Part II SOFTWARE

CONTENTS (SOFTWARE)

CHAPTER 1 GENERAL SYSTEM INFORMATION II-1

1.1 Introduction II-1
1.2 System Configuration II-2
1.3 Software Organization II-4

CHAPTER 2 SYSTEM FUNCTIONS II-5
2.1 Relationship of System Functions to OS II-7
2.2 System Initialize II-12
2.3 Reset II-14
2.4 Power-on II-14
2.5 Power-off II-22
2.6 Power Fail II-26
2.7 Auto Start II-28
2.8 Auto Power-off II-30
2.9 Alarm/Wake II-34
2.10 Menu II-40
2.11 System Display II-46

CHAPTER 3 CPM I (BDOS, BIOS) II-56
3.1 PINE CP/M II-57
3.2 BDOS Operations II-68
3.3 BIOS Operations II-82
3.4 BIOS Details II-95
3.5 Keyboard II-182
3.6 LCD Display (CONOUT Details) II-230
3.7 MTOS/MIOS Operations II-284
3.8 Disk Image Storage II-339
3.9 I/O Byte II-365

CHAPTER 4 CP/M II (Extended CP/M) II-366
4.1 User BIOS II-368
4.2 Jump Tables II-381
4.3 Hooks II-429
4.4 Bank Switching II-449
4.5 Resident Processing II-457
4.6 Executing a ROM-based Program II-461
4.7 Interrupts II-472

CHAPTER 5 I/O OPERATIONS II-499
5.1 Cartridges II-502
5.2 Serial Interfaces II-561
5.3 Disk Drive Unit II-562
5.4 External RAM Disk II-572
5.5 Barcode Reader and External Cassettes II-582
5.6 ROM Capsules II-586
5.7 Centronics Interface II-590
5.8 Printer II-592

CHAPTER 6 MEMORY II-595
6.1 Memory Maps II-596
6.2 System Area Details II-600

CHAPTER 7 MISCELLANEOUS TOPICS II-678
7.1 Versions II-679
7.2 DIP Switches II-682
7.3 Guide for Programming ROMs II-686

CHAPTER 1 GENERAL SYSTEM INFORMATION

1.1 Introduction II-1
1.1.1 Extended CP/M V2.2 ..... II-1
1.1.2 Item Keyboard II-1
1.1.3 PINE Devices II-1

1.2 System Configuration II-2

1.3 Software Organization II-4

CHAPTER 1 GENERAL SYSTEM INFORMATION

1.1 Introduction

1.1.1 Extended CP/M V2.2

The PINE uses an extension of CP/M V2.2 as its operating system.

PINE main memory is 64K bytes in size, part of which can be used as a RAM disk. A RAM disk allows for fast file access, and can be used in the same way as a floppy disk. A RAM disk, ROM capsule, microcassette, and ROM and RAM cartridges are available in place of floppy disks. The PINE is furnished with RAM disk and BASIC ROM capsule as standard. The other devices are optional.

In PINE CP/M, application programs on a ROM capsule can be executed in main memory, as in the standard CP/M. In PINE CP/M, however, they can also be executed directly in the ROM capsule, thus improving mamory efficiency and economizing on power requirements.

1.1.2 Item Keyboard

The PINE OS supports an optional item keyboard, as standard, which allows the system to be used exclusively for specific applications. Since the item keys on the item keybaord are arranged in a matrix and can be redefined, the user can use the keyboard for his or her special purposes. The user can replace the item keyboard with the standard keyboard at any time.

1.1.3 PINE Devices

The PINE supports a variety of devices as shown below:

(1) LCD
A virtual screen of 80 columns by 25 lines, a system screen, and user-defined character definitions are supported.

(2) Keyboard
The keyboard shift status is indicated by one of the LEDs. The keyboard supports auto repeat, N-key rollover, and function key (five) features. The user can define cursor key codes and the keys with a key code from 40H to 5EH and 60H to 7EH. The PINE also supports the item keyboard, whose item keys can be redefined.

(3) Disk drives
The PINE is designed so that part of external memory can be used in the same way as disk drives.

1) ROM capsule
Programs in a ROM capule can be executed either in RAM or ROM.

2) Microcassette
The PINE supports MTOS (file management on BDOS level) as standard and provides MTOS function calls in addition to ordinary BDOS function calls. Multiple write, retry, and verify features are also supported for microcassette processing.

3) ROM cartridges, RAM cartridges, and RAM disks can be used in the same way as floppy disks. ROM cartridges store read-only files. RAM cartridges and disks permit fast file access.

(4) Other devices and interfaces
In addition to above, the PINE supports devices such as floppy disks, clock, buzzer, printer, RS-232C interface, SIO (serial interface), and so on.

1.2 System Configuration

The PINE comprises power supply unit, keyboard, LCD, and interfaces. Figure 1.2.1 illustrates the PINE system configuration.

The PINE employs two CPUs: Z-80 as the main CPU and 7508 as the slave CPU. It has a four-bank memory configuration. The PINE's memory consists of 64KB of RAM, 32KB of OS ROM, and up to two 32KB ROM capsules.

The 4-bit slave CPU (7508) controls the keyboard, clock, and power supply. The main CPU (Z-80) controls the RS-232C, SIO, cartridge and Printer interfaces. It also controls optional cartridges such as microcassette, ROM and RAM cartridges, and cartridge printer.

The PINE also supports an external RAM disk unit connected to the sytem bus.

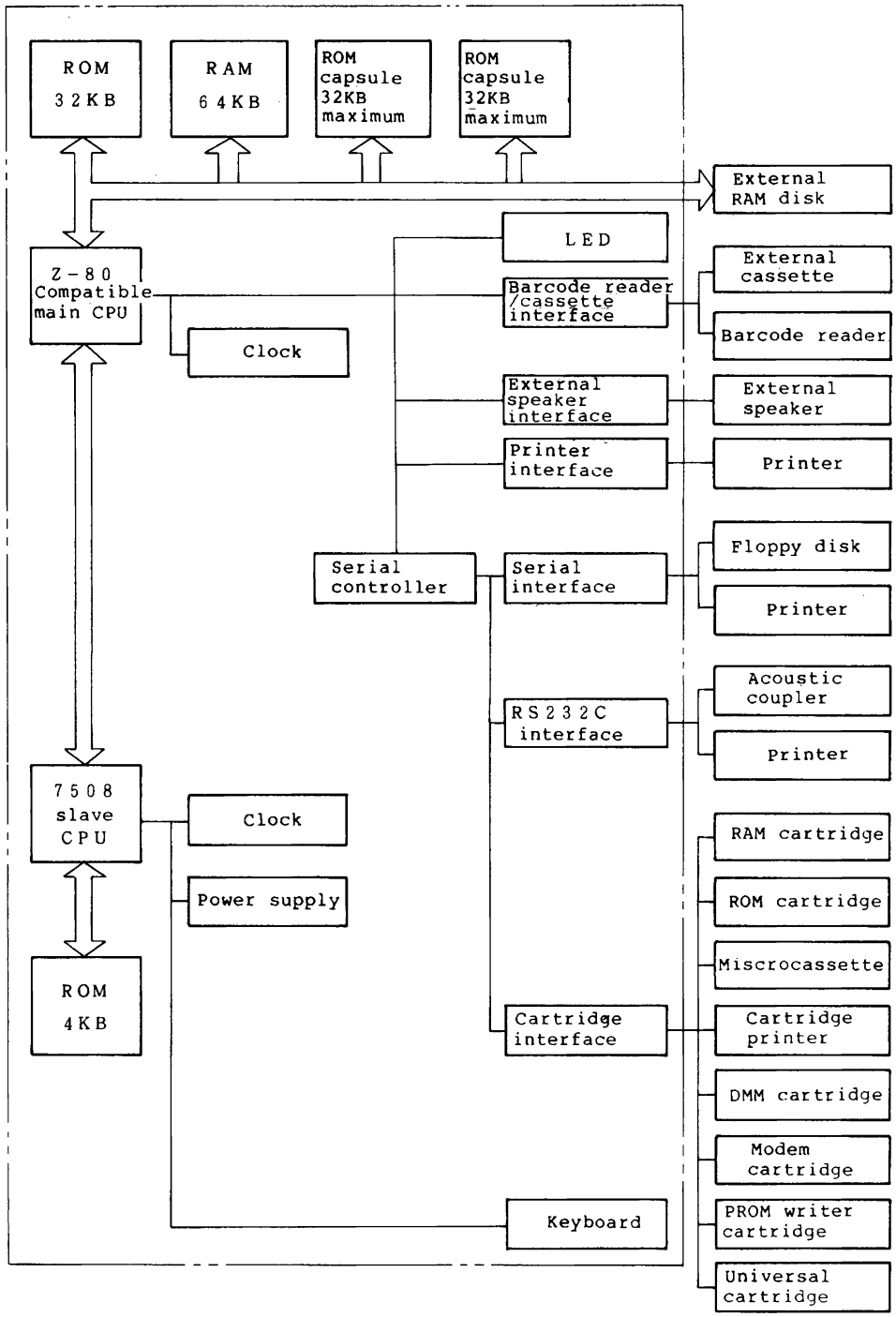


Fig. 1.2.1 System Configuration

1.3 Software Organization

The PINE software is physically stored in two places: OS ROM (32KB) and 75Ø8 internal ROM (4KB). See Chapter 2, "75Ø8 General" in PART I, hardware for information about the 75Ø8.

The PINE OS consists of the function modules below. Section numbers enclosed in parentheses indicate the sections of this manual in which the corresponding modules are described.

(1) Starter (2.2 - 2.8)
Performs system activation, initialization, reset, or power-on processing.

(2) Interrupt handler (4.7)
Handles 75Ø8 (keyboard, 1 sec, alarm, power-related), serial reception, ICF, OVF, and EXT interrupts.

(3) Menu (2.1Ø)
Displays executable program files or user-specified files on the menu screen. The user can start a desired program by selecting it from the menu.

(4) System display (2.11)
Permits checking or setting up of a PINE status condition or manual handling of a microcassette.

(5) Clock (2.9)
Performs alram or wake processing.

(6) CCP
Is the CP/M command processor containing six buit-in commands (DIR, ERA, REN, TYPE, SAVE, and USER).

(7) BDOS (3.2)
Controls CP/M disk files. BDOS handles ROM capsules, RAM disks, and ROM and RAM cartridges exactly the same way as it handles disks.

(8) MTOS (3.7)
Controls microcassette files. MTOS corresponds to CP/M BDOS.

(9) BIOS (3.3 - 3.6)
Is an I/O interface between CP/M and hardware.

(1Ø) MIOS (3.7)
Is an I/O interface between CP/M and microcassette.

(11) Relocator
Relocates resident programs in RAM.

(12) Resident processor
Performs part of interrupt and bank switching processing.

CHAPTER 2 SYSTEM FUNCTIONS

2.1 Relationship of System Functions to OS II-7
2.1.1 General II-7
2.1.2 System Flow II-8
2.1.3 State Transition II-9
2.1.4 Replacing the Keyboard II-11

2.2 System Initialize II-12
2.2.1 General II-12
2.2.2 Invoking the System Initialize Routine II-12
2.2.3 Internal Processing II-12
2.2.4 System Initialize Screen II-13

2.3 Reset II-14
2.3.1 General II-14
2.3.2 Invoking the Reset Processing Routine II-14
2.3.3 Internal Processing II-14
2.3.4 Miscellaneous II-16

2.4 Power-on II-18
2.4.1 General II-18
2.4.2 Starting Power-on Processing II-18
2.4.3 Internal Processing II-18
2.4.4 Restart Mode II-20
2.4.5 Continue Mode II-20
2.4.6 Miscellaneous II-21

2.5 Power-off II-22
2.5.1 General II-22
2.5.2 Causes of Power-off II-22
2.5.3 Internal Processing II-22
2.5.4 Miscellaneous II-25

2.6 Power Fail II-26
2.6.1 General II-26
2.6.2 Starting Power Fail Processing II-26
2.6.3 Internal Processing II-26
2.6.4 Miscellaneous II-27

2.7 Auto Start II-28
2.7.1 General II-28
2.7.2 Defining/cancelling an Auto Start String II-28
2.7.3 Auto Start Processing II-28
2.7.4 Miscellaneous II-30

2.8 Auto Power-off II-30
2.8.1 General II-30
2.8.2 Defining the Auto Power-off Function II-30
2.8.3 Operation II-31
2.8.4 Miscellaneous II-32

2.9 Alarm/Wake II-34
2.9.1 General II-34
2.9.2 Setting or Resetting the Alarm/Wake Function II-34
2.9.3 Operation II-36
2.9.4 Miscellaneous II-39

2.10 Menu II-40
2.10.1 General II-40
2.10.2 Operation II-40
2.10.3 Menu Specifications II-41
2.10.4 Miscellaneous II-45

2.11 System Display II-46
2.11.1 General II-46
2.11.2 Operation II-46
2.11.3 Internal Processing II-52
2.11.4 Miscellaneous II-54

CHAPTER 2 SYSTEM FUNCTIONS

2.1 Relationship of System Functions to OS

2.1.1 General

The PINE OS has the following unique functions:
(1) System initialize
(2) Reset
(3) Alarm, wake
(4) Menu
(5) System display

This section describes the reletionship of these functions to the operating system. See the following sections for details of individual functions.

2.1.2 System Flow

This subsection discusses the operation flow from the activation of the system to the return of control to CP/M.

The system is started when one of the following is performed:
(1) System initialization
(2) Reset
(3) Power-on
(4) BOOT
(5) WBOOT
(6) Alarm/wake

The operation flow after the system is started is shown in Figure 2.1.1.

When system is activated by the system initialize, reset, power-on, or alarm/wake functions, the address Ø start processing routine in OS ROM reads the 75Ø8 status and invokes the corresponding processing routine.

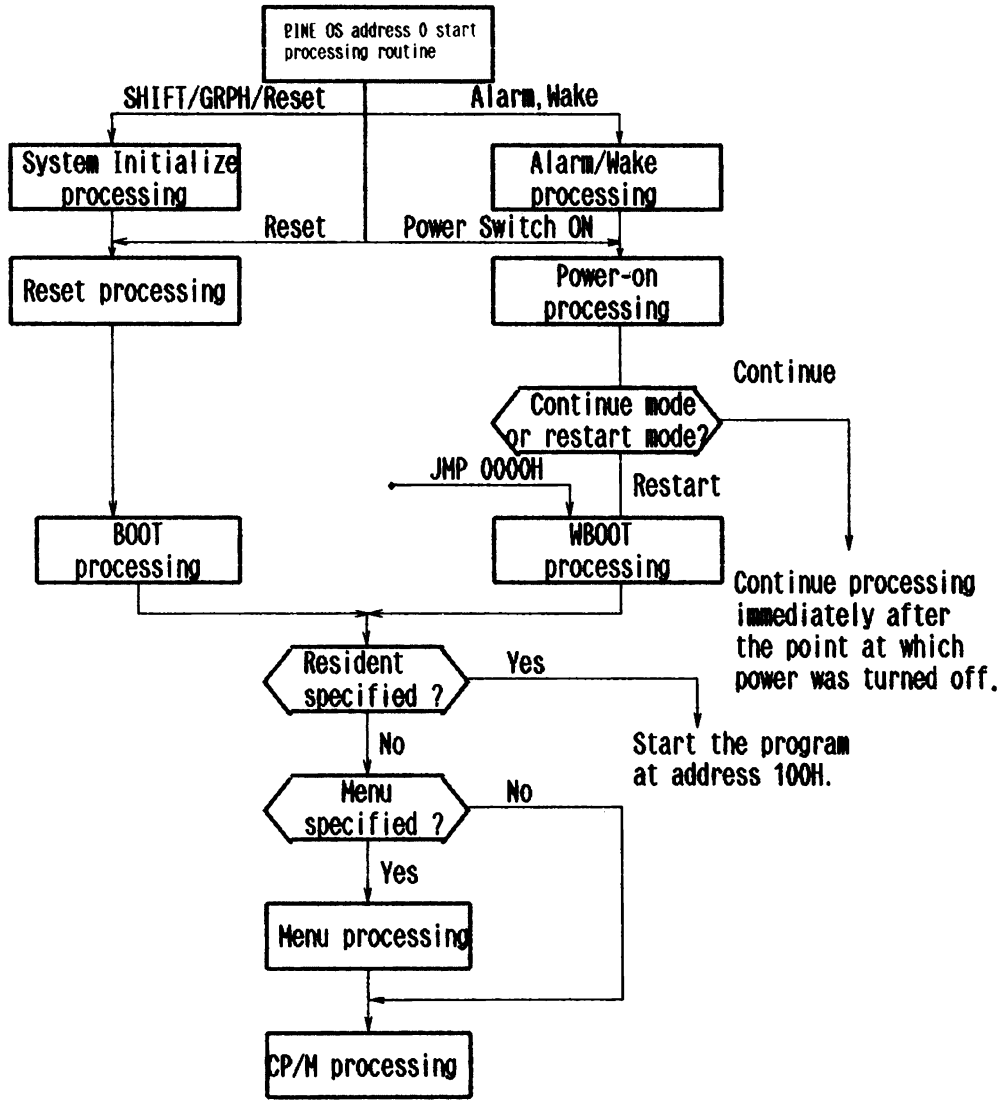


Fig. 2.1.1 System Operation Flow

2.1.3 State Transition

Figure 2.1.2 illustrates the state transition of the PINE.

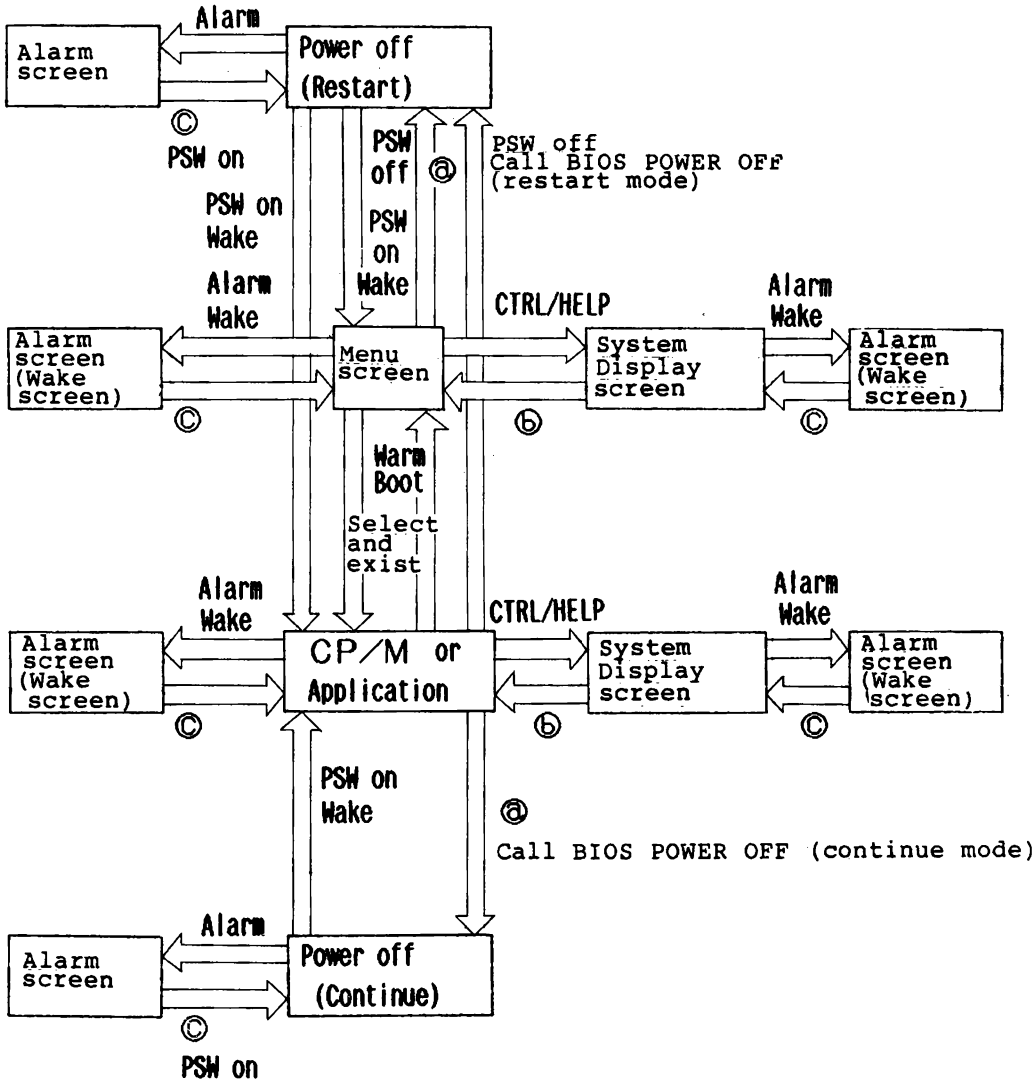
The PINE is always in one of the following six states:
(1) Power is off (restart mode).
(2) Power is off (continue mode).
(3) CP/M (or an application) is in execution.
(4) Menu screen is displayed.
(5) Alarm (wake) screen is displayed.
(6) System display screen is displayed.

The power-off state is either in the restart or continue mode depending on the conditions in which power was turned off.

When power is turned on from the restart mode power-off state, processing starts with the menu screen displayed or in the CP/M state. When power is turned on from the continue mode power-off state, processing resumes in the CP/M state that was established when power was turned off.

If power is turned off during a state transition from CP/M to the system display or alarm/wake screen, control is once again returend to CP/M. CP/M determines the power-off mode by checking the cause of power-off, and turns off power accordingly.

See Section 2.5 for power-off, Section 2.10 for menu, and Section 2.11 for system display.



ⓐ CTRL/PSW off, Auto power off, Power failure

ⓑ CTRL/PSW off, Auto power off, Power failure, ESC, PSW off

© CTRL/PSW off, 50 seconds elapsed, Power failure, ESC, PSW off

Fig. 2.1.2 State Transition Diagram

2.1.4 Replacing the Keyboard

In PINE, the user can easily replace the stnadard keyboard with the item keyboard.

The PINE OS checks the keyboard state during power on to determine whether or not the keyboard has been replaced (automatic keyboard identification).

If the standard keyboard has been replaced, the OS performs keyboard initialization and starts reset processing.

Figure 2.1.3 shows the operation flow of automatic keyboard identification. Table 2.1.4 lists the differences between the standard and item keyboards. See Section 3.5, **"Keyboard" for** hardware restrictions for individual keyboards.

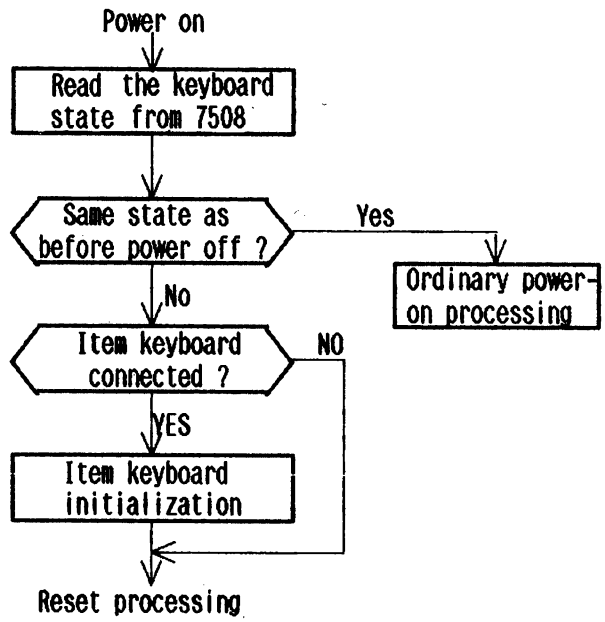


Fig. 2.1.3 Automatic Keyboard Identification

Item \ Keyboard type	Standard keyboard	Item keyboard
Menu screen display drive	C, B, A	A, B, C, D, H, I, J, K
Auto repeat function	Provided	Not provided
Power-off mode	Restart mode	Continue mode

Table 2.1.4 Differences Between the Standard and Item Keyboards

2.2 System Initialize

2.2.1 General

The PINE executes the system intialize routine to reset the execution environment when the system is started, program execution hangs up, or the RAM contents is destroyed by battery failure. The entire system resident area is initialized when the system initialize routine is executed.

2.2.2 Invoking the System Initalize Routine

The system initialize routine is invoked when:

(1) The RESET button is pressed and released while the right SHIFT and GRPH keys are held down. (Standard keyboard)
(2) The RESET button is pressed and released while the INIT and STOP keys are held down. (Item keyboard)
(3) The slave CPU RESET button on the main unit rear panel is pressed.
(4) Power is turned on for the first time after the external RAM disk is connected or disconnected.
(5) Power is turned off during system intialization and turned on again.
(6) An error is detected during error checking at power-on or reset time. Errors include the following:
    1) The checksum obtained at the last power-off does not match the checksum obtained at current power-on.
    2) The sum of the internal RAM disk size and the user BIOS area size exceeds 35.5K bytes. (Reset time)
    3) The address calculated from the CCP, RBDOS1, and RBIOS1 load addresses and their sizes does not match the starting address of the internal RAM disk. (Reset time)

2.2.3 Internal Processing

The system carries out the following during system initilaization:

(1) Sends a Keyboard Reset command to the 75Ø8 and initalizes the keyboard-related values stored in the 75Ø8.
(2) Initializes system work area RSYSAR1.
(3) Cold boots the system devices. The cold boot procedure consists of the following steps:
    1) Initialize the data in the system resident area (loading of data).
    2) Warm boot the sytem devices. (Check for a RAM disk, ROM capsule, and ROM and RAM cartridges and establish the sytem environment accordingly.)
    3) Enable 1-second interrupts.
    4) Disable alarm interrupts.
    5) Read in DIP switch data and initialize the keyboard-related parameters according to the keyboard type and language setting.
    6) Turn on LCD screen display and enables cursor blinking.
(4) Displays the system initialize screen to permit the user to set up the following:
    1) Year, month, day, hour, minute, and second
    2) Day of the week
    3) Internal RAM disk size
    4) User BIOS size
(5) Clears the auto start string or alarm message.
(6) Passes control to the reset processing routine (see Section 2.3, "Reset" for reset processing).

2.2.4 System Initialize Screen

The system initalize screen shown below appears when the system initalize routine is invoked:

```
SYSTEM INITIALIZE

(1) DATE/TIME (MMDDYYhhmmss) 053084153740
(2) WEEK (0 to 6)  3
(3) RAM DISK SIZE  (×1   kB)  26
(4) USERBIOS SIZE  (×256 B)  000
```

Repond to the system prompt as follows:

(1) Specify each of the year, month, day, hour, minute, and second with a two-digit number. Enter the last two digits of the year. Time must be specified in the 24-hour system.
(2) Specify the day of the week with a number Ø to 6.
Ø: Sunday, 1: Monday, ... , 6: Saturday
(3) Specify the internal RAM disk size with a two-digit number. The internal RAM disk size must be speicified in 1 KB increments and must be ØØ or Ø2 to 35. 26 (on the screen) is the system default. The system will ask for reentry if Ø1 or a value larger than 35 is entered.
(4) Specify the user BIOS size with a three-digit number in 256 byte units. ØØØ (on the screen) is the system default.

Note the following when entering the above data:

(1) Specify items (1) through (4) in that order.
(2) When the external RAM disk is connected, the internal RAM disk is made invalid (i.e., its size is automatically set to Ø byte) and no request for RAM disk size is displayed.
(3) The total size of the internal RAM disk and the user BIOS must be smaller than 35.5K bytes. If the sum exceeds 35.5K bytes, the system asks for reentry.
(4) Be careful when entering data because no check is made for invalid data (specifying 13 for month or 7 for day of the week).
(5) Data other than numbers or entries from other than cursor movement keys are not accepted.

The system initailize routine transfers control to the reset processing routine when items (1) through (4) are specified.

2.3 Reset

2.3.1 General

The system executes the reset processing routine when the RESET button on the side panel of the PINE main unit is pressed or when the keyboard is repladed. The reset processing routine initailizes the entire system resident area except for RSYSAR1 (the area initialized during system initialization).

2.3.2 Invoking the Reset Processing Routine

The reset processing routine is invoked when:
(1) The RESET button is pressed and released in power-on state.
(2) The power switch is turned on or the alarm/wake time is reached after a power failure occurred in power-off state.
(3) The power switch is turned on after the keyboard was replaced in power-off state.

Control is transferred to system initialize processing when the RESET button is pressed while the following keys are held down: Right SHIFT and GRAPH (when the standard keyboard is installed) or INIT and STOP (when the item keyboard is installed.)

2.3.3 Internal Processing

The system carries out the following during reset:

(1) Checks the shift state of the currently pressed key and passes control to the system initailize processing routine when only the right SHIFT and GRPH keys (standard keyboard) or the INIT and STOP keys (item keyboard) are pressed.
(2) Sends a Keyboard Reset command to the 7508 and initalizes the keyboard-related values stored in the 7508.
(3) Cold boots the system devices. The cold boot procedure consists of the following steps:
    1) Initialize the data in the system resident area (loading of data).
    2) Warm boot the sytem devices. (Check for installation of the RAM disk, ROM capsule, and ROM and RAM cartridges and establish the sytem environment accordingly.)
    3) Enable 1-second interrupts.
    4) Enable or disable alarm interrupts according to the current alarm/wake settings.
    5) Read in DIP switch data and initialize the keyboard-related parameters according to the keyboard type and language setting.
    6) Turn on LCD screen display and enable cursor blinking.
(4) Checks the sizes of the internal RAM disk and user BIOS area and passes control to the system initialize routine if their sum exceeds 35.3K bytes.
(5) Passes control to the reset processing routine if the address calculated from the CCP, RBDOS1, and RBIOS1 load addresses and their sizes does not match the starting address of the internal RAM disk.
(6) Updates the data in the disk parameter block for each of the currently installed devices (RAM disk, ROM capsule, ROM cartrige or RAM cartridge).
(7) Performs a sum check on the internal RAM disk. If a checksum mismatch occurs, the sytem displays a message asking for RAM disk formatting.

The system transfers control to BIOS BOOT when the above procedure is completed.

Figure 2.3.1 shows the operation flow of reset processing.

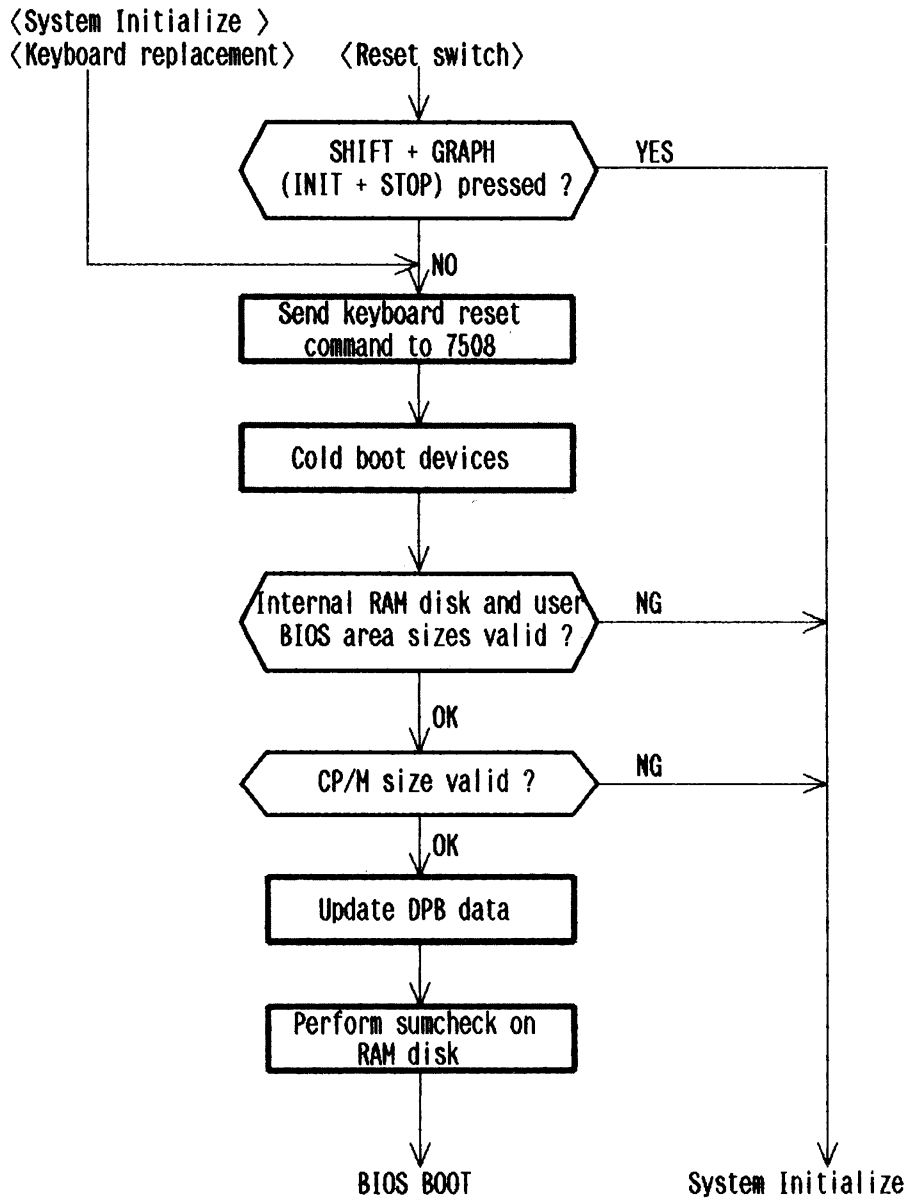


Fig. 2.3.1 Reset Processing Flow

2.3.4 Miscellaneous

2.3.4.1 RAM disk checking

(1) General
Since the RAM disk uses part of internal RAM, its contents can be unexpectedly destroyed by an application program that is being executed in internal RAM. Therefore, a sum check must be made during each read. The entire RAM disk must be checked at warm boot time. The system prompts the user to format the disk if a checksum error is detected.

(2) RAM disk formatting
The RAM disk is formatted in the following cases:
1) The system is initialized (when an external RAM disk is installed).
2) The RESET button is pressed, or power is turned on (power switch is set to ON or the wake time is reached) when:
- The internal RAM disk size is not zero and an internal RAM disk checksum error is detected (when no external RAM disk is connected); or
- The RESET button is pressed during external RAM disk read/write processing (when an external RAM disk is connected).

(3) Formatting procedure
1) The system prompts "RAM DISK FORMAT (Y/N) ?."
2) If the user response is Y, the system formats the RAM disk (file). If the user response is N, the system does nothing.

(4) Note
The contents of the RAM disk are not guaranteed if the user enters N. That is, a Bad Sector error may occur when the RAM disk is accessed. In such a case, the user must initialize the system and format the RAM disk. When an internal RAM disk is insatlled, no message is displayed during system initialization because the internal RAM disk is formatted automatically.

2.3.4.2 Initializing the sytem areas

Figure 2.3.2 shows when the system areas are initialized.

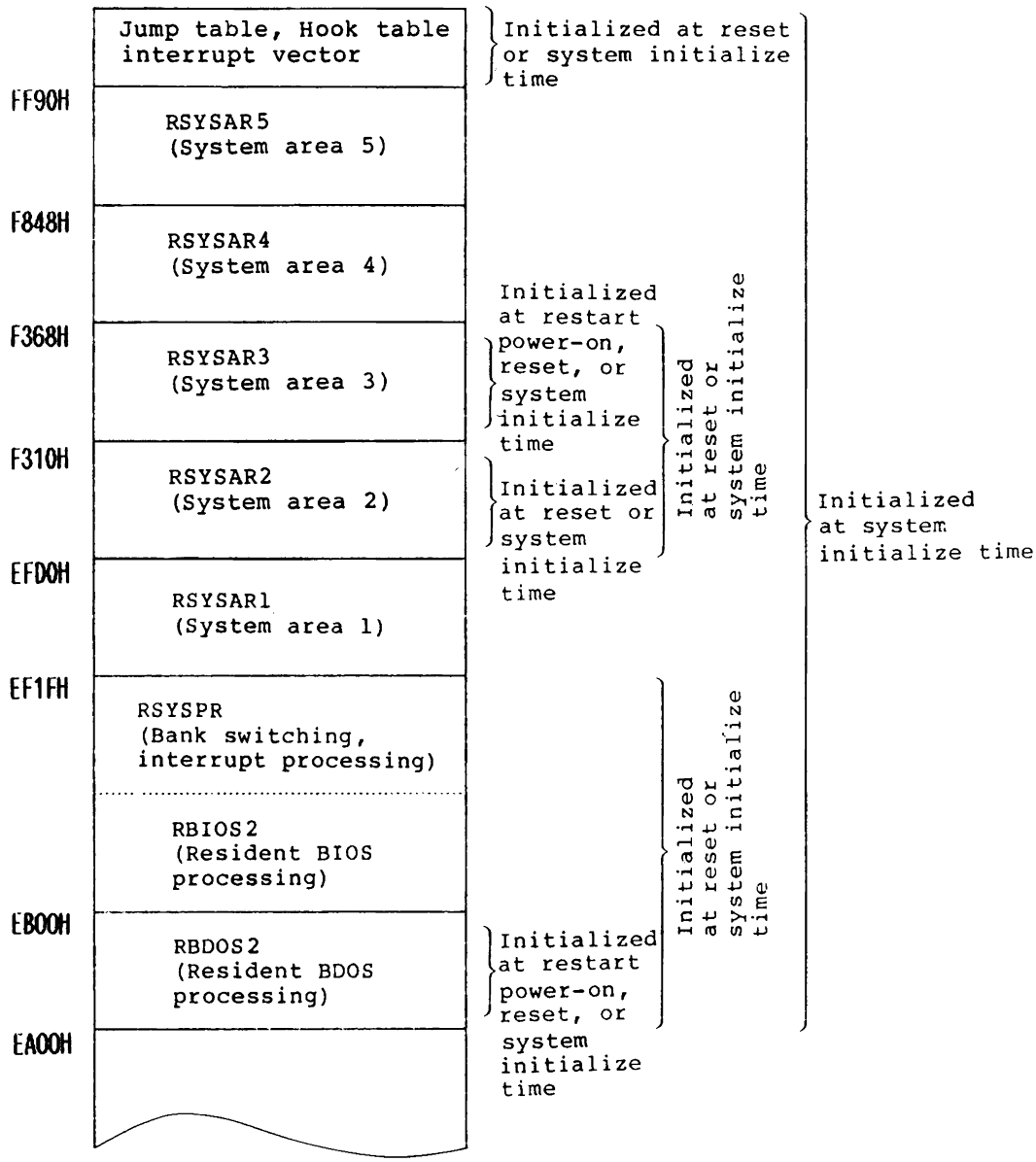


Fig. 2.3.2 System Areas and Their Initialization Times

2.4 Power-on

2.4.1 General

The power-on sequence is initiated when the power switch is turned on from the PINE power-off state. Power-on processing takes place in two modes: restart and continue. The power-on processing mode is determined by the method in which power was turned off.

2.4.2 Starting Power-on Processing

Power-on processing starts when:
(1) The power switch is turned on from the power-off state.
(2) The alarm time is reached in the power-off state and the power switch is turned on while the alarm screen is dispalyed.
(3) The wake time is reached in the power-off state. In this case, the wake string is used in place of the auto start string.

2.4.3 Internal Processing

The operations that the system performs during power-on include the following:
(1) Warm boots the system devices. Warm boot processing consists of the steps below.
    1) Restore the I/O port status to its original status at power-off.
    2) Check for cartridge installation.
    3) Check for ROM capsules installation.
    4) Check for ROM cartridge installation.
    5) Check for RAM cartridge installation.
    6) Check for RAM disk installation.
    7) Update the disk parameter blocks based on the results of steps 3) to 6).
(2) Transfers control to the sytem intialize processing routine if the RAM disk capacity does not match that calculated before power-off.
(3) Setting to check the type of keyboard installed, and transfers control to the reset processing routine if it does not match the keyboard type defined before power-off (automatic keyboard identification).
(4) Switches the screen to the user mode and displayes accordingly.
(5) Makes a sum check on the internal RAM disk. If a checksum mismatch is found, the system requests that the RAM disk be formatted.

(6) Checks the power-off mode and transfers control to the continue power-on processing routine if power was turned off in the continue mode, and to the restart power-on processing routine if power was turned off in the restart mode.

(7) The restart power-on processing routine generates a power-on sound, initializes the screen size and cursor type, and passes control to BIOS WBOOT.

(8) The continue power-on processing routine restores the system to the alarm state at power-off and generates a continue sound. It then restores the interrupt-related areas and registers to their state at power-off, and returns control to the point at which power was turend off.

Figure 2.4.1 shows the power-on processing flow.

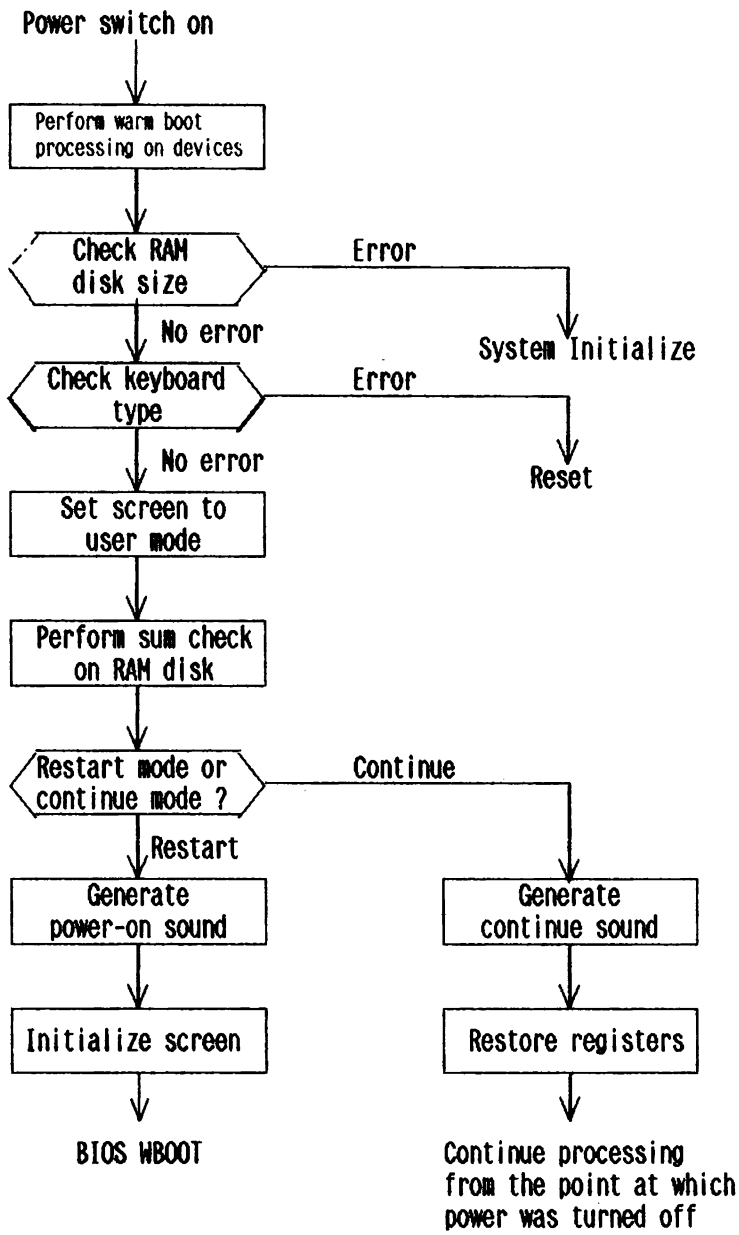


Fig. 2.4.1 Power-on Processing Flow

2.4.4 Restart Mode

2.4.4.1 Outline

The PINE turns off in the restart mode when the power switch is turned off while it is running CP/M or an application program. If the item keyboard is used, however, the power-off state defaults to continue mode when the power switch is turned off.

The PINE unconditionally turns off in the restart mode is power is turned off while a menu is displayed on the screen (even if the item keyboard is used). It starts in the restart mode whenever power is turned on from the restart mode power-off state.

2.4.4.2 Activation from restart mode power-off state

The PINE performs the following operations when started from the restart mode power-off state:

(1) When the power switch is turned on:
Passes control to 100H of bank 0 when Resident is specified. If an auto string is specified, the PINE passes it as a starting parameter (by loading it into the key buffer, as PF key pressing).
When Resident is not specified, the PINE transfers control to the specified menu or CCP state, and the auto start string is executed or passed as a starting parameter.
See Section 4.5, "Resident Processing" for Resident and Section 2.7, "Auto Start" for the auto start string.

(2) When alarm time is reached:
Generates an alram and displays the alarm screen. If the power switch is turned on in this state, the PINE performs the operations descrived above.

(3) When wake time is reached:
The PINE operates in the same way as it does when the power switch is turned on, except that it passes the wake string instead of the auto start string.

2.4.5 Continue Mode

2.4.5.1 Outline

If the power switch is turned off in the continue mode during execution of an application program, the next time the power is turned on the PINE can continue execution in the state established before power-off.

2.4.5.2 Activation from continue mode power-off state

The PINE performs the following when started from the continue mode power-off state:

(1) When the power switch is turned on:
Continues program execution at the point where power was turned off. Ignores the auto start string.

(2) When alarm time is reached:
Generates an alram and displays the alarm screen. If the power switch is turned on in this state, the PINE performs the operations descrived above.

(3) When wake time is reached:
Operates in the same way as it does when the power switch is turned on except that it ignores the wake string.

2.4.6 Miscellaneous

2.4.6.1 System operation flow from power-off to power-on

Figure 2.4.2 shows the system operation flow from power-off to power-on.

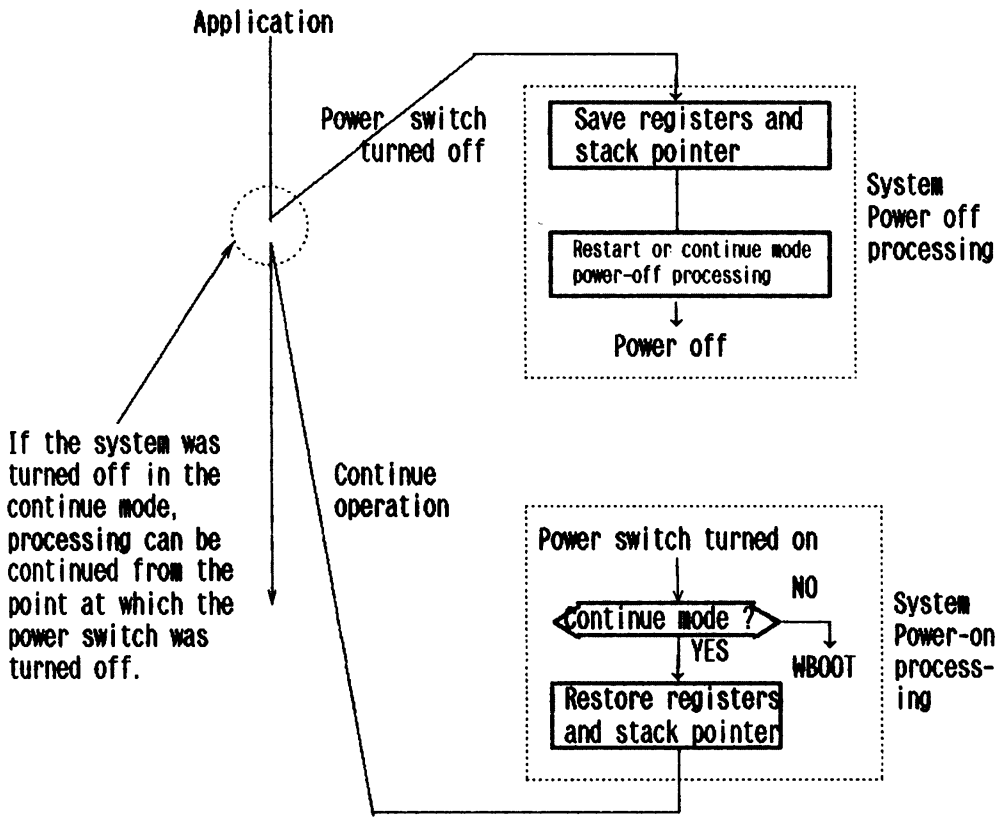


Fig. 2.4.2 Operation Flow from Power-off to Power-on

2.5 Power-off

2.5.1 General

The PINE turns off when 1) the power switch is turned off, 2) a power failure occurrs, 3) the auto power-off time is reached, or 4) BIOS POWEROFF is called.

Before performing power-off processing, the system saves information that will be required the next time that power is turned on. This section gives an outline of power-off processing.

For further information, see Section 2.6, "Power Fail," Section 2.8, "Auto Power-off," Section 3.4, "BIOS Details," and Section 4.7, "Interrupts."

2.5.2 Causes of Power-off

The causes of power-off and the actions performed by the system are as follows:

(1) The power switch is turned off.
Performs power-off processing according to a power off interrupt received from the 7508.

(2) A power failure occurrs.
Performs power-off processing according to a power fail interrupt received from the 7508 if the battery voltage drops below a specified value.

(3) The auto power-off time is reached.
Performs continue-mode power-off processing if a specific period of time (default is five minutes) elapses while BIOS CONIN is waiting for key entry.

(4) BIOS POWEROFF (WBOOT + 7BH) is called.

2.5.3 Internal Processing

Power-off is processed in three modes: continue, restart, and power-cut only modes.

2.5.3.1 Continue mode power-off

(1) Processing
The system saves parameters that are necessary to continue processing when power is turned on again, and carries out power-off processing. In the continue mode, execution can resume from the point at which power was turned off.

(2) Conditions
Power is turned off in the continue mode if one of the following conditions is met.

1) FRCECNTN (0F311H) is not 00H (when using the standard keyboard) or IFRCECNT (0F312H) is not 00H (when using the item keyboard).
2) FRCECNTN or IFRCECNT is 00H and:
   - BIOS POWEROFF is called in the continue mode.
   - Auto power-off time is reached.
   - The power switch is turned off with the CTRL key held down.
   - A power failure occurrs.

If, however, power is turned off during menu processing, the system turns off in the restart mode, even if condition 1) or 2) is satisfied.

2.5.3.2 Restart mode power-off

(1) Processing
The system carries out power-off processing so that a warm boot can be performed the next time power is turned on. Fundamental operations in continue power-off and restart power-off processing are the same, although system parameters are initilaized differently.

(2) Conditions
Power is turned off in the restart mode if one of the following conditions is met.
1) Menu processing flag is set to ON.
2) FRCECNTN (ØF311H) for the standard keyboard or IFRCECNT (ØF312H) for the item keyboard is ØØH and:
   - BIOS POWEROFF is called in the restart mode.
   - The power switch is turned off without the CTRL key held down.

2.5.3.3 Power-cut only mode

(1) Processing
The power-cut only mode is used exclusively by the system. In this mode, power is turned off in the mode established at the last power-off time (continue or restart).

(2) Conditions
Power is turned off in the power-cut only mode when one of the following conditions is met.
1) The alarm time is reached while power is off and then a power-off condition occurs while the alram screen is being displayed (i.e., control is transferred from the alarm screen through a cause other than power switch-on).
2) A power-off condition occurs during execution of address Ø start processing. Address Ø start processing includes system initialize, reset, and power-on processing.

2.5.3.4 Power-off processing

The system performs the following operations when power is turned off:
1) Writes buffer data onto the floppy disk.
2) Skips steps 3) thorugh 5) if Power-cut only is specified.
3) Saves registers.
4) If the sytem is processing a menu, initializes screen-related areas and performs restart power-off processing. If the system is not processing a menu, it checks whether power was turned off in the continue mode or restart mode.
5) Clears parameters if it is determined in the above step that power was turned off in the restart mode.
6) Performs power-off processing on cartridges.
   - Saves MCT farmware data in RAM if a microcassette is installed.
   - Turns cartridge power off if a ROM cartridge is insatlled.
7) Performs power-off processing on the keyboard.
   - Clears the key buffer.
   - Saves the current shift status.
8) Creates a power fail screen using the sytem screen if power fail processing is specified.
9) Makes a sum check on the sytem areas.

10) Displays the power fail screen and performs power fail processing if power fail processing is specified.
11) Sends a power-off command to the 7508 and turns off the main power.

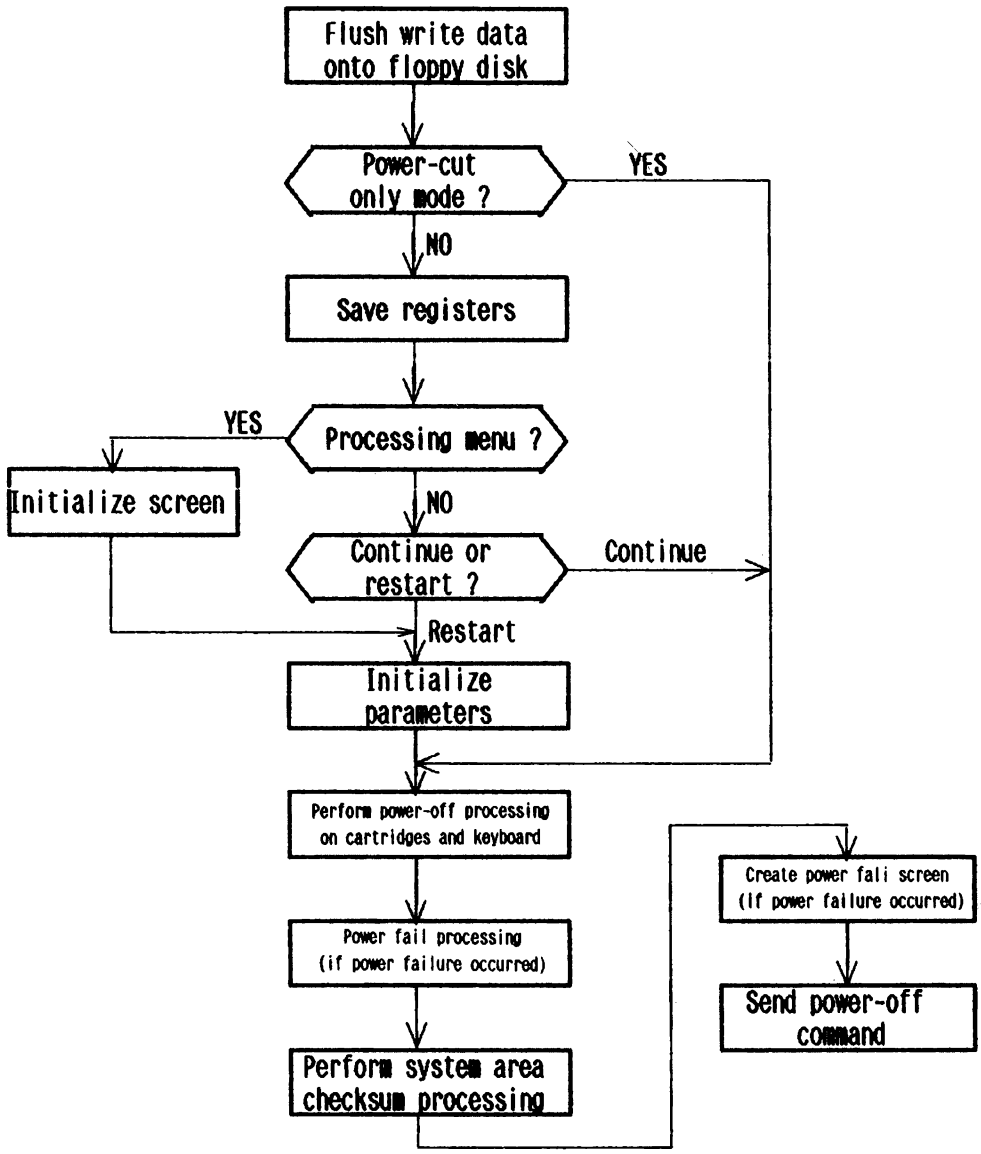


Fig. 2.5.1 Power-off Processing Flow

The power-off processing routine references the following system areas:

FRCECNTN (ØF311H) 1 byte
- Standard keyboard continue flag
  = ØØH: Specifies restart power-off
  = Nonzero: Specifies continue power-off
- Initial value is ØØH.

IFRCECTN (ØF312H) 1 byte
- Item keyboard continue flag
  = ØØH: Specifies restart power-off
  = Nonzero: Specifies continue power-off
- Initial value is ØFFH. That means that, the default power-off state is the continue mode, if an item keyboard is installed.

BTRYFG (ØEFEEH) 1 byte
- Power fail flag
  = ØØH: No power fail occurred
  = Nonzero: Power fail occurred
- This flag is set by the power fail interrupt processing routine.

2.5.4 Miscellaneous

The user can control power-off in an application program using the power-off disable flag YPOFDS (OEFEFH). See Section 4.7, "Interrupts" for controlling power-off.

2.6 Power Fail

2.6.1 General

A power fail condition occurs when the main battery voltage falls below a specified value (4.8 V for a NiCd battery and 4.Ø V for an Mn dry sell battery).

The system informs the user that a power fail has occurred by displaying the power fail screen.

2.6.2 Starting Power Fail Processing

Power fail processing starts when the 75Ø8 slave CPU sends a power fail interrupt to the main CPU.

2.6.2.1 Slave CPU operations

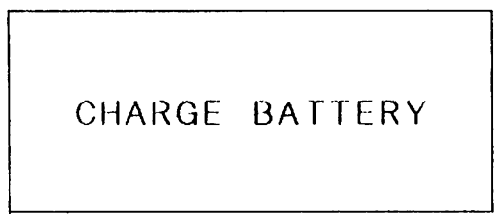
When the main battery voltage falls below approximately 4 V (dry cell battery) or 4.8 V (NiCd), the 75Ø8 sends a power fail interrupt to the main CPU to switch it to the back-up battery. The 75Ø8 then generates a power fail interrupt every second, and cuts off the power supply in 5Ø seconds if the main CPU does not perform power-off processing.

2.6.2.2 Main CPU operations

The main CPU, on receiving a power fail interrupt, performs power-off processing after completing the current I/O operation. See Section 4.7, "Interrupts" for power fail interrupts.

2.6.3 Internal Processing

When a power fail occurs, the system informs the user by displaying the following power fail screen after having completed continue-mode power-off processing (except for actually turning off power).



The system flashes the power fail screen on and off every second, and turns off power when any of the following conditions are met:

(1) 3Ø seconds pass after the initial display of the power fail screen.
(2) The user turns off the power switch after display of the power fail screen.
(3) 5Ø seconds pass after a power fail is detected.

The power fail screen is displayed via the system screen; the user screen is not affected. See Section 2.5, "Power-off" for power-off processing flow.

2.6.4 Miscellaneous

2.6.4.1 Power fail and alarm/wake functions

The PINE disables the alarm/wake funciton when a power fail occurrs and restores the previous alarm/wake state when the power switch is turned on again. That is, any alarm/wake conditions are ignored between a power fail and the next power-on.

2.6.4.2 Restoring the system after power-off

The user can restore the system by turning the power switch on after 1) connecting an AC adapter to the system, or 2) replacing the main battery.

2.6.4.3 Continue mode processing after a power fail

Processing terminated by a power fail can be continued by the system restoration procedure mentioned above if any of the following conditions is met:

(1) Power is automatically turned off 30 seconds after the initial display of the power fail screen.
(2) The user sets the power switch to OFF while the power fail screen is displayed.
(3) The slave CPU cuts off power while the power fail screen is displayed.

If the current I/O operation is not completed within 50 seconds after a power fail, the slave CPU cuts off power even before the power fail screen is displayed. In this case, the system performs reset processing (identical to that performed when the reset button is pressed) instead of continue processing, even if the above restoration procedure is followed.

2.6.4.4 Power fail in the power-off state

Backing up the 7508 and RAM consumes power, even if the PINE is in the power-off state. If the system is held in the power-off state for a long time, the main battery voltage may fall below the specified level, and the back-up battery may replace the main battery. When this occurs, power will not turn on when the power switch is set to ON; power can be turned on by the restoration procedure described in 2.6.4.2. In this case, reset processing is performed.

A sum check is always made on the system area at power-on time. If the checksum does not match the value calculated at power-off time, it is assumed that the memory was destroyed while power was off, and the system is initilaized.

2.7 Auto Start

2.7.1 General

When a warm boot is invoked by a power switch-on, the system loads the key buffer with a predefined auto start string and carries out subsequent processing as if it were entered from the keyboard. The auto start function is convenient for users who wish to run a specific program every time they start their system, or those who want to use the PINE as a turnkey system.

2.7.2 Defining/cancelling an Auto Start String

An auto start string can be defined or cancelled by 1) using the system display, 2) calling BIOS AUTOST (WBOOT + 81H), or 3) rewriting the corresponding system area.

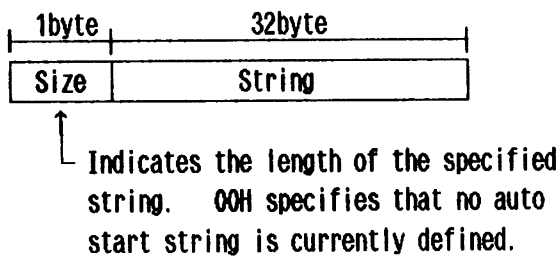
An auto start string may consist of up to 32 characters, including control codes. Control codes can be entered as in the example below:
Example: Entering a Carriage Return (CR) code
Type ^M (or ^m) when using the system display.
Type ØDH when using BIOS AUTOST.
In either case, code ØDH is stored in memory.

As in the above example, any control code entered via the system display must be a combination of ^ and an alphabetic character.

See Section 2.11, "System Display" for details of the system display and Section 3.4, "BIOS Details" for BIOS AUTOST.

The auto start string area AUTOSTRT (ØF3BDH) is the system area for containing an auto start string. An auto start string can be defined or cancelled by directly rewriting this area.

AUTOSTRT (ØF3BDH) 34 bytes
- Auto start string area



This area is initialized during system initialization.
34th byte do not use.

2.7.3 Auto Start Processing

2.7.3.1 Invoking the auto start function

The auto start function is invoked when the power switch is turned on either in the restart-mode power-off state or while the alarm screen is being displayed.

2.7.3.2 Auto start function operations

When the auto start function is invoked, the auto start stirng is loaded into the key buffer in the same way as a PF key definition, and power-on processing is performed. This sequence is illustrated in Figure 2.7.1.

2.7.3.3 Termination of the auto start function

Since the auto start string is handled in the same way as PF key definitions, the auto start function does not terminate if a STOP code (Ø3H) is received. The STOP code is passed to the application program.

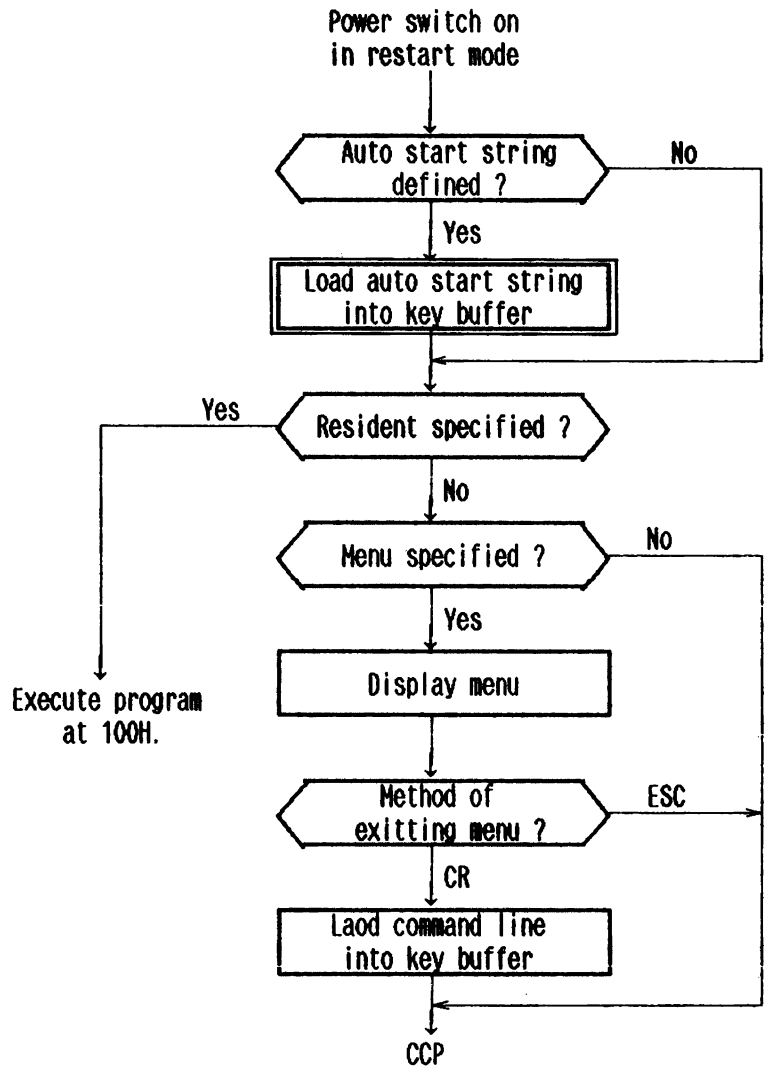


Fig. 2.7.1 Auto Start Function Processing Flow

2.7.4 Miscellaneous

2.7.4.1 Passing the auto start string

When Resident is specified:
The auto start function causes the application program to be directly executed regardless of whether or not Menu is specified. The function loads a predefined auto start string into the key buffer in the same way as a PF key definition.

When Resident is not specified:
If Menu is specified, the auto start string is treated as if it were entered from the keyboard to a menu command line. If CCP is specified, the auto start string is treated as if it were entered from the keyboard (that is, it is loaded into the CCP command buffer).

2.7.4.2 Menu function and auto start string

The menu function loads the current command line data into the key buffer when a CR is entered. Therefore, if a CR (ØDH) code is contained in an auto start string, the function takes only the part of the string ending with the CR, and ignores any data following the CR. (This is also true of a wake string.)

2.8 Auto Power-off

2.8.1 General

To save power, the PINE accepts keyboard input in the sleep mode. In addition to that, it automatically turns its own power off if it receives no data from the keyboard within a predetermined waiting period. This feature is called the auto power-off function.

This section describes how to define, operate, and disable the power-off function.

2.8.2 Defining the Auto Power-off Function

The interval between the time the PINE starts waiting for keyed-in data and the time it automatically turns off its power is set to five minutes by default. It can be redefined by rewriting the work areas ATSHUTOFF (ØEF40H) and ATSOTIME (ØEF41H).

ATSHUTOFF (ØEF40H) 1 byte
- Contains the auto power-off time in minutes.
- A ØØH in this area disables the auto power-off function.
- The initial (default) value is Ø5H (5 minutes).
- The system references this area only when determining whether or not the auto power-off function is disabled.

ATSOTIME (ØEF41H) 2 bytes
- Contains the auto powr-off time in seconds.
- ATSOTIME has the following relationship to ATSHUTOFF:
  ATSOTIME = ATSHUTOFF x 6Ø
  However, it can also be defined independently of ATSHUTOFF.
- The initial value (default) is Ø12CH (3ØØ seconds).

2.8.3 Operation

When the auto power-off function is activated, the PINE automatically turns off its power in the continue mode if it receives no data from the keyboard whithin the time specified in ATSOTIME (ØEF41H) while waiting for key input with the BIOS CONIN function. When the power switch is turned off and on again, or the wake time is reached, the PINE is restored to the previous state (key input wait state).

The following key input routine determines whether or not the auto power-off time has been reached. It uses the 16-bit counter TIMERØ (ØEF8FH) for 1-second interrupt:

(1) Set an auto power-off time.
The sum of ATSOTIME (ØEF41H) and TIMERØ (ØEF8FH) is loaded into TIMEEND (ØF77CH) as the time at which the auto power-off function is to be executed.

(2) Check for keyed-in data.
The PINE determines whether any data has been entered by checking the key buffer. If keyed-in data is present, auto power-off checking processing terminates.

(3) HALT
The CPU enters into the sleep state via a HALT instruction. It exits from that state when it receives an OVF or 75Ø8 1-second interrupt.

(4) Check auto power-off time.
If the value of ATSHUTOFF (ØEF4ØH) is nonzero and the M-flag set or reset depending on the result of (TIMEEND - TIMERØ) is 1, The PINE determines that the auto power-off time has been reached.

(5) Power off.
The PINE turns off power in the continue mode if the auto power-off time has been reached. If not, it loops back to step (2).

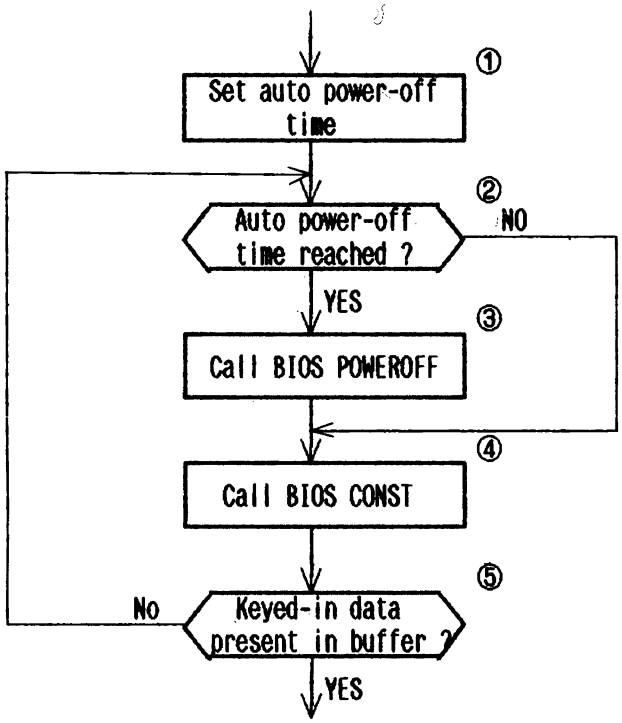
An auto power-off time is set at the entry or exit point of the sytem display or alarm/wake screen, or at the time of a continue-mode power-on.

The PINE turns on in the restart mode if the auto power-off time is reached during display of the menu screen.

2.8.4 Miscellaneous

2.8.4.1 Executing the auto powr-off function in an application

The auto power-off function may not be available for application programs which do not use CONIN for receiving console input, but rather receive console input by polling the keyboard port with CONST. To make the auto power-off function available for such programs, add the routine shown below to the application programs and so that they can execute the auto power-off function independently.



Note: Power can be conserved adding a HALT instruction before step ②.

Fig. 2.8.1 Auto Power-off Processing Flow

Step	Action	Description
1	Set auto power -off time	The sum (2 bytes) of ATSOTIME (ØEF41H) and TIMERØ (ØEF8FH) is loaded into TIMEEND (ØF77CH) as the auto power-off time.
2	Check auto power-off time	It is determined that the auto power-off time has not been reached if the value of ATSHUTOFF (ØEF4ØH) is ØØH. If it is nonzero, the difference of TIMEEND and TIMERØ is calculated. If the M-flag is found to be 1 as the result of this calculation, it is determined that the auto power-off time has been reached.
3	Call BIOS POWEROFF	If the auto power-off time has been reached, BIOS POWEROFF is called to turn off power.
4	Call BIOS CONST	BIOS CONST is called to check for keyed-in data.
5	Check for keyed-in data	The application program checks the parameters returned in step (4) to determine whether any entry has been made.

2.9 Alarm/Wake

2.9.1 General

The PINE is provided with alarm and wake functions. The alarm function sounds an alarm and displays the alarm screen at the preset time. The wake function powers on the PINE at the preset time from the power-off state and performs predefined operations.

This section explains how to set or reset the alarm or wake function. Their operations are also descirbed here.

The alarm/wake function is provided with several hooks through which the user can extend the function (see Section 4.3, "Hooks"). Display of the alarm/wake screen can be temporarily disabled (see Section 4.7, "Interrupts").

2.9.2 Setting or Resetting the Alarm/Wake Function

The PINE is furnished with a 75Ø8 4-bit CPU which controls the software timer (clock) and generates interrupts to the Z8Ø CPU at specified times. Even in the power-off state, the 75Ø8 is activated at every 1Ø seconds to check whether a preset alarm/wake time has been reached. When the time is reached, the 75Ø8 automatically powers on the PINE.

The alarm/wake function can be set or reset using the system display or by calling the BIOS TIMDAT (WBOOT+4BH) function. Setitng of the alarm/wake function via the sytem display can be disabled by rewriting ALRMSTDS (ØEF43H).

The following items must be specified for the alarm/wake function:

(1) Alarm/wake time (in month/day/hour/minute/second/day of the week format. The second must be set in 1Ø-second increments.)
(2) Alarm/wake string (up to 32 characters)
(3) Alarm/wake type (specify alarm or wake)

See Section 2.11, "System Display" for system display and Section 3.4, "BIOS Details" for BIOS TIMDAT.

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

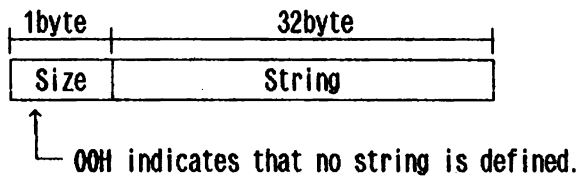
ALRMSTDS (ØEF43H) 1 byte
- Flag for enabling or disabling setting of the alram/wake function via the system display.
  = ØØH: Setting or cancelling via system display is enabled.
  = Nonzero: Setting or cancelling via system display is disabled.
- The initial value is ØØH.

ALRMTP (ØEF8AH) 1 byte
- Alarm/wake type
  = ØØH: Not specified (default)
  = ØlH: Alarm specified
  = Ø2H: Wake specified

ALRMAD (ØEF8BH) 2 bytes
- Alarm/wake string starting address
- Always points to ALRMMSG (ØF3DFH).

ALRMST (ØEF8DH) 1 byte
- Indicates whether or not the alarm/wake time has been reached.
  = ØØH: Not reached (default)
  = ØlH: Reached

ALRMMSG (ØF3DFH) 34 bytes
- Alarm/wake string area
- The first byte is loaded with the string size.



2.9.3 Operation

2.9.3.1 General

The alarm/wake function executes in different ways depending on whether PINE power is off or on. How alarm/wake processing proceeds in both power-off and power-on states is summarized below.

Type	PINE power state		
	Restart-mode power-off state	Continue-mode power-off state	Power-on state
Alarm	* Displays the alarm screen and turns power off in the restart mode. *Turns power on in the restart mode when the power switch is turned on while the alarm screen is displayed.	*Displays the alarm screen and turns power off in the continue mode. *Turns power on in the continue mode when the power switch is turned on while the alarm screen is displayed.	* Displays the alarm screen. (The display timing is checked in actual oepration.)
Wake	*Sounds an alarm and performs restart mode power-on processing. If a wake string is defined, performs processing accordingly.	*Sounds an alarm and performs continue mode power-on processing. Ignores any wake string.	*Displays the wake screen (The display timing is checked in actual oepration.)

2.9.3.2 Alarm/wake screen

The alarm/wake screen is displayed whenever the alarm time is reached, or when the wake time is reached in the power-on state.

(1) Displaying the alarm/wake screen
The alarm/wake display routine generates an alarm and displays the following alarm/wake screen:

<ALARM TIME> MM/DD hh:mm

"Alarm/Wake string(32 characters)"

Press ESC Key

In the wake screen, <WAKE TIME> appears in place of <ALARM TIME>.

(2) Terminating display of the alarm/wake screen
When the alram/wake screen is displayed, any of the following conditions causes the screen to be restored to its original state before the alarm/wake function was invoked:

1) 50 seconds pass after the screen was displayed.
2) The ESC key is pressed.
3) The power switch is turned off when it is turned on.
4) A power fail is detected.

When the alarm function is invoked in the power-off state the following condition also brings about the same result:

5) The power switch is turned on.

(3) Notes

(a) Keyboard shift state
Once the alarm/wake screen is displayed in the power-on state, the keyboard shift state is determined by the state in which control leaves the alarm screen. Key data that had been loaded into the key buffer before the screen was displayed is processed in the shift state established before the initial display of the screen.

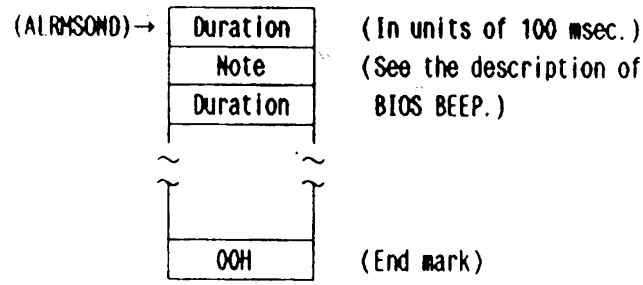
(b) Screen mode
The alarm/wake screen is displayed via the system screen. The screen mode reverts to its original state (user screen) when the display of the alarm/wake screen is terminated.

(c) Alarm/wake screen display time
The alarm/wake screen display time (50 seconds) is measured using the 16-bit 1-second interrupt timer TIMER0 (0EF8FH). The display time can be changed by rewriting ALRMTIME (0F308H)

(d) Alarm sound
The pattern of the alarm sound generated when the alarm/wake time is reached can be changed by rewriting ALRMSOND (0EF3EH). ALRMSOND is loaded with the address at which the the pattern of the alarm sound is stored. When rewriting ALRMSOND in an application program, the new address must be 8000H or up.

ALRMTIME (ØF3ØH) 1 byte
- Contains the time interval for which the alarm/wake screen is to be displayed.
- The time must be specified in seconds. Initial value is 32H (5Ø seconds).

ALRMSOND (ØEF3EH) 2 bytes
- Alarm sound pattern starting address



- An alarm consists of three occurrences of the above sound pattern.
- ALRMSOND initially points to an OS ROM address where sound pattern data Ø1H, 25H, Ø1H, 29H, ØØH is stored.

2.9.3.3 Alarm processing

The operation of the alarm processing routine differs when an alarm condition occurs in the power-off state and power-on state.

(1) When an alarm condition occurs in the power-off state
If an alarm condition occurs in the power-off state, the alarm processing routine turns on the main power, sounds an alarm, and displays the alarm screen.

