information
= ØØH: Normal termination.
= Nonzero: Abnormal termination.

Explanation:
WRITE writes 128-byte data to the disk based on the specified parameters.

Reference: Same as for READ.

Note: Same as for READ.

Related function: READ

(16) LISTST

Function: Returns the status of the currently assigned LST: device.

Entry address: WBOOT + 2AH or ØEB2DH

Entry parameter: None.

Return parameter: A = ØFFH: Ready (sending data on the list device parameter is allowed).
= ØØH: Busy (sending data on the list device is disallowed).

Explanation:
The current list device is determined by the I/O byte.

Reference:
LISTST determines whether the list device is busy or ready by checking the following:

Device	Signal(s) that LISTST checks:
RS-232C	DSR (Data Set Ready).
SIO	SIN (Status signal from the SIO interface).
Parallel	PERR and PBUSY (ERR and Busy signals).
Cartridge	F1 and OBF (Busy signal and output buffer status).
LCD	Nothing because LCD is always ready

See also: Section 3.9, "I/O Byte"

(17) SECTRAN

Function: Translates a logical sector to a physical sector.

Entry address: WBOOT + 2DH or ØEB3ØH

Entry parameter: BC = Logical sector.

Return parameter: HL = Physical sector.

Explanation:
SECTRAN performs no actual translation but only returns the physical sector number that is identical to the logical sector number because logical and physical sector numbers are identical on the PINE.

(18) PSET

Function: Performs a logical operation on the specified data and the VRAM data.

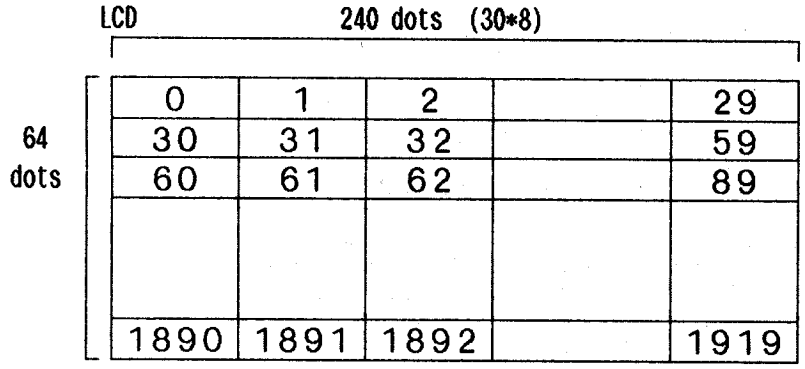
Entry address: WBOOT + 30H or ØEB33H

Entry parameter: B = Data on which the logical operation is to be performed.
C = Logical operation to be performed.
= Ø1H: AND
= Ø2H: OR
= Ø3H: XOR
= Others: PSET does nothing.
HL = VRAM logical address Ø<=HL<=1919

Return parameter: A = ØØH: Normal termination
= Ø1H: HL contains an address beyond the specified range.
C = Operation result (loaded with VRAM data if the entry parameter specified in the C register is other than Ø1H, Ø2H, and Ø3H).

Explanation:
PSET performs the logical operation specified in C on the VRAM data specified in HL and the contents of B, displays the result, and places the result in C. The B and HL registers retain the values specified on entry.

The relationship between VRAM relative addresses and their actual location on the LCD is shown below.



Notes:
PSET only writes the value specified in the B register into VRAM if the C register contains other than Ø1H, Ø2H, and Ø3H.
When reading VRAM data, specify B = ØØH and C = Ø2H or B = ØFFH and C = Ø1H.

```
                        ;       ************************************************
                        ;               BIOS PSET SAMPLE PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is moving VRAM data
                        ;               right by 1 byte.
                        ;
                        ;       <> assemble condition <>
                        ;
                        ;       .Z80
                        ;
                        ;       <> loading address <>
                        ;
                        ;       .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
EB03                    WBOOT   EQU     0EB03H          ; WBOOT entry address
EB33                    PSET    EQU     0EB33H          ; PSET entry address
                        ;
1000                    MAINSP  EQU     01000H          ; Stack pointer
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
0100                    START:
0100    31 1000                 LD      SP,MAINSP       ; Set stack pointer.
0103    06 40                   LD      B,64            ; Set vertical loop counter.
0105    21 0780                 LD      HL,1920         ; Maximum VRAM byte number.
                        ;
0108                    PLOOP1:
0108    C5                      PUSH    BC              ; Save loop counter 1.
0109    06 1D                   LD      B,30-1          ; Set horizontal loop counter.
010B    2B                      DEC     HL              ; Get new destination address.
                        ;
010C                    PLOOP2:
010C    C5                      PUSH    BC              ; Save loop counter 2.
                        ;
010D    0E 01                   LD      C,01H           ; AND function code.
010F    06 00                   LD      B,00H           ; Clear destination data.
0111    CD EB33                 CALL    PSET            ; Write VRAM with 00H.
                        ;
0114    2B                      DEC     HL              ; Get source address.
0115    0E 02                   LD      C,02H           ; OR function code.
0117    06 00                   LD      B,00H           ; Read VRAM data only.
0119    CD EB33                 CALL    PSET            ; Read VRAM data.
                        ;
011C    23                      INC     HL              ; Get destination address.
011D    41                      LD      B,C             ; Move reading data to setting register.
011E    0E 02                   LD      C,02H           ; OR function code.
0120    CD EB33                 CALL    PSET            ; Write VRAM.
                        ;
0123    2B                      DEC     HL              ; Set next destination address.
0124    C1                      POP     BC              ; Restore loop counter 2.
0125    10 E5                   DJNZ    PLOOP2          ; Not 0, then-loop.
                        ;
0127    C1                      POP     BC              ; Restore loop counter 1.
0128    10 DE                   DJNZ    PLOOP1          ; Not 0, then loop.
                        ;
012A    C3 EB03                 JP      WBOOT           ; Program end.
                        ;
                                END
```

(19) SCRNDUMP

Function: Takes a dump of the current VRAM contents.

