Step 1: Check header.
Take the sum of the bytes between ØCBFØH through ØCBFFH to verify whether it amounts to ØØH. If the check proves normal, examine the 2-byte field at ØCBFØH to determine whether it contains 'UB'.
Step 2: Correct (own) routine?
Check the routine name in the header to see whether the loading process can be bypassed (the own routine need not be reloaded). This check may be omitted.
Step 3: Check overwrite flag.
Test the overwrite flag in the header to determine whether the existing routine can be overwritten.
Step 4: Check user BIOS area size.
Perform the step described in 4.1.3 (2) to reserve the user BIOS area.
Step 5: Release processing.
Call the release processing routine whose release address is stored in the header.
Step 6: Load user BIOS routine.
Load the new routine into the user BIOS area and create a new header.

4.2 Jump Tables

4.2.1 General

The PINE provides jump tables that allow application programs to directly call system-supplied routines.

There are three type of jump tables:
1) Resident jump table
2) Hook jump table
3) OS ROM jump table

This section describes the resident and OS ROM jump tables. See 4.3, "Hooks" for a detailed description of the hook jump table.

4.2.2 Organization

(1) Resident jump table organization

The resident jump table is reserved at locations FF90H through FFBFH and can be called from any bank.

Figure 4.2.1 shows the organization of the resident jump table. Each routine in the resident jump table is described in Subsection 4.2.3.

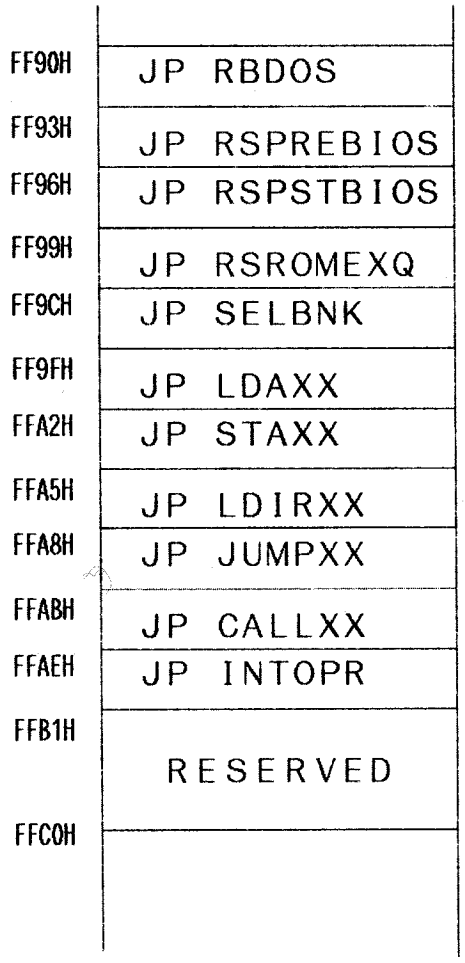


Figure 4.2.1 Resident Jump Table Organization

(2) OS ROM jump table organization

Figure 4.2.2 shows the organization of the OS ROM jump table. The OS ROM jump table is reserved at the beginning of OS ROM (system bank).
The OS must switch the active bank to the system bank to call a routine from the OS ROM jump table.

1) When calling from an application program

The application program must use BIOS CALLX (WBOOT + 66H) to directly call a routine in OS ROM.

2) When calling from extended BIOS or interrupt processing

In modules that cannot call any BIOS routines (i.e. extended BIOS or interrupt processing), the OS must switch the active bank to the system bank using CALLXX (FFABH) or SELBNK (FF9CH) via the resident jump table to directly call the routine in OS ROM.

See Subsection 4.2.4 for a description of each routine in the OS ROM jump table.

0000H	NOP (00H)
0001H	(JR OSTART)
0003H	JP BDOSTABL
0006H	JP BIOSTABL
0009H	JP MTOSTABL
000CH	JP MIOSTABL
000FH	JP GOBACK
0012H	JP SETERR
0015H	JP RSTERR
0018H	JP CALADRS
001BH	JP BIOSJTLD
001EH	JP RAMDKMNT
0021H	JP MDFDPB
0024H	JP RAMCRFMT
0027H	JP XMAKDIR
002AH	JP XMOUNT
002DH	JP XREMOVE
0030H	JP EPSPSND
0033H	JP EPSPRCV
0036H	JP XREDSP
0039H	JP XFONTGET
003CH	JP XUSRSCRN
003FH	JP XSYSSCRN
0042H	JP XSETCUSR
0045H	JP XWTVRAM
0048H	JP MELODY
004BH	JP WRT7508
004EH	JP CMD7508
0051H	JP HSRST
0054H	JP HSSCOM
0057H	JP HSSDAT
005AH	JP HSRCV
005DH	JP HSSBL
0060H	JP HSRBL
0063H	JP HSBRK
0066H	JP CHKMOD

Figure 4.2.2 OS ROM Jump Table Organization

Note: ØSTART is used by the system at power-on and reset time.

4.2.3 Resident Jump Table

The resident jump table includes eleven routines. This section describes how to specify each routine.

SELBNK, LDAXX, STAXX, LDIRXX, JUMPXX, CALLXX and INTOPR are used in modules, such as extended interrupt processing and extended BIOS functions, which are to control bank switching and interrupt processing.

(1) RBDOS (FF90H)

Function: Indicates the entry address of RBDOS2.

Entry parameters: Same as individual BDOS functions.

Return parameters: Same as individual BDOS functions.

Explanation:

1) The PINE provides two BDOS entry points in RAM, for the following reasons:
   - To indicate the upper limit of available RAM, so that ordinary CP/M application programs can operate normally.
   - To enable ROM-based programs to call BDOS functions without being aware of bank switching.
2) RBDOS points to the entry address of RBDOS2 in the common system area.
3) When BDOS is used by the other programs, ROM-based programs can call RBDOS (FF90H) in place of location 0005H.
4) See Section 3.2, "BDOS Operations" for BDOS functions.
5) Figure 4.2.3 shows the relationship between RBDOS1 and RBDOS2.

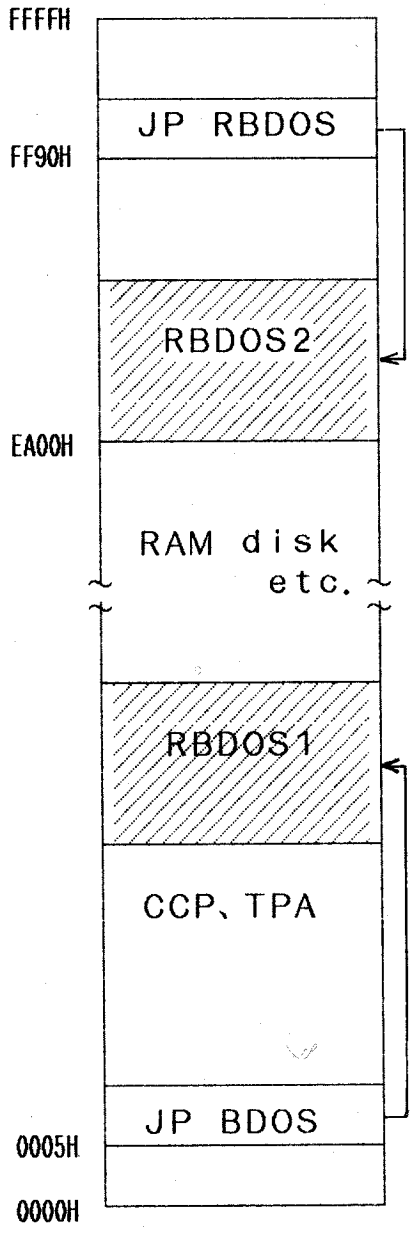


Figure 4.2.3 Relationship between RBDOS1 and RBDOS2

(2) RSPREBIOS (ØFF93H), RSPSTBIOS (ØFF96H)

Function: RSPREBIOS sets the BIOS in-process flag to disable the alarm and power-off functions.
RSPSTBIOS processes any alarm or power-off conditions which occurred following the execution of RSPREBIOS, and resets the BIOS in-process flag.

Entry parameter: None

Return parameter: None

Explanation:

1) All registers retain the previous values.
2) PREBIOS and PSTBIOS are called by OS during BIOS processing and disable and enable the alarm and power-off functions.
3) RSPREBIOS and RSPSTBIOS are supplied so that the user can use the PREBIOS and PSTBIOS functions in application programs.
4) RSPREBIOS and RSPSTBIOS may be called by application programs to:
   1. Disable the power-off function throughout the execution of the program.
   2. Suppress the display of the alarm screen throughout the execution of the program.
   3. Disable power failure conditions throughout the execution of the program.
   4. Allow the application program to directly call BIOS in OS ROM.

Notes:

1) If the BIOS stack has previously been used, do not call either RSPREBIOS or RSPSTBIOS because either routine switches the active stack to the BIOS stack.
2) Carry out RSPSTBIOS after the execution of RSPREBIOS. If RSPSTBIOS is not executed following RSPREBIOS, power-off or alarm processing is held pending.
3) If a call is made to BIOS after the execution of RSPREBIOS, PSTBIOS is automatically carried out at the end of the BIOS processing. Subsequently, alarm or power-off processing starts immediately when such a condition occurs. Directly call BIOS in OS ROM when calling a BIOS function after RSPREBIOS.
4) In the application program, if the processing time between the execution of RSPREBIOS and RSPSTBIOS is too long, the user must divide the application to shorten the processing time. This prevents the main power from being automatically turned off when the user fails to turn off power within 5Ø seconds of a power failure.

RSPREBIOS and RSPSTBIOS processing flowcharts are shown below.

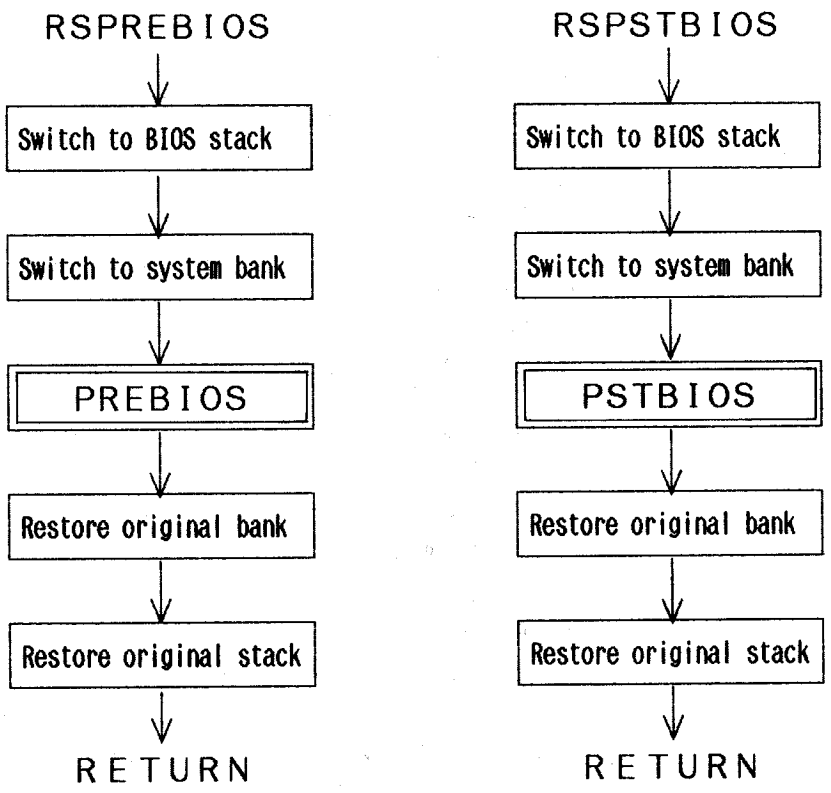


Figure 4.2.4 RSPREBIOS/RSPSTBIOS Processing Flow

PREBIOS and PSTBIOS processing flowcharts are shown below.