When control leaves the alram screen, the alram processing routine turns off power again in the mode in which power was turned off the last time (restart or continue mode). If, however, the alarm screen is terminated by turning on the power switch, the routine performs power-on processing instead.

Two hooks, ALMHKØ and ALMHK1, are used during processing of an alarm condition that occurs in the power-off state. See Section 4.3, "Hooks" for details.

(2) When an alarm condition occurs in the power-on state
If an alarm condition occurs in the power-on state, the 7508 sends an alarm interrupt to the system, and the system calls the alarm interrupt handling routine.

The alarm interrupt handling routine sounds an alarm and displays the alram screen if alarm screen display is enabled.

After leaving the alarm screen, the system returns to the state in which the alarm occurred.

The ALMHK2 and ALMHK3 hooks are used during procssing of an alarm condition that occurs in the power-on state. The ALMHK4 is provided for alarm display processing. See Section 4.3, "Hooks" for more information.

2.9.3.4 Wake processing

The operation of the wake processing routine differs when an wake condition occurs in the power-off and power-on state.

(1) When a wake condition occurs in the power-off state
If a wake condition occurs in the power-off state, the wake processing routine turns on the main power, sounds an alarm, and calls the power-on processing routine.

The power-on processing routine performs restart mode power-on processing if power was turned off in the restart mode. If a wake string is defined, the routine passes control to CCP or the Menu. If power was turned off in the continue mode, the routine performs continue mode power-on processing. Any wake string is ignored.

(2) When a wake condition occurs in the power-on state
If a wake condition occurs in the power-on state, the 7508 sends an alarm interrupt to the system, and the system calls the wake interrupt handling routine.

The operation of the wake interrupt handling routine is the same as that of the alarm interrupt handling routine described in 2.9.3.3. (2)

2.9.4 Miscellaneous

2.9.4.1 Alarm/wake functions and power fail

The alarm/wake function is disabled when a power fail occurs. The alarm/wake state before the power fail occurred is resotred when the power switch is turned on the next time. Note that no alarm/wake processing occurs during power-off due to a power fail.

2.9.4.2 Menu function and wake string

The menu function loads the current command line data into the key buffer when a CR is entered. Therefore, if a CR (0DH) code is contianed in a wake string, the function takes only the part of the string ending with the CR, and ignores any data following the CR.

2.9.4.3 Inhibiting alarm/wake screen display from application programs

Display of the alarm/wake screen can be controlled from application programs using the alarm/wake display disable flag YALMDS (ØEFFlH). See Section 4.7, "Interrupts" for control methods.

2.9.4.4 Alarm/wake hooks

The alarm/wake funtion has five hooks. These hooks allow the function to be extended with user-supplied routines such as an alarm/wake automatic update routine. See Section 4.3, "Hooks" for the location and use of the hooks.

2.10 Menu

2.10.1 General

The menu function displays a directory of files with specified extensions in specified drives. It allows the user to select and start a desired program easily.

2.10.2 Operation

2.10.2.1 Activation and termination

(1) Activation
Menu processing starts when menu display is specified and one of the following conditions is met:
(a) The power switch is turned on in the restart mode or power is turned on by the wake function. (The menu function is not started if Resident processing is specified or SUBMIT processing is being performed.)
(b) A warm boot is performed in an application program (even if it is invoked through CCP). Warm boots performed during SUBMIT processing are excluded.

The menu function can be specified using menu flag MENUFG (ØEF44H). Whether or not SUBMIT is being executed can be determined using the BDOS Reset disk system function (ØDH).

(2) Interruption
Menu processing is interrupted when:
(a) Alarm or wake time is reached.
(b) System display is invoked.
(c) Screen dump is invoked.

Neither (b) nor (c) can occur if the item keyboard is installed.

(3) Continuation
When the PINE is released from the interrupt state, menu processing resumes from the point at which it was interrupted. If the specified drive or file name was changed while operating in the system display mode, menu processing starts from the beginning.

(4) Termination
Menu processing terminates when:
(a) The ESC key is pressed and the system enters the CP/M command mode.
(b) The return key is pressed after an application program is selected from the menu.
(c) Auto power-off time is reached.
(d) A power fail is detected.
(e) The power switch is turned off.

Menu processing is always terminated in the restart mode if it is terminated by power-off. This is because file names to be displayed are changed if the ROM capsule or cartridge is replaced while power if off.

2.10.3 Menu Specifications

2.10.3.1 Menu-related parameters

The following parameters can be specified for menu display:
1) Whether or not a menu is to be displayed.
2) The drives whose directories are to be displayed in the menu.
3) The file name extensions, COM file names, and drive names of the files that are to be displayed in the menu.

These parameters can be redefined by using the system display function or rewriting the corresponding system area. See Section 2.11, "System Display" for details.

(1) Menu display
Whether or not a menu is to be displayed can be specified using MENUFG (0EF44H).

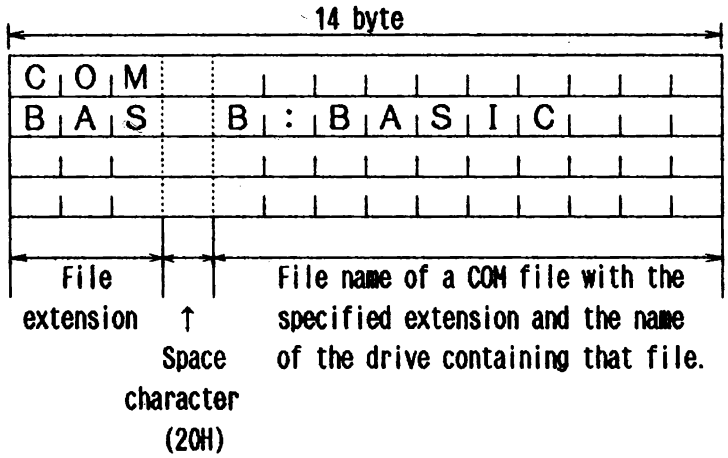
(2) Menu drives
The menu function displays the directories of the drives specified in MENUDRV(0EF45H). The directories are displayed in the same order in which the drive names are specified in MENUDRV. If the same drive name is specified two or more times, its directory is displayed the specified number of times.

Up to 11 drives can be specified. Remaining byte positions in the drive input field must be padded with spaces (20H).

(3) Menu files
The menu function displays the names of files having the extensions specified in the file table FTYPETBL (0EF50H).

File specifications must include a file extension. A file specification may consist of a maximum of 14 bytes. Up to four files may be specified at one time.

The structure of FTYPETBL is as follows:



Unused table entries or any extra byte positions left over after a file name is specified must be padded with spaces (20H).

The file and drive names (bytes 5 - 14) which follow a COM specification (bytes 1 - 3) are ignored.

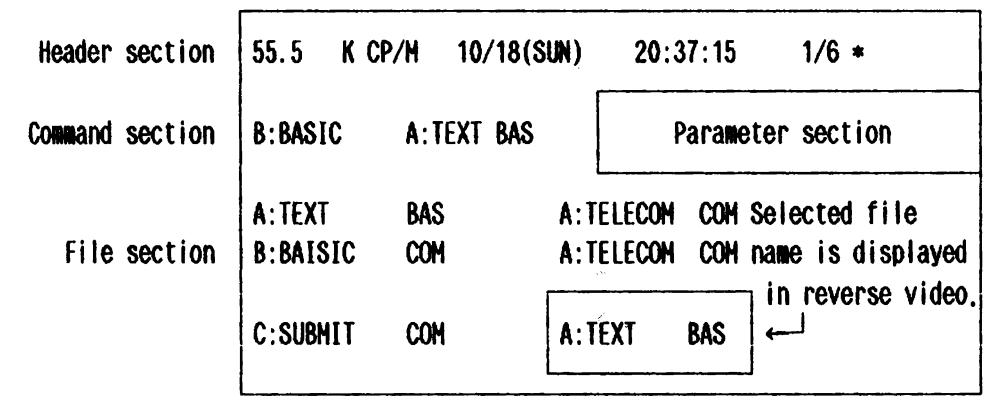
MENUFG (ØEF44H) 1 byte
- Menu display flag
    = ØØH: Menu displayed (default).
    = Nonzero: No menu displayed.
- Control goes directly to CCP if MENUFG is nonzero.

MENUDRV (OEF45H) 11 bytes
- Table containing the drive names of the drives whose directories are to be displayed in the menu.
- Specify drive names A through K in ASCII code.
- This area defaults to CBA if the standard keyboard is installed, and to ABCDHIJK if the item keyboard is installed.

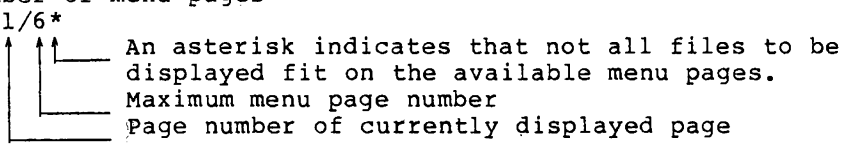
FTYPETBL (ØEF5ØH) 56 bytes
- Table containing the file types and the file names (including the drive name) of the files that are to be displayed in the menu.
- A file specification, which must include an extension, may be up to 14 bytes in length. Up to four files can be specified.
- Files must be specified in ASCII code.
- This area is initially loaded with 'COM' and 'BAS B:BASIC'.
- The structure of FTYPETBL is described above.

2.1Ø.3.2 Menu screen

The following figure shows an example of the menu display:



(1) Contents of header section
- CP/M sign-on message
- Month, day, day of the week, time (24-hour system)
- Number of menu pages



(2) Contents of command section
- The command section displays the selected files. When a file with a COM extension is selected, only its drive and file names are displayed.
- When an extension other than COM is specified, file names with that extension are displayed, preceded by the COM files associated with the extension (if any are specified in FTYPETBL).
- Parameter section

Entry of 14 to 40 characters is allowed. The allowable number of characters differs depending on the selected file type. If entry parameters do not fit in the command section, enter them via CCP.

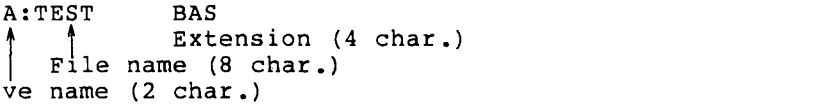
(3) Contents of file section
- Up to 12 file names can be displayed in a menu page. The user can select a desired file using arrow keys.

2.10.3.3 File specifications

(1) Menu display conditions
- Names of files whose drive names (A - K) are cataloged via the system display ca be displayed in the menu.
- If specified files are not present, the system displays no menu and goes to CCP.
- If an error (not ready, Bad sector, etc.) is detected in specified drives, names of files in those drives are not displayed.
- When the item keyboard is installed, names of files in drives A through D and H through K can be displayed.

(2) Number of files to be displayed
- 2 files per line
  12 files per page
  Maximum number of pages: 6 pages
  Maximum number of displayable files: 72 files
- If more than 72 files are specified, the extra files are not displayed. In this case, an asterisk is appended to the end of the header section.

(3) File display
- File names are displayed in the format below.



- The currently selected file is displayed in reverse video.

(4) File selection key

- A file can be selected using the following keys:
  ↑: Selects a file in an higher row of the screen. Does nothing if the cursor is positioned on a file in the top row.
  ↓: Selects a file in a lower row. Does nothing if the cursor is positioned on a file in the bottom row.
  →: Selects a file on the right. If the cursor is positioned on the right file, moves it to the left file in the row below. If the cursor is on the right file in the bottom row, moves it to the left file in the top row.
  ←: Selects a file on the left. If the cursor is positioned on the left file, moves it to the right file on the row above. If the cursor is on the left file in the top row, moves it to the right file in the bottom row.
- The menu screen can be switched using the following keys:
  SHIFT/↓: Moves the menu screen to the next page. Does nothing if the last page is currently displayed. When the item keyboard is used, the function of this key sequence differs depending on the SHIFT key state.
  SHIFT/↑: Moves the menu screen to the previous page. Does nothing if the first page is currently displayed. When the item keyboard is used, the function of this key sequence differs depending on the SHIFT key state.
  CTRL/↑: Moves the menu screen to the first page. Does nothing if the first page is currently displayed. (This key sequence is invalid when the item keyboard is installed.)
- When the menu page is switched, the file in the upper left corner of the new page is always selected.
- All parameters on the command lines are cleared when file selection is changed or the display page is moved.

2.10.3.4 File activation

When the user presses the return key after selecting a desired application program with the arrow keys, the PINE transfers control to CCP with the contents of the command section loaded in the key buffer.

The application program is started if the parameters entered for it are valid.

2.10.4 Miscellaneous

(1) A menu is displayed via the 25-row by 80-column user screen. The underline cursor is used in the menu screen. When menu processing is terminated, the screen size and cursor type must be initialized using ESC sequence data stored in CONSCRN1 (0EFF3H).

(2) A menu is always displayed on the LCD irrespective of the I/O byte setting for the CON: device.

2.11 System Display

2.11.1 General

The system display refers to the screen through which the user can set or reference the system status. The system display function is called from a BIOS key routine as a subroutine when the CTRL/HELP keys are pressed.

The user can set or reference items pertaining to the following using the system display:
(1) RAM cartridge
(2) Alarm/wake
(3) Auto start string
(4) Menu
(5) Microcassette drive

If the item keyboard is installed, the system display function cannot be activated by pressing the CTRL/HELP keys.

2.11.2 Operation

2.11.2.1 Activation and termination

(1) Activation
The system display function is called from a BIOS key routine when the CTRL/HELP keys are pressed while the PINE is in operation. This means that pressing of the CTRL/HELP keys does not activate the system display function unless a BDOS or BIOS key routine (CONST or CONIN) is being executed in an application program.
See Section 3.5, "Keyboard" for subroutine calls.

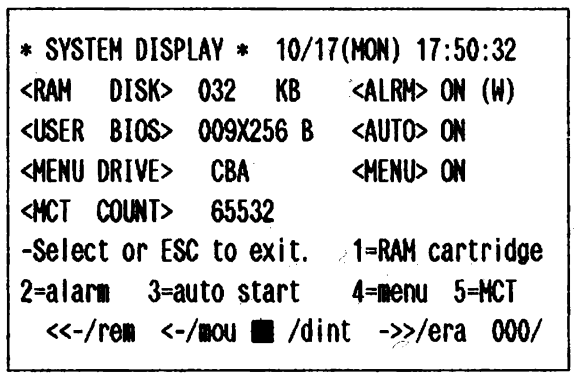
If the item keyboard is installed, system display cannot be invoked by pressing of the CTRL/HELP keys. This is because the CTRL key on the item keyboard, unlike that on the standard keyboard, does not affect the keyboard status.

(2) Interruption and restart
When an alarm or wake condition occurs or the screen dump function is invoked while the system is in the system display mode, the system terminates display processing and performs alarm, wake, or screen dump processing. System display processing resumes when the pertinent processing ends.

(3) Termination
System display processing ends when:
  (a) The ESC key is pressed while the initial screen is displayed.
  (b) The preset auto power-off time is reached.
  (c) The power switch is turned off.
  (d) A power fail is detected.
After termination of processing, the system restores the state which was operative before the system display function was called.

2.11.2.2 Operation

(1) Initial screen



(a) The date (month, day, day of the week) and time are automatically updated.
(b) The RAM DISK and USER BIOS fields display the corresponding capacities and cannot be redefined by the user.
(c) The MENU DRIVE and MENU fields can be redefined.
(d) The MCT (microcassette) COUNT field, "5=MCT" on the seventh line and the message on the eighth line are not displayed if no MCT is connected.
(e) MCT manual operations that the user can perform are displayed on the eighth line.
- The items on the eighth line are valid only when a microcassette cartridge is connected.
- The MCT COUNT field indicates the MCT counter value.
- Function keys are used for MCT operation. Currently active function keys are displayed on the eighth line (rem and mou are not displayed at the same time).
- Manual operation is disabled while a cassette file is open.
- In the remove state, operations other than remove are enabled. In the mount state, only remove operation is enabled.

  <<- (PF1): Fast forward
  <- (PF2): Wind tape
  ■ (PF3): Stop tape
  ->> (PF4): Rewind tape
  ooo (PF5): Clear tape counter
  rem (PF6): Remove processing (write tape directory)
  mou (PF7): Mount processing (read tape directory)
  dint (PF8): Initialize tape directory
  era (PF9): Erase tape

(f) Input keys
- 1 - 5: Perform corresponding processing.
- ESC: Exit from system display.
- PF1 - PF9: Perform MCT-related processing.
- Other keys: Ignored.

(g) When setting data (1 - 5 selected)
- The current state is displayed.
- BS, ←, →, and CTRL/E (erase end of line) are available as edit keys.
- All keys are enabled for entry.

(h) The ALARM field indicates whether or not the alarm/wake function is enabled. When the wake function is enabled, ON (W) is displayed in this field.
(i) The AUTO field indicates whether or not the auto start string is enabled.

(2) RAM cartridge
When 1 is selected in the initial screen, the messages on the lowest three lines change as shown below. If no RAM cartridge is connected, the system does nothing and returns control to the initial screen.

```
 - Select or ESC to return.
< RAM FORMAT >  1 = no  2 = Yes
```

(a) Selecting 1 immediately returns control to the initial screen.
(b) Selecting 2 initiates formatting of the RAM cartridge.
(c) The ESC key returns control to the initial screen.
(d) Any other keys are ignored.

(3) Alarm/wake
When 2 is selected in the initial screen, the messages on the lowest three lines change as shown below.

Initial alarm screen

```
- Select or ESC to return.
 < ALARM > 1 = off  2 = alarm  3 = wake  4 = MSG
```

(a) Selecting 1 through 4 initiates processing (b) through (e). The ESC key returns control to the initial screen. Any other keys are ignored.
(b) When 1 is selected, both the alarm and wake functions are disabled and control returns to the initial screen.
(c) When 2 is selected, the screen changes as follows:

```
- Set alarm time, ESC to cancel.
  MMDDhhmm
```

- If the alarm or wake function is already specified, the defined time is displayed in month/day/hour/minute format (each item is set with a two-digit number). If neither alarm nor wake function is specified, no time is displayed.
- Set each of the month, day, hour, and minute with a two-digit number. Items specified with an asterisk (*) are not referenced; the alarm or wake function is invoked repeatedly with these items ignored. Pressing the return key saves the display data and returns control to the initial alarm screen.
- The ESC key returns control to the initial screen.

(d) When 3 is selected, the screen changes as follows:

```
- Set Wake time, ESC to cancel.
  MMDDhhmm
```

- The only difference between the messages in (c) and (d) is that "alarm" is replaced with "wake." The time setting method is the same.
- If a time is specified for a function (alarm or wake) other than the one defined currently, the message defined for the current function is cleared.
Example: Any alarm time and message cataloged currently are cleared when a wake time is specified.

(e) When 4 is selected, the screen changes as follows:

```
- Input message/string, ESC to cancel.
```

- If an alarm message or wake string is already specified, its contents are also displayed below the above message. If neither is specified, only the above message is displayed.
- Both an alarm message and wake string may consist of up to 32 characters. A wake string may contain any characters, including control codes. A control code is specified as a combination of ^ and an alphanumeric character, and counts as two characters.
- Pressing the return key terminates processing, saves the specified message or string, and returns control to the initial auto start screen.
- The ESC key returns control to the initial screen.

(4) Auto start
When 3 is selected in the initial screen, the messages on the lowest three lines change as shown below.

```
- Select or ESC to return.
 < AUTO START >  1 = off  2 = assign
```

(a) Selecting 1 disables the auto start function and returns control to the initial screen.
(b) Any keys other than 1, 2, and ESC are ignored.
(c) The ESC key returns control to the initial screen.
(d) When 2 is selected, the screen changes as shown below.

```
- Input auto start string, ESC to cancel.
  B: BASIC^M
```

- An auto start string is displayed if one is already defined.
If a string has not been defined, a blank screen is displayed.
- An auto start string may consist of up to 32 characters. It may contain any characters including control codes. A control code is specified as a combination of ^ and an alphabetic character, and counted as two characters.
- Pressing the return key terminates processing, saves the specified message or string, and returns control to the initial auto start screen.
- The ESC key returns control to the initial screen.

(5) Menu
When 4 is selected in the initial screen, the messages on the lowest three lines change as shown below.

Initial menu screen

```
- Select or ESC to return.
< MENU >  1 = off  2 = on  3 = drive
          4 = ext1  5 = ext2  6 = ext3  7 = ext4
```

(a) Selecting 1 through 7 initiates processing (b) through (f). The ESC key returns control to the initial screen. Any other keys are ignored.
(b) When 1 is selected, control returns to the initial screen. The MENU field is set to OFF and menu display is disabled.
(c) When 2 is selected, control returns to the initial menu screen. The MENU field is set to ON and menu display is enabled at the next warm boot.
(b) When 3 is selected, the screen changes as follows:

```
- Input drive names, ESC to cancel.
  CBA
```

- The currently specified drive name is displayed.
- Enter the drive name of the drive whose directories are to be displayed (A - K). Pressing the return key returns control to the initial menu screen and the MENU DRIVE field displays the new drive name. This specification becomes valid the next time the menu function is activated.
- The ESC key returns control to the initial screen.

(e) When 4 is selected, the following message appears on the screen:

```
- Input extension name 1, ESC to cancel.
  COM
```

- The ESC key returns control to the initial screen.
- Enter data in the following format:

BAS␣B:BASIC

- BAS: Extension (3 char.)
- ␣: 1 space
- B:BASIC: File name of a COM file having the specified extension and the drive name of the drive containing the file.

The drive and file names may be omitted when specifying the COM extension.

- The return key saves the input data and returns control to the initial menu screen.

(f) The user can specify three more extensions for menu display by selecting 5 through 7.

(6) MCT (Microcassette drive)

When 5 is selected in the initial screen, the messages on the lowest three lines change as shown below.

```
- Select or ESC to return.
 < MCT >  stop   verify
  1 = stop  2 = nonstop  3 = verify  4 = nonverify
```

(a) The ESC key returns control to the initial screen.
(b) When 1 is selected, "stop" is displayed on the seventh line and the MCT operates in the stop mode.
(c) When 2 is selected, "nonstop" is displayed on the seventh line and the MCT operates in the nonstop mode.
(d) When 3 is selected, "verify" is displayed on the seventh line and the MCT operates in the verify mode.
(e) When 4 is selected, "nonverify" is displayed on the seventh line and the MCT operates in the nonverify mode.

2.11.3 Internal Processing

This subsection describes the system areas used by the system display function and system processing associated with system display.

2.11.3.1 Display operations

The system display function displays the following information:
(1) RAM disk size
Reads YSIZERAM (ØF77AH) (binary value in 1-KB units) and displays the RAM disk size.
(2) User BIOS area size
Reads USERBIOS (ØEF2DH) (binary value in 256-byte units) and displays the user BIOS area size.
(3) Alarm type
Reads ALRMTP (ØEF8AH) and displays the alarm type.

ALRMTP = ØØH: Alarm off
= Ø1H: Alarm on
= Ø2H: Wake on

(4) Auto start string
Reads AUTOSTRT (ØF3BDH) and indicates the presence or absence of an auto start string. AUTOSTRT is loaded with the length of the auto start string.

AUTOSTRT = ØØH: No auto start string defined.
≠ Nonzero: Auto start string defined.

(5) Menu ON/OFF
Reads MENUFG (ØEF44H) and indicates whether menu display is on or off.

MENUFG = ØØH: Menu display ON
= ØFFH: Menu display OFF

(6) Microcassette drive mount state
Reads CRGDEV (ØF53FH) and indicates whether or not an MCT cartridge is mounted.

CRGDEV = 48H: MCT cartridge mounted
≠ Other than 48H: No MCT cartridge mounted, or another type of cartridge mounted

2.11.3.2 Setting operations

The system display function performs the following setting operations:
(1) RAM cartridge
Identifies the presence or absence of the RAM cartridge by examining the value of CRGDEV (ØF53FH). The function formats the RAM cartridge using RAMCRFMT (ØØ24H). See Section 4.2, "Jump Tables" for the contents of RAMCRFMT.

(2) Alarm/wake
Sets or resets the alarm/wake function or displays the current alarm/wake settings using the BIOS TIMDAT routine. The system display function can be inhibited from setting or resetting the alarm/wake function by manipulating the value of ALRMSTDS (ØEF43H).

ALRMSTDS = ØØH: The alarm/wake function can be set/reset via the system display.
≠ Nonzero: The alram/wake function cannot be set/reset via the system display.

(3) Auto start
Sets or resets the auto start function by directly accessing AUTOSTRT (ØF3BDH).

(4) Menu
Wets or resets the menu function by directly accessing MENUFG (ØEF44H), MENUDRV (ØEF45H), and FTYPETBL (ØEF5ØH). See Section 2.1Ø, "Menu" for these areas.

(5) Microcassette drive
Identifies the MCT stop/nonstop mode setting by examining DFTATR (ØF2E4H), bit 7.

DFTATR, bit 7 = 1: Stop mode
= Ø: Nonstop mode

The stop/nonstop mode can be selected by setting or resetting bit 7 of DFTATR (ØF2E4H) and TACATR (ØF7D1H).

The system display function identifies the MCT verify/non-verify mode by examining VERFDFLT (ØEFBEH).

VERFDFLT = ØØH: Non-verify mode
= ØFFH: Verify mode

MCT manual operations are assigned to the following PF keys:

PF1: Initiates MIOS function Ø6H (Fast feed).
PF2: Initiates MIOS functions ØBH (Head on) and then Ø4H (Play).
PF3: Initiates MIOS functions Ø3H (Stop) and then ØCH (Head off).
PF4: Initiates MIOS function Ø8H (Rewind).
PF5: Initiates MIOS function Ø2H (Set tape counter).
PF6: Initiates MIOS function 252 (Remove).
PF7: Initiates MIOS function 253 (Mount).
PF8: Initiates MIOS function 255 (Make directory).
PF9: Initiates MIOS function 15H (Erase).

The following restrictions are placed on MCT manual operations.

- No manual operation can be performed when an MCT file is open. OPNMOD (ØF355H) is examined to check for the presence or absence of an open MCT file.

OPNMOD = ØØH: No open file.
= Ø1H: File open in read mode.
= Ø2H: File open in write mode.

- In the MCT mount state, only the remove operation can be performed. In the MCT remove state, any operations except for remove can be performed. Whether the MCT is in the mount or remove state is determined by the value of TAPMOD (ØF2E1H).

TAPMOD = ØØH: Remove state
= Ø1H: Mount state

See Section 3.7, "MTOS/MIOS Operations" for Microcassette drive

2.11.4 Miscellaneous

2.11.4.1 System display processing when the item keyboard is installed

When the item keybaord is installed, no system display screen is displayed even when the CTRL/HELP keys are pressed. This is because the CTRL key on the item keyboard is used only in combination with the STOP key; it has different functions from the CTRL key on the standard keyboard.

Even when the item keyboard is installed, the system display screen can be displayed from an application program, following the steps described below. However, MCT manual operation cannot be performed since the item keyboard has no PF keys.

(1) Calculate the starting address of the system display processing routine based on the CTRL/HELP address (ØFØDlH) in YSUBRTN.
(2) Call the address obtained in the above step using BIOS CALLX.
(3) The system display screen is displayed via the above two steps. When system display processing is completed, application program processing resumes at the point at which the system display routine starting address was called.

```
                                ;       TITLE   SYSTEM DISPLAY
                                ;       PAGE    45
                                ;       ************************************************
                                ;               SYSTEM DISPLAY BY USER CALL
                                ;       ************************************************
                                ;
                                ;       NOTE :
                                ;       <> assemble condition <>
                                ;
                                        .Z80
                                ;
                                ;       <> loading address <>
                                ;
                                        .PHASE  100H
                                ;
                                ;       <> constant values <>
                                ;
F0D1                            CTRLHELP        EQU     0F0D1H          ; CTRL/HELP function.
F52E                            DISBNK          EQU     0F52EH          ; Destination bank.
                                ;
EB03                            WBOOT           EQU     0EB03H          ; WBOOT entry address
EB69                            CALLX           EQU     0EB69H          ; CALLX entry address
                                ;
1000                            MAINSP          EQU     01000H          ; Stack pointer.
                                ;
0100                            START:
0100    31 1000                         LD      SP,MAINSP       ; Set stack pointer.

0103    DD 2A F0D1                      LD      IX,(CTRLHELP)
0107    3E FF                           LD      A,0FFH          ; Select system bank.
0109    32 F52E                         LD      (DISBNK),A      ;
010C    CD EB69                         CALL    CALLX           ; Call SETERR.
010F    C3 EB03                         JP      WBOOT

                                        END
```

CHAPTER 3 CP/M I (BDOS, BIOS) CONTENTS

3.1	PINE CP/M	II-58
	3.1.1 General	II-58
	3.1.2 Memory Addressing	II-60
	3.1.3 Constructing a CP/M System	II-64
	3.1.4 Terminating CP/M	II-66
3.2	BDOS Operations	II-68
	3.2.1 General	II-68
	3.2.2 BDOS Function Operation Flow	II-68
	3.2.3 BDOS Functions	II-70
	3.2.4 BDOS Errors	II-73
3.3	BIOS Operations	II-82
	3.3.1 General	II-82
	3.3.2 BIOS Function Operation Flow	II-82
	3.3.3 BIOS Hook	II-85
3.4	BIOS Details	II-95
	3.4.1 Programming Notes on the Use of BIOS Functions	II-95
	3.4.2 BIOS Entries	II-97
	3.4.3 BIOS Functional Descriptions	II-99
3.5	Keyboard	II-182
	3.5.1 General	II-182
	3.5.2 Keyboard Functions	II-182
	3.5.3 Standard Keyboard	II-183
	3.5.4 Item Keyboard	II-187
	3.5.5 Work Areas for Keyboard Processing	II-197
	3.5.6 Key Code Charts	II-212
	3.5.7 Hard Code Charts	II-224
3.6	LCD Display (CONOUT Details)	II-230
	3.6.1 General	II-230
	3.6.2 Screen Modes	II-231
	3.6.3 Virtual Screen	II-238
	3.6.4 Graphics Display	II-243
	3.6.5 CONOUT (BIOS)	II-245
	3.6.6 Character Generator	II-258
	3.6.7 Miscellaneous Considerations	II-265
	3.6.8 Screen-related Work Areas	II-270

3.7 MTOS/MIOS Operations II-284
3.7.1 General II-284
3.7.2 File Control II-285
3.7.3 Tape File Control Block (T-FCB) .. II-292
3.7.4 Using MTOS II-294
3.7.5 MTOS Functions II-304
3.7.6 Using MIOS II-319
3.7.7 MIOS Functions II-321
3.7.8 Work Areas Related to MCT Processing II-334

3.8 Disk Image Storage II-339
3.8.1 General II-339
3.8.2 Logical to Physical Drive Assignment II-340
3.8.3 Disk Details II-345
3.8.4 Disk-related Work Areas II-362

3.9 I/O Byte II-365
3.9.1 General II-365
3.9.2 Modifying the I/O Byte II-365
3.9.3 I/O Byte Contents II-365

CHAPTER 3 CP/M I (BDOS, BIOS)

3.1 PINE CP/M

3.1.1 General

The PINE operating system is an extended version of Standard CP/M Version 2.2. The new system modules of the PINE CP/M have been introduced in Chapter 2. In this chapter, the extended functions are described in full depth.

The structure of extended CP/M is shown in Figure 3.1.1.

PINE extended CP/M is provided with the following features:

1. Extended BIOS has thirty BIOS entries in addition to the standard BIOS entries to support expansion devices such as RS-232C interface, SIO, cartridge SIO, clock, buzzer, RAM disk, and ROM and RAM cartridges.
2. Most of OS programs are executed in ROM, leaving a large user area.
3. Frequently used application programs are executed in ROM, leaving a large user area.
4. Various types of memory-related devices (main RAM, ROM capsules, ROM and RAM cartridges, microcassette, etc.) are available as disk drives. This provides the PINE user with great operational ease.

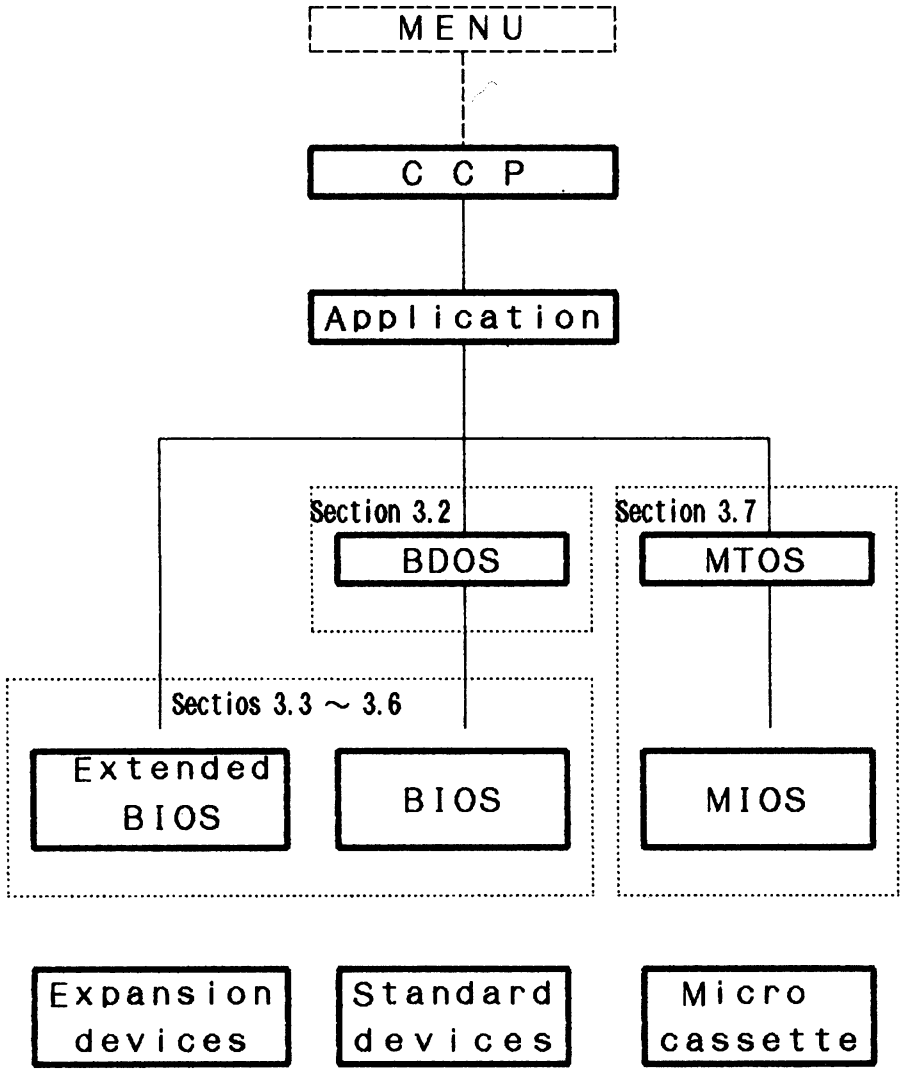


Fig. 3.1.1 CP/M Organization

3.1.2 Memory Addressing

The PINE supports four types of banks shown in Figure 3.1.2.

The four banks are:

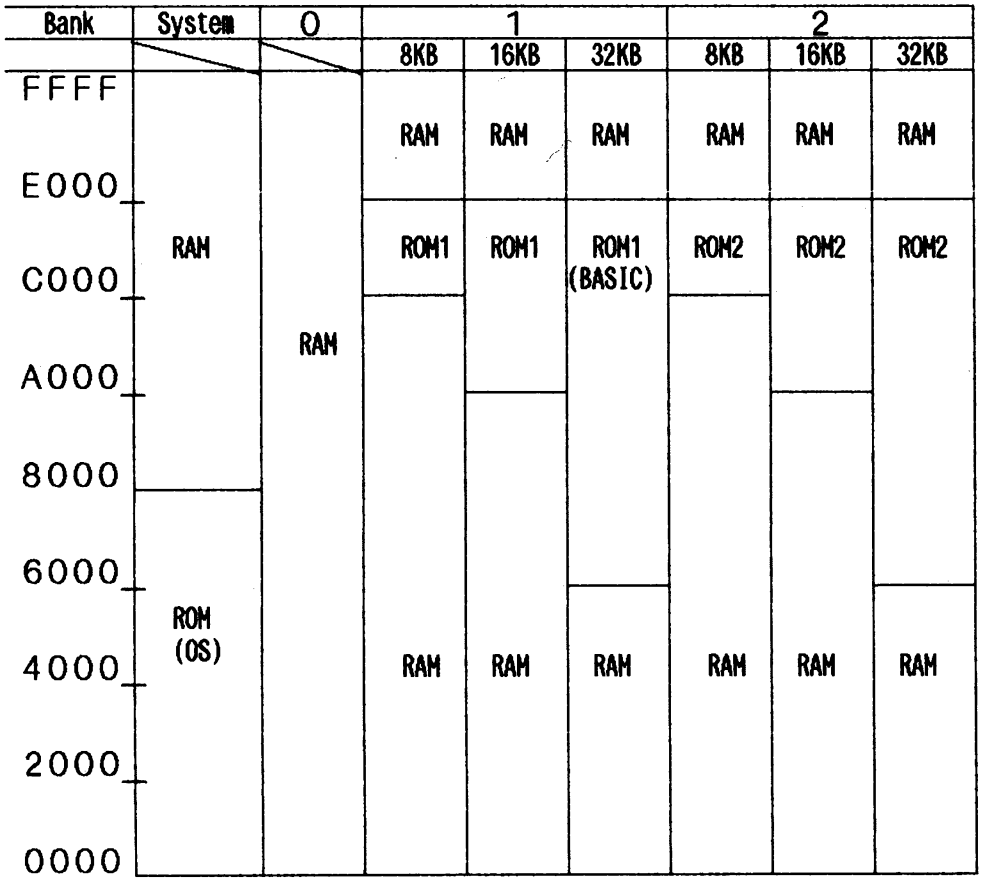
- System bank -- Contains OS ROM and the first half of RAM.
- Bank Ø -- Contains the entire RAM.
- Bank 1 -- Contains BASIC ROM (as shipped from the factory) and part of RAM.
- Bank 2 -- Contains application ROM and part of RAM.

The CP/M modules (CCP, BDOS, and BIOS) are loaded on bank Ø (all RAM) as they are on ordinary CP/M machines. This means that PINE application programs can use the CP/M functions in the same way as under standard CP/M.

Only the entry points to the BDOS and BIOS functions are loaded in RAM to reserve as large a RAM area as possible for application programs. Actual BDOS and BIOS operations are performed in OS ROM.

Each BDOS and BIOS function is provided with two entry points for convenience in executing application programs which run on both banks 1 and 2.

Figure 3.1.3 gives the RAM memory map for the PINE.



Note: The bank 1 of the standard PINE contains BASIC.

Fig. 3.1.2 Bank Structure

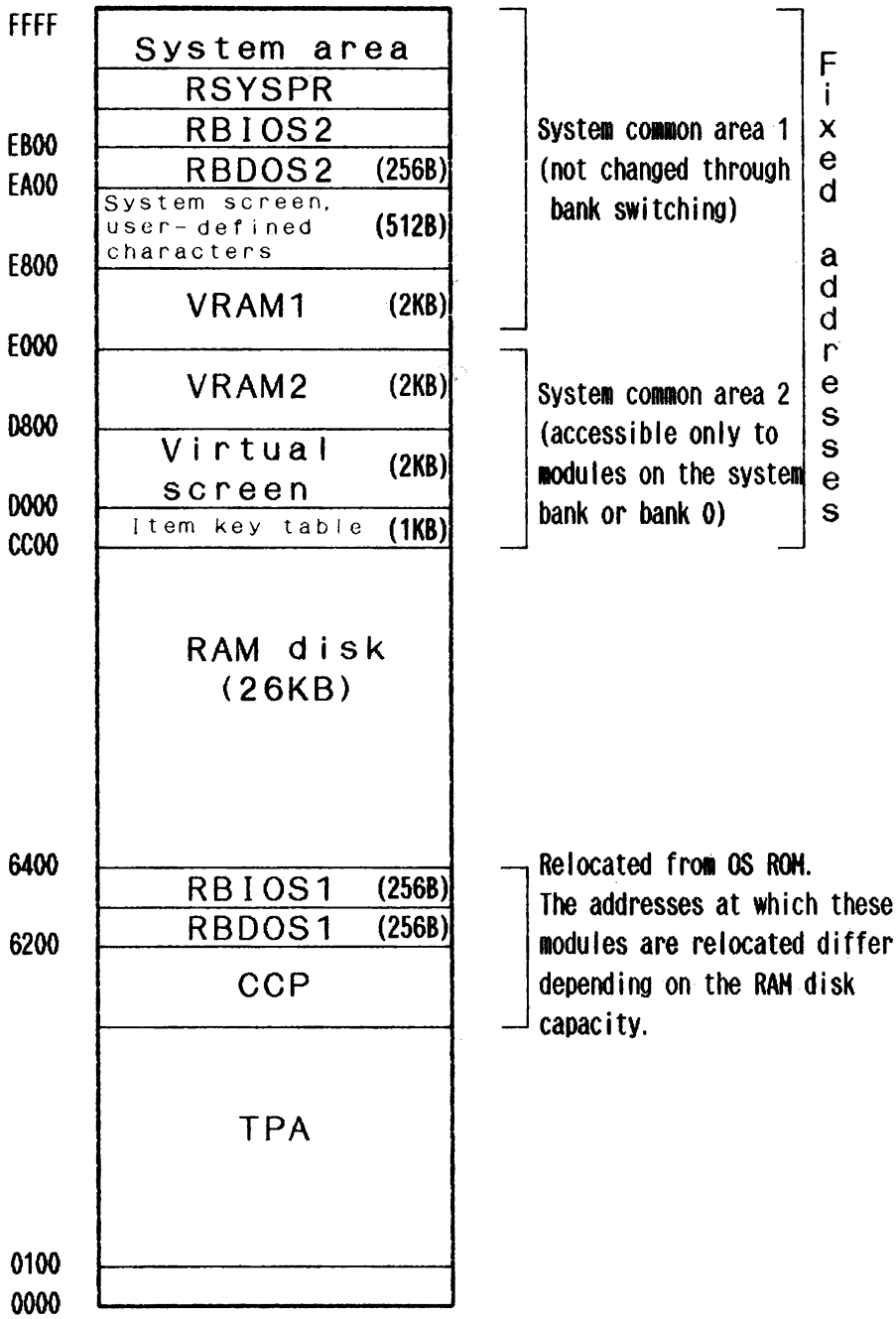


Fig. 3.1.3 RAM Memory Map

(RAM disk = 26KB, user BIOS area = ØKB)

Notes:

1. Either bank 1 or 2 has three different organizations depending on the ROM capacity (8KB, 16KB, or 32KB). When power is turned on, the system determines the organization of each bank by reading from the header area the capacity of ROM on the corresponding bank.
2. The contents of RBDOS1 and RBDOS2, and those of RBIOS1 and RBIOS2 are exactly the same.
   Load-and-go programs must call RBDOS1 or RBIOS1 (ordinary BDOS or BIOS call) to ask for system services.
   ROM-based programs must call RBDOS2 or RBIOS2 (at a fixed address) because RBDOS1 or RBIOS1 may be located on the background bank.
3. The amount of memory reserved for the virtual screen is fixed regardless of the virtual screen size. This means that changes in the virtual screen size do not affect the CP/M size.
4. The user BIOS size is initially set to Ø. If the user BIOS is specified, however, the area for the user BIOS is reserved between the RAM disk area and the item table area.
5. The system area used by the PINE is divided into the following two types:

   - System area for which initial values are set only when predetermined conditions are satisfied.
   - System area simply used as a temporary work area.

   The system area of the first type is subdivided into three types called RSYSAR1, RSYSAR2, and RSYSAR3, respectively.

   RSYSAR1: Initialized at system initialize time.
   RSYSAR2: Initialized at system initialize or reset time.
   RSYSAR3: Initialized at system initialize, reset, or WBOOT time.
   RSYSAR4: Not initialized.
   RSYSAR5: Stack and buffer.

See also:

- Section 4.1, "User BIOS"
- Section 4.4, "Bank Switching"
- Section 4.6, "Executing a ROM-based Program"
- Section 6.1, "Memory Map"

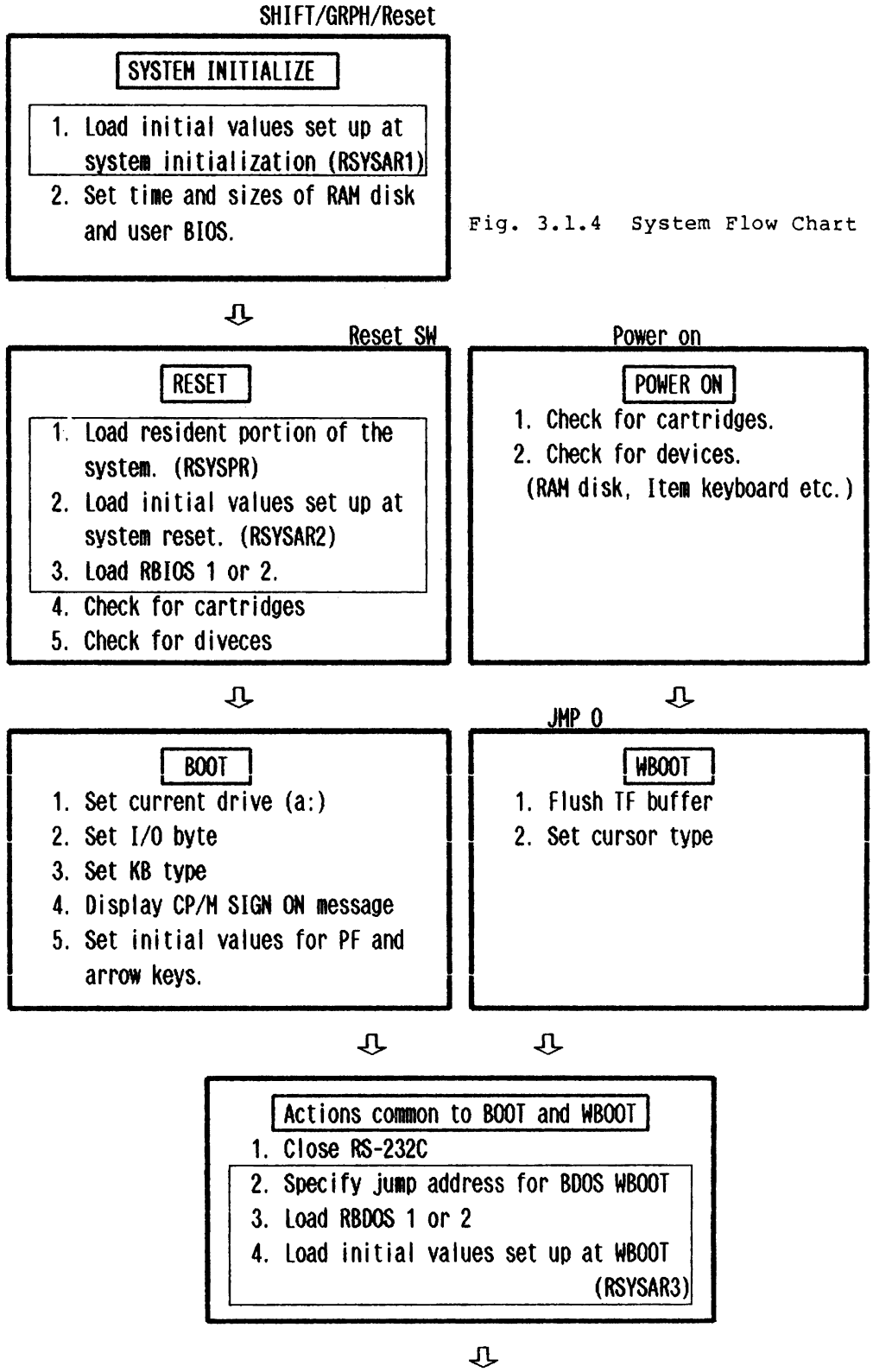
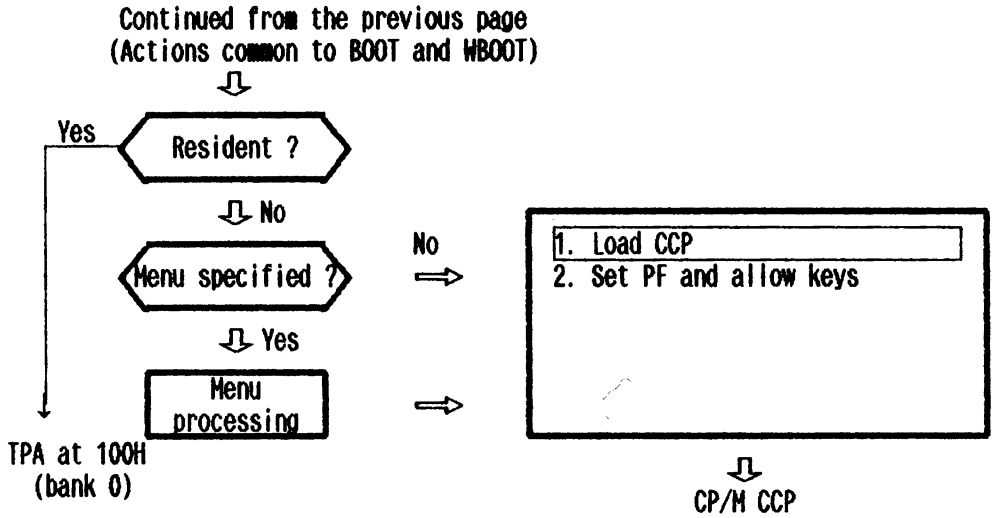


Fig. 3.1.4 System Flow Chart



* CP/M is loaded into RAM through the operations in the boxes enclosed in the larger boxes.

3.1.4 Terminating CP/M

CP/M is interrupted during processing by:

1. Pressing the STOP key.
   1) Purpose:
   To inform the system of the depression of the STOP key.
   2) System action:
   Clears the buffers for the keyboard (75Ø8 slave CPU) and BIOS, sets the stop flag BRKFLG (ØF Ø19H) to on, and returns Ø3H.
2. Pressing the CTRL and STOP keys.
   1) Purpose:
   To terminate the current I/O operation immediately.
   2) System action:
   Clears the keyboard buffer, sets the CTRL/STOP flag CSTOPFLG (ØF Ø1AH) to on, and returns Ø3H. The I/O operation is terminated immediately through the CTRL/STOP flag.
3. Turning off the power switch.
   1) Purpose:
   To turn off power in the restart mode (when a standard keyboard is installed).
   2) System action:
   Continues the I/O operation till the end of the current unit of work, terminates processing, and turns off power. Stops the buzzer (beep processing) immediately. The system performs a warm boot when power is turned on the next time.
4. Turning off the power switch while holding down the CTRL key.
   1) Purpose:
   To turn off power in the continue mode.
   2) System action:
   Continues the I/O operation till the end of the current unit of work, reserves all parameters necessary to continue the processing, and turns off power in the continue mode. The system continues processing at the next power on starting at the point where power was turned off.
5. The arrival of the auto power off time.
   1) Purpose:
   This function is provided to turn off power when the system has been waited for entry from the keyboard for a specified period of time. The function saves power by automatically turning off power when the user forgets to do so.
   2) System action:
   Preserves all parameters necessary to continue the processing and turns off power in the continue mode. The system continues processing at the next power on starting at the point where power was turned off.
6. The detection of a power failure.
   1) Cause:
   The battery voltage drops below a predetermined level.
   2) System action:
   Turns off power in the continue mode as in the action for 4.
7. Pressing the reset switch.
   1) Purpose:
   To cold start the system when the program hangs up. If the CP/M size or the RAM disk is destroyed, however, the system does not start normally.
   2) System action:

Performs a boot operation. The system reserves the RAM disk and user BIOS areas but restores the BIOS entries to initial values. The system also resets the 75Ø8 slave CPU (keyboard parameters are set to default values).

8. Pressing the reset switch while holding down both the right SHIFT and GRPH keys (when a standard keyboard is installed) or the STOP and INIT keys (when an ITEM keyboard is installed).
   1) Purpose:
   To initialize the system. The system starts unless the slave CPU is not in a hang-up state.
   2) System action:
   Initializes the system and resets the slave CPU (keyboard parameters are set to default values). The system starts I/O operations from the beginning.
9. Resetting the slave CPU.
   1) Purpose:
   To start the system when the slave CPU is in a hang-up state.
   2) System action:
   Initializes the entire system including the slave CPU.

Reference:
The following system areas are reserved for the STOP and CTRL/STOP keys and used as flags:

- BRKFLG (ØF819H) 1 byte

The STOP flag indicating whether the STOP key has been pressed. This flag is set when a keyboard interrupt occurs and reset by CONIN or CONST (if no entry has been made from the keyboard).
ØØH: STOP key not pressed.
Nonzero: STOP key pressed.

- CSTOPFLG (ØF81AH) 1 byte

The CTRL/STOP flag indicating whether the CTRL/STOP keys havebeen pressed. This flag is set by a keyboard interrupt and reset by CONIN or CONST (if no entry has been made from the keyboard).
ØØH: CTRL/STOP keys not pressed.
Nonzero: CTRL/STOP keys pressed.

3.2 BDOS Operations

3.2.1 General

As mentioned in Section 3.1, the operating system for the PINE is derived from the standard CP/M Version 2.2.

PINE BDOS is located in two places in RAM. This feature offers the following advantages:

- The upper limit of available RAM is indicated so that ordinary CP/M applications can execute without modification.
- The user can call BDOS from a ROM-based program without being aware of bank switching.

PINE expansion devices, especially microcassette, have several operational restrictions.

This section discusses BDOS operations focusing on the items unique to the PINE.

3.2.2 BDOS Function Operation Flow

When BDOS is called by a PINE application program, control is first transferred to the entry point to the BDOS in RAM. Then the OS switches banks and calls the real BDOS in OS ROM. Upon completion of processing, the OS switches the bank back to the original bank that was used when BDOS was called and returns control to the application program with return information loaded in registers.

The BIOS used by the BDOS in ROM calls the BIOS in OS ROM.

The procedure for calling BDOS differs depending on the type of the application program.

- Load-and-go programs call "JMP RBDOS1" at address 0005H.
- ROM-based programs call "JMP RBDOS2" at address 0FF90H.

Both Load-and-go and ROM-based programs use BDOS functions exactly the same way.

Figure 3.2.1 illustrates the operation flow from a BDOS call to the return of control to the application program.

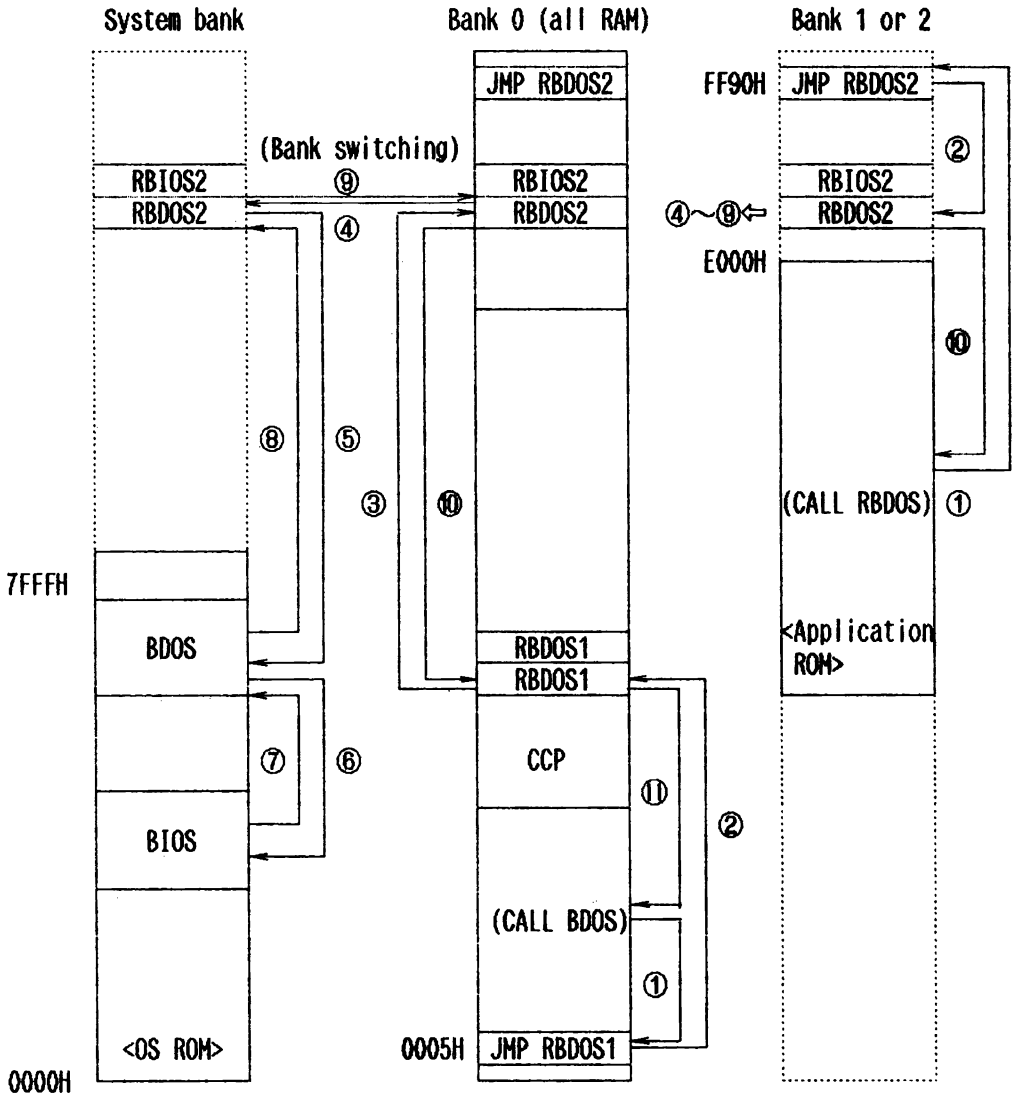


Fig. 3.2.1 BDOS Call Operation Flow

3.2.3 BDOS Functions

PINE application programs use the same interface as ordinary CP/M application programs when calling BDOS.

This subsection lists the BDOS functions with a brief description. Refer to "CP/M 2.2 Interface Guide" for detailed information about the use of BDOS functions.

Table 3.2.1 BDOS Calls

Number	Function Name	Input	Output
0	System Reset	C : 00H	None
1	Console Input	C : 01H	A : Input char
2	Console Output	C : 02H E : Output char	None
3	Reader Input	C : 03H	A : Input char
4	Punch Output	C : 04H E : Output char	None
5	List Output	C : 05H E : Outpur char	None
6	Direct Console I/O	C : 06H E : 0FFH (input) : Output char (output)	A : Input char (input) : None
7	Get IOBYTE	C : 07H	A : IOBYTE
8	Set IOBYTE	C : 08H E : IOBYTE	None
9	Print String	C : 09H DE : Address at which the string is stored.	None
10	Read Console Buffer	C : 0AH DE : Buffer address	Loads the buffer with entry from the console.
11	Get Console Status	C : 0BH	A : Console Status
12	Get Version Number	C : 0CH	HL : Version Number
13	Reset Disk System	C : 0DH	None
14	Select Disk	C : 0EH E : Disk number	None
15	Open File	C : 0FH DE : FCB address	A : Directory code

Number	Function Name	Input	Output
16	Close File	C : 10H DE : FCB address	A : Directory code
17	Search for First	C : 11H DE : FCB address	A : Directory code
18	Search for Next	C : 12H	A : Directory code
19	Delete File	C : 13H DE : FCB address	A : Directory code
20	Read Sequential	C : 14H DE : FCB address	A : Return code
21	Write Sequential	C : 15H DE : FCB address	A : Return code
22	Create File	C : 16H DE : FCB address	A : Directory code
23	Rename File	C : 17H DE : FCB address	A : Directory code
24	Get Login Vector	C : 18H	HL : Login vector
25	Get Disk Number	C : 19H	A : Disk Number
26	Set DMA Address	C : 1AH DE : DMA address	None
27	Get Allocation Address	C : 1BH	HL : Allocation address
28	Write Protect Disk	C : 1CH	None
29	Get R/O Vector	C : 1CH	HL : R/O vector
30	Set File Attributes	C : 1EH DE : FCB address	A : Directory code
31	Get DPB Address	C : 1FH	HL : DPB address
32	Set/get User Code	C : 20H E : 0FFH (Get) : User code (Set)	A : None (Set) : User code (Get)
33	Read Random	C : 21H DE : FCB address	A : Return code
34	Write Random	C : 22H DE : FCB address	A : Return code
35	Compute File Size	C : 23H DE : FCB address	FCB r0, r1, r2
36	Set Random Number	C : 24H DE : FCB address	FCB r0, r1, r2

Number	Function Name	Input	Output
37	Reset Disk Drive	C : 25H DE : Drive vector	None
38			
39			
40	Random Write with Zero File	C : 28H DE : FCB address	A : Return code
251	Verify File	C : 0FBH DE : T~FCB	
252	Remove Tape	C : 0FCH	
253	Mount Tape	C : 0FDH	
254	Read Tape ID	C : 0FEH	
255	Create Tape Directory	C : 0FFH DE : T~FCB	

Note: Functions 251 through 255 comprises Extended BDOS (MTOS) for microcassette. See Section 3.7, "MTOS/MIOS Operations" for details.

3.2.4 BDOS Errors

BDOS errors are divided into the following categories:

1. Bad Sector error
   1) Cause:
   An error was found while reading or writing a disk.
   2) Action:
   Terminate processing by pressing STOP or CTRL/C. If any other key is hit, the system continues processing, ignoring the BAD SECTOR error.

2. Select error
   1) Cause:
   An attempt was made to address a drive beyond the specified range or a drive that was not ready.
   2) Action:
   Press any key. The system sets the drive to the currently logged in drive.

3. R/O error
   1) Cause:
   An attempt was made to write a read-only disk.
   2) Action:
   Press any key. The system performs a warm boot.

4. File R/O error
   1) Cause:
   An attempt was made to write a read-only file.
   2) Action:
   Press any key. The system performs a warm boot.

A drive is reported as not ready when:
1. Floppy disk drive power is off.
2. The floppy disk drive cable is not connected.
3. No disk is inserted.
4. No cartridge is installed.
5. No ROM capsule is installed.

Table 3.2.2 shows the relationship between disk devices and BDOS errors.

Cause	RAM disk RAM cartridge	ROM capsule ROM cartridge	External RAM disk	Micro- cassette	BDOS action
Checksum error	○	–	–	–	Bad Sector
Directory full	○	–	○	○	Returns with A = 0FFH
Disk full	○	–	○	○	Returns with A = 0FFH
Write processing	–	○	–	–	R/O
Attempt to write write-protected device	○	–	○	○	R/O
End of tape encountered during write	–	–	–	○	Bad Sector
File not found in directory	○	○	○	○	Returns with A = 0FFH
File not found	–	–	–	○	Bad Sector
Drive not ready	○	○	○	○	Select
No cassette found during read or mount	–	–	–	○	Bad Sector
No cassette found during write or remove	–	–	–	○	Bad Sector

○ ··· probable

– ··· Not probable

Table 3.2.2 BDOS Error Recovery Actions

As described above, BDOS can indicate four types of error conditions. Since these errors are handled totally under BDOS control, they may destroy the current screen image or initiate a warm boot on receipt of user response from the keyboard after displaying the error.

One of the countermeasures to avoid this is to make the application program report and handle error conditions by itself.

The application program can achieve this by taking the following two measures against the error:

1. Receiving BDOS error information in a return code.
2. Rewriting the jump vector for BDOS error processing and performing use-supplied error processing.

(1) Receiving BDOS error information in a return code

1) Procedure

The application program can receive any BDOS error information in registers by calling location 0012H (SETERR) in OS ROM (system bank). It can also have BDOS return any error information by calling location 0015H (RSTERR) in OS ROM.

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

2) Return codes

The A and H registers are loaded with the following return codes when SETERR is executed:

Register / Error	A	H	
BAD SECTOR	0FFH	01H	Standard CP/M BDOS errors
BAD SELECT	0FFH	02H	
R/O DISK	0FFH	03H	
R/O FILE	0FFH	04H	
MCT ERROR	0FFH	05H	MCT only
		00H	

When the H register is loaded with 00H, the return code corresponding to the CP/M return information is also loaded in the A register.

For Bad Sector errors, BDOS stores more detailed error information in system area BIOSERROR (0F52BH).

BIOSERROR (ØF52BH) 1 byte

Loaded with one of the following information at the end of a BIOS disk read or write:

= ØØH: Normal termination
= Ø1H: Read error
= Ø2H: Write error
= Ø3H: Write protect error
= Ø4H: Time-out error
= Ø5H: Seek error
= Ø6H: Break error
= Ø7H: Power off error
= Ø8H: Mount error
= ØFEH: Other errors

3) Programming notes

a. Once SETERR is executed, BDOS only returns error status and performs no error processing until RSTERR or WBOOT is executed.

b. When SETERR is executed, the results are not guaranteed unless the application program performs its own error checking and recovery processing.

(2) Rewriting the jump vector for processing BDOS errors

1) Procedure

The jump vector for BDOS error processing is located at the beginning of BDOS in RAM. The application program can perform its own error processing by changing the contents of the jump vector.

The address and contents of the jump vector are shown below. (The address of RBDOS1 can be obtained from the contents of 006H and 007H.)

Address	Data	Contents
RBDOS1+03H	DW PERERR	Address of parameter error processing routine (Bad Sector)
RBDOS1+05H	DW SELERR	Address of Select error processing routine (Bad Select)
RBDOS1+07H	DW RODERR	Address of R/O Disk error processing routine (R/O Disk)
RBDOS1+09H	DW ROFERR	Address of R/O File error processing routine (R/O File)

2) Programming notes

a. Since the stack area for the BDOS is used during BDOS processing, it is necessary to restore the stack area for the application program when returning control directly to the application program.

b. Note that bank 0 (all RAM) is selected during BDOS processing when rewriting the jump vector from a ROM-based program.

c. When rewriting the jump vector from a ROM-based program, keep in mind which bank the program is using because the jump vector may be located in the background bank of the bank on which the ROM-based program resides.

d. The user error processing routine must not contain any BDOS call if it is to return control to the system (BDOS) after error processing.

e. Switch the active bank to the system bank before returning control to the system.

```
                                ;***************************************************
                                ;       BDOS ERROR RECOVERY SAMPLE PROGRAM
                                ;***************************************************
                                ;
                                ; NOTE :
                                ; <> assemble condition <>
                                ;
                                .Z80
                                ;
                                ; <> loading address <>
                                ;
                                .PHASE  100H
                                ;
                                ; <> constant values <>
                                ;
0012                            SETERR          EQU     00012H          ; SETERR routine address
0015                            RSTERR          EQU     00015H          ; RSTERR routine address
                                ;
EB03                            WBOOT           EQU     0EB03H          ; WBOOT entry address
EB09                            CONIN           EQU     0EB09H          ; CONIN entry address
EB0C                            CONOUT          EQU     0EB0CH          ; CONOUT entry address
EB63                            LDIRX           EQU     0EB63H          ; LDIRX entry address
EB69                            CALLX           EQU     0EB69H          ; CALLX entry address
                                ;
0005                            RBDOS1          EQU     00005H          ; RBDOS1 entry address
FF90                            RBDOS2          EQU     0FF90H          ; RBDOS2 entry address
                                ;
F52B                            BIOSERROR       EQU     0F52BH          ; BIOS error information
F52E                            DISBNK          EQU     0F52EH          ; Distination bank
                                ;
1000                            MAINSP          EQU     1000H           ; Stack pointer
1000                            BDOSSP          EQU     1000H           ; Stack pointer
                                ;
                                ;***************************************************
                                ;       SELECT BDOS ERROR RECOVER
                                ;***************************************************
0100                            START:
0100    31 1000                         LD      SP,MAINSP       ; Set stack pointer.
                                ;
0103    CD 0109                 LOOP:   CALL    SELERR          ; Select error recover type.
0106                            ;
                                ;***************************************************
                                ;       USER PROGRAM
                                ;***************************************************
                                ;
                                ;       NOTE :
                                ;               This part is user program.
                                ;
0106    C3 0106                         JP      LOOP            ; Loop permanent.
                                ;***************************************************
                                ;       SELECT BDOS ERROR RECOVERY
                                ;***************************************************
                                ;
                                ;       NOTE :
                                ;               Select BDOS error recovery type.
                                ;                1. Using SETERR and RSTERR
                                ;                2. Replacing BDOS error vector
                                ;
                                ;       <> entry parameter <>
                                ;               NON
                                ;       <> return parameter <>
                                ;               NON
                                ;       <> preserved registers <>
                                ;               NON
                                ;
                                ;       CAUTION :
                                ;               If BREAK key is pressed, then WBOOT
                                ;
0109                            SELERR:
0109    21 01E2                         LD      HL,MSG01        ; Select error recover message.
010C    CD 013B                         CALL    DSPMSG          ; Display message.
                                ;
010F    CD 0147                         CALL    KEYIN           ; Get input key code.
0112    D6 31                           SUB     31H             ; Using SETERR?
0114    28 05                           JR      Z,ERR010        ; Yes.
0116    3D                              DEC     A               ; Using error vector?
0117    28 0F                           JR      Z,ERR100        ; Yes.
0119    18 EE                           JR      SELERR          ; Others, then retry.
                                ;
                                ;       CALL SETERR ROUTINE.
                                ;
011B                            ERR010:
011B    DD 21 0012                      LD      IX,SETERR       ; Set calling address.
011F    3E FF                           LD      A,0FFH          ; Select system bank.
0121    32 F52E                         LD      (DISBNK),A      ;
0124    CD EB69                         CALL    CALLX           ; Call SETERR.
0127    C9                              RET                     ;
                                ;
                                ;       CHANGE ERROR VECTOR.
                                ;
0128                            ERR100:
0128    21 01DA                         LD      HL,VECTOR       ; New error vector address.
012B    ED 5B 0006                      LD      DE,(RBDOS1+1)   ; Get error vector address.
012F    13                              INC     DE              ;  RBDOS1 top addr. + 3
0130    13                              INC     DE              ;
0131    13                              INC     DE              ;
0132    01 0008                         LD      BC,0008H        ; Transmite byte no.
0135    3E 00                           LD      A,00H           ; Select bank 0 (RAM bank).
0137    CD EB63                         CALL    LDIRX           ; Change error vector.
013A    C9                              RET                     ;
```

```
                        ;       ************************************************
                        ;               DISPLAY MESSAGE
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Display message until fine 00H.
                        ;
                        ;       <> entry parameter <>
                        ;               HL : Message data top address
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
013B                    DSPMSG:
013B  7E                        LD      A,(HL)          ; Get data 1 byte.
013C  B7                        OR      A               ; End of data?
013D  C8                        RET     Z               ; Yes.
                        ;
013E  4F                        LD      C,A             ; Set display data.
013F  E5                        PUSH    HL              ; Save pointer.
0140  CD EB0C                   CALL    CONOUT          ; Display message 1 byte.
0143  E1                        POP     HL              ; Restore pointer.
0144  23                        INC     HL              ; Update pointer.
0145  18 F4                     JR      DSPMSG          ; Loop until find 0.
                        ;
                        ;       ************************************************
                        ;               INPUT A KEY DATA
                        ;       ************************************************
                        ;
                        ;       NOTE
                        ;               Get inputed key data.
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;               If BREAK key is pressed, then WBOOT
                        ;
0147                    KEYIN:
0147  CD EB09                   CALL    CONIN           ; Get inputed key code.
014A  FE 03                     CP      03H             ; Break code?
014C  CA EB03                   JP      Z,WBOOT         ; Yes, then WBOOT.
014F  C9                        RET
                        ;       ************************************************
                        ;               BDOS ERROR RECOVERY
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               BDOS error recovery
                        ;
                        ;       <> entry parameter <>
                        ;               H    : Error type 1
                        ;               A    : Error type 2
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;               If BREAK key is pressed, then WBOOT
                        ;
0150                    ERRCHK:
0150  4F                        LD      C,A             ; Save return code.
0151  7C                        LD      A,H             ; Error type.
0152  B7                        OR      A               ; Normal end?
0153  28 10                     JR      Z,BDOSER        ; Yes.
                        ;
0155  3D                        DEC     A               ; Bad sector?
0156  28 20                     JR      Z,BADSEC        ; Yes.
0158  3D                        DEC     A               ; Bad select?
0159  28 33                     JR      Z,BADSEL        ; Yes.
015B  3D                        DEC     A               ; Read only disk?
015C  28 3A                     JR      Z,RODISK        ; Yes.
015E  3D                        DEC     A               ; Read only file?
015F  28 41                     JR      Z,ROFILE        ; Yes.
0161  3D                        DEC     A               ; Micro cassette error?
0162  28 46                     JR      Z,MCTERR        ; Yes.
0164  C9                        RET                     ;
                        ;
                        ;       BDOS ERROR INFORMATION.
                        ;
0165                    BDOSER:
0165  C5                        PUSH    BC              ; Save return code.
0166  21 0239                   LD      HL,MSG04        ; BDOS error code message.
0169  CD 013B                   CALL    DSPMSG          ; Display message.
016C  C1                        POP     BC              ; Restore return code.
                        ;
016D  3E 30                     LD      A,30H           ; Change error code to ASCII.
016F  81                        ADD     A,C             ;  Return code + 30H
0170  4F                        LD      C,A             ;
0171  CD EB0C                   CALL    CONOUT          ; Display return code.
0174  CD 0147                   CALL    KEYIN           ; Input any key.
0177  C9                        RET                     ;
                        ;
                        ;       BAD SECTOR
                        ;
0178                    BADSEC:
0178  3A F52B                   LD      A,(BIOSERROR)   ; BIOS error type.
017B  87                        ADD     A,A             ; Get message address.
017C  21 0250                   LD      HL,MSG05        ;  Message table top address.
```

```
017F    06 00           LD      B,00H           ; Get target message pointer.
0181    4F              LD      C,A             ;
0182    09              ADD     HL,BC           ;
0183    5E              LD      E,(HL)          ; Get target message address.
0184    23              INC     HL              ;
0185    56              LD      D,(HL)          ;
0186    EB              EX      DE,HL           ; Set message address to HL.
0187    CD 013B         CALL    DSPMSG          ; Display message.
018A    CD 0147         CALL    KEYIN           ; Input any key.
018D    C9              RET                     ;
                ;
                ;       BAD SELECT.
                ;
018E                    BADSEL:
018E    21 02F6         LD      HL,MSG06        ; Bad select message.
0191    CD 013B         CALL    DSPMSG          ; Display message.
0194    CD 0147         CALL    KEYIN           ; Input any key.
0197    C9              RET                     ;
                ;
                ;       READ ONLY DISK.
                ;
0198                    RODISK:
0198    21 0308         LD      HL,MSG07        ; Read only disk message.
019B    CD 013B         CALL    DSPMSG          ; Display message.
019E    CD 0147         CALL    KEYIN           ; Input any key.
01A1    C9              RET                     ;
                ;
                ;       READ ONLY FILE.
                ;
01A2                    ROFILE:
01A2    21 031C         LD      HL,MSG08        ; Read only file message.
01A5    CD 013B         CALL    DSPMSG          ; Display message.
01A8    CD 0147         CALL    KEYIN           ; Input any key.
01AB    C9              RET                     ;
                ;
                ;       MICRO CASSETTE ERROR
                ;
01AC                    MCTERR:
01AC    21 0330         LD      HL,MSG09        ; Micro cassette error message.
01AF    CD 013B         CALL    DSPMSG          ; Display message.
01B2    CD 0147         CALL    KEYIN           ; Input any key.
01B5    C9              RET                     ;
                ;
                ;
01B6                    XBADSEC:
01B6    31 1000         LD      SP,BDOSSP       ; Set stack pointer.
01B9    CD 0178         CALL    BADSEC          ; Bad sector error.
01BC    C3 0106         JP      LOOP            ; Return to user program.
01BF                    XBADSEL:
01BF    31 1000         LD      SP,BDOSSP       ; Set stack pointer.
01C2    CD 018E         CALL    BADSEL          ; Bad select error.
01C5    C3 0106         JP      LOOP            ; Return to user program.
01C8                    XRODISK:
01C8    31 1000         LD      SP,BDOSSP       ; Set stack pointer.
01CB    CD 0198         CALL    RODISK          ; Read only disk error.
01CE    C3 0106         JP      LOOP            ; Return to user program.
01D1                    XROFILE:
01D1    31 1000         LD      SP,BDOSSP       ; Set stack pointer.
01D4    CD 01A2         CALL    ROFILE          ; Read only file error.
01D7    C3 0106         JP      LOOP            ; Return to user program.
                ;
                ;       NEW ERROR VECTOR
                ;
01DA                    VECTOR:
01DA    01B6            DW      XBADSEC         ; Bad sector
01DC    01BF            DW      XBADSEL         ; Bad select
01DE    01C8            DW      XRODISK         ; Read only disk
01E0    01D1            DW      XROFILE         ; Read only file
                ;
                ;       MESSAGE
                ;
01E2                    MSG01:
01E2    0C              DB      0CH
01E3    53 65 6C 65     DB      'Select BDOS error recover type.',0DH,0AH
01E7    63 74 20 42
01EB    44 4F 53 20
01EF    65 72 72 6F
01F3    72 20 72 65
01F7    63 6F 76 65
01FB    72 20 74 79
01FF    70 65 2E 0D
0203    0A
0204    20 20 31 20     DB      '  1 -- Using SETERR',0DH,0AH
0208    2D 2D 20 55
020C    73 69 6E 67
0210    20 53 45 54
0214    45 52 52 0D
0218    0A
0219    20 20 32 20     DB      '  2 -- Replacing error vector',0DH,0AH
021D    2D 2D 20 52
0221    65 70 6C 61
0225    63 69 6E 67
0229    20 65 72 72
022D    6F 72 20 76
0231    65 63 74 6F
0235    72 0D 0A
0238    00              DB      00H
0239                    MSG04:
0239    0D 0A           DB      0DH,0AH
023B    42 44 4F 53     DB      'BDOS return code is '
023F    20 72 65 74
0243    75 72 6E 20
0247    63 6F 64 65
024B    20 69 73 20
024F    00              DB      00H
0250                    MSG05:
0250    0260            DW      MSG050
```

```
0252  0273                          DW      MSG051
0254  0283                          DW      MSG052
0256  0294                          DW      MSG053
0258  02AD                          DW      MSG054
025A  02C2                          DW      MSG055
025C  02D2                          DW      MSG056
025E  02E3                          DW      MSG057
0260                          MSG050:
0260  0D 0A                       DB      0DH,0AH
0262  4E 6F 72 6D                 DB      'Normal return.',0DH,0AH
0266  61 6C 20 72
026A  65 74 75 72
026E  6E 2E 0D 0A
0272  00                          DB      00H
0273                          MSG051:
0273  0D 0A                       DB      0DH,0AH
0275  52 65 61 64                 DB      'Read error.',0DH,0AH
0279  20 65 72 72
027D  6F 72 2E 0D
0281  0A
0282  00                          DB      00H
0283                          MSG052:
0283  0D 0A                       DB      0DH,0AH
0285  57 72 69 74                 DB      'Write error.',0DH,0AH
0289  65 20 65 72
028D  72 6F 72 2E
0291  0D 0A
0293  00                          DB      00H
0294                          MSG053:
0294  0D 0A                       DB      0DH,0AH
0296  57 72 69 74                 DB      'Write protect error.',0DH,0AH
029A  65 20 70 72
029E  6F 74 65 63
02A2  74 20 65 72
02A6  72 6F 72 2E
02AA  0D 0A
02AC  00                          DB      00H
02AD                          MSG054:
02AD  0D 0A                       DB      0DH,0AH
02AF  54 69 6D 65                 DB      'Time over error.',0DH,0AH
02B3  20 6F 76 65
02B7  72 20 65 72
02BB  72 6F 72 2E
02BF  0D 0A
02C1  00                          DB      00H
02C2                          MSG055:
02C2  0D 0A                       DB      0DH,0AH
02C4  53 65 65 6B                 DB      'Seek error.',0DH,0AH
02C8  20 65 72 72
02CC  6F 72 2E 0D
02D0  0A
02D1  00                          DB      00H
02D2                          MSG056:
02D2  0D 0A                       DB      0DH,0AH
02D4  42 72 65 61                 DB      'Break error.',0DH,0AH
02D8  6B 20 65 72
02DC  72 6F 72 2E
02E0  0D 0A
02E2  00                          DB      00H
02E3                          MSG057:
02E3  0D 0A                       DB      0DH,0AH
02E5  50 6F 77 65                 DB      'Power off error.',0DH,0AH
02E9  72 20 6F 66
02ED  66 20 65 72
02F1  72 6F 72 2E
02F5  0D 0A
02F7  00                          DB      00H
02F8                          MSG06:
02F8  0D 0A                       DB      0DH,0AH
02FA  42 61 64 20                 DB      'Bad select.',0DH,0AH
02FE  73 65 6C 65
0302  63 74 2E 0D
0306  0A
0307  00                          DB      00H
0308                          MSG07:
0308  0D 0A                       DB      0DH,0AH
030A  52 65 61 64                 DB      'Read only disk.',0DH,0AH
030E  20 6F 6E 6C
0312  79 20 64 69
0316  73 6B 2E 0D
031A  0A
031B  00                          DB      00H
031C                          MSG08:
031C  0D 0A                       DB      0DH,0AH
031E  52 65 61 64                 DB      'Read only file.',0DH,0AH
0322  20 6F 6E 6C
0326  79 20 66 69
032A  6C 65 2E 0D
032E  0A
032F  00                          DB      00H
0330                          MSG09:
0330  0D 0A                       DB      0DH,0AH
0332  4D 69 63 72                 DB      'Micro cassette error.',0DH,0AH
0336  6F 20 63 61
033A  73 73 65 74
033E  74 65 20 65
0342  72 72 6F 72
0346  2E 0D 0A
0349  00                          DB      00H

                                  END
```

3.3 BIOS Operations

3.3.1 General

The major BIOS operations are carried out in OS ROM of the system bank as BDOS operations are.