Entry address: WBOOT + 33H or ØEB36H

Entry parameter: None.

Return parameter: None.

Explanation:

SCRNDUMP checks the I/O byte and dumps (outputs) the VRAM data that is currently displayed on the LCD screen onto the current LST: device. It does nothing but returns control to the calling program if the LST: device is the LCD.

Note:

SCRNDUMP controls the printer using the following procedure:

1. Outputs CR and LF and clears the printer buffer.
2. Outputs ESC + "A" + Ø8H to the printer to set the line spacing.
3. Outputs ESC + "K" + ØFØH + ØØH on the printer to set the number of dots per line.
4. Repeats step 3 for lines 1 through 8.
5. Outputs ESC + "2" on the printer to set the line spacing to 1/6 inch.

SCRNDUMP immediately terminates processing if power is turned off or the CTRL/STOP key is pressed.

SCRNDUMP does not execute normally on the following EPSON printers:

DX-2Ø, DX-1ØØ (daisy wheel printer)
MX-8Ø

SCRNDUMP deletes the cursor from the screen during processing.

Reference:

Screen dump can be inhibited by changing the value in the system area MDMMOD. If SCRNDUMP is called when screen dump is inhibited, it sounds the buzzer and returns control immediately to the calling program.

Abnormal termination (caused when the CTRL/STOP key is pressed, power is turned off, or screen dump is disabled) is reported via the system area LSTERR.

MDMMOD (ØF338H) 1 byte
- Screen dump disable flag
  = ØØH: Enabled (default)
  = Nonzero: Disabled (valid only when RS-232C or SIO is selected.)

LSTERR (ØF773H) 1 byte
- Screen dump return information
  = ØØH: Normal termination
  = ØFFH: Abnormal termination (CTRL/STOP key is pressed, power is turned off, or hard copy is disallowed.)

Related function: LIST

See also: 3.9, "I/O Byte"

(20) BEEP

Function: Sounds the buzzer.

Entry address: WBOOT + 36H or 0EB39H

Entry parameter:

Entry parameters for BEEP can be specified in the following two ways:

1. Specifying the note
   - B = Note (13<=B<=60. BEEP generates no sound when B = 0.)
   - C = Duration (1<=C<=255. The unit is 100 msec.)
2. Specifying the frequency (The MSB of B must be set to 1.)
   - C = Duration (1<=C<=255. The unit is 100 msec.)
   - D = Small loop counter value
   - E = Large loop counter value (BEEP generates no sound when DE = 0.)

Return parameter: A = Return information
- = 00H: Normal termination
- = 0FFH: Abnormal termination (alarm or power-off interrupt occurred, or CTRL/STOP key pressed)

Explanation:

The correspondence between the numbers and notes is shown in the table below. Parenthesized numbers in the table represent the large and small loop counter values, respectively.

When specifying the frequency, load the DE register pair with the large and small loop counter values. The period and frequency to be specified in DE can be calculated using the following formulas:

```
T1 = {13 * (D - 1) + 8} / 3.68 usec.
T2 = {3307 * (E - 1) + 240} / 3.68 usec.
Period  T = 2 * (T1 + T2)
Frequency  f = 1 / T Hz
```

The frequency of the buzzer falls mostly in the range from 200 to 4000 Hz.

	0	1	2	3	4
C	1 (05H, 30H)	13 (03H, 0FH)	25 (01H, FCH)	37 (01H, 75H)	49 (01H, 31H)
C♯	2 (04H, F1H)	14 (02H, EFH)	26 (01H, EDH)	38 (01H, 6DH)	50 (01H, 2DH)
D	3 (04H, B8H)	15 (02H, D2H)	27 (01H, DFH)	39 (01H, 66H)	51 (01H, 2AH)
D♯	4 (04H, 82H)	16 (02H, B7H)	28 (01H, D1H)	40 (01H, 5FH)	52 (01H, 26H)
E	5 (04H, 4FH)	17 (02H, 9DH)	29 (01H, C4H)	41 (01H, 59H)	53 (01H, 23H)
F	6 (04H, 1EH)	18 (02H, 85H)	30 (01H, B8H)	42 (01H, 53H)	54 (01H, 20H)
F♯	7 (03H, EFH)	19 (02H, 6EH)	31 (01H, ADH)	43 (01H, 4DH)	55 (01H, 1DH)
G	8 (03H, C4H)	20 (02H, 59H)	32 (01H, A2H)	44 (01H, 48H)	56 (01H, 1BH)
G♯	9 (03H, 9BH)	21 (02H, 44H)	33 (01H, 98H)	45 (01H, 43H)	57 (01H, 18H)
A	10 (03H, 75H)	* 22 (02H, 31H)	34 (01H, 8FH)	46 (01H, 3EH)	58 (01H, 16H)
A♯	11 (03H, 51H)	23 (02H, 1FH)	35 (01H, 85H)	47 (01H, 39H)	59 (01H, 13H)
E	12 (03H, 2FH)	24 (02H, 0EH)	36 (01H, 7DH)	48 (01H, 35H)	60 (01H, 11H)

The frequency of the note identified by * is 440 Hz.

Notes:

BEEP generates no sound if Ø is specified in the B register as the note. BEEP also generates no sound if it is called with the DE register loaded with Ø in the frequency specified mode.

BEEP immediately terminates processing when the CTRL/STOP key is pressed, or an alarm or power off interrupt is generated.