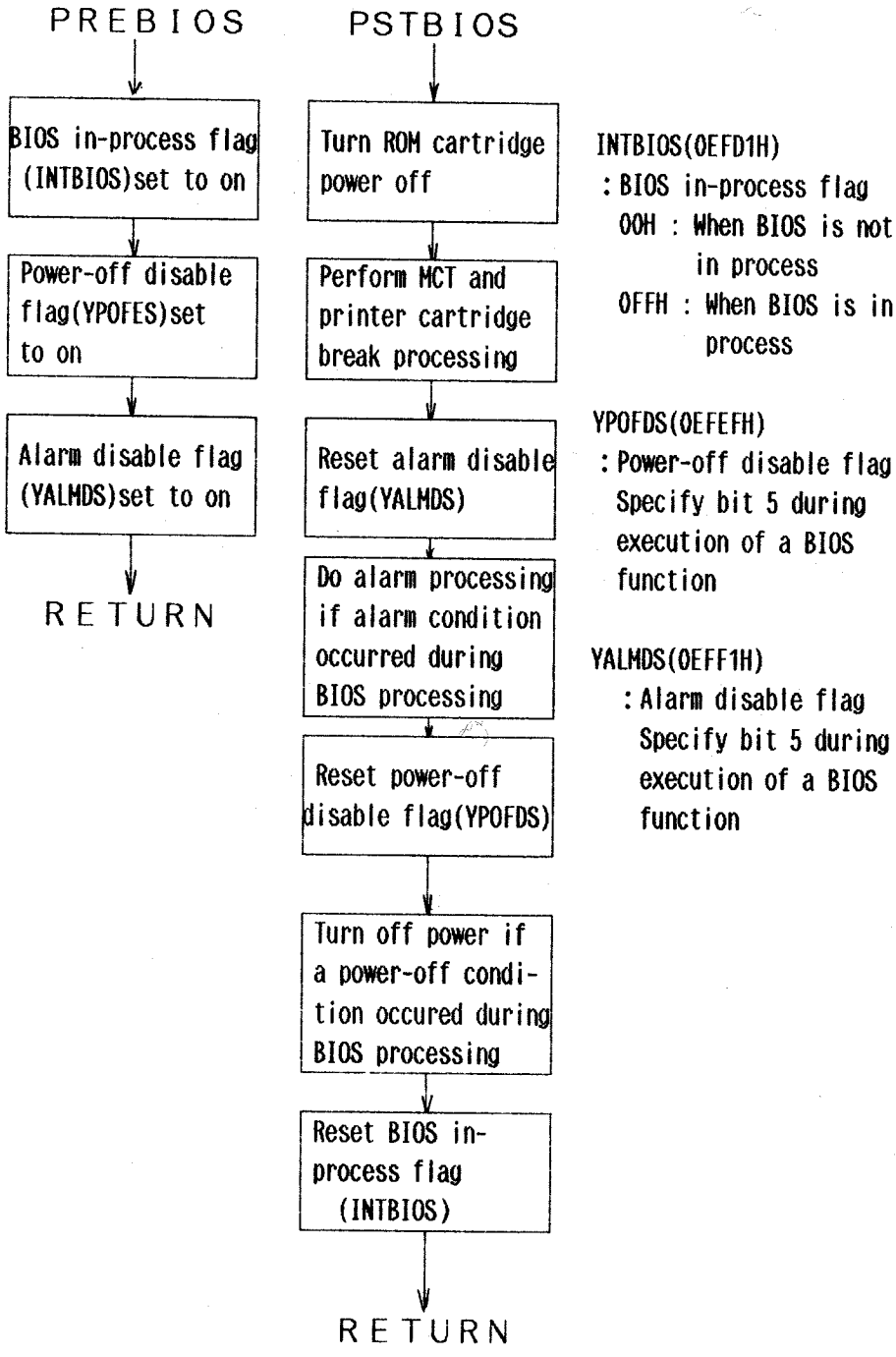


Figure 4.2.5 PREBIOS/PSTBIOS Processing Flow

(3) RSROMEXQ (ØFF99H)

Function: Transfers control to a ROM-based program for which resident is specified.

Entry parameter:

DE : FCB starting address of the program (file) to be executed

Return parameter: None

Explanation:

1) The PINE can execute ROM-based programs. This routine is used to executing ROM-based programs for which resident is specified.
2) An application program which uses ROM-based programs for which resident is specified must specify them at the beginning of the program. RSROMEXQ is called in such application programs.
3) RSROMEXQ checks whether or not the file specified in the specified FCB resides in a ROM capsule (drive B: or C:). If so, it reads the first sector and, after verifying that the program is ROM-based, transfers control to the specified address. If the file is not found in any ROM capsule, RSROMEXQ returns control to the calling program.
4) See Sections 4.5, "Resident Processing" and 4.6, "Executing a ROM Program."

The figure below illustrates the RSROMEXQ processing flow.

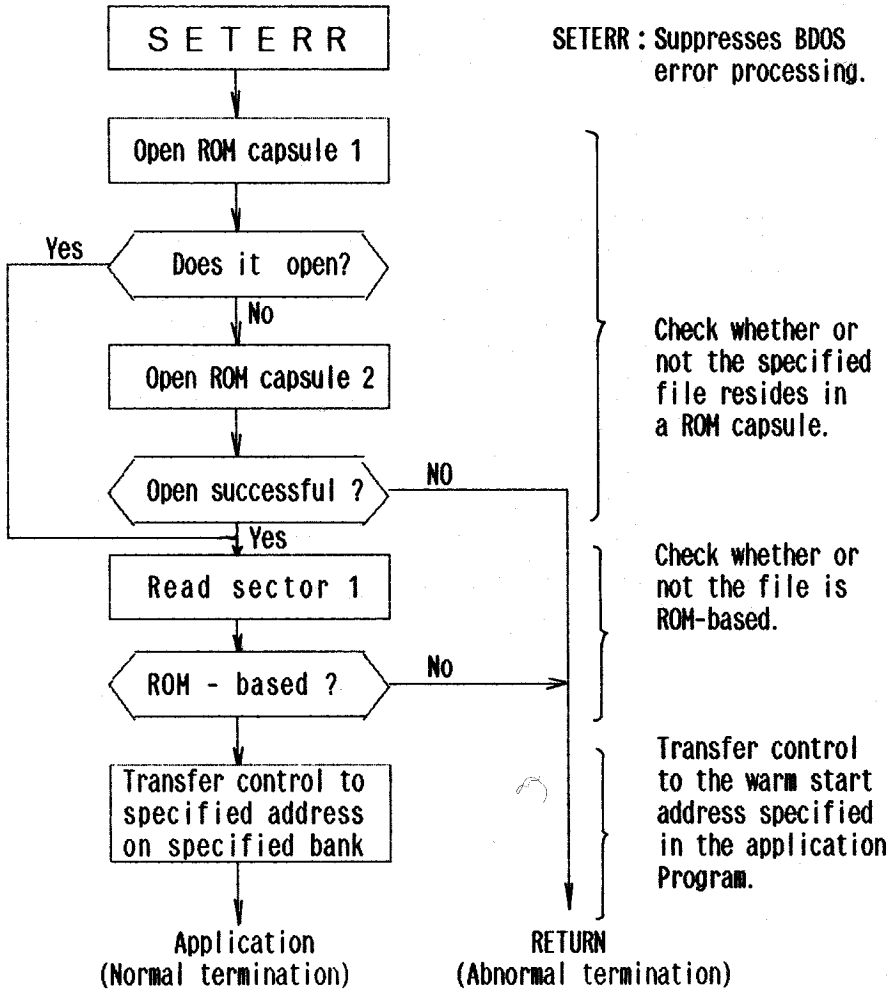


Figure 4.2.6 RSROMEXQ Processing Flow

The following example shows the processing flowchart of a ROM-based program for which resident is specified:

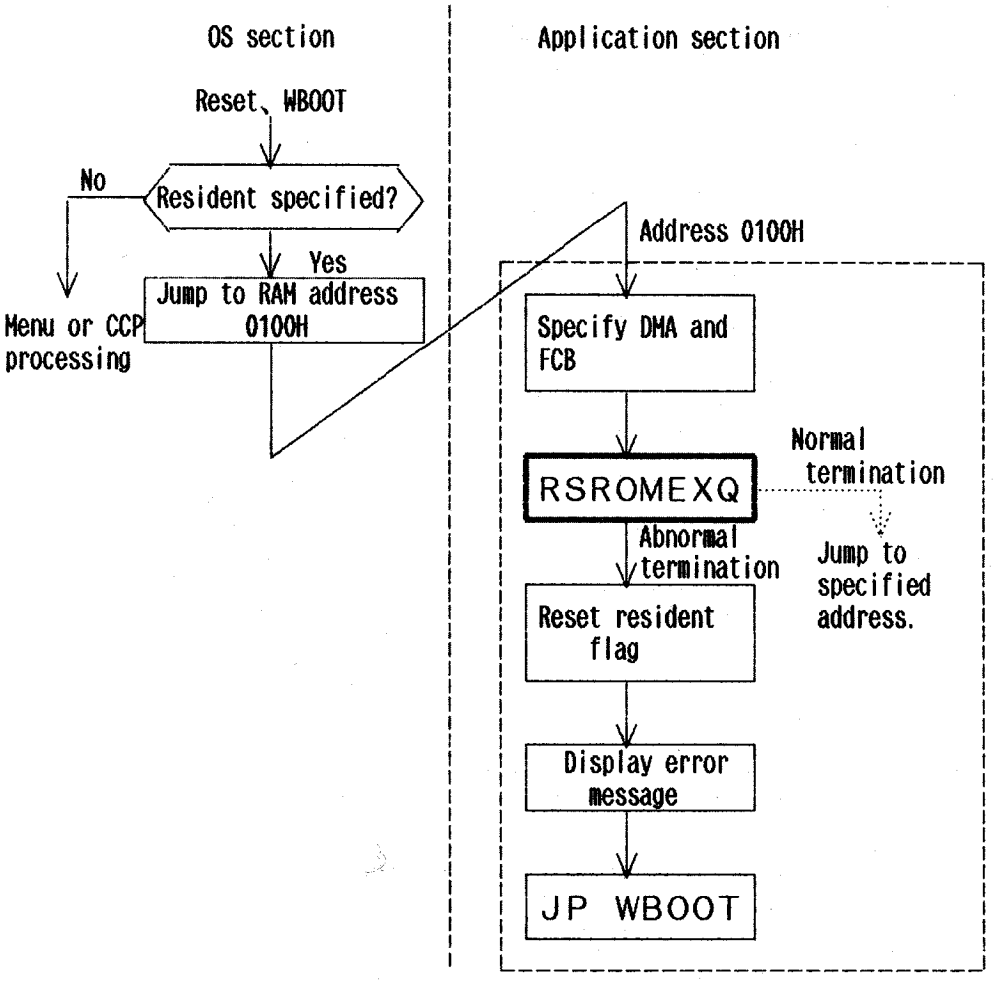


Figure 4.2.7 ROM-based Resident Program Processing Flow

ROM-based resident programs must include the processing enclosed within the broken lines. See Section 4.6, " Executing a ROM Program."

(4) SELBNK (FF9CH)

Function: Switches from the currently active bank to the specified bank.

Entry Parameter:
C: Bank to be selected
= ØFFH : System bank
= ØØH : Bank Ø (RAM)
= Ø1H : Bank 1
= Ø2H : Bank 2

Return Parameter:
C: Previously active bank
= ØFFH : System bank
= ØØH : Bank Ø (RAM)
= Ø1H : Bank 1
= Ø2H : Bank 2

Explanation:
All registers other than the C register retain their previous values.

Notes:
1) Since SELBANK makes no parameter check, normal operation cannot be guaranteed if a value outside the range of -1 to 2 is specified in C.
2) The stack area must be reserved in such a location that it can be used after the active bank is switched to the specified bank.
3) The CPU is in the EI state after executing this routine.

(5) LDAXX (ØFF9FH)

Function: Reads one byte of data from the specified address on the specified bank.

Entry parameters:
C : Bank from which data is to be read
= ØFFH : System bank
= ØØH : Bank Ø (RAM)
= Ø1H : Bank 1
= Ø2H : Bank 2
HL: Address of the data to be read

Return parameter:
A : Data from the specified address

Explanation:
1) All registers other than the A register hold their previous values.
2) This routine is equivalent to BIOS LOADX, but does not switch stacks.

Notes:
1) Since LDAXX makes no parameter check, normal operation cannot be guaranteed if a value outside the range of -1 to 2 is specified in C.
2) The stack area must be reserved in such a location that it can be used after the active bank is switched to the specified bank.
3) The CPU is in the EI state after executing this routine.

(6) STAXX (ØFFA2H)

Function: Writes one byte of data into the specified address on the specified bank.

Entry parameters:
A: Data to be written
C: Bank to which data is to be written
= ØFFH : System bank
= ØØH : Bank Ø (RAM)
= Ø1H : Bank 1
= Ø2H : Bank 2
HL: Address at which data is to be written

Return parameter: None

Explanation:
1) All registers retain their previous values.
2) This routine is equivalent to BIOS STORX, but does not switch stacks.
3) The C register should be set to ØØH because data can be written only in RAM.

Notes:
1) Since STAXX makes no parameter check, normal operation cannot be guaranteed if a value outside the range of -1 to 2 is specified in C.
2) The stack area must be reserved in such a location that it can be used after the active bank is switched to the specified bank.
3) The CPU is in the EI state after executing this routine.

(7) LDIRXX (ØFFA5H)

Function: Transfers the specified number of data bytes from the specified address in the specified bank to the specified address in bank Ø (RAM).