The PINE OS provides two entry points to BIOS in RAM to enable ROM-based programs to use BIOS without being aware of the banks.

PINE extended BIOS allows the user to have easy access to PINE-unique peripheral devices such as serial interfaces, a clock, buzzer, or expansion I/O units (ROM/RAM cartridge, MCT, etc.).

PINE BIOS is provided with user BIOS and a hook to user BIOS to facilitate system extension by the user.

3.3.2 BIOS Function Operation Flow

3.3.2.1 Outline

When a call to BIOS is made from an application program, the PINE OS takes the following actions as when a BDOS call is made:

1. Switches the active bank to the system bank within the BIOS in RAM.
2. Calls the real BIOS in OS ROM.
3. Upon completion of BIOS processing, switches the bank to the one that was active when the BIOS call was made, and returns control to the application program with return information and data loaded in registers.

The procedure for calling BIOS differs depending on the type of the application program.

1. The load-and-go program calls a BIOS function by specifying its address obtained by adding the function offset to the JMP WBOOT addres (ØØØØH).
2. The ROM-based program directly calls the entry in the BIOS jump table in RBIOS2 (ØEBØØH - ØEBFFH).

In both cases, the called function operates exactly the same way.

3.3.2.2 PREBIOS and PSTBIOS

The PINE introduces PREBIOS and PSTBIOS for processing BIOS calls without disruption due to interrupts and thus increasing system reliability.

There are some cases when the PINE cannot successfully resume the execution of a program, which is interrupted by an interrupt occurring during execution of a BIOS call, when control is returned from the interrupt servicing program if the interrupt is processed immediately.

This problem can be avoided if interrupts are disabled before the execution of any BIOS call and processing of any interrupts occurring during the execution of the BIOS call is performed after the BIOS call has been terminated. This is controlled by PREBIOS and PSTBIOS.

PREBIOS and PSTBIOS are automatically executed whenever a call to BIOS is made via the BIOS vector in RAM.

(1) PREBIOS
PREBIOS sets on three flags indicating that BIOS processing is in execution, that alarm processing is disabled, and that power off processing is disabled.

(2) PSTBIOS
PSTBIOS resets the flags that are set by PREBIOS and performs any alarm or power off processing that is held pending during BIOS processing.

See Sections 2.5, 2.9, and 4.7 for detailed information.

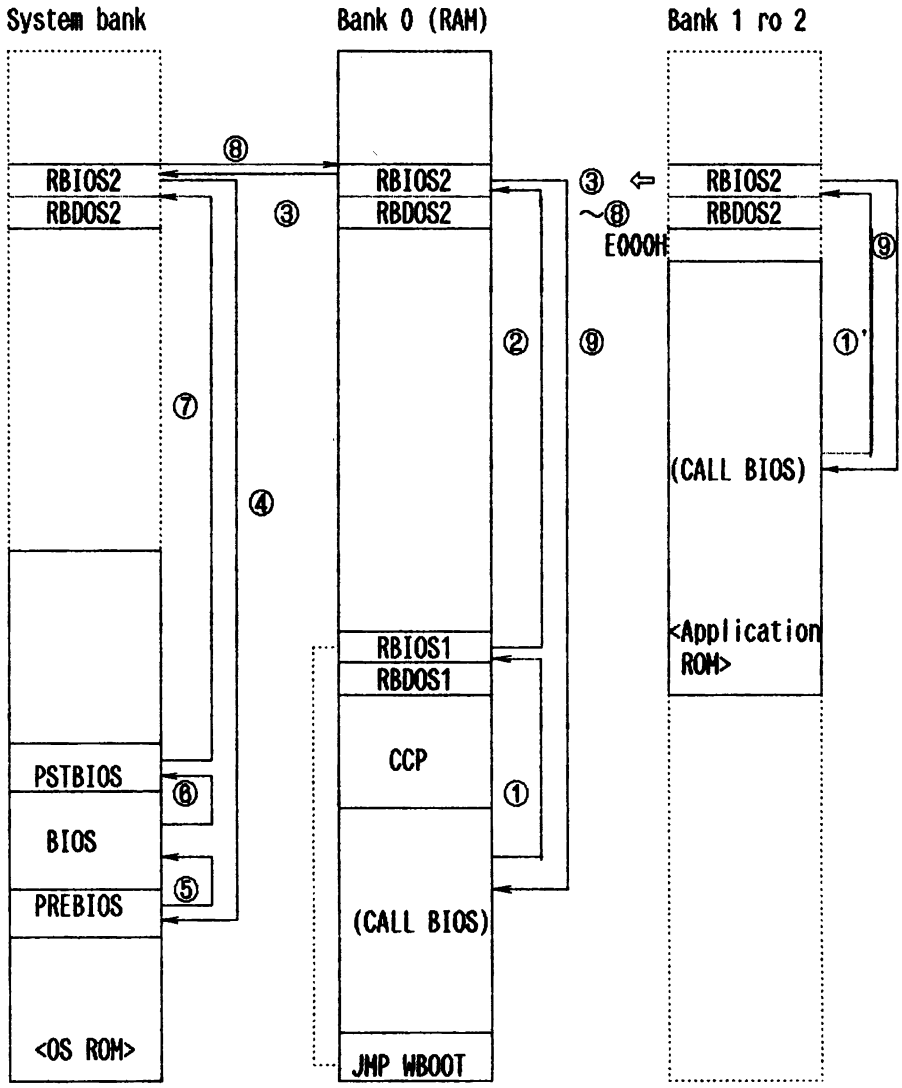


Fig. 3.3.1 BIOS Call Operation Flow

3.3.3 BIOS Hook

The PINE extended BIOS contains a hook to BIOS in addition to user BIOS. The BIOS hook permits the user to update or modify existing BIOS processing routines. This subsection explains how to use the BIOS hook.

3.3.3.1 Relationship of the BIOS hook to BIOS

The BIOS hook is referenced immediately before control is transferred from OS ROM on the system bank to a BIOS function. Figure 3.3.2 shows the relationship of the BIOS hoot to BIOS in a flowchart form. Note that only steps 3 through 8 in Figure 3.3.2 are taken when BIOS is called from BDOS.

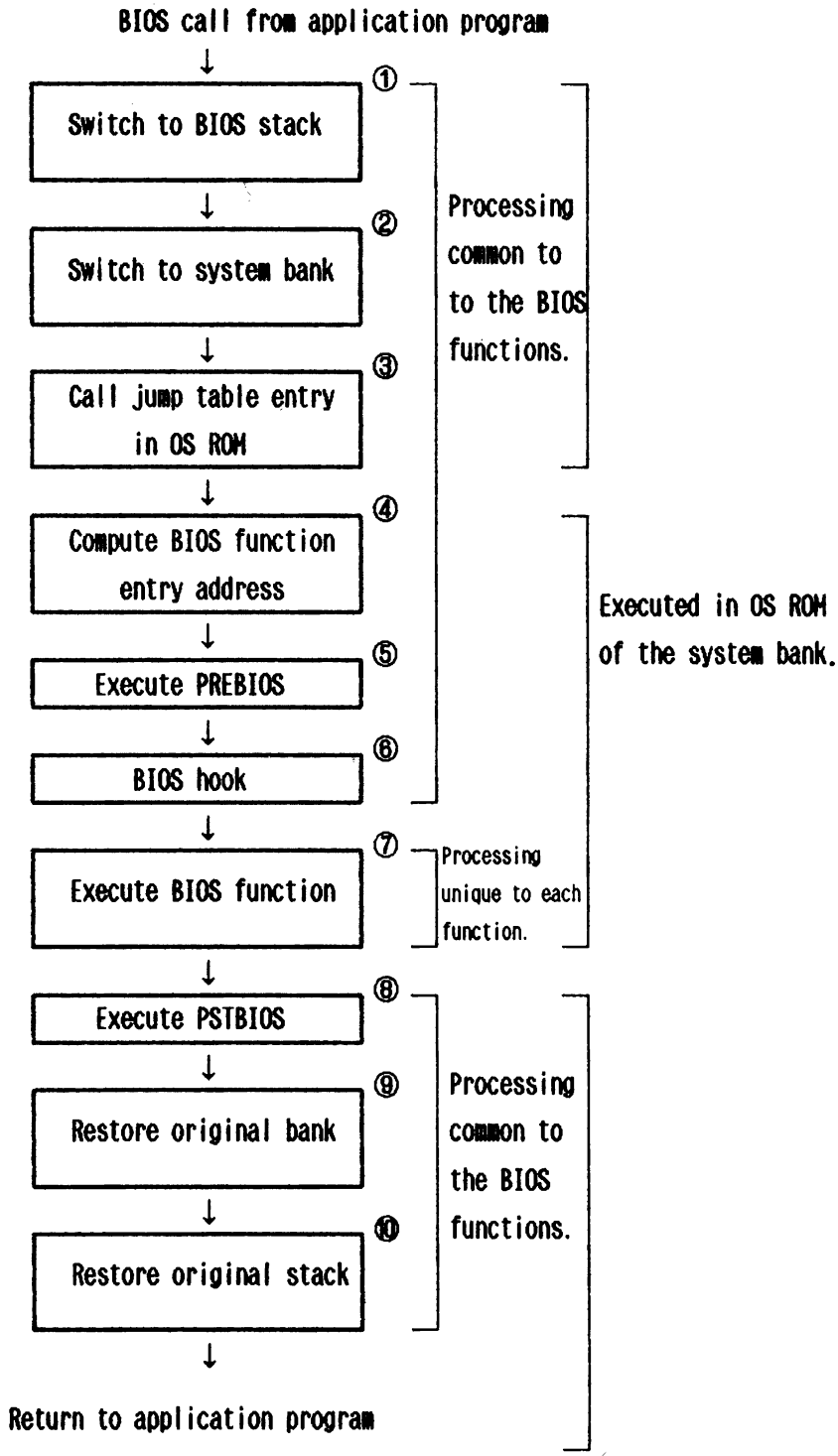


Fig. 3.3.2 BIOS Processing Flow

Step	Action	Description
1	Switch to BIOS stack	- Save the the current stack pointer to USRSBI. - Switch the stack pointer to BIOS stack.
2	Switch to system bank	- Check the BIOS function number. - Translate the DMA address to the system DMA address. - Check the current I/O byte and place its contents in RIOBYTE. - Switch the active bank to the system bank after saving original bank information.
3	Call jump table entry in OS ROM	- Call the corresponding jump table entry in OS ROM determined by the BIOS function number.
4	Compute BIOS function entry address	- Obtain the address at which the BIOS function is to be started based on the address of the called jump table entry.
5	Execute PREBIOS	- Carry out the PREBIOS processing described in 3.3.2.1.
6	BIOS hook	- Call the BIOS hook entry (ØFFE7H). - The register contents remain the same as when the BIOS function was called.
7	Execute BIOS function	- Call the corresponding BIOS function processing routine based on the BIOS function entry address obtained in step 4.
8	Execute PSTBIOS	- Carry out the PSTBIOS processing described in 3.3.2.1.
9	Restore original bank	- Restore the bank information saved in step 2 and switch the bank to the original bank. - Copy the data at the DMA address when the called function is read.
1Ø	Restore original stack	- Restore the stack pointer saved in step 1.

The following system areas are shared by the BIOS functions:

```
RIOBYTE (ØF529H)  1 byte
- I/O byte save area.
- The I/O byte format is presented in Section 3.9, "I/O Byte."

OLDBNK (ØF52CH)  1 byte
- Bank information save area.
     = ØFFH: System bank
     = ØØH:  Bank Ø
     = Ø1H:  Bank 1
     = Ø2H:  Bank 2

USRSBI (ØF535H)  2 bytes
- Area for saving the user stack for BIOS.

BIOSFN (ØF537H)  1 byte
- Area for storing a BIOS function number
     = ØØH: BOOT
     = Ø3H: WBOOT
        .
        .
        .
     = 8AH: CONTINUE

SAVEIX (ØF540H)  2 bytes
- Area for storing the contents of the IX register pair.

SAVEIY (ØF542H)  2 bytes
- Area for storing the contents of the IY register pair.
```

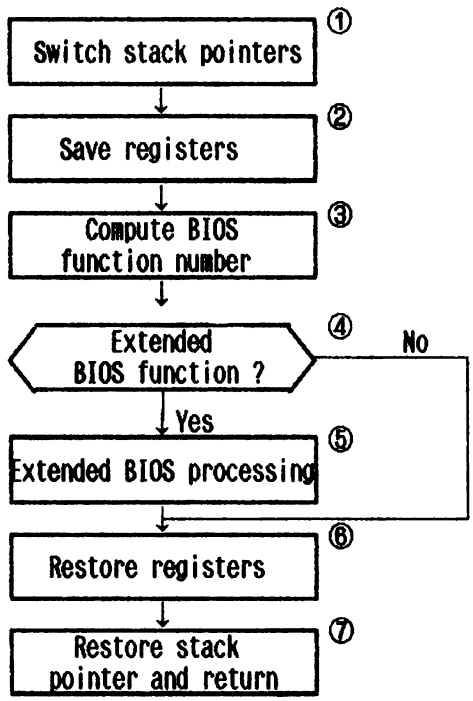
3.3.3.2 Using the BIOS hook

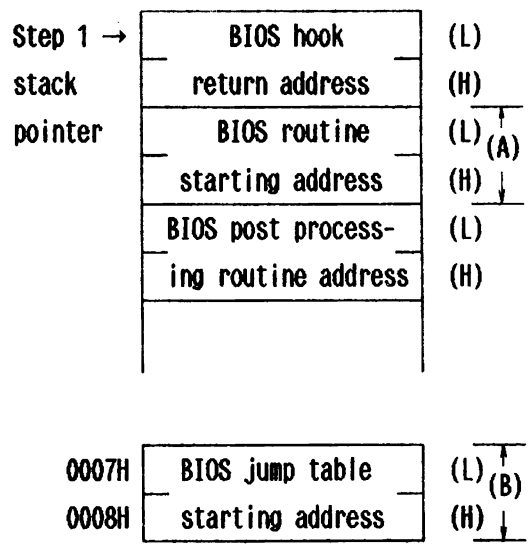
The procedure given below shows how to call BIOS through the BIOS hook.

(1) Hook processing routine logic
BIOS functions are always called through the BIOS hook.
The hook processing routine, therefore, must check for extended BIOS functions.

Sample hook processing routine:

Sample hook processing routine

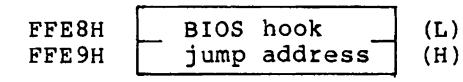


Step	Action	Description
1	Switch stack pointers	- The stack pointer currently points to the system BIOS stack area. If the routine is to use a large stack area, it must reserve its own stack area.
2	Save registers	- Registers are currently loaded with parameters. The registers that are to be used by the routine must be saved.
3	Compute BIOS function number	- Compute the BIOS function number of the called routine. - How to compute: Find the BIOS function number from the starting addresses of the called BIOS routine and the BIOS jump table.  Subtracting the value of (B) from (A) results in the offset of a BIOS function (ØØH, Ø3H, ...) with respect to the BOOT entry.

Step	Action	Description
4	Check for BIOS function	Compare the specified function number with the function number computed in step 3 to determine whether the given BIOS function is an extended one. The specified BIOS function is found to be a standard BIOS function if the following condition is met: (A) - (B) = 3n where n is the specified function number. n = ØØH ... BOOT n = Ø1H ... WBOOT n = Ø2H ... CONST n = 2DH ... CONTINUE
5	Extended BIOS processing	- Perform the extended BIOS function. - The user-supplied routine must be placed here.
6	Restore registers	- Restore the registers saved in step 2.
7	Restore stack pointer and return	- Restore the stack pointer saved in step 1. - Return. On return, control is passed to the main OS BIOS section. - When you don't want to utilize the BIOS function in the OS, pop the stack level two levels (4 bytes) and return. You will find that item 7 in Fig. 3.3.2 is skipped, and operation goes directly to item 8.

3.3.3.2 Rewriting the BIOS hook

The user can rewrite the BIOS hook by modifying its jump address.



The BIOS hook jump address is initialized to EF1FH. Address EF1FH contains the RET instruction.

Rewrite addresses FFE8H and FFE9H with the starting address of the new hook processing routine. Subsequently, any BIOS calls will be routed through the new BIOS hook processing routine.

3.3.3.3 Programming notes

(1) Since the system bank is selected when control is transferred to the BIOS hook, the hook processing routine must be placed at location 8000H or higher. It is desirable that a separate user BIOS area be reserved and the hook processing routine be implemented in that area.
(2) The BIOS hook is also given control by BIOS calls that are invoked by BDOS.
(3) The hook processing routine cannot call BIOS or BDOS functions. To use a BIOS function, directly call the BIOS function on OS ROM. No BDOS call can be made from the hook processing routine.
(4) When placing return information in IX and IY, save the contents of the IX and IY registers into SAVEIX (0F540H) and SAVEIY (0F542H), respectively.

```
                        ;       ****************************************
                        ;               BIOS HOOK SAMPLE PROGRAM
                        ;       ****************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> assemble condition <>
                        ;
                                .Z80
                        ;
                        ;
                        ;       <> loading address <>
                        ;
                                .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
FFE8                    BIOSHK          EQU     0FFE8H  ; BIOS hook address
                        ;
CB00                    LOADADDR        EQU     0CB00H  ; Extend BIOS load address
                        ;
0000                    WBOOT           EQU     00000H  ; Warm boot address
                        ;
                        ;       ****************************************
                        ;               BIOS HOOK DATA WRITE
                        ;       ****************************************
0100                    START:
0100    31 016D                 LD      SP,MAINSP       ; Set stack pointer.
                        ;
0103    CD 0120                 CALL    UBSZCHECK       ; Check User-BIOS size.
0106    DA 0000                 JP      C,WBOOT         ; Size error, then WBOOT.
                        ;
0109    21 0126                 LD      HL,LOADDATA     ; Extend BIOS routine load.
010C    11 CB00                 LD      DE,LOADADDR     ;
010F    01 0027                 LD      BC,LOADSIZE     ;
0112    ED B0                   LDIR                    ;
                        ;
0114    21 FFE8                 LD      HL,BIOSHK       ; Change BIOS hook data.
0117    11 CB00                 LD      DE,LOADADDR     ;
011A    73                      LD      (HL),E          ; Set low address.
011B    23                      INC     HL              ;
011C    72                      LD      (HL),D          ; Set high address.
                        ;
011D    C3 0000                 JP      WBOOT           ;
                        ;
                        ;       ****************************************
                        ;               USER-BIOS SIZE CHECK
                        ;       ****************************************
                        ;
                        ;       NOTE
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               CY : return information
                        ;                    = 0 : size O.K.
                        ;                    = 1 : size N.G.
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       <> constant values <>
EF2D                    USERBIOS        EQU     0EF2DH
0001                    UBSIZE          EQU     001H
                        ;
                        ;
0120                    UBSZCHECK:
                        ;
0120    3A EF2D                 LD      A,(USERBIOS)    ; USER-BIOS size --> A
0123    FE 01                   CP      UBSIZE          ; Check USER-BIOS size.
0125    C9                      RET
                        ;
                        ;       ****************************************
                        ;               EXTEND BIOS ROUTINE
                        ;       ****************************************
                        ;
                        ;       NOTE : This routine must be loaded to 0CB00H
                        ;
                        ;       <> entry parameter <>
                        ;               Depend on each BIOS parameters
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               ALL
                        ;
                        ;       <> constant values <>
CC00                    EXBIOSSP        EQU     0CC00H          ; Extend BIOS stack area (20H)
CBE0                    SAVESP          EQU     EXBIOSSP-20H    ; BIOS stack save area (02H)
                        ;
0003                    CONINF          EQU     03H             ; CONIN function number
0009                    TARGETBIOS      EQU     CONINF*3        ; Target BIOS function number
                        ;
0007                    BIOSJPTB        EQU     00007H          ; BIOS jump table address
                        ;
CB1F                    EXBIOSE         EQU     EXBIOSR-EXBIOS+LOADADDR
                                                                ; EXBIOSR addr in USER-BIOS area
                        ;
                        ;
0126                    LOADDATA:
0126                    EXBIOS:
                        ;
0126    ED 73 CBE0              LD      (SAVESP),SP     ; Save BIOS stack pointer.
012A    31 CC00                 LD      SP,EXBIOSSP     ; Set new stack pointer.
012D    E5                      PUSH    HL              ; Save registers to new stack.
012E    D5                      PUSH    DE              ;
012F    F5                      PUSH    AF              ;
                        ;
0130    2A CBE0                 LD      HL,(SAVESP)     ; Get default BIOS JUMP address.
```

```
0133    23                          INC     HL              ;
0134    23                          INC     HL              ;
0135    5E                          LD      E,(HL)          ;
0136    23                          INC     HL              ;
0137    56                          LD      D,(HL)          ;

0138    2A 0007                     LD      HL,(BIOSJPTB)   ; Get BIOS jump table top addr.
013B    EB                          EX      DE,HL           ;
013C    B7                          OR      A               ; Carry clear.
013D    ED 52                       SBC     HL,DE           ; Calculate offset value.
013F    7D                          LD      A,L             ;

0140    FE 09                       CP      TARGETBIOS      ; Target BIOS call ?
0142    C2 CB1F                     JP      NZ,EXBIOSE      ; No.


                                    You can insert your own extend-BIOS routine
                                    in this part.

                            ;
0145                        EXBIOSR:
0145    F1                          POP     AF              ; Register restore.
0146    D1                          POP     DE              ;
0147    E1                          POP     HL              ;
0148    ED 7B CBE0                  LD      SP,(SAVESP)     ; Recover stack pointer.
014C    C9                          RET                     ; Return to OS-BIOS process.
                            ;
0027                        LOADSIZE        EQU     $-LOADDATA      ; Extend-BIOS loading size
                            ;
014D                                DS      20H             ; Stack area for main routone
016D                        MAINSP          EQU     $
                            ;
                                    END
```

3.4 BIOS Details

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

3.4.1 Programming Notes on the Use of BIOS Functions

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

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

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

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

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

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

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

3.4.3 BIOS Functional Descriptions

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

Function: Gives a BIOS function outline.

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

Entry parameter: Parameters specified when the function is called.

Return parameter: Parameters returned after the function is executed.

Explanation: Detailed description of the BIOS function.

Note: Programming notes on the BIOS function.

Related functions: Other BIOS functions related to the function.

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

(1) BOOT

Function: Performs a CP/M cold boot.

Entry address: WBOOT - 3H or 0EB00H

Entry parameter: None.

Return parameter: None.

Explanation:

BOOT performs the following:

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

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

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

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

Related function: WBOOT

See also: Section 2.3, "Reset"

(2) WBOOT

Function: Performs a CP/M warm boot.

Entry address: WBOOT +- ØH or ØEBØ3H

Entry parameter: None.

Return parameter: C = Drive No.

Explanation:

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

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

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

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

See also: Section 2.4, "Power-on"

(3) CONST

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

Entry address: WBOOT + 03H or 0EB06H

Entry parameter: None.

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

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

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

(4) CONIN

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

Entry address: WBOOT + 06H or 0EB09H

Entry parameter: None.

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

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

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

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

When YPFCMFLG = ØFFH:

1. C register = ØØH
   A register = ØØH ØFEH
   (Indicates that a key other than PF keys is pressed. See the Key Entry Code Chart for key codes.)
2. C register = ØFFH
   A register = ØE0H: Indicates that PF1 is pressed.
   = ØE1H: Indicates that PF2 is pressed.
   = ØE2H: Indicates that PF3 is pressed.
   = ØE3H: Indicates that PF4 is pressed.
   = ØE4H: Indicates that PF5 is pressed.
   = ØE5H: Indicates that PF6 is pressed.
   = ØE6H: Indicates that PF7 is pressed.
   = ØE7H: Indicates that PF8 is pressed.
   = ØE8H: Indicates that PF9 is pressed.
   = ØE9H: Indicates that PF1Ø is pressed.

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

When YSFCMFLG = ØFFH:

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

Reference:

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

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

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

(5) CONOUT

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

Entry address: WBOOT + 09H or 0EB0CH

Entry parameter: C = Output data

Return parameter: None.

Explanation:

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

The following descriptions apply when the device is the LCD:

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

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

(6) LIST

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

Entry address: WBOOT + ØCH or ØEBØFH

Entry parameter: C = Output data

Return parameter: None.

Explanation:

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

Notes:

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

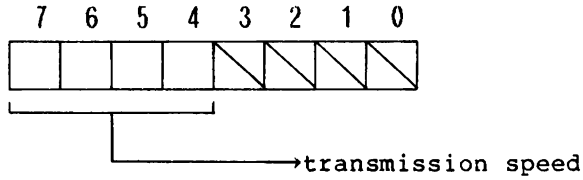
When LIST is directed to the serial port, the communication modes are initialized to 4800 bps, 8 bits, and no parity.

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

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

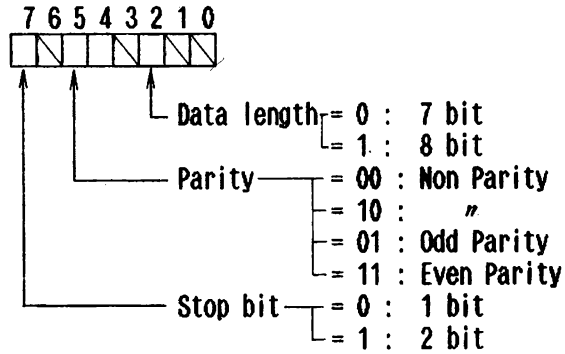
Reference:
The communication modes are specified in the following system areas:

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



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

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



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

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

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

(7) PUNCH

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

Entry address: WBOOT + ØFH or ØEB12H

Entry parameter: C = Output data

Return parameter: None.

Explanation:

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

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

Notes:

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

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

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

Reference:

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

```
SRSADR (ØEF31H)
          2 bytes: Receive buffer address
                   Default value: CONBUF (ØFC9BH)
          2 bytes: Receive buffer size
                   Default value: ØØFØH
SRSPAK (ØEF35H)
          1 byte: Transmission speed
                  Default value: ØDH --- 4800 bps
          1 byte: Bit length
                  Default value: Ø3H --- 8 bits
          1 byte: Parity
                  Default value: ØØH --- Non parity
          1 byte: Stop bit
                  Default value: Ø3H --- 2 stop bits.
          1 byte: Special parameter
                  Default value: ØFFH
```

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

Related function: RSIOX

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

(8) READER

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

Entry address: WBOOT + 12H or ØEB15H

Entry parameter: None.

Return parameter: A = Input data

Explanation:

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

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

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

Notes:

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

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

Related functions: PUNCH, RSIOX

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

(9) HOME

Function: Sets the disk seek track to Ø.

Entry address: WBOOT + 15H or ØEB18H

Entry parameter: None.

Return parameter: None.

Explanation:

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

(10) SELDSK

Function: Specifies the drive to be selected.

Entry address: WBOOT + 18H or 00EB1BH

Entry parameter:

C = Logical drive number.

00H:	Drive A	Internal/external RAM disk
01H:	Drive B	ROM capsule 1
02H:	Drive C	ROM capsule 2
03H:	Drive D	Floppy disk
04H:	Drive E	Floppy disk
05H:	Drive F	Floppy disk
06H:	Drive G	Floppy disk
07H:	Drive H	Microcassette
08H:	Drive I	RAM cartridge
09H:	Drive J	ROM cartridge 1
0AH:	Drive K	ROM cartridge 2

E = Access information

Bit 0 = 0: The first disk access after WBOOT.
= 1: Second or subsequent disk access after WBOOT.

Return parameter:

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

Explanation:

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

Reference:

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

See also: 3.8, "Disk Storage"

(11) SETTRK

Function: Specifies the track for read or write.

Entry address: WBOOT + 1BH or ØEB1EH

Entry parameter: BC = Track number

Return parameter: None.

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

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

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

Maximum value variable

See also: Section 3.8, "Disk Storage"

(12) SETSEC

Function: Specifies the sector for subsequent read or write.

Entry address: WBOOT + 1EH or ØEB21H

Entry parameter: C = sector No.

Return parameter: None.

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

See also: Section 3.8, "Disk Storage"

(13) SETDMA

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

Entry address: WBOOT + 21H or ØEB24H

Entry parameter: BC = DMA starting address

Return parameter: None.

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

(14) READ

Function: Reads data in 128 byte units.

Entry address: WBOOT + 24H or ØEB27H

Entry parameter: None.

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

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

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

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

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

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

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

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

(15) WRITE

Function: Writes 128-byte data to the disk.

Entry address: WBOOT + 27H or ØEB2AH

Entry parameter: C = Specifies the mode to write.
= ØØH: Standard write (write after blocking).
= Ø1H: Direct write (write immediately without blocking).
= Ø2H: Write to a sequential file.

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

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

Reference: Same as for READ.

Note: Same as for READ.

Related function: READ

(16) LISTST

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

Entry address: WBOOT + 2AH or ØEB2DH

Entry parameter: None.

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

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

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

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

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

(17) SECTRAN

Function: Translates a logical sector to a physical sector.

Entry address: WBOOT + 2DH or ØEB3ØH

Entry parameter: BC = Logical sector.

Return parameter: HL = Physical sector.

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

(18) PSET

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

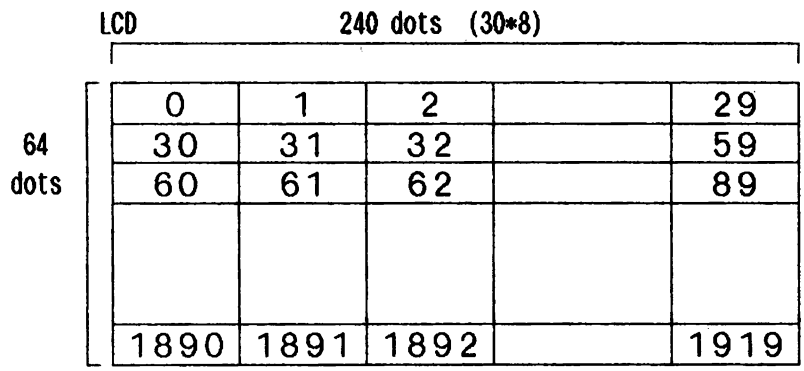
Entry address: WBOOT + 30H or 0EB33H

Entry parameter: B = Data on which the logical operation is to be performed.
C = Logical operation to be performed.
= 01H: AND
= 02H: OR
= 03H: XOR
= Others: PSET does nothing.
HL = VRAM logical address 0<=HL<=1919

Return parameter: A = 00H: Normal termination
= 01H: HL contains an address beyond the specified range.
C = Operation result (loaded with VRAM data if the entry parameter specified in the C register is other than 01H, 02H, and 03H).

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

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



Notes:
PSET only writes the value specified in the B register into VRAM if the C register contains other than 01H, 02H, and 03H.
When reading VRAM data, specify B = 00H and C = 02H or B = 0FFH and C = 01H.

```
                        ************************************************
                                BIOS PSET SAMPLE PROGRAM
                        ************************************************

                        NOTE :
                                This sample program is moving VRAM data
                                right by 1 byte.

                        <> assemble condition <>

                        .Z80

                        <> loading address <>

                        .PHASE  100H

                        <> constant values <>

EB03                    WBOOT           EQU     0EB03H          ; WBOOT entry address
EB33                    PSET            EQU     0EB33H          ; PSET entry address

1000                    MAINSP          EQU     01000H          ; Stack pointer

                        ************************************************
                                MAIN PROGRAM
                        ************************************************

0100                    START:
0100    31 1000                 LD      SP,MAINSP       ; Set stack pointer.
0103    06 40                   LD      B,64            ; Set vertical loop counter.
0105    21 0780                 LD      HL,1920         ; Maximum VRAM byte number.
                        ;
0108                    PLOOP1:
0108    C5                      PUSH    BC              ; Save loop counter 1.
0109    06 1D                   LD      B,30-1          ; Set horizontal loop counter.
010B    2B                      DEC     HL              ; Get new destination address.
                        ;
010C                    PLOOP2:
010C    C5                      PUSH    BC              ; Save loop counter 2.
                        ;
010D    0E 01                   LD      C,01H           ; AND function code.
010F    06 00                   LD      B,00H           ; Clear destination data.
0111    CD EB33                 CALL    PSET            ; Write VRAM with 00H.
                        ;
0114    2B                      DEC     HL              ; Get source address.
0115    0E 02                   LD      C,02H           ; OR function code.
0117    06 00                   LD      B,00H           ; Read VRAM data only.
0119    CD EB33                 CALL    PSET            ; Read VRAM data.
                        ;
011C    23                      INC     HL              ; Get destination address.
011D    41                      LD      B,C             ; Move reading data to setting register.
011E    0E 02                   LD      C,02H           ; OR function code.
0120    CD EB33                 CALL    PSET            ; Write VRAM.
                        ;
0123    2B                      DEC     HL              ; Set next destination address.
0124    C1                      POP     BC              ; Restore loop counter 2.
0125    10 E5                   DJNZ    PLOOP2          ; Not 0, then-loop.
                        ;
0127    C1                      POP     BC              ; Restore loop counter 1.
0128    10 DE                   DJNZ    PLOOP1          ; Not 0, then loop.
                        ;
012A    C3 EB03                 JP      WBOOT           ; Program end.
                        ;
                                END
```

(19) SCRNDUMP

Function: Takes a dump of the current VRAM contents.

Entry address: WBOOT + 33H or ØEB36H

Entry parameter: None.

Return parameter: None.

Explanation:

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

Note:

SCRNDUMP controls the printer using the following procedure:

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

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

SCRNDUMP does not execute normally on the following EPSON printers:

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

SCRNDUMP deletes the cursor from the screen during processing.

Reference:

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

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

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

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

Related function: LIST

See also: 3.9, "I/O Byte"

(20) BEEP

Function: Sounds the buzzer.

Entry address: WBOOT + 36H or ØEB39H

Entry parameter:

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

1. Specifying the note
   - B = Note (13<=B<=6Ø. BEEP generates no sound when B = Ø.)
   - C = Duration (1<=C<=255. The unit is 1ØØ msec.)
2. Specifying the frequency (The MSB of B must be set to 1.)
   - C = Duration (1<=C<=255. The unit is 1ØØ msec.)
   - D = Small loop counter value
   - E = Large loop counter value (BEEP generates no sound when DE = Ø.)

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

Explanation:

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

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

```
T1 = {13 * (D - 1) + 8} / 3.68 usec.
T2 = {33Ø7 * (E - 1) + 24Ø} / 3.68 usec.
Period  T = 2 * (T1 + T2)
Frequency  f = 1 / T Hz
```

The frequency of the buzzer falls mostly in the range from 2ØØ to 4ØØØ Hz.

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

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

Notes:

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

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

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

Reference:

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

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

OVF interrupts are always disabled.

See also: 4.7, "Interrupts"

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

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

                                    END
```

(21) TIMDAT

Function: Performs a specified clock function.

Entry address: WBOOT + 4BH or ØEB4EH

Explanation:

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

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

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

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

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

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

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

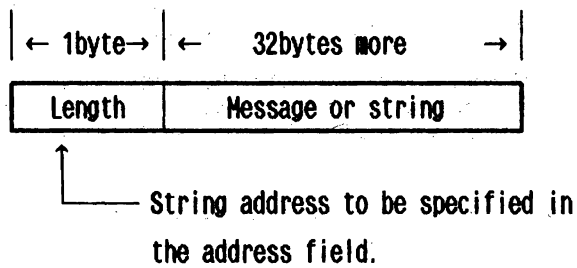
Time descriptor format

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

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

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

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



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

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

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

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

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

ALRMAD always points to ALMMSG (ØF3DFH).

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

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

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

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

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

(21-1) TIMDAT (1) Read time

Function: Reads the current time.

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

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

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

(21-2) TIMDAT (2) Set time

Function: Sets the time.

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

Return parameter: None.

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

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

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

Function: Enables the alarm/wake function.

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

Return parameter: None.

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

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

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

Function: Disables the alarm/wake function.

Entry parameter: C = 81H

Return parameter: None.

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

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

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

Function: Specifies the alarm/wake time.

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

Return parameter: None.

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

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

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

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

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

Function: Reads the alarm/wake time.

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

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

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

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

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

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

                                        END
```

(22) MEMORY

Function: Checks the current bank information.

Entry address: WBOOT + 4EH or ØEB51H

Entry parameter: None.

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

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

See also: Section 4.4, "Bank Switching"

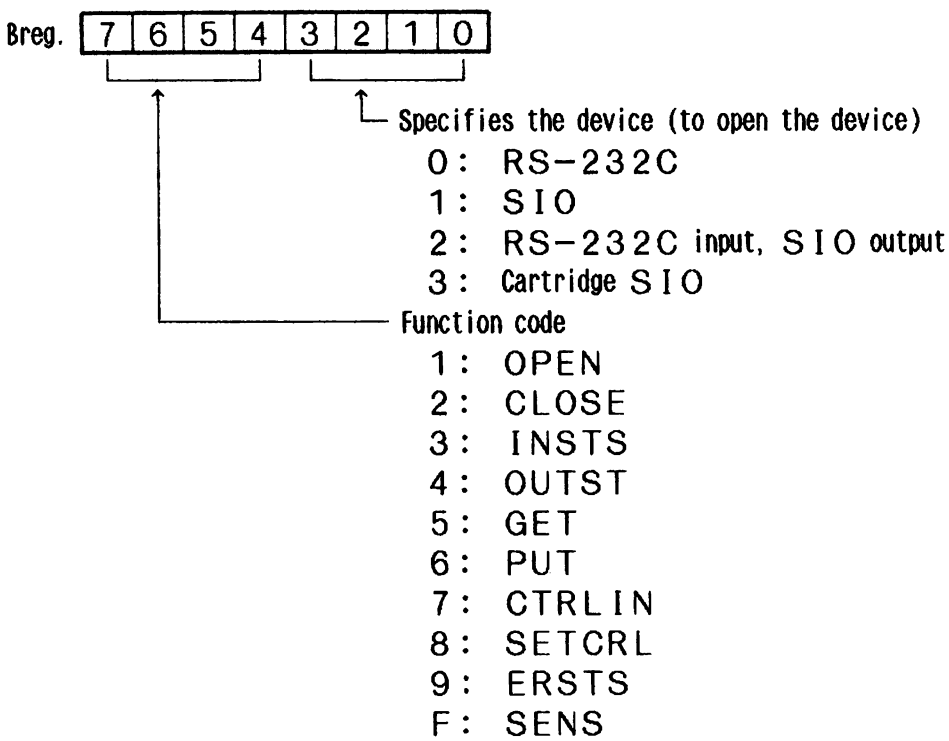
(23) RSIOX

Function: Performs a specified serial communication function.

Entry address: WBOOT + 51H or ØEB54H

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

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



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

Notes:

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

The definition of the terms used here are given below:

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

Reference:

RSIOX uses the system areas shown below:

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

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

RSPSTS (ØF787H) 1 byte
- RSIOX status flags.

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

RSPRBGP (ØF788H) 2 bytes
- Receive buffer get pointer

RSPRBPP (ØF78AH) 2 bytes
- Receive buffer put pointer

RSPRBAD (ØF78CH) 2 bytes
- Receive buffer starting address

RSPRBSZ (ØF78EH) 2 bytes
- Receive buffer size

RSPBITR (ØF790H) 1 byte
- Transmission speed (parameter specified at open time)

Ø2H: 110 bps	ØAH: 1200 bps	1ØH: 38400 bps
Ø4H: 150	ØCH: 2400	8ØH: 75/1200
Ø5H: 200	ØDH: 4800	81H: 1200/75
Ø6H: 300	ØEH: 9600	
Ø8H: 600	ØFH: 19200	

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

Ø2H: 7 bits Ø3H: 8 bits

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

ØØH: No parity Ø1H: Odd parity Ø3H: Even parity

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

Ø1H: 1 bit Ø3H: 2 bits

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

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

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

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

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

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

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

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

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

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

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

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

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

(23-1) RSIOX OPEN

Function: Opens the device to be used.

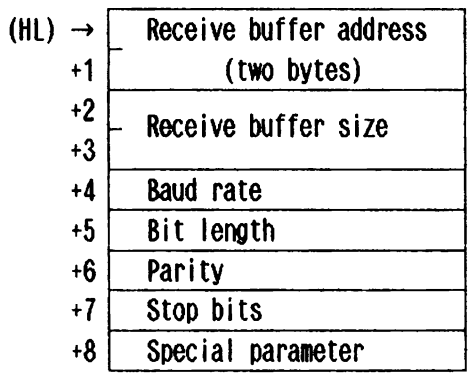
Entry parameter: B = 1XH (X = Ø - 3)
HL = Parameter block starting address
(9 bytes must be reserved. Detailed description is given later.)

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

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

Parameter block structure

The parameter block format is shown below.



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

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

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

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

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

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

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

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

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

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

Parameter block contents on return

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

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

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

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

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

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

(1) Receive buffer get point
The address of the next data to be gotten from the receive buffer
(3) Receive buffer put point
The receive buffer address into which the next data is to be put
(4) Receive buffer address
The receive buffer starting address
(5) Receive buffer size
The receive buffer size

Notes:

XON/XOFF control specification is invalid when the receive buffer size is smaller than 16 bytes.

SI/SO control must be specified when sending 8-bit data in the 7-bit code mode. Codes ØEH and ØFH cannot be transferred when SI/SO control is specified. This means that binary data cannot be sent nor received. See Section 5.2, "Serial Interfaces" for details.

XON/XOFF control must be specified when the transmission speed of the sender is greater than the processing speed of the receiver. This synchronizes the operations of the sender and receiver. Codes 11H and 13H cannot be transferred when XON/XOFF control is specified. Therefore, no binary data can be sent or received. See Section 5.2, "Serial Interfaces" for details.

(23-2) RSIOX CLOSE

Function: Closes the currently open interface.

Entry parameter: B = 2ØH

Return parameter: None.

Explanation:

RSIOX CLOSE disables serial interface receive interrupts and performs the close processing.

Note:

When a WBOOT is executed, the system automatically closes the serial interface.

(23-3) RSIOX INSTS

Function: Indicates whether received data is present in the receive buffer.

Entry parameter: B = 3ØH
HL = Starting address of the field for storing 9-byte return information

Return parameter: Z flag = 1: Normal termination
A = ØFFH: Received data present
= ØØH: No received data present
BC = Number of received data bytes in the buffer
HL = The address specified on entry The nine bytes starting at this address contain the return information described earlier (see RSIOX OPEN).

Z flag = Ø: Abnormal termination
A = Ø3H: The interface is not open.

Explanation:
RSIOX INSTS checks whether or not received data is present in the receive buffer and places the result in the return information block.

XON or XOFF codes in the XON/XOFF mode and SI and SO codes in the SI/SO mode are excluded from the received byte count.

(23-4) RSIOX OUTST

Function: Checks whether the interface is enabled for transmission.

Entry parameter: B = 4ØH
HL = Starting address of the field for storing 9-byte return information

Return parameter: Z flag = 1: Normal termination
A = ØØH: Transmission disabled
= ØFFH: Transmission enabled
HL = The address specified on entry The nine bytes starting at this address contain the return information described earlier (see RSIOX OPEN).

Z flag = Ø: Abnormal termination
A = Ø3H: The interface is not open.

Explanation:
RSIOX OUTST determines whether the interface is enabled or disabled for transmission by checking TxRDY. The interface is disabled for transmission if an XON is received when XON/XOFF control is specified.

RSIOX OUTST places the current transmission status to the return information block.

(23-5) RSIOX GET

Function: Gets one data byte from the receive buffer.

Entry parameter: B = 5ØH
HL = Starting address of the field for storing 9-byte return information

Return parameter: Z flag = 1: Normal termination
A = Received data
HL = The address specified on entry The nine bytes starting at this address contain the return information described earlier (see RSIOX OPEN).

Z flag = Ø: Abnormal termination
A = Ø3H: The interface is not open.
= Ø4H: CTRL/STOP is pressed.
= Ø5H: A receive buffer overflow occurred.

Explanation:

RSIOX GET gets one byte of received data from the receive buffer and loads it into the A register. If no data is present, RSIOS GET waits until a byte is received.

Any power-off or alarm/wake interrupts occurring during execution of RSIOX GET are processed. After processing an interrupt, RSIOX GET resumes processing at the point where the interrupt occurred.

(23-6) RSIOX PUT

Function: Transfers one data byte to the interface.

Entry parameter: B = 60H
C = Send data
HL = Starting address of the field for storing 9-byte return information

Return parameter: Z flag = 1: Normal termination
HL = The address specified on entry The nine bytes starting at this address contain the return information described earlier (see RSIOX OPEN).

Z flag = 0: Abnormal termination
A = 03H: The interface is not open.
= 04H: CTRL/STOP is pressed.

Explanation:

RSIOX PUT checks whether the interface is ready for transmission and, if it is, sends the given data to the interface.

RSIOX PUT determines whether the interface is ready in the same way as RSIOX OUTST. If the interface is not ready, RSIOX PUT waits until it is ready.

If the CTRL/STOP key is pressed while RSIOX PUT is waiting for the interface to get ready, RSIOX PUT returns control to the calling program with 04H in the A register.

Any power-off or alarm/wake interrupts occurring during execution of the RSIOX PUT routine are serviced. After the completion of the interrupt processing, the RSIOX PUT processing resumes at the point where the interrupt occurred.

(23-7) RSIOX CTLIN

Function: Reads the state of a given control line.

Entry parameter: B = 70H

Return parameter: Z flag = 1: Normal termination
A = Control line state

Z flag = 0: Abnormal termination
A = 03H: The interface is not open.

Explanation:
The format of the control line status parameter is shown below:

Bit	Description
7	DSR (Data Set Ready) state = Ø: Active = 1: Not active
6	Not used
5	CTS (Clear To Send) state = Ø: Not active = 1: Active
4	Not used.
3	CD (Carrier Detect) state = Ø: Not active = 1: Active
2 - Ø	Not used

Both CTS and CD are valid only in the RS-232C mode. DSR is valid in RS-232C or SIO mode. In SIO mode, however, the SIN line is used instead of the DSR line.

(23-8) RSIOX SETCTL

Function: Sets the given control line to the specified state.

Entry parameter: B = 8ØH
A = Control line state (described below)

Return parameter: Z flag = 1: Normal termination
Z flag = Ø: Abnormal termination
A = Ø3H: The interface is not open.

Explanation:
The format of the control line status parameter is shown below:

Bit	Description
7 - 2	Not used (must be Ø).
1	RTS (Request To Send) state = Ø: Not active = 1: Active
Ø	DTR (Data Transmit Ready) state = Ø: Not active = 1: Active

Both RTS are valid only in the RS-232C mode. DTR is valid in RS-232C or SIO mode. In SIO mode, however, the SIN line is used instead of the DTR line.

RSIOX SETCTL is used to reset the control line state that was specified when the interface was opened.

(23-9) RSIOX ERSTS

Function: Reads the error status of the interface and clears the error flags.

Entry parameter: B = 9ØH

Return parameter: Z flag = 1: Normal termination
A = Error status (described below)
Z flag = Ø: Abnormal termination
A = Ø3H: The interface is not open.

Explanation:
The format of the control line status parameter is shown below:

Bit	Description
7	DSR (Data Set Ready) state = Ø: Active = 1: Not active
6	Framing error state = Ø: No framing error occurred. = 1: Framing error occurred.
5	Receive overrun error state = Ø: No receive overrun error occurred. = 1: Receive overrun error occurred.
4	Parity error state = Ø: No parity error occurred. = 1: Parity error occurred.
3	CD (Carrier Detect) state = Ø: Not active = 1: Active
2	Receive buffer overflow error state = Ø: No receive buffer overflow error occurred. = 1: Receive buffer overflow error occurred.
1, Ø	Not used

RSIOX ERSTS returns the same status information for the CD and DSR lines regardless of the type of the device.

RSIOX ERSTS clears all error states after reading the current error states.

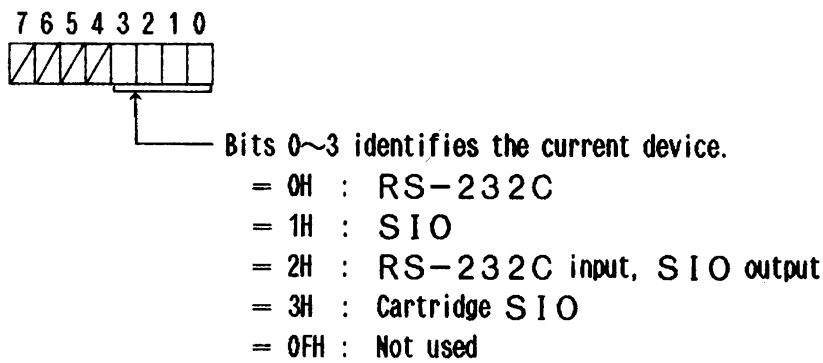
Note:
Once a receive buffer overflow error occurs, any subsequent bytes received until a byte is read from the receive buffer by the RSIOX GET routine are discarded.

(23-1Ø) RSIOX SENS

Function: Returns the serial interface in use.

Entry parameter: B = ØFØH

Return parameter: A = Current serial interface



Explanation:

RSIOX SENS checks and indicate in the A register the serial interface that is currently being used.

```
                        ;       ************************************************
                        ;               BIOS CALL SAMPLE PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> assemble condition <>
                        ;
                                .Z80
                        ;
                        ;       <> loading address <>
                        ;
                                .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
EB03                    WBOOT           EQU     0EB03H          ; WBOOT BIOS entry address
EB06                    CONST           EQU     WBOOT   +03H    ; CONST BIOS entry address
EB09                    CONIN           EQU     WBOOT   +06H    ; CONIN BIOS entry address
EB0C                    CONOUT          EQU     WBOOT   +09H    ; CONOUT BIOS entry address
EB54                    RSIOX           EQU     WBOOT   +51H    ; RSIOX BIOS entry address
EB69                    CALLX           EQU     WBOOT   +66H    ; CALLX BIOS entry address
                        ;
0010                    RSOPN           EQU     10H             ; RS232C OPEN function
0020                    RSCLS           EQU     20H             ; CLOSE function
0030                    RSIST           EQU     30H             ; INPUT STATUS function
0040                    RSOST           EQU     40H             ; OUTPUT STATUS function
0050                    RSGET           EQU     50H             ; GET function
0060                    RSPUT           EQU     60H             ; PUT function
0090                    RSERR           EQU     90H             ; ERROR STATUS function
                        ;
EF31                    SRSADR          EQU     0EF31H          ; System serial parameter.
F52E                    DISBNK          EQU     0F52EH          ; Destination bank area
                        ;
0000                    HELP            EQU     00H             ; HELP code
000D                    CR              EQU     0DH             ; Carriage return code
000A                    LF              EQU     0AH             ; Line feed code
001B                    ESC             EQU     1BH             ; Escape code
0009                    TAB             EQU     09H             ; Tab code
                        ;
003C                    XUSRSCRN        EQU     003CH           ; Change to system screen.
003F                    XSYSSCRN        EQU     003FH           ; Change to user screen.
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
0100                    START:
0100    31 1000                 LD      SP,1000H        ; Set stack pointer.
                        ;
0103    21 EF31                 LD      HL,SRSADR       ; Copy open parameter from system area.
0106    11 02DB                 LD      DE,OPNPRM       ; Application parameter area.
0109    01 0009                 LD      BC,9            ; Parameter number.
010C    ED B0                   LDIR                    ; Copy.
                        ;
010E    21 02DB                 LD      HL,OPNPRM       ; Open parameter.
0111    06 10                   LD      B,RSOPN         ; RS232C open function.
0113    CD EB54                 CALL    RSIOX           ; OPEN.
0116    B7                      OR      A               ; Error return?
0117    C2 EB03                 JP      NZ,WBOOT        ; Yes, then WBOOT.
                        ;
011A                    KEYCHK:
011A    CD EB06                 CALL    CONST           ; Get key inputed status.
011D    3C                      INC     A               ; Input any key?
011E    CC 0137                 CALL    Z,PUT           ; Yes, then put the data.
                        ;
0121    21 02DB                 LD      HL,OPNPRM       ; Get input status.
0124    06 30                   LD      B,RSIST         ;  Input status function.
0126    CD EB54                 CALL    RSIOX           ;  Get input status.
0129    3C                      INC     A               ; If there is receiving data,
012A    CC 0189                 CALL    Z,GET           ;  then get the data
012D    18 EB                   JR      KEYCHK          ; Loop.
                        ;
012F                    PEND:
012F    06 20                   LD      B,RSCLS         ; Close RSIOX
0131    CD EB54                 CALL    RSIOX           ;
0134    C3 EB03                 JP      WBOOT           ; Program end.
                        ;
                        ;       ************************************************
                        ;               PUT INPUTED DATA TO RS232C
                        ;       ************************************************
                        ;
                        ;       NOTE
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;               If inputed data is BREAK key, this program
                        ;                ends.
                        ;               If inputed data is HELP key, put from '0'
                        ;                to '9' to RS232C.
                        ;
                        ;
0137                    PUT:
0137    CD EB09                 CALL    CONIN           ; Get inputed data.
013A    4F                      LD      C,A             ;
013B    FE 03                   CP      03H             ; If inputed key is BREAK,
013D    CA EB03                 JP      Z,WBOOT         ;  then end of program.
                        ;
0140    FE 00                   CP      HELP            ; If inputed key is HELP,
0142    CA 0169                 JP      Z,SEND          ;  then send '0' to '9'.
0145    C5                      PUSH    BC              ; Save input key code.
```

```
0146  C5                      PUSH    BC              ; Save input key code.
0147  FE 0D                   CP      CR              ; If inputed key is RETURN,
0149  0E 0A                   LD      C,LF            ;  then LF console out.
014B  CC EB0C                CALL    Z,CONOUT        ;
014E  C1                     POP     BC              ; Restore input key code.
014F  CD EB0C                CALL    CONOUT          ; Console out inputing data.

0152  06 90                  LD      B,RSERR         ; Get error status.
0154  CD EB54                CALL    RSIOX           ;
0157  E6 74                  AND     01110100B       ; If error is happened.
0159  C4 01CF                CALL    NZ,RGSDSP       ;  then display the error status.

015C  C1                     POP     BC              ; Restore the input key code.
015D  21 02DB                LD      HL,OPNPRM       ; Put inputing data to RS232C.
0160  06 60                  LD      B,RSPUT         ;  Put function code.
0162  CD EB54                CALL    RSIOX           ;  Put data.
0165  C4 01CF                CALL    NZ,RGSDSP       ; If error return, then display error.
0168  C9                     RET                     ;

                             ;***************************************************
                             ;       SEND '0' TO '9'
                             ;***************************************************
                             ;
                             ;NOTE :
                             ;
                             ;<> entry parameter <>
                             ;       NON
                             ;<> return parameter <>
                             ;       NON
                             ;<> preserved registers <>
                             ;       NON
                             ;
                             ;CAUTION :
                             ;       If HELP key is pressed, then return.
0169                         SEND:
0169  0E 30                  LD      C,'0'           ; Start character code.
016B                         SEND10:
016B  C5                     PUSH    BC              ; Save send data.
016C  21 02DB                LD      HL,OPNPRM       ; Put data to RS232C.
016F  06 60                  LD      B,RSPUT         ;  Put function.
0171  CD EB54                CALL    RSIOX           ;  Put.
0174  C1                     POP     BC              ; Restore send data.
0175  0C                     INC     C               ; Send data update.
0176  3E 3A                  LD      A,'9'+1         ; Send '0' to '9'?
0178  B9                     CP      C               ;
0179  20 F0                  JR      NZ,SEND10       ; No.
                             ;
017B  CD EB06                CALL    CONST           ; Check input status.
017E  B7                     OR      A               ; No key is pressed?
017F  28 E6                  JR      Z,SEND          ; Yes.
0181  CD EB09                CALL    CONIN           ; Get pressed key code.
0184  FE 00                  CP      HELP            ; HELP key is pressed?
0186  20 E1                  JR      NZ,SEND         ; No.
0188  C9                     RET                     ;

                             ;***************************************************
                             ;       GET RECIVED DATA
                             ;***************************************************
                             ;
                             ;NOTE :
                             ;
                             ;<> entry parameter <>
                             ;       NON
                             ;<> return parameter <>
                             ;       NON
                             ;<> preserved registers <>
                             ;       NON
                             ;
                             ;CAUTION :
                             ;
0189                         GET:
0189  06 90                  LD      B,RSERR         ; Check error status.
018B  CD EB54                CALL    RSIOX           ; Get error status.
018E  E6 74                  AND     01110100B       ; Error is happened?
0190  C4 01CF                CALL    NZ,RGSDSP       ; Yes, then display error.

0193  21 02DB                LD      HL,OPNPRM       ; Get received data.
0196  06 50                  LD      B,RSGET         ;  Get function code.
0198  CD EB54                CALL    RSIOX           ;  Get.
019B  C4 01CF                CALL    NZ,RGSDSP       ; If error, then display error.

019E  4F                     LD      C,A             ; Console out received data.
019F  CD 01A9                CALL    RVSON           ; Reverse on.
01A2  CD EB0C                CALL    CONOUT          ; Display received data.
01A5  CD 01B6                CALL    RVSOFF          ; Reverse off.
01A8  C9                     RET                     ;

                             ;***************************************************
                             ;       REVERSE MODE ON
                             ;***************************************************
                             ;
                             ;  E :
                             ;
                             ;<> entry parameter <>
                             ;       NON
                             ;<> return parameter <>
                             ;       NON
                             ;<> preserved registers <>
                             ;       BC
                             ;
                             ;CAUTION :
                             ;
01A9                         RVSON:
01A9  C5                     PUSH    BC              ; Save BC register.
01AA  0E 1B                  LD      C,ESC           ; Reverse on command.
01AC  CD EB0C                CALL    CONOUT          ;  ESC + '0'
```

```
01AF    0E 30                   LD      C,'0'           ;
01B1    CD EB0C                 CALL    CONOUT          ;
01B4    C1                      POP     BC              ; Restore BC register.
01B5    C9                      RET                     ;

                        ;***************************************************
                        ;       REVERSE MODE OFF
                        ;***************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               BC
                        ;
                        ;       CAUTION :
                        ;
01B6                    RVSOFF:
01B6    C5                      PUSH    BC              ; Save BC register
01B7    0E 1B                   LD      C,ESC           ; Reverse off command
01B9    CD EB0C                 CALL    CONOUT          ;  ESC + '1'
01BC    0E 31                   LD      C,'1'           ;
01BE    CD EB0C                 CALL    CONOUT          ;
01C1    C1                      POP     BC              ; Restore BC register
01C2    C9                      RET                     ;
                        ;
                        ;***************************************************
                        ;       DISPLAY MESSAGE UNTIL FIND 0
                        ;***************************************************
                        ;
                        ;       NOTE
                        ;
                        ;       <> entry parameter <>
                        ;               HL : Message data top address.
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
01C3                    MSGDSP:
01C3    7E                      LD      A,(HL)          ; Get display data.
01C4    B7                      OR      A               ; Data is end code?
01C5    C8                      RET     Z               ; Yes.

01C6    4F                      LD      C,A             ;
01C7    E5                      PUSH    HL              ; Save data address.
01C8    CD EB0C                 CALL    CONOUT          ; Console out the data.
01CB    E1                      POP     HL              ; Restore data address.
01CC    23                      INC     HL              ; Data adddress update.
01CD    18 F4                   JR      MSGDSP          ; Loop.
                        ;
                        ;***************************************************
                        ;       DISPLAY REGISTERS
                        ;***************************************************
                        ;
                        ;       NOTE
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               All registers
                        ;
                        ;       CAUTION :
                        ;
01CF                    RGSDSP:
01CF    F5                      PUSH    AF              ; Save all registers.
01D0    C5                      PUSH    BC              ;
01D1    D5                      PUSH    DE              ;
01D2    E5                      PUSH    HL              ;

01D3    F5                      PUSH    AF              ; Save display resisters
01D4    C5                      PUSH    BC              ;
01D5    D5                      PUSH    DE              ;
01D6    E5                      PUSH    HL              ;
                        ;
01D7    DD 21 003F              LD      IX,XSYSSCRN     ; Change to system screen.
01DB    3E FF                   LD      A,0FFH          ;  Set system bank.
01DD    32 F52E                 LD      (DISBNK),A      ;
01E0    CD EB69                 CALL    CALLX           ;  Call OS jump table.

01E3    0E 0C                   LD      C,0CH           ; Clear screen & home.
01E5    CD EB0C                 CALL    CONOUT          ;
                        ;
01E8    21 0286                 LD      HL,HLDSP        ; HL register display.
01EB    C1                      POP     BC              ;  HL register data.
01EC    CD 024C                 CALL    BINASC          ;  Convert binary to ASCII.
01EF    21 0291                 LD      HL,DEDSP        ; DE register display.
01F2    C1                      POP     BC              ;  DE register data.
01F3    CD 024C                 CALL    BINASC          ;  Convert binary to ASCII.
01F6    21 029D                 LD      HL,BCDSP        ; BC register display.
01F9    C1                      POP     BC              ;  BC register data.
01FA    CD 024C                 CALL    BINASC          ;  Convert binary to ASCII.
01FD    21 02A8                 LD      HL,AFDSP        ; AF register display.
0200    C1                      POP     BC              ;  AF register data
0201    CD 024C                 CALL    BINASC          ;  Convert binary to ASCII
```

```
0204    21 02AE                 LD      HL,DTDSP        ; Return information area display.
0207    ED 5B 02DB              LD      DE,(OPNPRM)     ;
020B    CD 024A                 CALL    BINASC0         ;
020E    ED 5B 02DD              LD      DE,(OPNPRM+2)   ;
0212    CD 024A                 CALL    BINASC0         ;
0215    ED 5B 02DF              LD      DE,(OPNPRM+4)   ;
0219    CD 024A                 CALL    BINASC0         ;
021C    ED 5B 02E1              LD      DE,(OPNPRM+6)   ;
0220    CD 024A                 CALL    BINASC0         ;
0223    ED 5B 02E3              LD      DE,(OPNPRM+8)   ;
0227    16 00                   LD      D,0             ;
0229    CD 024A                 CALL    BINASC0         ;

022C    21 026B                 LD      HL,RGSMSG       ; Message display.
022F    CD 01C3                 CALL    MSGDSP          ;
0232                    RETRY:
0232    CD EB09                 CALL    CONIN           ; Key input.
0235    FE 00                   CP      HELP            ; HELP key?
0237    20 F9                   JR      NZ,RETRY        ; No.
                        ;
0239                    DSPEND:
0239    DD 21 003C              LD      IX,XUSRSCRN     ; Change to user screen.
023D    3E FF                   LD      A,0FFH          ;  OS bank
023F    32 F52E                 LD      (DISBNK),A      ;
0242    CD EB69                 CALL    CALLX           ;
                        ;
0245    E1                      POP     HL              ; Restore registers.
0246    D1                      POP     DE              ;
0247    C1                      POP     BC              ;
0248    F1                      POP     AF              ;
0249    C9                      RET                     ;
                        ;
                        ;       ************************************************
                        ;               CHANGE BINARY TO ASCII
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               BC : Binary data.
                        ;               HL : ASCII data setting address.
                        ;       <> return parameter <>
                        ;               HL : HL + 2
                        ;          preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
024A                    BINASC0:
024A    43                      LD      B,E             ; E --> B
024B    4A                      LD      C,D             ; D --> C
                        ;
024C                    BINASC:
024C    78                      LD      A,B             ; Change B register.
024D    CD 0251                 CALL    BIN10           ; Convert.
0250    79                      LD      A,C             ; Change C register.
                        ;
0251                    BIN10:
0251    F5                      PUSH    AF              ; Save binary data.
0252    1F                      RRA                     ; Shift 4 bit.
0253    1F                      RRA                     ;
0254    1F                      RRA                     ;
0255    1F                      RRA                     ;
0256    CD 025A                 CALL    CONV00          ; Binary --> ASCII
0259    F1                      POP     AF              ; Restore binary data.
                        ;
025A                    CONV00:
025A    E6 0F                   AND     0FH             ; LSB 4 bit.
025C    FE 0A                   CP      10              ; 0 -- 9 ?
025E    38 06                   JR      C,CONV20        ; Yes.
0260    D6 09                   SUB     9               ; Change 'A' to 'F'.
0262    F6 40                   OR      01000000B       ;
0264    18 02                   JR      CONV25          ;
0266                    CONV20:
0266    F6 30                   OR      00110000B       ; Change '0' to '9'.
0268                    CONV25:
0268    77                      LD      (HL),A          ; Set converted data to (hl)
0269    23                      INC     HL              ; Pointer update.
026A    C9                      RET                     ;
                        ;       ************************************************
                        ;               MESSAGE DATA
                        ;       ************************************************
                        ;
026B                    RGSMSG:
026B    0D 0A                   DB      CR,LF
026D    50 61 72 61             DB      'Parameter display',CR,LF
0271    6D 65 74 65
0275    72 20 64 69
0279    73 70 6C 61
027D    79 0D 0A
0280    48 4C 20 2D             DB      'HL -- '
0284    2D 20
0286                    HLDSP:
0286                            DS      4
028A    09                      DB      TAB
028B    44 45 20 2D             DB      'DE -- '
028F    2D 20
0291                    DEDSP:
0291                            DS      4
0295    0D 0A                   DB      CR,LF
0297    42 43 20 2D             DB      'BC -- '
029B    2D 20
029D                    BCDSP:
029D                            DS      4
02A1    09                      DB      TAB
02A2    41 46 20 2D             DB      'AF -- '
```

```
02A6    2D 20
02A8                            AFDSP:
02A8                                    DS      4
02AC    0D 0A                           DB      CR,LF
02AE                            DTDSP:
02AE                                    DS      20
02C2    0D 0A                           DB      CR,LF
02C4    50 72 65 73                     DB      'Press HELP to continue'
02C8    73 20 48 45
02CC    4C 50 20 74
02D0    6F 20 63 6F
02D4    6E 74 69 6E
02D8    75 65
02DA    00                              DB      0
                                ;
                                ;       ************************************************
                                ;               WORK AREA
                                ;       ************************************************
                                ;
02DB                            OPNPRM:
02DB                                    DS      9               ; RSIOX open parameter area

                                        END
```

(24) MASKI

Function: Sets an interrupt mask or checks the current mask status.

Entry address: WBOOT + 57H or ØEB5AH

Entry parameter: B = Interrupt mask data
C = 75Ø8 interrupt mask data (described later)

Return parameter: B = Old interrupt mask status
C = Old 75Ø8 interrupt mask status

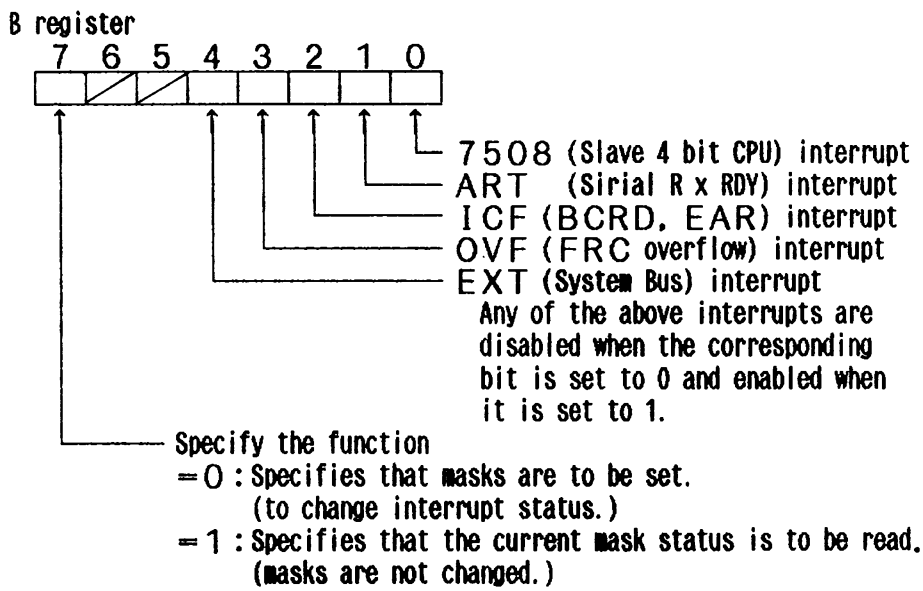
Explanation:
MASKI enables or disables the five types of interrupts supported by the PINE and three types of interrupts that are associated with the 75Ø8. See the next page for entry and return parameters.

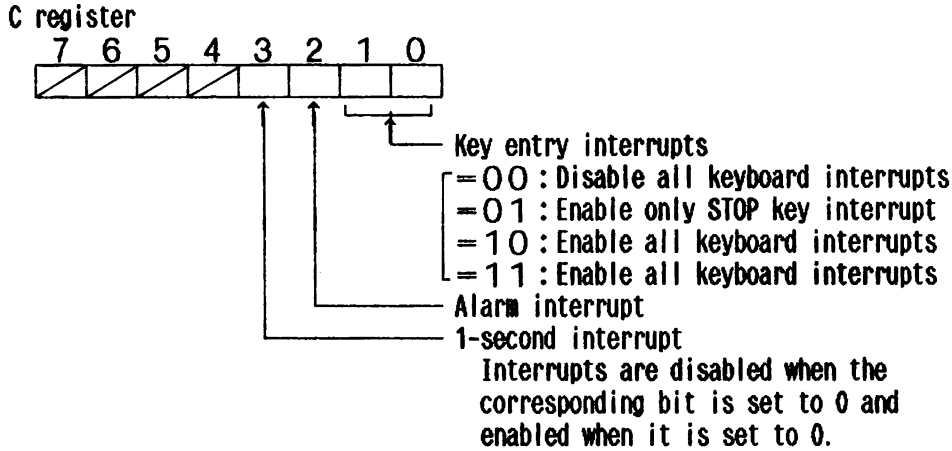
Notes:
When changing the interrupt status, read the current interrupt status with MASKI and set or reset the necessary bit.

It is desirable to restore the original interrupt status after user processing is completed.

See also: Section 4.7, "Interrupts"

Entry parameters





Return parameters

MASKI returns the interrupt mask status at the time when the routine is called. The correspondence between the bits of the BC register and the interrupt types or status is the same as that shown above, except B register bit 7.

Reference:

The current interrupt status is loaded in the following system areas:

ISTS7508 (0EF93H) 1 byte
- Current 7508 interrupt mask status.
Bits 7 - 4: Don't care.
Bit 3: 1-second interrupt
= 0: Disabled
= 1: Enabled
Bit 2: Alarm interrupt
= 0: Disabled
= 1: Enabled
Bits 1, 0: Keyboard interrupt
= 00: All keyboard interrupts disabled.
= 01: Only STOP key interrupt enabled.
= 10: All keyboard interrupts enabled.
= 11: All keyboard interrupts enabled.
The default setting is 0BH.

RZIER (0F53EH) 1 byte
- Current interrupt mask status.
Bits 7 - 5: Don't care.
Bit 4: EXT interrupt
= 0: Disabled
= 1: Enabled
Bit 3: OVF interrupt
= 0: Disabled
= 1: Enabled
Bit 2: ICF interrupt
= 0: Disabled
= 1: Enabled
Bit 1: ART interrupt
= 0: Disabled
= 1: Enabled
Bit 0: 7508 interrupt
= 0: Disabled
= 1: Enabled
The default setting is 09H.

Enabling only STOP key interrupts has no effect when the item keyboard is installed. Attempting to do so will disable all keyboard interrupt.

```
                                ;***************************************************
                                ;       BIOS MASKI SAMPLE PROGRAM
                                ;***************************************************
                                ;
                                ;NOTE :
                                ;       This sample program is that all interrupt
                                ;        makes disable except for STOP key inputing.
                                ;
                                ;<> assemble condition <>
                                ;
                                .Z80
                                ;
                                ;<> loading address <>
                                ;
                                .PHASE  100H
                                ;
                                ;<> constant values <>
                                ;
EB5A                    MASKI   EQU     0EB5AH  ; MASKI entry address.
EB03                    WBOOT   EQU     0EB03H  ; WBOOT entry address.
EB09                    CONIN   EQU     0EB09H  ; CONIN entry address.
                                ;
1000                    MAINSP  EQU     1000H   ; Stack pointer.
                                ;
                                ;***************************************************
                                ;       MAIN PROGRAM
                                ;***************************************************
                                ;
                                ;NOTE :
                                ;
0100                    START:
0100    31 1000                 LD      SP,MAINSP       ; Set stack pointer.
                                ;
0103    06 80                   LD      B,80H           ; Get current interrupt status.
0105    CD EB5A                 CALL    MASKI           ;
0108    C5                      PUSH    BC              ; Save current interrupt status.
                                ;
0109    79                      LD      A,C             ; Disable all key interrupt except
010A    E6 FC                   AND     11111100B       ;  STOP key.
010C    F6 01                   OR      00000001B       ;
010E    4F                      LD      C,A             ;
010F    CD EB5A                 CALL    MASKI           ; Set new interrupt status.
                                ;
                                ;Application inserts the process in this part
                                ; which needs to disable interrupt.
                                ;In case of this sample program, STOP key only can input.
                                ;
0112    CD EB09                 CALL    CONIN           ; Key in. (Only STOP key)
                                ;
0115    C1                      POP     BC              ; Restore interrupt status.
0116    CD EB5A                 CALL    MASKI           ; Restore old interrupt.
                                ;
0119    C3 EB03                 JP      WBOOT           ; Jump WBOOT.
                                ;
                                END
```

(25) LOADX

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

Entry address: WBOOT + 5AH or ØEB5DH

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

Return parameter: A = Contents of the read data

Explanation:

All registers other register than A hold the previous values.

Since LOADX makes no parameter check, normal operation is not guaranteed if a value other than -1 to 2 is specified in C.