BEEP stops the cursor from blinking while sounding the buzzer to prevent note fluctuations which would otherwise be caused by OVF interrupts. For similar reasons, other interrupts may also be disabled to avoid sound fluctuations. By default, all interrupts except STOP key interrupts are disabled.

Reference:

Interrupts during BEEP processing can be enabled or disabled by changing the flags in the following area:

```
BPINTEBL (ØFØF5H)  1 byte
- Flags for controlling interrupts during BEEP processing.
     Bit 7: One-second interrupt
          = 1: Disabled.
          = Ø: Unchanged.
     Bit 6: Fixed to Ø.
     Bit 5: Fixed to Ø.
     Bit 4: EXT interrupt
          = 1: Disabled.
          = Ø: Unchanged.
     Bit 3: Fixed to Ø.
     Bit 2: ICF interrupt
          = 1: Disabled.
          = Ø: Unchanged.
     Bit 1: ART interrupt
          = 1: Disabled.
          = Ø: Unchanged.
     Bit Ø: Interrupts other than STOP key interrupts.
          = 1: Disabled.
          = Ø: Unchanged.
OVF interrupts are always disabled.
```

See also: 4.7, "Interrupts"

```
                        ;       ************************************************
                        ;               BIOS BEEP SAMPLE PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is melody of 'White
                        ;                     by using beep.
                        ;
                        ;       <> assemble condition <>
                        ;
                        ;       .Z80
                        ;
                        ;       <> loading address <>
                        ;
                        ;       .PHASE  100H
                        ;
                        ;       <> constant values <>
F0F5                    BPINTEBL        EQU     0F0F5H
                        ;
EB03                    WBOOT           EQU     0EB03H
EB39                    BEEP            EQU     0EB39H
                        ;
1000                    MAINSP          EQU     01000H
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
0100                    START:
0100    31 1000                 LD      SP,MAINSP       ; Set stack pointer.
                        ;
0103    3A F0F5                 LD      A,(BPINTEBL)    ; Get beep interrupt table.
0106    F6 80                   OR      10000000B       ; Disable 1 sec interrupt during beep.
0108    32 F0F5                 LD      (BPINTEBL),A    ; Set new beep interrupt table.
                        ;
010B    21 011E                 LD      HL,SONG         ; Set song data top address
                        ;
010E                    LOOP:
010E    46                      LD      B,(HL)          ; Sound type.
010F    23                      INC     HL              ; Next pointer.
0110    4E                      LD      C,(HL)          ; Sound length.
0111    23                      INC     HL              ; Next pointer.
0112    79                      LD      A,C             ; If sound length is 0,
0113    B7                      OR      A               ;  then end of data.
0114    CA EB03                 JP      Z,WBOOT         ; End of data, then WBOOT.
                        ;
0117    E5                      PUSH    HL              ; Save song table pointer.
0118    CD EB39                 CALL    BEEP            ; Sound.
011B    E1                      POP     HL              ; Restore song table pointer.
011C    18 F0                   JR      LOOP            ; Loop.
                        ;
                        ;       SONG DATA
                        ;
011E                    SONG:
011E    11 02 11 02             DB      17,2, 17,2, 17,2, 17,2, 17,2, 17,2, 17,9
0122    11 02 11 02
0126    11 02 11 02
012A    11 09
012C    11 03 14 08             DB      17,3, 20,8, 17,2, 15,2, 17,9, 17,3, 17,8
0130    11 02 0F 02
0134    11 09 11 03
0138    11 08
013A    14 02 14 02             DB      20,2, 20,2, 20,6, 20,2, 22,2, 20,2, 18,6
013E    14 06 14 02
0142    16 02 14 02
0146    12 06
0148    12 02 12 02             DB      18,2, 18,2, 18,2, 18,6, 00,7
014C    12 02 12 06
0150    00 07
0152    0F 02 0F 02             DB      15,2, 15,2, 15,2, 15,2, 15,2, 15,2, 15,9
0156    0F 02 0F 02
015A    0F 02 0F 02
015E    0F 08
0160    0F 02 0F 02             DB      15,2, 15,2, 17,2, 17,6, 17,2, 17,2, 17,9
0164    11 02 11 06
0168    11 02 11 02
016C    11 09
016E    11 03 11 08             DB      17,3, 17,8, 15,2, 17,2, 15,6, 13,2, 15,2
0172    0F 02 11 02
0176    0F 06 0D 02
017A    0F 02
017C    0D 02 0D 12             DB      13,2, 13,18,00,6
0180    00 06
0182    19 02 19 02             DB      25,2, 25,2, 25,2, 25,2, 25,2, 25,2, 25,8
0186    19 02 19 02
018A    19 02 19 02
018E    19 08
0190    18 02 19 02             DB      24,2, 25,2, 24,3, 22,9, 22,9, 22,3, 24,2
0194    18 03 16 09
0198    16 09 16 03
019C    18 02
019E    18 02 18 02             DB      24,2, 24,2, 24,2, 24,2, 24,2, 24,6, 22,2
01A2    18 02 16 02
01A6    16 02 18 06
01AA    16 02
01AC    18 02 16 08             DB      24,2, 22,8, 20,2, 17,2, 20,12
01B0    14 02 11 02
01B4    14 0C
01B6    19 02 19 02             DB      25,2, 25,2, 25,2, 25,2, 25,2, 25,2, 25,6
01BA    19 02 19 02
01BE    19 02 19 02
01C2    19 08
```

```
01C4    18 02 19 02             DB      24,2, 25,2, 24,3, 22,6, 24,3, 22,8, 22,2
01C8    18 03 16 06
01CC    18 03 16 08
01D0    16 02
01D2    16 02 18 03             DB      22,2, 24,3, 24,8, 22,2, 20,2, 22,8, 20,2
01D6    16 08 16 02
01DA    14 02 16 08
01DE    14 02
01E0    16 02 14 02             DB      22,2, 20,2, 20,16
01E4    14 10
01E6    00 00                   DB      00,0

                                END
```

(21) TIMDAT

Function: Performs a specified clock function.