Entry parameters:
BC: Number of bytes to be transferred
DE: Starting address of the destination to which data is to be transferred
HL: Starting address of the data to be transferred
A : Bank from which data is to be transferred
= ØFFH : System bank
= ØØH : Bank Ø
= Ø1H : Bank 1
= Ø2H : Bank 2

Return parameters:
BC : ØØØØH
DE : DE + BC
HL : HL + BC

Explanation:
This routine is equivalent to BIOS LDIRX, but does not switch stacks.

Notes:
1) Since LDIRXX makes no error check, normal operation cannot be guaranteed if a value outside the range of -1 to 2 is specified in C.
2) The stack area must be reserved in such a location that it can be used after the active bank is switched to the specified bank.
3) The CPU is in the EI state after executing this routine.

(8) JUMPXX (ØFFA8H)

Function: Causes the CPU to jump to the specified address on the specified bank.

Entry parameters:
IX : Jump address
DISBNK (ØF52EH) : Destination bank number
= ØFFH : System bank
= ØØH : Bank Ø
= Ø1H : Bank 1
= Ø2H : Bank 2

Return parameter: None

Explanation:
1) All registers retain their previous values.
2) This routine is equivalent to BIOS JUMPX, but does not switch stacks.

Notes:
1) Since JUMPXX makes no parameter check, normal operation cannot be guaranteed if a value outside the range of -1 to 2 is specified in DISBNK.
2) The stack area must be reserved in such a location that it can be used after the active bank is switched to the destination bank.
3) The CPU is in the EI state after executing this routine.

(9) CALLXX (ØFFABH)

Function: Calls the specified address on the specified bank.

Entry parameters:
IX : Called routine address
DISBNK (ØF52EH) : Called bank number
= ØFFH : System bank
= ØØH : Bank Ø
= Ø1H : Bank 1
= Ø2H : Bank 2

Return parameters:
All registers retain their values when CALLXX is called.

Explanation:
This routine is equivalent to BIOS CALLX, but does not switch stacks.

Notes:
1) Since CALLXX makes no parameter check, normal operation cannot be guaranteed if a value outside the range of -1 to 2 is specified in DISBNK.
2) The stack area must be reserved in such a location that it can be used after the active bank is switched to the specified bank.
3) The CPU is in the EI state after executing this routine.

(10) INTOPR (0FFAEH)

Function: Directly calls MASKI in OS ROM.

Entry parameters:
B : Interrupt mask data
C : 7508 interrupt mask data

Return parameters:
B : Old interrupt mask status
C : Old 7508 interrupt mask status

Explanation:
1) This routine is equivalent to BIOS MASKI, but does not switch stacks.
2) See Section 3.4, "BIOS Details" for details of entry and return parameters.
3) This routine is used to control the interrupt status when a call cannot be made to BIOS (for example, in extended BIOS or interrupt processing).

Note: The stack area must be reserved at location 8000H or higher so that the same stack area can be used after this routine is called.

4.2.4 OS ROM Jump Table

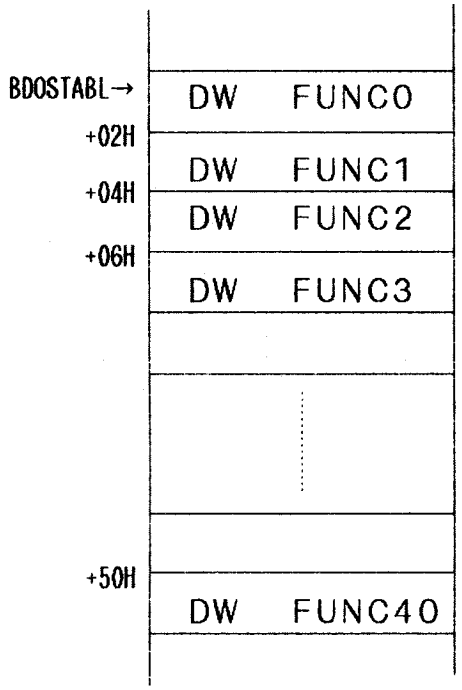
The OS ROM jump table includes 33 routines. This subsection describes the specifications for individual routines.

(1) BDOSTABL (0003H)

Function: Returns the starting address of the BDOS function vector stored in OS ROM.

Explanation:

1) This jump vector is used to directly call BDOS functions in OS ROM.
2) Note that BDOS in OS ROM neither switches stacks nor performs error recovery processing.

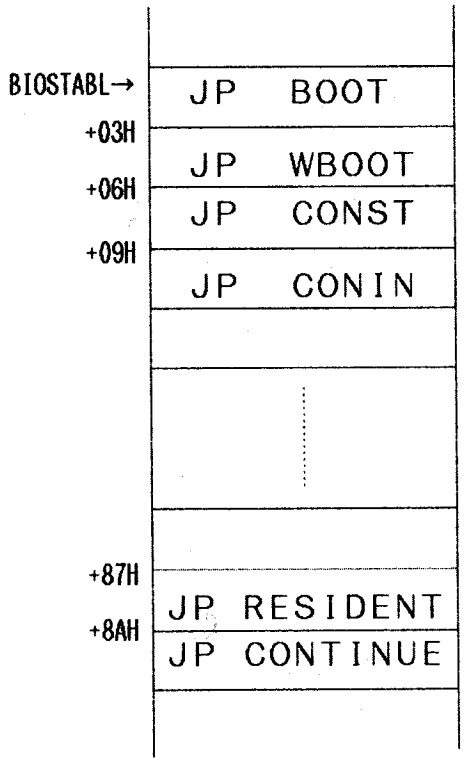


(2) BIOSTABL (0006H)

Function: Returns the starting address of the BIOS jump table stored in OS ROM.

Explanation:

1) This jump vector is used to directly call BIOS functions in OS ROM.
2) Note that BIOS in OS ROM neither switches stacks nor executes PREBIOS or PSTBIOS.

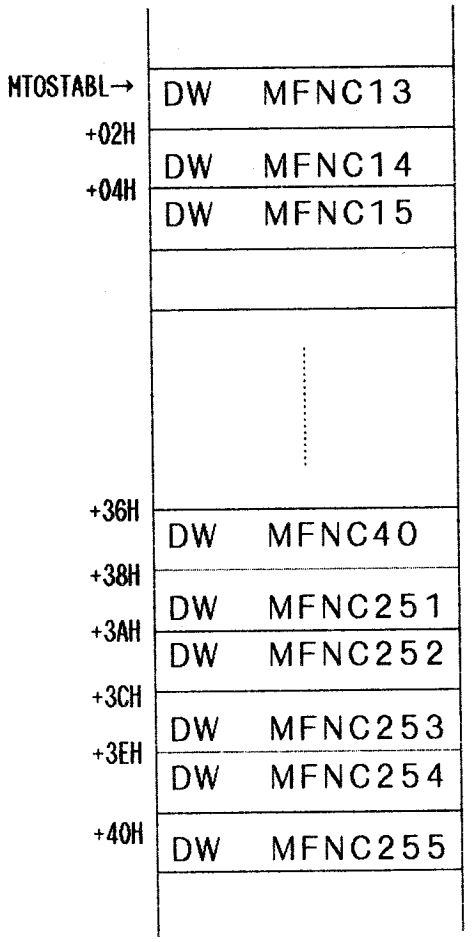


(3) MTOSTABL (0009H)

Function: Returns the starting address of the MTOS function vector stored in OS ROM.

Explanation:

1) This jump vector is used to directly call MTOS functions in OS ROM.
2) Note that MTOS neither switches stacks nor performs error recovery processing.

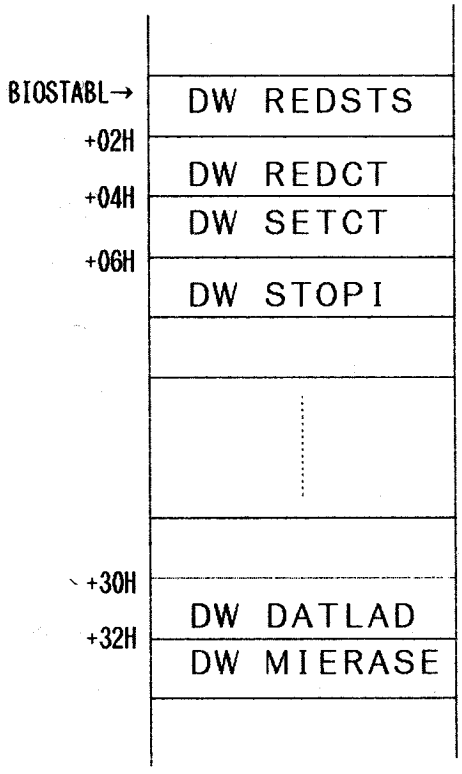


(4) MIOSTABL (ØØØCH)

Function: Returns the starting address of the MIOS function vector stored in OS ROM.

Explanation:

1) This jump vector is used to directly call MIOS functions in OS ROM.
2) Note that MIOS neither switches stacks nor checks whether or not MCT is installed.



(5) GOBACK (ØØØFH)

Function: Performs BDOS termination processing.

Entry parameter: None

Return parameter: None

Explanation:

This routine performs the termination processing described below after each BDOS function is executed.

1. Recover the current disk
2. Set return information

(6) SETERR (ØØ12H)

Function: Rewrites the jump vector for BDOS error processing.

Entry parameter: None

Return parameter: None

Explanation:

1) When a disk access error occurs after the jump vector for BDOS error processing is rewritten by this routine, no error message is displayed but BDOS error information is returned to the application program as a return code.
2) This routine is used by application programs to perform error recovery processing when an error occurs at disk access time.
3) BDOS errors are divided into the following categories:
   1. Bad sector error
      An error was found while reading or writing a disk.
   2. Select error
      An attempt was made to select a drive beyond the specified range or a drive that was not ready.
   3. R/O disk error
      An attempt was made to write to a read-only disk.
   4. R/O file error
      An attempt was made to write to a read-only file.
4) The A and H registers are loaded with the following return codes after SETERR is executed:

Register / Error	A Register	H Register
BIOS normal termination	BDOS return code	00H
Bad Sector	0FFH	01H
Bad Select	0FFH	02H
R/O Disk	0FFH	03H
R/O File	0FFH	04H
MCT Error	0FFH	05H

5) When the H register is loaded with ØØH, the return code corresponding to the CP/M return information is loaded into the A register.
6) See 3.2.4, "BDOS Errors" for the use of SETERR.

(7) RSTERR (ØØ15H)

Function: Initializes the jump vector for BDOS error processing.

Entry parameter: None

Return parameter: None

Explanation:

1) This routine is used to initialize the jump vector for BDOS error processing. After this routine is called, the corresponding error message is displayed when a disk error occurs.
2) WBOOT is also used to initialize the jump vector for BDOS error processing.

(8) CALADRS (ØØ18H)

Function: Calculates the starting addresses of individual system areas in CP/M according to the sizes of the current RAM disk and the user BIOS area.

Entry parameters:

SIZRAM (ØEF2CH) : Internal RAM disk size
USERBIOS (ØEF2DH) : User BIOS area size

Return parameters:

BC : Starting address of the user BIOS area
DE : Starting address of RBIOS1
IX : Starting address of RBDOS1
IY : Starting address of CCP

Explanation:

1) This routine is used to calculate the starting addresses of the CCP, BDOS, BIOS and user BIOS areas when the internal RAM disk or user BIOS size is modified.
2) The results returned by this routine are inserted into the following areas:
   BC ——→ TOPRAM (ØEF94H)
   DE ——→ BIlLAD (ØEF26H)
   IX ——→ BDSLAD (ØEF24H)
   IY ——→ CCPLAD (ØEF22H)
3) See 4.1.3, "User BIOS Area" for the use of this routine and a description of each system area.

(9) BIOSJTLD (ØØ1BH)

Function: Generates the RBIOS1 jump table.

Entry parameter: None

Return parameter: None

Explanation:

1) BIOSJTLD is used to modify the contents of the RBIOS1 jump table when the size of the RAM disk or user BIOS area is modified.
2) This routine generates the new jump table linked to RBIOS2 (in resident area) referring to BIlLAD (ØEF26H).

(1Ø) RAMDKMNT (ØØ1EH)

Function: Checks whether or not a RAM disk is installed.

Entry parameters:

A : Capacity of the internal RAM disk (in K bytes)
CY : Function specification
= Ø : No RAM disk is to be formatted.
= 1 : RAM disk is to be formatted.