See also: Section 4.4, "Bank Switching"

```
                        ;***************************************************
                        ;       LOADX SAMPLE PROGRAM
                        ;***************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is reading from
                        ;               data in target bank.
                        ;
                        ;       <> assemble condition <>
                        ;
                        ;       .Z80
                        ;
                        ;       <> loading address <>
                        ;
                        ;       .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
                        ;       BIOS entry
                        ;
EB03                    WBOOT   EQU     0EB03H          ; Warm Boot entry
EB06                    CONST   EQU     WBOOT   +03H
EB09                    CONIN   EQU     WBOOT   +06H
EB0C                    CONOUT  EQU     WBOOT   +09H    ; Console out entry
EB5D                    LOADX   EQU     WBOOT   +5AH
                        ;
                        ;       Bank 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)
                        ;
                        ;
001B                    ESC     EQU     1BH
0005                    EOL     EQU     05H
0008                    BS      EQU     08H
000D                    CR      EQU     0DH
000A                    LF      EQU     0AH
0003                    STOP    EQU     03H
                        ;
                        ;***************************************************
                        ;       MAIN PROGRAM
                        ;***************************************************
                        ;
                        ;       NOTE :
                        ;               This program is dumping memory.
                        ;
0100                    MAIN:
0100    31 1000                 LD      SP,1000H        ; Set stack pointer.

0103    21 02B9                 LD      HL,MSG01        ; Display opening message.
0106    CD 028E                 CALL    DSPMSG          ;
                        ;
0109                    MAIN10:
0109    CD 01CD                 CALL    GETADDR         ; Get address.
010C    DA EB03                 JP      C,WBOOT         ; End if STOP pressed.
010F    CD 0232                 CALL    GETBNK          ; Select bank.
0112    DA EB03                 JP      C,WBOOT         ; End if STOP is pressed.

0115    CD 011A                 CALL    DUMPF           ; Dump memory.
0118    18 EF                   JR      MAIN10          ; Loop.
                        ;
                        ;***************************************************
                        ;       DUMP FUNCTION
                        ;***************************************************
                        ;
                        ;       NOTE :
                        ;               Dump memory function
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;               If Space bar is pressed, then stop display.
                        ;               If ESC key is pressed, then exit this routine.
                        ;
011A                    DUMPF:
011A    21 02CE                 LD      HL,MSG02        ; Display dump guide line.
011D    CD 028E                 CALL    DSPMSG          ;
                        ;
0120                    DUMP01:
0120    11 03B6                 LD      DE,ASCDATA      ; Getting memory save area.
0123    2A 03BE                 LD      HL,(ADDR)       ; Dump start address.
0126    CD 018D                 CALL    ASCDSP4         ; Display by ASCII.
0129    06 08                   LD      B,08H           ; Loop counter.
012B    3A 03C0                 LD      A,(BANK)        ; Dump target bank.
012E    4F                      LD      C,A             ;
                        ;
012F                    DUMP10:
012F    CD 02A7                 CALL    SPACE           ; Display space.
0132    CD EB5D                 CALL    LOADX           ; Get memory.
0135    12                      LD      (DE),A          ; Set getting data.
0136    CD 0183                 CALL    ASCDSP2         ; Display by ASCII.
0139    23                      INC     HL              ; Pointer update.
013A    13                      INC     DE              ;
013B    10 F2                   DJNZ    DUMP10          ; Loop 8 times.

013D    22 03BE                 LD      (ADDR),HL       ; New address.
0140    CD 02A7                 CALL    SPACE           ; Display 2 spaces.
0143    CD 02A7                 CALL    SPACE           ;
0146    CD 015E                 CALL    ASCDSPX         ; Display memory by ASCII.
```

```
0149  CD EB06          CALL    CONST           ; Input any key?
014C  3C               INC     A               ;
014D  20 D1            JR      NZ,DUMP01       ; No.
                       ;
014F  CD EB09          CALL    CONIN           ; Get inputed key.
0152  FE 1B            CP      ESC             ; ESC?
0154  C8               RET     Z               ; Yes.
0155  FE 20            CP      20H             ; Space?
0157  20 C7            JR      NZ,DUMP01       ; No.
                       ;
0159  CD EB09          CALL    CONIN           ; Stop dump until any key inputed.
015C  18 C2            JR      DUMP01          ;
                       ;
                       ;
                       ;
015E                   ASCDSPX:
015E  F5               PUSH    AF              ; Save registers.
015F  C5               PUSH    BC              ;
0160  D5               PUSH    DE              ;
0161  E5               PUSH    HL              ;
                       ;
0162  21 03B6          LD      HL,ASCDATA      ; Save data address.
0165  06 08            LD      B,08H           ; Loop counter.
0167                   ASCDX1:
0167  7E               LD      A,(HL)          ; Get data.
0168  FE 20            CP      20H             ; Control code?
016A  30 02            JR      NC,ASCDX2       ; No.
                       ;
016C  3E 2E            LD      A,'.'           ; Change data to '.'.
016E                   ASCDX2:
016E  CD 029A          CALL    CONOUTS         ; Display data.
0171  23               INC     HL              ;
0172  10 F3            DJNZ    ASCDX1          ; Loop 8 times.
                       ;
0174  0E 0D            LD      C,CR            ; Carriage return.
0176  CD EB0C          CALL    CONOUT          ;
0179  0E 0A            LD      C,LF            ; Line feed.
017B  CD EB0C          CALL    CONOUT          ;
                       ;
017E  E1               POP     HL              ; Restore registers.
017F  D1               POP     DE              ;
0180  C1               POP     BC              ;
0181  F1               POP     AF              ;
0182  C9               RET                     ;
                       ;
                       ;
                       ;
0183                   ASCDSP2:
0183  F5               PUSH    AF              ; Save registers.
0184  C5               PUSH    BC              ;
0185  D5               PUSH    DE              ;
0186  E5               PUSH    HL              ;
                       ;
0187  21 0000          LD      HL,0000H        ; Set binary data to HL
018A  6F               LD      L,A             ;  A --> HL
018B  18 16            JR      ASCD42          ;
                       ;
                       ;
                       ;
018D                   ASCDSP4:
018D  F5               PUSH    AF              ; Save registers.
018E  C5               PUSH    BC              ;
018F  D5               PUSH    DE              ;
0190  E5               PUSH    HL              ;
                       ;
0191  11 1000          LD      DE,4096         ; 1000H
0194  CD 01BB          CALL    ASCD45          ; Get 16**3
0197  CD 029A          CALL    CONOUTS         ; Display data.
                       ;
019A  11 0100          LD      DE,256          ; 100H
019D  CD 01BB          CALL    ASCD45          ; Get 16**2
01A0  CD 029A          CALL    CONOUTS         ; Display data.
                       ;
01A3                   ASCD42:
01A3  11 0010          LD      DE,16           ; 10H
01A6  CD 01BB          CALL    ASCD45          ; Get 16**1
01A9  CD 029A          CALL    CONOUTS         ; Display data.
                       ;
01AC  7D               LD      A,L             ; Get 16**0
01AD  C6 30            ADD     A,30H           ;  Change to ASCII.
01AF  4F               LD      C,A             ;
01B0  CD 01C6          CALL    ASCD48          ;
01B3  CD 029A          CALL    CONOUTS         ; Display data.
                       ;
01B6  E1               POP     HL              ; Save registers.
01B7  D1               POP     DE              ;
01B8  C1               POP     BC              ;
01B9  F1               POP     AF              ;
01BA  C9               RET                     ;
                       ;
                       ;
                       ;
01BB                   ASCD45:
01BB  0E 30            LD      C,'0'           ;
                       ;
01BD                   ASCD46:
01BD  B7               OR      A               ; Reset carry bit.
01BE  ED 52            SBC     HL,DE           ;
01C0  38 03            JR      C,ASCD47        ;
                       ;
01C2  0C               INC     C               ; Counter increase.
01C3  18 F8            JR      ASCD46          ; Loop
                       ;
01C5                   ASCD47:
01C5  19               ADD     HL,DE           ; Restore data
```

```
01C6                    ASCD48:
01C6    79                      LD      A,C             ;
01C7    FE 3A                   CP      ':'             ; If larger than '9',
01C9    D8                      RET     C               ;  then convert to 'A' -- 'F'.
01CA    C6 07                   ADD     A,'A'-':'       ;
;
01CC    C9                      RET                     ;
;
;
;               ************************************************
;                       INPUT ADDRESS DATA
;               ************************************************
;
;               NOTE :
;                       Get address data routine
;
;               <> entry parameter <>
;                       NON
;               <> return parameter <>
;                       CY  : Return information
;                          =0 -- Normal end
;                          =1 -- ESC key inputed
;               <> preserved registers <>
;                       NON
;
;               CAUTION :
;
01CD                    GETADDR:
01CD    21 02F5                 LD      HL,MSG03        ; Display inputing address message.
01D0    CD 028E                 CALL    DSPMSG          ;
;
01D3    21 03B3                 LD      HL,INCNT        ; Inputed data counter reset.
01D6    36 00                   LD      (HL),00H        ;
01D8    11 03B4                 LD      DE,INDATA       ; Inputed data store area.
;
01DB                    GETA10:
01DB    CD 02AF                 CALL    CONINS          ; Get inputed key.
01DE    FE 03                   CP      STOP            ; STOP?
01E0    37                      SCF                     ;  Set carry flag.
01E1    C8                      RET     Z               ; Yes.
01E2    FE 08                   CP      BS              ; Back space?
01E4    28 18                   JR      Z,GETA20        ; Yes.
01E6    FE 0D                   CP      CR              ; Carriage return?
01E8    28 26                   JR      Z,GETA30        ; Yes.
;
01EA    CD 021D                 CALL    CHKHEX          ; Check HEX data.
01ED    38 EC                   JR      C,GETA10        ; Not hexa data.
;
01EF    4F                      LD      C,A             ;
01F0    7E                      LD      A,(HL)          ; Inputed counter check.
01F1    FE 04                   CP      04H             ; Max 4 character.
01F3    30 E6                   JR      NC,GETA10       ; Character over.
;
01F5    79                      LD      A,C             ;
01F6    CD 029A                 CALL    CONOUTS         ; Display inputed char.
01F9    12                      LD      (DE),A          ; Store data.
01FA    34                      INC     (HL)            ; Counter update.
01FB    13                      INC     DE              ; Pointer update.
01FC    18 DD                   JR      GETA10          ; Loop.
;
01FE                    GETA20:
01FE    AF                      XOR     A               ; Back space process.
01FF    BE                      CP      (HL)            ; No inputed charcter?
0200    28 D9                   JR      Z,GETA10        ; Yes.
0202    35                      DEC     (HL)            ; Counter decrement.
0203    1B                      DEC     DE              ; Pointer decrement.
0204    3E 08                   LD      A,BS            ; Cursor left.
0206    CD 029A                 CALL    CONOUTS         ;
0209    3E 05                   LD      A,EOL           ; Erase end of line.
020B    CD 029A                 CALL    CONOUTS         ;
020E    18 CB                   JP      GETA10          ;
;
0210                    GETA30:
0210    46                      LD      B,(HL)          ; Carriage return process.
0211    21 03B4                 LD      HL,INDATA       ; Change ASCII to binary.
0214    CD 0262                 CALL    ASCBIN          ;
0217    ED 53 03BE              LD      (ADDR),DE       ; Store converted data.
021B    B7                      OR      A               ; Carry off.
021C    C9                      RET                     ;
;
;
021D                    CHKHEX:
021D    FE 30                   CP      '0'             ; 00H -- 2FH?
021F    D8                      RET     C               ; Yes.
0220    FE 3A                   CP      ':'             ; 30H -- 39H?
0222    3F                      CCF                     ;
0223    D0                      RET     NC              ; Yes.
0224    FE 41                   CP      'A'             ; 3AH -- 40H?
0226    D8                      RET     C               ; Yes.
0227    FE 47                   CP      'G'             ; 'A' -- 'F'?
0229    3F                      CCF                     ;
022A    D0                      RET     NC              ; Yes.
022B    FE 61                   CP      'a'             ; 47H -- 60H?
022D    D8                      RET     C               ; Yes.
022E    FE 67                   CP      'g'             ; 'a' -- 'f'?
0230    3F                      CCF                     ;
0231    C9                      RET                     ;
```

```
                        ;***********************************************
                        ;       SELECT BANK
                        ;***********************************************
                        ;
                        ;       NOTE :
                        ;               Select bank routine
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;          return parameter <>
                        ;               CY  : Return information
                        ;                  =0 -- Normal end
                        ;                  =1 -- ESC key inputed
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0232                    GETBNK:
0232    21 033C                 LD      HL,MSG04        ; Display selecting bank message.
0235    CD 028E                 CALL    DSPMSG          ;
                        ;
0238                    GETB05:
0238    CD EB09                 CALL    CONIN           ; Key in.
023B    FE 03                   CP      STOP            ; STOP?
023D    37                      SCF                     ;
023E    C8                      RET     Z               ; Yes.
                        ;
023F    FE 31                   CP      '1'             ; '1' is system bank.
0241    0E FF                   LD      C,SYSBANK       ;
0243    28 14                   JR      Z,GETB10        ;
0245    FE 32                   CP      '2'             ; '2' is bank 0.
0247    0E 00                   LD      C,BANK0         ;
0249    28 0E                   JR      Z,GETB10        ;
024B    FE 33                   CP      '3'             ; '3' is bank 1.
024D    0E 01                   LD      C,BANK1         ;
024F    28 08                   JR      Z,GETB10        ;
0251    FE 34                   CP      '4'             ; '4' is bank 2.
0253    0E 02                   LD      C,BANK2         ;
0255    28 02                   JR      Z,GETB10        ;
0257    18 DF                   JR      GETB05          ; Other inputed character.
                        ;
0259                    GETB10:
0259    CD 029A                 CALL    CONOUTS         ; display inputed code.
                        ;
025C    79                      LD      A,C             ; Set data.
025D    32 03C0                 LD      (BANK),A        ;
0260    B7                      OR      A               ; Carry off.
0261    C9                      RET                     ;
                        ;
                        ;
                        ;***********************************************
                        ;       CHANGE ASCII TO BINARY
                        ;***********************************************
                        ;
                        ;       NOTE :
                        ;               Change ASCII HEX data to binary data.
                        ;
                        ;       <> entry parameter <>
                        ;               HL : ASCII data top address.
                        ;               B  : Data count
                        ;       <> return parameter <>
                        ;               DE : Binary data
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0262                    ASCBIN:
0262    11 0000                 LD      DE,0000H
0265    78                      LD      A,B
0266    B7                      OR      A
0267    C8                      RET     Z
                        ;
0268                    ASC10:
0268    C5                      PUSH    BC
0269    7E                      LD      A,(HL)
026A    CD 0284                 CALL    ASC20
026D    4F                      LD      C,A
026E    06 00                   LD      B,00H
0270    EB                      EX      DE,HL
0271    09                      ADD     HL,BC
0272    EB                      EX      DE,HL
0273    C1                      POP     BC
0274    05                      DEC     B
0275    C8                      RET     Z
                        ;
0276    C5                      PUSH    BC
0277    06 04                   LD      B,04H
0279                    ASC15:
0279    B7                      OR      A
027A    CB 13                   RL      E
027C    CB 12                   RL      D
027E    10 F9                   DJNZ    ASC15
                        ;
0280    23                      INC     HL
0281    C1                      POP     BC
0282    18 E4                   JR      ASC10
                        ;
                        ;
0284                    ASC20:
0284    D6 30                   SUB     '0'
0286    FE 0A                   CP      0AH
0288    D8                      RET     C
0289    E6 DF                   AND     11011111B
028B    D6 1B                   SUB     'A'-'0'+10
028D    C9                      RET
```

```
                        ;
                        ;       ************************************************
                        ;               MESSAGE DISPLAY
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Display message until found 00H.
                        ;
                        ;       <> entry parameter <>
                        ;               HL : Message data top address.
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
028E                    DSPMSG:
028E    7E              LD      A,(HL)
028F    B7              OR      A
0290    C8              RET     Z

0291    4F              LD      C,A
0292    E5              PUSH    HL
0293    CD EB0C         CALL    CONOUT
0296    E1              POP     HL
0297    23              INC     HL
0298    18 F4           JR      DSPMSG

                        ************************************************
                                CONSOLE OUT
                        ************************************************

                        NOTE :

                        * entry parameter <>
                                A : Console out data
                        <> return parameter <>
                                NON
                        <> preserved registers <>
                                All registers

                        CAUTION
                        ;
029A                    CONOUTS:
029A    F5              PUSH    AF
029B    C5              PUSH    BC
029C    D5              PUSH    DE
029D    E5              PUSH    HL
029E    4F              LD      C,A
029F                    CONO10:
029F    CD EB0C         CALL    CONOUT
02A2    E1              POP     HL
02A3    D1              POP     DE
02A4    C1              POP     BC
02A5    F1              POP     AF
02A6    C9              RET
                        ;
                        ;
02A7                    SPACE:
02A7    F5              PUSH    AF
02A8    C5              PUSH    BC
02A9    D5              PUSH    DE
02AA    E5              PUSH    HL
02AB    0E 20           LD      C,20H
02AD    18 F0           JR      CONO10

                        ************************************************
                                CONSOLE IN
                        ************************************************

                        NOTE :

                        <> entry parameter <>
                                NON
                        * return parameter <>
                                A : Console in data
                        <> preserved registers <>
                                All registers without AF

                        CAUTION :
                        ;
02AF                    CONINS:
02AF    C5              PUSH    BC
02B0    D5              PUSH    DE
02B1    E5              PUSH    HL
02B2    CD EB09         CALL    CONIN
02B5    E1              POP     HL
02B6    D1              POP     DE
02B7    C1              POP     BC
02B8    C9              RET
                        ;
02B9                    MSG01:
02B9    53 74 61 72     DB      'Start dump program',CR,LF
02BD    74 20 64 75
02C1    6D 70 20 70
02C5    72 6F 67 72
02C9    61 6D 0D 0A
02CD    00              DB      00H
02CE                    MSG02:
02CE    0D 0A           DB      CR,LF
02D0    0D 0A           DB      CR,LF
02D2    41 64 64 72     DB      'Addr 00 01 02 03 04 05 06 07  ASCII',CR,LF
02D6    20 30 30 20
02DA    30 31 20 30
02DE    32 20 30 33
02E2    20 30 34 20
```

```
02E6    30 35 20 30
02EA    36 20 30 37
02EE    20 20 41 53
02F2    43 49 49 0D
02F6    0A
02F7    00                      DB      00H
02F8                    MSG03:
02F8    0D 0A                   DB      CR,LF
02FA    49 6E 70 75             DB      'Input dump start address (hexa data)',CR,LF
02FE    74 20 64 75
0302    6D 70 20 73
0306    74 61 72 74
030A    20 61 64 64
030E    72 65 73 73
0312    20 28 68 65
0316    78 61 20 64
031A    61 74 61 29
031E    0D 0A
0320    20 20 28 45             DB      '  (Exit by pressing STOP)',CR,LF
0324    78 69 74 20
0328    62 79 20 70
032C    72 65 73 73
0330    69 6E 67 20
0334    53 54 4F 50
0338    29 0D 0A
033B    00                      DB      00H
033C                    MSG04:
033C    0D 0A                   DB      CR,LF
033E    53 65 6C 65             DB      'Select dump bank',CR,LF
0342    63 74 20 64
0346    75 6D 70 20
034A    62 61 6E 6B
034E    0D 0A
0350    20 20 31 20             DB      '  1 -- System bank',CR,LF
0354    2D 2D 20 53
0358    79 73 74 65
035C    6D 20 62 61
0360    6E 6B 0D 0A
0364    20 20 32 20             DB      '  2 -- Bank 0 (RAM)',CR,LF
0368    2D 2D 20 42
036C    61 6E 6B 20
0370    30 20 28 52
0374    41 4D 29 0D
0378    0A
0379    20 20 33 20             DB      '  3 -- Bank 1',CR,LF
037D    2D 2D 20 42
0381    61 6E 6B 20
0385    31 0D 0A
0388    20 20 34 20             DB         4 -- Bank 2',CR,LF
038C    2D 2D 20 42
0390    61 6E 6B 20
0394    32 0D 0A
0397    20 20 28 45             DB         (Exit by pressing STOP)',CR,LF
039B    78 69 74 20
039F    62 79 20 70
03A3    72 65 73 73
03A7    69 6E 67 20
03AB    53 54 4F 50
03AF    29 0D 0A
03B2    00                      DB      00H
                        ;
03B3                    INCNT:
03B3                            DS      1
03B4                    INDATA:
03B4                            DS      2
03B6                    ASCDATA:
03B6                            DS      8
03BE                    ADDR:
03BE                            DS      2
03C0                    BANK:
03C0                            DS      1

                                END
```

(26) STORX

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

Entry address: WBOOT + 5DH or ØEB6ØH

Entry parameter: A = Data to be written
C = Bank to which data is to be written
= ØFFH: System bank
= ØØH: Bank Ø
= Ø1H: Bank 1
= Ø2H: Bank 2
HL = Address of the data to be written

Return parameter: None.

Explanation:
All registers retain the previous values.

Since STORX makes no parameter check, normal operation is not guaranteed if a value other than -1 to 2 is specified in C.

Note:
The C register should be set to ØØH because data can be written only in RAM.

See also: Section 4.4, "Bank Switching"

(27) LDIRX

Function: Transfers the specified number of data bytes on the specified bank onto bank Ø.

Entry address: WBOOT + 6ØH or ØEB63H

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

Return parameter: BC = ØØØØH
DE = (DE on entry) + (BC on entry)
HL = (HL on entry) + (BC on entry)

Explanation:
Since LDIRX makes no error check, normal operation is not guaranteed if a value other than -1 to 2 is specified in C. This routine is equivalent to the LDIR instruction with a bank switching capability.

See also: Section 4.4, "Bank Switching"

(28) JUMPX

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

Entry address: WBOOT + 63H or ØEB66H

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

Return parameter: None.

Explanation:
All registers retain the previous values when control is transferred to the destination of jump.

Since JUMPX makes no parameter check, normal operation is not guaranteed if a value other than -1 to 2 is specified in DISBNK.

Notes:
The system-supplied stack in the BIOS is used when this routine is called. Consequently, the called routine must reserve a stack for itself. Otherwise, the CPU will hang up when the routine calls BIOS or BDOS.

Since the system is in a state in which it is still executing BIOS, the user program should forces the system to exit that state by calling RSPSTBIOS (ØFF96H).

See also: Section 4.4, "Bank Switching"

(29) CALLX

Function: Calls the specified bank address.

Entry address: WBOOT + 66H or ØEB69H

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

Return parameter:
All registers except IX and IY retain the previous values.

Explanation:
When control is transferred to the called routine, all registers retains the values set up when CALLX is called.

Since CALLX makes no parameter check, normal operation is not guaranteed if a value other than -1 to 2 is specified in DISBNK.

Notes:

When this routine is called, the system-supplied stack in the BIOS is used and the system is in a state in which it is still executing BIOS. Accordingly, the user must take the notes given in the JUMPX description into consideration.

If the called program reserves its own stack, it must restore the original stack when returning control.

This routine should be used to call utility routines in the system ROM.

See also: Section 4.4, "Bank Switching"

(30) GETPFK

Function: Reads the character string from a specified PF key.

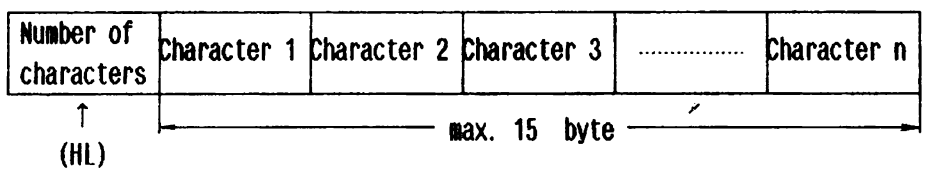
Entry address: WBOOT + 69H or 0EB6CH

Entry parameter: C = PF key number
= 00H - 09H: Specifies a PF key on the standard keyboard.
= 40H - 7EH: Specifies an item function key.
HL = Starting address of the 16-byte buffer into which the character string is to be read.

Return parameter: HL = Retains the previous value.

Explanation:

GETPFK gets the character string defined for a given PF key. The format of the character string is shown below.



"Number of characters" indicates the number of characters in the string defined for the PF key. It must be in the range from 00H to 0FH. 00H indicates that no string is defined for the PF key.

Related function: PUTPFK

See also: Section 3.5, "Keyboard"

```
                        ;***********************************************
                        ;       BIOS GETPFK SAMPLE PROGRAM
                        ;***********************************************
                        ;
                        ;NOTE :
                        ;       This sample program is displaying present
                        ;       defined function key list.
                        ;
                        ;<> assemble condition <>
                        ;
                        .Z80
                        ;
                        ;<> loading address <>
                        ;
                        .PHASE  100H
                        ;
                        ;<> constant values <>
                        ;
EB03                    WBOOT           EQU     0EB03H  ; WBOOT entry address.
EB09                    CONIN           EQU     0EB09H  ; CONIN entry address.
EB0C                    CONOUT          EQU     0EB0CH  ; CONOUT entry address.
EB6C                    GETPFK          EQU     0EB6CH  ; GETPFK entry address.
                        ;
1000                    MAINSP          EQU     1000H   ; Stack pointer.
                        ;
000D                    CR              EQU     0DH     ; Carriage return code.
000A                    LF              EQU     0AH     ; Line feed code.
0003                    BREAK           EQU     03H     ; STOP code.
                        ;
                        ;***********************************************
                        ;       MAIN PROGRAM
                        ;***********************************************
                        ;
                        ;NOTE :
                        ;
0100                    START:
0100    31 1000                 LD      SP,MAINSP       ; Set stack pointer.
                        ;
0103    06 0A                   LD      B,09H-00H+1     ; Loop counter.
0105    0E 00                   LD      C,00H           ; PF1 to PF10
                        ;
0107                    LOOP1:
0107    C5                      PUSH    BC              ; Save counter & PFK code.
0108    21 014E                 LD      HL,PFKBUF       ; PFK data reading area.
010B    CD EB6C                 CALL    GETPFK          ; Get PFK data.
010E    CD 012A                 CALL    PFKDSP          ; Display PFK data.
0111    C1                      POP     BC              ; Restore counter & PFK code.
0112    0C                      INC     C               ; Increase PFK code.
0113    10 F2                   DJNZ    LOOP1           ; Loop 10 times.
                        ;
0115    06 3F                   LD      B,7EH-40H+1     ; Loop counter.
0117    0E 40                   LD      C,40H           ; ITEM FK code.
                        ;
0119                    LOOP2:
0119    C5                      PUSH    BC              ; Save counter & ITEM FK code.
011A    21 014E                 LD      HL,PFKBUF       ; PFK data reading area.
011D    CD EB6C                 CALL    GETPFK          ; Get PFK data.
0120    CD 012A                 CALL    PFKDSP          ; Display PFK data.
0123    C1                      POP     BC              ; Restore counter & ITEM FK code.
0124    0C                      INC     C               ; Increase ITEMFK code.
0125    10 F2                   DJNZ    LOOP2           ; Loop 63 times.
                        ;
0127    C3 EB03                 JP      WBOOT           ; End of main program.
                        ;
                        ;***********************************************
                        ;       DISPLAY FUNCTION KEY DATA
                        ;***********************************************
                        ;
                        ;NOTE :
                        ;       Display the function key data and
                        ;        when input any key except STOP, return.
                        ;       If STOP key is pressed, then WBOOT.
                        ;
                        ;<> entry parameter <>
                        ;       HL : PFK string top address.
                        ;<> return parameter <>
                        ;       NON
                        ;<> preserved registers <>
                        ;       NON
                        ;
                        ;CAUTION :
                        ;
012A                    PFKDSP:
012A    7E                      LD      A,(HL)          ; Get string length.
012B    B7                      OR      A               ; Length is 0?
012C    28 0D                   JR      Z,PFKEND        ; Yes.
                        ;
012E    47                      LD      B,A             ; String length --> counter
012F    23                      INC     HL              ; String pointer 1 increase.
                        ;
0130                    PFKLOOP:
0130    E5                      PUSH    HL              ; Save pointer.
0131    C5                      PUSH    BC              ; Save counter.
0132    4E                      LD      C,(HL)          ; Get display data.
0133    CD EB0C                 CALL    CONOUT          ; Display the data.
0136    C1                      POP     BC              ; Restore counter.
0137    E1                      POP     HL              ; Restore pointer.
0138    23                      INC     HL              ; Pointer update.
0139    10 F5                   DJNZ    PFKLOOP         ; Loop (by string length)
                        ;
013B                    PFKEND:
013B    0E 0D                   LD      C,CR            ; Display CR & LF.
013D    CD EB0C                 CALL    CONOUT          ;
0140    0E 0A                   LD      C,LF            ;
0142    CD EB0C                 CALL    CONOUT          ;
```

```
0145    CD EB09                 CALL    CONIN           ; Get any inputed key.
0148    FE 03                   CP      BREAK           ; STOP code?
014A    CA EB03                 JP      Z,WBOOT         ; Yes, then WBOOT.
014D    C9                      RET                     ; else return.
                        ;
                        ;       ************************************************
                        ;       WORK AREA
                        ;       ************************************************
                        ;
014E                    PFKBUF:
014E                            DS      16              ; PFK data reading area.

                                END
```

(31) PUTPFK

Function: Defines a character string for a PF key.

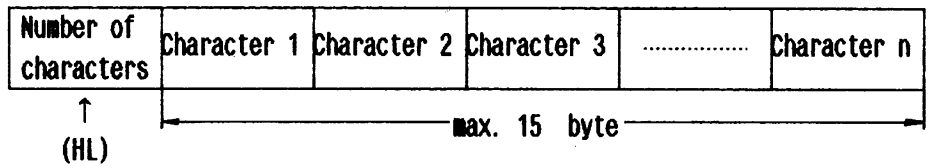
Entry address: WBOOT + 6CH or ØEB6FH

Entry parameter: C = PF key number or control information
= ØØH - Ø9H: Specifies a PF key on the standard keyboard.
= 4ØH - 7EH: Specifies an item function key.
= ØFFH: Cancels all item functions.
= ØFEH: Resets the item flag.
= ØFDH: Sets the item flag.
HL = Starting address of the character string to be assigned

Return parameter: HL = Retains the previous value.

Explanation:

PUTPFK assigns a character string to a PF key in the 16-byte format shown below. The maximum string length is 15 characters.



"Number of characters" specifies the number of characters to be assigned to the specified PF key. It must be in the range from ØØH to ØFH ØØH indicates that no string is defined for the PF key.

Item functions are enabled if the item flag is set. This ensures that the standard keyboard keys whose number are in the range 4ØH to 7EH can be used as function keys by assigning item functions with the item flag set.

See also: Section 3.5, "Keyboard"

Reference:

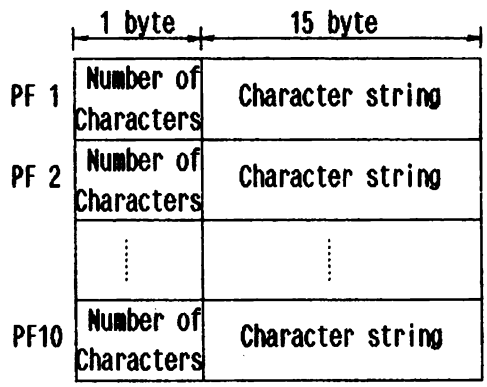
The system uses the following areas for setting function keys:

YPFRSTR (ØF12H) 2 bytes
- PF key table starting address

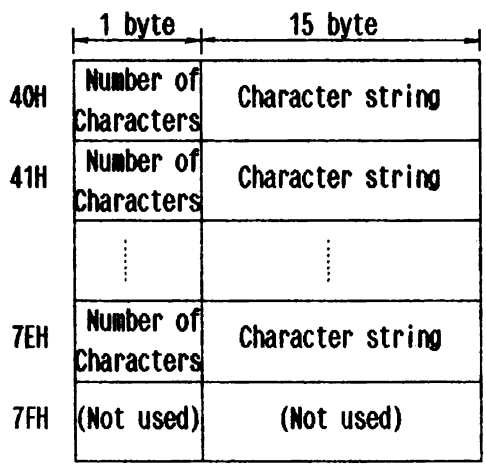
The default value is WPFKTBL (ØF545H). The YPFRSTR must be located at address 8ØØØH or higher.

ITEMFLG (ØF1DH) 1 byte
- Item mode flag
  - = ØØH: Normal mode
  - = 8ØH: Item mode

WPFKTBL (ØF545H) 16Ø bytes
- PF key table



RWITEMTOP (ØCCØØH) 1Ø24 bytes
- Item function key table



PFKTAB (ØF2DH) 16Ø bytes
- PF key initial data

The table format is the same as that of WPFKTBL. The contents of this area are copied into WPFKTBL by BOOT or WBOOT that is invoked during execution of a nonresident program.

The PF keys are initialized as follows:

```
PF1: dir ␣      PF5: basic ␣        PF9: filink (CR)↵
  2: type ␣       6: config (CR)↵    1Ø:
  3: stat ␣       7: submit ␣
  4: pip          8: term (CR)↵
```

The initial values of the item function keys (loaded in RWITEMTOP) are listed below:

The numbers are item function key numbers, and the values are the Character codes of the characters for which the keys are defined.

Number	Value	Number	Value	Number	Value	Number	Value	Number	Value	Number	Value	Number	Value	Number	Value
40H	40H	48H	48H	50H	50H	58H	58H	60H	1BH	68H	21H	70H	29H	78H	3AH
41H	41H	49H	49H	51H	51H	59H	59H	61H	13H	69H	22H	71H	5EH	79H	3BH
42H	42H	4AH	4AH	52H	52H	5AH	5AH	62H	00H	6AH	23H	72H	5FH	7AH	3CH
43H	43H	4BH	4BH	53H	53H	5BH	5BH	63H	12H	6BH	24H	73H	60H	7BH	3DH
44H	44H	4CH	4CH	54H	54H	5CH	5CH	64H	0BH	6CH	25H	74H	7BH	7CH	3FH
45H	45H	4DH	4DH	55H	55H	5DH	5DH	65H	0CH	6DH	26H	75H	7CH	7DH	3FH
46H	46H	4EH	4EH	56H	56H	5EH	20H	66H	7FH	6EH	27H	76H	7DH	7EH	20H
47H	47H	4FH	4FH	57H	57H	5FH		67H	08H	6FH	28H	77H	7EH	7FH	

Key numbers 40H through 7FH for the standard keyboard are all cleared (undefined).

(32) READSW

Function: Reads switch status.

Entry address: WBOOT + 6FH or ØEB72H

Entry parameter: C = Selection number
= Ø2H: Read DIP switch settings.
= Ø4H: Read power switch status.

Return parameter: A = DIP switch settings or power switch status.

Explanation:

When C = Ø2H
The DIP switch settings are read into the A register in the following format:

DIP sw	8	7	6	5	4	3	2	1
A reg.	7	6	5	4	3	2	1	0

A Ø in an A register bit corresponds to the OFF state and a 1 to the ON state. It is set to 1 if the corresponding switch is ON.

When C = Ø4H
A = ØØH: Power switch is off.
= Ø1H: Power switch is on.

When C = other than Ø2H and Ø4H
READSW does nothing.

See also: Section 7.2, "DIP Switches"

```
                        ;       ************************************************
                        ;               READ DIP SWICH & POWER SWITCH
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is reading switch
                        ;               status and displaying it.
                        ;       <> assemble condition <>
                        ;
                                .Z80
                        ;
                        ;       <> loading address <>
                        ;
                                .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
                        ;       BIOS entry
                        ;
EB03                    WBOOT           EQU     0EB03H          ; Warm Boot entry
EB0C                    CONOUT          EQU     WBOOT   +09H    ; Console out entry
EB72                    READSW          EQU     WBOOT   +6FH    ; Read switch entry
                        ;
                        ;
0009                    TAB             EQU     009H    ; Tab code
000A                    LF              EQU     00AH    ; Line feed
000C                    CLS             EQU     00CH    ; Clear screen
000D                    CR              EQU     00DH    ; Carriage return
001B                    ESC             EQU     01BH    ; Escape
00E2                    ON              EQU     0E2H    ; ON code
00E3                    OFF             EQU     0E3H    ; OFF code
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
0100                    MAIN:
0100    31 1000                 LD      SP,1000H        ; Set stack pointer.
                        ;
0103    CD 011C                 CALL    SETCHAR         ; Set user-defined char.
                        ;
0106    0E 02                   LD      C,02H           ; Read dip switch.
0108    CD EB72                 CALL    READSW          ;
010B    CD 0144                 CALL    DSET            ; Set dip switch data.
                        ;
010E    0E 04                   LD      C,04H           ; Read power switch.
0110    CD EB72                 CALL    READSW          ;
0113    CD 0151                 CALL    PSET            ; Set power switch ata.
                        ;
0116    CD 0163                 CALL    DSPMSG          ; Display switch status.
                        ;
0119    C3 EB03                 JP      WBOOT           ; End
                        ;
                        ;       ************************************************
                        ;               SET USER DEFINE CHARACTER
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Set user defined character.
                        ;               E0H & E1H is used.
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;          return parameter <>
                        ;               NON
                        ;          preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
011C                    SETCHAR:
011C    21 012D                 LD      HL,CHARDATA     ; Data top address.
011F    46                      LD      B,(HL)          ; Get data counter.
0120    23                      INC     HL              ;
                        ;
0121                    SET10:
0121    4E                      LD      C,(HL)          ; Get conout data.
0122    E5                      PUSH    HL              ; Save registers.
0123    C5                      PUSH    BC              ;
0124    CD EB0C                 CALL    CONOUT          ; Console out.
0127    C1                      POP     BC              ; Restore registers.
0128    E1                      POP     HL              ;
0129    23                      INC     HL              ; Pointer update
012A    10 F5                   DJNZ    SET10           ; Loop
012C    C9                      RET                     ;
                        ;
012D                    CHARDATA:
012D    16                      DB      22              ; Data number.
012E    1B E0 E2                DB      ESC,0E0H,0E2H   ; 0E2H char data.
0131    3F 3F 3F 3F             DB      3FH,3FH,3FH,3FH
0135    21 21 21 3F             DB      21H,21H,21H,3FH
                        ;
0139    1B E0 E3                DB      ESC,0E0H,0E3H   ; 0E3H char data.
013C    3F 21 21 21             DB      3FH,21H,21H,21H
0140    3F 3F 3F 3F             DB      3FH,3FH,3FH,3FH
```

```
                        ;       ************************************************
                        ;               SET DIP SWITCH DATA
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Set dip switch data to message area.
                        ;
                        ;       <> entry parameter <>
                        ;               A  : Dip switch data.
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0144                    DSET:
0144    21 0190                 LD      HL,DIPSW        ; Data setting addr.
0147    06 08                   LD      B,8             ; Loop counter.
                        ;
0149                    DSET10:
0149    07                      RLCA                    ; Set switch status to CY.
014A    CD 0159                 CALL    SETONOFF        ; Set data.
014D    23                      INC     HL              ; Next setting address.
014E    10 F9                   DJNZ    DSET10          ; Loop.
                        ;
0150    C9                      RET
                        ;
                        ;       ************************************************
                        ;               SET POWER SWITCH DATA
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Set power switch data to message area.
                        ;
                        ;       <> entry parameter <>
                        ;               A  : Power switch data.
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION ;
                        ;
0151                    PSET:
0151    21 01B4                 LD      HL,POWSW        ; Data setting address.
0154    0F                      RRCA                    ; Switch data --> CY
0155    CD 0159                 CALL    SETONOFF        ; Set data.
0158    C9                      RET                     ;
                        ;
                        ;       ************************************************
                        ;               SELECT BDOS ERROR RECOVERY
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Select BDOS error recovery type.
                        ;                1. Using SETERR and RSTERR
                        ;                2. Replacing BDOS error vector
                        ;
                        ;       <> entry parameter <>
                        ;               CY : ON/OFF information.
                        ;                  =1 -- ON
                        ;                  =0 -- OFF
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               BC,DE,HL
                        ;
                        ;       CAUTION :
                        ;
0159                    SETONOFF:
0159    C5                      PUSH    BC              ; Save register.
015A    06 E2                   LD      B,ON            ; ON code --> B
015C    38 02                   JR      C,MSET10        ; ON.
015E    06 E3                   LD      B,OFF           ; Set OFF code.
0160                    MSET10:
0160    70                      LD      (HL),B          ; Set data.
0161    C1                      POP     BC              ; Restore register.
0162    C9                      RET                     ;
                        ;
                        ;       ************************************************
                        ;               DISPLAY SWITCH MESSAGE
                        ;       ************************************************
                        ;
                        ;       NOTE
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0163                    DSPMSG:
0163    21 0172                 LD      HL,MSG          ; Message data top addr.
0166                    DSP10:
0166    7E                      LD      A,(HL)          ; Get data.
0167    B7                      OR      A               ; End of data?
0168    C8                      RET     Z               ; Yes.
                        ;
0169    4F                      LD      C,A             ; Set parameter.
016A    E5                      PUSH    HL              ; Save register.
016B    CD EB0C                 CALL    CONOUT          ; Console out.
```

```
016E   E1                      POP     HL              ; Restore register.
016F   23                      INC     HL              ; Pointer update
0170   18 F4                   JR      DSP10           ; Loop.
                       ;
                       ;       Message and work area
                       ;
0172                   MSG:
0172   0C                      DB      CLS
0173   09 09 4F 4E             DB      TAB,TAB,'ON 87654321',CR,LF
0177   20 38 37 36
017B   35 34 33 32
017F   31 0D 0A
0182   44 49 50 20             DB      'DIP SWITCH',TAB,'
0186   53 57 49 54
018A   43 48 09 20
018E   20 20
0190                   DIPSW:  DS      8
0198   0D 0A                   DB      CR,LF
019A   09 09 4F 46             DB      TAB,TAB,'OFF',CR,LF
019E   46 0D 0A
                       ;
01A1   09 09 4F 4E             DB      TAB,TAB,'ON',CR,LF
01A5   0D 0A
01A7   50 4F 57 45             DB      'POWER SWITCH',TAB
01AB   52 20 53 57
01AF   49 54 43 48
01B3   09
01B4                   POWSW:  DS      1
01B5   0D 0A                   DB      CR,LF
01B7   09 09 4F 46             DB      TAB,TAB,'OFF',CR,LF
01BB   46 0D 0A
01BE   00                      DB      00H
                       ;
                               END
```

(33) RDVRAM

Function: Reads data from the virtual screen.

Entry address: WBOOT + 75H or 0EB78H

Entry parameter: B = Starting column number in which the read is to begin
C = Starting row number in which the read is to begin
DE = Number of characters to be read
HL = Address of the area for storing the read data (the area must not be smaller than the byte count specified in DE)

Return parameter: A = Return information
= 00H: Normal termination
= 01H: Read error
= 0FFH: Parameter error

Explanation:

The range of values allowed for the B and C registers varies with the size of the virtual screen.

1 <= B <= Number of columns in virtual screen (40 or 80)
1 <= C <= Number of lines in virtual screen (8 - 25)

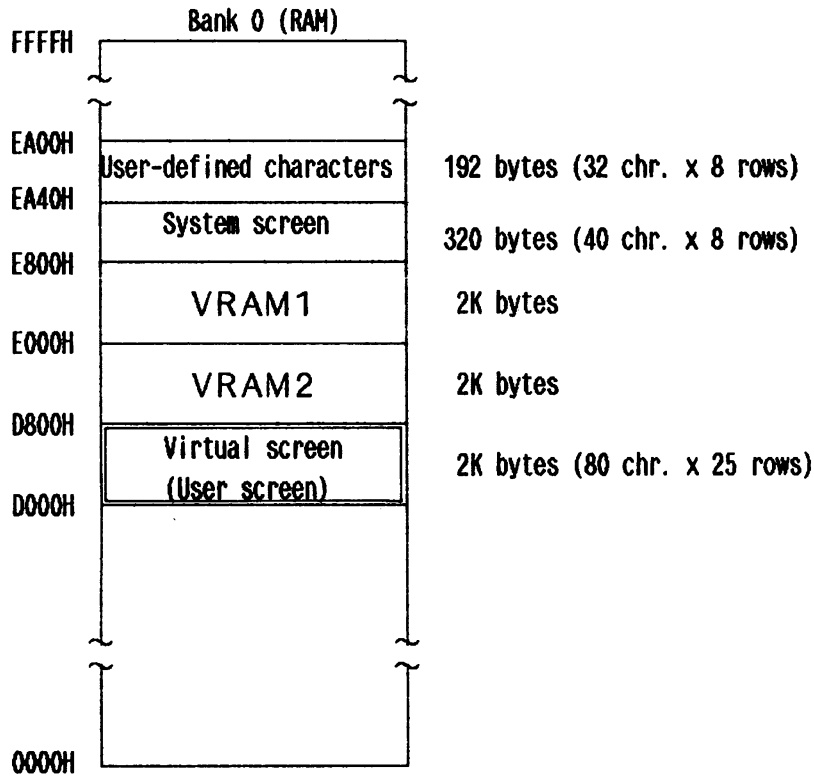
When the number specified in DE is too large and display extends beyond the screen, 20H (space) codes are returned as extra characters until the number of the returned characters matches the value specified in DE. If this condition occurs, the A register is loaded with a return code of 01H.

RDVRAM returns the read data in ASCII codes.

See also: Section 3.6, "LCD Display"

Reference:

The virtual screen (user screen) is located in memory as shown below:



Note:

In the system screen mode, RDVRAM reads data from the system screen.

(34) MCMTX

Function: Performs a microcassette function.

Entry address: WBOOT + 78H or ØEB7BH

Entry parameter:
MCMTX executes one of the 23 microcassette functions that is specified by the B register. The microcassette functions are listed below.

B reg.	Function	Description
00H	MIRDST	Read tape status
01H	MIRDCT	Read tape counter
02H	MISTCT	Set tape counter
03H	MISTOP	Stop motor
04H	MIPLAY	Start motor in PLAY mode
05H	MIREC	Start motor in RECORD mode
06H	MIFF	Start motor in FAST FEED mode
07H	MISREW	Start motor in SLOW REWIND mode
08H	MIREW	Start motor in REWIND mode
09H	MIFFTE	Fast feed to tape end
0AH	MIRWTT	Rewind to tape top
0BH	MIHDON	Turn head on
0CH	MIHDOF	Turn head off
0DH	MISKTP	Seek tape
0EH	MISTMP	Set tape move protect counter
0FH	MIRSMP	Reset tape move protect counter
10H	MIRDBL	Read one block
11H	MIWTBL	Write one block
12H		Undefined
13H		Undefined
14H	MIGTWD	Get write protection pin status
15H	MILEDON	Turn LED on
16H	MILEDOF	Turn LED off
17H	MISDAT	Save data area
18H	MILDAT	Load data area

See Section 3.7, "MTOS/MIOS Operations" for detailed information about the individual functions.

(35) POWEROFF

Function: Turns off PINE main power.

Entry address: WBOOT + 7BH or ØEB7EH

Entry parameter: C = Power off mode
= ØØH: Turn off power in continue mode.
= ØlH: Turn off power in restart mode.

Return parameter: None.

Explanation:
POWEROFF saves the current system status and turns off PINE main power. Once this routine is executed, only the 75Ø8 slave CPU and RAM are supplied with power. The 75Ø8 only updates the clock every 1Ø seconds and is mostly in the sleep state.

Notes:
Reset the continue mode with BIOS CONTINUE when turning off system power in the restart mode. Otherwise, the continue mode specification will be given priority and power may not be turned off in the restart mode.

See Section 2.5, "Power-off" for power-off processing.

See also: Section 2.5, "Power-off."

(36) USERBIOS

Function: Provides an entry point to USERBIOS.

Entry address: WBOOT + 7EH or ØEB8lH

Entry parameter: User-supplied

Return parameter: User-supplied

Explanation:
USERBIOS provides an entry point through which the user can call his own BIOS routine. The user BIOS routine itself is located in the user BIOS area.

The starting address of the user-supplied BIOS routine must be set in this entry address.

The default of USERBIOS is the address of a subroutine which contains nothing but a RET instruction.

Note:
USERBIOS is set to the default value by a reset or system initialize.

See also: Section 4.1, "User BIOS"

(37) AUTOST

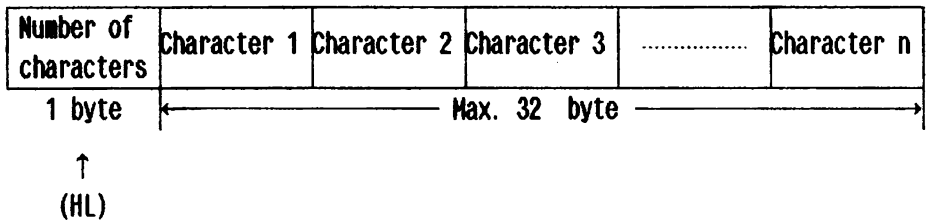
Function: Defines or cancels an auto start string.

Entry address: WBOOT + 81H or ØEB84H

Entry parameter: C = Function
= ØØH: Cancel an auto start string.
= Ø1H: Define an auto start string.
HL = Starting address of the string to be defined (Valid only when C = Ø1H.)

Return parameter: None.

Explanation:
The string format is shown below:



"Number of characters" indicates the length of the specified string and must be in the range from ØØH to 2ØH. ØØH specifies that the auto start string is to be cancelled.

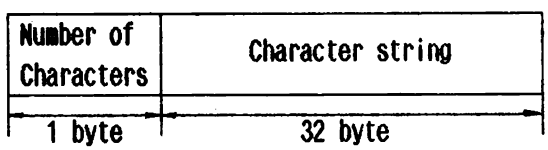
See Section 2.7 for the auto start function.

Note:
When the specified string is shorter than 32 bytes, 2ØH (space) codes are appended to the end of the string to fill the 32-byte field. When a length longer than 32 bytes is specified, the extra bytes are ignored.

See also: Section 2.7, "Auto Start Function"

Reference:
The system uses the following areas for defining an auto start string:

AUTOSTRT (ØF3BDH) 34 bytes
- Auto start string area



"Number of characters" indicates the length of the specified character string. A ØØH indicates that no auto start string is defined. The 34th byte is not used.

```
                        ;***************************************************
                        ;       SET AUTO START STRING
                        ;***************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program sets auto start string.
                        ;               If '^' + character are inputed, they are
                        ;               translated into control character.
                        ;
                        ;       <> assemble condition <>
                        ;
                        ;       .Z80
                        ;
                        ;       <> loading address <>
                        ;
                        ;       .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
                        ;       BDOS entry
                        ;
0005                    BDOS            EQU     00005H  ; BDOS entry address.
                        ;
0009                    STRING_OUT      EQU     09H     ; BDOS function
000A                    STRING_IN       EQU     0AH     ;
                        ;
                        ;       BIOS entry
                        ;
EB03                    WBOOT           EQU     0EB03H          ; Warm Boot entry
EB84                    AUTOST          EQU     WBOOT   +81H    ; Auto start entry
                        ;
                        ;       System area
                        ;
                        ;***************************************************
                        ;       MAIN PROGRAM
                        ;***************************************************
                        ;
                        ;       NOTE :
                        ;               This program sets auto start string,
                        ;               whitch is inputed from keyboard.
                        ;
0100                    MAIN:
0100    31 1000                 LD      SP,1000H        ; Set stack pointer.
                        ;
0103    0E 09                   LD      C,STRING_OUT    ; Console out string.
0105    11 0153                 LD      DE,MSG01        ;  Message address.
0108    CD 0005                 CALL    BDOS            ;
                        ;
010B    0E 0A                   LD      C,STRING_IN     ; Input into console buffer.
010D    11 017D                 LD      DE,IN_BUFF      ;  Buffer address.
0110    CD 0005                 CALL    BDOS            ;
                        ;
0113    CD 0121                 CALL    CHK_CTRL        ; Check control code.
                        ;
0116    21 017E                 LD      HL,IN_BUFF+1    ; Set auto sart string.
0119    0E 01                   LD      C,01H           ;  Set.
011B    CD EB84                 CALL    AUTOST          ;
                        ;
011E    C3 EB03                 JP      WBOOT           ; Program end.
                        ;
                        ;***************************************************
                        ;       CHANGE ^+CHARACTER TO CONTROL CODE
                        ;***************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0121                    CHK_CTRL:
0121    21 017E                 LD      HL,IN_BUFF+1    ; Inputed character no. check.
0124    7E                      LD      A,(HL)          ;  Get inputed character no.
0125    B7                      OR      A               ;  No character inputed?
0126    C8                      RET     Z               ;  Yes.
                        ;
0127    4F                      LD      C,A             ; Char. No. --> C
0128    23                      INC     HL              ; Char. data top addr. --> HL
0129    54                      LD      D,H             ; Char. data top addr. --> DE
012A    5D                      LD      E,L             ;
                        ;
012B                    CHK10:
012B    1A                      LD      A,(DE)          ; Get character.
012C    FE 5E                   CP      '^'             ;  '^' code?
012E    28 07                   JR      Z,CHK20         ; Yes.
                        ;
0130    77                      LD      (HL),A          ; Set data.
0131    13                      INC     DE              ; Get pointer update.
0132    23                      INC     HL              ; Put pointer update.
0133    0D                      DEC     C               ; Counter decrement.
0134    C8                      RET     Z               ; End of char.
0135    18 F4                   JR      CHK10           ; Loop.
                        ;
0137                    CHK20:
0137    13                      INC     DE              ; Get pointer increment.
0138    0D                      DEC     C               ; Counter check.
0139    C8                      RET     Z               ; No char. exists.
                        ;
013A    1A                      LD      A,(DE)          ; Get character.
013B    D6 40                   SUB     40H             ; 40H to 7FH?
013D    38 EC                   JR      C,CHK10         ; No.
013F    FE 40                   CP      40H             ;
```


ClibPDF - www.fastio.com

```
0141    30 E8                       JR      NC,CHK10        ; No.

0143    E6 DF                       AND     11011111B       ; 00H to 1FH
0145    77                          LD      (HL),A          ; Set new data.
0146    13                          INC     DE              ; Get pointer update.
0147    23                          INC     HL              ; Put pointer update.
0148    3A 017E                     LD      A,(IN_BUFF+1)   ; Character No. decrement.
014B    3D                          DEC     A               ;
014C    32 017E                     LD      (IN_BUFF+1),A   ;
014F    0D                          DEC     C               ; Character remain?
0150    C8                          RET     Z               ; No.
0151    18 D8                       JR      CHK10           ; Loop until end of char.
                            ;
                            ;       Message and work area.
                            ;
0153                        MSG01:
0153    49 4E 50 55                 DB      'INPUT AUTO START STRING (Max 32 char.)'
0157    54 20 41 55
015B    54 4F 20 53
015F    54 41 52 54
0163    20 53 54 52
0167    49 4E 47 20
016B    28 4D 61 78
016F    20 33 32 20
0173    63 68 61 72
0177    2E 29
0179    0D 0A 09 24                 DB      0DH,0AH,09H,'$'
                            ;
                            ;
017D                        IN_BUFF:
017D    20                          DB      32              ; Max input character No.
017E                                DS      1               ;  Inputed char. No. area.
017F                                DS      32              ;  Inputed data area.
                            ;
                                    END
```

(38) RESIDENT

Function: Turns on or off the resident function.

Entry address: WBOOT + 84H or ØEB87H

Entry parameter: C = Controls the resident function.
= ØØH: Disable resident.
= Ø1H: Enable resident.

Return parameter: None.

Explanation:
RESIDENT enables or disables the resident function. For details of this function, see Section 4.5.

Reference:
The system uses the following area for resident control:

RESEXQ (ØEF28H) 1 byte
- Resident execution flag
= ØØH: Resident function not specified.
= Nonzero: Resident function specified.

See also: Section 4.5, "Resident Processing"

(39) CONTINUE

Function: Sets or resets the continue mode.

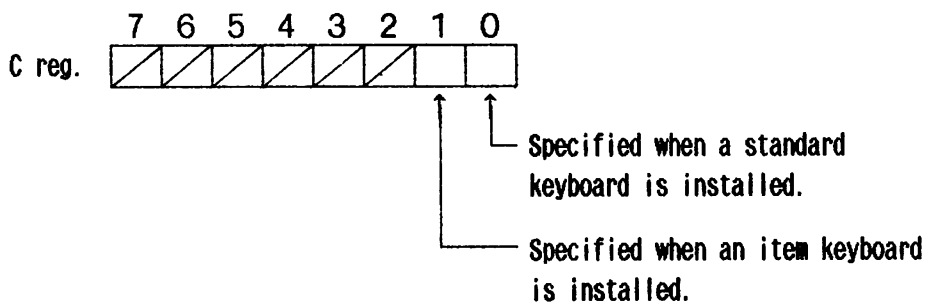
Entry address: WBOOT + 87H or ØEB8AH

Entry parameter: C = Controls the continue mode.

Return parameter: None.

Explanation:
CONTINUE sets or resets the continue flag.

The entry parameter has the following format:



A 1 in bit Ø or 1 sets the continue mode and a Ø resets the continue mode.

The continue mode may be set for the standard and item keyboards independently.

Reference:

The system uses the following areas to control the continue mode:

FRCECNTN (ØF311H) 1 byte
- Continue flag for the standard keyboard
  - = ØØH: Restart mode (default)
  - = Nonzero: Continue mode

IFRCECNT (ØF312H) 1 byte
- Continue flag for the item keyboard
  - = ØØH: Restart mode
  - = Nonzero: Continue mode (default)

3.5 Keyboard

3.5.1 General

The PINE supports the standard and the item keyboards. The PINE keyboards are controlled by the slave CPU (7508), which communicates with the main CPU via an serial interface.

The 7508 checks the state of all keys approximately every 30 ms. If a key entry has been made, the 7508 loads the key position code into its own key buffer and sends an interrupt signal to the main CPU.

The main CPU, on receipt of the interrupt, fetches the position code via the interrupt processing routine and loads it into its key buffer.

The data in the key buffer is taken and processed one byte at a time by the BIOS CONIN or CONST routine.

This section describes the difference between the standard and item keyboards and the method of controlling key entries.

3.5.2 Keyboard Functions

3.5.2.1 Keys

(1) Number of keys
    Standard keyboard: 72
    Item keyboard: 58
(2) Number of switch keys
    Standard keyboard: 6 (CTRL, SHIFT x 2, CAPS, GRPH, KANA or NUM)
    Item keyboard: 4 (SHIFT, STOP, INIT, CTRL)

3.5.2.2 Key scanning

The 7508 scans the keyboard every 30 ms. If the depression of a key other than switch keys is sensed, the 7508 generates the position of the key in the matrix as a make code. If a switch key is pressed, it outputs a make code when the key is pressed and outputs a break code when it is released. If two or more keys are pressed consecutively, the 7508 outputs the corresponding codes accordingly. If two or more keys are pressed simultaneously, the 7508 outputs the code of whichever key is scanned first. If the keys are switch keys, it outputs the codes of all switch keys that are pressed.

3.5.2.3 Auto repeat function

All standard keyboard keys other than the switch keys, the STOP, ESC, PAUSE, and HELP keys, and function keys PF1 - PF5 are repeatable. The auto repeat function is disabled by default when an item keyboard is installed. The auto repeat on/off state, repeat start time, and repeat interval time can be changed using the BIOS CONOUT routine (see the paragraphs on the CONOUT BIOS function.

3.5.2.4 Inhibiting keyboard interrupts

The PINE can be disabled or enabled for keyboard interrupts using commands. It may also be enabled only for STOP key interrupts. The steps below show how the 7508 processes keyboard interrupts when only STOP key interrupts are enabled.

(1) The 7508 loads the buffer with the codes of the keys that are pressed while interrupts are disabled.
(2) When the buffer becomes full, the 7508 does not buffer subsequent key codes. However, it places the code associated with the STOP key at the end of the buffer if the key is pressed.
(3) The 7508 generates an interrupt when the STOP key is pressed. The data in the 7508 buffer is then read by the main CPU starting at the beginning of the buffer.
To disable/enable key interrupts use is made of BIOS MASKI.

When an item keyboard is installed, however, the user cannot enable only STOP key interrupts. Attempting to do so disables all keyboard interrupts.

3.5.2.5 LEDs

Both the standard keyboard and the item keyboard have three LEDs.

On the standard keyboard, the LEDs are used to indicate the CAPS and NUM key status. On the item keyboard, the LEDs are used to indicate the SHIFT key status.

The LEDs can be controlled by the user using the BIOS CONOUT routine. The user, however, should not use the LEDs that are used by the system.

3.5.3 Standard Keyboard

3.5.3.1 Outline

(1) Number of keys: 72
(2) Number of switch keys: 6 (CTRL, SHIFT x 2, CAPS, GRPH, or NUM)
(3) Number of LEDs: 3

3.5.3.2 Key Functions

(1) Auto repeat keys
All keys except the following 15 keys are repeatable:
STOP, ESC, PAUSE, HELP, PF1 - PF5, CTRL, SHIFT (left and right), CAPS, GRPH, NUM

(2) Auto repeat control
The auto repeat on/off state, repeat start time, and repeat interval time can be changed using the BIOS CONOUT routine.

3.5.3.3 Special Keys

The PINE keyboard is furnished with 14 special keys: STOP, ESC, PAUSE, HELP, PF1 - PF5, INS, ←, →, ↑, and ↓. These keys function in different ways depending on the keyboard mode (normal, shift, or control).

Key	Mode	Code	Function
STOP	Normal	Ø3H	Clear key buffer.
	Shift	Ø3H	Clear key buffer.
	Control	Ø3H	Clear key buffer. Terminate I/O immediately.
ESC	Normal	1BH	None.
	Shift	1BH	None.
	Control	None	Call subroutine.
PAUSE	Normal	13H	None.
	Shift	13H	None.
	Control	None	Call subroutine.
HELP	Normal	ØØH	None.
	Shift	ØØH	None.
	Control	None	Call subroutine.
PF1	Normal	-	Function 1
	Shift	-	Function 6
	Control	None	Call subroutine.
PF2	Normal	-	Function 2
	Shift	-	Function 7
	Control	None	Call subroutine.
PF3	Normal	-	Function 3
	Shift	-	Function 8
	Control	None	Call subroutine.
PF4	Normal	-	Function 4
	Shift	-	Function 9
	Control	None	Call subroutine.
PF5	Normal	-	Function 5
	Shift	-	Function 1Ø
	Control	None	Call subroutine.
INS	Normal	Ø8H	None.
	Shift	None	Switch cursor tracking mode.
	Control	None	Display the screen containing the cursor.
→	Normal	1CH	None.
	Shift	None	Scroll the screen 2Ø columns to the right.
	Control	None	Scroll the screen 4Ø columns to the right.
←	Normal	1DH	None.
	Shift	None	Scroll the screen 2Ø columns to the left.
	Control	None	Scroll the screen 4Ø columns to the left.
↑	Normal	1EH	None.
	Shift	None	Scroll the screen one line up.
	Control	None	Scroll the screen eight lines up.
↓	Normal	1FH	None.
	Shift	None	Scroll the screen one line down.
	Control	None	Scroll the screen eight lines down.

The functions here refer to those functions which are automatically started by the system. They are fully described on the following pages. The above codes can be read by the CONIN routine.

3.5.3.4 STOP and CTRL/STOP

STOP and CTRL/STOP keys are processed by the same interrupt processing routine and perform the following functions:

STOP: Clears the key buffer and returns ^C (Ø3H).

CTRL/STOP: Clears the key buffer and returns ^C (Ø3H). Also terminates the current I/O operation.

The following flags are set when the STOP or CTRL/STOP key is pressed:

BRKFLG (ØFØ19H) 1 byte
- Indicates that the STOP key has been pressed.
    = ØØH: STOP key not pressed.
    = ØFFH: STOP key pressed.

This flag is reset to ØØH if the BIOS CONIN or CONST routine is called when the key buffer is empty.

CSTOPFLG (ØFØ1AH) 1 byte
- Indicates that the CTRL/STOP has been pressed.
    = ØØH: CTRL/STOP not pressed.
    = Ø1H: CTRL/STOP pressed.

This flag is reset to ØØH if the BIOS CONIN or CONST routine is called when the key buffer is empty.

CSTOPMCT (ØFØ1BH) 1 byte
- Is the flag for terminating microcassette processing.
    = ØØH: CTRL/STOP key not pressed.
    = Ø1H: CTRL/STOP key pressed.

This flag is reset to ØØH by the PSTBIOS routine.

CSTOPPRN (ØF3Ø9H) 1 byte
- Is the flag for terminating cartridge printer processing.
    = ØØH: CTRL/STOP not pressed.
    = Ø1H: CTRL/STOP pressed.

This flag is reset to ØØH by the PSTBIOS routine.

During an I/O operation that takes a long processing time or that does not terminate until some conditions are met, the PINE checks CSTOPFLG and terminates the operation if the flag is set. The PINE references CSTOPMCT or CSTOPPRN when handling I/O operations on the microcassette or cartridge printer.

3.5.3.5 Programmable Function Keys

The keys PF1 to PF5 at the top of the keyboard are programmable. They are used with or without the SHIFT key and are numbered PF6 (SHIFT/PF1) to PF1Ø (SHIFT/PF5) when used with the SHIFT key. Any string of up to 15 characters can be assigned to each of these keys.

PFKs can be defined in the system by:
- Using the PUTPFK (WBOOT + 6CH) BIOS function.
- Rewriting the system PF key table.
- Rewriting the pointer to the system PF key table.

A detailed description of the definition methods follows.

(1) Using the PUTPFK
See the description about BIOS PUTPFK.

(2) Rewriting the PF key table
The PINE uses two types of PF key tables. One is named PFKTAB and the other WPFKTBL. WPFKTBL is referenced by the system whenever a PF key is pressed. PFKTAB contains the data that is used by WBOOT to initialize WPFKTBL when resident is not specified.

The relationship between these tables is shown in Figure 3.5.1.

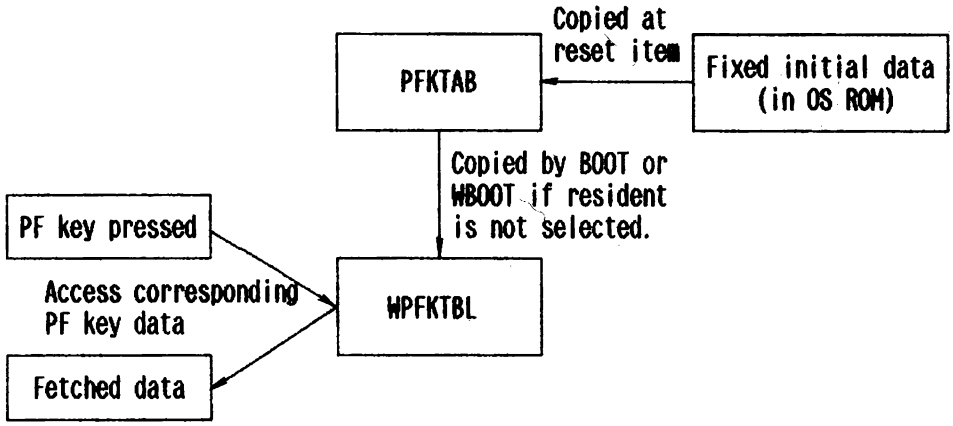


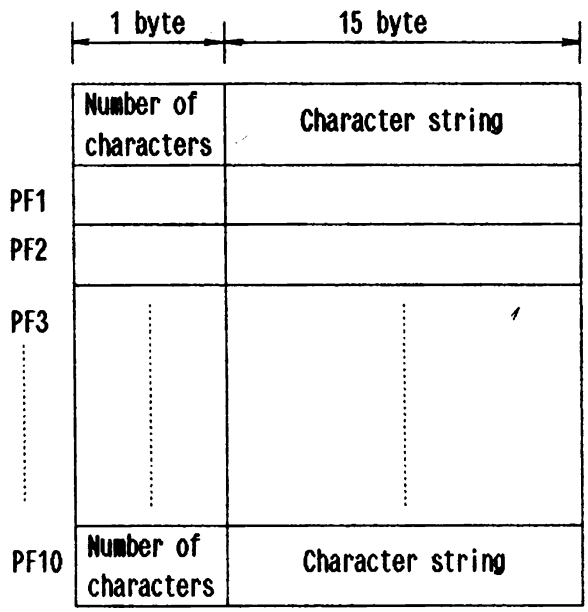
Fig. 3.5.1 Conceptual Diagram of the PF Key Tables

Change the data in WPFKTBL to make the PF key definitions valid only during execution of the application program. Change the data in PFKTAB to make the definition changes permanent (valid after warm boots).

(3) Rewriting the pointer to the PF key table
In the PINE OS, the starting address of the PF key table is stored in YPFKSTR (in Figure 3.5.1 the system references YPFKSTR to find the PF key table address each time it accesses PF key data). PF key definitions can be modified by changing the address in YPFKSTR.

The procedure for changing the pointer to the PF key table is given below.

1) Create a PF key table at location 8000H or higher in RAM. The table format is shown below.



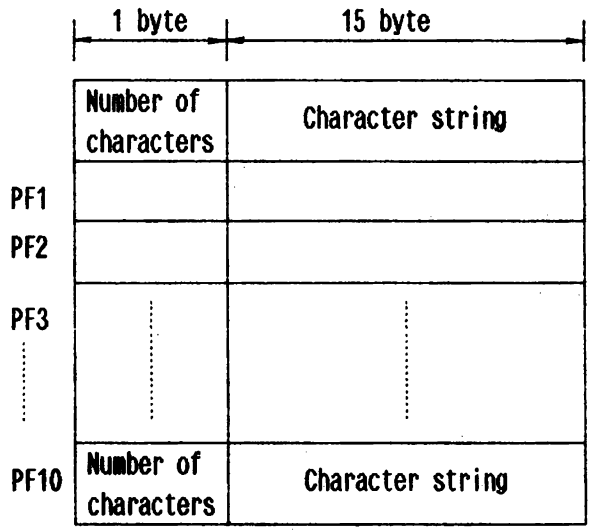
2) Load ØØH into the Number of characters field for unnecessary PF keys. No key code will be returned for these keys when they are pressed.

3) Load the table starting address into YPFKSTR.

This complete the table update process. YPFKSTR is initialized during a warm boot.

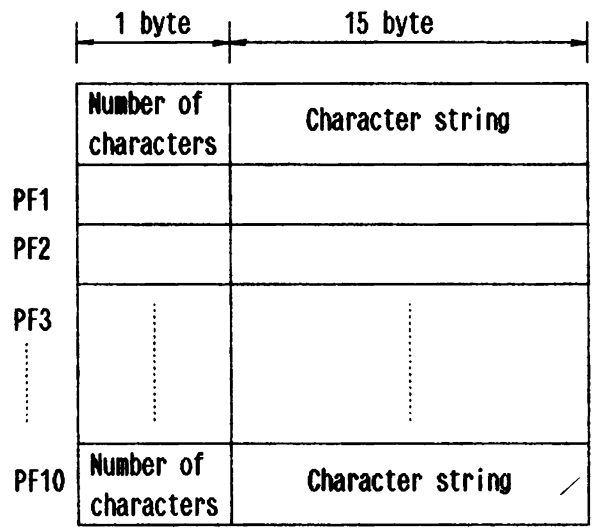
YPFKSTR (ØF12H) 2 bytes
- PF key table starting address
The initial value is WPFKTBL (ØF545H). (The table must be located at 8000H or higher location in RAM.)

PFKTAB (ØF02DH) 160 bytes
- PF key initial data area



This table is copied into WPFKTBL during a cold or warm boot when resident is not specified.

WPFKTBL (ØF545H) 160 bytes
- PF key table



3.5.3.6 Keys Calling a Subroutine

User-supplied service programs can be executed in the form of subroutine calls by pressing the specific keys.

When the ESC, PAUSE, HELP, or PF1 to PF5 is pressed with the CTRL key held down, the system calls the associated subroutine based on the call address stored in the subroutine table (YSUBRTN). The actual subroutine is executed when the BIOS CONIN or CONST is called.

Note the following points when modifying the table:

1) The modified call address must be 8000H or higher.
2) The called subroutine must reserve its own stack area when it is to use a large amount of stack area. The original stack must be restored on return.
3) In general, system calls such as BDOS and BIOS cannot be used in the subroutine. Call directly the corresponding module in OS ROM if necessary.
4) YSUBRTN is not initialized during a warm boot. Therefore, if the subroutine is to be used only in the application program, the table must be restored with the original values when the application program terminates. (Service program is not necessarily existed on the called address.)

YSUBRTN (ØFØCDH) 16 bytes
- Subroutine table

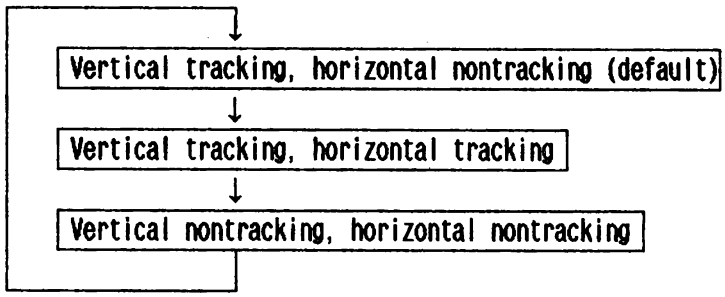
Key sequence	Address	Default function
CTRL/ESC	ØFØCDH	Does nothing but return.
CTRL/PAUSE	ØFØCFH	Does nothing but return.
CTRL/HELP	ØFØD1H	Performs system display processing.
CTRL/PF1	ØFØD3H	Does nothing but return.
CTRL/PF2	ØFØD5H	Does nothing but return.
CTRL/PF3	ØFØD7H	Does nothing but return.
CTRL/PF4	ØFØD9H	Does nothing but return.
CTRL/PF5	ØFØDBH	Performs screen dump processing.

3.5.3.7 Screen Control

The PINE uses the concept of virtual screen to implement CRT functions. Some keys are assigned special functions such as switching screen modes or defining the window to be displayed.

(1) SHIFT/INS (Switches the track mode.)

The PINE provides three scroll modes for the window screen. The scroll mode changes cyclically as the SHIFT/INS is pressed.



These modes control the behavior of the screen when the cursor goes out of the window. See Section 3.6, "LCD Display" for more information.

(2) CTRL/INS (Displays the cursor screen.)

This key sequence displays the portion of the screen in which the cursor is presently positioned regardless of the current scroll mode. It does nothing if the cursor is in the current screen.

(3) SHIFT/arrow keys, CTRL/arrow keys (Scrolls the screen.)

The SHIFT or CTRL key is pressed in combination with one of the arrow keys to display the portion of the virtual screen other than the window that is currently being displayed.

SHIFT/→	Scrolls the window 20 columns to the right. Does nothing when the virtual screen is 40 columns wide or the right screen is already displayed.
SHIFT/←	Scrolls the window 20 columns to the left. Does nothing when the virtual screen is 40 columns wide or the left screen is already displayed.
SHIFT/↑	Scrolls the window one line up. Does nothing when the window is already at the top of the virtual screen.
SHIFT/↓	Scrolls the window one line down. Does nothing when the window is already at the bottom of the virtual screen.
CTRL/→	Displays the right screen when the virtual screen is 80 columns wide. Does nothing if the right screen is already displayed.
CTRL/←	Displays the left screen when the virtual screen is 80 columns wide. Does nothing if the left screen is already displayed.
CTRL/↑	Scrolls up the window one screenful of lines. Display starts at the top of the virtual screen when it moves up beyond the top. wide or the right screen is already displayed.
CTRL/↓	Scrolls down the window one screenful of lines. Display starts at the bottom of the virtual screen when it moves down the bottom.

3.5.3.8 Changing Codes

The key codes for arrow keys, SHIFT/arrow keys, and CTRL/arrow keys can be changed using BIOS CONOUT or by rewriting the associated areas. See Section 3.6 for BIOS CONOUT.

The user can assign any code to each of these keys. When 80H or 0F8H to 0FFH is entered, however, the OS takes special actions and does not return the code.

Code	Function
80H	Does nothing. The key entry is regarded as invalid.
0F6H	Changes the screen scroll mode.
0F7H	Displays the screen containing the cursor.
0F8H	Scrolls the window horizontally 20 columns to the right.
0F9H	Scrolls the window horizontally 20 columns to the left.
0FAH	Scrolls the window one line up.
0FBH	Scrolls the window one line down.
0FCH	Displays the right screen of the 80-column virtual screen.
0FDH	Displays the left screen of the 80-column virtual screen.
0FEH	Scrolls up the window a screenful of lines.
0FFH	Scrolls down the window a screenful of lines.

The areas to be rewritten when changing key codes are listed below. The initial values in the list are the copy of the 14 bytes starting at DFLTRT (0F01FH). They are copied during a cold or warm boot if resident is not specified.

The changed key codes can be made permanent over a reset or warm boot by changing the contents of the 14-byte area starting at DFLTRT.

CHGRIGT (0F328H) 1 byte
- Code for the ← key.
Initial value is 1CH.

CHGLEFT (0F329H) 1 byte
- Code for the → key.
Initial value is 1DH.

CHGUP (0F32AH) 1 byte
- Code for the ↑ key.
Initial value is 1EH.

CHGDOWN (0F32BH) 1 byte
- Code for the ↓ key.
Initial value is 1FH.

CHGSRIGT (0F32CH) 1 byte
- Code for the SHIFT/→ key.
Initial value is 0F8H.

CHGSLEFT (0F32DH) 1 byte
- Code for the SHIFT/← key.
Initial value is 0F9H.

CHGSUP (0F32EH) 1 byte
- Code for the SHIFT/↑ key.
Initial value is 0FAH.

CHGSDOWN (0F32FH) 1 byte
- Code for the SHIFT/↓ key.
Initial value is 0FBH.

CHGSSCRN (ØF33ØH) 1 byte
- Code for the SHIFT/INS key.
Initial value is ØF6H.

CHGCRIGT (ØF331H) 1 byte
- Code for the CTRL/→ key.
Initial value is ØFCH.

CHGCLEFT (ØF332H) 1 byte
- Code for the CTRL/← key.
Initial value is ØFDH.

CHGCUP (ØF333H) 1 byte
- Code for the CTRL/↑ key.
Initial value is ØFEH.

CHGCDOWN (ØF334H) 1 byte
- Code for the CTRL/↓ key.
Initial value is ØFFH.

CHGCSCRN (ØF335H) 1 byte
- Code for the CTRL/INS key.
Initial value is ØF7H.