Entry address: WBOOT + 4BH or ØEB4EH

Explanation:

TIMDAT executes one of the nine clock functions specified in the C register.

C = ØØH: Read time
  = ØFFH: Set time
  = 8ØH: Alarm/wake enable
  = 81H: Alarm/wake disable
  = 82H: Set alarm/wake
  = 83H: (TMDT83 hook)
  = 84H: Read alarm/wake
  = 85H: (TMDT85 hook)
  = 86H: (TMDT86 hook)

TIMDAT will do nothing if it is called with the C register loaded with a value other than the above values.

Use the time descriptor as the parameter when setting or reading the time or the alarm/wake time.

TIMDAT assumes the following clock specifications:
- Leap year processing is performed automatically.
- The time is represented in the 24-hour system.

A detailed description of the TIMDAT functions is given in a later section. The contents of the DE register pair are preserved while the clock function is performed.

See Section 4.3, "Hooks" for the entries of the hook provided for TIMDAT (TMDT83, TMDT85, and TMDT86).

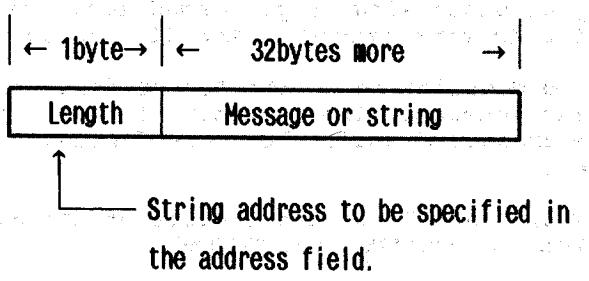
Time descriptor format

The time descriptor is 11 bytes long and consists of the following fields:

Year (lowest two digits) in BCD code: 1 byte, ØØ - 99
Month in BCD code (two digits): 1 byte, Ø1 - 12
Day in BCD code (two digits): 1 byte, Ø1 - 31
Hour in BCD code (two digits): 1 byte, ØØ - 23
Minute in BCD code (two digits): 1 byte, ØØ - 59
Second in BCD code (two digits): 1 byte, ØØ - 59
Day of the week: 1 byte, ØØ - Ø6 (ØØ: Sun., Ø1: Mon, -- Ø6: Sat.)
Type: 1 byte
Address: 2 bytes
Status: 1 byte

Type: Specifies the alarm/wake type.
= ØØH: No specification.
= Ø1H: Sets up the alarm function.
= Ø2H: Sets up the wake function.

Address: Specifies the starting address of the alarm message or the wake string.



Status: Indicates whether an alarm interrupt has been generated.
= ØØH: Not generated.
= Ø1H: Generated.

Reference:
The system uses the following areas to manage alarm/wake data:

ALRMTP (ØEF8AH) 1 byte
- Indicates the alarm/wake type.
  = ØØH: No specification (default).
  = Ø1H: Alarm is specified.
  = Ø2H: Wake is specified.

This area is referenced by the alarm/wake functions and their related processing routines.

ALRMAD (ØEF8BH) 2 bytes
- Contains the alarm/wake message starting address.

ALRMAD always points to ALMMSG (ØF3DFH).

ALRMST (ØEF8DH) 1 byte
- Indicates whether an alarm/wake interrupt has been generated.
  = ØØH: Not generated (default).
  = Ø1H: Generated.

This area is loaded with ØØH when the "Set alarm/wake" or "Read alarm/wake" TIMDAT function is executed or with Ø1H when an alarm/wake interrupt occurs.

ALRMMSG (ØF3DFH) 34 bytes
- Area for saving the alarm/wake message.

The first byte contains the message length. The message specified for "Set alarm/wake" TIMDAT function is saved in this area.

See also: Section 2.9, "Alarm/Wake"
Section 3.4, "Hooks"

(21-1) TIMDAT (1) Read time

Function: Reads the current time.

Entry parameter: C = ØØH
DE = Time descriptor starting address
(7-byte area is required for the time descriptor.)

Return parameter:
The current time is placed in the time descriptor.

Explanation:
TIMDAT (1) loads the first seven bytes of the time descriptor with the current time (the year, month, day, hour, minute, second, and day of the week).

(21-2) TIMDAT (2) Set time

Function: Sets the time.

Entry parameter: C = ØFFH
DE = Time descriptor starting address

Return parameter: None.

Explanation:
TIMDAT (2) sets the time based on the data loaded in the first seven bytes of the time descriptor (year, month, day, hour, minute, second, and day of the week). The BCD digits whose binary state is all 1s retain the previous time values. This means that the time can be partly updated by setting the digits that need not be updated to all 1s.

Note:
Since TIMDAT (2) makes no validity check, the validity of the subsequent clock information is not guaranteed if logically invalid data is specified for this function.

(21-3) TIMDAT (3) Alarm/wake enable

Function: Enables the alarm/wake function.

Entry parameter: C = 8ØH: Alarm/Wake enable

Return parameter: None.

Explanation:
Alarm/wake interrupts are generated at the specified times once TIMDAT (3) is executed.

The system automatically enables the alarm/wake function when the Set alarm/wake function is executed.

(21-4) TIMDAT (4) Alarm/wake disable

Function: Disables the alarm/wake function.