Return parameters:

CY : Flag indicating whether or not the external RAM disk is installed
= Ø : External RAM disk is installed.
= 1 : No external RAM disk is installed.
A : Capacity of the external RAM disk (in K bytes)

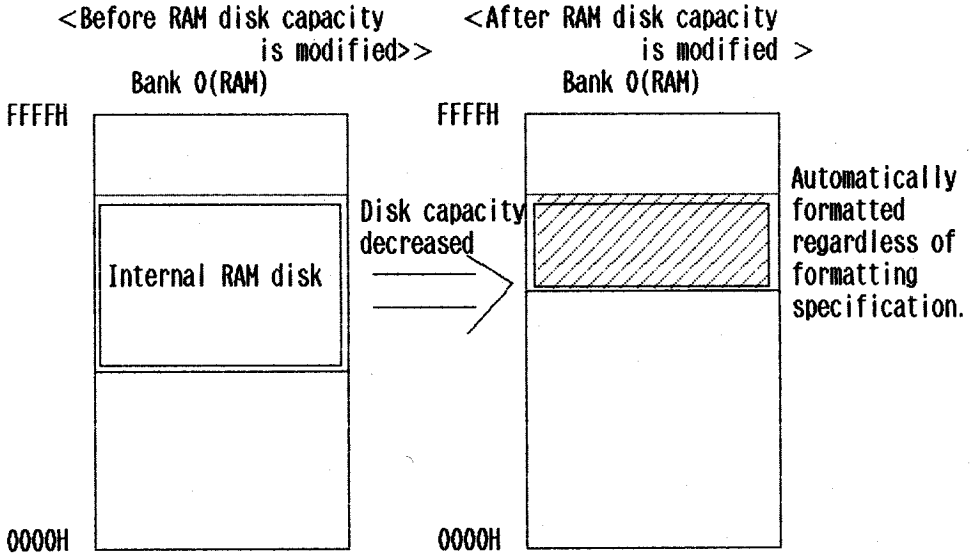
Explanation:

1) RAMDKMNT checks whether or not an external RAM disk is installed, and the size and starting address of the external RAM disk. Also, if specified, RAMDKMNT formats the RAM disk.
2) Call this routine when modifying the size of the internal RAM disk.

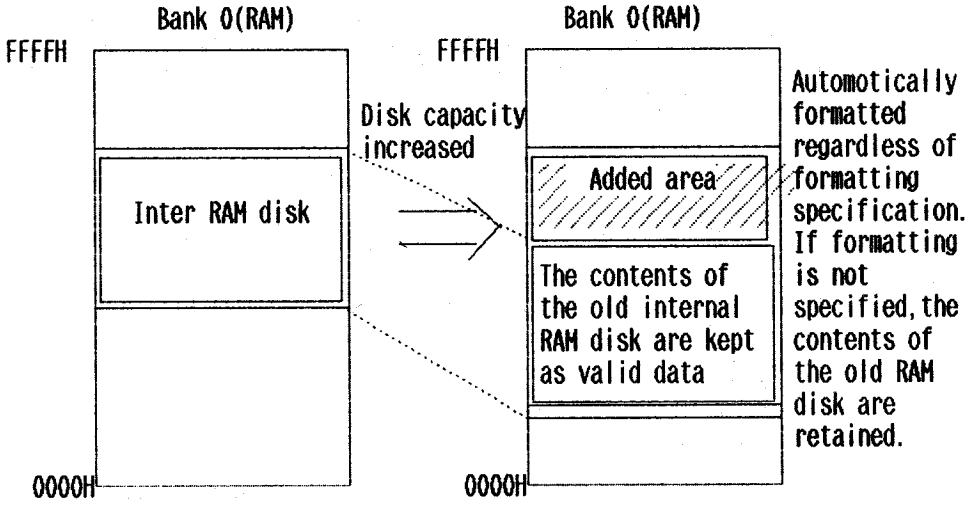
Notes:

1) If the size of the internal RAM disk is decreased during the execution of RAMDKMNT, the whole of the disk is automatically formatted regardless of whether or not formatting is specified.
2) If the size of the internal RAM is increased, the old data area remains unchanged; only the added area is automatically formatted.

○ Disk capacity decreased



○ Disk capacity decreased



The area enclosed in double lines represents the internal RAM disk area, and the shading indicates the area which is automatically formatted.

(11) MDFDPB (0021H)

Function: Corrects the disk parameter block.

Entry parameter: None

Return parameter: None

Explanation:

1) MDFDPB corrects the disk parameter block according to the current state of the disk whose size has been changed (A:, B:, C:, I:, J: or K:).
2) Before calling this routine, it is necessary to call the routine which checks whether or not the disk is installed.
3) Call this routine after modifying the size of the internal RAM disk.

(12) RAMCRFMT (0024H)

Function: Formats RAM cartridges.

Entry parameter: None

Return parameter: None

Explanation:

1) If a RAM cartridge is installed, this routine formats it.
2) If no RAM cartridge is installed, this routine returns control to the calling program without doing anything.
3) This routine is used to format RAM cartridges in system display processing.

(13) XMAKDIR (0027H)

Function: Initializes the MCT and creates a directory file.

Entry parameter:

DE : Starting address of the tape-file control block (T-FCB)

Return parameters:

A, H : Return information

Explanation:

1) XMAKDIR is used to initialize the MCT and to create a directory file. This function is equivalent to MTOS MAKDIR.
2) Refer to MTOS MAKDIR for details.
3) XMAKDIR, like MTOS MAKDIR, returns control to the calling program with return information loaded in the A and H registers, TOSRCD (0F7CEH) and IOSRCD (0F7CFH).

Note: When no MCT is mounted but the unit is in the mount state, XMAKDIR returns error information.

(14) XMOUNT (ØØ2AH)

Function: Performs MCT mount processing.

Entry parameter: None

Return parameters:
A, H : Return information

Explanation:
1) XMOUNT reads the MCT directory file into the RAM directory. This function is equivalent to MTOS MOUNT.
2) Refer to MTOS MOUNT for details.
3) XMOUNT, like MTOS MOUNT, returns control to the calling program with return information loaded in the A and H registers, TOSRCD (ØF7CEH) and IOSRCD (ØF7CFH).

Note: When no MCT is mounted but the unit is in the mount state, XMOUNT returns error information.

(15) XREMOVE (ØØ2DH)

Function: Performs MCT remove processing.

Entry parameter: None

Return parameters:
A, H : Return information

Explanation:
1) XREMOVE writes the RAM directory into the MCT directory file. This function is equivalent to MTOS REMOVE.
2) Refer to MTOS REMOVE for details.
3) XREMOVE, like MTOS REMOVE, returns control to the calling program with return information loaded in the A and H registers, TOSRCD (ØF7CEH) and IOSRCD (ØF7CEH).

Note: When no MCT is mounted but the unit is in the remove state, XREMOVE returns error information.

(16) EPSPSND (0030H)

Function: Sends the EPSP data from the SIO.

Entry parameters:

HL : Starting address of the EPSP send packet
A : Sending mode
LSB = 0 : Send only
= 1 : Send and receive

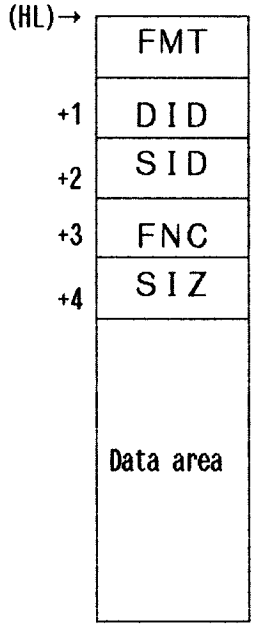
Return parameters:

CY : Return information
= 0 : Processing completed
= 1 : Processing interrupted
A : Return code
= 00H : Normal termination
= 61H : No device connected
= 62H : Communication error
= 63H : Time over
= 64H : CTRL/STOP pressed

Explanation:

1) EPSPSND sends the packet data specified in the HL register to the disk drive unit.
2) When this routine is executed with 'send and receive' specified in the A register, the receive data from the terminal floppy is stored at the address specified in the HL register or higher.
3) Packets are transmitted/received according to a protocol which is called EPSP (EPSON Serial Communication Protocol).

The format of the packet is shown below.
See Section 5.3, "Disk drives unit" for details on the functions of EPSPSND.



FMT : Format
= 00H : Transmission from PINE to FDD (disk drive)
= 01H : Transmission from FDD to PINE

DID : Destination ID

SID : Source ID
ID = 23H : PINE
= 31H : FDD1
= 32H : FDD2

FNC : Function
(See 5.3, Disk drive unit)