3.5.3.9 Key Interrupt Processing

The following table shows the actions taken by the keyboard interrupt processing routines, BIOS CONIN, and CONOUT in response to key entries.

STOP, CTRL/STOP
- Interrupt processing routine: Clears the key buffer and loads it with the position code.
- CONST: Returns control to the calling program with ØFFH in the A register.
- CONIN: Returns control to the calling program with Ø3H in the A register.

ESC, PAUSE, HELP, INS
- Interrupt processing routine: Loads the key buffer with the position code.
- CONST: Returns control to the calling program with ØFFH in the A register.
- CONIN: Returns control to the calling program with the key code in the A register.

Subroutine call
- Interrupt processing routine: Loads the key buffer with the position code.
- CONST: Performs the corresponding processing and checks for the presence of keyed data.
- CONIN: Performs the corresponding processing and waits for keyed data.

Code programmable keys (ØF6H - ØFFH, 8ØH)
- Interrupt processing routine: Loads the key buffer with the position code.
- CONST: Performs the corresponding processing and checks for the presence of keyed data.
- CONIN: Performs the corresponding processing and waits for keyed data.

Code programmable keys (other than above codes)
- Interrupt processing routine: Loads the key buffer with the position code.
- CONST: Returns control to the calling program with ØFFH in the A register.
- CONIN: Returns control to the calling program with the key code in the A register.

PF keys
- Interrupt processing routine: Loads the key buffer with the position code.
- CONST: Returns control to the calling program with ØFFH in the A register.
- CONIN: Returns control to the calling program with the first one character in the A register.

Switch keys
- Interrupt processing routine: Loads the key buffer with a make or a break code.
- CONST: Determines the keyboard state and checks for keyed data again.
- CONIN: Determines the keyboard state and waits for keyed data again.

Other keys
- Interrupt processing routine: Loads the key buffer with the position code.
- CONST: Returns control to the calling program with ØFFH in the A register.
- CONIN: Returns control to the calling program with the key code in the A register.

3.5.3.1Ø Key Buffers

The PINE has two key buffers: one in the main CPU (Z-8Ø) and the other in the 75Ø8.

(1) 75Ø8 key buffer

The 75Ø8 key buffer may be seven characters (default) or one character wide. The size can be specified by sending the following commands to the 75Ø8:

ØCH: Reserves a 7-character area as the key code buffer.
1CH: Reserves a 1-character area as the key code buffer.

These commands are sent to the 75Ø8 by the WRT75Ø8 routine cataloged the system jump table. See Section 4.2, "Jump Tables" for details.

(2) Main key buffer

BIOS reserves a 33-byte area as the buffer to buffer key code of up to 32 characters. Figure 3.5.2 shows the structure of key buffer KBUF (ØF86ØH).

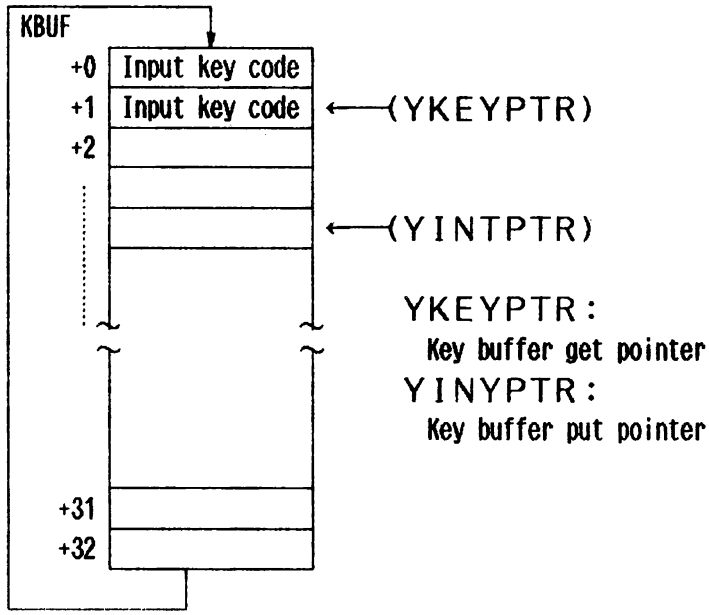


Fig. 3.5.2 Key buffer Structure

The key buffer is of the ring structure. Input or output to or from the key buffer is performed in the following sequence:
1) On receiving a keyboard interrupt, the keyboard interrupt processing routine loads the key code into the address pointed to by YINTPTR and increments the pointer by one. When the incremented pointer matches YKEYPTR, the routine discards the input key code.

2) The key entry processing routine determines that no key entry has been made if YKEYPTR and YINTPTR have the same value. If their values differ, the routine fetches the key code at the address pointed to by YKEYPTR and then increments the pointer by one.

KBUF (ØF86ØH) 33 bytes
- Key buffer for key entries
The key buffer is 33 bytes long, it can actually hold only 32 bytes of data at most.

YINTPTR (ØF ØØBH) 2 bytes
- Key buffer put pointer
The pointer to the buffer location into which the keyboard interrupt processing routine puts the received key code. The pointer is incremented by the keyboard interrupt processing routine.

YKEYPTR (ØF ØØDH) 2 bytes
- Key buffer get pointer
The pointer to the buffer location from which the input key code is gotten. YINTPTR = YKEYPTR indicates that no key code is present in the buffer.

3.5.3.11 Automatic Key Entry at Power-on Time

The PINE provides special functions to enter a predefined string automatically at power-on time: the auto start function and the wake function. They load an auto start string and a wake string, respectively, when power is switched on in the restart mode.

Both strings may be as long as 32 characters. Any defined string is treated in the same way as keyed-in data and used as parameters to be placed on the menu or CCP command line or interpreted as parameters after a transient command is activated. The wake string can be defined using the system display or BIOS TIMDAT. The auto start string can be defined using the system display or BIOS AUTOST function. See Section 2.7, "Auto Start Function" and Section 2.9, "Alarm/Wake" for details.

3.5.3.12 Keyboard States

The keyboard state is determined by the combination of the switch key (CTRL, SHIFT, or GRPH) that is pressed and the current keyboard shift mode (Normal, Caps, Num).

The switch keys are assigned precedence in the decreasing order of CTRL, GRPH, and SHIFT.

Table 3.5.3 lists the keyboard states.

Mode / Switch key	NORMAL	CAPS	NUM
———	Normal	Caps	Num
CTRL	Control		
SHIFT	Normal Shift	Caps Shift	Num Shift
GRPH	Graphic		
CTRL+SHIFT	Control Shift (Item mode) *		
CTRL+GRPH	Control		
SHIFT+GRPH	Graph Shift		
CTRL+SHIFT+GRPH	Control Shift (Item mode) *		

Table 3.5.3 Keyboard States

*: See 3.5.3.14 for item mode.

3.5.3.13 Mode Transition

The mode transition diagrams for the PINE keyboard are shown in Figures 3.5.4.

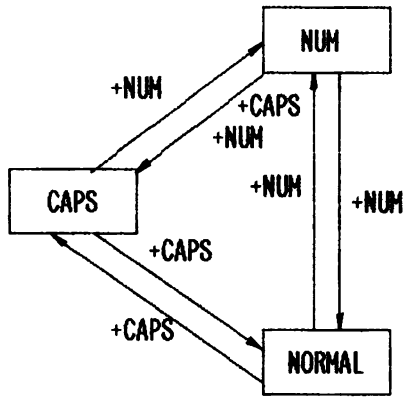
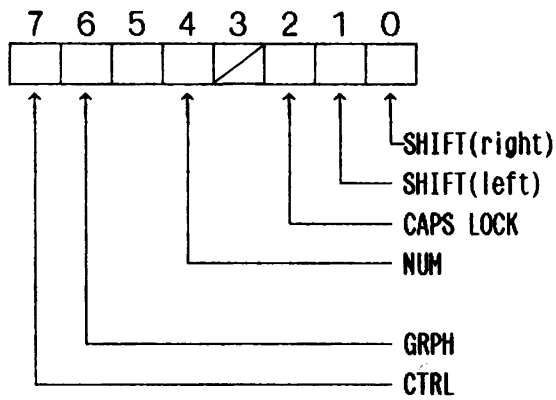


Fig. 3.5.4 Mode Transition (Export Version OS)

When power is turned on, the keyboard mode is first set to the mode that was established when power was turned off. The mode is set to Normal when a reset occurs.

The current keyboard state can be identified by checking the value of YSHFDT. YSHFDT contains the status of all switch keys. When referencing YSHFDT, it is necessary to set up the current keyboard state by running BIOS CONST in advance.

YSHFDT (ØFØØFH) 1 byte
- Indicates the current shift mode.



Bit n = 1 indicates that the corresponding mode is ON. Bit n = Ø indicates that the mode is OFF. This variable has a value of ØØH when the keyboard is in the Normal mode.

3.5.3.14 Item Function

The PINE OS supports the item function on the standard keyboard. The item function defines character strings of up to 15 characters as key codes (4ØH - 7FH) and treats them in the same way as ordinary keyed-in strings. See Subsection 3.5.4, "Item Keyboard" for details.

3.5.4 Item Keyboard

3.5.4.1 Outline

The PINE supports the item keyboard that allows the system to be used exclusively for specific applications.

The item keyboard consists of the item, numeric keypad, cursor movement key, and display sections. The keys in the item section can be redefined. Strings of up to 15 characters can be defined for a maximum of 62 items. This feature makes the item keyboard suitable for a wide variety of applications.

The user can easily replace the standard keyboard with the item keyboard. This means that the PINE can be used either for general or special purposes.

3.5.4.2 Key configuration

The keys on the item keyboard are grouped as follows:

Item section: 32 keys (including SHIFT)
Numeric keypad section: 19 keys
Cursor movement key section: 4 keys
System key section: 3 keys
LED section: 3 LEDs (one is used by the system)

The SHIFT key in the item section and the three keys in the system key section (STOP, INIT, and CTRL) are switch keys.

3.5.4.3 Keyboard functions

The PINE keyboard has the following functions:
(1) Auto repeat function
All keys except the SHIFT, STOP, INIT, and CTRL are repeatable. They are set to norepeat mode by default.

(2) Auto repeat control function
The user can turn on and off the auto repeat function and specify the repeat start time and repeat interval time through BIOS CONOUT.

3.5.4.4 Key specifications

(1) Keyboard layout

Figure 3.5.5 shows the layout of the item keyboard (furnished with an overlay sheet).

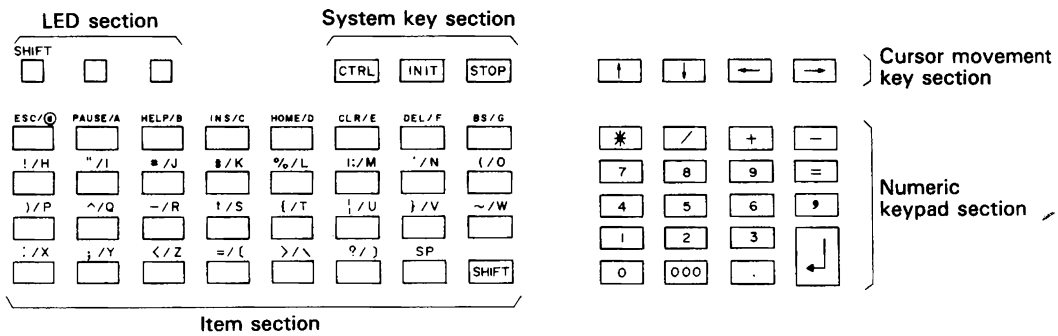


Fig. 3.5.5 Item Keyboard Layout

3.5.4.5 Item section

The item section consists of 32 keys including the SHIFT key. All item section keys other than the SHIFT key can be redefined. That is, 62 items can be defined using the combination of these 31 keys and the SHIFT key. The maximum length of the string defined for each key is 15 characters.

Except the SHIFT key, the 31 item keys have no label printed on their keytop.

The item keyboard is provided, as standard, with two types of overlay sheet: one for the system and the other for the user. The system overlay sheet contains the codes provided by the OS by default and is used for user program development and execution. The user overlay sheet is used during execution of user programs. The user must write the items on the sheet for himself.

The SHIFT key is a software lock key and alternates between the Normal and Shift modes. Its state is indicated by the left-most LED in the LED section. The shift status is held in IMSHFT (ØF91CH).

IMSHFT (ØF91CH) 1 byte
- Item keyboard shift status
  = ØØH: Normal mode
  = 2ØH: Shift mode

(1) Position codes

The key position code identifies the physical position of an item key. It is loaded in the C register when assigning an item function for a key using the BIOS PUTPFK routine.

Figure 3.5.6 shows the position codes in the Normal and Shift modes.

40H	41H	42H	43H	44H	45H	46H	47H
48H	49H	4AH	4BH	4CH	4DH	4EH	4FH
50H	51H	52H	53H	54H	55H	56H	57H
58H	59H	5AH	5BH	5CH	5DH	5EH	SHIFT

Normal mode

60H	61H	62H	63H	64H	65H	66H	67H
68H	69H	6AH	6BH	6CH	6DH	6EH	6FH
70H	71H	72H	73H	74H	75H	76H	77H
78H	79H	7AH	7BH	7CH	7DH	7EH	SHIFT

Shift mode

Fig. 3.5.6 Item Key Position Codes

(2) Default item key codes

The default item key codes are listed in Figure 3.5.7. The characters in the upper rows are those which printed on the overlay sheet. The numbers in the lower rows are key codes themselves.

@	A	B	C	D	E	F	G
40H	41H	42H	43H	44H	45H	46H	47H
H	I	J	K	L	M	N	O
48H	49H	4AH	4BH	4CH	4DH	4EH	4FH
P	Q	R	S	T	U	V	W
50H	51H	52H	53H	54H	55H	56H	57H
X	Y	Z	[	\	]	SP	SHIFT
58H	59H	5AH	5BH	5CH	5DH	20H	

Normal mode

ESC	PAUSE	HELP	INS	HOME	CLR	DEL	BS
1BH	13H	00H	12H	0BH	0CH	7FH	08H
!	''	#	$	%	&	'	(
21H	22H	23H	24H	25H	26H	27H	28H
)	^			{	¦	}	~
29H	5EH	5FH	60H	7BH	7CH	7DH	7EH
:	;	<	=	>	?	SP	SHIFT
3AH	3BH	3CH	3DH	3EH	3FH	20H	

Shift mode

Fig. 3.5.7 Default Item Key Codes

Default key codes are set during system initialization, reset, or a warm boot started with resident not specified. They are also set when the keyboard is replaced by an item keyboard.

(3) Defining/cancelling item functions

Item functions can be defined or cancelled using the BIOS PUTPFK routine.

Item functions are set to their default values when an application program is activated. Therefore, the item functions must be defined or cancelled in the application program as required. It is desirable for some applications that unnecessary keys return no code. In such a case, the user should define only the necessary keys using the following steps:
1) Cancel all item functions.
2) Define only the necessary item functions.

See "BIOS PUTPFK" for the detailed procedure for defining and cancelling item functions.

3.5.4.6 Numeric keypad and cursor movement key sections

The numeric keypad and cursor movement key sections consist of 19 and 4 keys, respectively. When one of these keys is pressed, the code corresponding to the character printed on the keytop is returned.

Figure 3.5.8 shows the codes assigned to the keys in the numeric keypad and cursor movement key sections. The characters or symbols in the upper rows are also printed on the key top. The numbers in the lower rows are key codes.

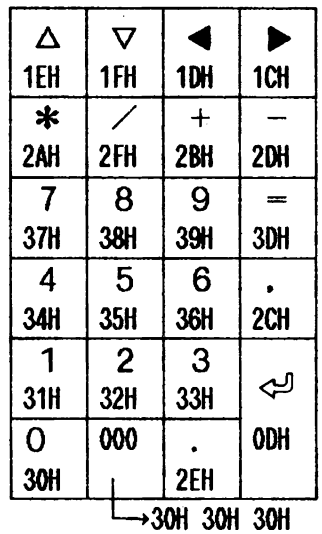


Fig. 3.5.8 Key Codes of the Keys in the Numeric Keypad and Cursor Movement Sections

Basically, the functions of the keys in the numeric keypad and cursor movement key sections cannot be redefined (BIOS supports no routine for this purpose). To meet the requirements of a variety of applications, however, the PINE is designed so that the user can redefine the functions of the keys in these sections as he likes.

Redefinition of key functions is accomplished by directly rewriting the redefinition table (ITCDTB) in the resident area in RAM. Any values from ØØH through ØFFH can be specified. If ØFFH is specified for a key, three "Ø" (3ØH) codes are returned when that key is pressed.

ITCDTB (ØFØDDH) 24 bytes
- Item keyboard redefinition table

Address	Key position	Printed character	Initial value
FODEH	1	△	1EH
FODFH	2	▽	1FH
FOE0H	3	◀	1DH
FOE1H	4	▶	1CH
FOE2H	5	*	2AH
FOE3H	6	/	2FH
FOE4H	7	+	2BH
FOE5H	8	−	2DH
FOE6H	9	7	37H
FOE7H	10	8	38H
FOE8H	11	9	39H
FOE9H	12	=	3DH
FOEAH	13	4	34H
FOEBH	14	5	35H
FOECH	15	6	36H
FOEDH	16	.	2CH
FOEEH	17	1	31H
FOEFH	18	2	32H
FOF0H	19	3	33H
FOF1H	20	0	30H
FOF2H	21	000	FFH
FOF3H	22	•	2EH
FOF4H	23	↵	ODH

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16
17	18	19	23
20	21	22	

3.5.4.7 System key section

Three system keys (CTRL, INIT, and STOP) make the standard system functions, except the system display and screen dump (they can be accomplished through BIOS), available to the user who uses an item keyboard. The procedures for using the standard system functions follow.

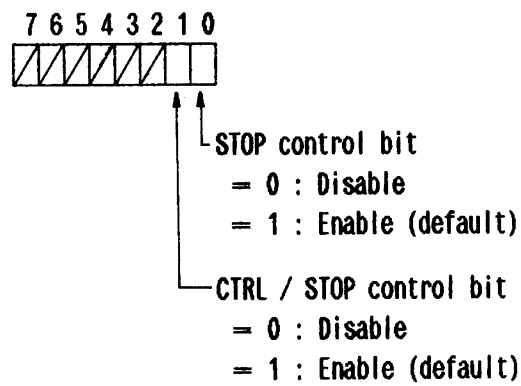
(1) System initialize
Press the reset switch while holding down the STOP and INIT keys. See Section 2.2, "System Initialize" for details of the system actions.

(2) Continue mode power off
Continue mode power-off is specified as standard when an item keyboard is installed. Even when this setting is changed to restart mode power-off, continue mode power-off will be restored if the power switch is turned off with the CTRL key held down. See Section 2.5, "Power-off" for system actions.

(3) STOP, CTRL/STOP
The STOP and CTRL/STOP keys on the item keyboard perform the same function as the STOP or CTRL/STOP key on the standard keyboard. These key functions can be disabled by setting the system function flag (ISYSREG) to Ø.

(4) Programming notes
Both STOP and CTRL/STOP key functions are disabled when the PINE is enabled only for STOP key interrupts. The CTRL key on the item keyboard, unlike that on the standard keyboard, does not change the keyboard state.

ISYSREG (ØF1EH) 1 byte
- System function flag



3.5.4.8 LED section

The item keyboard has three LEDs. One of them is used by the system to indicate the keyboard state. That LED indicates the Normal mode when it is lit and the Shift mode when it is turned off. The other two LEDs are available to the user.

Turning on and off the LED is controlled through BIOS CONOUT.

3.5.4.9 Starting an item

When an item keyboard is installed, the PINE can be turned off, by default, in the continue mode when the power switch is turned off. Once an application is started, the PINE can resume processing in the mode that was established when power was turned off when the power switch is turned on. Therefore, the user need only consider the mode in which his application is to be started for the first time.

Application programs are stored in the following three storage devices:

1. External cassette (only BASIC)
2. Resident program
3. Drives (A: to K:)

(1) Starting an application program from the external cassette
Select BASIC from the menu and load and execute the program from BASIC.

(2) Starting an application program in resident mode
A BASIC or application program can be made resident by: 1) starting it from the standard keyboard, 2) performing resident processing on the program, 3) turning off the power switch, and 4) switching to the item keyboard. The resident program is automatically activated when the power switch on the item keyboard is turned on.

(3) Starting an application program from a drive (A: - K:)
Select the target program from the menu. This automatically starts the program. When an item keyboard is installed, only drives A: to D: and H: to K: are displayed in the menu.

3.5.4.10 Terminating an item

(1) When an item keyboard is installed, the PINE, by default, is switched off in the continue mode only by turning off the power switch so that it can continue processing in the mode selected before power off when the power switch is turned on.

(2) Power must be turned off in the restart mode instead of continue mode depending on the application. The PINE power-off state are controlled through the continue flag. The continue flag can be manipulated by the application program using the BIOS CONTINUE routine or by directly rewriting the system area (IFRCECNT). See the descriptions about BIOS CONTINUE for details.

(3) In the restart mode, the application program can be automatically started (cold start) by defining an auto start string. After Resident processing, it can be warm-started simply by turning on the power switch.

3.5.4.11 Item functions on the standard keyboard.

The keys on the standard keyboard may also be used as item function keys by setting the item flag or pressing the CTRL and SHIFT keys simultaneously. In either case, the codes of the depressed keys are returned if no function is defined for them.

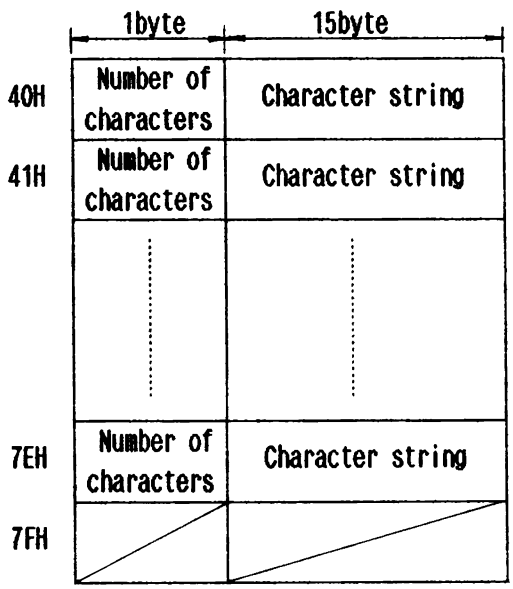
(1) Setting the item flag
Set the item flag (ITEMFLG) to 80H using the BIOS PUTPFK routine. The keys that are assigned a code from 40H to 7FH can be used as item function keys.
Note: Codes entered while pressing the shift key are not displayed even if defined.

(2) CTRL/SHIFT
Make key entries while pressing the CTRL and SHIFT keys (item mode) on the standard keyboard. The keys that will generate codes 40H to 7AH when pressed with neither CTRL nor SHIFT keys held down are defined as item function keys. Key codes 40H to 7AH correspond to the position codes of item functions.

In the CTRL SHIFT mode, keys with A to Z marked on the keytops return:
Uppercase characters 41H to 5AH in the CAPS mode.
Lowercase characters 61H to 7AH in the non-CAPS mode.
The keys associated with the position codes 40H and 5BH through 5EH return the same code regardless of the CAPS mode.

ITEMFLG (F01DH) 1 byte
- Item mode flag
= 00H: Normal mode
= 80H: Item mode

RWITEMTOP (0CC00H) 1024 bytes
- Item function key table



00H <= Number of characters <= 0FH
Keys whose number-of-characters field contains 00H are undefined and no code is returned when they are pressed.

3.5.4.12 Others

(1) When switching from standard keyboard to item keyboard, note that the old DIP switch settings specifying the character set are valid even after the keyboard is replaced (display character fonts vary depending on the specified character set).

(2) When the power switch is turned on, the PINE automatically determines whether an item keyboard is connected. If the keyboard is switched from standard to item keyboard when power is turned on, the system automatically performs a reset and initializes the item keyboard (loading default values).

(3) The default settings when the item keyboard is installed are as follows:
- Keys are not repeatable.
- Power is switched off in the continue mode when only the power switch is turned off.
- System function keys (STOP and CTRL/STOP) are enabled.
- Item function keys are assigned default codes.

(4) The functional differences between the item and standard keyboards are as follows:
- The screen dump subroutine cannot be called through the item keyboard CTRL and PF5 keys, though it can be called via BIOS.
- The system display function cannot be activated by the item keyboard CTRL and HELP keys. To execute this function, find the system display subroutine starting address from the CTRL/HELP address (ØFØDl) in YSUBRTN and call the subroutine via the BIOS CALLX routine.

3.5.5 Work areas for keyboard processing

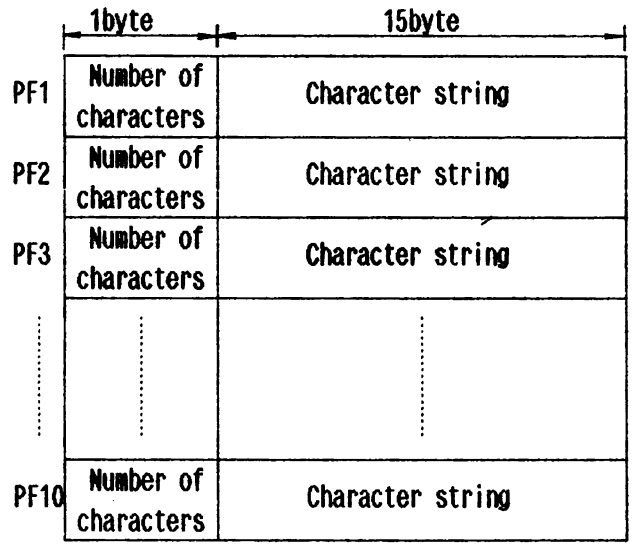
KBUF (ØF86ØH) 33 bytes
- Input key buffer
Stores the key codes received from the 75Ø8. The input key buffer is of the ring structure and controlled by the two pointers called YINTPTR and YKEYPTR. See 3.5.3.1Ø for the input key buffer.

DFLTRT (ØF Ø1FH) 14 bytes
- Loaded with the default codes of the cursor movement keys, SHIFT/INS, and CTRL/INS.

Address	Default code	Key	Address	Default code	Key
F01FH	1CH	→	F026H	FBH	SHIFT/ ↓
F020H	1DH	←	F027H	F6H	SHIFT/INS
F021H	1EH	↑	F028H	FCH	CTRL/ →
F022H	1FH	↓	F029H	FDH	CTRL/ ←
F023H	F8H	SHIFT/ →	F02AH	FEH	CTRL/ ↑
F024H	F9H	SHIFT/ ←	F02BH	FFH	CTRL/ ↓
F025H	FAH	SHIFT/ ↑	F02CH	F7H	CTRL/INS

Even when this area is rewritten, the codes that the keys return are changed only at the next warm boot. See 3.5.3.8 for details.

PFKTAB (ØF02DH) 160 bytes
- PF key default data table



The number of characters may be in the range from ØØH to ØFH. ØØH indicates that no string is defined. If this area is rewritten, the PF key data is changed at the next warm boot. See 3.5.3.5 for details.

YSUBRTN (ØFØCDH) 16 bytes
- Subroutine call jump table

Key for activating subroutine	Address	Default function
CTRL/ESC	FOCDH	Does nothing but return.
CTRL/PAUSE	FOCFH	Does nothing but return.
CTRL/HELP	FOD1H	System display.
CTRL/PF1	FOD3H	Does nothing but return.
CTRL/PF2	FOD5H	Does nothing but return.
CTRL/PF3	FOD7H	Does nothing but return.
CTRL/PF4	FOD9H	Does nothing but return.
CTRL/PF5	FODBH	Screen dump.

See 3.5.3.6 for subroutine calls.

ITCDTB (ØFØDDH) 24 bytes
- Redefinition table for item keyboard keys

Address	Character on keytop	Default code	Address	Character on keytop	Default code
FODDH	STOP	03H	FOE9H	=	3DH
FODEH	△	1EH	FOEAH	4	34H
FODFH	▽	1FH	FOEBH	5	35H
FOEOH	◀	1DH	FOECH	6	36H
FOE1H	▶	1CH	FOEDH	,	2CH
FOE2H	*	2AH	FOEEH	1	31H
FOE3H	/	2FH	FOEFH	2	32H
FOE4H	+	2BH	FOFOH	3	33H
FOE5H	−	2DH	FOF1H	0	30H
FOE6H	7	37H	FOF2H	000	FFH
FOE7H	8	38H	FOF3H	•	2EH
FOE8H	9	39H	FOF4H	↵	ODH

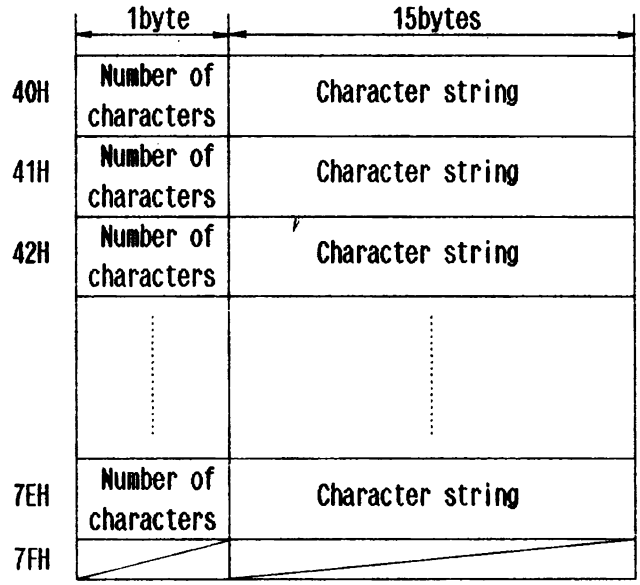
See 3.5.4.6 for more information.

CHGRIGT (ØF328H) 14 bytes
- Table containing key codes of the cursor movement keys, SHIFT/INS, and CTRL/INS.

Address	Default code	Key	Address	Default code	Key
F328H	1CH	→	F32FH	FBH	SHIFT/ ↓
F329H	1DH	←	F330H	F6H	SHIFT/INS
F32AH	1EH	↑	F331H	FCH	CTRL/ →
F32BH	1FH	↓	F332H	FDH	CTRL/ ←
F32CH	F8H	SHIFT/ →	F333H	FEH	CTRL/ ↑
F32DH	F9H	SHIFT/ ←	F334H	FFH	CTRL/ ↓
F32EH	FAH	SHIFT/ ↑	F335H	F7H	CTRL/INS

14-byte data starting at DFLTRT is copied at every warm boot. The key entry routine references this table. See 3.5.3.8 for details.

RWITEMTOP (ØCCØØH) 1Ø24 bytes
- Item functiqn table



See 3.5.4.5 and 3.5.4.11.

KEYF (ØFØØ8H) 1 byte
- Console buffer (KEYD) status flag
    = ØØH: No input data present
    = ØFFH: Input data present
This flag is set by CONST and reset by BIOS CONIN.

KEYD (ØFØØ9H) 1 byte
- Console input buffer
Data is loaded by BIOS CONST into this area.

KEYS (ØFØØAH) 1 byte
- Console buffer key status flag
    = ØØH: Nocheck mode data
    = ØFFH: Check mode data
This flag indicates whether the data in the console buffer is check mode data. It is valid when the current console is the standard keyboard. See the description about BIOS CONIN for check mode.

YINTPTR (ØFØØBH) 2 bytes
- Key buffer (KBUF) put pointer
This area contains the key buffer address into which the key interrupt handling routine is to load the code received from the 7508.

YKEYPTR (ØFØØDH) 2 bytes
- Key buffer (KBUF) get pointer
This area contains the key buffer address from which the key entry processing routine is to get input key codes.

YSHFDT (ØFØØFH) 1 byte
- Standard keyboard switch key status flag
    Bit 7 -- CTRL (Set to 1 when pressed.)
        6 -- GRPH (Set to 1 when the mode is on.)
        5 -- Don't care
        4 -- NUM (Set to 1 when the mode is on.)
        3 -- Don't care
        2 -- CAPS (Set to 1 when the mode is on.)
        1 -- SHIFT (left) (Set to 1 when pressed.)
        Ø -- SHIFT (right) (Set to 1 when pressed.)
See 3.5.3.13 for mode transition.

ZSHFT (ØFØ1ØH) 2 bytes
- Loaded with the starting address of the key code conversion table selected based on the YSHFDT contents.

YPFKSTR (ØFØ12H) 2 bytes
- Loaded with the starting address of the PF key table. See 3.5.3.5.

YPFKPTR (ØFØ14H) 2 bytes
- The get pointer for PF key or item key strings.

YPFKCNT (ØFØ16H) 1 byte
- The counter indicating the number of characters left in the buffer containing the PF key or item key string.

YPFCMFLG (ØFØ17H) 1 byte
- PF key check mode flag
    = ØØH: Nocheck mode
    = ØFFH: Check mode
See the description about BIOS CONIN for check mode.

YSFCMFLG (ØF018H) 1 byte
- Special key check mode flag
     = ØØH: Nocheck mode
     = ØFFH: Check mode
See the description about BIOS CONIN.

BRKFLG (ØF019H) 1 byte
- The flag indicating whether the STOP key has been pressed.
     = ØØH: STOP key pressed.
     = ØFFH: STOP key not pressed.
See 3.5.3.4 for details.

CSTOPFLG (ØF01AH) 1 byte
- The flag indicating whether the CTRL/STOP key has been pressed.
     = ØØH: CTRL/STOP key not pressed.
     = Ø1H: CTRL/STOP key pressed.
See 3.5.3.4 for details.

CSTOPMCT (ØF01BH) 1 byte
- Microcassette termination flag
     = ØØH: CTRL/STOP key not pressed.
     = Ø1H: CTRL/STOP key pressed.
See 3.5.3.4 for details.

IMSHFT (ØF01CH) 1 byte
- Item keyboard shift mode flag
     = ØØH: Normal mode
     = 2ØH: Shift mode
See 3.5.3.5 for shift mode.

ITEMFLG (ØF01DH) 1 byte
- Item mode flag (standard keyboard)
     = ØØH: Normal mode
     = 8ØH: Item mode
See 3.5.3.11 for item mode.

ISYSREG (ØF01EH) 1 byte
- System function flag (item keyboard)
     Bits 7 - 2: Don't care.
     Bit 1: CTRL/STOP control bit (disabled when this bit is Ø).
          Ø: STOP control bit (disabled when this bit is Ø).
See 3.5.3.7 for details.

CSTOPPRN (ØF309H) 1 byte
- Cartridge printer termination flag
     = ØØH: CTRL/STOP key not pressed.
     = Ø1H: CTRL/STOP key pressed.
See 3.5.3.4 for details.

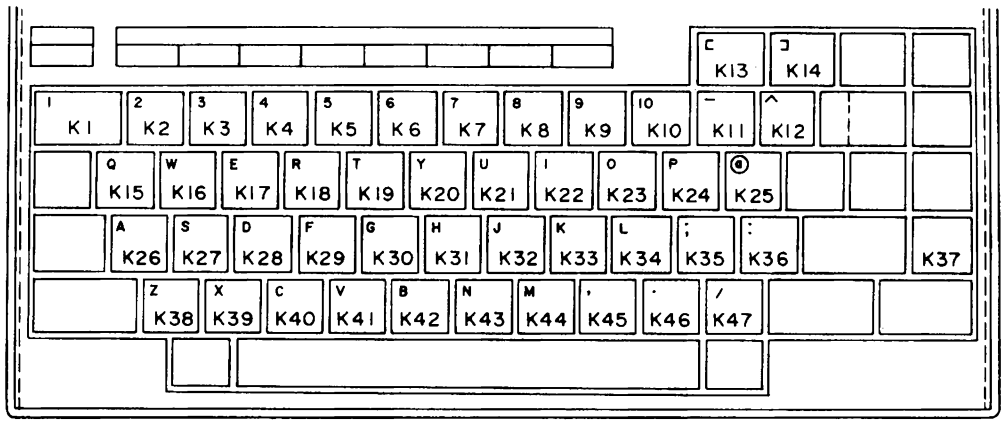
KEYFLG (ØF4D8H) 1 byte
- Switch key status flag used for keyboard interrupt processing
     Bits 7, 6 -- Don't care.
     Bit 5 -- NUM
          4 -- SHIFT (right)
          3 -- GRPH
          2 -- CAPS
          1 -- SHIFT (left)
          Ø -- CTRL

3.5.6 Key Code Charts

This subsection shows the key codes of the standard and item keyboards.

3.5.6.1 Key code chart (standard keyboard)

The pages that follow contain key input code charts 1 to 10. The correspondence between the key numbers in the charts and the keyboard is shown below.



The relationship among key input code charts, keyboard types, and keyboard modes is as follows:

Mode / key-board type	NORMAL Normal shift	CAPS Caps shift	NUM Num shift	GRAPH Graph shift	CONTROL Control shift
ASCII	Table 1	Table 1	Table 2	Table 4	Table 5
France	Table 1 + Table 10	Table 1 + Table 10	Table 3	Table 4	Table 6
Germany	Table 1 + Table 10	Table 1 + Table 10	Table 3	Table 4	Table 7
England	Table 1 + Table 10	Table 1 + Table 10	Table 2	Table 4	Table 5
Denmark, Sweden Norway	Table 1 + Table 10	Table 1 + Table 10	Table 3	Table 4	Table 8

Key input code Table 1
(1) Type: ASCII
(2) Mode: Normal, Normal shift, Caps, Caps shift
(3) Display mode: Characters on keytops

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	HELP		SP	0	@	P	‘	p								
1			!	1	A	Q	a	q								
2		INS	"	2	B	R	b	r								
3	STOP	PAUSE	#	3	C	S	c	s								
4			$	4	D	T	d	t								
5			%	5	E	U	e	u								
6			&	6	F	V	f	v								
7			´	7	G	W	g	w								
8	BS		(	8	H	X	h	x								
9	TAB		)	9	I	Y	i	y								
A			*	:	J	Z	j	z								
B	HOME	ESC	+	;	K	[	k	{								
C	CLR	→	,	<	L	\	l	¦								
D	RETURN	←	-	=	M	]	m	}								
E		↑	.	>	N	^	n	~								
F		↓	/	?	O	-	o	DEL								

Notes:
- In the Caps shift mode, the relationship between the uppercase and lowercase characters are reversed.
- PF1 to PF1Ø return the predefined character string.
- SCRN (SHIFT/INS) returns no code.
- STOP, ESC, PAUSE, HELP, TAB, space bar, and return key return the same code whether the keyboard is in normal or shift mode.
- SHIFT/←, →, ↑, and ↓ return no code.

Key input code chart 2
(1) Type: ASCII, England
(2) Mode: Numeric, Numeric shift
(3) Display mode: Characters on keytops

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	HELP		SP	0(H)												
1				1(J)												
2		INS		2(K)												
3	STOP	PAUSE		3(L)												
4				4(U)												
5				5(I)												
6				6(O)												
7				7												
8	BS		(([)	8												
9	TAB		)(])	9												
A			*(:)													
B	HOME	ESC	+(;)													
C	CLR	→	.													
D	RETURN	←	-(=)													
E		↑	•			^										
F		↓	/	?(@)				DEL								

Notes:
- PF1 to PF1Ø returns the predefined character string.
- SCRN (SHIFT/INS) returns no code.
- STOP, ESC, PAUSE, HELP, TAB, space bar, and return key return the same code whether the keyboard is in normal or shift mode.
- SHIFT/←, →, ↑, and ↓ return no code.
- The characters in the parentheses are the uppercase characters for the corresponding keys.

Key input code Table 3
(1) Type: France, German, Denmark, Sweden, Norway
(2) Mode: Numeric, Numeric shift
(3) Display mode: Key numbers, characters on keytops

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	HELP		SP	K10 K44												
1				K1 K32												
2		INS		K2 K33												
3	STOP	PAUSE		K3 K34												
4				K4 K21												
5				K5 K22												
6				K6 K23												
7				K 7												
8	BS		K11	K 8												
9	TAB		K12	K 9												
A			K35													
B	HOME	ESC	K24													
C	CLR	→	K45													
D	RET	←	K25													
E		↑	K46			K14										
F		↓	K36	K47				DEL								

Notes:
- PF1 to PF1Ø return the predefined character string.
- SCRN (SHIFT/INS) returns no code.
- STOP, ESC, PAUSE, HELP, TAB, space bar, and return key return the same code whether the keyboard is in normal or shift mode.
- SHIFT/←, →, ↑, and ↓ return no code.

Key input code Table 4
(1) Type: All types
(2) Mode: Graph, Graph shift
(3) Display mode: Key numbers, characters on keytops

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	HELP		SP						K27	K21					K10	
1									K39	K22					K1	
2		INS							K16	K23					K2	
3	STOP	PAUSE							K28	K24					K3	
4									K26	K25					K4	
5									K19	K33					K5	
6									K18	K41					K6	
7									K15	K45					K7	
8	BS								K17	K44					K8	
9	TAB								K38	K43					K9	
A									K40	K42					K11	
B	HOME	ESC							K32	K35					K12	
C	CLR	→							K29	K46					K13	
D	RETURN	←							K30	K36					K37	
E		↑							K31	K47					K14	
F		↓						DEL	K20	K34					S/K14	

Notes:
- PF1 to PF1Ø return the predefined character string.
- SCRN (SHIFT/INS) returns no code.
- STOP, ESC, PAUSE, HELP, TAB, space bar, and return key return the same code whether the keyboard is in normal or shift mode.
- SHIFT/←, →, ↑, and ↓ return no code.

Key input code Table 5
(1) Type: ASCII, England
(2) Mode: Control Control shift
(3) Display mode: Characters on keytops

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	@	P	SP													0
1	A	Q														1
2	B	R														2
3	C	S														3
4	D	T														4
5	E	U														5
6	F	V														6
7	G	W														7
8	H BS	X														8
9	I TAB	Y														9
A	J	Z														–
B	K HOME	[														;
C	L CLR	\														:
D	M RET	]														,
E	N	^														.
F	O	/						DEL								

Notes:
- STOP, ESC, PAUSE, HELP, and PF1 to PF5 call a subroutine.
- TAB, return key, and space bar return the same code whether the keyboard is in normal or shift mode.
- INS, ←, →, ↑, and ↓ return no code.

Key input code Table 6
(1) Type: France
(2) Mode: Control Control shift
(3) Display mode: Characters on keytops

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	à	P	SP													ù
1	A	Q														<
2	B	R														#
3	C	S														&
4	D	T														é
5	E	U														"
6	F	V														'
7	G	W														(
8	H BS	X														è
9	I TAB	Y														!
A	J	Z														‘
B	K HOME	)														,
C	L CLR	Ç														;
D	M RET	§														:
E	N	^														=
F	O	–						DEL								

Notes:
- STOP, ESC, PAUSE, HELP, and PF1 to PF5 call a subroutine.
- TAB, return key, and space bar return the same code whether the keyboard is in normal or shift mode.
- INS, ←, →, ↑, and ↓ return no code.

Key input code Table 7
(1) Type: Germany
(2) Mode: Control Control shift
(3) Display mode: Characters on keytops

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	β	P	SP													0
1	A	Q														1
2	B	R														2
3	C	S														3
4	D	T														4
5	E	U														5
6	F	V														6
7	G	W														7
8	H BS	X														8
9	I TAB	Y								\						9
A	J	Z														<
B	K HOME	Ä														#
C	L CLR	Ö														+
D	M RET	Ü														,
E	N	^														.
F	O	—						DEL								

Notes:
- STOP, ESC, PAUSE, HELP, and PF1 to PF5 call a subroutine.
- TAB, return key, and space bar return the same code whether the keyboard is in normal or shift mode.
- INS, ←, →, ↑, and ↓ return no code.

Key input code Table 8
(1) Type: Sweden, Denmark, Norway
(2) Mode: Control Control shift
(3) Display mode: Characters on keytops

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	É	P	SP													0
1	A	Q														1
2	B	R														2
3	C	S														3
4	D	T														4
5	E	U														5
6	F	V														6
7	G	W														7
8	H BS	X														8
9	I TAB	Y														9
A	J	Z														<
B	K HOME	(*1)														+
C	L CLR	(*1)														'
D	M RET	Å														,
E	N	Ü														•
F	O	–						DEL								

Notes:
- STOP, ESC, PAUSE, HELP, and PF1 to PF5 call a subroutine.
- TAB, return key, and space bar return the same code whether the keyboard is in normal or shift mode.
- INS, ←, →, ↑, and ↓ return no code.

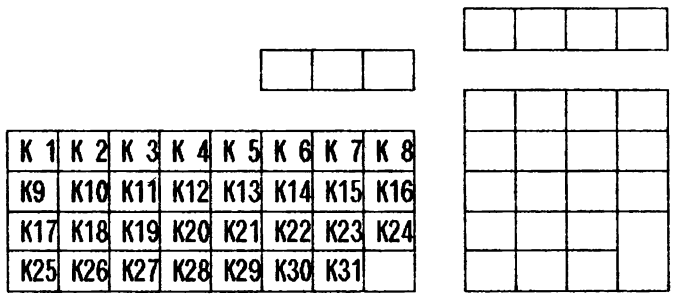
(*1)	Denmark	Sweden	Norway
1BH	? Æ	Ä	Æ
1CH	Ø	Ö	Ø

Table 9 Correspondence between the key codes and characters assorted by country

	U.S.A.	France	Germany	England	Denmark	Sweden	Italy	Spain	Norwa
23H	#	#	#	£	#	#	#	Pt	#
24H	$	$	$	$	$	¤	$	$	¤
40H	@	à	§	@	@	É	@	@	É
5BH	[	°	Ä	[	Æ	Ä	°	¡	Æ
5CH	\	Ç	Ö	\	Ø	Ö	\	Ñ	Ø
5DH	]	§	Ü	]	Å	Å	é	¿	Å
5EH	^	^	^	^	^	Ü	^	^	Ü
60H	`	`	`	`	`	é	ù	`	é
7BH	{	é	ä	{	æ	ä	à	¨	æ
7CH	¦	ù	ö	¦	ø	ö	ò	ñ	ø
7DH	}	è	ü	}	å	å	è	}	å
7EH	~	¨	ß	~	~	ü	ì	~	ü

3.5.6.2 Key code chart (item keyboard)

The following pages contain key input code charts. The correspondence between the key numbers in the charts and the keyboard is shown below.



Key input code table 1
(1) Mode: Normal
(2) Display mode: Key numbers and characters on keytops

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0			K31	0 (*1)	K1	K17										
1				1	K2	K18										
2				2	K3	K19										
3	STOP			3	K4	K20										
4				4	K5	K21										
5				5	K6	K22										
6				6	K7	K23										
7				7	K8	K24										
8				8	K9	K25										
9				9	K10	K26										
A			*		K11	K27										
B			+		K12	K28										
C		▶	,		K13	K29										
D		◀	−	=	K14	K30										
E		△	•		K15											
F		▽	/		K16											

Note:
- CTRL and INIT return no code.

(*1) Three 30H codes are returned when the 000 key is pressed.

Key input code table 2
(1) Mode: Shift
(2) Display mode: Key numbers and characters on keytops

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	K3		K31	0 (*1)			K20									
1			K9	1												
2		K4	K10	2												
3	STOP	K2	K11	3												
4			K12	4												
5			K13	5												
6			K14	6												
7			K15	7												
8	K8		K16	8												
9			K17	9												
A			*	K25												
B	K5	K1	+	K26				K21								
C	K6	▶	,	K27				K22								
D	↵	◀	−	= K28				K23								
E		△	•	K29		K18		K24								
F		▽	/	K30		K19		K7								

Notes:
- CTRL and INIT return no code.

(*1) Three 30H codes are returned when the 000 key is pressed.

3.5.7 Hard Code Charts

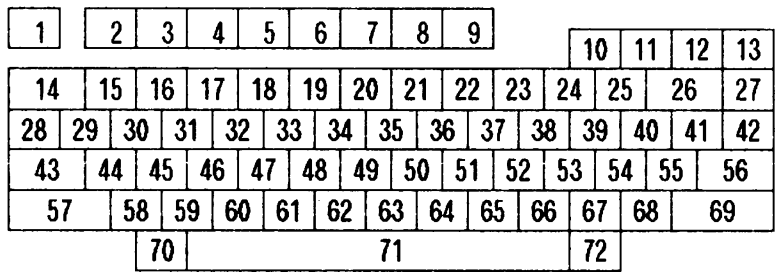
This subsection describes the hard codes of the standard and item keyboards.

Hard codes refers to those codes which are used by the 7508 slave CPU to generate keyboard interrupts and send input key information to the main CPU when keys are pressed or shift keys are released.

The main program routine loads each hard code into the key buffer (KBUF). If the returned hard code is the make code for the STOP key (0B6H) and if the item key board is installed, the program converts the code to 10H, i.e., the STOP key code for the standard keyboard, and places the converted code into the key buffer.

(1) Hard code chart (standard keyboard)

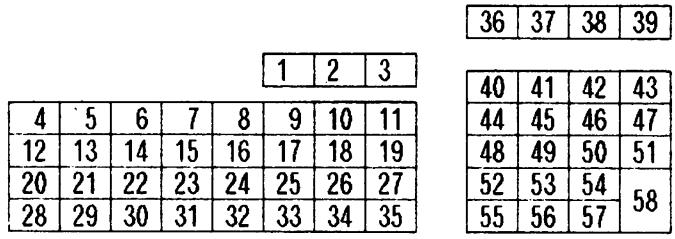
The keys on the standard keyboard are assigned hard codes as shown below. Break and Make in the table indicate the code is generated at the time the corresponding key is released and pressed, respectively.



High order / Low order	0	1	2	3	4	5	6	7	8	9	A	B
0	2	1	29	46	62	21	37	54	12			
1	3	14	30	47	63	22	38	55	13			
2	4	15	31	48	64	23	39	56			43 Break	43 Make
3	5	16	32	49	65	24	40	71			57 Break	57 Make
4	6	17	33	50	66	25	41	58			70 Break	70 Make
5	7	18	34	51	67	26	42	59			72 Break	72 Make
6	8	19	35	52	10	27	44	60			68 Break	68 Make
7	9	20	36	53	11	28	45	61			69 Break	69 Make

(2) Hard code chart (item keyboard)

The keys on the item keyboard are assigned the hard codes as shown below. Break and Make in the table indicate that the code is returned at the time the corresponding key is pressed and released, respectively.



					1	2	3
4	5	6	7	8	9	10	11
12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27
28	29	30	31	32	33	34	35

36	37	38	39
40	41	42	43
44	45	46	47
48	49	50	51
52	53	54	58
55	56	57	

High order / Low order	0	1	2	3	4	5	6	7	8	9	A	B
0			43	51	4	20	12	28				
1		36	44	52	5	21	13	29				
2		37	45	53	6	22	14	30			1 Break	1 Make
3		38	46	54	7	23	15	31				
4		39	47	55	8	24	16	32				
5		40	48	56	9	25	17	33			2 Break	2 Make
6		41	49	57	10	26	18	34			3 Break	3 Make
7		42	50	58	11	27	19				35 Break	35 Make

```
                        ;***********************************************
                        ;       DISPLAY KEY STATUS
                        ;***********************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is displaying the
                        ;               current key status.
                        ;
                        ;       <> 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 input entry
EB0C                    CONOUT          EQU     WBOOT   +09H    ; Console out entry
                        ;
                        ;       System area
                        ;
F00F                    YSHFDT          EQU     0F00FH          ; Normal keyboard key status
F01F                    IMSHFT          EQU     0F01FH          ; ITEM keyboard key status
F775                    YKCOUNTRY       EQU     0F775H          ; Keyboard country
                        ;
0003                    STOP            EQU     03H             ; Stop code
0005                    EOL             EQU     05H             ; Erase end of line
000D                    CR              EQU     0DH             ; Carriage return
000A                    LF              EQU     0AH             ; Line feed
000B                    HOME            EQU     0BH             ; Home code
0012                    CLS             EQU     12H             ; Clear screen
001B                    ESC             EQU     1BH             ; ESC code
                        ;
                        ;***********************************************
                        ;       MAIN PROGRAM
                        ;***********************************************
                        ;
                        ;       NOTE :
                        ;               This program is displaying the current
                        ;               shift key status.
                        ;
0100                    MAIN:
0100    31 1000                 LD      SP,1000H        ; Set stack pointer.
                        ;
0103    0E 12                   LD      C,CLS           ; Clear screen & home
0105    CD EB0C                 CALL    CONOUT          ;
0108    CD 0186                 CALL    CUSROFF         ; Cursor off.
                        ;
010B                    MAIN10:
010B    76                      HALT                    ; Halt (sleep mode)
                        ;
010C    CD 0122                 CALL    KEYST           ; Display key status.
                        ;
010F    CD EB06                 CALL    CONST           ; Input any key?
0112    3C                      INC     A               ;
0113    20 F6                   JR      NZ,MAIN10       ; No.
                        ;
0115    CD EB09                 CALL    CONIN           ; Get inputed key.
0118    FE 03                   CP      STOP            ; STOP?
011A    20 EF                   JR      NZ,MAIN10       ; No.
                        ;
011C    CD 017B                 CALL    CUSRON          ; Cursor on.
011F    C3 EB03                 JP      WBOOT           ; End.
                        ;
                        ;***********************************************
                        ;       DISPLAY KEY STATUS
                        ;***********************************************
                        ;
                        ;       NOTE :
                        ;               Display current key status.
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0122                    KEYST:
0122    21 01DE                 LD      HL,SVSHFT       ; Old key status.
0125    11 F00F                 LD      DE,YSHFDT       ; Current key status (Normal keyboard)
                        ;
0128    3A F775                 LD      A,(YKCOUNTRY)   ; ITEM keyboard?
012B    E6 80                   AND     80H             ;
012D    28 03                   JR      Z,KEY10         ; No.
012F    11 F01F                 LD      DE,IMSHFT       ; Current key status (ITEM)
0132                    KEY10:
0132    1A                      LD      A,(DE)          ; Get current status.
0133    BE                      CP      (HL)            ; Same as old one?
0134    C8                      RET     Z               ; Yes.
                        ;
0135    77                      LD      (HL),A          ; Save current key status.
                        ;
0136    0E 0B                   LD      C,HOME          ; Move cursor to home position.
0138    CD 016F                 CALL    SVCONOUT        ;
                        ;
```

```
013B    21 0191                 LD      HL,SHFTTBL      ; Shift data table.
013E    06 08                   LD      B,08H           ; Loop counter.

0140                    KEY20:
0140    0E 05                   LD      C,EOL           ; Erase end of line.
0142    CD 016F                 CALL    SVCONOUT        ;

0145    1F                      RRA                     ; LSB --> CY
0146    DC 0159                 CALL    C,DSPMSG        ; If ON, then dsp message.
0149    23                      INC     HL              ; Pointer update.
014A    23                      INC     HL              ;
014B    05                      DEC     B               ; Counter decrement.
014C    C8                      RET     Z               ; End.

014D    0E 0D                   LD      C,CR            ; Move cursor to next line.
014F    CD 016F                 CALL    SVCONOUT        ;
0152    0E 0A                   LD      C,LF            ;
0154    CD 016F                 CALL    SVCONOUT        ;
0157    18 E7                   JR      KEY20           ; Loop.

                        ;***************************************************
                        ;       DISPLAY MESSAGE
                        ;***************************************************
                        ;
                        ;       NOTE :
                        ;               Display message
                        ;
                        ;       <> entry parameter <>
                        ;               HL : Data table top address.
                        ;                 (HL+0) -- Byte number
                        ;                 (HL+1) -- Display data
                        ;                       :       :
                        ;
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               All registers
                        ;
                        ;       CAUTION :
                        ;
0159                    DSPMSG:
0159    F5                      PUSH    AF              ; Save registers
015A    C5                      PUSH    BC
015B    D5                      PUSH    DE
015C    E5                      PUSH    HL

015D    5E                      LD      E,(HL)          ; Get data top address.
015E    23                      INC     HL              ;
015F    56                      LD      D,(HL)          ;
0160    EB                      EX      DE,HL           ;

0161    46                      LD      B,(HL)          ; Get displaying data number.
0162    23                      INC     HL              ; HL is displaying data top address.
                        ;
0163                    DSP10:
0163    4E                      LD      C,(HL)          ; Get display data.
0164    CD 016F                 CALL    SVCONOUT        ; Display.
0167    23                      INC     HL              ; Pointer update.
0168    10 F9                   DJNZ    DSP10           ; Loop.

016A    E1                      POP     HL              ; Restore registers.
016B    D1                      POP     DE              ;
016C    C1                      POP     BC              ;
016D    F1                      POP     AF              ;
016E    C9                      RET                     ;
                        ;
                        ;       Conout data with unchanging all registers
                        ;
016F                    SVCONOUT:
016F    F5                      PUSH    AF
0170    C5                      PUSH    BC
0171    D5                      PUSH    DE
0172    E5                      PUSH    HL
0173    CD EB0C                 CALL    CONOUT
0176    E1                      POP     HL
0177    D1                      POP     DE
0178    C1                      POP     BC
0179    F1                      POP     AF
017A    C9                      RET
                        ;
                        ;       Cursor on
                        ;
017B                    CUSRON:
017B    0E 1B                   LD      C,ESC
017D    CD EB0C                 CALL    CONOUT
0180    0E 33                   LD      C,'3'
0182    CD EB0C                 CALL    CONOUT
0185    C9                      RET
                        ;
                        ;       Cursor off
                        ;
0186                    CUSROFF
0186    0E 1B                   LD      C,ESC
0188    CD EB0C                 CALL    CONOUT
018B    0E 32                   LD      C,'2'
018D    CD EB0C                 CALL    CONOUT
0190    C9                      RET
                        ;
                        ;
0191                    SHFTTBL:
0191    01A1                    DW      SHFTR
0193    01AE                    DW      SHFTL
0195    01BA                    DW      CAPS
0197    01C4                    DW      NON
0199    01C6                    DW      NUM
019B    01C4                    DW      NON
```

```
019D  01CE                        DW      GRPH
019F  01D6                        DW      CTRL
                                  ;
01A1                              SHFTR:
01A1  0C                          DB      12
01A2  53 48 49 46                 DB      'SHIFT(right)'
01A6  54 28 72 69
01AA  67 68 74 29
01AE                              SHFTL:
01AE  0B                          DB      11
01AF  53 48 49 46                 DB      'SHIFT(left)'
01B3  54 28 6C 65
01B7  66 74 29
01BA                              CAPS:
01BA  09                          DB      9
01BB  43 41 50 53                 DB      'CAPS LOCK'
01BF  20 4C 4F 43
01C3  4B
01C4                              NON:
01C4  01                          DB      1
01C5  00                          DB      00H
01C6                              NUM:
01C6  07                          DB      7
01C7  4E 55 4D 45                 DB      'NUMERIC'
01CB  52 49 43
01CE                              GRPH:
01CE  07                          DB      7
01CF  47 52 41 50                 DB      'GRAPHIC'
01D3  48 49 43
01D6                              CTRL:
01D6  07                          DB      7
01D7  43 4F 4E 54                 DB      'CONTROL'
01DB  52 4F 4C
                                  ;
                                  ;
01DE                              SVSHFT:
01DE  FF                          DB      0FFH

                                  END
```

```
                        ;       ************************************************
                        ;               KEY IN BY USING HALT
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is how to input key
                        ;               by using halt.
                        ;
                        ;       <> 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 input entry
EB0C                    CONOUT          EQU     WBOOT   +09H    ; Console out entry
EB7E                    POWEROFF        EQU     WBOOT   +7BH    ; Power off entry
                        ;
                        ;       System area
                        ;
EF40                    ATSHUTOFF       EQU     0EF40H          ; Auto power off time (minuite)
EF41                    ATSOTIME        EQU     0EF41H          ; Auto power off time (second)
EF8F                    TIMER0          EQU     0EF8FH          ; 1 sec counter
F77C                    TIMEEND         EQU     0F77CH          ; Auto power off time setting area
                        ;
                        ;
0003                    STOP            EQU     03H             ; Stop code
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This program is using CONST routine.
                        ;               And at the time of auto power off,
                        ;               power off by user.
                        ;
0100                    MAIN:
0100    31 1000                 LD      SP,1000H        ; Set stack pointer.
                        ;
0103                    MAIN10:
0103    2A EF41                 LD      HL,(ATSOTIME)   ; Set auto power off time.
0106    ED 5B EF8F              LD      DE,(TIMER0)     ; (ATSOTIME)+(TIMER0) --> (TIMEEND)
010A    19                      ADD     HL,DE           ;
010B    22 F77C                 LD      (TIMEEND),HL    ;
                        ;
010E                    MAIN20:
010E    3A EF40                 LD      A,(ATSHUTOFF)   ; Check auto power off.
0111    B7                      OR      A               ; Disable?
0112    28 0F                   JR      Z,MAIN30        ; Yes.

0114    2A F77C                 LD      HL,(TIMEEND)    ; Check power off time.
0117    ED 5B EF8F              LD      DE,(TIMER0)     ; (TIMEEND)-(TIMER0) < 0 ?
011B    B7                      OR      A               ;
011C    ED 52                   SBC     HL,DE           ;
011E    0E 00                   LD      C,00H           ; Yes. then continue mode power off.
0120    FC EB7E                 CALL    M,POWEROFF      ;
                        ;
0123                    MAIN30:
0123    76                      HALT                    ; Halt (Sleep mode)

0124    CD EB06                 CALL    CONST           ; Input any key?
0127    3C                      INC     A               ;
0128    20 E4                   JR      NZ,MAIN20       ; No.

012A    CD EB09                 CALL    CONIN           ; Get inputed key.
012D    FE 03                   CP      STOP            ; Stop code?
012F    CA EB03                 JP      Z,WBOOT         ; Yes, then end.
0132    4F                      LD      C,A             ; Display inputed key.
0133    CD EB0C                 CALL    CONOUT          ;
0136    18 CB                   JR      MAIN10          ; Loop.

                                END
```

3.6 LCD Display (CONOUT Details)

3.6.1 General

3.6.1.1 Hardware general description

The PINE LCD displays 240 x 64 dots of information in a uniform matrix using 1/64-duty dynamic drives. The basic specifications for the PINE LCD controller are given below.

(1) LCD panel ------ 240 x 64 dots (40 characters x 8 lines)
(2) LCD driver ----- X driver: SED 1120 x 4
Y driver: SED 1130 x 1
(3) VRAM area ------ Reserved in main RAM (in 2K-byte unit)
Valid display area: 30 x 64 = 1920 bytes
(4) Display mode --- Bit image (character generator not provided)
(5) Scroll --------- Vertical dot scroll
(6) Cursor display - Not provided.
(7) Frame frequency- Variable.
(8) Display on/off
function -- Provided.
(9) Low power consumption achieved by use of speed-up signals.

VRAM is located in the main RAM on bank 0. The LCD controller reads and displays the VRAM contents in bit image.

3.6.1.2 Software general description

(1) The PINE OS supports two types of screens: the system screen and the user screen. These two screens can be controlled independently.
(2) The PINE OS uses the concept of virtual screen and provides a screen of 80 characters by 25 lines.
(3) The cursor is displayed under software control. Cursor blinking is controlled by overflow interrupts.
(4) The character generator is located in OS ROM and displays 6 x 8 dot characters.
(5) The display functions are implemented in the form of ESC sequences.

3.6.2 Screen Modes

3.6.2.1 Screen configuration

The PINE uses two types of screen: the system screen (40 characters x 8 lines) and the user screen (a maximum of 80 characters x 25 lines). These screens are independent of each other. That is, manipulating one screen does not affect the other screen at all.

The user can use only the user screen. The system screen is used by the OS functions such as system display, alarm, and power fail.

Figure 3.6.1 illustrates the relationship between the system screen and the user screen.

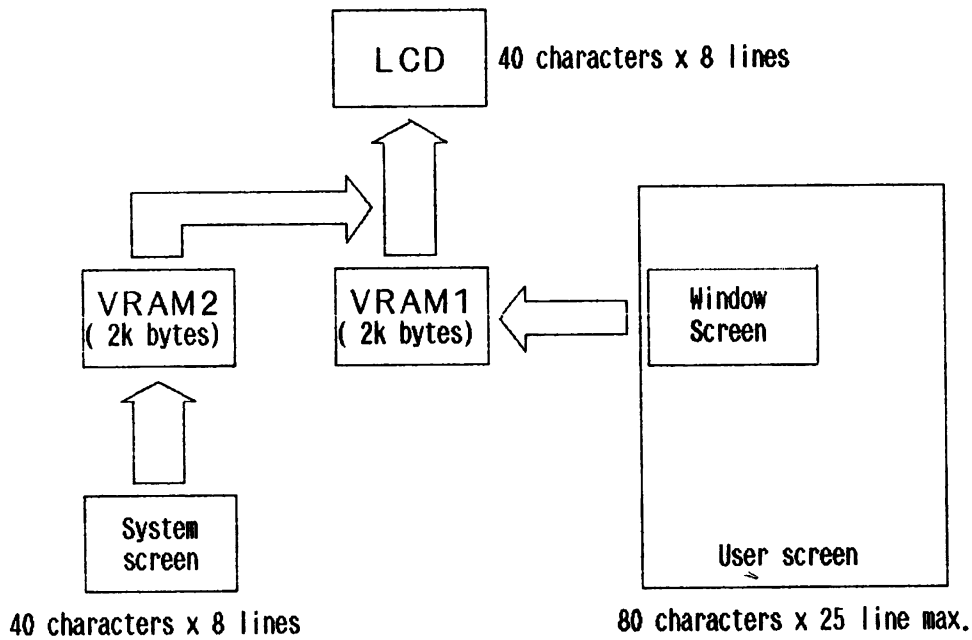


Fig. 3.6.1 System Screen and User Screen

3.6.2.2 Memory map

Figure 3.6.2 shows the memory map for the screen-related areas.

VRAM 1 is the VRAM for the user screen and located in the shared system area (at ØEØØH and above). It holds data in bit image; that is, each bit in VRAM 1 corresponds to a single dot on the LCD.

VRAM 2 is used by the system screen. When the screen mode is switched from user to system screen mode, the bytes in VRAM 2 have one-to-one correspondence with dots on the LCD.

The user-defined character area is used to store the fonts of the user-defined characters.

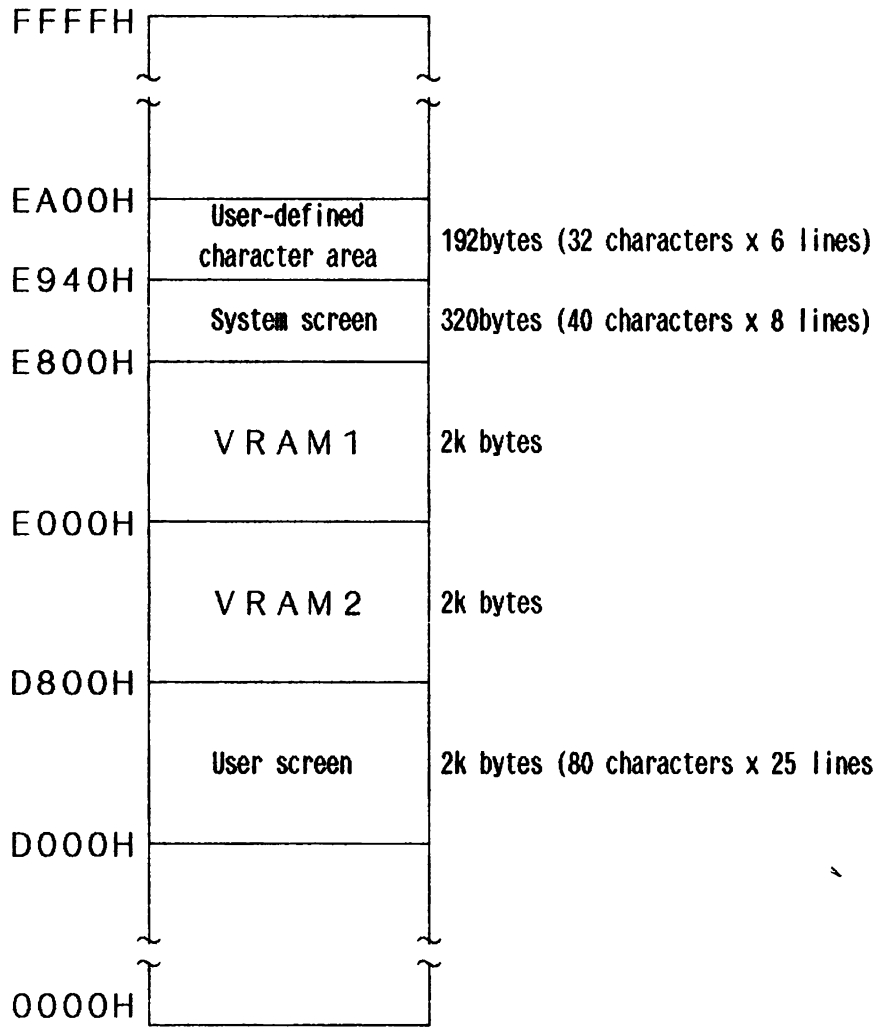


Fig. 3.6.2 Screen-related Area Memory Map

3.6.2.3 VRAM structure

On the following pages are descriptions of the structure and the use of VRAM.

Each bit in VRAM corresponds to a single dot on the LCD panel. The relationship between the LCD panel and VRAM is shown in Figure 3.6.3.

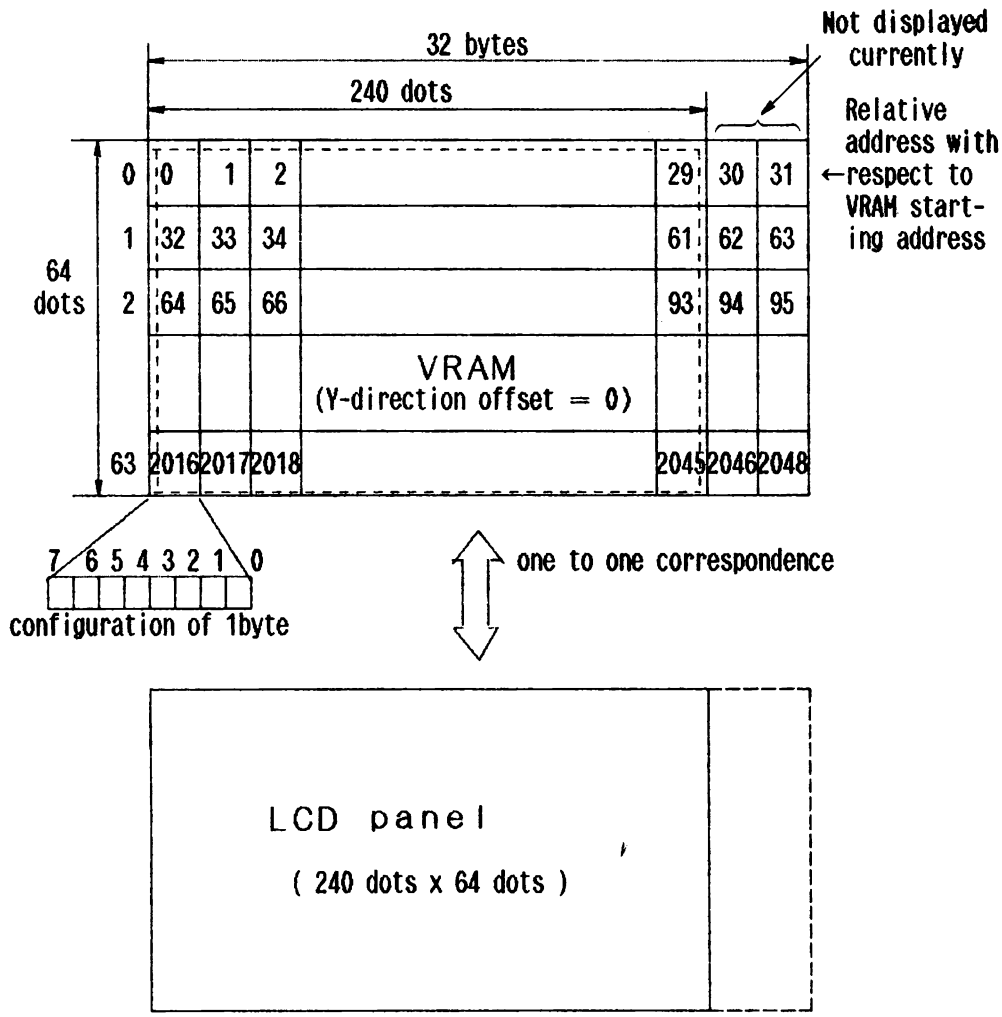


Fig. 3.6.3 Relationship Between VRAM and LCD Panel

The address at which display of VRAM data on the LCD panel begins is determined by:

- VRAM starting address (LSCRVRAM)
- Y direction offset (LVRAMYOF)

The Y-direction offset indicates the vertical correspondence between VRAM and the LCD panel. Display begins at the location offset dots away from the VRAM bottom and wraps around to the VRAM top when it reaches the VRAM bottom. The system uses this feature when performing a vertical scroll.

Figure 3.6.4 illustrates the relationship between the VRAM relative addresses and the LCD panel when the offset is set to two dots.

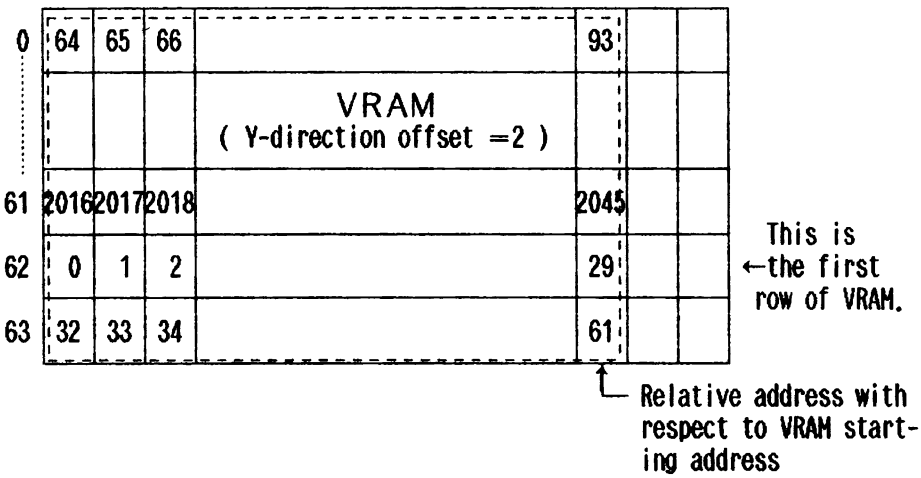


Fig. 3.6.4 Relationship Between VRAM and LCD Panel (When Y-direction Offset = 2)

The following system areas are used to control VRAM display:

LSCRVRAM (ØF294H) 2 bytes
- Contains the starting address of the VRAM (VRAM 1 or VRAM 2) whose contents are currently being displayed on the LCD display. The address must be 8ØØØH or higher and must be on a 2K-byte boundary.

LVRAMYOF (ØF2ADH) 1 byte
- Contains the VRAM Y-direction offset.
Ø ≤ LVRAMYOF ≤ 63
The offset value is usually a multiple of 8.

3.6.2.4 Switching the screen mode

The PINE OS supports two screen modes: the system screen mode used by the system and the user screen mode used by the user. Mode switching is automatically carried out by the OS whenever required. The OS has subroutines for this purpose.

(1) System screen mode
The system screen mode is the one in which the system screen is used with VRAM2. This mode is used mainly by the system. The system handles the switching to this mode as a kind of interrupt generated in the user screen mode. This mode is used by system display, alarm screen, and power fail screen functions.

(2) User screen mode
The user screen mode is the one in which the user screen is used with VRAM1. This mode is available to the user. Normal screen display is performed in this mode.

(3) Mode switching
Screen modes can be switched using the XUSRSCRN and XSYSSCRN subroutines. The entry addresses of these subroutines are found in the OS jump table (see Section 4.2, "Jump Tables").

XUSRSCRN -- Switches the screen mode to user screen.
XSYSSCRN -- Switches the screen mode to system screen.

(4) Note
The user may also use the system screen in the system screen mode. When the system screen is used by the user, the contents of the system screen will be lost if 1) the system display is made by means of the CTRL and HELP keys, 2) the alarm screen is displayed at an alarm/wake time, or 3) the power fail screen is displayed to signal a voltage drop condition. The screen mode is switched back to user screen when the screens is exited.

The screen is always in the user screen mode when the POWER is powered on whether the power-off state is in the continue mode or restart mode.

3.6.2.5 System screen

The system screen is used mainly by the system to display the system display, alarm screen, and power fail screen. It consists of 8 lines of 40 characters. This size is fixed.

The system screen is shown in Figure 3.6.5.

The system screen is functionally equivalent to a user screen whose size is fixed at 40 characters x 8 lines.

In the system screen mode, the cursor always moves in the tracking mode and thus does not move out of the screen.

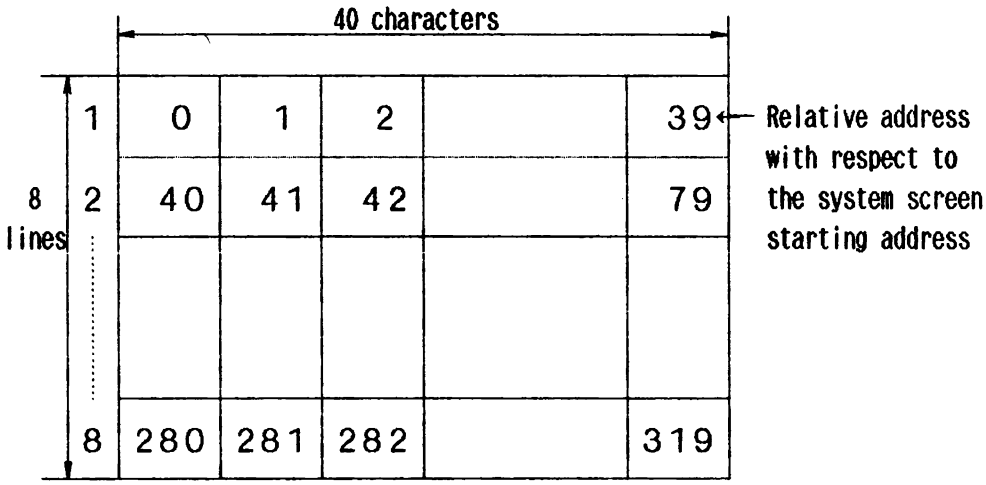


Fig. 3.6.5 System Screen Structure

System screen data is stored in the form of character generator codes.