Entry parameter: C = 81H

Return parameter: None.

Explanation:
No alarm/wake interrupt occurs once TIMDAT (4) is executed.

Since the alarm/wake data is preserved even after this function is executed, it is again made valid when the Alarm/wake enable function is executed. However, no alarm/wake interrupt is generated while the alarm/wake function is held disabled.

(21-5) TIMDAT (5) Set alarm/wake

Function: Specifies the alarm/wake time.

Entry parameter: C = 82H
DE = Time descriptor starting address

Return parameter: None.

Explanation:
TIMDAT (5) must be called after filling the year to address fields in the time descriptor. The year field and the value in the unit place in the second field are ignored, however (the minimum unit of set time is 10 seconds).

Any BCD digit which is set to 0FH (four bits are all set to 1) in the fields from the month to the day of the week are regarded as matching any time value. For example, the alarm/wake function will be invoked at the specified time every day if the month, day, and day of the week are set to all 1s.

Once this function is executed, the alarm/wake function is enabled automatically.

Note: Since TIMDAT (5) makes no parameter check, normal alarm/wake processing cannot be guaranteed if invalid data is specified. Especially, data other than 00H to 02H must not be specified in the type field.

(21-6) TIMDAT (6) Read alarm/wake

Function: Reads the alarm/wake time.

Entry parameter: C = 84H
DE = Time descriptor starting address
(11-byte area is required for the time descriptor.)

Return parameter: The alarm/wake status is placed in the time descriptor.