SIZ : Data size
Number of data bytes-1

Note: Since transmit processing is performed without regard to bank switching, the starting address of the packet (entry parameter, HL reg.) must be specified as 8000H or higher.

```
                        ;       ************************************************
                        ;               FDD UTILTY (READ SECTOR)
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is using EPSP
                        ;               utilities.
                        ;
                        ;       <> assemble condition <>
                        ;
                                .Z80
                        ;
                        ;       <> loading address <>
                        ;
                                .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
                        ;       BIOS entry
                        ;
EB03                    WBOOT           EQU     0EB03H          ; Warm Boot entry
EB06                    CONST           EQU     WBOOT   +03H    ; Console status entry
EB09                    CONIN           EQU     WBOOT   +06H    ; Console in entry
EB0C                    CONOUT          EQU     WBOOT   +09H    ; Console out entry
EB69                    CALLX           EQU     WBOOT   +66H    ; Call extra entry
                        ;
                        ;       System area
                        ;
F52E                    DISBNK          EQU     0F52EH          ;
F931                    PKT_TOP         EQU     0F931H          ;
F931                    PKT_FMT         EQU     PKT_TOP
F936                    PKT_RDT         EQU     0F936H
F93A                    SCRCH_BUF       EQU     0F93AH          ;
F9B6                    PKT_STS         EQU     0F9B6H
                        ;
                        ;          k value
                        ;
00FF                    SYSBANK         EQU     0FFH            ; System bank
0000                    BANK0           EQU     000H            ; Bank 0 (RAM)
0001                    BANK1           EQU     001H            ; Bank 1 (ROM capsel 1)
0002                    BANK2           EQU     002H            ; Bank 2 (ROM capsel 2)
                        ;
                        ;       OS ROM jump table
                        ;
0030                    EPSPSND         EQU     0030H
0033                    EPSPRCV         EQU     0033H
                        ;
                        ;
0031                    DID             EQU     31H
0023                    SID             EQU     23H
0077                    FNC             EQU     77H
0002                    SIZ             EQU     02H
                        ;
                        ;
0003                    BREAKKEY        EQU     03H     ;BREAK key code
000A                    LF              EQU     0AH     ;Line Feed code
000D                    CR              EQU     0DH     ;Cariage return code
0020                    SPCCD           EQU     20H     ;Space code
002E                    PERIOD          EQU     2EH     ;Period code
003F                    QMARK           EQU     3FH     ;Question mark code (3FH)
                        ;
                        ;
0000                    TERMINATOR      EQU     00H     ;Terminator code
                        ;
                        ;       BDOS function code table.
                        ;
1000                    SPWK            EQU     01000H          ;SP bottom address
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Read FDD block & display the data.
                        ;
0100                    MAIN:
0100    31 1000                 LD      SP,SPWK         ;Set Stack Pointer
                        ;
0103                    READ:
0103    CD 023F                 CALL    BREAKCHK        ;Check BREAK key (CTRL/C) press or not
0106    CA 017F                 JP      Z,ABORT
                        ;
0109    CD 0120                 CALL    SENDCMD         ;Send command to FDD.
010C    C2 016D                 JP      NZ,DISKERR      ; Disk access error.
                        ;
010F    3A F9B6                 LD      A,(PKT_STS)     ; Return parameter.
0112    B7                      OR      A               ;
0113    20 61                   JR      NZ,READERR      ; Read error.
                        ;
0115    CD 0188                 CALL    PRDATA          ; Display FDD data.
                        ;
0118    CD 0152                 CALL    SETNEXT
011B    30 E6                   JR      NC,READ         ;Repeat FDD read until reached end.
011D    C3 EB03                 JP      WBOOT
                        ;
```

```
                        ;
                        ;       ************************************************
                        ;               SEND READ-COMMAND AND 1 SECTOR READ
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               Same as EPSPSND
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
0120                    SENDCMD:
0120    21 F931                 LD      HL,PKT_FMT      ; EPSP packet top address.
0123    AF                      XOR     A               ;  Set FMT code.
0124    77                      LD      (HL),A          ;
0125    23                      INC     HL              ;
0126    36 31                   LD      (HL),DID        ;  Set DID code.
0128    23                      INC     HL              ;
0129    36 23                   LD      (HL),SID        ;  Set SID code.
012B    23                      INC     HL              ;
012C    36 77                   LD      (HL),FNC        ;  Set FNC code.
012E    23                      INC     HL              ;
012F    36 02                   LD      (HL),SIZ        ;  Set SIZ data.
0131    23                      INC     HL              ;
0132    3A 0266                 LD      A,(SEKDSK)      ;  Set drive code.
0135    77                      LD      (HL),A          ;
0136    23                      INC     HL              ;
0137    3A 0267                 LD      A,(SEKTRK)      ;  Set seek track number.
013A    77                      LD      (HL),A          ;
013B    23                      INC     HL              ;
013C    3A 0266                 LD      A,(SEKSEC)      ;  Set seek sector number.
013F    77                      LD      (HL),A          ;
                        ;
0140    3E FF                   LD      A,SYSBANK       ; Select OS bank.
0142    32 F52E                 LD      (DISBNK),A      ;
0145    DD 21 0030              LD      IX,EPSPSND      ; Call address (EPSP send)
0149    3E 01                   LD      A,01H           ; Receive after send.
014B    21 F931                 LD      HL,PKT_TOP      ; Packet top address.
014E    CD EB69                 CALL    CALLX           ; Go !!
                        ;
0151    C9                      RET                     ;
                        ;       ************************************************
                        ;               COUNT UP TRACK/SECTOR
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               CY : Return information
                        ;                  =0 -- Normal end.
                        ;                  =1 -- End of floppy
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0152                    SETNEXT:
0152    3A 0268                 LD      A,(SEKSEC)      ; Sector number increment.
0155    3C                      INC     A               ;
0156    32 0268                 LD      (SEKSEC),A      ;
0159    FE 41                   CP      65              ;  Larger than 64?
015B    3F                      CCF                     ;
015C    D0                      RET     NC              ;  No.
015D    3E 01                   LD      A,01H           ; Set initial value.
015F    32 0266                 LD      (SEKSEC),A      ;
                        ;
0162    3A 0267                 LD      A,(SEKTRK)      ; Track number increment.
0165    3C                      INC     A               ;
0166    32 0267                 LD      (SEKTRK),A      ;
0169    FE 0A                   CP      10              ;  Larger than 9?
016B    3F                      CCF                     ;
016C    C9                      RET                     ;
                        ;
                        ;*********************************
                        ;*                               *
                        ;*       ERROR ROUTINE           *
                        ;*                               *
                        ;*********************************
                        ;
016D                    DISKERR:
016D    21 0282                 LD      HL,DKERRMSG     ;FDD access error message.
0170    CD 01DC                 CALL    DSPMSG
0173    C3 EB03                 JP      WBOOT
0176                    READERR:
0176    21 02A8                 LD      HL,RDERRMSG     ;FDD read error message.
0179    CD 01DC                 CALL    DSPMSG
017C    C3 EB03                 JP      WBOOT
017F                    ABORT:
017F    21 02CC                 LD      HL,ABORTMSG     ;Display ABORT message.
0182    CD 01DC                 CALL    DSPMSG
0185    C3 EB03                 JP      WBOOT
                        ;
                        ;
```

```
                        ;*****************
                        ;*   PRDATA    *
                        ;*****************
                        ;
                        ;       Display FDD data (128 byte) by HEX format.
                        ;
                        ;       on entry :
                        ;       on exit  ; none
                        ;
                        ;       Registers are not preserved.
0188                    PRDATA:
0188    21 02D6                 LD      HL,TRKMSG       ;Track number.
018B    CD 01DC                 CALL    DSPMSG
018E    DD 21 0267              LD      IX,SEKTRK
0192    DD 22 0269              LD      (READPTR),IX
0196    06 01                   LD      B,1             ;Display data quantity
0198    CD 01E8                 CALL    DSPDATA         ;Display track number.
                        ;
019B    21 02E4                 LD      HL,SECMSG       ;Sector number
019E    CD 01DC                 CALL    DSPMSG
01A1    DD 21 0268              LD      IX,SEKSEC
01A5    DD 22 0269              LD      (READPTR),IX
01A9    06 01                   LD      B,1             ;Display data quantity
01AB    CD 01E8                 CALL    DSPDATA         ; Display secter number.
01AE    21 027F                 LD      HL,CRLF
01B1    CD 01DC                 CALL    DSPMSG          ;Line feed
                        ;
01B4    DD 21 F936              LD      IX,PKT_RDT
01B8    DD 22 0269              LD      (READPTR),IX    ;Read data top address
01BC    06 08                   LD      B,8             ;Display 16 byte data on each line
                        ;                               ;therefore it takes 8 line to list out
                        ;                               ;128 byte data
01BE                    PRDT00:
01BE    C5                      PUSH    BC
01BF    06 10                   LD      B,16
01C1    CD 01E8                 CALL    DSPDATA         ;FDD data (16 byte)
                        ;
01C4    0E 20                   LD      C,SPCCD
01C6    CD EB0C                 CALL    CONOUT
01C9    21 026B                 LD      HL,CHRPKT       ;Character image of FDD data, that stored
01CC    CD 01DC                 CALL    DSPMSG          ; behind CHRPKT
                        ;
01CF    21 027F                 LD      HL,CRLF
01D2    CD 01DC                 CALL    DSPMSG          ;Line feed
                        ;
01D5    CD 024C                 CALL    CHKWAIT         ;Wait next go.
01D8    C1                      POP     BC
01D9    10 E3                   DJNZ    PRDT00          ;Repeat PRMCTDT until B=0
                        ;
01DB    C9                      RET
                        ;
                        ;*************
                        ;*   DSPMSG   *
                        ;*************
                        ;
                        ;       Display string data to the console until find 00H.
                        ;
                        ;       on entry ; HL = Top address of string data
                        ;                       Data 00H is a terminator of string.
                        ;
                        ;       on exit  ; none
                        ;
                        ;       Registers are not preserved.
                        ;
01DC                    DSPMSG:
01DC    7E                      LD      A,(HL)          ;Display data
01DD    B7                      OR      A               ;Check terminator
01DE    C8                      RET     Z               ;If find terminater then Return
                        ;
01DF    E5                      PUSH    HL
01E0    4F                      LD      C,A
01E1    CD EB0C                 CALL    CONOUT          ;Display data to the console
01E4    E1                      POP     HL
01E5    23                      INC     HL              ;Update pointer
01E6    18 F4                   JR      DSPMSG          ;Repeat DSPMSG until find terminater
                        ;
                        ;
                        ;****************
                        ;*   DSPDATA    *
                        ;****************
                        ;
                        ;       Convert 1 byte data, that addressed by IX, to HEX format and
                        ;       LIST out it to printer. And store character image of data to
                        ;       CHRPKT.
                        ;
                        ;       on entry ;  B = Data quantity that to be LIST out
                        ;                   (READPTR) = Indicate data address
                        ;
                        ;       on exit  ; (READPTR) = Next data address
                        ;                  Character image of datas are stored behind CHRPKT.
                        ;
                        ;       Registers are not preserved.
                        ;
01E8                    DSPDATA:
01E8    78                      LD      A,B
01E9    B7                      OR      A
01EA    C8                      RET     Z               ;If data quqntity = 0 then return
                        ;
01EB    21 026B                 LD      HL,CHRPKT       ;HL=Start address of character data
01EE    DD 2A 0269              LD      IX,(READPTR)    ;IX=MCT data top address
01F2                    DSPDT00:
01F2    C5                      PUSH    BC
01F3    E5                      PUSH    HL
01F4    DD E5                   PUSH    IX
                        ;
01F6    DD 7E 00                LD      A,(IX)          ;A=DATA
```

```
01F9    36 2E                   LD      (HL),PERIOD     ;Store PERIOD mark (default data)
01FB    CB 7F                   BIT     7,A             ;If data is 80H through FFH or 00H through
01FD    20 05                   JR      NZ,DSPDT10      ;1FH then change to PERIOD mark and store
01FF    FE 20                   CP      SPCCD           ;it in CHRPKT
0201    38 01                   JR      C,DSPDT10       ;
0203    77                      LD      (HL),A          ;Store read data to CHRPKT
0204                    DSPDT10:
0204    CD 0225                 CALL    TOHEX           ;Convert to HEX
0207    C5                      PUSH    BC              ;Save lower 4bit hex data
0208    48                      LD      C,B
0209    CD EB0C                 CALL    CONOUT          ;LIST out upper 4bit hex data
020C    C1                      POP     BC
020D    CD EB0C                 CALL    CONOUT          ;LIST out lower 4bit hex data
0210    0E 20                   LD      C,SPCCD
0212    CD EB0C                 CALL    CONOUT          ;LIST out space
                        ;
0215    DD E1                   POP     IX
0217    E1                      POP     HL
0218    C1                      POP     BC
0219    23                      INC     HL
021A    DD 23                   INC     IX
021C    10 D4                   DJNZ    DSPDT00         ;Repeat until B=0
                        ;
021E    DD 22 0269              LD      (READPTR),IX    ;Store next data address
0222    36 00                   LD      (HL),TERMINATOR ;Store terminator of character data
                        ;
0224    C9                      RET
                        ;
                        ;*************
                        ;*   TOHEX   *
                        ;*************
                        ;
                        ;       Convert input data (A reg) to 2 byte HEX data (BC reg)
                        ;
                        ;       on entry :  A = input data
                        ;
                        ;       on exit  : BC = HEX data of input data
                        ;                       (B = upper 4bit data)
                        ;                       (C = lower 4bit data)
                        ;
0225                    TOHEX:
0225    F5                      PUSH    AF
                        ;
0226    1F                      RRA                     ;Shift upper 4bit to lower 4 bit
0227    1F                      RRA
0228    1F                      RRA
0229    1F                      RRA
                        ;
022A    CD 0234                 CALL    TOHEX10         ;Convert upper 4bit
022D    47                      LD      B,A
                        ;
022E    F1                      POP     AF              ;Convert lower 4bit
022F    CD 0234                 CALL    TOHEX10
0232    4F                      LD      C,A
0233    C9                      RET
                        ;
                        ;
                        ;***************
                        ;*   TOHEX10   *
                        ;***************
                        ;
                        ;       Convert lower 4bit of input data to HED dat.
                        ;
                        ;          entry : A = input data
                        ;       on exit  : A = HEX data of input data lower 4bit
                        ;
0234                    TOHEX10:
0234    E6 0F                   AND     0FH             ;Mask upper 4bit
0236    FE 0A                   CP      0AH
0238    38 02                   JR      C,TOHEX20
                        ;
023A    C6 07                   ADD     A,07H           ;If 0AH through 0FH then "A" TO "F"
                        ;
023C                    TOHEX20:
023C    C6 30                   ADD     A,30H
023E    C9                      RET
                        ;
                        ;****************
                        ;*   BREAKCHK   *
                        ;****************
                        ;
                        ;       Check BREAK key (CTRL/C) press or not
                        ;
                        ;       on entry : none
                        ;
                        ;       on exit  : Z flag = 1 --- Break key is pressed
                        ;                         = 0 --- Break key is not pressed
                        ;
023F                    BREAKCHK:
023F    CD EB06                 CALL    CONST
0242    3C                      INC     A
0243    C0                      RET     NZ              ;If key buffer is empty then return
                        ;
0244    CD EB09                 CALL    CONIN
0247    FE 03                   CP      BREAKKEY        ;Check BREAK key or not
0249    C8                      RET     Z               ;If BREAK key then return
024A    18 F3                   JR      BREAKCHK        ;Repeat BREAKCHK until buffer is empty
                        ;
024C                    CHKWAIT:
024C    CD EB06                 CALL    CONST
024F    3C                      INC     A
0250    C0                      RET     NZ
                        ;
0251    CD EB09                 CALL    CONIN
0254    FE 03                   CP      BREAKKEY
0256    CA EB03                 JP      Z,WBOOT
```

```
0259  FE 20                     CP      SPCCD
025B  20 EF                     JR      NZ,CHKWAIT
                        ;
025D  CD EB09                   CALL    CONIN
0260  FE 03                     CP      BREAKKEY
0262  CA EB03                   JP      Z,WBOOT
0265  C9                        RET
                        ;
                        ;*************************
                        ;*                       *
                        ;*      WORK AREA        *
                        ;*                       *
                        ;*************************
                        ;
0266                    SEKDSK:
0266  01                        DB      01H             ;
0267                    SEKTRK:
0267  04                        DB      04H             ; Directory part.
0268                    SEKSEC:
0268  01                        DB      01H
                        ;
0269                    READPTR:
0269                          DS      2               ;Pointer of READPKT
026B                    CHRPKT:
026B                          DS      20              ;Character data packet of MCT read data
                        ;
                        ;*************************
                        ;*                       *
                        ;*      MESSAGE AREA     *
                        ;*                       *
                        ;*************************
                        ;
027F                    CRLF:
027F  0D 0A                   DB      CR,LF
0281  00                      DB      TERMINATOR
                        ;
0282                    DKERRMSG:
0282  0D 0A                   DB      CR,LF
0284  46 6C 6F 70             DB      'Floppy disk drive access error !!'
0288  70 79 20 64
028C  69 73 6B 20
0290  64 72 69 76
0294  65 20 61 63
0298  63 65 73 73
029C  20 65 72 72
02A0  6F 72 20 21
02A4  21
02A5  0D 0A                   DB      CR,LF
02A7  00                      DB      TERMINATOR
                        ;
02A8                    RDERRMSG:
02A8  0D 0A                   DB      CR,LF
02AA  46 6C 6F 70             DB      'Floppy disk drive read error !!'
02AE  70 79 20 64
02B2  69 73 6B 20
02B6  64 72 69 76
02BA  65 20 72 65
02BE  61 64 20 65
02C2  72 72 6F 72
02C6  20 21 21
02C9  0D 0A                   CR,LF
02CB  00                      TERMINATOR
                        ;
02CC                    ABORTMSG:
02CC  0D 0A                   DB      CR,LF
02CE  41 62 6F 72             DB      'Aborted'
02D2  74 65 64
02D5  00                      DB      TERMINATOR
                        ;
02D6                    TRKMSG:
02D6  0D 0A                   DB      CR,LF
02D8  54 72 61 63             DB      'Track No = '
02DC  6B 20 4E 6F
02E0  20 3D 20
02E3  00                      DB      TERMINATOR
                        ;
02E4                    SECMSG:
02E4  20 20 20 20             DB            Sector No  = '
02E8  20 53 65 63
02EC  74 6F 72 20
02F0  4E 6F 20 20
02F4  3D 20
02F6  00                      DB      TERMINATOR
                        ;
                              END
```

(17) EPSPRCV (0033H)

Function: Receives the EPSP data from the SIO.