3.6.2.6 User screen

Although the PINE has a physical display screen of 40 characters by 8 lines, it provides larger virtual screens to meet the needs of the applications programs which require larger screens. See 3.5.3 for the virtual screen.

The user screen size may be in the range from 40 characters by 8 lines to 80 characters by 25 lines or 40 characters by 50 lines.

The structure of the system screen is shown in Figure 3.6.6.

User screen data can be read using the BIOS RDVRAM routine.

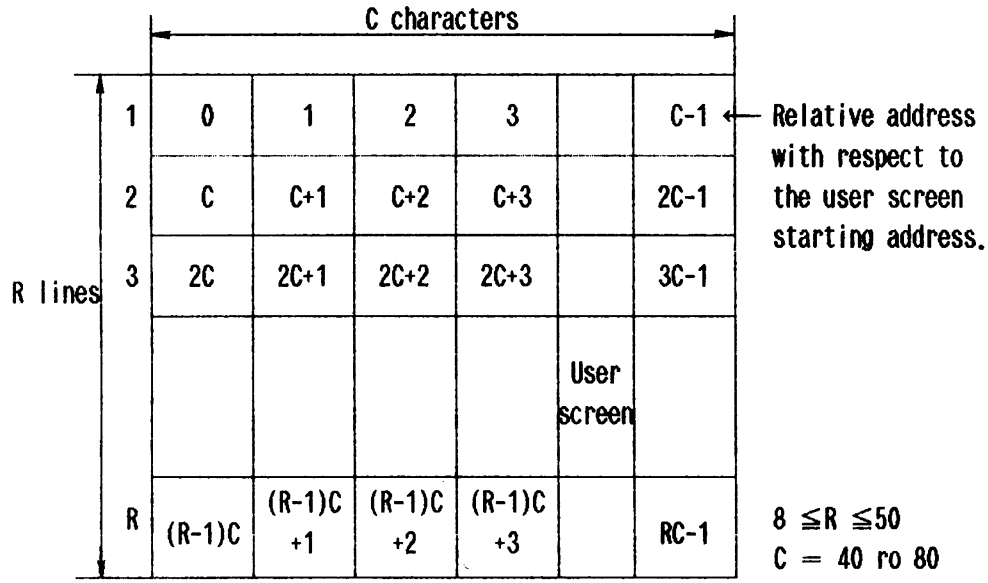


Fig. 3.6.6 User Screen Structure

User screen data is stored in the form of character generator codes.

3.6.3 Virtual Screen

3.6.3.1 Outline

The PINE uses the concept of virtual screen which allows the user to use a screen larger than it actually is (40 characters x 8 liens). The virtual screen may as large as 80 characters by 25 lines or 40 characters by 50 lines.

The portion of the virtual screen that is actually displayed is referred to as a window (40 characters x 8 lines) through which the part of the virtual screen can be viewed.

3.6.3.2 Virtual screen structure

The structure of the virtual screen is shown in Figure 3.6.7.

The virtual screen may be 40 or 80 characters wide. Its length may be any value in the range from 8 to 50 lines provided that (width x length) does not exceed 2000.

The window size is fixed at 40 characters by 8 lines. The entire contents of the virtual screen can be viewed by scrolling the window up and down or right and left over the virtual screen.

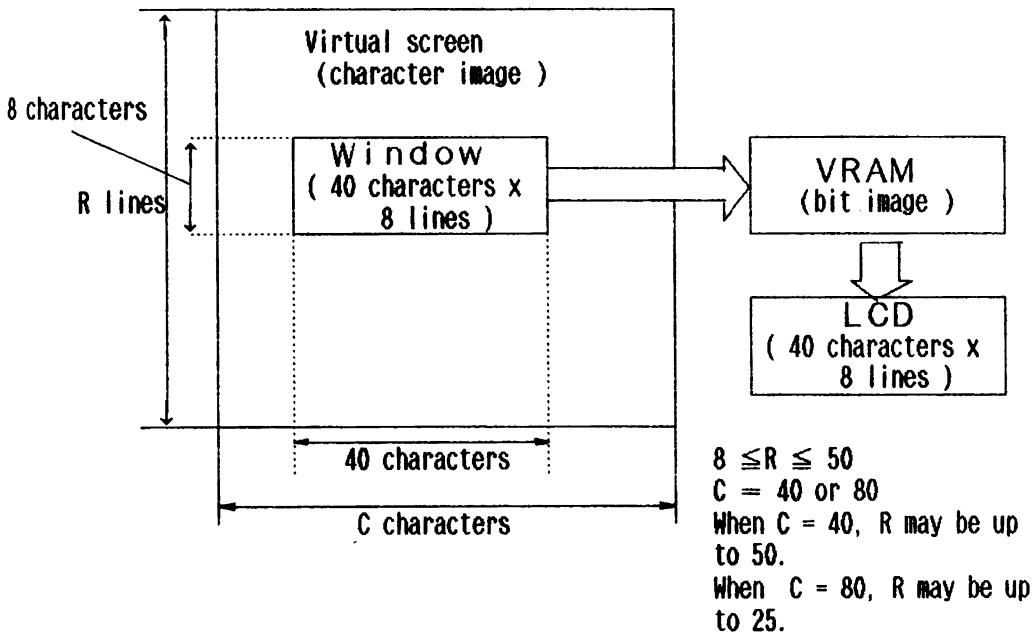


Fig. 3.6.7 Virtual Screen Structure

3.6.3.3 Scroll modes

The window scrolls over the virtual screen in the following three modes:

- Tracking mode
- Nontracking mode
- Horizontally nontracking mode

The window moves, as the cursor moves, in different manners depending on the scroll mode.

The scroll mode can be specified from the keyboard or using commands (ESC sequences).

(1) Tracking mode
In this mode, the window scrolls following the cursor. If the mode is switched from nontracking to tracking when the cursor is out of the window, the window automatically moves up to the point where the cursor is stationed. The tracking mode is valid in both vertical and horizontally directions.

(2) Nontracking mode
In this mode, the cursor does not follow the cursor movement. The window remains in the current position even when data is written into the virtual screen.

The display changes when the cursor moves out of the virtual screen, which triggers an automatic window scroll. However, the cursor in the virtual screen stays in the original position.

(3) Horizontally nontracking mode
This mode is valid only when the virtual screen is 80 characters wide. In this mode, the window follows the cursor movement only vertically and does not follow horizontally.

(4) Specifying the scroll mode
The scroll mode can be changed by pressing the SHIFT and INS keys simultaneously or sending an ESC sequence to the LCD through the BIOS CONOUT routine.

The mode changes cyclically from tracking to nontracking, from nontracking to horizontally nontracking, from horizontally nontracking to tracking, and so on as the SHIFT and INS keys are pressed. See Section 3.5, "Keyboard" for the SHIFT and INS keys.

The scroll mode can also be set by sending an ESC sequence (ESC + 95H) via the BIOS CONOUT routine. See the descriptions of the ESC sequences for how to set the scroll mode.

The system area shown below indicates the current scroll mode.

LSCROLMD (0F2A3H) 1 byte
- Current scroll mode
  - = 00H: Tracking mode
  - = 01H: Nontracking mode
  - = 02H: Horizontally nontracking mode

3.6.3.4 Scrolling the window

The window can be scrolled by 1) moving the cursor, 2) using commands (ESC sequences), or 3) manipulating keys.

(1) Horizontal cursor movement
When the virtual screen is 80 characters wide and in the tracking mode, the window scrolls right and left following the horizontal movement of the cursor.

The horizontal scroll step (w) may be 20 or 40 columns. It can be specified using the ESC sequence "SET SCROLL STEP" (ESC + 94H).
The window scroll margin (m) may be in the range from 0 to 10. This can be specified using the ESC sequence "SET SCROLL MARGIN" (ESC + 98H). The the scroll margin is valid only when the horizontal scroll step is set to 20 columns.

Figures 3.6.8 and 3.6.9 show the relationships between the cursor and the window.

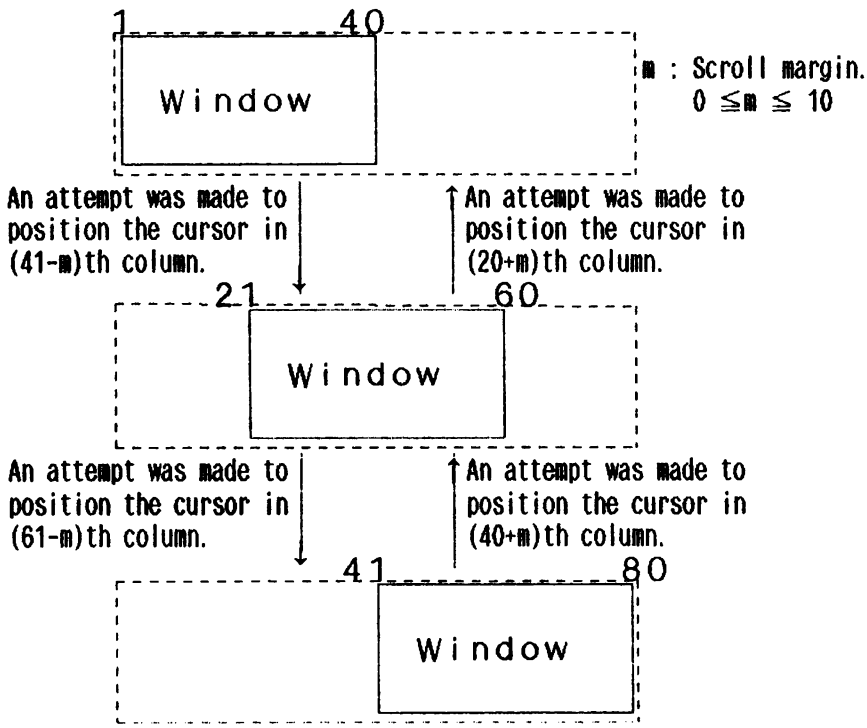


Fig. 3.6.8 Cursor Movements and the Window (When w = 20)

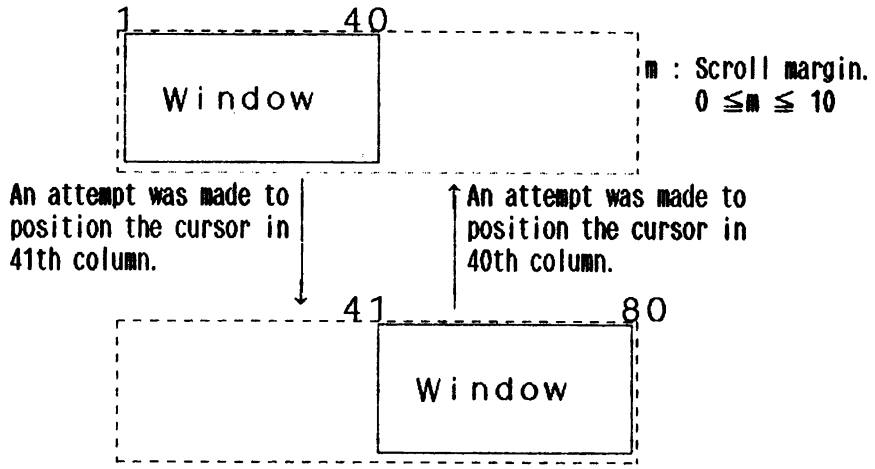


Fig. 3.6.9 Cursor Movements and the Window (When w = 40)

(2) Vertical cursor movement
The vertical scroll step for cursor movement is always set to 1. Whenever the cursor moves beyond the window vertically, the window scrolls one line up or down. The window remains in the current position in the nontracking mode, however.

The vertical scroll step specified by "SET SCROLL STEP" has no effect when the cursor is moving. It is valid only when the window scrolls vertically.

(3) Scrolling under command control
The window can be scrolled vertically or horizontally using commands (ESC sequences).

Horizontal scroll is valid only when the virtual screen is 80 characters wide. In other cases, nothing takes place when horizontal scroll is specified.

Horizontal scroll can be initiated by sending the ESC sequence "SCROLL RIGHT (LEFT) N CHAR" (ESC + 92H or 93H). The scroll step can be specified using the ESC sequence "SET SCROLL STEP" (ESC + 94H).

In the horizontal scroll mode, the cursor remains in the current position and only the window moves over the virtual screen.

Vertical scroll is valid when the virtual screen length is other than 8 lines. Nothing will take place if its length is set to 8 lines.

Vertical scroll can be started by sending the ESC sequence "SCROLL UP (DOWN) M LINE" (ESC + 96H or 97H). The scroll step can be specified using the ESC sequence "SET SCROLL STEP" (ESC + 94H).

In the vertical scroll mode, the cursor remains in the current position and only the window moves over the virtual screen.

(4) Scrolling under key control

In PINE OS, scrolling of the window in the horizontal and vertical directions can be controlled using special keys.

SHIFT/→ Scrolls the window 20 columns to the right.
SHIFT/← Scrolls the window 20 columns to the left.
SHIFT/↑ Scrolls the window one line up.
SHIFT/↓ Scrolls the window one line down.
CTRL/→ Scrolls the window 40 columns to the right.
CTRL/← Scrolls the window 40 columns to the left.
CTRL/↑ Scrolls the window eight lines up.
CTRL/↓ Scrolls the window eight lines down.

Any combination of these keys causes the window to scroll without moving the cursor. See Section 3.5, "Keyboard" for details.

(5) System areas used for controlling window scrolling

LSCROLX (0F2A4H) 1 byte
- Indicates the horizontal scroll step. The scroll step must be 20 or 40 columns. The default value is 20 columns.

LSCROLY (0F2A5H) 1 byte
- Indicates the vertical scroll step. The scroll step must be 1 to 8 lines. The default value is one line.

LLMARGIN (0F2DFH) 1 byte
- Indicates the left margin. The left margin must be 0 to 10 columns. The default value is 5 columns.

LRMARGIN (0F2E0H) 1 byte
- Indicates the right margin. The right margin must be 0 to 10 columns. The default value is 5 columns.

3.6.4 Graphics Display

3.6.4.1 Outline

Graphics data can be written directly into VRAM either in the system screen mode or in the user screen mode.

Graphics data may be mixed with character data in the same screen. If a portion of the currently displayed graphics data goes off the window as the window scrolls, that portion of data is deleted. That is, graphics data will not be redisplayed when the window scrolls back to the original position.

3.6.4.2 Graphics coordinates

The graphics screen is made up of 239 dots by 63 dots with coordinates (Ø,Ø) or origin assigned to the upper left corner of the LCD display screen.

As described in 3.6.2., the Y-direction offset value (in LVRAMYOF) must be taken into consideration when establishing the correspondence between VRAM and LCD..

Figure 3.6.4 shows the correspondence between VRAM and LCD viewed from the LCD's standpoint and Figure 3.6.1Ø shows that viewed from the VRAM's standpoint.

The address of coordinates (X, Y) in VRAM can be obtained using the formula

$$\{(y + Y) \bmod 64\} * 32 + (X / 8)$$

where $Ø \leq X \leq 239$, $Ø \leq Y \leq 63$, and y is a Y-direction offset. The integer part of the result is the relative address with respect to the top of VRAM. The fractional part designates the bit position of the coordinates in the relative address.

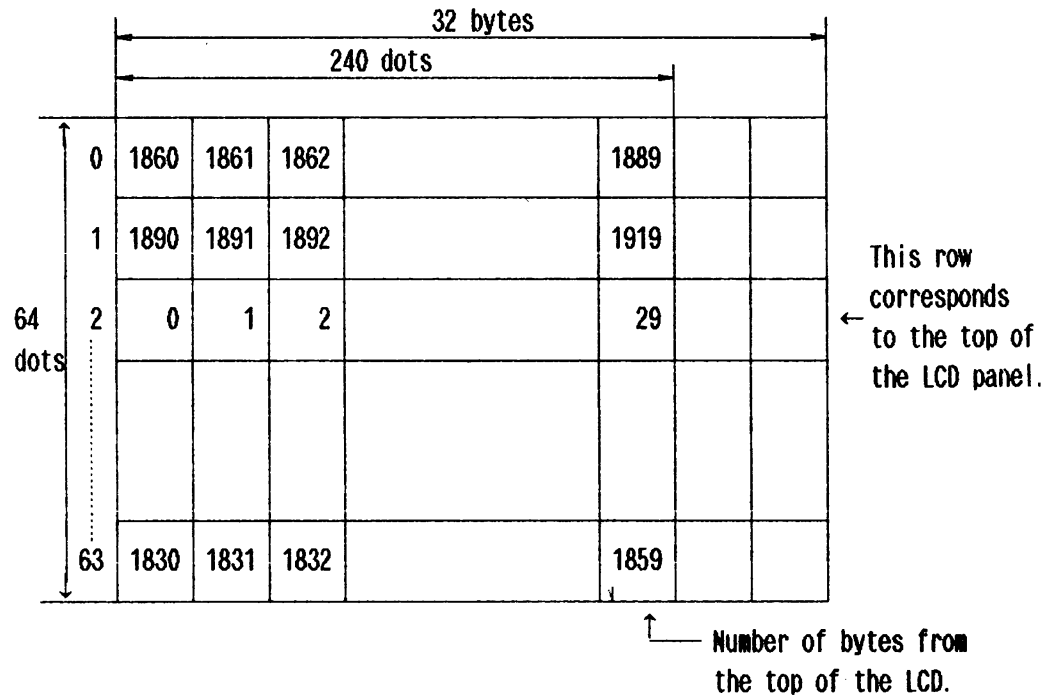


Fig. 3.6.1Ø Relationship Between VRAM and LCD Panel (When Y-direction Offset = 2)

3.6.4.3 Writing graphics data

Graphics data can be displayed on the screen by calling the BIOS PSET routine or by writing the data directly into VRAM.

(1) BIOS PSET
The BIOS PSET routine performs a specified logical operation (AND, OR, XOR) on the VRAM data and the specified data on a byte basis. See Section 3.3, "BIOS Operations" for further information.

(2) Writing data directly into VRAM
Data can be written into VRAM directly with ease when the user screen is used because VRAM is located in the shared system area (RAM area at location ØEØØØH and above).

The formula given on the previous page is used to find the VRAM address into which the data is to be written. Its absolute RAM address can be obtained by adding the VRAM starting address (LSCRVRAM) to the VRAM address.

3.6.5 CONOUT (BIOS)

3.6.5.1 Outline

The PINE CONOUT BIOS routine controls the keyboard, LEDs, and buzzer as well as the LCD.

The description that start at the next page hold when the CON: field of the I/O byte is set to LCD.

The pages that follow lists the functions of the PINE CONOUT BIOS routine. A full descriptions of the CONOUT functions are found in 3.6.5.3, "CONOUT specifications."

Code	Function	code	Function
02H	Screen left	ESC+92H	Scroll right n char
05H	Erase end of screen	ESC+93H	Scroll left n char
06H	Screen right	ESC+94H	Set scroll step
07H	Bell	ESC+95H	Set scroll mode
08H	Back space	ESC+96H	Scroll up m line
09H	Tab	ESC+97H	Scroll down m line
0AH	Line feed	ESC+98H	Set scroll margin
0BH	Home	ESC+A0H	INS LED on
0CH	Clear screen & Home	ESC+A1H	INS LED off
0DH	Carriage return	ESC+A2H	CAPS LOCK LED on
10H	Screen up	ESC+A3H	CAPS LOCK LED off
11H	Screen down	ESC+A4H	NUM LED on
1AH	Erase end of screen	ESC+A5H	NUM LED off
1BH	Escape (ESC)	ESC+B0H	Function key check mode on
1CH	Cursor right	ESC+B1H	Function key check mode off
1DH	Cursor left	ESC+D0H	Set screen size
1EH	Cursor up	ESC+D2H	Direct display
1FH	Cursor down	ESC+D4H	Locate top of screen
		ESC+D5H	Locate bottom of screen
ESC+'%'	Access CGROM directly	ESC+D6H	Select cursor kind
ESC+'('	Block reverse	ESC+D7H	Find cursor
ESC+'*'	Clear screen & Home	ESC+E0H	Set download character
ESC+'0'	Reverse on	ESC+F0H	Keyboard repeat on/off
ESC+'1'	Reverse off	ESC+F1H	Set keyboard repeat start time
ESC+'2'	Cursor off		
ESC+'3'	Cursor on	ESC+F2H	Set keyboard repeat interval time
ESC+'='	Set cursor position		
ESC+'C'	Set character-set table	ESC+F3H	Set arrow key code
ESC+'P'	Screen dump	ESC+F4H	Set scroll key code
ESC+'T'	Erase end of line	ESC+F5H	Set control key code
ESC+'Y'	Erase end of screen	ESC+F6H	Clear key buffer
ESC+7BH	Secret	ESC+F7H	Set key shift
ESC+7DH	Non secret		
ESC+90H	Partial scroll up		
ESC+91H	Partial scroll down		

3.6.5.2 How to use the CONOUT function

(1) Entry address: WBOOT + Ø9H or ØEBØCH
(2) Entry parameter: C = Output data
(3) Return parameter: None.
(4) How to use:
The BIOS CONOUT routine inputs a function code and data. The function code and data are distinguished by their numeric values as shown below:
- Function code: ØØH - lFH
lBH (ESC) + nl + n2 + --- + nk
- Data: 2ØH - ØFFH

The CONOUT routine does nothing but preserves the original state when it detects an invalid function code or parameter in the ESC sequence received. The routine does not check the ESC sequence for parameter errors until it receives all parameters.

3.6.5.3 CONOUT specifications

This large subsection describes the operations that the PINE OS performs when it receives commands through the CONOUT BIOS function.

The word "screen" here refers to either the user screen if the screen is currently in the user mode and to the system screen if the screen is in the system mode.

"The cursor goes out of the window" here means that the cursor is located literally outside the window when the cursor moved in the vertical direction and, when the cursor moved in the horizontal direction, that the cursor moves beyond the window whose sides are delineated by the scroll margins.

- Ø2H Screen left
Moves the window one screenful of columns to the left over the screen.

The left edge of the window is aligned with that of the screen when an attempt is made to move the window beyond the left end of the screen. The cursor remains in the original position on the screen.

- Ø5H Erase end of line
Clears with spaces to the end of the line from the current cursor position (inclusive) on the screen. The cursor remains in the original position on the line.

- Ø6H Screen right
Moves the window one screenful of columns to the right over the screen.

The right edge of the window is aligned with that of the screen when an attempt is made to move the window beyond the right end of the screen. The cursor remains in the original position on the screen.

- Ø7H Bell
Sounds the buzzer at 88Ø Hz for one second. CONOUT is not exited until the buzzer stops.

- 08H Backspace
Moves the cursor one column to the left on the screen.

The cursor moves to the end of the previous line when it is at the beginning of the line when this function is executed. The cursor does not move when it is in the home position (upper left corner) of the screen. How the window behaves when the cursor goes out of the window depends on the scroll mode.

- 09H Tab
Searches for the next tab position starting at the current cursor position on the screen forward and positions the cursor in the first tab position encountered.

If no tab position is found on the line, the function moves the cursor to the beginning of the next line. If the cursor is at the bottom of the screen when this function is executed, the function positions the cursor at the beginning of the line. How the window behaves when the cursor goes out of the window depends on the scroll mode.
Tab position = (1 + 8*n) column n = 0, 1, 2, ---

- 0AH Line feed
Moves the cursor down one line on the screen.

If the cursor is on the bottom line of the screen when this function is executed, the function scrolls the screen one line down (deletes the first line on the screen and puts a blank line at the bottom of the screen) and moves the cursor one line down. How the window behaves when the cursor goes out of the window depends on the scroll mode.

- 0BH Home
Positions the cursor to the home position (upper left corner) of the screen.

How the window behaves when the cursor goes out of the window depends on the scroll mode.

- 0CH Clear screen & home
Clears the entire screen with spaces and performs the Home function (0BH).

- 0DH Carriage return
Moves the cursor to the first column on the current line.

How the window behaves when the cursor goes out of the window depends on the scroll mode.

When this function is executed immediately after the cursor is moved from the last column into the first column on the next line as the result of displaying a character (20H - 0FFH), the function positions the cursor in the first column on the previous line.

- 10H Screen up
Moves the window one screenful of columns down over the screen.

The last line of the window is aligned with the bottom of the screen when an attempt is made to moves the window across the bottom of the screen. The cursor is held in the original position on the screen.

- 11H Screen down
Moves the window one screenful of columns the screen.

The first line of the window is aligned with the top of the screen when an attempt is made to moves the window across the top of the screen. The cursor is held in the original position on the screen.

- 1AH Erase end of screen
Clears to the end of the screen from the current cursor position (inclusive) on the screen. The cursor stays in the original position on the screen.

- 1BH Escape
Initiates an ESC sequence. The commands entered in the form of an ESC sequence are described later.

- 1CH Cursor right
Moves the cursor one column to the right on the screen.

If the cursor is in the last column on a line when this function is executed, the function positions the cursor in the first column on the next line. The function does nothing if the cursor is in the last column on the last line. How the window behaves when the cursor goes out of the window depends on the scroll mode.

- 1DH Cursor left
Performs the same function as Back space (08H).

- 1EH Cursor up
Moves the cursor one line up on the screen.

This function does nothing if it is executed when the cursor is positioned on the first line of the screen. How the window behaves when the cursor goes out of the window depends on the scroll mode.

- 1FH Cursor down
Moves the cursor one line down on the screen.

This function does nothing if it is executed when the cursor is positioned on the last line of the screen. How the window behaves when the cursor goes out of the window depends on the scroll mode.

- 20H - 0FFH Character display
Displays the character associated with the given character code in the current cursor position and then moves the cursor one column to the right.

If the cursor is in the last column on a line, this function displays a character and positions the cursor in the first column on the next line. If the cursor is in the last column on the last line when the function is executed, the display automatically scrolls one line down and the cursor is placed at the beginning of the next line. How the window behaves when the cursor goes out of the window depends on the current scroll mode.

- ESC '%' Access CGROM directly

Reads the character specified by the given code from the character generator and displays it in the cursor position on the screen.

The cursor is positioned in the same way as when the character display function (20H - 0FFH) is executed.

<Command sequence>
1st byte: ESC
2nd byte: '%' n: Code
3rd byte: n 00H ≤ n ≤ 0FFH

- ESC 'C' Block reverse

Displays the specified length of data in reverse video starting at the specified position on the screen.

The reverse video function is cancelled when the data displayed in reverse video goes out of the window during scrolling.

<Command sequence>
1st byte: ESC
2nd byte: 'C' Y: Y-coordinate 1 ≤ Y ≤ 8
3rd byte: Y X: X-coordinate 1 ≤ X ≤ 40
4th byte: X n: Number of characters displayed in
5th byte: n (H) reverse video.
6th byte: n (L) 1 ≤ n ≤ 320

- ESC '*' Clear screen & home

Performs the same function as Clear screen & home (0CH).

<Command sequence>
1st byte: ESC
2nd byte: '*'

- ESC '0' Reverse on

Turns on the reverse display mode and displays the subsequent output characters in reverse video.

The reverse video function is cancelled when the data displayed in reverse video goes out of the window during scrolling.

<Command sequence>
1st byte: ESC
2nd byte: '0'

- ESC '1' Reverse off

Turns off the reverse display mode.

<Command sequence>
1st byte: ESC
2nd byte: '1'

- ESC '2' Cursor off

Suppresses the cursor display. The cursor can subsequently move around, though invisible.

<Command sequence>
1st byte: ESC
2nd byte: '2'

- ESC '3' Cursor on
Displays the cursor.

<Command sequence>
1st byte: ESC
2nd byte: '3'

- ESC '=' Set cursor position
Specifies the cursor position on the screen and positions it in that position.

When the cursor goes out of the window in the tracking mode:
  - If the cursor moves in vertical direction: This function scrolls the window so that the cursor is positioned on the fourth line of the window. If the window goes off the screen, the function aligns the bottom or the top of the window with that of the screen.
  - If the cursor moves in the horizontal direction: This function does not move the window if the cursor position after the move will be within the window. If the cursor goes off the window, however, the function scrolls the window in increments of the scroll step specified.
When the cursor goes out of the window in the nontracking mode:
  The function does not scroll the window.

<Command sequence>
1st byte: ESC
2nd byte: '=' m: Row position
3rd byte: m+1FH 1 ≤ m ≤ Number of lines on the screen
4th byte: n+1FH n: Column position
                 1 ≤ n ≤ Number of columns on the screen

- ESC 'C' Set character set table
Sets up the character set for the given language.

The default language is set by DIP switches. This function does not affect the characters already displayed on the screen.

<Command sequence>
1st byte: ESC ID: Character set identification character
2nd byte: 'C' 'U': USA 'W': Sweden
3rd byte: ID 'F': France 'I': Italy
              'G': Germany 'S': Spain
              'E': England 'N': Norway
              'D': Denmark

- ESC 'P' Screen dump
Outputs the currently displayed VRAM data to the printer.

See BIOS SCRNDUMP for details.

<Command sequence>
1st byte: ESC
2nd byte: 'P'

- ESC 'T' Erase end of line
Performs the same function as Erase end of line (Ø5H).

<Command sequence>
1st byte: ESC
2nd byte: 'T'

- ESC 'Y' Erase end of screen
Performs the same function as Erase end of screen (1AH).

<Command sequence>
1st byte: ESC
2nd byte: 'Y'

- ESC 7BH Secret
Displays characters in the secret mode.

In this mode, characters are converted to spaces when directed to the screen for display.

<Command sequence>
1st byte: ESC
2nd byte: 7BH

- ESC 7DH Nonsecret
Cancels the screen mode.

<Command sequence>
1st byte: ESC
2nd byte: 7DH

- ESC 90H Partial scroll up
Scrolls m lines starting at the nth line up by one line.

This function erases the data on the nth line and changes the data on the (n + m - 1)th line to spaces. If (n + m - 1) is larger than the number of lines on the screen, the function automatically adjusts the value of m so that (n + m - 1) matches the maximum line number.

The cursor is held in the original position on the screen.

<Command sequence>
1st byte: ESC n: Number of the line at which scroll starts.
2nd byte: 90H 1 ≤ n ≤ Number of lines on the screen
3rd byte: n - 1 m: Scroll range
4th byte: m 1 ≤ m ≤ Number of lines on the screen

- ESC 91H Partial scroll down
Scrolls m lines starting at the nth line down by one line.

This function erases the data on the (n+m-1)th line and changes the data on the nth line to spaces. If (n + m - 1) is larger than the number of lines on the screen, the function automatically adjusts the value of m so that (n + m - 1) matches the maximum line number.

The cursor is held in the original position on the screen.

<Command sequence>
1st byte: ESC n: Number of the line at which scroll starts.
2nd byte: 91H 1 ≤ n ≤ Number of lines on the screen
3rd byte: n - 1 m: Scroll range
4th byte: m 1 ≤ m ≤ Number of lines on the screen

- ESC 93H Scroll right n char
Moves the window to the right in the increment of the horizontal scroll step specified.

The right edge of the window is aligned with that of the screen if an attempt is made to move the window beyond the right end of the screen. The cursor remains in the original position on the screen.

<Command sequence>
1st byte: ESC
2nd byte: 92H

- ESC 93H Scroll left n char
Moves the window to the left in the increment of the horizontal scroll step specified.

The left edge of the window is aligned with that of the screen if an attempt is made to move the window beyond the left end of the screen. The cursor remains in the original position on the screen.

<Command sequence>
1st byte: ESC
2nd byte: 93H

- ESC 94H Set scroll step
Specifies the number of lines or columns the window is to be moved in a single scroll operation.

<Command sequence>
1st byte: ESC n: Number of columns to scroll
2nd byte: 94H n = 20 or 40
3rd byte: n m: Number of lines to scroll
4th byte: m 1 ≤ m ≤ 8

The default values of n and m are 20 and 1, respectively.

- ESC 95H Set scroll mode
Specifies the scroll mode.

<Command sequence>
1st byte: ESC M: Scroll mode
2nd byte: 95H = 00H: Tracking mode
3rd byte: M = 01H: Nontracking mode
= 02H: Horizontally nontracking mode

The default mode is horizontally nontracking mode.

- ESC 96H Scroll up m lines
Moves the window up in the increment of the vertical scroll step specified.

The top of the window is aligned with that of the screen if an attempt is made to move the window beyond the screen. The cursor is held in the original position on the screen.

<Command sequence>
1st byte: ESC
2nd byte: 96H

- ESC 97H Scroll down m lines
Moves the window down in the increment of the vertical scroll step specified.

The bottom of the window is aligned with that of the screen if an attempt is made to move the window beyond the screen. The cursor is held in the original position on the screen.

<Command sequence>
1st byte: ESC
2nd byte: 97H

- ESC 98H Set scroll margin
Specifies the horizontal scroll margin.

<Command sequence>
1st byte: ESC
2nd byte: 98H n: Scroll margin
3rd byte: n Ø ≤ n ≤ 1Ø

The default values of n is 5.

- ESC AØH INS LED on
Turns on the INS LED.

The INS LED is the third LED from the top on the standard keyboard and the third LED from the left on the item keyboard.

<Command sequence>
1st byte: ESC
2nd byte: ØAØH

- ESC ØA1H INS LED off
Turns off the INS LED.

<Command sequence>
1st byte: ESC
2nd byte: ØA1H

- ESC A2H CAPS LOCK LED on
Turns on the CAPS LOCK LED.

The CAPS LOCK LED is on the top of the standard keyboard and in the leftmost position on the item keyboard.

<Command sequence>
1st byte: ESC
2nd byte: ØA2H

- ESC A3H CAPS LOCK LED off
Turns off the CAPS LOCK LED.

<Command sequence>
1st byte: ESC
2nd byte: ØA3H

- ESC A4H NUM LED on
Turns on the NUM LED.

The NUM LED is in the next to top position on the standard keyboard and in the next to leftmost position on the item keyboard.

<Command sequence>
1st byte: ESC
2nd byte: ØA4H

- ESC A5H NUM LED off
Turns off the NUM LED.

<Command sequence>
1st byte: ESC
2nd byte: ØA5H

- ESC BØH Function key check mode on
Turns on the mode (PF key check mode) in which PF keys return their inherent codes instead of the defined strings.

See BIOS CONIN for details.

<Command sequence>
1st byte: ESC
2nd byte: ØBØH

- ESC B1H Function key check mode off
Cancels athe PF key check mode.

When a defined PF key is pressed after this function is executed, the associated string is returned.

<Command sequence>
1st byte: ESC
2nd byte: ØB1H

- ESC DØH Set screen size
Defines the user screen size. This function does nothing in the system screen mode.

When the screen size is specified, the function reserves that size of area in memory and executes Clear screen & home (ØCH). The CP/M size remains unchanged even when the screen size is changed.

<Command sequence>
1st byte: ESC n: Number of screen lines
2nd byte: ØDØH 8 ≤ n ≤ 5Ø
3rd byte: n m: Number of screen columns
4th byte: m m = 4Ø or 8Ø
m * n ≤ 2Ø48

The default setting is 8Ø columns by 25 lines.

- ESC D2H Direct display
Displays the specified character in the specified position in the window.

The portion of the display data that is scrolled out of the window is deleted. The character to be displayed must be specified in character generator code. The cursor is held in the original position on the screen.

<Command sequence>
1st byte: ESC
2nd byte: ØD2H
3rd byte: Y Y: Line number 1 ≤ Y ≤ 8
4th byte: X X: Column number 1 ≤ X ≤ 4Ø
5th byte: n n: Character code ØØH ≤ n ≤ ØFFH

- ESC D4H Locate top of screen
Positions the window at the beginning of the screen.

The cursor is held in the original position on the screen.

<Command sequence>
1st byte: ESC
2nd byte: ØD4H

- ESC D5H Locate bottom of screen
Moves the window to the end of the screen.

The cursor is held in the original position on the screen.

<Command sequence>
1st byte: ESC
2nd byte: ØD5H

- ESC D6H Select cursor kind
Selects the type of the cursor.

<Command sequence>
1st byte: ESC
2nd byte: ØD6H
3rd byte: n

n: Type of the cursor
= ØØH: Block and blink
= Ø1H: Block and nonblink
= Ø2H: Underline and blink
= Ø3H: Underline and nonblink

The default is block and blink.

- ESC D7H Find cursor
Moves the window over the screen so that the cursor line will appear in the window.

This function does nothing when the cursor is already in the current window. It moves the window in the same way as Set cursor position (ESC '=').

<Command sequence>
1st byte: ESC
2nd byte: ØD7H

- ESC EØH Set download character
Defines user-defined characters with the codes ØE0H to ØFFH.

<Command sequence>
1st byte: ESC
2nd byte: ØEØH
3rd byte: n
4th byte: P(1)
5th byte: P(2)
6th byte: P(3)
7th byte: P(4)
8th byte: P(5)
9th byte: P(6)
1Øthe byte: P(7)
11th byte: P(8)

n: Character code
ØEØH ≤ n ≤ ØFFH
P(1) - P(8): Character pattern

	7	6	5	4	3	2	1	0
4			0	0	1	0	0	0
5			0	1	0	1	0	0
6			1	0	0	0	1	0
7			1	0	0	0	1	0
8			1	1	1	1	1	0
9			1	0	0	0	1	0
10			1	0	0	0	1	0
11			0	0	0	0	0	0

6 x 8 dot pattern (pattern for 'A')

- ESC FØH Keyboard repeat ON/OFF
Controls the keyboard repeat function.

<Command sequence>
1st byte: ESC n: Repeat switch status
2nd byte: ØFØH = ØØH: Repeat ON
3rd byte: n = Ø1H: Repeat OFF

The default setting is Repeat ON for the standard keyboard and Repeat OFF for the item keyboard.

- ESC F1H Set keyboard repeat starting time
Specifies the keyboard repeat starting time

<Command sequence>
1st byte: ESC S: Repeat starting time (in 1/64 sec increments)
2nd byte: ØF1H Ø ≤ S ≤ 127
3rd byte: S

The default value is approximately 656 msec (S = 42).

- ESC F2H Set keyboard repeat interval time
Specifies the keyboard auto repeat interval.

<Command sequence>
1st byte: ESC S: Repeat interval (in 1/256 sec increments)
2nd byte: ØF2H Ø ≤ S ≤ 127
3rd byte: S

The default value is approximately 7Ø msec (S = 18).

- ESC F3H Set arrow key code
Defines the arrow key codes.

When an arrow key is pressed after this function is executed, the corresponding code is returned. See Section 3.5, "Keyboard" for detailed information.

<Command sequence>
1st byte: ESC
2nd byte: ØF3H n1 - n4: Arrow key code
3rd byte: n1 ØØH ≤ n1 - n4 ≤ ØFFH
4th byte: n2 n1: Code of →
5th byte: n3 n2: Code of ←
6th byte: n4 n3: Code of ↑
  n4: Code of ↓
The default setting is n1 = 1CH, n2 = 1DH, n3 = 1EH, and n4 = 1FH.

For the item keyboard, the arrow key code cannot be changed.

- ESC F4H Set scroll key code
Defines codes for the SHIFT + arrow key combinations.

When a SHIFT/arrow key sequence is entered after this function is executed, the corresponding code is returned. See Section 3.5, "Keyboard" for detailed information.

<Command sequence>
1st byte: ESC
2nd byte: ØF4H n1 - n4: SHIFT/arrow key code
3rd byte: n1 ØØH ≤ n1 - n4 ≤ ØFFH
4th byte: n2 n1: Code of SHIFT/→
5th byte: n3 n2: Code of SHIFT/←
6th byte: n4 n3: Code of SHIFT/↑
  n4: Code of SHIFT/↓

The default settings are n1 = ØF8H, n2 = ØF9H, n3 = ØFAH, and n4 = ØFBH.

- ESC F5H Set CTRL key code

Defines codes for the CTRL + arrow key combinations.

When a CTRL key sequence is entered after this function is executed, the corresponding code is returned. See Section 3.5, "Keyboard" for details.

<Command sequence>
1st byte: ESC
2nd byte: ØF5H n1 - n4: CTRL/arrow key code
3rd byte: n1 ØØH ≤ n1 - n4 ≤ ØFFH
4th byte: n2 n1: Code of CTRL/→
5th byte: n3 n2: Code of CTRL/←
6th byte: n4 n3: Code of CTRL/↑
  n4: Code of CTRL/↓

The default settings are n1 = ØFCH, n2 = ØFDH, n3 = ØFEH, and n4 = ØFFH.

- ESC F6H Clear key buffer

Clears the keyboard buffer to delete the type ahead key codes. The data in the 7508 buffer remains unaffected.

<Command sequence>
1st byte: ESC
2nd byte: ØF6H

- ESC F7H Set key shift

Defines the key shift modes.

<Command sequence>
1st byte: ESC n: Shift mode
2nd byte: ØF7H ØØH ≤ n ≤ ØFFH
3rd byte: n

The keyboard shift modes are defined on a bit basis. A 1 in a bit turns on the shift mode for the corresponding shift key.

Bit 7 (MSB)	CTRL
6	GRPH
5	---
4	NUM
3	---
2	CAPS
1	SHIFT (L)
Ø (LSB)	SHIFT (R)

For the item keyboard, nothing takes place when a shift mode is specified.

3.6.6 Character Generator

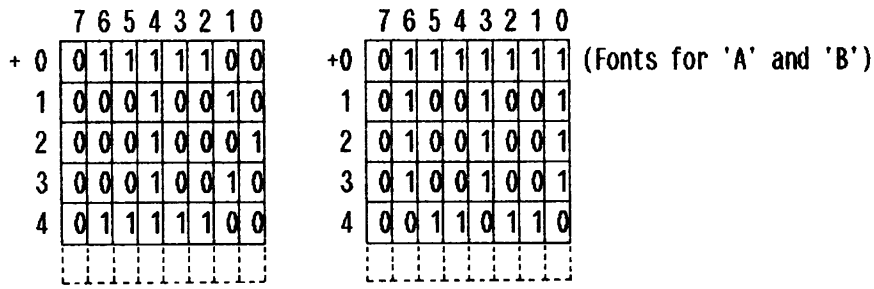
The PINE has a character generator in OS ROM.

```
ØØH - 7FH     5 x 8 dot character fonts
8ØH - 9FH     6 x 8 dot character fonts
ØAØH - ØDFH   5 x 8 dot character fonts
ØEØH - ØFFH   6 x 8 dot character fonts
```

ØEØH through ØFFH are used by the user to define user-defined characters. The fonts for the user-defined characters are stored in the user-defined character definition area. Initially, the fonts for ØEØH and ØElH are defined by the system. Other codes are initialized to spaces.

3.6.6.1 Font format

A font is made up of 5 or 6 bytes. Fonts are stored sideways as shown below:



Since characters are displayed on the LCD in 6 x 8 dot font, a ØØH code is appended to the end of each 5-byte font data to form a 6 x 8 dot font.

3.6.6.2 Character sets

The PINE supports the character sets for the languages of the following countries.

```
USA ASCII
France
Germany
England
Denmark
Sweden
Italy
Spain
Norway
```

The user can change the character set using the ESC sequence "Set character set table." The initial value is set by DIP switches. See Section 7.2, "DIP Switches" for further information.

3.6.6.3 Character generator codes arranged by country

The BIOS CONOUT routine converts character codes into character generator codes of the specified country and displays it on the LCD. The routine also places data in the user or system screen in the form of character generator codes.

Table 3.6.11 shows the relationships between the character codes and the character generator codes arranged by country.

Code	U.S.A.	France	Germany	England	Denmark	Sweden	Italy	Spain	Norway
23H	23H (#)	23H (#)	23H (#)	0FH (£)	23H (#)	23H (#)	23H (#)	18H (Pt)	23H (#)
24H	24H ($)	24H ($)	24H ($)	24H ($)	24H ($)	18H (¤)	24H ($)	24H ($)	18H (¤)
40H	40H (@)	00H (à)	03H (§)	40H (@)	40H (@)	16H (É)	40H (@)	40H (@)	16H (É)
5BH	5BH ([)	01H (°)	08H (Ä)	5BH ([)	10H (Æ)	08H (Ä)	01H (°)	1CH (¡)	10H (Æ)
5CH	5CH (\)	02H (Ç)	09H (Ö)	5CH (\)	11H (Ø)	09H (Ö)	5CH (\)	1DH (Ñ)	11H (Ø)
5DH	5DH (])	03H (§)	0AH (Ü)	5DH (])	12H (Å)	12H (Å)	04H (é)	1EH (¿)	12H (Å)
5EH	5EH (^)	5EH (^)	5EH (^)	5EH (^)	5EH (Ü)	0AH (Ü)	5EH (^)	5EH (^)	0AH (Ü)
60H	60H (`)	60H (`)	60H (`)	60H (`)	60H (`)	04H (é)	05H (ù)	60H (`)	04H (é)
7BH	7BH ({)	04H (é)	0BH (ä)	7BH ({)	13H (æ)	0BH (ä)	00H (à)	07H (¨)	13H (æ)
7CH	7CH (¦)	05H (ù)	0CH (ö)	7CH (¦)	14H (ø)	0CH (ö)	19H (ò)	1FH (ñ)	14H (ø)
7DH	7DH (})	06H (è)	0DH (ü)	7DH (})	15H (å)	15H (å)	06H (è)	7DH (})	15H (å)
7EH	7EH (~)	07H (¨)	0EH (ß)	7EH (~)	7EH (~)	0DH (ü)	1AH (ì)	7EH (~)	0DH (ü)

Table 3.6.11 Relationships between ASCII Codes and Character Generator Codes

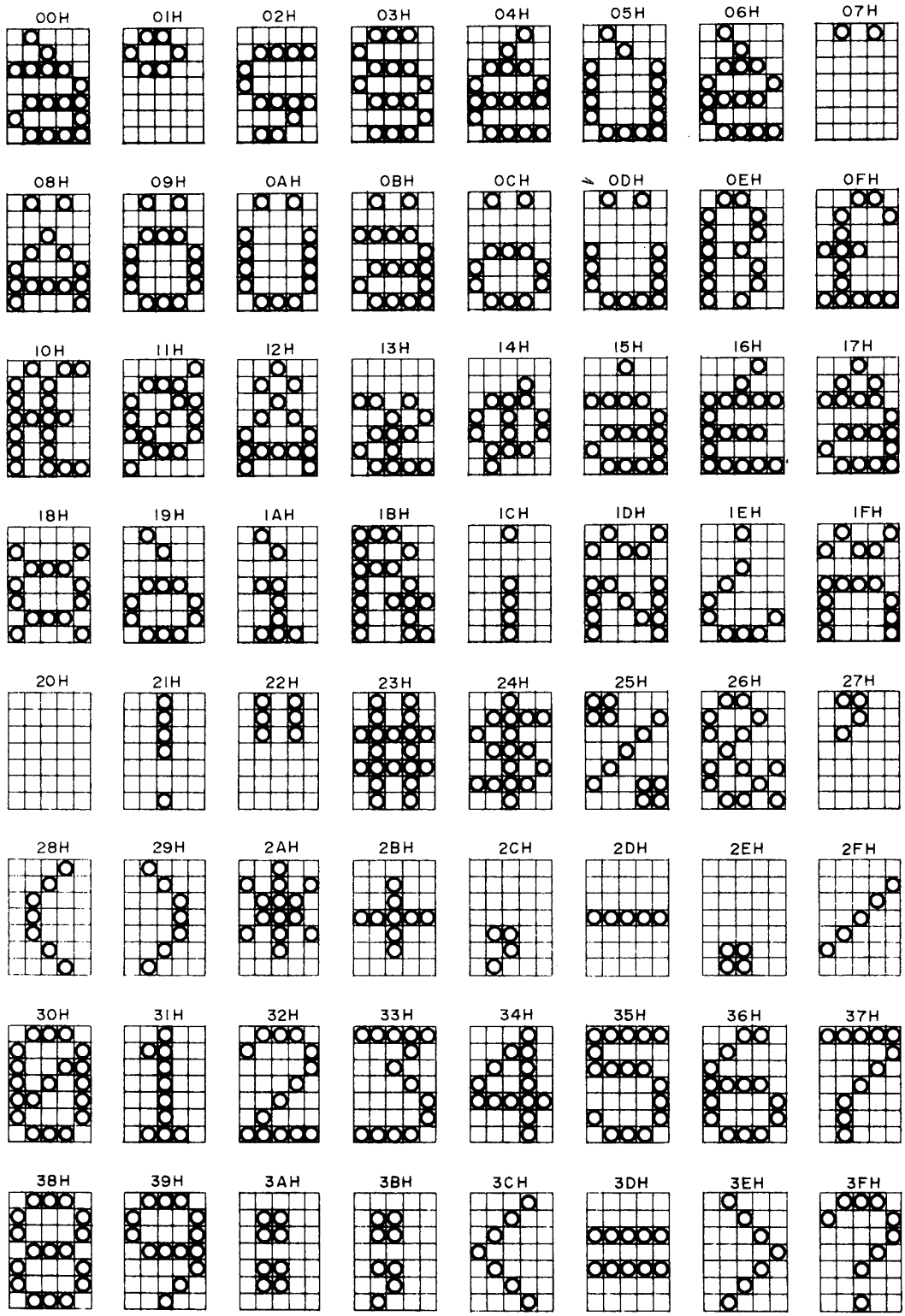
3.6.6.4 Character generator code charts

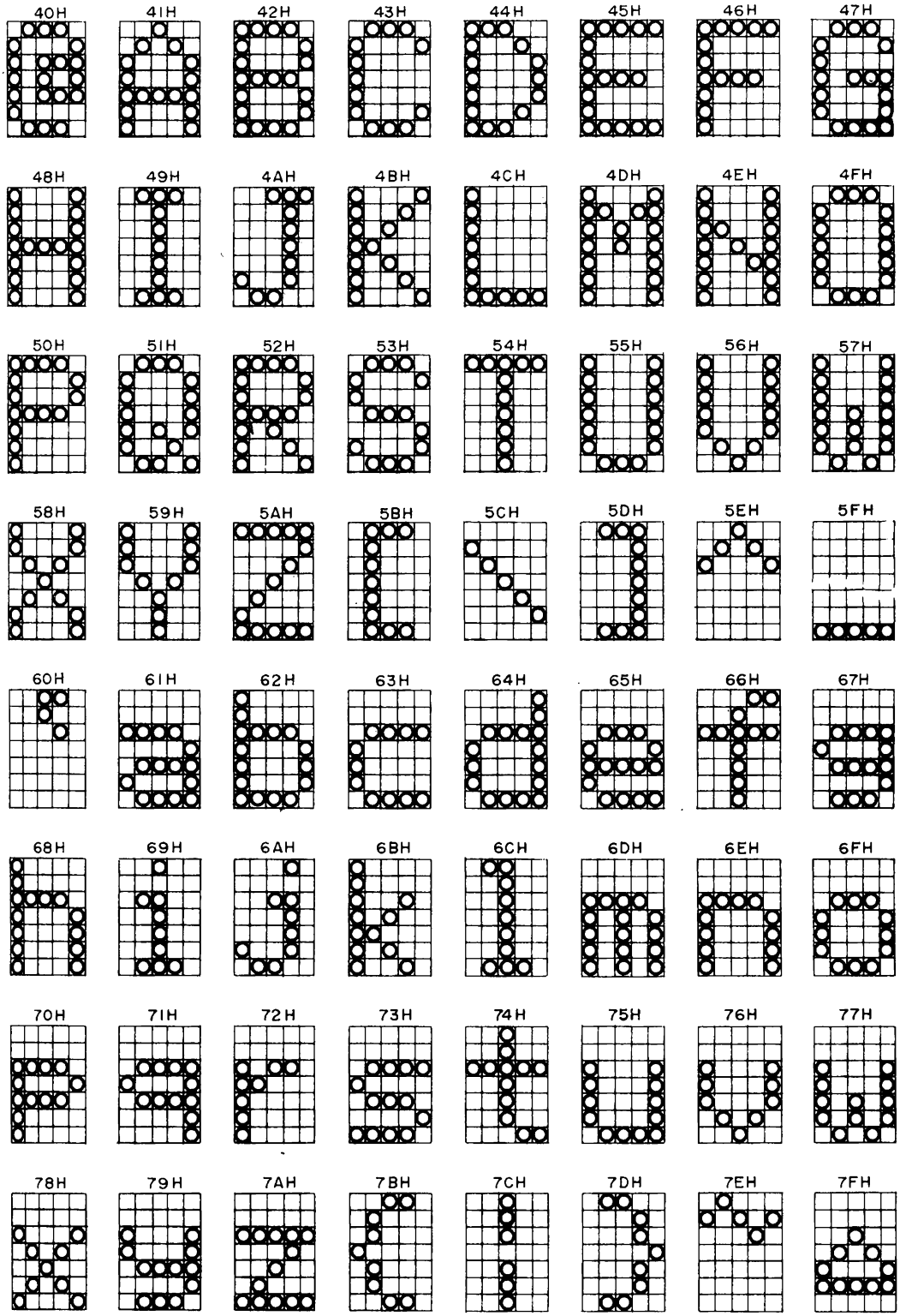
The pages that follow list charts of the character generator data (fonts).

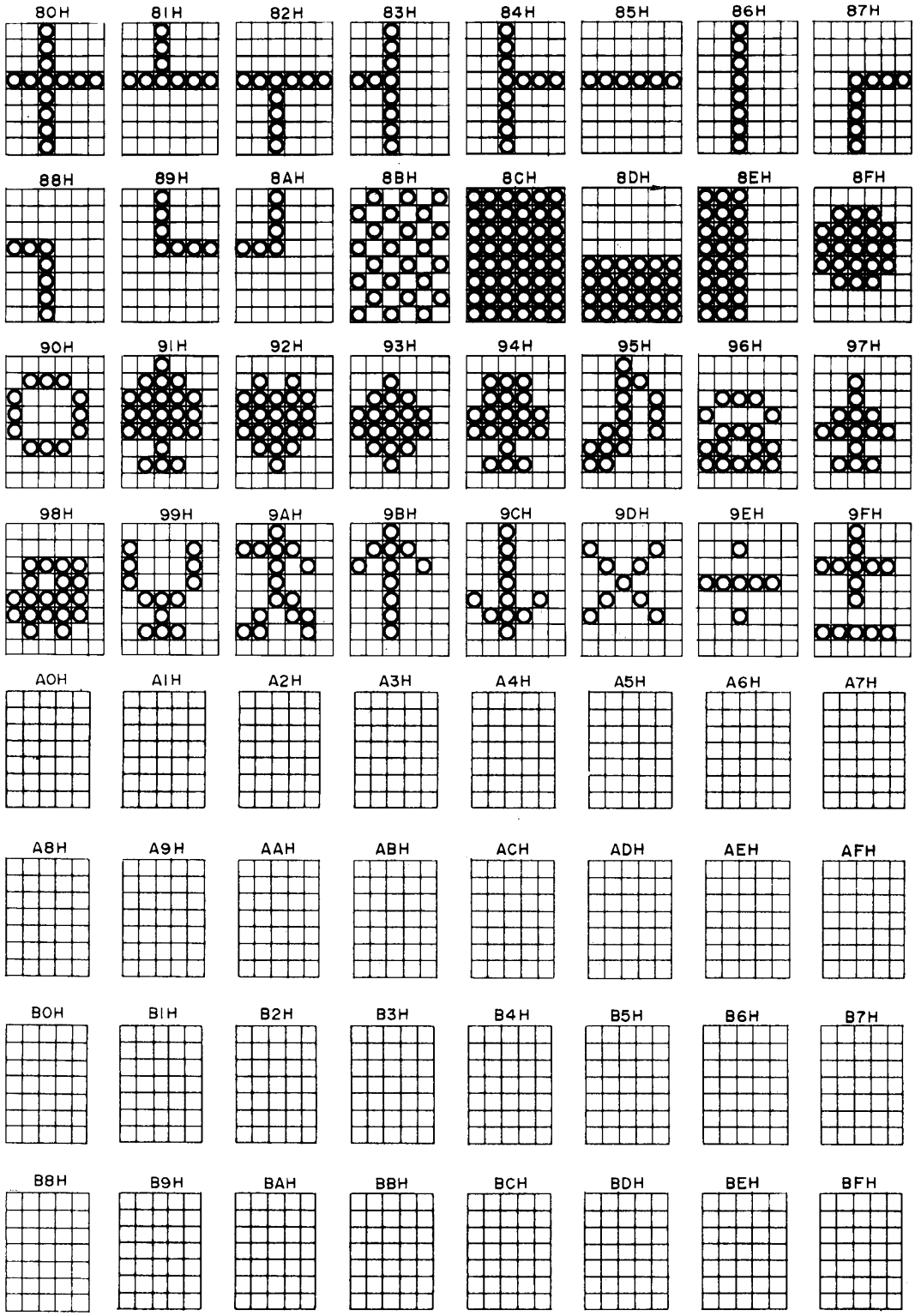
Font data can be read using the OS utility XFONTGET. For more information, see Section 4.2, "Jump Tables."

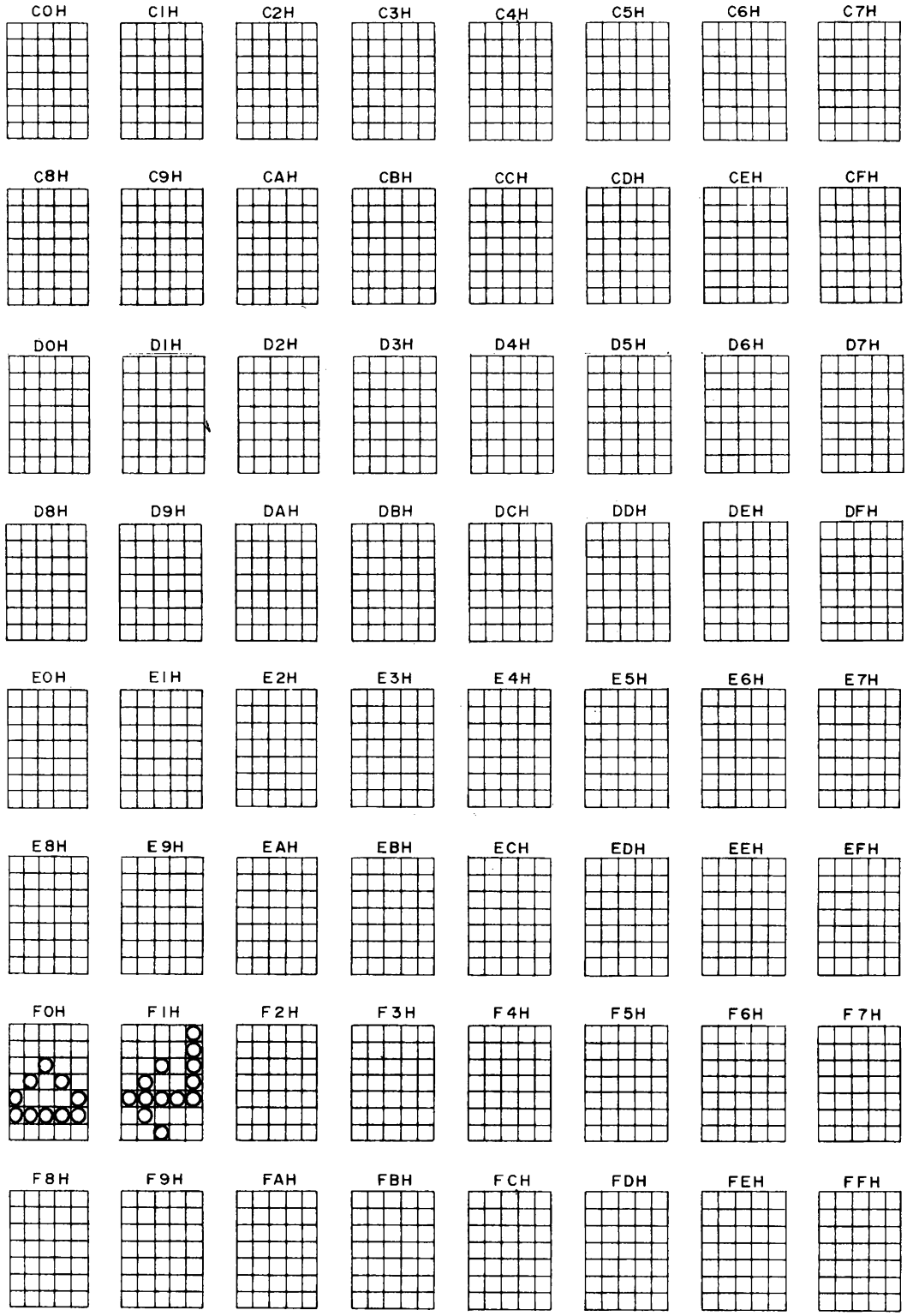
Character generator code chart

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	à	Æ	sp	0	@	P	`	p	┼	○					△	
1	°	Ø	!	1	A	Q	a	q	─	♠					↵	
2	ç	Å	”	2	B	R	b	r	┬	♥						
3	§	æ	#	3	C	S	c	s	┤	♦						
4	é	ø	$	4	D	T	d	t	┘	♣						
5	ù	å	%	5	E	U	e	u	├	♪						
6	è	É	&	6	F	V	f	v	│	☎						
7	¨	â	’	7	G	W	g	w	┌	▲						
8	Ä	☒	(	8	H	X	h	x	┐	🚗						
9	Ö	ò	)	9	I	Y	i	y	└	🏆						
A	Ü	ì	*	:	J	Z	j	z	┴	🚶						
B	ä	Pt	+	;	K	[	k	{	▒	↑						
C	ö	ì	,	<	L	\	l	:	■	↓						
D	ü	Ñ	–	=	M	]	m	}	▬	×						
E	ß	¿	.	>	N	^	n	~	▪	÷						
F	£	ñ	/	?	O	–	o	△	●	±						


00H
01H
02H
03H
04H
05H
06H
07H
08H
09H
0AH
0BH
0CH
0DH
0EH
0FH
10H
11H
12H
13H
14H
15H
16H
17H
18H
19H
1AH
1BH
1CH
1DH
1EH
1FH
20H
21H
22H
23H
24H
25H
26H
27H
28H
29H
2AH
2BH
2CH
2DH
2EH
2FH
30H
31H
32H
33H
34H
35H
36H
37H
38H
39H
3AH
3BH
3CH
3DH
3EH
3FH


40H
41H
42H
43H
44H
45H
46H
47H
48H
49H
4AH
4BH
4CH
4DH
4EH
4FH
50H
51H
52H
53H
54H
55H
56H
57H
58H
59H
5AH
5BH
5CH
5DH
5EH
5FH
60H
61H
62H
63H
64H
65H
66H
67H
68H
69H
6AH
6BH
6CH
6DH
6EH
6FH
70H
71H
72H
73H
74H
75H
76H
77H
78H
79H
7AH
7BH
7CH
7DH
7EH
7FH


80H
81H
82H
83H
84H
85H
86H
87H
88H
89H
8AH
8BH
8CH
8DH
8EH
8FH
90H
91H
92H
93H
94H
95H
96H
97H
98H
99H
9AH
9BH
9CH
9DH
9EH
9FH
A0H
A1H
A2H
A3H
A4H
A5H
A6H
A7H
A8H
A9H
AAH
ABH
ACH
ADH
AEH
AFH
B0H
B1H
B2H
B3H
B4H
B5H
B6H
B7H
B8H
B9H
BAH
BBH
BCH
BDH
BEH
BFH


C0H
C1H
C2H
C3H
C4H
C5H
C6H
C7H
C8H
C9H
CAH
CBH
CCH
CDH
CEH
CFH
D0H
D1H
D2H
D3H
D4H
D5H
D6H
D7H
D8H
D9H
DAH
DBH
DCH
DDH
DEH
DFH
E0H
E1H
E2H
E3H
E4H
E5H
E6H
E7H
E8H
E9H
EAH
EBH
ECH
EDH
EEH
EFH
F0H
F1H
F2H
F3H
F4H
F5H
F6H
F7H
F8H
F9H
FAH
FBH
FCH
FDH
FEH
FFH

3.6.7 Miscellaneous Considerations

3.6.7.1 Initializing screen-related parameters

The screen-related parameters include the following:

(1) User screen size
(2) Window scroll mode
(3) Cursor display on/off
(4) Cursor type

These parameters are initialized at reset and warm boot times.

The timing of parameter initialization is illustrated in Figure 3.6.12.

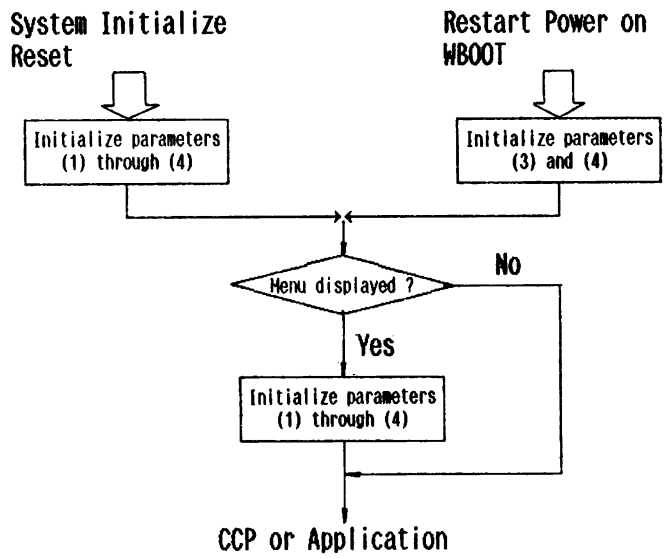


Fig. 3.6.12 Parameter Initialization Timing Diagram

The screen-related parameters are reserved in the system area in such a format that they can be directly set up using the CONOUT BIOS function.

CONSCRN1 (ØEFF3H) 8 bytes
- Screen parameter 1
This area is used to specify the user screen size and the window scroll mode.

1st byte: ESC (1BH)
2nd byte: ØDØH
3rd byte: Number of screen lines
4th byte: Number of screen columns
5th byte: ESC (1BH)
6th byte: 95H
7th byte: Scroll mode
8th byte: ØØH

The first four bytes specify the screen size. The initial value is 8Ø columns by 25 lines.

The fifth to seventh bytes specify the window scroll mode. The initial value is horizontally nontracking mode.

See the descriptions of the ESC sequences for details.

CONSCRN2 (ØEFFBH) 6 bytes
- Screen parameter 2
This area is used to specify cursor display on/off and the cursor type.

1st byte: ESC (1BH)
2nd byte: Cursor display on/off
3rd byte: ESC (1BH)
4th byte: ØD6H
5th byte: Cursor type
6th byte: ØFFH

The first two bytes specify whether the cursor is to be turned on or off. The initial value is cursor display on (33H, '3').

The third to fifth bytes specify the cursor type. The initial value is block and blink.

ØFFH identifies the end of the parameters.

3.6.7.2 Cursor display during execution of BIOS CONOUT

When BIOS CONOUT receives a command such as displaying a character, moving the screen, or changing the cursor mode, it turns off the cursor before processing the command and turns it on after completing the execution of the command.

Consequently, the cursor would flicker if such a command is controlled by CONOUT. This can be avoided in two ways:
- Turns off the cursor while CONOUT is processing.
- Obtain the address of the subroutine in CONOUT that actually processes the command and call that subroutine directly.

(1) Turning off the cursor
Turn off the cursor using ESC + '2' before displaying a character or moving the screen.

(2) Obtaining the actual command handling subroutine address
Execute the pertinent command using CONOUT and look into the function address area LFKADDR (ØF2A8H) to get the address of the actual command processing subroutine.

Once the address is obtained, cursor turning on/off processing can be bypassed by directly calling the address in OS ROM via BIOS CALLX.

Note that the command processing subroutine does not control the cursor. That is, the cursor does not move forward as characters are displayed on the screen. (Use this technique when displaying characters in one part of the screen while having the cursor blink in another part of the screen.)

In practice, direct execution is carried out by calling OS ROM procedures as explained below.
(a) When an ESC sequence has parameters (when including ESC there are more than 3 bytes of data), LESCPRM (F2ACH) is called after setting the parameters. Furthermore, at this time the last parameter is put into the C register.
(b) When an ESC sequence does not have parameters (for example ESC + 'P'), it is sufficient to simply make a call.
(c) To display a character (20H - FFH), the call is made after putting the parameter into the C register.
(d) In the case of a control codes (ØØH - 1FH), it is sufficient to simply make a call.

3.6.7.3 Multiple Use of Virtual Screens

As mentioned earlier, in principle it is possible to use only two types of screen, the user screen and system screen, however for simple applications it is also possible to use a number of additional screens.

Here we will explain methods for multiple use of virtual screens, and some points to note.

(1) Concepts

The operating system maps a window in the virtual screen onto the LCD. Accordingly, it is possible to use different virtual screens within an application program, by providing buffers for use by the virtual screens, and specifying the buffers with the system area display parameters. In Figure 3.6.12 mapping is executed by the operating system, and switching is controlled by the application program.

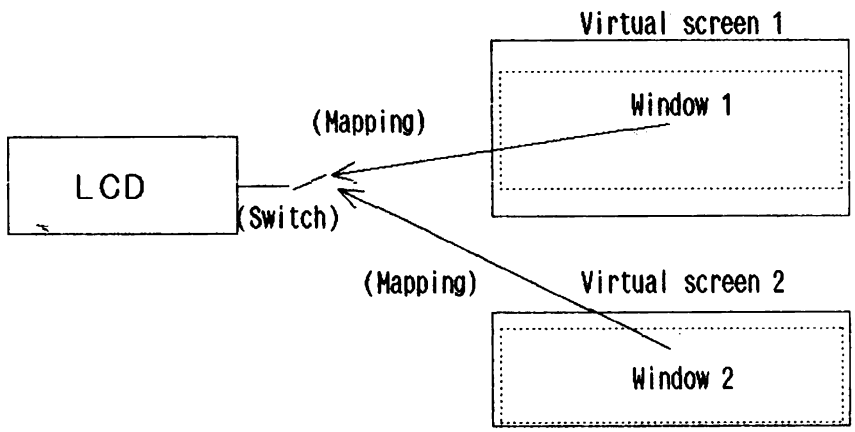


Fig. 3.6.12 Multiple Use of Virtual Screens

(2) Procedure

For multiple use of virtual screens, the following two methods are available.

(a) Multiple use of virtual screens only.
(b) Multiple use of virtual screens and VRAM.

The main difference between (a) and (b) above, is whether or not graphic data and inverse displays etc. can be preserved when the screen is switched.

(a) Multiple use of virtual screens only.
This procedure is used when only character data is to be used.
The procedure is illustrated in Figure 3.6.13.

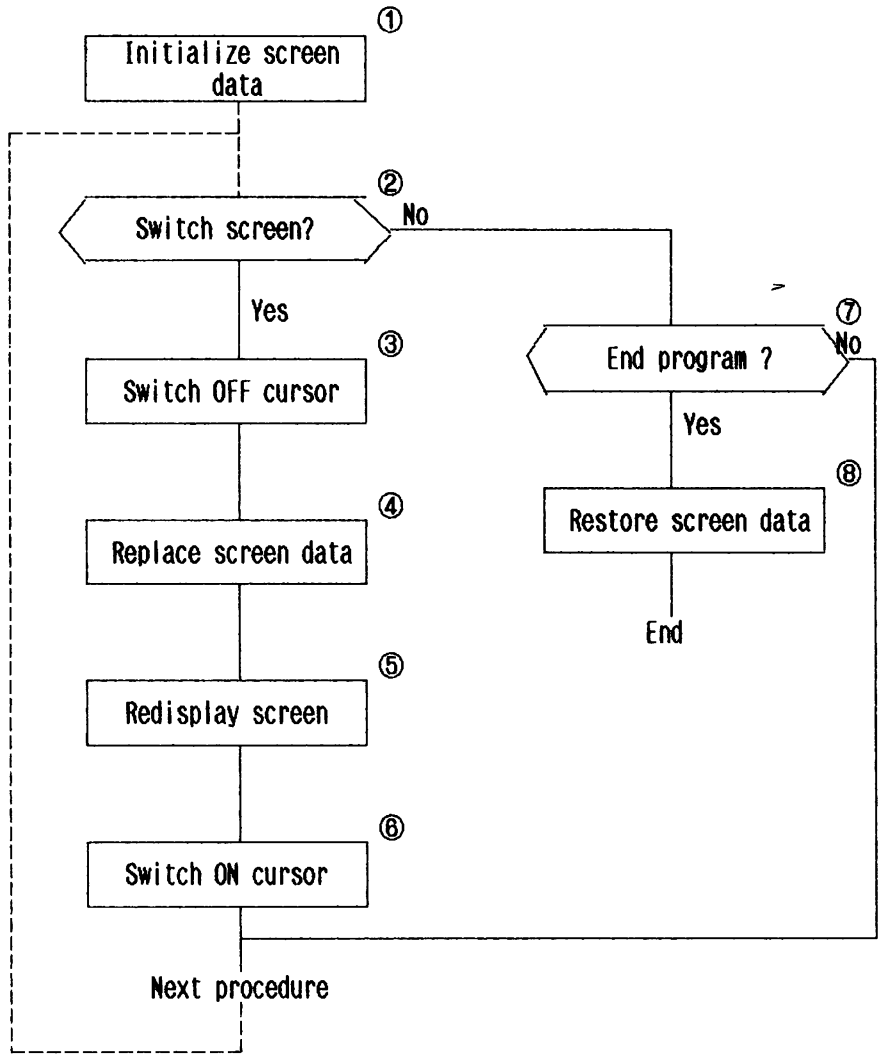


Fig. 3.6.13 Procedure for Switching Screen

(b) Multiple use of virtual screens and VRAM.
This procedure is used when character data and graphic data are used mixed together.
The procedure is illustrated in Figure 3.6.14

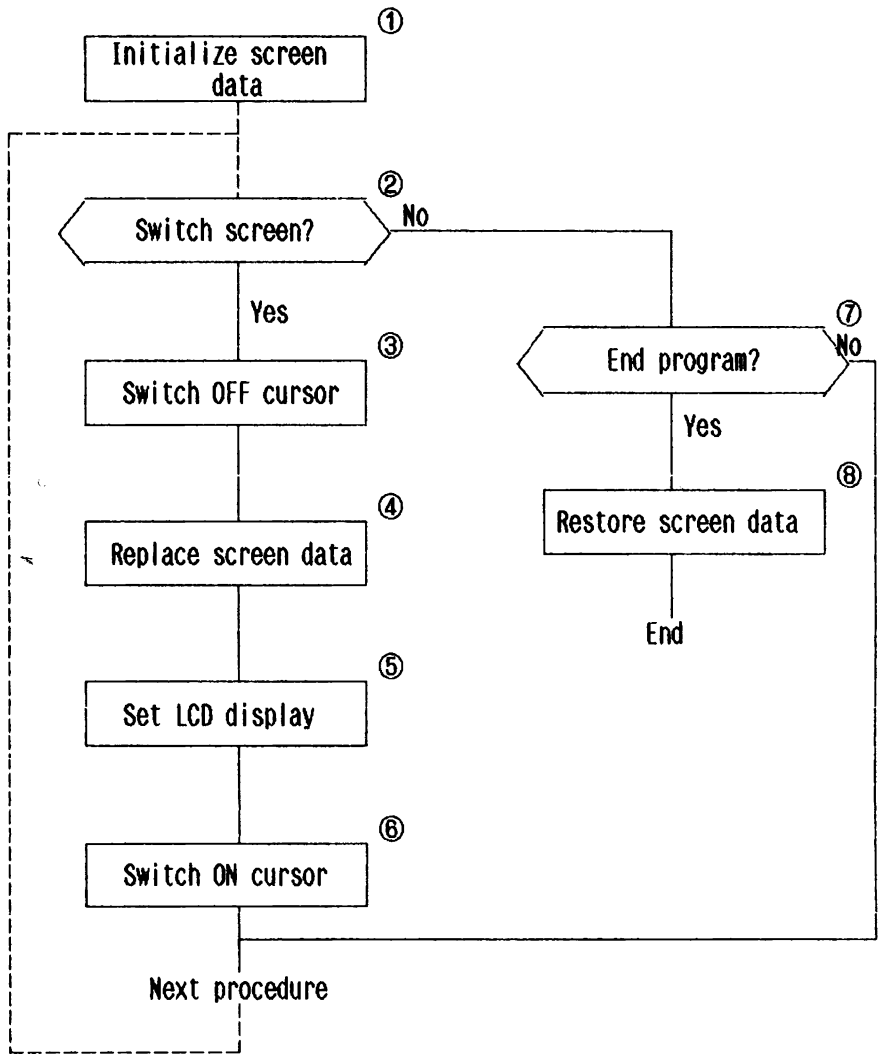


Fig. 3.6.14 Procedure for Switching VRAM

Step	Process	Content
1	Initialize screen data	* When use is to be made of VRAM or virtual screen areas other than areas previously prepared by the system, the areas to be used are initialized by this process beforehand. * The data area for replacement of system area display parameters is initialized beforehand. The display parameters are 39 bytes of data from LSCADDR(F290H) to LESCPRM(F2B6H).
2	Switch screen ?	* Tests if condition requiring screen mode to be changed is satisfied or not.
3	Switch OFF cursor	* A display screen operation is to be carried out, so this process switches OFF the cursor beforehand.
4	Replace screen data	* Replaces system area display parameters(39 bytes from F290H) with previously prepared data.
5	Redisplay screen	* In the case of virtual screen replacement only, a call is made to XREDSP (redisplay) in the OS jump table.. * In the case of VRAM replacemt, the VRAM address and offset value in the Y direction are output through a port.
6	Switch ON cursor	* Switches ON the cursor when the display screen operation has been completed. * The cursor is switched ON using the BIOS CONOUT.

Step	Process	Content
7	End program ?	* Tests if application program should finish or not.
8	Restore screen data	* When the application program is terminated, LSCADDR(F290H) is restored to D000H and LSRVRAM (F294H) is restored to E000H. * Termination of the application program with the virtual screen and VRAM still in the state set by the application program must be avoided.

(3) Points to Note
When processing involving switching of screens is carried out, attention must be given to the following points.