Explanation:
The current alarm/wake settings are loaded into the time descriptor after TIMDAT (6) is executed. The year field and the first digit of the second field are always set to all 0FFH and 0FH, respectively. This is because they are never set by TIMDAT (5).

```
                        ;       ************************************************
                        ;               BIOS TIMDAT SAMPLE PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is reading clock,
                        ;                and displaying the time.
                        ;
                        ;       <> assemble condition <>
                        ;
                                .Z80
                        ;
                        ;       <> loading address <>
                        ;
                                .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
EB03                    WBOOT           EQU     0EB03H  ; WBOOT entry address.
EB06                    CONST           EQU     0EB06H  ; CONST entry address.
EB09                    CONIN           EQU     0EB09H  ; CONIN entry address.
EB0C                    CONOUT          EQU     0EB0CH  ; CONOUT entry address.
EB4E                    TIMDAT          EQU     0EB4EH  ; TIMDAT entry address.
                        ;
1000                    MAINSP          EQU     01000H  ; Stack pointer.
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Display time until press BREAK key.
                        ;
0100                    START:
0100    31 1000                 LD      SP,MAINSP       ; Set stack pointer.
                        ;
0103    21 0191                 LD      HL,CUSROFF      ; Cursor off data.
0106    CD 013D                 CALL    DSPMSG          ; Cursor off.
                        ;
0109    21 0197                 LD      HL,MSG01        ; Date & time message.
010C    CD 013D                 CALL    DSPMSG          ; Dispay message.
                        ;
010F                    LOOP:
010F    CD EB06                 CALL    CONST           ; Key in check.
0112    3C                      INC     A               ; Input any key?
0113    20 07                   JR      NZ,SKIP         ; No.
0115    CD EB09                 CALL    CONIN           ; Get inputed key.
0118    FE 03                   CP      03H             ; BREAK key?
011A    28 18                   JR      Z,TIMEEND       ; Yes.
011C                    SKIP:
011C    11 01E6                 LD      DE,NTIME        ; Time discrepter.
011F    0E 00                   LD      C,00H           ; Read time function.
0121    CD EB4E                 CALL    TIMDAT          ; Read time.
                        ;
0124    CD 0149                 CALL    TIMECHK         ; New & old time compare.
0127    28 E6                   JR      Z,LOOP          ; If same, then loop.
                        ;
0129    CD 0159                 CALL    TIMESET         ; Set new time data
012C    21 01CD                 LD      HL,MSG02        ; Display time data
012F    CD 013D                 CALL    DSPMSG          ;
0132    18 DB                   JR      LOOP            ; Loop
                        ;
0134                    TIMEEND:
0134    21 0194                 LD      HL,CUSRON       ; Cursor on data.
0137    CD 013D                 CALL    DSPMSG          ; Cursor on
013A    C3 EB03                 JP      WBOOT           ; Jump WBOOT.
                        ;
                        ;       ************************************************
                        ;               DISPLAY MESSAGE UNTIL FIND 0
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               HL : Message data top address.
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
013D                    DSPMSG:
013D    7E                      LD      A,(HL)          ; Get message data.
013E    B7                      OR      A               ; End mark?
013F    C8                      RET     Z               ; Yes, then return.
                        ;
0140    4F                      LD      C,A             ; Set display data to c reg.
0141    E5                      PUSH    HL              ; Save message pointer.
0142    CD EB0C                 CALL    CONOUT          ; Display message.
0145    E1                      POP     HL              ; Restore message pointer.
0146    23                      INC     HL              ; Pointer update.
0147    18 F4                   JR      DSPMSG          ; Loop until find 0.
                        ;
                        ;       ************************************************
                        ;               CHECK OLD & NEW TIME
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       return parameter <>
                        ;               ZF : Return ifomation
                        ;                       =1 : New time is same as old one.
                        ;                       =0 : New time is different from old one.
                        ;       <> preserved registers <>
                        ;               NON
```

```
0149                    ;
                        TIMECHK:
0149    21 01E6                 LD      HL,NTIME        ; New time data.
014C    11 01ED                 LD      DE,OTIME        ; Old time data.
014F    06 06                   LD      B,06H           ; Data counter.
                        ;
0151                    TLOOP:
0151    1A                      LD      A,(DE)          ; Get old time data.
0152    BE                      CP      (HL)            ; Compare it with new one.
0153    C0                      RET     NZ              ; If disagree, then return.
0154    13                      INC     DE              ; Poninters update.
0155    23                      INC     HL              ;
0156    10 F9                   DJNZ    TLOOP           ; Loop 6 times.
0158    C9                      RET                     ;
                        ;
                        ;       ************************************************
                        ;               SET TIME DATA
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
0159                    TIMESET:
0159    21 01E6                 LD      HL,NTIME        ; Set time data to old time area.
015C    11 01ED                 LD      DE,OTIME        ;
015F    01 0006                 LD      BC,6            ; Year/month/date/hour/minuite/second
0162    ED B0                   LDIR                    ; Move new data to old area.
                        ;
0164    21 01E6                 LD      HL,NTIME        ; Set BCD data to message area with ASCII.
0167    11 01D1                 LD      DE,DATE         ; HL is source, DE is destination,
016A    06 03                   LD      B,03H           ; B is counter.
016C                    SET10:
016C    CD 0180                 CALL    SETASCII        ; Convert BCD to ASCII.
016F    23                      INC     HL              ; Pointer update.
0170    13                      INC     DE              ;
0171    10 F9                   DJNZ    SET10           ; Loop 3 times. (Year/month/date)
                        ;
0173    11 01DD                 LD      DE,TIME         ; Time date setting area.
0176    06 03                   LD      B,03H           ;
0178                    SET20:
0178    CD 0180                 CALL    SETASCII        ; Convert BCD to ASCII.
017B    23                      INC     HL              ; Pointer update.
017C    13                      INC     DE              ;
017D    10 F9                   DJNZ    SET20           ; Loop 3 times. (Hour/minuite/second)
017F    C9                      RET                     ;
                        ;       ************************************************
                        ;               SET ASCII DATA FROM BCD DATA
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               HL : BCD data address.
                        ;               DE : ASCII data setting address.
                        ;       <> return parameter <>
                        ;               DE : Entry DE + 2
                        ;       <> preserved registers <>
                        ;               HL
                        ;
0180                    SETASCII:
0180    7E                      LD      A,(HL)          ; Get BCD data.
0181    F5                      PUSH    AF              ; Save BCD data.
0182    0F                      RRCA                    ; Move MSB 4 bit to LSB 4bit.
0183    0F                      RRCA                    ;
0184    0F                      RRCA                    ;
0185    0F                      RRCA                    ;
0186    CD 018A                 CALL    NEXT            ; Set ASCII data by 1 byte.
0189    F1                      POP     AF              ; Restore BCD data.
018A                    NEXT:
018A    E6 0F                   AND     0FH             ; Check LSB 4 bit.
018C    C6 30                   ADD     A,30H           ; Change to ASCII data.
018E    12                      LD      (DE),A          ; Set ASCII data.
018F    13                      INC     DE              ; Setting pointer update.
0190    C9                      RET                     ;
                        ;
                        ;
0191                    CUSROFF:
0191    1B 32 00                DB      1BH,'2',00H             ; Cursor off data.
0194                    CUSRON:
0194    1B 33 00                DB      1BH,'3',00H             ; Cursor on data.
                        ;
0197                    MSG01:
0197    0C                      DB      0CH
0198    50 72 65 73             DB      'Present date is         .',0DH,0AH
019C    65 6E 74 20
01A0    64 61 74 65
01A4    20 69 73 20
01A8    20 20 20 20
01AC    20 20 20 20
01B0    2E 0D 0A
01B3    50 72 65 73             DB      'Present time is
01B7    65 6E 74 20
01BB    74 69 6D 65
01BF    20 69 73 20
01C3    20 20 20 20
01C7    20 20 20 20
01CB    2E
01CC    00                      DB      00H
01CD                    MSG02:
01CD    1B 3D 20 30             DB      1BH,'=',20H,30H         ; Direct cursor.
```

```
01D1                            DATE:
01D1    30 30 2F 30                     DB      '00/00/00'
01D5    30 2F 30 30
01D9    1B 3D 21 30                     DB      1BH,'='.21H.30H         ; Direct cursor.
01DD                            TIME:
01DD    30 30 3A 30                     DB      '00:00:00'
01E1    30 3A 30 30
01E5    00                              DB      00H
                                ;
                                ;
                                ;
01E6                            NTIME:
01E6                                    DS      7
01ED                            OTIME:
01ED                                    DS      7
                                :
                                        END
```

(22) MEMORY

Function: Checks the current bank information.

Entry address: WBOOT + 4EH or ØEB51H

Entry parameter: None.

Return parameter: C = Bank information
= ØFFH: System bank
= ØØH: Bank Ø (all RAM)
= Ø1H: Bank 1 (ROM capsule 1)
= Ø2H: Bank 2 (ROM capsule 2)

Explanation:
The user can identify the bank on which his application program is currently executing by calling this function from that program.