Entry parameter:
HL : Starting address of the EPSP receive packet

Return parameters:
A : Return code
= 00H : Normal termination
= 61H : No device connected
= 62H : Communication error
= 63H : Time over
= 64H : CTRL/STOP pressed
B : Information about receive packet
(valid when A = 00H)
= 00H : Packet with header
= 01H : Packet without header

Explanation:
1) EPSPRCV stores the receive data from the unit in the packet specified in the HL register.
2) The format of the receive packet is the same as described in (16) EPSPSND.

Note: Since receive processing is performed without regard to bank switching, the starting address of the packet (entry parameter, HL reg.) must be specified as 8000H or higher.

(18) XREDSP (0036H)

Function: Displays the LCD screen again.

Entry parameter: None

Return parameter: None

Explanation:
1) XREDSP displays the currently displayed data on the screen again. The cursor and window positions are unchanged.
2) Only character data is re-displayed by this routine. Note that graphics data disappears.
3) This routine is used to directly rewrite the data on the user screen and display the window again.

(19) XFONTGET (0039H)

Function: Gets character generator font data provided by the OS.

Entry parameters:
HL : Starting address of the area in which the data is to be stored
C : Character generator code

Return parameter:
The character generator font data is stored in the area (8 bytes) specified in the HL parameter.

Explanation:
1) XFONTGET is used to get a character generator font.
2) Characters are stored in 6 x 8-dot font as shown below.

	7	6	5	4	3	2	1	0
(HL)→	0	0	1	0	0	0	0	0
+1	0	1	0	1	0	0	0	0
2	1	0	0	0	1	0	0	0
3	1	0	0	0	1	0	0	0
4	1	1	1	1	1	0	0	0
5	1	0	0	0	1	0	0	0
6	1	0	0	0	1	0	0	0
7	0	0	0	0	0	0	0	0

(Fonts for 'A')

Note: If a character generator code not available in the OS is specified, the space code (all 0s) is returned.

(20) XUSRSCRN (0003CH)

Function: Switches to the user screen mode.

Entry parameter: None

Return parameter: None

Explanation:
1) XUSRSCRN switches the screen mode to user screen and allows information be displayed on the LCD.
2) If the user screen mode has already been selected, XUSRSCRN does nothing.

Reference:
1) Screen mode switch processing includes the following steps:
1. Replace work areas related to each screen.
Exchange the 39 data bytes starting at LSCADDR (0F290H) and those starting at LWORKBF (0F2B7H).
2. Output the VRAM starting address to I/O ports. (0P08H)
3. Output the Y-direction offset to I/O ports. (0P09H)

(21) XSYSSCRN (ØØ3FH)

Function: Switches to the system screen mode.

Entry parameter: None

Return parameter: None

Explanation:

1) XSYSSCRN switches the screen mode to system screen and allows information to be displayed on the LCD.
2) If the system screen mode has already been selected, XSYSSCRN does nothing.

Notes:

1) The system screen mode is used by the OS for (1) system display screen processing, (2) alarm screen processing and (3) power fail screen processing. When each type of processing is terminated, the screen switches to the user screen mode.
   Therefore, if processing (1) to (3) occurs while the system screen is being used in an application program, the original screen is not re-displayed when the processing is terminated.
2) When power is turned on, in both 'continue' and 'restart' modes, the user screen mode is selected.

(22) XSETCUSR (ØØ42H)

Function: Moves the cursor on the screen.

Entry parameters:

B : Number of lines to move the cursor
(Ø ≤ B ≤ maximum number of lines - 1)
C : Number of columns to move the cursor
(Ø ≤ C ≤ maximum number of columns - 1)

Return parameter: None

Explanation:

1) All registers except the A register retain their previous values.
2) XSETCUSR is equivalent to "Set cursor position" in BIOS CONOUT. This routine is revised to perform this function at a high speed.

Notes:

1) This routine is used between the executions of PREBIOS and PSTBIOS.
2) This routine makes no parameter error check.

(23) XWTVRAM (0045H)

Function: Directly transfers the specified character to VRAM.

Entry parameter:
C : ASCII code corresponding to the character to be transferred

Return parameter: None

Explanation:
1) All registers except the A register retain their previous values.
2) This routine is equivalent to "1 char. display" in BIOS CONOUT, but it does not write data into the screen buffer. XWTVRAM transfers one character to VRAM, displays it at the current cursor position, and moves the cursor one column forward.

Notes:
1) This routine is used between the executions of PREBIOS and PSTBIOS.
2) When no cursor is displayed on the LCD, this routine does nothing.

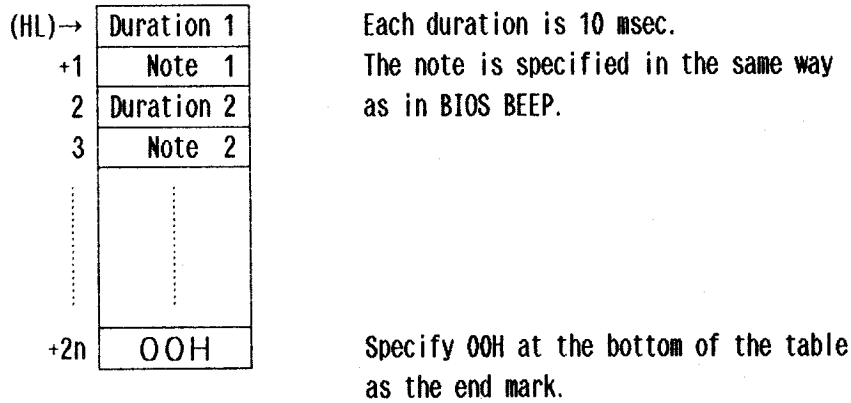
(24) MELODY (0048H)

Function: Generates sounds at the specified note.

Entry parameters:
HL : Starting address of the note table
C : Number of times to repeat the sound

Return parameter: None

Explanation:
1) MELODY generates sounds the specified number of times at the specified note.
2) The structure of the note table is shown below.



Notes:
1) The note table must be reserved at location 8000H or higher in RAM.
2) This routine uses BIOS BEEP processing.

(25) WRT7508 (004BH)

Function: Sends a command to the 7508 slave CPU.

Entry parameter:
C : Command data to be sent to the 7508 CPU

Return parameter: None

Explanation:
1) All registers except the A register retain their previous values.
2) WRT7508 is used to send the specified command to the 7508 CPU.
3) See Chapter 3, "7508 CPU" in Part I, "Firmware" for 7508 commands.

Note: Interrupts to the 7508 must be disabled before sending a command. Use BIOS to control interrupts.

(26) CMD7508 (0004EH)

Function: Sends a command to the 7508 slave CPU and returns one byte of response data.

Entry parameter:
C : Command data to the 7508

Return parameter:
A : Response data from the 7508

Explanation:
1) All registers except the A register retain their previous values.
2) CMD7508 sends the specified command to the 7508 and returns a 1-byte response in the A register.
3) See Chapter 3, "7508 CPU" in Part I, "Firmware" for 7508 commands.

Notes:
1) Interrupts to 7508 must be disabled before CMD7508 is executed. Use BIOS MASKI to control interrupts.
2) If two or more data bytes are to be transferred from the 7508, it is necessary to directly read the extra data bytes from I/O ports.

(27) HSRST (0051H)

Function: Reads the HS-mode cartridge input buffer state.

Entry parameter: None

Return parameters:

```
Z-flag  : State of input buffer (IBF)
          = 0 : Full
          = 1 : Empty
CY-flag : Type of input buffer data (F1)
          = 0 : Data
          = 1 : Command
```

Explanation:

1) All registers except the A register retain their previous values.
2) HSRST is used to check the state of data transferred from the cartridge in HS mode.

Note: The cartridge must be switched to HS mode before this routine is called.

(28) HSSCOM (0054H)

Function: Sends a command to a cartridge in HS mode.

Entry parameter:

```
C : Command data to be sent
```

Return parameter: None

Explanation:

1) All registers except the A and C registers retain their previous values.
2) HSSCOM checks the output buffer state (OBF) to the HS mode cartridge and, if sending of a command is enabled, sends the specified command to the cartridge. If not, this routine waits until sending is enabled.

Note: The cartridge must be switched to HS mode before this routine is called.

(29) HSSDAT (ØØ57H)

Function: Sends data to a cartridge in HS mode.

Entry parameter:
C : Data to be sent

Return parameter: None

Explanation:
1) All registers except the A and C registers retain their previous values.
2) HSSDAT checks the output buffer state (OBF) of the HS mode cartridge and, if sending of data is enabled, sends the specified data to the cartridge. If not, this routine waits until sending is enabled.

Note: The cartridge must be switched to HS mode before this routine is called.

(3Ø) HSRCV (ØØ5AH)

Function: Receives data from a cartridge in HS mode.

Entry parameter: None

Return parameters:
C : Input data or command
CY-flag : Type of input data (F1)
= Ø : Data
= 1 : Command

Explanation:
1) All registers except the A and C registers retain their previous values.
2) HSRCV checks the input buffer state (IBF) of the HS-mode cartridge and, if there is any data in the input buffer, returns control to the calling program with this data loaded in the C register. If there in no input data, this routine waits until data is received.

Note: The cartridge must be switched to HS mode before this routine is called.

(31) HSSBL (ØØ5DH)

Function: Sends a block to a cartridge in HS mode.

Entry parameters:
BC : Number of the block bytes to be sent
DE : Starting address of the block to be sent

Return parameter: None

Explanation:
HSSBL sends the specified block of data starting at the specified address to the HS-mode cartridge using HSSDAT.

Note: The block starting address specified by the DE parameter must be address 8ØØØH or higher in RAM.

(32) HSRBL (0060H)

Function: Inputs a block from a cartridge in HS mode.

Entry parameters:
BC : Number of the block bytes to be input
DE : Starting address of the input data storage area

Return parameters:
CY-flag : Return information
= 0 : Normal termination
= 1 : Command received

Explanation:
1) HSRBL inputs the specified block of data from the HS-mode cartridge using HSRCV and stores it at the specified address in the storage area.
2) Receiving a command during the execution of HSRBL causes the input processing to stop and returns control to the calling program with 1 loaded in the CY-flag.