(a) In cases in which the position of the virtual screen or position of the VRAM is changed, care must be taken not to input Restart/Power off.
This is to avoid misoperation when the application program which will next be executed modifies screen size etc.
During switching of screens, it is necessary for either Continue mode to have been previously set, or for Power off to have been forbidden at the application level.

(b) When the position of the virtual screen or position of the VRAM is changed, RAM locations must be reserved from 8000H upward.
Also, the leading VRAM address must be specified on an 800H (2k) byte boundary.

3.6.8 Screen-related Work Areas

DSPFLAG (ØEFB6H) 1 byte
- LCD display flag
    = ØØH: LCD display off
    = 8ØH: LCD display on

BLNKSTAT (ØEFB7H) 1 byte
- Blink state flag
    = ØØH: Blink disabled
    = 8ØH: Blink enabled

This flag is examined by the blink processing routine.

BLNKCNT (ØEFB8H) 1 byte
- Blink counter

This counter is incremented by one when an overflow interrupt is generated. The cursor is displayed in reverse video when BLNKTIME is reached.

BLNKRVRS (ØEFB9H) 1 byte
- Indicates the cursor status during blinking.
    = ØØH: Cursor display on
    = ØFFH: Cursor display off

BLNKTIME (ØEFBAH) 1 byte
- Specifies the blink interval time.

The initial value is Ø4H. With this setting, cursor display is switched between normal and reverse video every 5ØØ ms or so.

LSCADDR (ØF29ØH) 2 bytes
- Starting address of the buffer for the currently displayed screen

The screen buffer size is stored in LSCSIZE.

    8ØØØH ≤ LSCADDR ≤ ØFFFFH

LSCSIZE (ØF292H) 2 bytes
- Screen buffer size

    Ø ≤ LSCSIZE ≤ 2Ø48

LSCSIZE is equal to LSCSIZEX * LSCSIZEY.

LSCRVRAM (ØF294H) 2 bytes
- Starting address of the currently displayed VRAM

    8ØØØH ≤ LSCRVRAM ≤ ØFFFFH

2K bytes of memory must be allocated for VRAM.

LCURSOR (ØF296H) 2 bytes
- Cursor state flag

    Bit 7: Cursor mode
        Ø: Display on 1: Display off
    Bits 6 - 2: Don't care.
    Bit 1: Cursor type
        Ø: Block 1: Underline
    Bit Ø: Cursor blink
        Ø: On 1: Off

LCRVRSW (ØF297H) 1 byte
- Reverse display flag
     Bits 7 - 1: Don't care.
     Bit Ø: Reverse state
            Ø: Reverse display mode off
            1: Reverse display mode on

LSCCPOSX (ØF298H) 1 byte
- X-coordinate of the cursor on the screen
     1 ≤ X ≤ Number of columns on the screen

LSCCPOSY (ØF299H) 1 byte
- Y-coordinate of the cursor on the screen
     1 ≤ Y ≤ Number of lines on the screen

LSCSIZEX (ØF29AH) 1 byte
- Screen width
     LSCSIZEX = 4Ø or 8Ø

LSCSIZEY (ØF29BH) 1 byte
- Screen length
     8 ≤ LSCSIZEY ≤ 5Ø

LWDXMIN (ØF29CH) 1 byte
- X-coordinate of the upper left corner of the window
This variable indicates where the current window is located on the screen.

LWDYMIN (ØF29DH) 1 byte
- Y-coordinate of the upper left corner of the window
This variable indicates where the current window is located on the screen.

LWDCPOSX (ØF29EH) 1 byte
- X-coordinate of the cursor on the screen
     = ØFDH: Inside the right margin
a   = ØFEH: Inside the left margin
     = ØFFH: Outside the window
     Others: Inside the window (Ø ≤ X ≤ 39)

LWDCPOSY (ØF29EH) 1 byte
- Y-coordinate of the cursor on the screen
     = ØFFH: Outside the window
     Others: Inside the window (Ø ≤ Y ≤ 7)

LVRAMYOF (ØF2AØH) 1 byte
- Y-direction offset of VRAM
     Ø ≤ LVRAMYOF ≤ 63

LWDTYPE (ØF2A1H) 1 byte
- Window type
     = ØFEH: Right screen (scroll step = 4Ø)
     = ØFFH: Left screen (scroll step = 4Ø)
     = ØØH: Screen width is 4Ø
     = Ø1H: Left screen (scroll step = 2Ø)
     = Ø2H: Center screen (scroll step = 2Ø)
     = Ø3H: Right screen (scroll step = 2Ø)

LSECRETW (ØF2A2H) 1 byte
- Secret mode on/off state
     = ØØH: Secret mode off
     = Ø1H: Secret mode on

LSCROLMD (ØF2A3H) 1 byte
- Scroll mode
    = ØØH: Tracking mode
    = Ø1H: Nontracking mode
    = Ø2H: Horizontally nontracking mode

LSCROLX (ØF2A4H) 1 byte
- Horizontal scroll step
    LSCROLX = 2Ø or 4Ø

LSCROLY (ØF2A5H) 1 byte
- Vertical scroll step
    1 ≤ LSCROLY ≤ 8

LCRWAIT (ØF2A6H) 1 byte
- Carriage return check flag
    = ØØH: Carriage return check mode off
    = Ø1H: Carriage return check mode on
The carriage return check mode is turned on when the cursor moves from the end of a line to the beginning of the next line after a character is displayed.

LFKSTAT (ØF2A7H) 1 byte
- CONOUT function status
    Bit 7: Cursor off flag
        Ø: Cursor off not required 1: Cursor off required
    Bit 6: Indicates whether the CR check flag is to be cleared.
        Ø: Not cleared 1: Cleared
    Bits 5, 4: Don't care
    Bits 3 - 1: Number of function parameters

LFKADDR (ØF2A8H) 2 bytes
- Address of a CONOUT function processing routine
When a CONOUT function is specified, this area is loaded with the address of its processing routine. Control is transferred to this address after all parameters have been received.

LESCFLG (ØF2AAH) 1 byte
- ESC sequence receive flag
    = ØØH: ESC code not received
    = Ø1H: ESC code received
    = Ø2H: ESC code and function received

LESCCNT (ØF2ABH) 1 byte
- Number of ESC sequence parameters received.

LESCPRM (ØF2ACH) 11 bytes
- The area for storing ESC sequence receive parameters.

LWORKBF (ØF2B7H) 39 bytes
- The area for holding the screen parameters saved when the screen mode is changed.
When the screen mode is changed, the 39 bytes of data from LSCADDR to LESCPRM are exchanged with the 39 bytes of data in this area.

LSCMODE (ØF2DEH) 1 byte
- Screen mode
    = ØØH: System screen mode
    = Ø1H: User screen mode

LLMARGIN (ØF2DFH) 1 byte
- Left margin size
    Ø ≤ LLMARGIN ≤ 1Ø

LRMARGIN (ØF2EØH) 1 byte
- Right margin size
    Ø ≤ LRMARGIN ≤ 1Ø

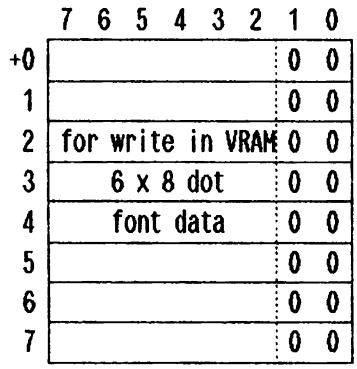
RLCGENX (ØF35CH) 2 bytes
- Reserved for system

RLCGENN (ØF35EH) 2 bytes
- Reserved for system

RLCGENG (ØF36ØH) 2 bytes
- Reserved for system

RLCGENK (ØF362H) 2 bytes
- Reserved for system

LCHRFONT (ØF7C6H) 8 bytes
- Character font data conversion area



RWUVSCTOP (ØDØØØH) 8ØØH bytes
- User screen buffer
Loaded with character codes.

RWVRAM2TOP (ØD8ØØH) 8ØØH bytes
- VRAM2 area
VRAM for the system screen.

RWVRAM1TOP (ØEØØØH) 8ØØH bytes
- VRAM 1 area
VRAM for the user screen.

RWSYSCTOP (ØE8ØØH) 24Ø bytes
- System screen buffer
Loaded with character codes.

RWEXCHRTOP (ØE94ØH) 192 bytes
- User-defined character area
This area can contain up to 32 user-defined characters in 6 x 8 dot font.

```
                        ;       ************************************************
                        ;               CHANGE SCREEN AREA PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is changing screen
                        ;               area.
                        ;               So, you can use screen more than only one.
                        ;
                        ;       <> assemble condition <>
                        ;
                                .Z80
                        ;
                        ;       <> loading address <>
                        ;
                                .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
                        ;       BIOS entry address.
                        ;
EB03                    WBOOT           EQU     0EB03H
EB09                    CONIN           EQU     WBOOT   +06H
EB0C                    CONOUT          EQU     WBOOT   +09H
EB69                    CALLX           EQU     WBOOT   +66H
                        ;
                        ;       System area
                        ;
F290                    LSCADDR         EQU     0F290H  ; Screen buffer top addr.
F294                    LSCRVRAM        EQU     0F294H  ; VRAM area top addr.
F2A0                    LVRAMYOF        EQU     0F2A0H  ; VRAM Y-offset value.
EF94                    TOPRAM          EQU     0EF94H  ; User BIOS area top addr.
                        ;
                        ;       OS ROM jump table
                        ;
0036                    XREDSP          EQU     00036H  ; Re-display window
                        ;
                        ;
                        ;
001B                    ESC             EQU     1BH     ; ESC code
0003                    STOP            EQU     03H     ; STOP code
0000                    HELP            EQU     00H     ; HELP code
                        ;
                        ;
C000                    VADDR           EQU     0C000H  ; New VRAM address.
                        ;
                        ;       IO register address
                        ;
0009                    ZYOFF           EQU     09H     ;
0008                    ZVADR           EQU     08H     ;
                        ;
                        ;
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE ;
                        ;               This routine sets new screen data.
                        ;
                        ;       CAUTION :
                        ;               This program uses User BIOS area for VRAM.
                        ;               But this routine doesn't check that
                        ;               other program already User BIOS area.
                        ;
                        ;               If you stop program, you must restore
                        ;               old screen status.
                        ;               If you forget this, other system area
                        ;               will be destroyed.
                        ;
0100                    START:
0100    31 1000                 LD      SP,1000H        ; Set stack pointer.
                        ;
                        ;       Set initial data
                        ;
0103    2A EF94                 LD      HL,(TOPRAM)     ; User BIOS area check.
0106    11 C001                 LD      DE,0C001H       ; User BIOS area top addr <= C000H?
0109    AF                      XOR     A               ;
010A    ED 52                   SBC     HL,DE           ;
010C    D2 EB03                 JP      NC,WBOOT        ; No. then WBOOT
                        ;
010F    01 3228                 LD      BC,50*256+40    ; Set default screen size
0112    CD 01AF                 CALL    SETSCR          ;
                        ;
0115    21 F290                 LD      HL,LSCADDR      ; Save current screen data.
0118    11 0224                 LD      DE,SCRSAVE      ;
011B    01 0027                 LD      BC,27H          ;
011E    ED B0                   LDIR                    ;
                        ;
0120    01 2228                 LD      BC,34*256+40    ; Set new screen size.
0123    CD 01AF                 CALL    SETSCR          ;
                        ;
0126    21 C000                 LD      HL,VADDR        ; Clear new VRAM area
0129    36 00                   LD      (HL),0          ;
012B    54                      LD      D,H             ;
012C    5D                      LD      E,L             ;
012D    13                      INC     DE              ;
012E    01 07FF                 LD      BC,2048-1       ;
0131    ED B0                   LDIR                    ;
                        ;
                        ;       Main loop
                        ;
0133                    LOOP:
0133    CD EB09                 CALL    CONIN           ; Get inputed key code.
0136    FE 03                   CP      STOP            ; If STOP,
0138    CA 01C5                 JP      Z,PEND          ;  then end
013B    FE 1B                   CP      ESC             ; If ESC,
```

```
013D  28 0A                   JR      Z,CHNGSCR       ;  then change screen.
013F  FE 00                   CP      HELP            ; If HELP,
0141  28 2E                   JR      Z,CHNGVRAM      ;  then change VRAM.
0143  4F                      LD      C,A             ; Console out inpued data.
0144  CD EB0C                 CALL    CONOUT          ;
0147  18 EA                   JR      LOOP            ; Loop
0149                        CHNGSCR:
0149  0E 1B                   LD      C,ESC           ; Erase cursor
014B  CD EB0C                 CALL    CONOUT          ;
014E  0E 32                   LD      C,'2'           ;
0150  CD EB0C                 CALL    CONOUT          ;
                            ;
0153  21 01D6                 LD      HL,WORKBF1      ; Source addr.
0156  CD 0190                 CALL    WKCHNG          ; Change screen
                            ;
0159  3E FF                   LD      A,0FFH          ; Set destination bank
015B  32 F52E                 LD      (0F52EH),A      ;
015E  DD 21 0036              LD      IX,XREDSP       ; Re-display window
0162  CD EB69                 CALL    CALLX           ;
                            ;
0165  0E 1B                   LD      C,ESC           ; Cursor on.
0167  CD EB0C                 CALL    CONOUT          ;
016A  0E 33                   LD      C,'3'           ;
016C  CD EB0C                 CALL    CONOUT          ;
016F  18 C2                   JR      LOOP            ;
                            ;
                            ;
0171                        CHNGVRAM:
0171  0E 1B                   LD      C,ESC           ;Erase cursor
0173  CD EB0C                 CALL    CONOUT
0176  0E 32                   LD      C,'2'
0178  CD EB0C                 CALL    CONOUT
                            ;
017B  21 01FD                 LD      HL,WORKBF2      ;Source addr.
017E  CD 0190                 CALL    WKCHNG          ;Change screen
                            ;
0181  CD 019F                 CALL    SETVRAM         ;VRAM data set.
                            ;
0184  0E 1B                   LD      C,ESC           ;Cursor on.
0186  CD EB0C                 CALL    CONOUT
0189  0E 33                   LD      C,'3'
018B  CD EB0C                 CALL    CONOUT
018E  18 A3                   JR      LOOP
                            ;
                            ;
0190                        WKCHNG:
0190  11 F290                 LD      DE,LSCADDR      ;Destination addr.
0193  06 27                   LD      B,27H           ;Exchange byte no.
                            ;
0195                        WKC10:
0195  4E                      LD      C,(HL)          ;Exchange data
0196  1A                      LD      A,(DE)          ; (DE) <--> (HL)
0197  77                      LD      (HL),A
0198  79                      LD      A,C
0199  12                      LD      (DE),A
                            ;
019A  23                      INC     HL              ;Pointer update
019B  13                      INC     DE
019C  10 F7                   DJNZ    WKC10           ;Loop until b=0
                            ;
019E  C9                      RET
                            ;
                            ;
019F                        SETVRAM:
019F  AF                      XOR     A               ;Display off
01A0  D3 09                   OUT     (ZYOFF),A
01A2  3A F295                 LD      A,(LSCRVRAM+1)  ;Set VRAM addr.
01A5  D3 08                   OUT     (ZVADR),A
01A7  3A F2A0                 LD      A,(LVRAMYOF)    ;Set Y-offset
01AA  F6 80                   OR      10000000B
01AC  D3 09                   OUT     (ZYOFF),A
01AE  C9                      RET
                            ;       ************************************************
                            ;               CHANGE SCREEN SIZE ROUTINE
                            ;       ************************************************
                            ;
                            ;       NOTE :
                            ;               This routine is changing screen size.
                            ;
                            ;       <> entry parameter <>
                            ;               C    : New screen size for vertical
                            ;               B    : New screen size for horizontal
                            ;       <> return parameter <>
                            ;               NON
                            ;       <> preserved registers <>
                            ;               NON
                            ;
                            ;       CAUTION :
                            ;
01AF                        SETSCR:
01AF  C5                      PUSH    BC              ; Change screen size.
01B0  0E 1B                   LD      C,ESC           ;
01B2  CD EB0C                 CALL    CONOUT          ;
01B5  0E D0                   LD      C,0D0H          ;
01B7  CD EB0C                 CALL    CONOUT          ;
01BA  C1                      POP     BC              ;
01BB  C5                      PUSH    BC              ;
01BC  48                      LD      C,B             ; Size of Y
01BD  CD EB0C                 CALL    CONOUT          ;
01C0  C1                      POP     BC              ; Size of X
01C1  CD EB0C                 CALL    CONOUT          ;
01C4  C9                      RET                     ;
                            ;
                            ;
01C5                        PEND:
01C5  21 0224                 LD      HL,SCRSAVE      ;Restore screen data
```

```
01C8    CD 0190                         CALL    WKCHNG
                                ;
01CB    0E 0C                           LD      C,0CH           ;Clear screen
01CD    CD E80C                         CALL    CONOUT
                                ;
01D0    CD 019F                         CALL    SETVRAM
01D3    C3 0000                         JP      0000H           ;WBOOT
                                ;
                                ;
01D6                            WORKBF1:
01D6    D550                            DW      0D000H+40*34    ;Screen addr.
01D8    0140                            DW      40*8            ;Screen size
01DA    E000                            DW      0E000H          ;VRAM top addr
01DC    00                              DB      0               ;Cursor status
01DD    00                              DB      0               ;Reverse status
01DE    0101                            DW      0101H           ;Cursor position in screen
01E0    28                              DB      40              ;Screen size X
01E1    08                              DB      8               ;Screen size Y
01E2    0101                            DW      0101H           ;Window left-upper position
01E4    0000                            DW      0000H           ;Cursor position in window
01E6    00                              DB      0               ;VRAM Y-offset
01E7    00                              DB      0               ;Window type
01E8    00                              DB      0               ;Secret mode
01E9    00                              DB      0               ;Scroll mode
01EA    0000                            DW      0               ;Scroll step
01EC    00                              DB      0               ;Carriage return wait flag
01ED    00                              DB      0               ;Function status
01EE    0000                            DW      0               ;Function addr
01F0    00                              DB      0               ;ESC flag
01F1    00                              DB      0               ;ESC count
01F2    00 00 00 00                     DB      0,0,0,0         ;Parameter store area
01F6    00 00 00 00                     DB      0,0,0,0
01FA    00 00 00                        DB      0,0,0
                                ;
01FD                            WORKBF2:
01FD    D690                            DW      0D000H+40*42    ;Screen addr.
01FF    0140                            DW      40*8            ;Screen size
0201    C000                            DW      VADDR           ;VRAM top addr
0203    00                              DB      0               ;Cursor status
0204    00                              DB      0               ;Reverse status
0205    0101                            DW      0101H           ;Cursor position in screen
0207    28                              DB      40              ;Screen size X
0208    08                              DB      8               ;Screen size Y
0209    0101                            DW      0101H           ;Window left-upper position
020B    0000                            DW      0000H           ;Cursor position in window
020D    00                              DB      0               ;VRAM Y-offset
020E    00                              DB      0               ;Window type
020F    00                              DB      0               ;Secret mode
0210    00                              DB      0               ;Scroll mode
0211    0000                            DW      0               ;Scroll step
0213    00                              DB      0               ;Carriage return wait flag
0214    00                              DB      0               ;Function status
0215    0000                            DW      0               ;Function addr
0217    00                              DB      0               ;ESC flag
0218    00                              DB      0               ;ESC count
0219    00 00 00 00                     DB      0,0,0,0         ;Parameter store area
021D    00 00 00 00                     DB      0,0,0,0
0221    00 00 00                        DB      0,0,0
                                ;
0224                            SCRSAVE:
0224                            DS      27H             ;Screen data save area
                                END
```

```
                        ;       ************************************************
                        ;               CONSOLE DIRECT DISPLAY SAMPLE
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is using console
                        ;               out direct display.
                        ;
                        ;               This is same as TIMDAT sample program.
                        ;
                        ;       <> assemble condition <>
                        ;
                                .Z80
                        ;
                        ;       <> loading address <>
                        ;
                                .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
EB03                    WBOOT           EQU     0EB03H  ; WBOOT entry address.
EB06                    CONST           EQU     0EB06H  ; CONST entry address.
EB09                    CONIN           EQU     0EB09H  ; CONIN entry address.
EB0C                    CONOUT          EQU     0EB0CH  ; CONOUT entry address.
EB4E                    TIMDAT          EQU     0EB4EH  ; TIMDAT entry address.
EB69                    CALLX           EQU     0EB69H  ; CALLX entry address.
                        ;
                        ;       Bank value
                        ;
00FF                    SYSBANK         EQU     0FFH
0000                    BANK0           EQU     000H
0001                    BANK1           EQU     001H
0002                    BANK2           EQU     002H
                        ;
                        ;
1000                    MAINSP          EQU     01000H  ; Stack pointer.
                        ;
                        ;       System area
                        ;
F2AC                    LESCPRM         EQU     0F2ACH  ; ESC sequence parameter area.
F2A8                    LFKADDR         EQU     0F2A8H  ; CONOUT execute addr.
F52E                    DISBNK          EQU     0F52EH  ; Bank data.
                        ;
                        ;
000D                    CR              EQU     0DH
000A                    LF              EQU     0AH
0012                    CLS             EQU     12H
001B                    ESC             EQU     1BH
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Display time until press BREAK key.
                        ;               And key input any key.
                        ;
0100                    START:
0100    31 1000                 LD      SP,MAINSP       ; Set stack pointer.
                        ;
0103    CD 0136                 CALL    GETFKAD         ; Get direct display function addr.
                        ;
0106    21 01C6                 LD      HL,MSG01        ; Date & time message.
0109    CD 0143                 CALL    DSPMSG          ; Dispay message.
                        ;
010C                    LOOP:
010C    76                      HALT
                        ;
010D    CD EB06                 CALL    CONST           ; Key in check.
0110    3C                      INC     A               ; Input any key?
0111    20 0B                   JR      NZ,SKIP         ; No.
0113    CD EB09                 CALL    CONIN           ; Get inputed key.
0116    FE 03                   CP      03H             ; BREAK key?
0118    28 19                   JR      Z,TIMEEND       ; Yes.
                        ;
011A    4F                      LD      C,A             ; Display inputed character.
011B    CD EB0C                 CALL    CONOUT          ;
                        ;
011E                    SKIP:
011E    11 0225                 LD      DE,NTIME        ; Time discrepter.
0121    0E 00                   LD      C,00H           ; Read time function.
0123    CD EB4E                 CALL    TIMDAT          ; Read time.
                        ;
0126    CD 017E                 CALL    TIMECHK         ; New & old time compare.
0129    28 E1                   JR      Z,LOOP          ; If same, then loop.
                        ;
012B    CD 018E                 CALL    TIMESET         ; Set new time data.
012E    CD 014F                 CALL    DSPTIME         ; Display time data
0131    18 D9                   JR      LOOP            ; Loop
                        ;
0133                    TIMEEND:
0133    C3 EB03                 JP      WBOOT           ; Jump WBOOT.
                        ;
                        ;       ************************************************
                        ;               GET DIRECT DISPLAY FUNCTION ADDRESS.
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This routine sends dummy console out function.
                        ;               And get the function execute address in
                        ;               OS ROM.
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
```

```
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0136                    GETFKAD:
0136    21 021F                 LD      HL,DIRECT       ; Dummy console out data.
0139    CD 0143                 CALL    DSPMSG          ;
                        ;
013C    2A F2A8                 LD      HL,(LFKADDR)    ; Get the function execute addr.
013F    22 0233                 LD      (FKDIRECT),HL   ; Save the address.
0142    C9                      RET                     ;
                        ;
                        ;       ************************************************
                        ;               DISPLAY MESSAGE UNTIL FIND 0
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;       <> entry parameter <>
                        ;               HL ; Message data top address.
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
0143                    DSPMSG:
0143    7E                      LD      A,(HL)          ; Get message data.
0144    B7                      OR      A               ; End mark?
0145    C8                      RET     Z               ; Yes, then return.
                        ;
0146    4F                      LD      C,A             ; Set display data to c reg.
0147    E5                      PUSH    HL              ; Save message pointer.
0148    CD EB0C                 CALL    CONOUT          ; Display message.
014B    E1                      POP     HL              ; Restore message pointer.
014C    23                      INC     HL              ; Pointer update.
014D    18 F4                   JR      DSPMSG          ; Loop until find 0.
                        ;
                        ;       ************************************************
                        ;               DISPLAY TIME DATA
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Display time data by calling OS ROM directly.
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
014F                    DSPTIME:
014F    21 020B                 LD      HL,MSG02        ; Date message.
0152    CD 0158                 CALL    DSPT10          ; Display the message.
                        ;
0155    21 0215                 LD      HL,MSG03        ; Time message.
0158                    DSPT10:
0158    11 F2AC                 LD      DE,LESCPRM      ; CONOUT parameter area.
015B    7E                      LD      A,(HL)          ; Set Y-coordinate.
015C    12                      LD      (DE),A          ;
015D    23                      INC     HL              ;
015E    13                      INC     DE              ;
                        ;
015F    7E                      LD      A,(HL)          ; Set X-coordinate.
0160    12                      LD      (DE),A          ;
0161    23                      INC     HL              ;
                        ;
0162    06 08                   LD      B,08H           ; Loop counter.
0164                    DSPT20:
0164    4E                      LD      C,(HL)          ; Display character.
0165    3E FF                   LD      A,SYSBANK       ; Set system bank value.
0167    32 F52E                 LD      (DISBNK),A      ;
016A    DD 2A 0233              LD      IX,(FKDIRECT)   ; OS ROM call address.
016E    C5                      PUSH    BC              ; Save registers.
016F    D5                      PUSH    DE              ;
0170    E5                      PUSH    HL              ;
0171    CD EB69                 CALL    CALLX           ; Go !!
0174    E1                      POP     HL              ; Restore registers.
0175    D1                      POP     DE              ;
0176    C1                      POP     BC              ;
                        ;
0177    1A                      LD      A,(DE)          ; Increment X-coordinate.
0178    3C                      INC     A               ;
0179    12                      LD      (DE),A          ;
017A    23                      INC     HL              ; Message pointer update.
017B    10 E7                   DJNZ    DSPT20          ; Loop.
                        ;
017D    C9                      RET                     ;
                        ;
                        ;       ************************************************
                        ;               CHECK OLD & NEW TIME
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;         entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               ZF : Return ifomation
                        ;                    =1 : New time is same as old one.
                        ;                    =0 : New time is different from old one.
                        ;       <> preserved registers <>
                        ;               NON
                        ;
```

```
017E                    TIMECHK:
017E    21 0225                 LD      HL,NTIME        ; New time data.
0181    11 022C                 LD      DE,OTIME        ; Old time data.
0184    06 06                   LD      B,06H           ; Data counter.
                        ;
0186                    TLOOP:
0186    1A                      LD      A,(DE)          ; Get old time data.
0187    BE                      CP      (HL)            ; Compare it with new one.
0188    C0                      RET     NZ              ; If disagree, then return.
0189    13                      INC     DE              ; Ponlnters update.
018A    23                      INC     HL              ;
018B    10 F9                   DJNZ    TLOOP           ; Loop 6 times.
018D    C9                      RET                     ;
                        ;       ************************************************
                        ;               SET TIME DATA
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
018E                    TIMESET:
018E    21 0225                 LD      HL,NTIME        ; Set time data to old time area.
0191    11 022C                 LD      DE,OTIME        ;
0194    01 0006                 LD      BC,6            ; Year/month/date/hour/minuite/second
0197    ED B0                   LDIR                    ; Move new data to old area.
                        ;
0199    21 0225                 LD      HL,NTIME        ; Set BCD data to message area with ASCII.
019C    11 020D                 LD      DE,DATE         ; HL is source, DE is destination.
019F    06 03                   LD      B,03H           ; B is counter.
01A1                    SET10:
01A1    CD 01B5                 CALL    SETASCII        ; Convert BCD to ASCII.
01A4    23                      INC     HL              ; Pointer update.
01A5    13                      INC     DE              ;
01A6    10 F9                   DJNZ    SET10           ; Loop 3 times. (Year/month/date)
                        ;
01A8    11 0217                 LD      DE,TIME         ; Time date setting area.
01AB    06 03                   LD      B,03H           ;
01AD                    SET20:
01AD    CD 01B5                 CALL    SETASCII        ; Convert BCD to ASCII.
01B0    23                      INC     HL              ; Pointer update.
01B1    13                      INC     DE              ;
01B2    10 F9                   DJNZ    SET20           ; Loop 3 times. (Hour/minuite/second)
01B4    C9                      RET                     ;
                        ;       ************************************************
                        ;               SET ASCII DATA FROM BCD DATA
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               HL : BCD data address.
                        ;               DE : ASCII data setting address.
                        ;       <> return parameter <>
                        ;               DE : Entry DE + 2
                        ;       <> preserved registers <>
                        ;               HL
                        ;
01B5                    SETASCII:
01B5    7E                      LD      A,(HL)          ; Get BCD data.
01B6    F5                      PUSH    AF              ; Save BCD data.
01B7    0F                      RRCA                    ; Move MSB 4 bit to LSB 4bit.
01B8    0F                      RRCA                    ;
01B9    0F                      RRCA                    ;
01BA    0F                      RRCA                    ;
01BB    CD 01BF                 CALL    NEXT            ; Set ASCII data by 1 byte.
01BE    F1                      POP     AF              ; Restore BCD data.
01BF                    NEXT:
01BF    E6 0F                   AND     0FH             ; Check LSB 4 bit.
01C1    C6 30                   ADD     A,30H           ; Change to ASCII data.
01C3    12                      LD      (DE),A          ; Set ASCII data.
01C4    13                      INC     DE              ; Setting pointer update.
01C5    C9                      RET                     ;
                        ;
                        ;
01C6                    MSG01:
01C6    0C                      DB      0CH
01C7    50 72 65 73             DB      'Present date is          .',CR,LF
01CB    65 6E 74 20
01CF    64 61 74 65
01D3    20 69 73 20
01D7    20 20 20 20
01DB    20 20 20 20
01DF    2E 0D 0A
01E2    50 72 65 73             DB      'Present time is          .',CR,LF
01E6    65 6E 74 20
01EA    74 69 6D 65
01EE    20 69 73 20
01F2    20 20 20 20
01F6    20 20 20 20
01FA    2E 0D 0A
01FD    49 6E 70 75             DB      'Input line = '
0201    74 20 6C 69
0205    6E 65 20 3D
0209    20
020A    00                      DB      00H
020B                    MSG02:
020B    01 11                   DB      01H,11H                 ; Direct display
020D                    DATE:
```

```
020D  30 30 2F 30                    DB      '00/00/00'
0211  30 2F 30 30
0215                          MSG03:
0215  02 11                          DB      02H,11H                 ; Direct display
0217                          TIME:
0217  30 30 3A 30                    DB      '00:00:00'
021B  30 3A 30 30
                              ;
021F                          DIRECT:
021F  1B D2 01 01                    DB      ESC,0D2H,1,1,20H        ; Direct display dummy
0223  20
0224  00                             DB      00H
                              ;
                              ;
                              ;
0225                          NTIME:
0225                                DS      7
022C                          OTIME:
022C                                DS      7
                              ;
0233                          FEDIRECT:
0233                                DS      2
                                    END
```

3.7 MTOS/MIOS Operations

The PINE is provided with an optional microcassette drive. It is connected to the PINE main unit via a cartridge interface (MS mode). Tape data is also directed to the buzzer in the main unit or an external loudspeaker. The microcassette drive can be controlled either manually or by software. The user can handle microcassettes in the same way as ordinary disk drives.

A microcassette can store only sequential files. It contains a directory file at its beginning so that the system can handle microcassette files on a file basis.

3.7.1 General

The PINE controls I/O operations on microcassette tape (referred to simply as tape from now on) at two levels of control programs (MTOS and MIOS). MTOS is an operating system which corresponds to CP/M BDOS and manages files on tape with unique interface modules. MIOS corresponds to CP/M BIOS and controls the microcassette drive motor and processes tape data on a record basis.

In PINE CP/M, the microcassette drive (MCT) is assigned to drive H:. The application program can handle it as an ordinary external drive without being aware that it is actually an MCT drive.

Each MCT tape has a directory file at its beginning. This file is used to control accesses to the files on the MCT. Whenever an access is made to an MCT file, the directory file is loaded into the directory area in RAM (RAM directory). The results of operations on MCT files are all managed in the RAM directory. Accordingly, when an MCT file is updated, the contents of the RAM directory must also be rewritten into the directory file on the MCT. Reading in this directory file into memory is called "MOUNT" and writing it onto MCT is called "REMOVE".

- MOUNT
The MOUNT function loads the tape directory into the RAM directory. After a mount, any directory update is processed in the RAM directory. This function must be executed before accessing an MCT file. (The OS supports the Auto Mount function.)

- REMOVE
The REMOVE function writes the RAM directory onto the tape directory. This function must be executed after manipulating an MCT file. If the tape is removed without the execution of this function, the tape contents are not guaranteed. Or, in the worst case, the contents of the microcassette that is mounted next may be destroyed.

The PINE MCT is furnished with an LED which indicates whether tape MOUNT/REMOVE can be executed. The LED is turned off by a mount and turned on by a remove. Tape can be mounted or removed while the LED is on.

Figure 3.7.1 shows the relationship of MTOS and MIOS to CP/M.

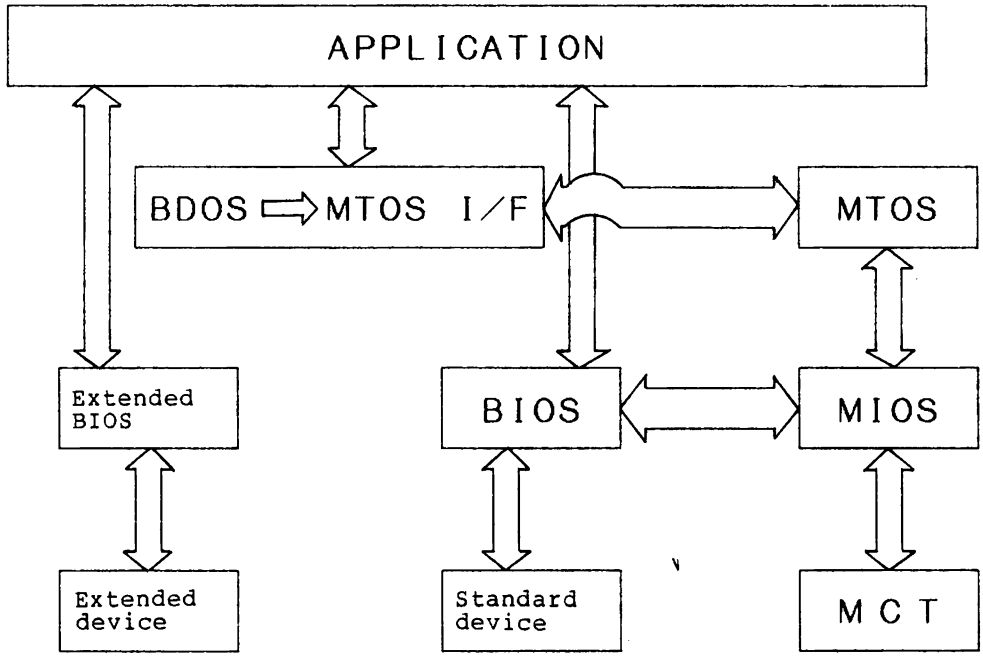


Fig. 3.7.1 MTOS/MIOS Control Flow

3.7.2 File Control

This subsection explains the MCT file structure and file control method.

The structure of an MCT file is shown in Figure 3.7.2.

The PINE directory file can contain a maximum of 12 entries. Its contents are loaded into the RAM directory for control of file accesses.

Each file consists of the header, data, and EOF sections.

Header area:	Contains the information pertaining to the organization of the file.
Data area:	Contains actual file data. The data area is normally made of one or more blocks. The number of blocks depends on the file size.
EOF area:	Is the last block identifying the end of the file.

The PINE can write a block several times to increase data reliability (normally, a block is written twice). Each block has an ID field which stores the block type, block number, and the ordinal number of writes. The ID field is referenced during subsequent read operations.

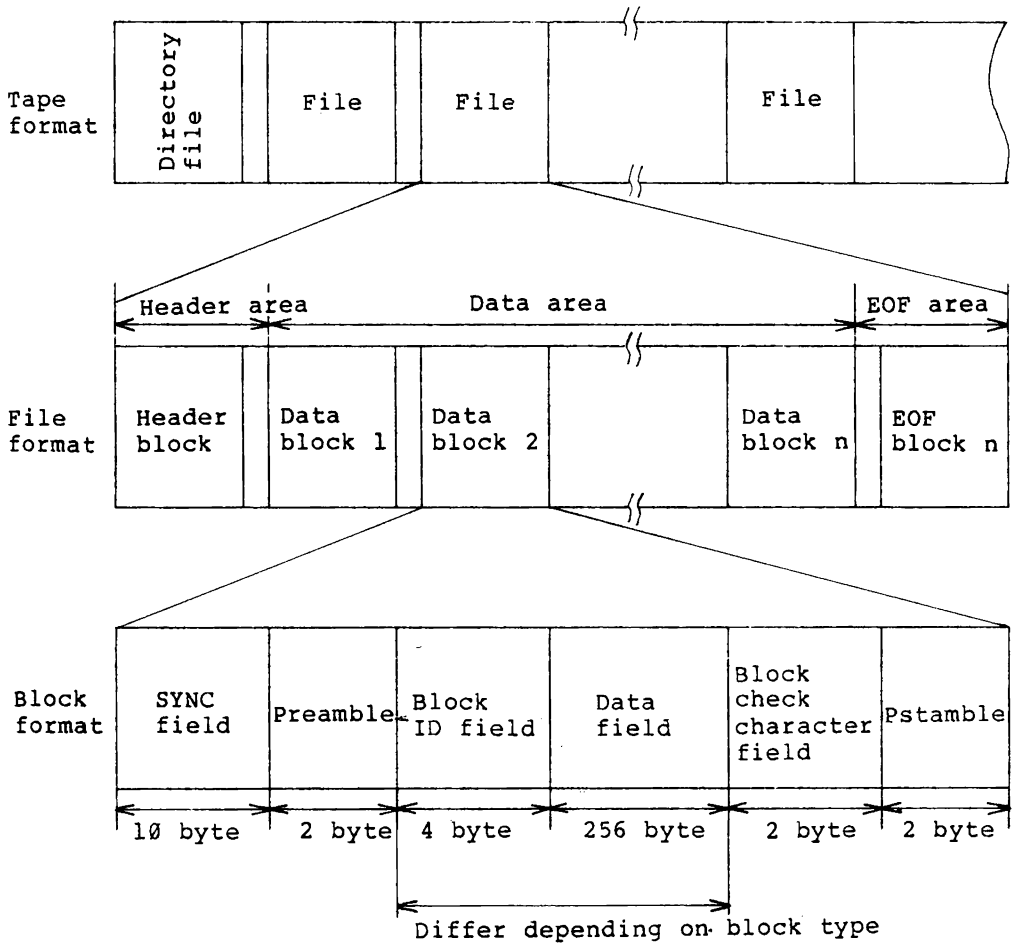
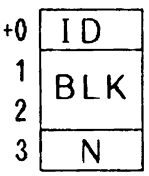


Fig. 3.7.2 Tape Structure

3.7.2.1 Block format

Each block has the format as shown below. The formats of the data fields are described in 3.7.2.2.

Field name	Size (bytes)	Description
SYNC field	10	Used for tape synchronization. Contains 80 Ø bits.
Preamble	2	Contains ØFFH and ØAAH.
Block ID field	4	Identifies the block.  Byte Ø: Block identifier "H": Header block "D": Data block "E": End of file block Bytes 1 and 2: 2-byte binary number indicating the block number. Byte 3: Block ID number indicating the ordinal number of writes.
Data filed	256	Contains data bytes.
BCC (Block Check character) field	2	BCC characters computed using CRC (Cyclic Redundancy Check). The CCITT CRC method is used. Calculation extends from the first block number to the BCC field itself.
Postamble	2	Contains ØFFH and ØAAH.

3.7.2.2 Data field formats

The data field formats differ among the header, data, and EOF blocks. The following pages describe the format of each field.

(1) Header block data field format

Column from	to	Size	Item	Description
0	3	4	ID field	Identifies an 'HDR1' header (ASCII).
4	11	8	File name	Contains the file name.
12	19	8	File type	Contains the file type.
20		1	Record type	'F': Fixed length 'v': Variable length 'n': Fixed length The same block is written n times.
21		1	Block mode	' ': Specifies that the intergaps are to be long enough to stop the tape successfully. 's': Specifies that the intergaps are not to be long enough to stop the tape successfully.,
22	26	5	Block length	Indicates the length of the block (ASCII number from '00000' to 'FFFFF').
27	28	2	Counter value	Indicates the tape count at which this file starts.
29		1		Free
30		1		File attribute
31		1		Free
32	37	6	Date of creation	Date on which this file is created. The month, day, and year are set in that order in 2-byte ASCII codes, respectively.
38	43	6	Creation time	Time at which this file is created. The hour, minute, second are set in that order in 2-byte codes, respectively.
44	49	6		Free
50	51	2	Volume number	Tape volume number ('01' -)
52	59	8	System name	Name of the system under which the file is created.
60	61	2	System file number	File number by which the file is controlled in the directory. 0000H - 0FFFFH
62	67	8		Reserved for the system.
68	255			Free

(2) Data block data field format
The data field of a data block contains actual data bytes. There is no restriction on the data format except that the data field size is 256 bytes.

(3) EOF block data field

Column from	to	Size	Item	Description
0	3	4	ID field	Contains 'EOF' (ASCII).
4	11	8	File name	Contains the file name.
12	19	8	File type	Contains the file type.
20	21	2	File number	Contains the file number by which the file is controlled in the directory ('0000' to 'FFFF').
22	23	2	Counter value	Contains the last counter value for the preceding block.
24	255			Free

3.7.2.3 Directory File Structure

The directory file comprises the following three blocks:

1) Directory ID (first block)
The directory ID contains the information obtained when the tape directory is created for the first time, the information set up when the MCT is used last, and the information pertaining to the MCT itself such as the current total block count and the total record count.

2) Directory (second and third blocks)
The directory contains the information pertaining to the files on the MCT. One file entry is 32 bytes long and one block of directory contains information for 8 files. Two blocks are reserved for the directory because each MCT contains a maximum of 12 files.

3) RAM directory
The directory file is loaded into RAM by the MOUNT function for controlling subsequent accesses to the MCT. The directory in RAM is called the RAM directory. Its structure is identical to that on the MCT.

(1) Directory ID (first block)

Column from	to	Size	Item	Description
0	7	8	Tape name	Loaded with the tape name and volume number specified when the directory is created.
8	9	2	Volume number	
10	17	8		Reserved for the system.
18	23	6	Creation date	Loaded with the date at which the directory is created. The month, day, and year are set in this order in 2-byte codes, respectively.
24	29	6	Creation time	Loaded with the time at which the directory is created. The hour (ØØ-23), minute, and second are sets in this order in 2-byte codes, respectively.
30	35	6	Remove date	Loaded with the date at which the tape is removed last. The month, day, and year are set in this order in 2-byte codes, respectively.
36	41	6	Remove time	Loaded with the time at which the tape is removed last. The hour (ØØ-23), minute, and second are set in this order in 2-byte codes, respectively.
42	43	2	Total mount count	Loaded with the total number of mounts. This field is initialized to Ø and incremented at each mount operation.
44	45	2	Total block count	Loaded with the total number of blocks on the MCT except those for the directory file.
46	47	2	Total record count	Loaded with the total number of records except those for the directory file.
48		1	Total file count	Loaded with the total number of files on the MCT.
49		1	System flag 1	Tape ID flag (used by the automatic MTOS identification feature.)
50		1	System flag 2	Tape TOS controller (used by the automatic MTOS identification feature.)
51	52	2	Last file number	The file number of the last file on the MCT.
53	255			Free

(2) Directory (second and third blocks)

Column from	to	Size	Item	Description
0	1	2	File number	Is the number of the file by which MTOS manages the file. MTOS manages a file not by its file name but by the file number assigned to it.
2		1	Presence flag	Indicates the file cataloged in this directory entry is valid or invalid. 00H: Valid 0FFH: Invalid
3		1	File attribute 1	Indicates the logical characteristics of the file and the mode during read operations. Bit Description 7 0: Nonstop mode access 1: Stop mode access 6 - 4 Don't care. 3 - 0 Number of retires during read
4		1	File attribute 2	Reserved by the system.
5	6	2	Number of blocks	Total number of blocks in the file. The header block is counted as 0000H.
7	8	2	Number of records	Total number of records in the file. The first 128-byte record in the first block is counted as record 0000H.
9	10	2	Starting count	Tape counter value at which the header of this file starts.
11	12	2	Ending count	Tape counter value at the end of an EOF block.
13	14	2	Number of opens	Number of open operations performed.
15		1	User number	User number assigned by the CP/M system.
16	23	8	File name	Loaded with the name of the file.
24	31	8	File type	Loaded with the file type of the file. The highest order three bytes of this field are used.

3.7.3 Tape File Control Block (T-FCB)

3.7.3.1 T-FCB general

When using MTOS services, the application program must use a packet, called the T-FCB, which corresponds to the CP/M FCB. The T-FCB is placed in the same memory location as FCB but their formats are different.

3.7.3.2 T-FCB format

Figure 3.7.3 shows the T-FCB format.

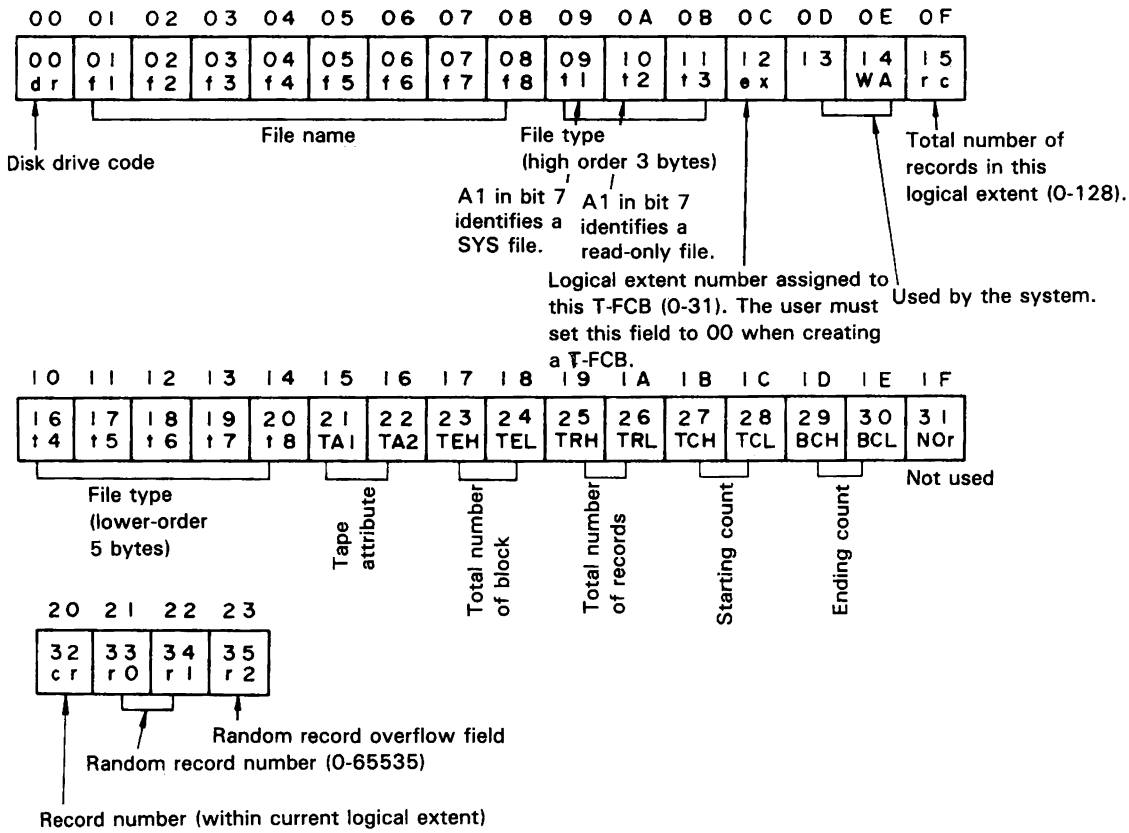


Fig. 3.7.3 T-FCB Format

3.7.3.3 T-FCB description

Column from to	Field name	Description
Ø	dr	Loaded with the disk drive code. Must be set to ØØH when drive H is selected or Ø8H when another drive is selected.
1 8	f1 - f8	Loaded with the file name.
9 11	t1 - t3	Loaded with the file type. A 1 in t1, bit 7 identifies a read-only file. A 1 in t2, bit 7 indicates an SYS file.
12	ex	Logical extent number assigned to this T-FCB. The user must set this field to ØØH when creating a T-FCB.
13 14		Used by the system.
15	rc	Total number of records in this logical extent (ØØH - 8ØH).
16 2Ø	t4 - t8	Loaded with the lower five bytes of the file type.
21 22	TA1, TA2	Loaded by the system with a tape attribute.
23 24	TBH, TBL	Loaded with the total number of blocks reserved for the file. This field is incremented by one each time a block is written onto the file.
25 26	TRH, TRL	Loaded with the total number of records stored in the file. This field is incremented by one each time a record is written onto the file.
27 28	TCH, TCL	Loaded with the starting tape count of the file.
29 3Ø	BCH, BCL	Loaded with the ending tape count of the file. This field is updated each time a record is written onto the file.
31	Ncr	Not used.
32	Cr	Loaded with the record number in the current logical extent. The user must set this field to ØØH when creating a T-FCB for the first time.
33 34	rØ, r1	Loaded with the random record number (ØØØØH - ØFFFFH).
35	r2	A nonzero value in this field indicates that a random record overflow condition has occurred.

3.7.3.4 Notes on the use of T-FCB

The contents of the T-FCB are updated whenever an access is made to the associated file on the MCT.

Incorrect file size values will be returned when a STAT command is executed for the H drive. This is because MTOS uses the file allocation vector field (FCB+16 through FCB+31) for purposes different from the original ones. In CP/M, the file allocation vector field indicates the utilization status of the file space on the disk and the CP/M STAT command examines this field to calculate the size of the file.

3.7.4 Using MTOS

3.7.4.1 Outline

MTOS/MIOS is included into CP/M through an interface. MTOS functions are compatible with BDOS functions. Whether an MTOS function or BDOS function is specified is determined by the following rules:

1. Functions not supported by MTOS are carried out by BDOS.
2. Functions requiring an FCB (T-FCB) as a parameter are identified by the disk drive code at the beginning of the FCB (T-FCB).
3. Other MTOS functions are serviced only when the MCT drive is logged in.

The application program must call BDOS when using MTOS functions. MTOS calls differ from BDOS calls in that drive H must be specified as the drive code at the beginning of the FCB.

Although FCB and T-FCB are located in the same memory location as mentioned in 3.7.3, their format and use are different. In the following sections, only T-FCB is used to represent a file control block.

3.7.4.2 MTOS Programming Considerations

(1) MOUNT and REMOVE
As mentioned in 3.7.2, MTOS loads the directory file at the beginning of the MCT into the RAM directory area in memory when controlling a tape file. Therefore, the following considerations must be taken when handling MCT.

1) Whenever a microcassette tape is inserted into the PINE microcassette slot, it is necessary to performa MOUNT operation. In practice, however, the user need to perform nothing for this purpose because the automatic mount feature is executed when he accesses the H drive via BDOS (not calling MIOS directly).

2) When a write or rename operation is performed on the MCT, however, the associated file information is updated only in the RAM directory. It is therefore necessary to restore the updated file information from the RAM directory onto the MCT at the end of the file processing. This is called a REMOVE operation. The user must perform a REMOVE operation immediately before removing an MCT from the PINE whenever a file is written on the MCT.

3) Whether an MCT is mounted or removed is indicated by the LED identifying the MCT tape MOUNT/REMOVE state. The user can remove the MCT while that LED is on.

(2) File access

MTOS controls the file acessing by examining the directory at the beginning of the MCT.

Each file is blocked and accessed on a block basis. Blocks are identified by sequentially assigned block numbers.

Files are given a file name but controlled by a file number assigned by the system at file open time. The file number counter is set to 0000H when the directory is created and incremented each time a file is created. The maximum file number is 0FFFFH.

The following restrictions on file accessing must be noted:

1) No more than one file can be opened simultaneously.
2) When a file is opened in the write mode, no other files can be opened until that file is closed.
3) A new file may be opened when another file has been opened in the read mode. The currently open file, however, is closed.
4) An attempt to open the already-opened file on to write it will cause an error.

The user must observe the following restrictions on sequential file access:

1) In the write mode:
Once a record is written in a block, no record can be placed in the next block until that block is filled. No data can be written in any locations other than the record areas in the current and next blocks.

2) In the read mode:
No record area can be accessed other than those in the current and next blocks.

3) Read from a file opened in the write mode:
Any record in the current block may be read.

MTOS accesses an MCT in one block (256 bytes) at a time while the system accesses in one record (128 bytes) at a time. This means that MTOS performs blocking/deblocking to make each block.

(3) Note on the creation of an MCT file

A new file created on the H drive (MCT) is placed after the last file on the MCT. Consequently, a disk full condition may occur even when there is only one file on the MCT if that file is stored as shown in Figure 3.7.4.

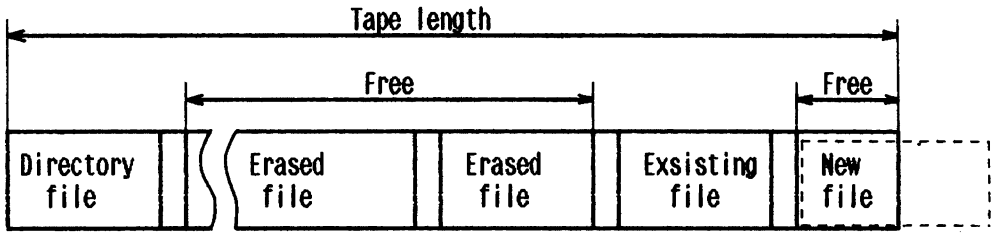


Fig. 3.7.4 Disk Full Condition

3.7.4.3 Miscellaneous Considerations on MTOS

(1) MTOS and MIOS termination and power-off processing

MCT processing may be terminated by stopping the MCT (CTRL/STOP) or turning off MCT power. MTOS and MIOS react as summarized below when one of the above conditions occurs.

The CTRL and STOP keys must be pressed simultaneously a little longer than ordinary keys are because keyboard interrupts are disabled during a write onto the current block.

When MCT is stopped:

- MTOS
  If the current operation is a seek, write, or read, MTOS stops the operation, stops the motor, unloads the read/write head, and returns control to the calling program with the return code ØFFH in the A register.
- MIOS
  MIOS terminates processing as Bad sector error if the current operation is a read or write. The file is closed. MIOS does not store the directory file if the current operation is a write.

When power is turned off:

- MTOS
  If a power off condition occurs during a read or write, MTOS stops the motor and unloads the read/write head after completing the access to the current block. MTOS then returns control to the calling program.
- MIOS
  Power is turned off after MIOS processing is completed. Subsequent MIOS processing when power is restored differs depending on the mode in which power was turned off. When power was turned off in the continue mode, MCT processing continues at the point when power was turned off. In the restart mode, the file that was open when power was turned off is closed. The file that was opened in the write mode is not cataloged in the directory.

(2) Automatic MTOS identification feature
1) Auto mount
MTOS carries out an automatic mount when an MTOS function is called on an MCT which has not been mounted (auto mount feature). This allows the user to operate the MCT without being aware of the need of mount processing.

2) Auto remove check
MTOS does not perform remove processing unconditionally; when performing remove processing, MTOS reads in the directory ID and compares the creation date and time with those in the RAM directory. It carries out the actual remove processing only when a match occurs. The auto remove check function is disabled by default.

(3) MTOS flags for the automatic MTOS identification feature

MTOS uses a flag to control automatic MTOS operations such as auto mount and remove. This flag is referenced by the automatic MTOS identification feature to identify the operation to be performed. The flag is labelled TOSCTL. It is initialized when an MCT is mounted. The contents of TOSCTL are determined as summarized below based on the system flag 2 in the directory ID.

TOSCTL is set to 50H when the system flag 2, bit 7 is 0.
TOSCTL is set to the value of the system flag 1 when the system flag 2, bit 7 is 1.
(The contents of TOSCTL is described later.)

The TOSCTL value for an MCT may be altered using the procedure shown in Figure 3.7.5. After this procedure, MTOS automatically operates on the MCT in the specified control mode when it is mounted.

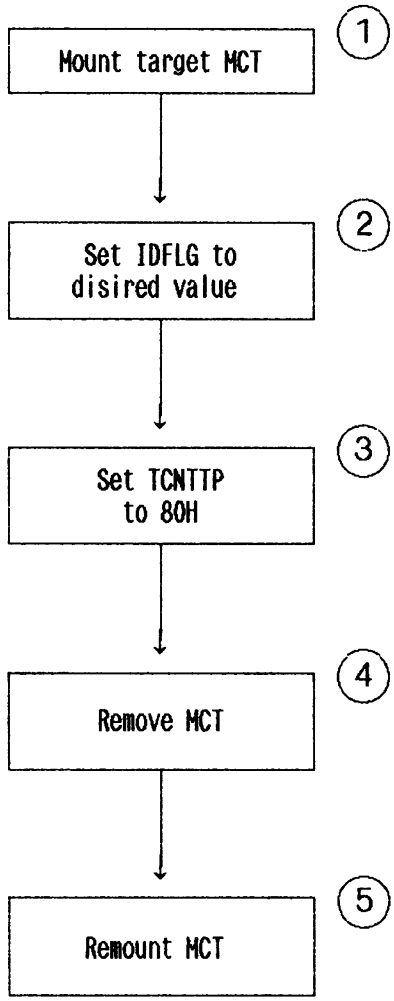


Fig. 3.7.5 Procedure for TOSCTL Alteration

Step	Action	Description
1	Mount the target MCT	Load the tape directory into the RAM directory area.
2	Set the desired value in IDFLG	Specify the desired MTOS control mode in RAM directory area IDFLG (system flag 1).
3	Set 80H in TCNTTP	Load 80H in RAM directory area TCNTTP (system flag 2). This causes the TOSCTL value to be initialized as specified by the user.
4	Remove the MCT	Copy the RAM directory into the tape directory. After this, MTOS operation on this tape is controlled by the mode specified by the user.
5	Remount the MCT	Load the tape directory into the RAM directory area. This causes the IDFLG to be set to the value of TCNTTP specified in step 3. After this, the TOSCTL is set to the value of IDFLG set up in step 2 when this tape is mounted.

```
TOSCTL (0EFBBH)  1 byte
- MTOS control flag
     Bit 7      Always set to 0.
         6 = 1: Enables counter adjustment.
           = 0: Disables counter adjustment.
         5 = 1: Determines access mode based on system status.
           = 0: Determines access mode based on directory status.
         4 = 1: Enables auto mount.
             0: Disables auto mount.
         3 = 1: Enables auto remove check.
           = 0: Disables auto remove check.
         2     Always set to 0.
         1     Always set to 0.
         0 = 1: Disconnects MTOS from CP/M.
           = 0: Includes MTOS into CP/M.
     TOSCTL defaults to 50H.

IDFLG (0FA13H)  1 byte
- System flag 1 in RAM directory
The bit assignments are identical to those of TOSCTL.

TCNTTP (0FA14H)  1 byte
- System flag 2 in RAM directory.
     Bit 7 = 1: Loads the IDFLG value into TOSCTL at a mount.
           = 0: Loads 50H into TOSCTL at a mount.
     Bits 6 - 0: Always set to 0.
```

(4) Tape access mode

MTOS and MIOS allow accesses to MCTs in two modes, i.e., stop and nonstop modes. These modes can be switched by means of the system display or by changing the value in the associated system area. Accesses are retried several times if the requested block is not accessed successfully. The rest of this subsection describes the stop and nonstop modes.

1) Stop and nonstop mode operations

Functional differences between the stop and nonstop modes are listed below.

i) Stop mode
- Function
  In the stop mode, the motor is stopped each time one block of data has been read or written. The motor is started again when the next block is read or written.
- Advantage
  Since the motor is in the stopped state after a block has been read or written, the next block may be read or written at any given time.
- Disadvantage
  Because the times required for turning the motor on and off are added to the read or write time, read or write operations in the stop mode take longer processing time than in the nonstop mode.

ii) Nonstop mode
- Function
  During a read or write in the nonstop mode, the motor is not stopped after one block of data has been read or written.
- Advantage
  Because no time is required for turning the motor on and off as in the stop mode, read or write operation in the nonstop mode takes shorter processing time.
- Disadvantage
  The subsequent read operation must be started before the next block passes through the read or write head; otherwise, a read error would occur. If this mode is used for single-block writes, the inter-block gap size will be increased. This results in a waste of tape space and an increase in the access time in the read mode.

2) Block read and write

MTOS writes a block of data onto an MCT n times (n defaults to 2) to increase tape reliability. Because of this redundancy, MTOS needs to read only one block out of n blocks successfully in the read mode. If errors occur in all (n) blocks written as a single block, MTOS makes several times of retries.

The user can specify write data to be verified during file close after write. In the verify mode, MTOS only checks whether the written file can be read in normally and does not check the file contents. (The default is nonverify mode.)

3) Changing the tape access mode
The value of TACMOD is established when a new file is created for write or when an existing file is opened for read.

The user can change the value of TACMOD using the procedure shown in Figure 3.7.6. Step 2 in the figure is unnecessary for write because TACMOD is unconditionally set to the value of TACATR during a write operation.

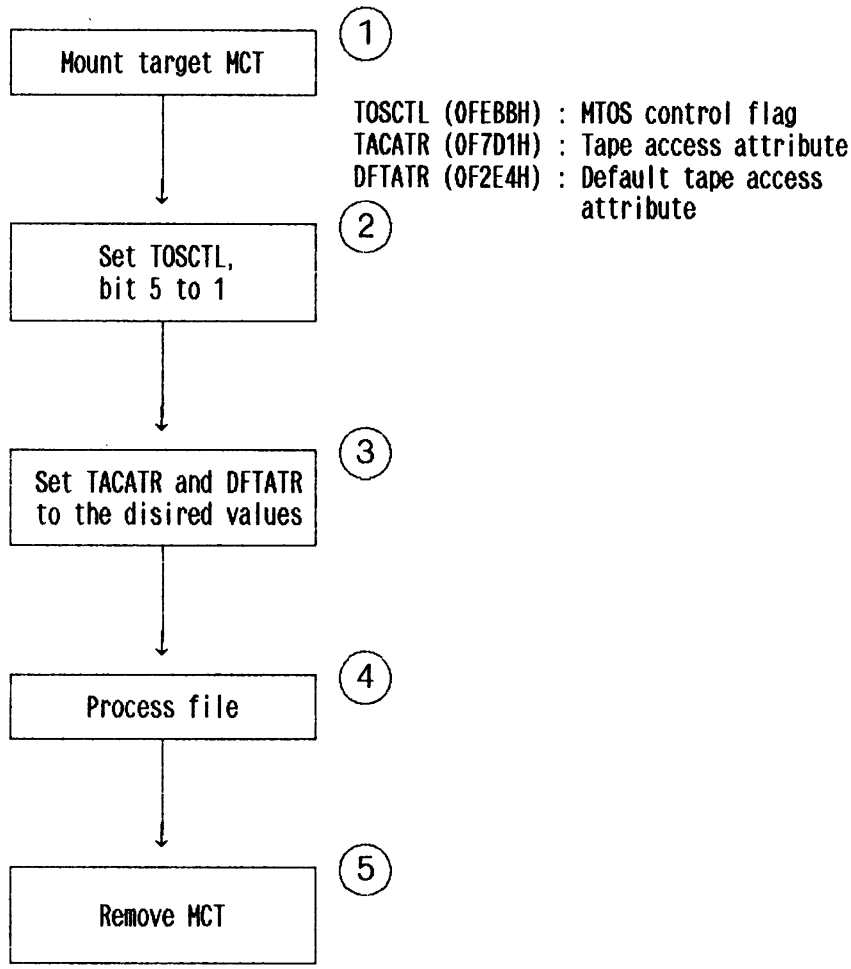


Fig. 3.7.6 Procedure for TACMOD Alteration

Step	Action	Description
1	Mount the target MCT	Load the tape directory into the RAM directory area.
2	Set TOSCTL, bit 5 to 1	Set the TACMOD to the mode specified by the file attribute 1 field in the RAM directory. This operation is unnecessary for write because the TACMOD is always set to the value of TACATR during a write.
3	Set TACATR and DFTATR to the desired values	Change TACATR and DFTATR to the desired values and specify the stop mode and number of retries. Once the value of DFTATR is altered, the new value is held valid until the next reset. This is because DFTATR is copied into TACATR during a warm boot.
4	Process file	For write, create a file and write data into the file. For read, open and read in a file.
5	Remove MCT	Copy the RAM directory into the tape directory.

VERFDFLT (ØEFBEH) 1 byte
- Verify mode flag for write (for default setting)
    = ØØH: Nonverify mode
    = Nonzero: Verify mode
The default setting is the nonverify mode. VERFFG is set to the VERFDFLT value during a warm boot.

VERFFG (ØEFBFH) 1 byte
- Verify mode flag for write
    = ØØH: Nonverify mode
    = Nonzero: Verify mode
VERFFG is initialized to the VERFDFLT value at a warm boot.

DFTATR (ØF2E4H) 1 byte
- Tape access attribute area (for default setting)
    Bit 7 Ø: Nonstop mode 1: Stop mode
    Bits 6 - 4: Always set to 1ØØB.
    Bits 3 - Ø: Number of retries (Ø - 15)
The access mode and the number of retries are initially set to the stop mode and 2, respectively.
This area is set to the TACATR value at a warm boot.

TACATR (ØF7D1H) 1 byte
- Tape access attribute area
The bit assignments are identical to those for DFTATR.
TACATR is set to the value of DFTATR at a warm boot.

TACMOD (Ø7D2H) 1 byte
- Tape access mode
The bit assignments are identical to those for DFTATR. TACMOD is unconditionally initialized to the value of TACATR when a file is created. It is initialized to the value of TOSCTL (ØEFBBH), bit 5 when opening a file. A 1 in this bit specifies that TACMOD is initialized to the value of TACATR. A Ø in the bit indicates that the access mode is to be equal to the value specified in the file attribute 1 field in the RAM directory.

BLKWRTNO (ØF2FDH) 1 byte
- Number of block writes +1 (Ø2H - ØFFH)
MTOS writes the block of data the specified number of times. The initial value is Ø3H (twice).

BMDCNT (ØF357H) 1 byte
- Block mode check counter
Loaded with the number of read retires made during a block read. This counter is checked when a read error occurred. The initial value is Ø2H.

3.7.5 MTOS Functions

3.7.5.1 Outline

This subsection presents a description of the MTOS functions with entry and return information. The return information includes return codes that are returned when CP/M error reporting is not suppressed (referred to as return code 1) and return codes that are returned when CP/M error reporting is suppressed (referred to as return code 2). Error reporting can be suppressed by calling SETERR. See 3.2.4 for more detail.

When the latter information is returned by MIOS, the application program can obtain the error details by referring to BIOSERROR (ØF52BH) and TOSRCD (ØF7CEH).

The PINE assigns drive H (physical device number is Ø8H) to MCT, so before accessing a file, the application program must set up the drive code field of the T-FCB as follows:

When the current drive is H:	ØØH
When the current drive is other than H:	Ø8H

In the descriptions of the MTOS functions that follow, the descriptions of the use and error information of the MTOS functions that are identical to those of the BDOS functions are omitted (except the setting of the drive code at the beginning of the T-FCB).

3.7.5.2 BDOS calls

The following MTOS-related BDOS calls are described in this sub-subsection:

	BDOS call number	Function	Page
1	14	Select disk drive	II-304
2	15	Open file	II-304
3	16	Close file	II-305
4	17	Frist search	II-306
5	19	Delete file	II-306
6	20	Read sequential	II-307
7	21	Write sequential	II-307
8	22	Create file	II-308
9	23	Rename file	II-309
10	30	Set file attribute	II-309
11	33	Read random	II-310
12	34	Write random	II-310
13	251	Verify MCT	II-311
14	252	Remove(close MCT)	II-311
15	253	Mount (open MCT)	II-312
16	254	Read MCT ID number	II-312
17	255	Create directory file	II-313

Name SELDSK BDOS No. 14

Function Selects drive H:.

Entry parameter
C = ØEH
E = Drive No.
Load the E register with 7 when selecting drive H:.

Return parameter
Return code 1
None.
Return code 2

TOSRCD	A reg.	H reg.	Result
ØØH	ØØH	ØØH	Normal termination.
Ø4H	FFH	Ø4H	Mount unsuccessful.
-	FFH	Ø1H	Error in MIOS.

Explanation
This function selects the drive specified in the E register. If no MCT is mounted when drive H is specified, the function automatically performs the MCT mount function.

Name OPEN BDOS No. 15

Function Opens an MCT file.

Entry parameter
C = ØFH
DE = T-FCB address

Return parameter
Return code 1
A = ØØH: Normal termination
ØFFH: File not found
Return code 2

TOSRCD	A reg.	H reg.	Result
ØØH	ØØH	ØØH	Normal termination.
ØØH	FFH	ØØH	No file found.
Ø4H	FFH	Ø4H	Mount unsuccessful.
Ø8H	FFH	ØØH	File not found.
Ø9H	FFH	Ø5H	File is already open.
11H	FFH	Ø5H	File number associated with the specified file was not found in the MCT directory.
-	FFH	Ø1H	Error in MIOS.

Explanation
The OPEN function searches the RAM directory for the file specified in the T-FCB, and if it is found, reads the header block of that file into the T-FCB for subsequent read or write operations on the file. The function performs an automatic MCT mount function if no MCT is mounted on the MCT.

An error is generated if the file is already open in the write mode. If the file is already open in the read mode, the function closes that file and opens a new file.

Tape access mode (stop or non-stop) is determined by the file attribute in the directory file when the file is opened in the read mode. See 3.7.4 for tape access modes.

Note
Load each of the T-FCB+12, T-FCB+15, T-FCB+32 with ØØH when opening a file.

Name CLOSE BDOS No. 16

Function Closes an MCT file.

Entry parameter
C = 10H
DE = T-FCB address

Return parameter
Return code 1
A = 00H: Normal termination
0FFH: Close error

Return code 2

TOSRCD	A reg.	H reg.	Result
00H	00H	00H	Normal termination.
00H	FFH	00H	No file found.
02H	FFH	05H	No MCT mounted.
08H	FFH	00H	File not found.
10H	FFH	00H	Verify error.
11H	FFH	05H	File number associated with the specified file was not found in the MCT directory.
-	FFH	01H	Error in MIOS.

Explanation
The directory of a file on which write operations are performed during processing must be updated at the end of the processing. This update is carried out by the CLOSE function.

When this function is called during write file processing, an EOF block is written on the MCT and the directory in memory is updated.

When the verify flag is on, the function automatically verifies the validity of the file contents before closing the file. The file will not be cataloged into the directory file on tape if an error is found.

Note
Once the RAM directory is updated, the directory file on the MCT must also be updated with the REMOVE function before the MCT is demounted from the PINE main unit.

Name FIRST SEARCH BDOS No. 17

Function Searches the file directory on MCT for the first time.

Entry parameter
C = 11H
DE = T-FCB address

Return parameter
Return code 1
A = 00H: File found
0FFH: File not found

Return code 2

TOSRCD	A reg.	H reg.	Result
00H	00H	00H	File found.
00H	FFH	00H	File not found.
04H	FFH	04H	Mount unsuccessful.
-	FFH	01H	Error in MIOS.

Explanation
The FIRST SEARCH function searches the RAM directory for the file specified in the T-FCB address in the DE register. If the file is found, the function loads the first 32 bytes of the DMA buffer with the file directory information and pads the remaining bytes with 0E5Hs.

Name DELETE BDOS No. 19

Function Deletes the specified file from the MCT directory.

Entry parameter
C = 13H
DE = T-FCB address

Return parameter
Return code 1
A = 00H: Delete successful
0FFH: File not found

Return code 2

TOSRCD	A reg.	H reg.	Result
00H	00H	00H	Normal termination.
00H	FFH	00H	No file found.
04H	FFH	04H	Mount unsuccessful.
06H	FFH	03H	R/O file.
0FH	FFH	02H	File number associated with the specified file was not found in the MCT directory.
-	FFH	01H	Error in MIOS.

Explanation
The DELETE function deletes the file specified in the T-FCB address in the DE register from the directory on drive H: (MCT). If no MCT is mounted, this function performs the mount function automatically.

Note
A file is actually deleted when the corresponding presence flag in the RAM directory is reset. Therefore, the user must execute a REMOVE function before demounting an MCT from the PINE.

Name SEQUENTIAL READ BDOS No. 20

Function Reads an MCT file sequentially.

Entry parameter
C = 14H
DE = T-FCB address

Return parameter
Return code 1
A = 00H: Normal termination
Nonzero: An EOF was encountered
Return code 2

TOSRCD	A reg.	H reg.	Result
00H	00H	00H	Normal termination.
04H	FFH	04H	Mount unsuccessful.
09H	FFH	05H	Invalid file specified.
0AH	01H	00H	File not open.
0BH	FFH	05H	Illegal record access.
0CH	01H	00H	EOF detected.
-	FFH	01H	Error in MIOS.

Explanation
The SEQUENTIAL READ function reads a record specified in the cr field (T-FCB+32) from the file specified in the T-FCB and returns that record to the DMA.

Note
The file to be read must be opened before this function is called. Although the function attempts the automatic MCT mount function if no MCT is mounted when it is executed, an error will be signaled by MTOS because the specified file is not open.

Name SEQUENTIAL WRITE BDOS No. 21

Function Writes an MCT file sequentially.

Entry parameter
C = 15H
DE = T-FCB address

Return parameter
Return code 1
A = 00H: Normal termination
Nonzero: An error occurred
Return code 2

TOSRCD	A reg.	H reg.	Result
00H	00H	00H	Normal termination.
04H	FFH	04H	Mount unsuccessful.
06H	FFH	03H	R/O file.
09H	FFH	05H	Invalid file specified.
0AH	01H	00H	File not open.
0BH	FFH	05H	Illegal record access.
0FH	FFH	02H	R/O disk.
-	FFH	01H	Error in MIOS.

Explanation
The SEQUENTIAL WRITE function writes the record specified in the cr field (T-FCB+32) to the file specified in the T-FCB. Blocking/deblocking is performed during a sequential write since the DMA buffer size is 128 bytes while the data block size is 256 bytes. Two records in the data block can be read immediately.

Note
The requested file must be opened and closed before and after write processing is performed, respectively. A REMOVE function must be executed before any MCT is to be removed from the PINE main unit.

Although this function performs the automatic MCT mount function if no MCT is mounted when it is executed, an error is signaled by MTOS because the specified file is not opened.

Reference
If an EOT (End Of Tape) error occurs during a sequential write, MIOS reports it to the application program by loading the A register and IOSRCD (F7CFH) field with nonzero value and 02H, respectively. If CP/M error reporting is suppressed, however, BDOS only signals a "BAD SECTOR" error.

Name — MAKE — BDOS No. 22

Function — Makes a directory entry for an MCT file.

Entry parameter — C = 16H
DE = T-FCB address

Return parameter — Return code 1
A = 00H: File creation successful
0FFH: Directory full

Return code 2

TOSRCD	A reg.	H reg.	Result
00H	00H	00H	Normal termination.
04H	FFH	04H	Mount unsuccessful.
05H	FFH	00H	MCT directory full.
07H	FFH	05H	The specified name is already assigned to an existing file.
09H	FFH	05H	Another file already open.
0FH	FFH	02H	R/O disk.
-	FFH	01H	Error in MIOS.

Explanation
The MAKE function catalogs the file specified by the T-FCB in the DE registers into the MCT directory in RAM. If no MCT is mounted, this function performs the automatic MOUNT function. Up to 12 files can be managed under the MTOS. A directory full error occurs if an attempt is made to create the 13th file.

Name RENAME BDOS No. 23

Function Renames an MCT file.

Entry parameter
C = 17H
DE = T-FCB address

Return parameter
Return code 1
A = ØØH: Normal termination
ØFFH: File not found

Return code 2

TOSRCD	A reg.	H reg.	Result
ØØH	ØØH	ØØH	Normal termination.
ØØH	FFH	ØØH	No file found.
Ø4H	FFH	Ø4H	Mount unsuccessful.
Ø6H	FFH	Ø3H	R/O file.
Ø8H	FFH	ØØH	File not found.
ØFH	FFH	Ø2H	R/O disk.
-	FFH	Ø1H	Error in MIOS.

Explanation
The RENAME function renames a file on drive H: (MCT). This function must be called with the T-FCB loaded with the old file name (dr to t3 fields) and a new file name (dØ to dl5 fields).

Note
A REMOVE function must be executed before an MCT is removed from the PINE main unit.

Name FILE ATTRIBUTE BDOS No. 3Ø

Function Sets MCT file attributes.

Entry parameter
C = lEH
DE = T-FCB address

Return parameter
Return code 1
A = ØØH: Normal termination
ØFFH: File not found

Return code 2

TOSRCD	A reg.	H reg.	Result
ØØH	ØØH	ØØH	Normal termination.
ØØH	FFH	ØØH	No file found.
Ø4H	FFH	Ø4H	Mount unsuccessful.
Ø8H	FFH	ØØH	File not found.
-	FFH	Ø1H	Error in MIOS.

Explanation
The FILE ATTRIBUTE function changes the attribute of an MCT file cataloged in the RAM directory. The file attribute is specified by the tl and t2 fields of the T-FCB.

MSB of tl = Ø: R/W file
= 1: R/O file
MSB of t2 = Ø: DIR file
= 1: SYS file

Note
Since the change is actually made in the RAM directory, a REMOVE function must be executed before the MCT is removed form the PINE.

Name RANDOM READ BDOS No. 33

Function Performs a random read of an MCT file.