See also: Section 4.4, "Bank Switching"

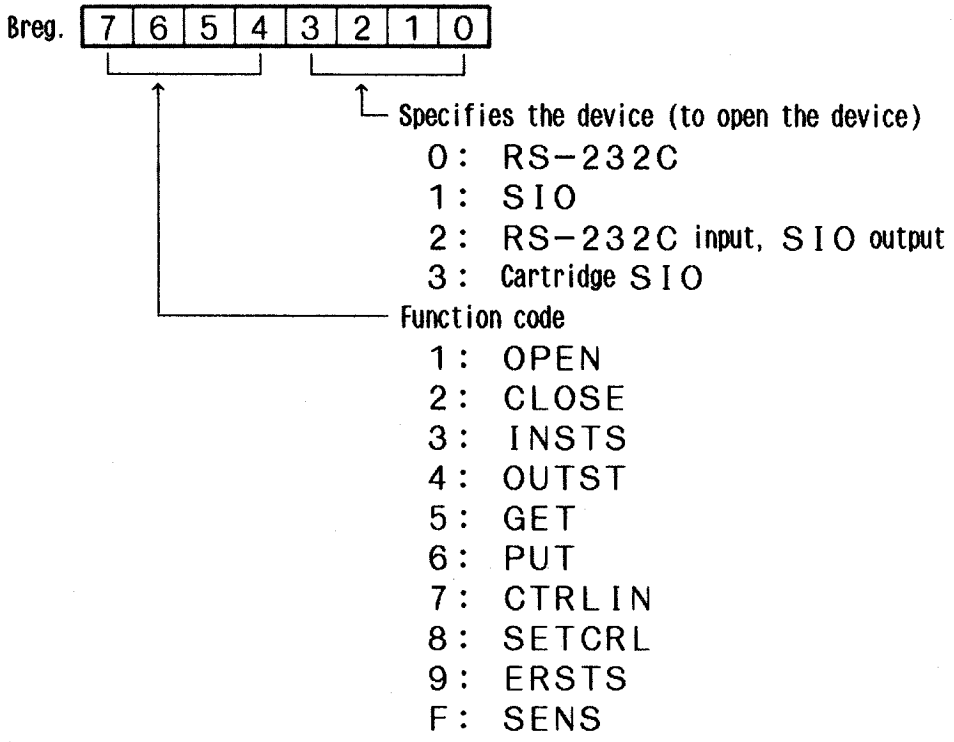
(23) RSIOX

Function: Performs a specified serial communication function.

Entry address: WBOOT + 51H or ØEB54H

Explanation:
RSIOX is a BIOS routine for handling serial communication which supports the RS-232C interface, SIO, or cartridge SIO.

The B register specifies the serial device and the function to be performed on that device. Its format is shown below.



The PINE OS supports three types of serial communication interfaces: RS-232C, SIO, and cartridge SIO. The user can use only one device at a time. To solve this inconvenience, the OS provides various facilities which help the user make effective use of the serial devices. See Section 5.2, "Serial Interfaces" for details.

Notes:

The system only controls the switching of the serial communication mode. For example, when the cartridge SIO is specified as the communication device, the system only opens the cartridge SIO and does not change its operating mode (DB, IO, HS, or OT). See Section 5.1, "Cartridges" for further information.

The definition of the terms used here are given below:

Serial: General term for RS-232C, SIO, and cartridge SIO communications.
RS-232C: Communication via the RS-232C connector.
SIO: Communication via the serial connector.
Cartridge SIO: Communication via the cartridge connector.

Reference:

RSIOX uses the system areas shown below:

RSXON (ØEF2CH) 1 byte
- Loaded with the XON code when XON/XOFF is specified. The initial value is 13H.

RSXOFF (ØEF2DH) 1 byte
- Loaded with the XOFF code when XON/XOFF is specified. The initial value is 1ØH.

RSPSTS (ØF787H) 1 byte
- RSIOX status flags.
  - Bit 7: Enables or disables the DSR line (= Ø: Enable)
  - 6: Indicates whether a framing error occurred (= 1: Framing error occurred)
  - 5: Indicates whether a receive overrun error occurred (=1: Occurred)
  - 4: Indicates whether a parity error occurred (= 1: Occurred)
  - 3: Enables or disables CD control (=1: Enable)
  - 2: Indicates whether a receive buffer overflow occurred (= 1: Occurred)
  - 1: Indicates whether the receive buffer is full (=1: Full)
  - Ø: Indicates whether the device is open (= 1: Open)

RSPRBGP (ØF788H) 2 bytes
- Receive buffer get pointer

RSPRBPP (ØF78AH) 2 bytes
- Receive buffer put pointer

RSPRBAD (ØF78CH) 2 bytes
- Receive buffer starting address

RSPRBSZ (ØF78EH) 2 bytes
- Receive buffer size

RSPBITR (ØF79ØH) 1 byte
- Transmission speed (parameter specified at open time)

Ø2H: 11Ø bps	ØAH: 12ØØ bps	1ØH: 38400 bps
Ø4H: 15Ø	ØCH: 24ØØ	8ØH: 75/12ØØ
Ø5H: 2ØØ	ØDH: 48ØØ	81H: 12ØØ/75
Ø6H: 3ØØ	ØEH: 96ØØ	
Ø8H: 6ØØ	ØFH: 192ØØ	

RSPBITL (ØF791H) 1 byte
- Bit length (parameter specified at open time)

Ø2H: 7 bits Ø3H: 8 bits

RSPPAR (ØF792H) 1 byte
- Parity (parameter specified at open time)

ØØH: No parity ØlH: Odd parity Ø3H: Even parity

RSPSTOPB (ØF793H) 1 byte
- Stop bit (parameter specified at open time)