Note: The block starting address specified by the DE parameter must be address 8000H or higher in RAM.

```
                        ;       ************************************************
                        ;               MCT UTILTY (READ BLOCK)
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is using HS mode
                        ;               utilities.
                        ;
                        ;       <> assemble condition <>
                        ;
                        ;       .Z80
                        ;
                        ;       <> loading address <>
                        ;
                        ;       .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
                        ;       BIOS entry
                        ;
EB03                    WBOOT           EQU     0EB03H          ; Warm Boot entry
EB06                    CONST           EQU     WBOOT   +03H    ; Console status entry
EB09                    CONIN           EQU     WBOOT   +06H    ; Console in entry
EB0C                    CONOUT          EQU     WBOOT   +09H    ; Console out entry
EB69                    CALLX           EQU     WBOOT   +66H    ; Call extra entry
EB7B                    MCMTX           EQU     WBOOT   +78H    ; MTOS entry
000A                      MIRWTT        EQU       0AH           ; Rewind to tape top
                        ;
                        ;       System area
                        ;
F52E                    DISBNK          EQU     0F52EH          ;
F53F                    CRGDEV          EQU     0F53FH          ;
FB97                    TOSBUF          EQU     0FB97H          ;
                        ;
                        ;          k value
                        ;
00FF                    SYSBANK         EQU     0FFH            ; System bank
0000                    BANK0           EQU     000H            ; Bank 0 (RAM)
0001                    BANK1           EQU     001H            ; Bank 1 (ROM capsel 1)
0002                    BANK2           EQU     002H            ; Bank 2 (ROM capsel 2)
                        ;
                        ;       OS ROM jump table
                        ;
0051                    HSRST           EQU     0051H   ;Read buffer status of cartridge on HS mode
0054                    HSSCOM          EQU     0054H   ;Send command to cartridge on HS mode
0057                    HSSDAT          EQU     0057H   ;Send data to cartridge on HS mode
005A                    HSRCV           EQU     005AH   ;Receive data from cartridge on HS mode
005D                    HSSBL           EQU     005DH   ;Send block data to cartridge on HS mode
0060                    HSRBL           EQU     0060H   ;Receive block data from cartridge on HS mode
                        ;
                        ;
0003                    BREAKKEY        EQU     03H     ;BREAK key code
000A                    LF              EQU     0AH     ;Line Feed code
000D                    CR              EQU     0DH     ;Cariage return code
0020                    SPCCD           EQU     20H     ;Space code
002E                    PERIOD          EQU     2EH     ;Period code
003F                    QMARK           EQU     3FH     ;Question mark code (3FH)
                        ;
0000                    TERMINATOR      EQU     00H     ;Terminator code
007F                    MASKMSB         EQU     7FH     ;Mask MSB code
0104                    BLOCKLEN        EQU     260     ;Block length
                        ;
                        ;       BDOS function code table
                        ;
0006                    CONSOLE         EQU     06H
                        ;
                        ;
FB97                    READPKT         EQU     TOSBUF          ;Receive MCT data packet
1000                    SPWK            EQU     01000H          ;SP bottom address
                        ;
0051                    MCTWRITE        EQU     51H     ;Write Data and Non Stop
0052                    MCTWRITES       EQU     52H     ;Write Data and Stop
0053                    MCTREAD         EQU     53H     ;Read Data and Non Stop
0054                    MCTREADS        EQU     54H     ;Read Data ans Stop
                        ;
                        ;
0048                    MCTCRTRDG       EQU     48H     ;MCT cartridge of CRGDEV work
                        ;
                        ;       Return code from SLAVE CPU              (PX-8 or HC-80/HC-88)
                        ;
0000                    RCD00           EQU     00H     ;NORMAL fine
0001                    RCD01           EQU     01H     ;SLAVE CPU is breaked
0002                    RCD02           EQU     02H     ;Command error
0003                    RCD03           EQU     03H     ;Communication error
0011                    RCD04           EQU     11H     ;LCD parameter is inconsistent error
0012                    RCD05           EQU     12H     ;User definable graphic code is not specified
0013                    RCD06           EQU     13H     ;Specified non acceptable user difinable code
0041                    RCD07           EQU     41H     ;Read error (can not manupilate the HEAD)
0042                    RCD08           EQU     42H     ;TAPE is stoped when operating it
0043                    RCD09           EQU     43H     ;Write protect error
0044                    RCD10           EQU     44H     ;MCT data error
0045                    RCD11           EQU     45H     ;MCT CRC error
0061                    RCD12           EQU     61H     ;ESPS can not communicate
0062                    RCD13           EQU     62H     ;ESPS communication error
0063                    RCD14           EQU     63H     ;ESPS over time error
0071                    RCD15           EQU     71H     ;Receive BEEP comand when execute BEEP command
                        ;
```

```
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Read MCT block & display the data.
                        ;
0100                    MAIN:
0100    31 1000                 LD      SP,SPWK         ;Set Stack Pointer
                        ;
0103    3A F53F                 LD      A,(CRGDEV)      ;Check MCT cartridge is connected or not
0106    FE 48                   CP      MCTCRTRDG       ;
0108    C2 0136                 JP      NZ,NONMCTERR    ;If not connected then error
                        ;
010B    21 02D7                 LD      HL,TAPEMSG      ;Insert tape message
010E    CD 01DF                 CALL    DSPMSG
0111    CD EB09                 CALL    CONIN           ;Wait until press any key
0114    21 0339                 LD      HL,CRLF
0117    CD 01DF                 CALL    DSPMSG
                        ;
011A    06 0A                   LD      B,MIRWTT        ;BIOS MIOS rewind to begining of tape
011C    CD EB7B                 CALL    MCMTX
                        ;
011F                    READ:
011F    CD 0242                 CALL    BREAKCHK        ;Check BREAK key (CTRL/C) press or not
0122    CA 0145                 JP      Z,ABORT
                        ;
0125    CD 0151                 CALL    READBLOCK       ;Read 1 block from MCT
                        ;
0128    FE 42                   CP      RCD08           ;Check tape end
012A    CA 013B                 JP      Z,TAPEEND
012D    B7                      OR      A               ;Check return code
012E    C2 0140                 JP      NZ,READERR
                        ;
0131    CD 0192                 CALL    PRMCTDATA       ;Print (LIST out) MCT data
0134    18 E9                   JR      READ            ;Repeat MCT read until reached tape end
                                                        ;or read error occurred
                        ;
                        ;
                        ;*********************************
                        ;*                               *
                        ;*       ERROR ROUTINE           *
                        ;*                               *
                        ;*********************************
                        ;
0136                    NONMCTERR:
0136    21 02AC                 LD      HL,NONMCTMSG    ;MCT cartridge is not connected
0139    18 0A                   JR      ABORT
                        ;
013B                    TAPEEND:
013B    21 032C                 LD      HL,TPENDMSG     ;TAPE end message
013E    18 05                   JR      ABORT
                        ;
0140                    READERR:
0140    21 033C                 LD      HL,RDERRMSG     ;MCT read error message
0143    18 00                   JR      ABORT
                        ;
0145                    ABORT:
0145    CD 01DF                 CALL    DSPMSG
0148    21 0362                 LD      HL,ABORTMSG     ;Display ABORT message
014B    CD 01DF                 CALL    DSPMSG
014E    C3 EB03                 JP      WBOOT
                        ;
                        ;
                        ;*****************
                        ;*   READBLOCK   *
                        ;*****************
                        ;
                        ;       Read 1 block from MCT with stop mode. The followings are the format
                        ;       of receiving data.
                        ;
                        ;       DATA(262 byte) + ACK(1 byte)
                        ;
                        ;       DATA  ---    2 byte = tape counter of begining block
                        ;                    4 byte = block discrimination field
                        ;                  256 byte = data
                        ;
                        ;
                        ;       on exit  ; A = Return code (ACK)
                        ;                  Read data (262 byte) stored behind RCVPKT1+1,
                        ;
                        ;                  RCVPKT1      ----- ACK
                        ;                  RCVPKT1+1    --+
                        ;                                 |-- data
                        ;                  RCVPKT1+262  --+
                        ;
                        ;       Registers are not preserved.
                        ;
0151                    READBLOCK:
0151    0E 54                   LD      C,MCTREADS      ;MCT Read Data and Stop command
0153    DD 21 0054              LD      IX,HSSCOM       ;Send command
0157    CD 01D6                 CALL    CALLXSYS
                        ;
015A    0E 01                   LD      C,01H           ;Quantity of read data (260 byte)
015C    DD 21 0057              LD      IX,HSSDAT       ;Send data
0160    CD 01D6                 CALL    CALLXSYS
0163    0E 04                   LD      C,04H
0165    DD 21 0057              LD      IX,HSSDAT
0169    CD 01D6                 CALL    CALLXSYS
                        ;
016C    01 0107                 LD      BC,263          ;Quantity of receiving data
016F    21 FB98                 LD      HL,RCVPKT1+1    ;Store receiving data behind RCVPKT1
0172    CD 017C                 CALL    RCVBLKDAT       ;Recive block data from MCT
                        ;
0175    3A FC9E                 LD      A,(RCVPKT1+263) ;Get return code
0178    32 FB97                 LD      (RCVPKT1),A     ;Store return code
```

```
017B    C9                      RET
                        ;
                        ;*****************
                        ;*   RCVBLKDAT   *
                        ;*****************
                        ;
                        ;       Receive block data from MCT and store it behind (HL).
                        ;
                        ;       on entry : BC = Quantity of receiving data
                        ;                  HL = Top address of storing receive data
                        ;
                        ;       on exit  : none
                        ;
                        ;       Registers are not preserved.
                        ;
017C                    RCVBLKDAT:
017C    78                      LD      A,B             ;If receive data = 0 then end
017D    B1                      OR      C
017E    C8                      RET     Z
                        ;
017F                    RCVBLK00:
017F    C5                      PUSH    BC
0180    E5                      PUSH    HL
0181    DD 21 005A              LD      IX,HSRCV        ;Receive data address
0185    CD 01D6                 CALL    CALLXSYS
0188    E1                      POP     HL
0189    71                      LD      (HL),C          ;Store receive data in packet
018A    23                      INC     HL
018B    C1                      POP     BC
018C    0B                      DEC     BC
018D    78                      LD      A,B             ;Check end of receive
018E    B1                      OR      C
018F    20 EE                   JR      NZ,RCVBLK00     ;Repeat RCVBLK00 until BC=0
                        ;
0191    C9                      RET
                        ;
                        ;*****************
                        ;*   PRMCTDATA   *
                        ;*****************
                        ;
                        ;       Display MCT data (262 byte) by HEX format
                        ;
                        ;       on entry : MCT data is contains behind READPKT+1
                        ;       on exit  : none
                        ;
                        ;       Registers are not preserved.
                        ;
0192                    PRMCTDATA:
0192    DD 21 FB98              LD      IX,READPKT+1
0196    DD 22 0275              LD      (READPTR),IX    ;Read data top address
                        ;
019A    21 036C                 LD      HL,CONTMSG      ;Block begining counter message
019D    CD 01DF                 CALL    DSPMSG
01A0    06 02                   LD      B,2             ;Display data quantity
01A2    CD 01EB                 CALL    DSPDATA         ;Display block begining counter
                        ;
01A5    21 038D                 LD      HL,BLOCKMSG     ;Block discrimination message
01A8    CD 01DF                 CALL    DSPMSG
01AB    06 04                   LD      B,4             ;Display data quantity
01AD    CD 01EB                 CALL    DSPDATA
01B0    21 0339                 LD      HL,CRLF
01B3    CD 01DF                 CALL    DSPMSG          ;Line feed
                        ;
01B6    06 10                   LD      B,16            ;Display 16 byte data on each line
                        ;                               ;therefore it takes 16 line to list out
                        ;                               ;256 byte data
01B8                    PRMCTDT00:
01B8    C5                      PUSH    BC
01B9    06 10                   LD      B,16
01BB    CD 01EB                 CALL    DSPDATA         ;MCT data (16 byte)
                        ;
01BE    0E 20                   LD      C,SPCCD
01C0    CD EB0C                 CALL    CONOUT
01C3    21 0277                 LD      HL,CHRPKT       ;Character image of MCT data, that stored
01C6    CD 01DF                 CALL    DSPMSG          ; behind CHEPKT
                        ;
01C9    21 0339                 LD      HL,CRLF
01CC    CD 01DF                 CALL    DSPMSG          ;Line feed
                        ;
01CF    CD 024F                 CALL    CHKWAIT         ;Wait next go.
01D2    C1                      POP     BC
01D3    10 E3                   DJNZ    PRMCTDT00       ;Repeat PRMCTDT until B=0
                        ;
01D5    C9                      RET
                        ;
                        ;*****************
                        ;*   CALLXSYS    *
                        ;*****************
                        ;
                        ;       Select SYSTEM BANK and call BIOS CALLX.
                        ;
                        ;       on entry : IX = Destination routine address
                        ;                  (DISBNK) = indicate destination BANK
                        ;
                        ;       on exit  : none
                        ;
                        ;       Registers are not preserved.
                        ;
01D6                    CALLXSYS:
01D6    3E FF                   LD      A,SYSBANK
01D8    32 F52E                 LD      (DISBNK),A
01DB    CD EB69                 CALL    CALLX
01DE    C9                      RET
```

```
                        ;*************
                        ;*   DSPMSG   *
                        ;*************
                        ;
                        ;       Display string data to the console until find 00H.
                        ;
                        ;       on entry : HL = Top address of string data
                        ;                      Data 00H is a terminater of string.
                        ;
                        ;       on exit  : none
                        ;
                        ;       Registers are not preserved.
                        ;
01DF                    DSPMSG:
01DF    7E                      LD      A,(HL)          ;Display data
01E0    B7                      OR      A               ;Check terminater
01E1    C8                      RET     Z               ;If find terminater then Return
                        ;
01E2    E5                      PUSH    HL
01E3    4F                      LD      C,A
01E4    CD EB0C                 CALL    CONOUT          ;Display data to the console
01E7    E1                      POP     HL
01E8    23                      INC     HL              ;Update pointer
01E9    18 F4                   JR      DSPMSG          ;Repeat DSPMSG until find terminater
                        ;
                        ;
                        ;****************
                        ;*   LISTDATA   *
                        ;****************
                        ;
                        ;       Convert 1 byte data, that addressed by IX, to HEX format and
                        ;       LIST out it to printer. And store character image of data to
                        ;       CHRPKT.
                        ;
                        ;       on entry ;  B = Data quantity that to be LIST out
                        ;                   (READPTR) = Indicate data address
                        ;
                        ;       on exit  ; (READPTR) = Next data address
                        ;                  Character image of datas are stored behind CHRPKT.
                        ;
                        ;       Registers are not preserved.
                        ;
01EB                    DSPDATA:
01EB    78                      LD      A,B
01EC    B7                      OR      A
01ED    C8                      RET     Z               ;If data quqntity = 0 then return
                        ;
01EE    21 0277                 LD      HL,CHRPKT       ;HL=Start address of character data
01F1    DD 2A 0275              LD      IX,(READPTR)    ;IX=MCT data top address
01F5                    DSPDT00:
01F5    C5                      PUSH    BC
01F6    E5                      PUSH    HL
01F7    DD E5                   PUSH    IX
                        ;
01F9    DD 7E 00                LD      A,(IX)          ;A=DATA
01FC    36 2E                   LD      (HL),PERIOD     ;Store PERIOD mark (default data)
01FE    CB 7F                   BIT     7,A             ;If data is 80H through FFH or 00H through
0200    20 05                   JR      NZ,DSPDT10      ;1FH then change to PERIOD mark and store
0202    FE 20                   CP      SPCCD           ;it in CHRPKT
0204    38 01                   JR      C,DSPDT10       ;
0206    77                      LD      (HL),A          ;Store read data to CHRPKT
0207                    DSPDT10:
0207    CD 0228                 CALL    TOHEX           ;Convert to HEX
020A    C5                      PUSH    BC              ;Save lower 4bit hex data
020B    48                      LD      C,B
020C    CD EB0C                 CALL    CONOUT          ;LIST out upper 4bit hex data
020F    C1                      POP     BC
0210    CD EB0C                 CALL    CONOUT          ;LIST out lower 4bit hex data
0213    0E 20                   LD      C,SPCCD
0215    CD EB0C                 CALL    CONOUT          ;LIST out space
                        ;
0218    DD E1                   POP     IX
021A    E1                      POP     HL
021B    C1                      POP     BC
021C    23                      INC     HL
021D    DD 23                   INC     IX
021F    10 D4                   DJNZ    DSPDT00         ;Repeat until B=0
                        ;
0221    DD 22 0275              LD      (READPTR),IX    ;Store next data address
0225    36 00                   LD      (HL),TERMINATOR ;Store terminator of character data
                        ;
0227    C9                      RET
                        ;
                        ;*************
                        ;*   TOHEX   *
                        ;*************
                        ;
                        ;       Convert input data (A reg) to 2 byte HEX data (BC reg)
                        ;
                        ;       on entry :  A = input data
                        ;
                        ;       on exit  : BC = HEX data of input data
                        ;                       (B = upper 4bit data)
                        ;                       (C = lower 4bit data)
                        ;
0228                    TOHEX:
0228    F5                      PUSH    AF
                        ;
0229    1F                      RRA                     ;Shift upper 4bit to lower 4 bit
022A    1F                      RRA
022B    1F                      RRA
022C    1F                      RRA
                        ;
022D    CD 0237                 CALL    TOHEX10         ;Convert upper 4bit
0230    47                      LD      B,A
                        ;
0231    F1                      POP     AF              ;Convert lower 4bit
```

```
0232  CD 0237                   CALL    TOHEX10
0235  4F                        LD      C,A
0236  C9                        RET
                        ;
                        ;***************
                        ;*   TOHEX10   *
                        ;***************
                        ;
                        ;       Convert lower 4bit of input data to HED dat.
                        ;
                        ;        * entry : A = input data
                        ;       on exit  : A = HEX data of input data lower 4bit
                        ;
0237                    TOHEX10:
0237  E6 0F                      AND     0FH             ;Mask upper 4bit
0239  FE 0A                      CP      0AH
023B  38 02                      JR      C,TOHEX20
                        ;
023D  C6 07                      ADD     A,07H           ;If 0AH through 0FH then "A" TO "F"
                        ;
023F                    TOHEX20:
023F  C6 30                      ADD     A,30H
0241  C9                        RET
                        ;
                        ;****************
                        ;*   BREAKCHK   *
                        ;****************
                        ;
                        ;       Check BREAK key (CTRL/C) press or not
                        ;
                        ;       on entry : none
                        ;
                        ;       on exit  : Z flag = 1 --- Break key is pressed
                        ;                         = 0 --- Break key is not pressed
                        ;
0242                    BREAKCHK:
0242  CD EB06                    CALL    CONST
0245  3C                        INC     A
0246  C0                        RET     NZ              ;If key buffer is empty then return
                        ;
0247  CD EB09                    CALL    CONIN
024A  FE 03                      CP      BREAKKEY        ;Check BREAK key or not
024C  C8                        RET     Z               ;If BREAK key then return
024D  18 F3                      JR      BREAKCHK        ;Repeat BREAKCHK until buffer is empty
                        ;
024F                    CHKWAIT:
024F  CD EB06                    CALL    CONST
0252  3C                        INC     A
0253  C0                        RET     NZ
                        ;
0254  CD EB09                    CALL    CONIN
0257  FE 03                      CP      BREAKKEY
0259  CA EB03                    JP      Z,WBOOT
025C  FE 20                      CP      SPCCD
025E  20 EF                      JR      NZ,CHKWAIT
                        ;
0260  CD EB09                    CALL    CONIN
0263  FE 03                      CP      BREAKKEY
0265  CA EB03                    JP      Z,WBOOT
0268  C9                        RET
                        ;
                        ;*************************
                        ;*                       *
                        ;*      WORK AREA        *
                        ;*                       *
                        ;*************************
                        ;
0269                    READMCT:                        ;SLAVE MCT Read Data and Stop
0269  0271                      DW      SNDPKT1         ;
026B  0004                      DW      SNDLN1          ;
026D  FB97                      DW      RCVPKT1         ;
026F  0107                      DW      RCVLN1          ;
                        ;
0271  54                SNDPKT1:        DB      MCTREADS        ;MCT Read Data and Stop command
0272  01                              DB      001H            ;High byte of data quantity (260)
0273  04                              DB      004H            ;Low  byte of data quantity (260)
0274  3F                              DB      QMARK           ;Block discrimination code
                                                              ;'?' means that no check
0004                    SNDLN1          EQU     4               ;Sending parameter length
FB97                    RCVPKT1         EQU     READPKT         ;MCT read data receive packet
0107                    RCVLN1          EQU     263             ;Read data byte
                        ;
0275                    READPTR:        DS      2               ;Pointer of READPKT
0277                    CHRPKT:         DS      20              ;Character data packet of MCT read data
                        ;
                        ;*************************
                        ;*                       *
                        ;*      MESSAGE AREA     *
                        ;*                       *
                        ;*************************
                        ;
028B                    MCNERRMSG:
028B  0D 0A                     DB      CR,LF
028D  49 6E 63 6F               DB      'Incompatible machine error !'
0291  6D 70 61 74
0295  69 62 6C 65
0299  20 6D 61 63
029D  68 69 6E 65
02A1  20 65 72 72
02A5  6F 72 20 21
02A9  0D 0A                     DB      CR,LF
02AB  00                        DB      TERMINATOR
                        ;
```

```
02AC                                    NONMCTMSG:
02AC    0D 0A                                   DB      CR,LF
02AE    4D 43 54 20                             DB      'MCT cartridge is not connected error !'
02B2    63 61 72 74
02B6    72 69 64 67
02BA    65 20 69 73
02BE    20 6E 6F 74
02C2    20 63 6F 6E
02C6    6E 65 63 74
02CA    65 64 20 65
02CE    72 72 6F 72
02D2    20 21
02D4    0D 0A                                   DB      CR,LF
02D6    00                                      DB      TERMINATOR
                                        ;
02D7                                    TAPEMSG:
02D7    0D 0A                                   DB      CR,LF
02D9    49 6E 73 65                             DB      'Insert micro cassette tape,'
02DD    72 74 20 6D
02E1    69 63 72 6F
02E5    20 63 61 73
02E9    73 65 74 74
02ED    65 20 74 61
02F1    70 65 2C
02F4    61 6E 64 20                             DB      'and confirm ready to print out,'
02F8    63 6F 6E 66
02FC    69 72 6D 20
0300    72 65 61 64
0304    79 20 74 6F
0308    20 70 72 69
030C    6E 74 20 6F
0310    75 74 2E
0313    0D 0A                                   DB      CR,LF
0315    54 68 65 6E                             DB      'Then press RETURN key.'
0319    20 70 72 65
031D    73 73 20 52
0321    45 54 55 52
0325    4E 20 6B 65
0329    79 2E
032B    00                                      DB      TERMINATOR
                                        ;
032C                                    TPENDMSG;
032C    0D 0A                                   DB      CR,LF
032E    54 41 50 45                             DB      'TAPE end !!'
0332    20 65 6E 64
0336    20 21 21
0339                                    CRLF;
0339    0D 0A                                   DB      CR,LF
033B    00                                      DB      TERMINATOR
                                        ;
033C                                    RDERRMSG:
033C    0D 0A                                   DB      CR,LF
033E    4D 69 63 72                             DB      'Micro cassette tape read error !!'
0342    6F 20 63 61
0346    73 73 65 74
034A    74 65 20 74
034E    61 70 65 20
0352    72 65 61 64
0356    20 65 72 72
035A    6F 72 20 21
035E    21
035F    0D 0A                                   DB      CR,LF
0361    00                                      DB      TERMINATOR
                                        ;
0362                                    ABORTMSG:
0362    0D 0A                                   DB      CR,LF
0364    41 62 6F 72                             DB      'Aborted'
0368    74 65 64
036B    00                                      DB      TERMINATOR
                                        ;
036C                                    CONTMSG:
036C    0D 0A                                   DB      CR,LF
036E    42 6C 6F 63                             DB      'Block beginning tape count  = '
0372    6B 20 62 65
0376    67 69 6E 6E
037A    69 6E 67 20
037E    74 61 70 65
0382    20 63 6F 75
0386    6E 74 20 20
038A    3D 20
038C    00                                      DB      TERMINATOR
                                        ;
038D                                    BLOCKMSG:
038D    0D 0A                                   DB      CR,LF
038F    42 6C 6F 63                             DB      'Block discrimination field  = '
0393    6B 20 64 69
0397    73 63 72 69
039B    6D 69 6E 61
039F    74 69 6F 6E
03A3    20 66 69 65
03A7    6C 64 20 20
03AB    3D 20
03AD    00                                      DB      TERMINATOR
                                        ;
                                        ;
                                                END
```

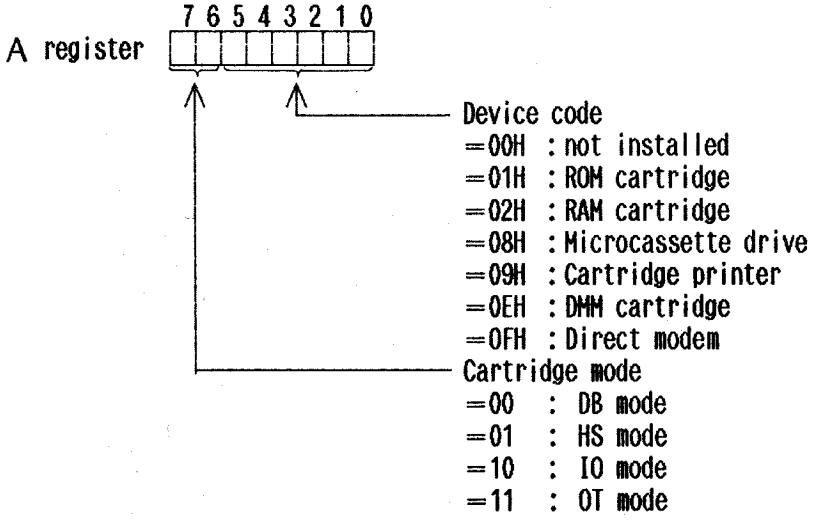
(33) CHKMOD (ØØ63H)

Function: Checks cartridges.

Entry parameter: None

Return parameter:
A : Return information (cartridge mode, device code)

Explanation:
1) All registers except the A register retain their previous values.
2) CHKMOD checks the state of the cartridges which are currently installed and returns control to the calling program with the results loaded in the A register.



Programming notes
1. PINE OS supports the HS, IO, and DB modes. It checks for the presence of the CSEL signal from the IO Status Register (P16H) to see whether the cartridge interface is in the HS mode. If the interface is not in the HS mode, OS tests the higher 4 bits of the Cartridge Status Register (P13H). The cartridge is in the IO mode if the upper nibble of the Cartridge Status Register is ØEH or more and otherwise in the DB mode.
2. The polarity of the CRS cartridge reset signal is specified by CRSTPTN (ØEFC1H).
3. See Section 5.1, "Cartridges" and Part I 4.1, "Cartridge Interface" for details on PINE cartridges.