ØlH: 1 bit Ø3H: 2 bits

```
RSPSPP (ØF794H)  1 byte
- Special parameter (parameter specified at open time)
     Bits 7 - 5: Not used.
     Bit 4: Enables or disables XON/XOFF control (= Ø: Enable)
         3: Not used.
         2: Enables or disables SI/SO control (=Ø: Enable)
         1: Enables or disables RTS control (=1: Enable)
         Ø: Enables or disables DTR control (=1: Enable)

RSPBEAD (ØF795H)  2 bytes
- Receive buffer end address + 1

RSXONSZ (ØF797H)  2 bytes
- Number of receive data bytes when an XON code is sent
(Receive buffer size / 4)

RSXOFSZ (ØF799H)  2 bytes
- Number of receive data bytes beyond which an XOFF code is
sent
(Receive buffer size * 3 / 4)

CHRMSK (ØF79BH)  1 byte
- Receive data mask pattern
     7FH: For 7-bit data      ØFFH: For 8-bit data

RSSSPP (ØF79CH)  1 byte
- Special parameter for system processing
     Bit 7: Not used.
         6: XOFF receive flag (= 1: Received)
         5: XOFF send flag (=1: Send)
         4: Enables or disables XON/XOFF control (=1: Enable)
         3: Not used.
         2: Enables or disables SI/SO control (= 1: Enable)
         1: Enables or disables RTS control (=1: Enable)
         Ø: Enables or disables DTR control (=1: Enable)

RSPAKAD (ØF79DH)  2 bytes
- RSIOX parameter packet address
(The value of the RSIOX parameter specified in the HL
register.)

RSRDL (ØF79FH)  2 bytes
- Number of data stored in the receive buffer

SISOCNT (ØF7A1H)  2 bytes
- Number of SI or SO codes in RSRDL
Actual data count is (RSRDL - SISOCNT)

SSXMODE (ØF7A3H)  1 byte
- SI/SO send mode
     = ØØH: SI send mode
     = Ø1H: SO send mode
     = Ø2H: SI/SO does not send mode (initial setting)

RSXMODE (ØF7A4H)  1 byte
- SI/SO receive mode
     = ØØH: SI receive mode (initial setting)
     = 8ØH: SO receive mode

RSFDEV (ØF7A5H)  1 byte
- The four lowest order bits of the RSIOX parameter in the B
register
```

RSODEV (ØF7A6H) 1 byte
- Device mode
    = ØØH: RS232C
    = Ø1H: SIO
    = Ø2H: RS232C input, SIO output
    = Ø3H: Cartridge
    = ØFFH: Not opened

(23-1) RSIOX OPEN

Function: Opens the device to be used.

Entry parameter: B = 1XH (X = Ø - 3)
HL = Parameter block starting address
(9 bytes must be reserved. Detailed description is given later.)

Return parameter: A = Return information
= ØØH: Normal termination (Z flag = 1)
= Ø2H: Already open (Z flag = Ø)
HL = Return information block starting address
(Holds the value specified on entry. Detailed description is give later.)

Explanation:
RSIOX OPEN selects the given device for serial communication, specifies the communication mode based on the conditions set up in the specified parameter block, and enables serial interrupts to ready the device for communication.

Parameter block structure

The parameter block format is shown below.

(HL) →	Receive buffer address (two bytes)
+1	
+2	Receive buffer size
+3	
+4	Baud rate
+5	Bit length
+6	Parity
+7	Stop bits
+8	Special parameter

(1) Receive buffer starting address
Specifies the starting address of the receive buffer.

(2) Receive buffer size
Specifies the size of the receive buffer.

(3) Baud rate
Specifies the baud rate. The table below lists the codes that correspond to the available baud rates.

Code	Baud rate (send)	Baud rate (receive)
02H	110 bps	110 bps
04H	150	150
05H	200	200
06H	300	300
08H	600	600
0AH	1200	1200
0CH	2400	2400
0DH	4800	4800
0EH	9600	9600
0FH	19200	19200
10H	38400	38400
80H	75	1200
81H	1200	75

(4) Bit length
Specifies the character length in bits.
02H --- 7 bits/character
03H --- 8 bits/character

(5) Parity
Specifies the type of parity checking.
00H --- No parity
01H --- Odd
03H --- Even

(6) Stop bits
Specifies the number of stop bits.
01H --- 1 bit
03H --- 2 bits

(7) Special parameter
Specifies the communication modes and status on a bit basis.

Bit	Description
0	Enables or disables DTR (Data Terminal Ready) control. 0: Disable 1: Enable
1	Enables or disables RTS (Request to Send) control. 0: Disable 1: Enable
2	Enables or disables SI/SO (Shift In/Shift Out) control. 0: Enable 1: Disable
3	Not used.
4	Enables or disables XON/XOFF control 0: Enable 1: Disable
5 - 7	Not used.

DTR is valid only when RS-232C or SIO is specified. RTS is valid only when RS-232C is specified.

Parameter block contents on return

The contents of the parameter block are changed as follows on exit from this function:

(HL) →	Status flag (One byte)
+1	Receive buffer get point (two bytes)
+2	
+3	Receive buffer put point (two bytes)
+4	
+5	Receive buffer address (two bytes)
+6	
+7	Receive buffer size (two bytes)
+8	

(1) Status flags
Indicate the serial interface status on a bit basis.

Bit	Description
Ø	Indicates whether the interface is open. Ø: Not open. 1: Open.
1	Indicates whether the receive buffer is full. Ø: Not full. 1: Full.
2	Indicates whether a receive buffer overflow occurred. Ø: No overflow occurred. 1: Overflow occurred.
3	Indicates the state of the CD (Carrier Detect) line. Ø: Not active. 1: Active.
4	Indicates whether a parity error occurred. Ø: No parity error occurred. 1: Parity error occurred.
5	Indicates whether an overrun error occurred. Ø: No overrun error occurred. 1: Overrun error occurred.
6	Indicates whether a framing error occurred. Ø: No framing error occurred. 1: Framing error occurred.
7	Indicates state of the DSR (Data Set Ready) line. Ø: Active. 1: Not active.

CD is valid only when RS-232C is specified. DSR is valid only when RS-232C or SIO is specified. When SIO is specified, the SIN line is used instead of the DSR line.

Once an error occurs, bits 2, 4, 5, and 6 hold the error status until the ERSTS function is executed.