Entry parameter
C = 21H
DE = T-FCB address

Return parameter
Return code 1
A = ØØH: Normal termination
ØFFH: MTOS error
Ø1H: File error
Ø4H: Random access error

Return code 2

TOSRCD	A reg.	H reg.	Result
ØØH	ØØH	ØØH	Normal termination.
Ø4H	FFH	Ø4H	Mount unsuccessful.
Ø9H	FFH	Ø5H	Another file already open.
ØAH	Ø1H	ØØH	File not open.
ØBH	FFH	Ø5H	Illegal record access.
ØCH	Ø1H	ØØH	EOF detected.
ØDH	Ø4H	ØØH	Random access error.
-	FFH	Ø1H	Error in MIOS.

Explanation
The RANDOM READ function reads a record specified in the rØ and r1 fields (T-FCB+33, 34) from the file addressed by the T-FCB and places the record in the DMA. This function performs the same function as the SEQUENTIAL READ function except that the read operation takes place at a particular record.

Name RANDOM WRITE BDOS No. 34

Function Performs a random write to an MCT file.

Entry parameter
C = 22H
DE = T-FCB address

Return parameter
Return code 1
A = ØØH: Normal termination
ØFFH: MTOS error
Ø1H: File not open

Return code 2

TOSRCD	A reg.	H reg.	Result
ØØH	ØØH	ØØH	Normal termination.
Ø4H	FFH	Ø4H	Mount unsuccessful.
Ø6H	FFH	Ø3H	R/O file.
Ø9H	FFH	Ø5H	Another file already open.
ØAH	Ø1H	ØØH	File not open.
ØBH	FFH	Ø5H	Illegal record access.
ØFH	FFH	Ø2H	R/O disk.
-	FFH	Ø1H	Error in MIOS.

Explanation
The RANDOM WRITE function writes a record to the file addressed by the T-FCB with a record number specified in the rØ and r1 fields (T-FCB+33, 34). This function performs the same function as the SEQUENTIAL WRITE function except that it starts a write at a particular record.

Name VERIFY BDOS No. 251

Function Verifies an MCT file.

Entry parameter
C = ØFBH
DE = T-FCB address

Return parameter
Return code 1
A = ØØH: Normal termination
ØFFH: Verify error
Return code 2

TOSRCD	A reg.	H reg.	Result
ØØH	ØØH	ØØH	Normal termination.
Ø4H	FFH	Ø4H	Mount unsuccessful.
Ø8H	FFH	ØØH	File not found.
Ø9H	FFH	Ø5H	File already open.
1ØH	FFH	ØØH	Verify error.
11H	FFH	Ø5H	File number corresponding to the specified file was not found in the directory.
-	FFH	Ø1H	Error in MIOS.

Explanation
The VERIFY function reads a file addressed by the T-FCB in the DE registers and checks the file on a block basis to verify that the values of the blocks match the values calculated when the file was stored.

When called, this function performs an automatic MCT mount function and opens it before verifying the specified file.

An error is signaled if the function is specified for a file which is already open.

Name REMOVE BDOS No. 251

Function Writes the RAM directory into MCT directory file.

Entry parameter
C = ØFCH

Return parameter
Return code 1
A = ØØH: Normal termination
= ØFFH: Remove error

Return code 2

TOSRCD	A reg.	H reg.	Result
ØØH	ØØH	ØØH	Normal termination.
Ø2H	FFH	Ø5H	MCT not mounted.
Ø3H	FFH	Ø5H	Cannot remove.
12H	FFH	Ø5H	Auto remove check error.
-	FFH	Ø1H	Error in MIOS.

Explanation
The REMOVE function writes the directory data from the RAM directory onto the MCT directory file.

No actual write takes place if no change has been made to the RAM directory (i.e., neither write, delete, nor rename operations have been performed).

Name MOUNT BDOS No. 253

Function Reads the MCT directory file to the RAM directory.

Entry parameter C = ØFDH

Return parameter Return code 1
A = ØØH: Normal termination
ØFFH: Mount error

Return code 2

TOSRCD	A reg.	H reg.	Result
ØØH	ØØH	ØØH	Normal termination.
Ø1H	FFH	Ø5H	Not removed.
Ø4H	FFH	Ø4H	Cannot mount.
-	FFH	Ø1H	Error in MIOS.

Explanation
The MOUNT function reads the MCT directory file and loads file control information into the system RAM directory.

Name READ TAPE ID BDOS NO. 254

Function Reads the ID number of the current MCT.

Entry parameter C = ØFEH

Return parameter Return code 1
A = ØØH: Not mounted
ØFFH: Normal termination

Return code 2

TOSRCD	A reg.	H reg.	Result
-	ØØH	-	MCT not mounted
-	FFH	-	Normal termination

Explanation
The READ TAPE ID function places the ID information pertaining to the currently mounted tape (columns Ø - 53 of the directory file's first block) into the DMA buffer. The function returns nothing when no MCT is mounted.

Name MAKE DIRECTORY BDOS No. 255

Function Creates a directory file on MCT.

Entry parameter
C = ØFFH
DE = T-FCB address

Return parameter
Return code 1
A = ØØH: Normal termination
ØFFH: MCT mounted
Return code 2

TOSRCD	A reg.	H reg.	Result
ØØH	ØØH	ØØH	Normal termination.
Ø1H	FFH	Ø5H	The function was called with the MCT mounted.
-	FFH	Ø1H	Error in MIOS.

Explanation
The MAKE DIRECTORY function rewinds the tape to the beginning to create a directory file.

The first 1Ø bytes of the directory file are copied from the the first 1Ø bytes of the T-FCB.
Bytes Ø - 7: Tape name (ASCII)
Bytes 8 and 9: Volume number (ASCII)

The number of directory entries is set to Ø. A RAM directory is also created by this function.

3.7.5.3 Return Codes from MTOS

Each MTOS or MIOS function returns several return codes. An MTOS function returns control to the calling program with error information loaded in the A and H registers and TOSRCD. The values placed in the A and H registers depends on the error code in the TOSRCD.

Furthermore, an MTOS function reports four types of CP/M error conditions, or returns control to the caller with the corresponding error codes loaded in A and H registers. See Subsection 3.2.4, "BDOS Errors" for the error report modes.

Table 3.7.6 lists the TOSRCD return codes and the meanings of the corresponding return codes that are placed in the A and H registers.

Table 3.7.7 lists the TOSRCD return codes returned by the MTOS functions. The contents of TOSRCD can be determined by referring to ØF7CEH.

Table 3.7.6 Meaning of Error Codes

TOSRCD return value	A reg.	H reg.	Meaning
ØØ	ØØ	ØØ	Normal termination.
Ø1	FF	Ø5	Not removed.
Ø2	FF	Ø5	Not mounted.
Ø3	FF	Ø5	Cannot remove.
*Ø4	FF	Ø5 (4)	Cannot mount.
Ø5	FF	ØØ	Directory full.
*Ø6	FF	ØØ (3)	R/O file.
Ø7	FF	Ø5	File already exists.
Ø8	FF	ØØ	File not found.
Ø9	FF	Ø5	File already open.
ØA	Ø1	ØØ	File not open.
ØB	FF	Ø5	Illegal record access.
ØC	Ø1	ØØ	EOF reached.
ØD	Ø4	ØØ	Random access error.
ØE	-	-	---
*ØF	FF	ØØ (2)	R/O drive.
1Ø	FF	ØØ	Verify error.
11	FF	Ø5	File number corresponding to the specified file not found in the directory.
12	FF	Ø5	Auto remove check error.

*: Numbers enclosed in parentheses identify the values returned when CP/M error reporting is suppressed.

Table 3.7.7 Errors Reported by MTOS Functions

MTOS Call No.	Function name	TOSRCD / Parameter	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
12 ○	Get version No.																			
13	Reset disk system																			
14	Select disk drive					○														
15	Open file	DE ← T-FCB				○				○	○								○	
16	Close file	DE ← T-FCB		○						○								○ *	○ *	
17	First search	DE ← T-FCB				○														
18	Next search																			
19	Delete file	DE ← T-FCB				○		○									○			
20	Sequential read	DE ← T-FCB				○					○	○	○	○						
21	Sequential write	DE ← T-FCB				○		○			○	○	○				○			
22	Create file	DE ← T-FCB				○	○		○		○						○			
23	Rename file	DE ← T-FCB				○		○		○							○			
24	Get login vector																			
25	Get login disk No.																			
26 ○	Set DMA address																			
27	Get allocation address																			

MTOS Call No.	Function name	TOSRCD / Parameter	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
28	Set write protect																			
29	Get R/O vector																			
30	Set file attribute					○				○										
31	Disk parameter block address																			
32 ○	Get user code																			
33	Random read	DE ← T-FCB				○					○	○	○	○	○					
34	Random write	DE ← T-FCB				○		○			○	○	○				○			
35	Compute file size	DE ← T-FCB																		
36	Set random record	DE ← T-FCB																		
37	Reset disk drive	DE ← Drive vector																		
38	Undefined																			
39	Undefined																			
40 ○	Random write with zero fill																			

MTOS Call No.	Function name	TOSRCD / Parameter	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	10	11	12
251	Verify	DE ← T-FCB				○				○	○							○	○	
252	Remove			○	○															○
253	Mount		○			○														
254	Read tape ID																			
255	Create tape directory	DE ← T-FCB	○																	

Notes 1: MTOS functions identified by an o mark are not supported by MTOS.
2: See related pages for details about the functions that return parameters in registers.
3: * denotes that the code is returned when verify if specified.

3.7.6 Using MIOS

3.7.6.1 Outline

MIOS is located between MTOS and MCT firmware and made up of 23 functions.

3.7.6.2 How to use MIOS (MCMTX)

Each MIOS function is described in the same format as a BIOS function.

<Entry address>
WBOOT + 78H or ØEB7BH
<Entry parameter>
B = Function number (ØØH - 18H)
Some other registers are loaded with parameter data depending on the function.
<Return parameter>
Different parameters are returned for different functions.
<Explanation>
26 functions are defined for the PINE OS. They are identified by the value in the B register

Descriptions of the parameters and the use of each MIOS function is given in 3.7.7.

Function number	Name	Function	Page
00H	MIRDST	Read tape status	II-320
01H	MIRDCT	Read tape counter	II-321
02H	MISTCT	Set tape counter	II-321
03H	MISTOP	Stop motor	II-321
04H	MIPLAY	Turn motor in play mode	II-322
05H	MIREC	Turn motor in record mode	II-322
06H	MIFF	Turn motor in fast feed mode	II-322
07H	MISREW	Turn motor in slow rewind mode	II-322
08H	MIREW	Turn motor in rewind mode	II-323
09H	MIFFTE	Fast feed to tape end	II-323
0AH	MIRWTT	Rewind to tape top	II-323
0BH	MIHDON	Turn head on	II-324
0CH	MIHDOF	Turn head off	II-324
0DH	MISKTP	Seek tape	II-324
0EH	MISTMP	Set move protect counter	II-325
0FH	MIRSMP	Reset move protect counter	II-325
10H	MIRDBL	Read one block	II-325
11H	MIWTBL	Write one block	II-327
12H	——	——	
13H	——	——	
14H	MIGTWD	Get write protect pin status	II-327
15H	MILEDON	Turn LED on	II-328
16H	MILEDOF	Turn LED off	II-328
17H	MISDAT	Save data area	II-328
18H	MILDAT	Restore data area	II-329

3.7.7 MIOS Functions

3.7.7.1 Outline

This subsection describes the entry and return parameters and error codes for each MIOS function. A table of error codes is located at the end of this subsection.

3.7.7.2 Programming note on the use of MIOS functions

(1) Reserve the buffer area for the MIOS (pointed to by TOSDMA) in a location between 8000H and 0FFFFH.
(2) The number of data bytes specified for reading or writing MCT blocks must be in the range from 4 to 270.
(3) Each MIOS function makes an MCT mount check before execution. If the MIOS function is called with no MCT mounted, it immediately returns control to the calling program with the A and C registers loaded with 08H respectively.

3.7.7.3 MIOS function details

The following pages describe the use of the MIOS functions.

Error information for each function is returned in the A and C registers as follows:

Areg.	Meaning
00H	Normal termination.
01H	Read error.
02H	Write error.
03H	Write protect error.
04H	Time over error.
05H	Seek error.
06H	Break error (CTRL/STOP pressed).
07H	Power off error (Power turned off).
08H	MCT cartridge not mounted.
FEH	Other errors.

Creg.	Meaning
01H	Head move error (error during a head move).
02H	Motor stop error (Motor stopped).
03H	Write protect error.
04H	Data error (Data bits 0 and 1 unrecognizable).
05H	CRC error
06H	Block mode error (invalid block mode).
07H	Tape error (Tape not found).
08H	MCT cartridge not mounted.

The values in the A and C registers are copied into BIOSERROR (ØF52BH) and IOSRCD (ØF7CFH) respectively. In the following descriptions, only the error information returned to the A register is listed.

Name MIRDST

Function Reads MCT Status.

Entry parameter B = ØØH

Return parameter
H = MCT status
A = ØØH: Normal termination
Ø8H: Mount error

Explanation
The MIRDST function reads the current microcassette status and places it in the H register. The bit assignments for the H register are as follows:

Bit	Contents
7	Head position (0 : OFF 1 : On)
6	Motor (0 : Stop 1 : Move)
5	Wind (0 : No 1 : Wind)
4	Fast feed (0 : No 1 : Fast feed)
3	Play (0 : No 1 : Play)
2	Record (0 : No 1 : Record)
1, 0	Reserved for the system.

Name MIRDCT

Function Reads the MCT counter.

Entry parameter B = 01H

Return parameter HL = Counter value
A = 00H: Normal termination
08H: Mount error

Explanation
The MIRDCT function reads the value of the 16-bit MCT counter and places it in the HL register pair. The counter increments when the tape is moved forward and decrements when it is moved backward.

Name MISTCT

Function Sets the MCT counter.

Entry parameter B = 02H
DE = Counter value to be set

Return parameter A = 00H: Normal termination
08H: Mount error

Explanation
The MISTCT function sets the MCT counter to the value specified in the DE register pair. The counter is set to 0000H immediately after an MCT is mounted. If no MCT is mounted, this function returns control to the caller with the A register loaded with 08H.

Name MISTOP

Function Stops MCT.

Entry parameter B = 03H

Return parameter A = 00H: Normal termination
08H: Mount error

Explanation
The MISTOP function stops the MCT motor after turning off the loudspeaker. The function turns off the loudspeaker not by issuing a command to the MCT but by switching off the loudspeaker on the PINE main unit.

Name MIPLAY

Function Drives the MCT motor in the play mode.

Entry parameter B = 04H

Return parameter A = 00H: Normal termination
08H: Mount error

Explanation
The MIPLAY function turns on the loudspeaker and rotates the MCT motor in the play mode. The read data signal is sent if the read/write head is on. The function turns on the loudspeaker not by issuing a command to the MCT but by switching on the loudspeaker on the PINE main unit.

Name MIREC

Function Drives the MCT motor in the record mode.

Entry parameter B = 05H

Return parameter A = 00H: Normal termination
08H: Mount error

Explanation
The MIREC function rotates the MCT motor in the record mode. Tape data is erased if the read/write head is on.

Name MIFF

Function Drives the MCT motor in the fast feed mode.

Entry parameter B = 06H

Return parameter A = 00H: Normal termination
08H: Mount error

Explanation
The MIFF function rotates the MCT motor in the fast feed mode.

Name MISREW

Function Drives the MCT motor at a slow speed in the backward direction.

Entry parameter B = 07H

Return parameter A = 00H: Normal termination
08H: Mount error

Explanation
The MISREW function rotates the MCT motor in the slow rewind mode. The read/write head is automatically turned off in this mode.

Name MIREW

Function Drives the MCT motor in the reverse direction.

Entry parameter B = 08H

Return parameter A = 00H: Normal termination
08H: Mount error

Explanation
The MIREW function rotates the MCT motor in the rewind mode. The read/write head is automatically turned off in this mode.

Name MIFFTE

Function Moves the MCT rapidly in the forward direction to the end.

Entry parameter B = 09H

Return parameter A = 00H: Normal termination
05H: Seek error
06H: Break error
07H: Power off error
08H: Mount error
0FEH: Other errors

Explanation
The MIFFTE function resets the move protect count and rotates the MCT motor in the fast feed mode until the MCT end is reached.

Name MIRWTT

Function Rewinds the MCT to the start.

Entry parameter B = 0AH

Return parameter A = 00H: Normal termination
05H: Seek error
06H: Break error
07H: Power off error
08H: Mount error
0FEH: Other errors

Explanation
The MIRWTT function resets the move protect count and moves the MCT backward up to the start of tape. The tape counter is set to 0000H when the start of tape is reached.

Name MIHDON

Function Turns on the MCT head.

Entry parameter B = ØBH

Return parameter A = ØØH: Normal termination
ØBH: Mount error
ØFEH: Other errors

Explanation
The MIHDON function turns on the MCT head (i.e., loads the head).

Name MIHDOF

Function Turns off the MCT head.

Entry parameter B = ØCH

Return parameter A = ØØH: Normal termination
ØBH: Mount error
ØFEH: Other errors

Explanation
The MIHDOF function turns off the MCT head (i.e., unloads the head).

Name MISKTP

Function Seeks for the specified MCT position.

Entry parameter B = ØDH
DE = Counter value

Return parameter A = ØØH: Normal termination
Ø6H: Break error
Ø7H: Power off error
ØBH: Mount error
ØFEH: Other errors

Explanation
The MISKTP function moves tape to the count specified by the DE register pair as follows:

Current count < desired count:	Moves tape in the forward direction (fast feed).
Current count = desired count:	Does nothing.
Current count < desired count:	Moves tape in the reverse direction (rewind).

The MCT motor must be stopped before a seek (MISTOP). After a seek operation is completed, the MCT head is automatically turned on.

Name MISTMP

Function Sets the MCT move protect count.

Entry parameter B = ØEH
DE = Move protect count

Return parameter A = ØØH: Normal termination
Ø8H: Mount error

Explanation
The MISTMP function sets the MCT move protect count (total count to which MCT can run). After a move protect count is set, the tape automatically stops when an attempt to seek past this count is made.

Example:
Current count 5ØØ
Move protect count 3ØØ
Assume that the tape has been rewound to count 4ØØ and an attempt is made to feed the tape up to count 7ØØ. Because the move protect count is decremented by 1ØØ when the tape is rewound to 4ØØ, the tape can now feed only 2ØØ counts and, therefore, stops at count 6ØØ.

Name MIRSMP

Function Resets the MCT protect count.

Entry parameter B = ØFH

Return parameter A = ØØH: Normal termination
Ø8H: Mount error

Explanation
The MIRSMP function resets the MCT move protect count (total count to which MCT can run).

Name	MIRDBL
Function	Reads one block of data from an MCT file.
Entry parameter	A = 1ØH B = Block number position offset C = Read mode DE = Bytes of data to be read HL = Block number TOSDMA (ØF351H) = Starting address of the buffer for storing read data
Return parameter	A = ØØH: Normal termination Ø1H: Read error Ø6H: Break error Ø8H: Mount error ØFEH: Other errors

Explanation
The bit assignments for the C register are as follows:

Bit	
7	Ø: Nonstop mode, 1: Stop mode
6 - 4	Not use
3 - Ø	Number of retries (ØØH - ØFH)

Load the DE register pair with the number of bytes to be read. The number must be in the range 4 (block identification field requires four bytes at least) to 27Ø. The address of the buffer pointed to by TOSDMA must be between 8ØØØH and ØFFFFH.

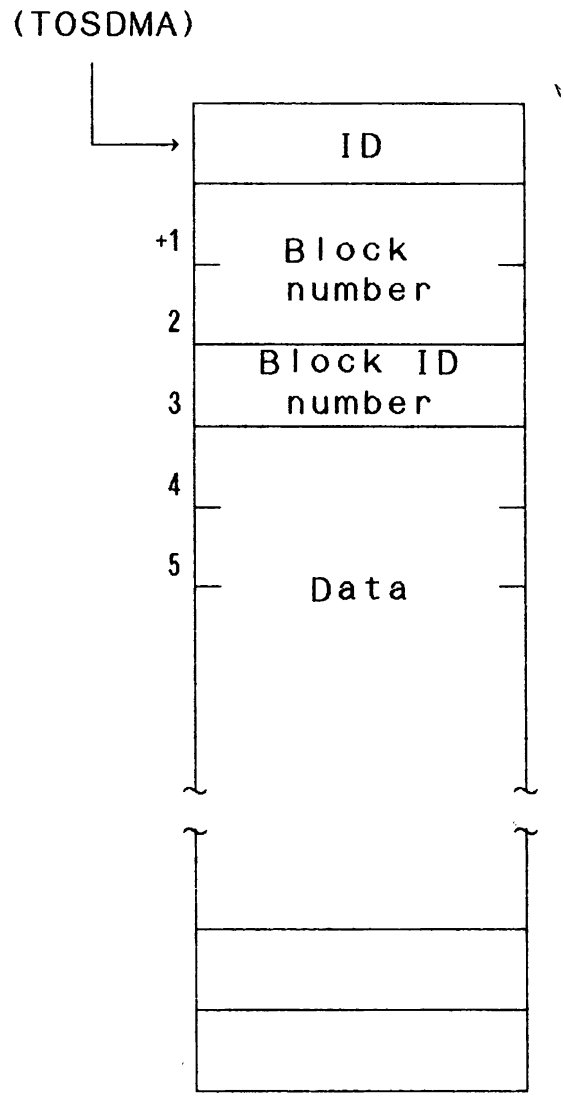
The MIRDBL function reads the number of bytes specified in DE from a block in the mode specified in C into the buffer addressed by TOSDMA. The block must satisfy the following conditions:

(Condition 1)
The value in the read data block ID field matches the ID code loaded at the beginning of the read buffer (addressed by TOSDMA) during execution of this function. Accordingly, the ID code of the block to be read must be loaded as the first read buffer data when executing the MIRDBL function. The ID code "?" matches any ID code (that is, no check is made on this field).

(Condition 2)
The block number of the read data placed in the two bytes starting at the address designated by (buffer address + value in A) matches the block number specified in HL before the execution of this function.

An error is generated if the following conditions occur:
(Block number of read block) + 2 < (Block number specified in HL)
(Block number of read block) > (Block number specified in HL)

Note
The A register (offset value) is normally loaded with Ø1H. This is because 1-block data is read into the buffer as shown below. The data block format is described in 3.7.2.



Name MIWTBL

Function Writes one block of data into an MCT file.

Entry parameter
B = 11H
C = Write mode
DE = Bytes of write data
TOSDMA = Starting address of the buffer storing write data

Return parameter
A = 00H: Normal termination
02H: Write error
03H: Write protect error
06H: Break error
08H: Mount error
0FEH: Other errors

Explanation
Bit 7 of the C register identifies the tape access mode. A 0 in the bit indicates the nonstop mode; a 1 indicates the stop mode.

Load TOSDMA with a buffer address in the range from 8000H to 0FFFFH. Load the DE register pair with the number of bytes to be written. The number must be in the range from 4 (block identification field requires four bytes at least) to 270.

The MIWTBL function writes on MCT the number of bytes specified in DE from the buffer addressed by TOSDMA in the mode specified in C. The write data is written the number of times specified in BLKWRTNO (0F2FDH). When the data is to be written two or more times, the function refers to the block ID number loaded in the location (Address pointed to by TOSDMA) + 3.

Name MIGTWD

Function Reads the MCT write protect pin status.

Entry parameter
B = 14H

Return parameter
A = 00H: Not protected
03H: Protected
08H: Mount error

Explanation
The MIGTWD function reads the MCT write protect pin status and places it in the A register.

Name MILEDON

Function Turns on the MCT LED.

Entry parameter B = 15H

Return parameter A = ØØH: Normal termination
Ø8H: Mount error

Explanation
The MILEDON function turns on the LED (labeled FREE) on the MCT. The FREE LED is used by the system to identify the mount or remove state of the MCT and cannot be used by the user.

Name MILEDOF

Function Turns off the MCT LED.

Entry parameter B = 16H

Return parameter A = ØØH: Normal termination
Ø8H: Mount error

Explanation
The MILEDOF function turns off the LED (labeled FREE) on the MCT. The FREE LED is used by the system to identify the mount or remove state of MCT and cannot be used by the user.

Name MISDAT

Function Saves the system data pertaining to the MCT into the data area on the PINE main unit.

Entry parameter B = 17H

Return parameter A = ØØH: Normal termination
Ø8H: Mount error

Explanation
The MISDAT function saves the system data pertaining to the MCT cartridge (MICROCASSETTE DRIVE) into the data area MCTSVDAT (ØF7FCH) on the main unit.

This function is used by the system to keep the MCT cartridge status when the power of the PINE main unit is turned off. The user cannot use this function.

Name MILDAT

Function Restores the system data pertaining to the MCT from the data area on the PINE main unit.

Entry parameter B = 18H

Return parameter A = ØØH: Normal termination
ØØH: Mount error

Explanation
The MILDAT function restores the system data pertaining to the MCT cartridge from the data area MCTSVDAT (ØF7FCH) on the PINE main unit.

This function is executed by the system, when the PINE main unit is powered on, to restore the MCT cartridge status of immediately before the last power off. The user cannot use this function.

3.7.7.4 Return codes from MIOS

Control returns from MIOS with the A and C registers loaded with return codes. The return codes are also placed in BIOSERROR and IOSRCD.

BIOSERROR is loaded with an error code common to the BIOS while lOSRCD is loaded with error information returned from the firmware.

BIOSERROR (ØF52BH)
- Return information from BIOS
  = ØØH: Normal termination
    Ø1H: Read error
    Ø2H: Write error
    Ø3H: Write protect error
    Ø4H: Time out error
    Ø5H: Seek error
    Ø6H: Break error
    Ø7H: Power off error
    Ø8H: Mount error
    ØFE: Other errors

IOSRCD (ØF7CFH)
- Return information from MIOS
  = ØØH: Normal termination
    Ø1H: Head move error (error during a head move)
    Ø2H: Motor stop error (motor stopped)
    Ø3H: Write protect error
    Ø4H: Data error (data bits Ø and 1 unrecognizable)
    Ø5H: CRC error
    Ø6H: Block mode error (invalid block mode)
    Ø7H: Tape error (tape not found)

MIOS Function number	Function	BIOSERROR								IOSRCD							
		Code	1	2	3	4	5	6	7	FE	1	2	3	4	5	6	7
00H	MIRDST																
01H	MIRDCT																
02H	MISTCT																
03H	MISTOP																
04H	MIPLAY																
05H	MIREC																
06H	MIFF																
07H	MISREW									○	○						*
08H	MIREW									○	○						
09H	MIFFTE					○	○	○	○		○					○	
0AH	MIRWTT					○	○	○	○		○					○	
0BH	MIHDON									○	○						
0CH	MIHDOF									○	○						
0DH	MISKTP						○	○	○		○		○			○	
0EH	MISTMP																
0FH	MIRSMP																
10H	MIRDBL		○					○			○	○		○	○	○	
11H	MIWTBL			○	○			○			○	○	○				
14H	MIGTWD																
15H	MILEDON																
16H	MILEDOF																
17H	MISDAT																
18H	MILDAT																

*: The error is generated because the head is automatically turned off in the rewind mode.

3.7.8 Work Areas Related to MCT Processing

The work areas used during MCT processing are listed below.

TOSCTL (ØEFBBH) 1 byte
- MTOS control flag

This flag controls the MTOS operation status. See 3.7.4, "Using MTOS."

STOSCTL (ØEFBCH) 1 byte
- MTOS control subflag

Bit 7: Specifies force mount.
1: Force mount Ø: Non-force mount
Bit 6: Specifies force remove.
1: Force remove Ø: Non-force remove
Bits 5 - Ø: Don't care.

This flag is not used by the PINE OS.

TOSSTS (ØEFBDH) 1 byte
- MTOS status flag

Bit 7: MTOS directory search bit
1: MTOS Ø: BDOS
Bit 6: Motor stop bit (used when an error occurs)
1: No Ø: Stop
Bit 5: MTOS error bit
1: MTOS Ø: BDOS
Bit 4: Directory counter correction bit
1: Correct Ø: No
Bit 3: Bad select report bit
1: No Ø: Report
Bits 2- Ø: Don't care.

This flag is used by the system.

VERFDFLT (ØEFBEH) 1 byte
- Verify flag for write (for default setting)

= ØØH: No
= Nonzero: Verify

See 3.7.4, "Using MTOS."

VERFFG (ØEFBFH) 1 byte
- Verify flag for write

= ØØH: No
= Nonzero: Verify

See 3.7.4, "Using MTOS."

MCTDETCH (ØEFCØH) 1 byte
- MCT detach control flag

= ØØH: No
= Nonzero: Detach

When this flag is set to a value other than ØØH, the MCT is forced into the remove state if no MCT cartridge is mounted at power-on time. The flag is initialized to ØFFH.

TAPMOD (ØF2E1H) 1 byte
- MCT status flag

= ØØH: Remove state
= Ø1H: Mount state

MCTDCD (ØF2E2H) 1 byte
- MCT drive code

Used to check whether the drive code in the FCB identifies an MCT. The flag is initialized to Ø7H (drive H:).

ROMCT (ØF2E3H) 1 byte
- Indicates the MCT R/O status.
     = ØØH: R/W
     = Ø1H: R/O

DFTATR (ØF2E4H) 1 byte
- Tape access attribute flag (for default setting)
Loaded with the default data in TACATR. See 3.7.4, "Using MTOS,"

WRTATR (ØF2E5H) 1 byte
- Not used.

SRDATR (ØF2E6H) 1 byte
- The flag controlling the tap access attribute for system read
The bit assignments are identical to those of TACMOD. This flag is referenced when reading the header or EOF area on tape.
The flag is initialized to ØC2H.

DRP (ØF2E7H) 2 bytes
- Directory read start counter
This counter is initialized to ØØ3CH.

DWP (ØEFB9H) 2 bytes
- Directory write start counter
This counter is initialized to ØØ4DH.

DEP (ØF2EBH) 2 bytes
- Directory end counter
This counter is initialized to ØØD9H.

DMR (ØEFBDH) 2 bytes
- The margin between the directory and the file area.
The initial value is ØØ68H.

IFG (ØF2EFH) 2 bytes
- Inter-file gap
The initial value is ØØ68H.

SKG (ØF2F1H) 2 bytes
- Inter-file-seek gap
The initial value is ØØ34H.

ERG (ØF2F3H) 2 bytes
- Erasing gap generated during file creation.
The initial value is ØØ34H.

LERG (ØF2F5H) 2 bytes
- Trailing erasing gap generated during file close.
The initial value is Ø04EH.

BKG (ØF2F7H) 2 bytes
- Inter-block-number gap (not used)
The initial value is ØØØ2H.

PRIPG (ØF2F9H) 2 bytes
- Inter-block gap (not used)
The initial value is ØØØ4H.

CCDFLT (ØF2FBH) 1 byte
- Default count for tape counter correction
The initial value is Ø5H.

CRCCNT (ØF2FCH) 1 byte
- Tape counter correction counter
The value of this counter is decremented by one each time a file is created. When the counter value reaches ØØH, the machine rewinds the tape to the top. The correction value is indicated by CCDFLT.

BLKWRTNO (ØF2FDH) 1 byte
- Number of times a block plus 1 is to be written.
The initial value is Ø3H (twice). See 3.7.4, "Using MTOS."

LMCTRED (ØF2FEH) 4 bytes
- Error return point entered during a MCT read through BIOS.
Loaded with the following instruction codes:

```
LD   A, ØFFH
JR   LMCTWRT+2
```

LMCTWRT (ØF3Ø2H) 6 bytes
- Error return point entered during MCT write through BIOS.
Loaded with the following instruction codes:

```
LD   A, ØFFH
LD   (BIOSERROR), A
RET
```

TOSDMA (ØF351H) 2 bytes
- MCT DMA buffer starting address
The initial value points to TOSBUF (ØEB97H).

ALCVAD (ØF353H) 2 bytes
- MCT allocation vector starting address

OPNMOD (ØF355H) 1 byte
- Current MCT open status
  = ØØH: Not open
    Ø1H: Read mode open
    Ø2H: Write mode open

MISSTS (ØF356H) 1 byte
- MIOS status flag
  Bit 7: Move protect bit
    1: Move protect Ø: No
  Bits 6 - Ø: Reserved for the system
When the move protect bit is set on, the machine is move-protected, that is, it is automatically stopped after feeding the tape for the tape count specified in TAPMVC (ØF35AH). The initial setting is move protect on.

BMDCNT (ØF357H) 1 byte
- Block mode check retry count
The number of read retires made when a block mode error occurred during a block read. The default value is Ø2H.

RRBKMD (ØF358H) 1 byte
- Random read block mode
This area is not used by the PINE OS.

RNIDPT (ØF359H) 1 byte
- Random access ID pointer
This area is not used by the PINE OS.

TAPMVC (ØF35AH) 2 bytes
- System move protect count
The initial value is ØØC8H.

TOSRCD (ØF7CEH) 1 byte
- MTOS return code area
     = ØØH: Normal termination
       Ø1H: Not removed
       Ø2H: Not mounted
       Ø3H: Cannot remove
       Ø4H: Cannot mount
       Ø5H: Directory full
       Ø6H: R/O file
       Ø7H: File already existed
       Ø8H: File not found
       Ø9H: File already open
       ØAH: File not open
       ØBH: Record number error
       ØCH: EOF reached
       ØDH: Random access error
       ØEH: not use, reserved for the system
       ØFH: R/O drive
       1ØH: Verify error
       11H: File number error
       12H: Auto remove check error
See 3.7.5, "MTOS Functions" for details.

IOSRCD (ØF7CFH) 1 byte
- MIOS return code area
     = ØØH: Normal termination
       Ø1H: Head error
       Ø2H: Motor stop error
       Ø3H: Write protect error
       Ø4H: Data error
       Ø5H: CRC error
       Ø6H: Block mode error
       Ø7H: Tape error
       Ø8H: Mount error
This area is loaded with the return code from the MCT cartridge.
See 3.7.6, "Using MIOS" for details.

CURMCT (ØF7DØH) 1 byte
- Current MCT drive code
Loaded with the MCT drive code when the current drive is the MCT drive.

TACATR (ØF7D1H) 1 byte
- Tape access attribute
Indicates the stop or nonstop mode and the number of retries.
See 3.5.4, "Using MTOS."

TACMOD (ØF7D2H) 1 byte
- Tape access mode
See 3.5.4, "Using MTOS."

BLKNO (ØF7D3H) 2 bytes
- Current block number area

RECNO (ØF7D5H) 2 bytes
- Current record number area

BLKREC (ØF7D7H) 2 bytes
- DMA buffer first record number

```
RECEXT (ØF7D9H)  1 byte
- TOSDMA record presence flag
     = ØØH: No record present
       Ø1H: First record present
       Ø2H: Second record present
       Ø3H: Two records present

ORECEXT (ØF7DAH)  1 byte
- Old TOSDMA record presence flag

DIRPNT (ØF7DBH)  1 byte
- Directory file counter

FILPNT (ØF7DCH)  1 byte
- Current file pointer in the directory

SRCADD (ØF7DDH)  2 bytes
- T-FCB address referenced during file search

MAXBLK (ØF7DFH)  2 bytes
- Current file maximum block number

MAXREC (ØF7E1H)  2 byte
- Current file maximum record number

REMFLG (ØF7E3H)  1 byte
- Remove flag
     = ØØH: Remove not required.
     = Nonzero: Remove required (for directory update).

SKCKCT (ØF7E4H)  1 byte
- Seek check retry counter

LSTCNT (ØF7E5H)  2 bytes
- Tape last counter

USERCNT (ØF7E7H)  2 bytes
- User seek counter
When IDFLG(ØFA13H) bit 6 is 1, the value in this area is used to
find the tape count for the last free area.  The counter value of
the last free area is usually calculated using the data in the
directory.

RWMOD (ØF7E9H)  1 byte
- Read/write mode flag
     Bit 7: Access type bit
            Ø: Sequential access    1: Random access
     Bits 6 - 1: Don't care.
     Bit Ø: Read/write bit
            Ø: Read      1: Write

ALCVEC (ØF7EAH)  8 bytes
- MCT allocation vector

REGBC (ØF7F2H)  2 bytes
- BC register save area

REGDE (ØF7F4H)  2 bytes
- DE register save area
```

BSTCNT (ØF7F6H) 2 bytes
- Block start count during read

RNDNO (ØF7F8H) 2 bytes
- Random record number (not used)

RNDCNT (ØF7FAH) 2 bytes
- Random record tape counter (not used)

MCTSVDAT (ØF7FCH) 32 bytes
- MCT cartridge system data save area

MCTDTSV (ØF9BAH) 4Ø bytes
- Reserved for the system

IDTOP (ØF9E2H) 53 bytes
- Tape ID field area
This area constitutes a part of the RAM directory area or the directory ID field. See 3.7.2, "File Control."

DIRTOP (ØFA17H) 384 bytes
- Tape directory area
This area is the second and third blocks of the RAM directory area and contain control information for 12 files. See 3.7.2, "File Control."

TOSDMA (ØFB97H) 26Ø bytes
- MCT DMA buffer

3.8 Disk Storage

3.8.1 General

The PINE assigns memory-related I/O devices to disk drives to allow the user to handle I/O devices easily.

The I/O device and disk drive assignments are defined as follows:

Drive	A:	RAM disk (internal, external)
	B:	ROM capsule 1
	C:	ROM capsule 2
	D:	External disk (floppy disk)
	E:	External disk
	F:	External disk
	G:	External disk
	H:	Microcassette
	I:	RAM cartridge
	J:	RAM cartridge 1
	K:	RAM cartridge 2

See Chapter 5, "I/O Operations" for details of the interfaces to cartridges, disk units, and external RAM disks.

3.8.2 Logical to Physical Drive Assignment

On the PINE, the user can reassign logical drives to physical drives as needed. The logical to physical drive assignment settings are listed in the previous subsection. Only the association between drive H: and microcassette is fixed and cannot be changed.

3.8.2.1 How to change an assignment

The assignment of logical drives to physical drives are controlled through a table located in the system area DISKTBL. Logical drives can be reassigned by rewriting this table. More specifically, logical drives can be reassigned to physical drives by rewriting physical codes in DISKTBL. The reassignments remain valid until the next reset (BOOT) operation.

Example: To assign drive B: to the floppy disk drive, change the contents of DISKTBL + 1 (ØF1ØØH) from Ø1H to Ø3H.

DISKTBL (ØFØFFH) 11 bytes
- Logical-to-physical drive assignment table

Address	Initial value	Assigned logical drive
ØF0FFH	ØØH	A:
ØF1ØØH	Ø1H	B:
ØF1Ø1H	Ø2H	C:
ØF1Ø2H	Ø3H	D:
ØF1Ø3H	Ø4H	E:
ØF1Ø4H	Ø5H	F:
ØF1Ø5H	Ø6H	G:
ØF1Ø6H	Ø7H	H:
ØF1Ø7H	Ø8H	I:
ØF1Ø8H	Ø9H	J:
ØF1Ø9H	ØAH	K:

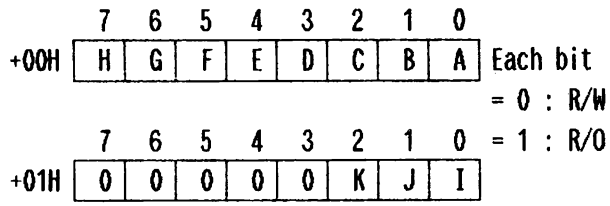
Physical drive code
= ØØH: RAM disk
  Ø1H: ROM capsule 1
  Ø2H: ROM capsule 2
  Ø3H: Floppy disk
  Ø4H: Floppy disk
  Ø5H: Floppy disk
  Ø6H: Floppy disk
  Ø7H: Microcassette
  Ø8H: RAM cartridge
  Ø9H: ROM cartridge 1
  ØAH: ROM cartridge 2

3.8.2.2 Programming notes

The following points must be taken into consideration when reassign logical drives to physical drives:

(1) Do not specify values ØBH through ØFFH as physical drive codes. Any drive assigned with one of these values cannot be selected.
(2) The 11 bytes in DISKTBL must contain unique values.
(3) When rewriting the DISKTBL table, also rewrite DISKROV (ØF1ØCH), a vector identifying the R/O state of the drives. The reassignments are made valid when a warm boot is performed.
(4) When rewriting the DISKTBL table, also rewrite the areas in the table that hold the addresses of ROM capsules 1 and 2 in (ROMCPNO1 at ØF1ØAH and ROMCPNO2 at ØF1ØBH).
(5) Do not change the assignment of drive H:.

DISKROV (ØF1ØCH) 2 bytes
- Disk R/O vector



The bits in the disk R/O vector are associated with logical drives A: through K: as shown above. Logical drives B:, C:, J:, and K: are initially set to R/O (Ø6H, Ø6H).

ROMCPNO1 (ØF1ØAH) 1 byte
- Drive code for ROM capsule 1
Loaded with the offset of ROM capsule 1 with respect to the beginning of DISKTBL. The initial value is Ø1H.

ROMCPNO2 (ØF1ØBH) 1 byte
- Drive code for ROM capsule 2
Loaded with the offset of ROM capsule 2 with respect to the beginning of DISKTBL. The initial value is Ø2H.

```
                        ;       ************************************************
                        ;               CHANGE DISK TRANSLATE TABLE
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is changing disk
                        ;               translate table.
                        ;
                        ;               Calling sequence.
                        ;                       SAMPLE12 (Logical):=(Physical)
                        ;                  (ex. SAMPLE12 A:=B)
                        ;
                        ;       <> assemble condition <>
                        ;
                        ;               .Z80
                        ;
                        ;       <> loading address <>
                        ;
                        ;               .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
                        ;       BIOS entry
                        ;
EB03                    WBOOT           EQU     0EB03H          ; Warm Boot entry
EB0C                    CONOUT          EQU     WBOOT   +09H    ; Console out entry
                        ;
                        ;       System area
                        ;
F0FF                    DISKTBL         EQU     0F0FFH          ; Top of User BIOS
F10C                    DISKROV         EQU     0F10CH          ; RBIOS1 loading addr
F10A                    ROMCPNO1        EQU     0F10AH          ; RBDOS1 loading addr
                        ;
0080                    INDATA          EQU     00080H
                        ;
                                .stant
                        ;
000D                    CR              EQU     0DH
000A                    LF              EQU     0AH
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This program is changing disk translate table.
                        ;                       First parameter is Logical drive.
                        ;                       Second parameter is Physical drive.
                        ;
                        ;
0100                    MAIN:
0100    31 1000                 LD      SP,1000H        ; Set stack pointer.
                        ;
                        ;
0103    CD 016E                 CALL    GETDATA         ; Get user inputed data.
0106    DA 0114                 JP      C,ERROR_END     ; If parameter end, then end.
                        ;
0109    CD 019B                 CALL    SETMSG          ; Set return message data.
010C    CD 011D                 CALL    CHANGE          ; Change disk translate table.
                        ;
010F    21 026E                 LD      HL,MSG01        ; Normal end message.
0112    18 03                   JR      PEND            ;
                        ;
0114                    ERROR_END:
0114    21 0294                 LD      HL,MSG02        ; Error end message.
0117                    PEND:
0117    CD 0262                 CALL    DSPMSG          ; Display message.
011A    C3 EB03                 JP      WBOOT           ;
                        ;       ************************************************
                        ;               CHANGE DISK DRIVE TABLE
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               B  : Logical drive NO.
                        ;               C  : Physical drive NO.
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;               If parameter error, then WBOOT.
                        ;
011D                    CHANGE:
011D    58                      LD      E,B             ; Get target logical drive NO. address.
011E    16 00                   LD      D,00H           ;  DISKTBL + BC --> HL
0120    21 F0FF                 LD      HL,DISKTBL      ;
0123    19                      ADD     HL,DE           ;
0124    71                      LD      (HL),C          ; Set new physical drive No.
                        ;
0125    1C                      INC     E               ; Calculate read only bit position.
0126    43                      LD      B,E             ; Drive No. (1 to 11)
0127    5A                      LD      E,D             ; DE is 0000H.
0128    37                      SCF                     ; Carry on.
                        ;
0129                    LOOP10:
0129    CB 13                   RL      E               ; Shift left by 1 bit.
012B    CB 12                   RL      D               ;
012D    10 FA                   DJNZ    LOOP10          ; Loop by Drive No.
                        ;
012F    21 F10C                 LD      HL,DISKROV      ; Disk R/O table.
0132    79                      LD      A,C             ; If R/O disk (1,2,9,10)
0133    FE 01                   CP      01H             ;  then set read only bit.
```

```
0135    28 0C                   JR      Z,RO_DISK       ;
0137    FE 02                   CP      02H             ;
0139    28 08                   JR      Z,RO_DISK       ;
013B    FE 09                   CP      09H             ;
013D    28 04                   JR      Z,RO_DISK       ;
013F    FE 0A                   CP      0AH             ;
0141    20 09                   JR      NZ,RW_DISK      ;
                        ;
0143                    RO_DISK:
0143    7E                      LD      A,(HL)          ; Set R/O bit.
0144    B3                      OR      E               ;
0145    77                      LD      (HL),A          ;
0146    23                      INC     HL              ;
0147    7E                      LD      A,(HL)          ;
0148    B2                      OR      D               ;
0149    77                      LD      (HL),A          ;
014A    18 0B                   JR      ROM_CAPSEL      ;
                        ;
014C                    RW_DISK:
014C    3E FF                   LD      A,0FFH          ; Reset read only bit.
014E    AB                      XOR     E               ;
014F    A6                      AND     (HL)            ;
0150    77                      LD      (HL),A          ;
0151    23                      INC     HL              ;
0152    3E FF                   LD      A,0FFH          ;
0154    AA                      XOR     D               ;
0155    A6                      AND     (HL)            ;
0156    77                      LD      (HL),A          ;
                        ;
0157                    ROM_CAPSEL:
0157    79                      LD      A,C             ; Key code. (Disk code)
0158    B7                      OR      A               ; RAM disk.
0159    C8                      RET     Z               ; Yes.
015A    FE 03                   CP      03H             ; ROM capsel 1 or 2?
015C    D0                      RET     NC              ; No.
                        ;
015D    06 0B                   LD      B,11            ; Table length.
015F    21 F0FF                 LD      HL,DISKTBL      ; Table address.
0162    CD 0252                 CALL    SEARCH          ; Search data.
0165    D8                      RET     C               ; Not found.
                        ;
0166    06 00                   LD      B,00H           ; Set position No. in table.
0168    21 F109                 LD      HL,ROMCPNO1-1   ;
016B    09                      ADD     HL,BC           ;
016C    77                      LD      (HL),A          ;
                        ;
016D    C9                      RET
                        ;
                        ;       ************************************************
                        ;               GET PARAMETER FROM BUFFER
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Get disk translate data from
                        ;               CCP input buffer.
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               CY-flag : Return parameter.
                        ;                  =0 -- Normal return
                        ;                  =1 -- Error return
                        ;                        (Parameter error)
                        ;               Case of Normal return
                        ;                 B : Logical drive NO.
                        ;                       (00H to 0AH)
                        ;                 C : Physical drive NO.
                        ;                       (00H to 0AH)
                        ;
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;               Case of normal retyrn, this routine
                        ;               sets finishing message.
                        ;
016E                    GETDATA:
016E    21 0080                 LD      HL,INDATA       ; Parameter inputed address.
0171    7E                      LD      A,(HL)          ; Get inputed byte number.
0172    FE 05                   CP      05H             ; Parameter count check.
0174    D8                      RET     C               ; Error. then return with carry on.
                        ;
0175    23                      INC     HL              ; Pointer update
0176    23                      INC     HL              ;
0177    7E                      LD      A,(HL)          ; Get first data.
0178    E6 DF                   AND     0DFH            ; Capital change.
017A    D6 41                   SUB     'A'             ; 'A' to 'K' check.
017C    D8                      RET     C               ; If error,
017D    FE 0B                   CP      'L'-'A'         ;  then return with carry on.
017F    3F                      CCF                     ;
0180    D8                      RET     C               ;
                        ;
0181    47                      LD      B,A             ; First parameter --> B
                        ;
0182    23                      INC     HL              ; Pointer update.
0183    7E                      LD      A,(HL)          ; ':=' check.
0184    FE 3A                   CP      ':'             ;
0186    37                      SCF                     ;
0187    C0                      RET     NZ              ;
0188    23                      INC     HL              ;
0189    7E                      LD      A,(HL)          ;
018A    FE 3D                   CP      '='             ;
018C    37                      SCF                     ;
018D    C0                      RET     NZ              ;
                        ;
018E    23                      INC     HL      ; Pointer update.
018F    7E                      LD      A,(HL)  ; Get second parameter.
```

```
0190    E6 DF                   AND     0DFH            ;
0192    D6 41                   SUB     'A'             ; 'A' to 'K' check.
0194    D8                      RET     C               ;
0195    FE 0B                   CP      'L'-'A'         ;
0197    3F                      CCF                     ;
0198    D8                      RET     C               ;
                        ;
0199    4F                      LD      C,A             ; Second parameter --> C
019A    C9                      RET                     ;
                        ;
                        ;       ************************************************
                        ;               SET DRIVE NAME
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Set drive name to message area.
                        ;
                        ;       <> entry parameter <>
                        ;               B  : Drive No.
                        ;               C  : Table No.
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               BC
                        ;
                        ;       CAUTION :
                        ;
019B                    SETMSG:
019B    C5                      PUSH    BC              ; Save register.
019C    78                      LD      A,B             ;
019D    C6 41                   ADD     A,'A'           ; Get ASCII data.
019F    32 027B                 LD      (DRIVE),A       ; Set drive code.
                        ;
01A2    21 01C1                 LD      HL,DEVICETBL    ; Data search table.
01A5    79                      LD      A,C             ;
01A6    87                      ADD     A,A             ;
01A7    5F                      LD      E,A             ;
01A8    16 00                   LD      D,00H           ;
01AA    19                      ADD     HL,DE           ; Get target address in table. (HL)
01AB    5E                      LD      E,(HL)          ;
01AC    23                      INC     HL              ;
01AD    56                      LD      D,(HL)          ;
                        ;
01AE    EB                      EX      DE,HL           ; Message top address. (HL)
01AF    11 0281                 LD      DE,DEVICE       ; Data setting address.
01B2    4E                      LD      C,(HL)          ; Get data count.
01B3    23                      INC     HL              ;
01B4    06 00                   LD      B,00H           ;
01B6    ED B0                   LDIR                    ;
01B8    21 024E                 LD      HL,ENDMARK      ; Set end data.
01BB    0E 04                   LD      C,04H           ;
01BD    ED B0                   LDIR                    ;
                        ;
01BF    C1                      POP     BC              ; Restore register.
01C0    C9                      RET                     ;
                        ;
                        ;       Message table & data
                        ;
01C1                    DEVICETBL:
01C1    01D7                    DW      DV_A            ; RAM disk
01C3    01E0                    DW      DV_B            ; ROM capsel 1
01C5    01ED                    DW      DV_C            ; ROM capsel 2
01C7    01FA                    DW      DV_D            ; FDD 1
01C9    0200                    DW      DV_E            ; FDD 2
01CB    0206                    DW      DV_F            ; FDD 3
01CD    020C                    DW      DV_G            ; FDD 4
01CF    0212                    DW      DV_H            ; MCT cartridge
01D1    0220                    DW      DV_I            ; RAM cartridge
01D3    022E                    DW      DV_J            ; ROM cartridge 1
01D5    023E                    DW      DV_K            ; ROM cartridge 2
                        ;
01D7                    DV_A:
01D7    08 52 41 4D             DB      8,'RAM disk'
01DB    20 64 69 73
01DF    6B
01E0                    DV_B:
01E0    0C 52 4F 4D             DB      12,'ROM capsel 1'
01E4    20 63 61 70
01E8    73 65 6C 20
01EC    31
01ED                    DV_C:
01ED    0C 52 4F 4D             DB      12,'ROM capsel 2'
01F1    20 63 61 70
01F5    73 65 6C 20
01F9    32
01FA                    DV_D:
01FA    05 46 44 44             DB      5,'FDD 1'
01FE    20 31
0200                    DV_E:
0200    05 46 44 44             DB      5,'FDD 2'
0204    20 32
0206                    DV_F:
0206    05 46 44 44             DB      5,'FDD 3'
020A    20 33
020C                    DV_G:
020C    05 46 44 44             DB      5,'FDD 4'
0210    20 34
0212                    DV_H:
0212    0D 4D 43 54             DB      13,'MCT cartridge'
0216    20 63 61 72
021A    74 72 69 64
021E    67 65
0220                    DV_I:
0220    0D 52 41 4D             DB      13,'RAM cartridge'
0224    20 63 61 72
0228    74 72 69 64
022C    67 65
```

```
022E                    DV_J:
022E    0F 52 4F 4D             DB      15,'ROM cartridge 1'
0232    20 63 61 72
0236    74 72 69 64
023A    67 65 20 31
023E                    DV_K:
023E    0D 52 4F 4D             DB      13,'ROM cartridge 2'
0242    20 63 61 72
0246    74 72 69 64
024A    67 65 20 32

                        ;
024E                    ENDMARK:
024E    2E 0D 0A 00             DB      '.',CR,LF,00H
                        ;
                        ;       ************************************************
                        ;               SEARCH TABLE
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               A    : Key code
                        ;               B    : Length of table
                        ;               HL   : Top of table address
                        ;       <> return parameter <>
                        ;               CY   : Return information
                        ;                      =0 : Found
                        ;                      =1 : Not found
                        ;               A    : Key No.
                        ;               HL   : Object item position in table
                        ;       <> preserved registers <>
                        ;               BC,DE
                        ;
                        ;       CAUTION :
                        ;
0252                    SEARCH:
0252    C5                      PUSH    BC              ; Save register.
0253    0E 00                   LD      C,00H           ; Initialize key NO.
0255                    SEAR1:
0255    BE                      CP      (HL)            ; Compare key with item.
0256    28 07                   JR      Z,SEAR2         ; If found, then exit,
0258    23                      INC     HL              ;  else check next item.
0259    0C                      INC     C               ; Key NO. update.
025A    10 F9                   DJNZ    SEAR1           ; If not end of table, then repeat,
025C    0E FF                   LD      C,0FFH          ;  else set not found sign.
025E    37                      SCF                     ;
025F                    SEAR2:
025F    79                      LD      A,C             ; Set key NO.
0260    C1                      POP     BC              ; Restore register.
0261    C9                      RET                     ;
                        ;
                        ;       ************************************************
                        ;               DISPLAY MESSAGE
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Display message until find 00H.
                        ;
                        ;       <> entry parameter <>
                        ;               HL   : Message data top address.
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0262                    DSPMSG:
0262    4E                      LD      C,(HL)          ; Get display data.
0263    0C                      INC     C               ; Data is 00H?
0264    0D                      DEC     C               ;
0265    C8                      RET     Z               ; Yes.
                        ;
0266    E5                      PUSH    HL              ;
0267    CD EB0C                 CALL    CONOUT          ; Display data.
026A    E1                      POP     HL              ;
026B    23                      INC     HL              ; Pointer update.
026C    18 F4                   JR      DSPMSG          ;
                        ;
                        ;       Message and Work area
                        ;
026E                    MSG01:
026E    43 68 61 6E             DB      'Change drive '
0272    67 65 20 64
0276    72 69 76 65
027A    20
027B                    DRIVE:  DS      1
027C    3A 20 74 6F             DB      ': to '
0280    20
0281                    DEVICE: DS      19
                        ;
0294                    MSG02:
0294    50 61 72 61             DB      'Parameter error!',CR,LF,00H
0298    6D 65 74 65
029C    72 20 65 72
02A0    72 6F 72 21
02A4    0D 0A 00
                        ;
                                END
```

3.8.3 Disk Details

3.8.3.1 ROM capsule

(1) General

A ROM capsule is made up of a ROM capsule carrier that carries ROM chips. The PINE main unit can house two ROM capsules. Data in these capsules, once loaded into PINE memory, can be accessed immediately through bank switching.

The ROM chips for the PINE ROM capsules may be 28-pin DIP CMOS mask ROM chips or CMOS EPROM chips that are compatible with 2764, 27128, or 27256.

ROM capsules are used to store application programs for the PINE. The application programs in ROM capsules may be executed as are or loaded into RAM for execution.

A ROM capsule can contain M-format (format for PX-8 ROM capsule) ROM chips or ROM cartridge chips. (Programs in such ROM are always loaded into RAM for execution or the ROM capsule is used as a data file.)

A ROM capsule has a maximum capacity of 32K bytes. Two ROM capsules may be chained to form a single ROM disk drive.

(2) Drive name and capacity

(a) Drive name
When two ROM capsules are used separately ----- Drives B: and C:
When two ROM capsules are chained ---------- Drive B: or C:

(b) Capacity

	2764	27128	27256
Capacity	8 KB	16 KB	32 KB
Track	Ø	Ø - 1	Ø - 3
Sector	Ø - 63	Ø - 63	Ø - 63
Maximum directory entries	31	31	31

Note: The values in the above table are obtained when the number of directory entries is 28 to 31. If the number of directory entries is less than 28, the number of logical tracks available on each ROM is greater than the above values. For example, the maximum logical track and sector numbers of 27256 are 4 and 6, respectively, if the number of directory entries is 1 to 3. This is because the PINE always assumes 31 logical entries though it can handle an arbitrary number of physical entries (maximum is 31).

(3) Formats

(a) General

Application programs are classified into two groups according to how they are executed by the PINE CP/M.

1. Programs loaded into TPA for execution as under the ordinary CP/M (for all drives) ----- Load-and-go programs
2. Programs in the ROM capsule that are executed immediately in ROM ---------- ROM-based programs

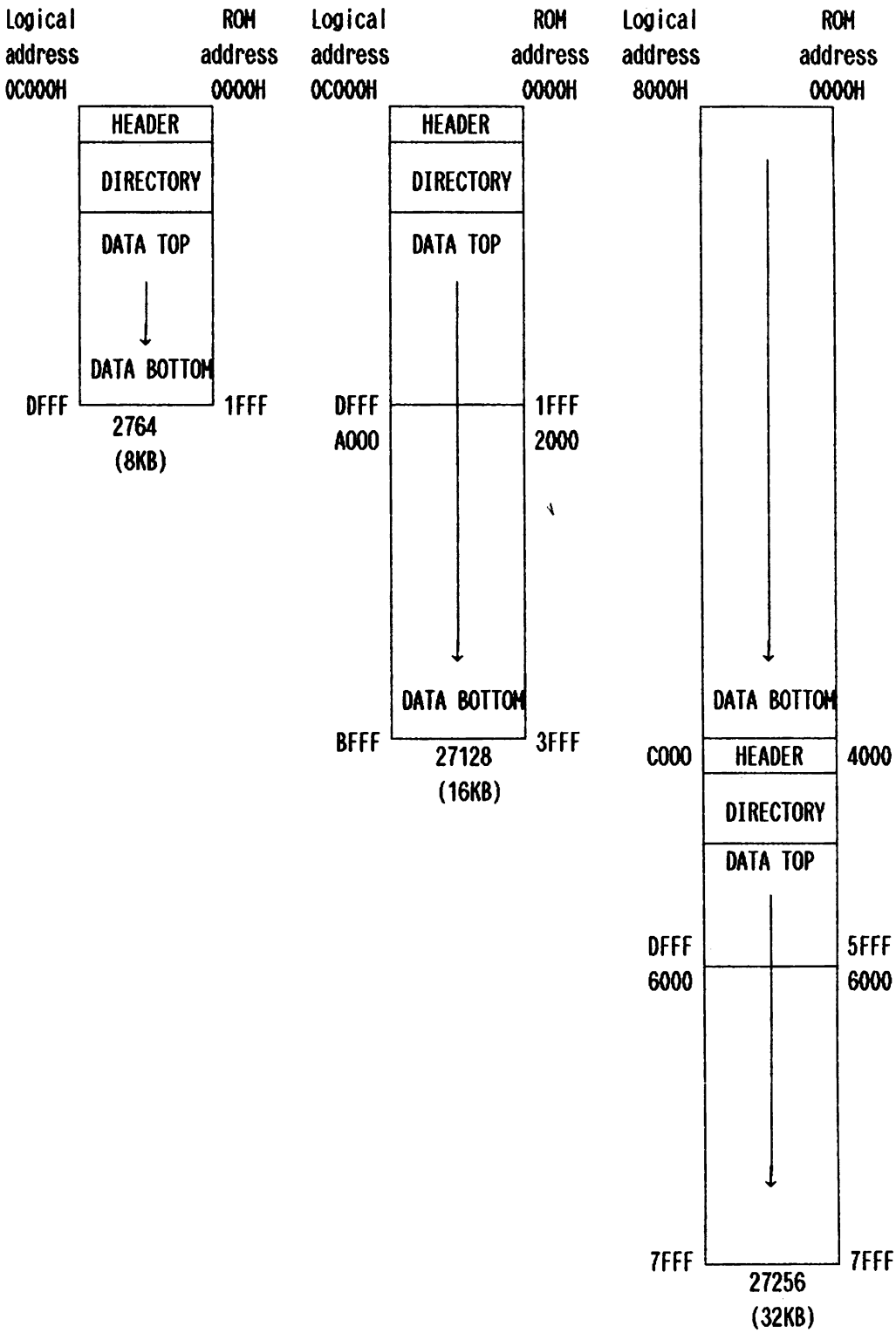
The PINE supports the following two ROM formats for the ROM capsule:

1. M format
The M format is used for both PINE ROM capsules and ROM cartridges. Load-and-go programs and ROM file data must be stored in this format.
2. P format
The P format is supported only by the PINE. ROM-based programs must be stored in ROM in this format. This format is used only for ROM capsules.

The OS automatically identifies the ROM format and calculates the address of the ROM, so that user need not identify the ROM format by himself.

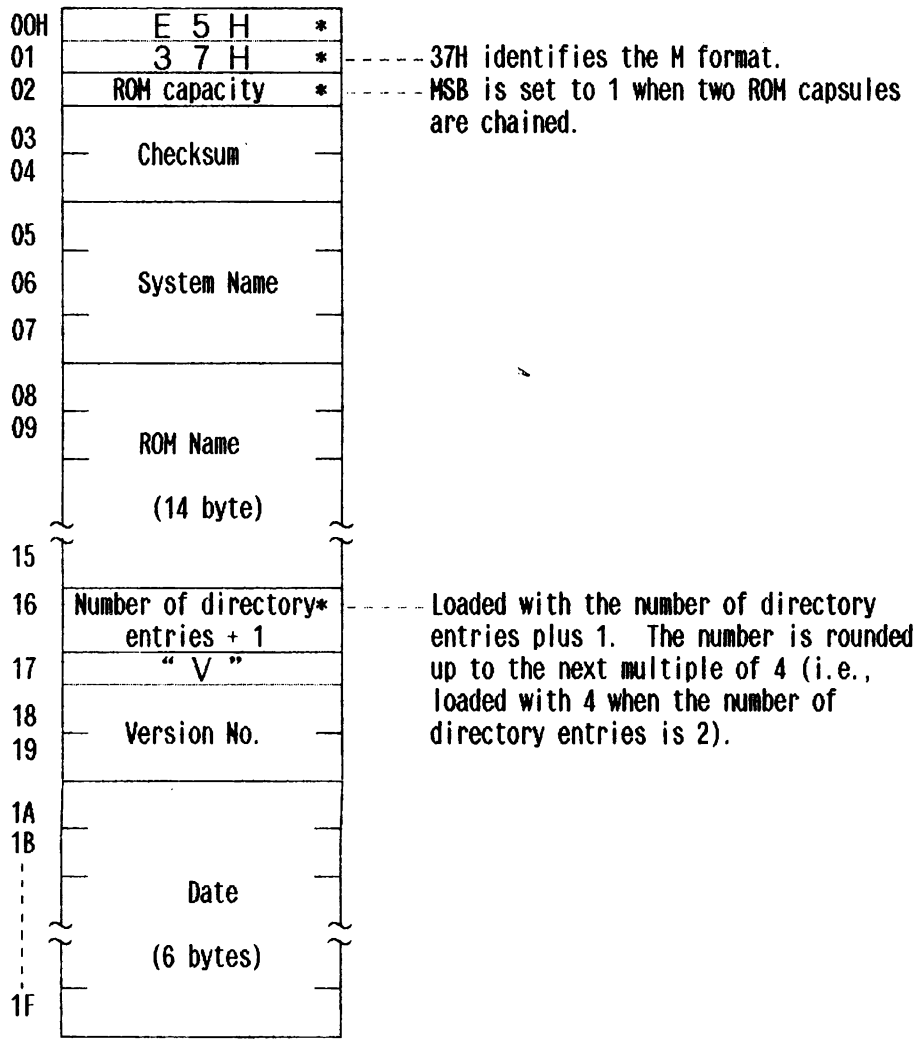
See Section 4.6, "Executing a ROM Program" for ROM-based programs.

(b) M format


Logical address
0C000H
ROM address
0000H
HEADER
DIRECTORY
DATA TOP
DATA BOTTOM
DFFF
1FFF
2764
(8KB)
Logical address
0C000H
ROM address
0000H
HEADER
DIRECTORY
DATA TOP
DFFF
A000
1FFF
2000
DATA BOTTOM
BFFF
3FFF
27128
(16KB)
Logical address
8000H
ROM address
0000H
DATA BOTTOM
C000
HEADER
4000
DIRECTORY
DATA TOP
DFFF
6000
5FFF
6000
7FFF
7FFF
27256
(32KB)

1) Header area

The header area has the following format (32 bytes):



The fields marked by * must always be loaded.

2) Directory area

The directory area can contain a maximum of 31 entries. One directory entry is 32 bytes long. Its format is the same as that of the CP/M FCB (File Control Block).

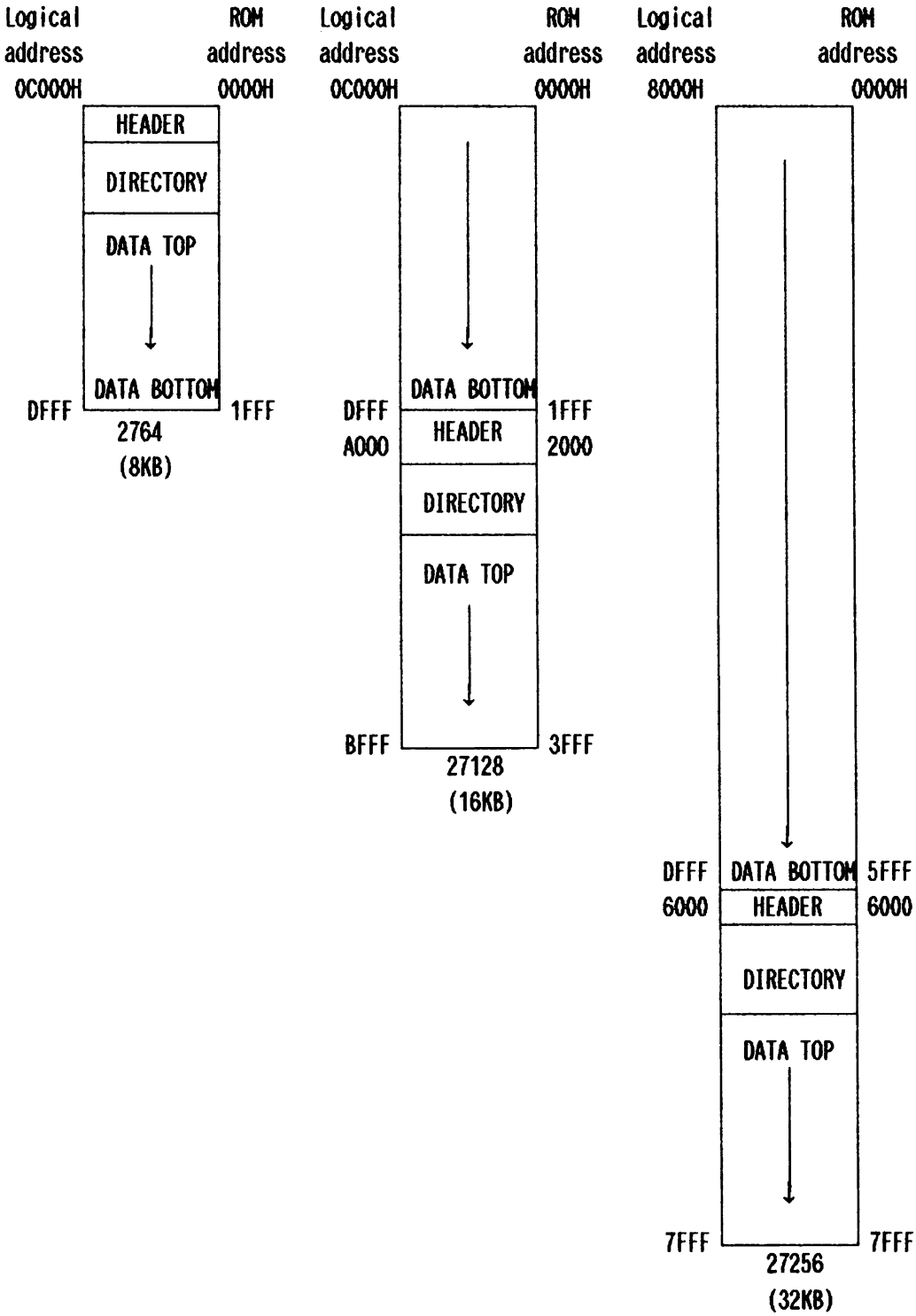
3) Data area

The data area start address varies depending on the number of directory entries. Assuming that the number of entries is n, the data area start address is found by adding (20H x m) to the header start address, where m is equal to (n + 1) rounded up to the next multiple of 4.

Any accesses to the locations between (header start address + 20H x m) and (header start address + 3FFH) logically return 0E5H. Logically, therefore, the data area start address is set to (header start address + 400H).

When two ROM capsules are used to form a single ROM drive (chained ROM capsules), the first ROM area that is left unused (20H x m bytes) is used as a data area together with the second ROM capsule. The second ROM capsule contain neither header nor directory and consists only of a data area.

(c) P format


Logical address
0C000H
ROM address
0000H
HEADER
DIRECTORY
DATA TOP
DATA BOTTOM
DFFF
1FFF
2764
(8KB)
Logical address
0C000H
ROM address
0000H
DATA BOTTOM
DFFF
1FFF
A000
HEADER
2000
DIRECTORY
DATA TOP
BFFF
3FFF
27128
(16KB)
Logical address
8000H
ROM address
0000H
DFFF
DATA BOTTOM
5FFF
6000
HEADER
6000
DIRECTORY
DATA TOP
7FFF
7FFF
27256
(32KB)

1) Header area

The second byte is loaded with 50H when the P format is used. The other fields are the same as in the M format.

2) Directory area

Same as with the M format.

3) Data area

Same as in the M format. ROM-based program stored in this format, however, must has an ID identifying their program type (executed on ROM) at their beginning. See Section 4.6, "Executing a ROM Program" for more information about ROM-based programs.

Load-and-go programs may also be loaded into the P-format ROM.

4) Other
(a) When two ROMs are used in series
When two ROMs are used in series, it is necessary to use two ROMs of the same capacity.

(b) Preparing ROMs
For points to note when preparing ROMs, please refer to 7.3, Guide for Programming ROMs

3.8.3.2 ROM cartridge

(1) General

The PINE ROM cartridge is an optional read-only external storage. It is connected to the PINE main unit via the PINE cartridge interface.

The ROM cartridge can contain two capsule-type, 28-pin DIP CMOS or NMOS-type mask ROM, or EPROM chips compatible with 2764, 27128, or 27256. It can also house PX-8 ROM chips or M format ROM chips which are used as ROM capsules. The ROM cartridge ROM can store only load-and-go programs or data files; it cannot hold ROM-based programs.
A ROM cartridge has a maximum capacity of 64K bytes. Two ROM chips (capsule-type) may be put together for form a single storage drive.

(2) Drive name and capacity

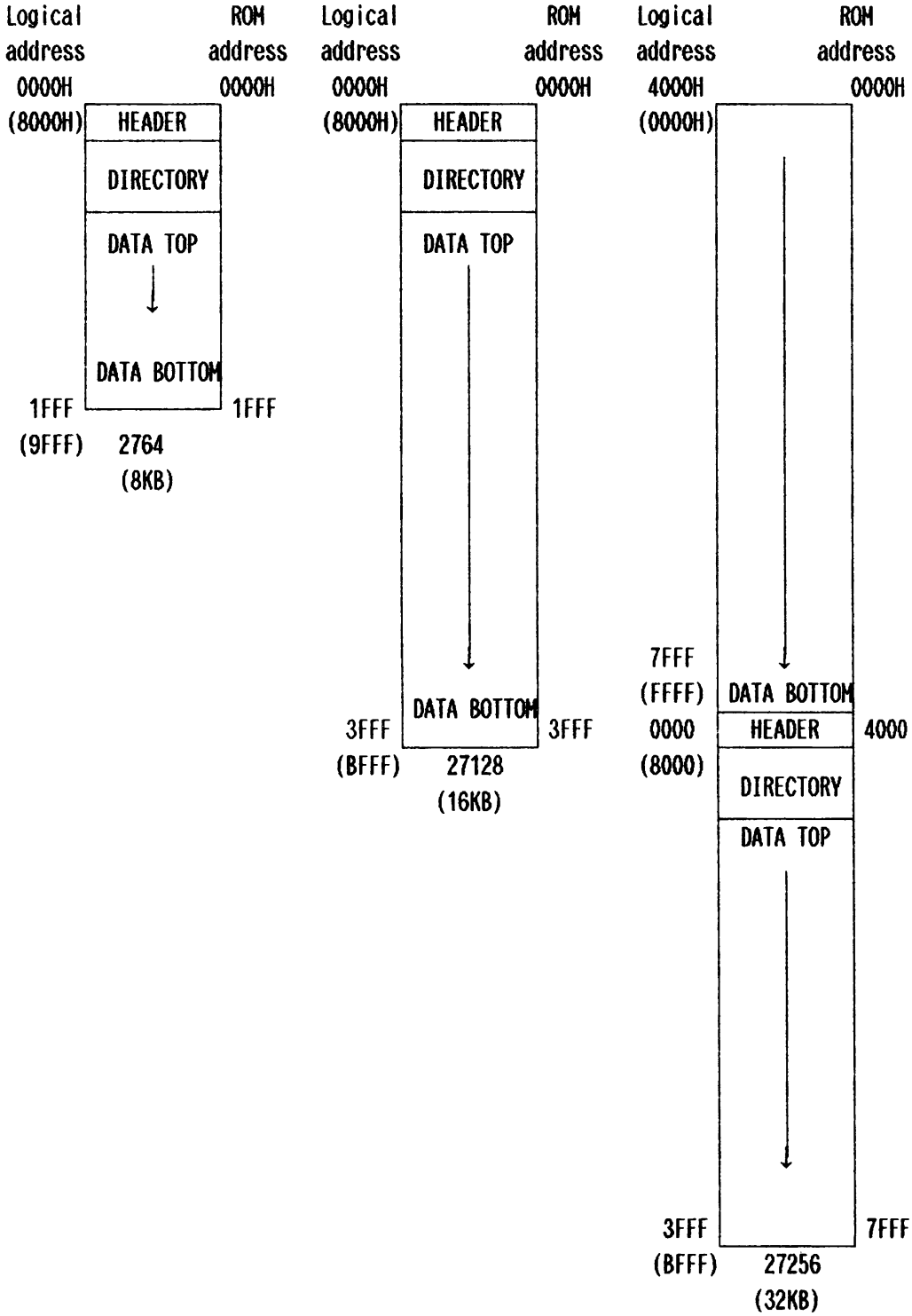
(a) Drive name
When two ROM chips are used separately ------- Drives J: and K:
When tow ROM chips are chained ------------ Drive J: or K:

(b) Capacity

Same as with the ROM capsule. (See 3.8.3.1.)

(3) Format

The format of the ROM cartridge ROM is shown in Figure 3.8.1. The formats of the header, directory, and data areas are the same as those of the M format ROM capsule ROM.



Note: Addresses enclosed in parentheses are logical addresses used when two ROM chips are used.

Fig. 3.8.1 ROM Cartridge ROM Format

(4) Miscellaneous Considerations

(a) Power control

The OS controls the power to the ROM cartridge to reduce power consumption. See Section 5.1, "Cartridges" for details.

(b) Notes on use of two ROM chips

The two ROM chips to be chained must have the same capacity.

3.8.3.3 Internal RAM disk

(1) General

The internal RAM disk is allocated in a part of the main RAM (bank Ø) and allows the PINE main unit to operate as a CP/M machine by itself.

The internal RAM disk cannot be set up when an external RAM disk is installed. If an external RAM disk is installed, it is always selected when drive A: is specified.

The PINE performs checksum tests on the internal RAM disk to check for inadvertent data destruction.

(2) Drive name and capacity

(a) Drive name
A:

(b) Capacity

1) The internal RAM disk size may be specified in 1K byte units from ØK to 35K bytes (it cannot be specified in 1K-byte size) provided that the total size of the internal RAM disk and the user BIOS does not exceed 35.5K bytes. This restriction is placed to realize a 2ØK CP/M.

2) The internal RAM disk size can be established during system initialization or by the user using the CONFIG program.

3) The internal RAM disk can be established during system initialization only when no external RAM disk is installed. The default capacity is 26K bytes. It is set to ØK byte when the external RAM disk is installed.

4) Track: Ø - 4
Sector: Ø - 63
Number of directory entries: 16

(3) Format

Figure 3.8.2 shows the internal RAM disk format.

(a) Directory area

The directory area can contain as many as 16 entries with each directory entry being 32 bytes long. The directory area is 512 bytes long and always present.

(b) Checksum area

This area is loaded with the results of checksum tests made in 128-byte blocks. One byte in this area indicates the results of the test on a 128-byte block. Checksum data is updated during writes and checked during reads. The checksum area is fixed at 512 bytes and always reserved.

(c) Data area

The data area starts at (internal RAM disk start address + 1K byte). The size of the data area depends on the internal RAM disk size.

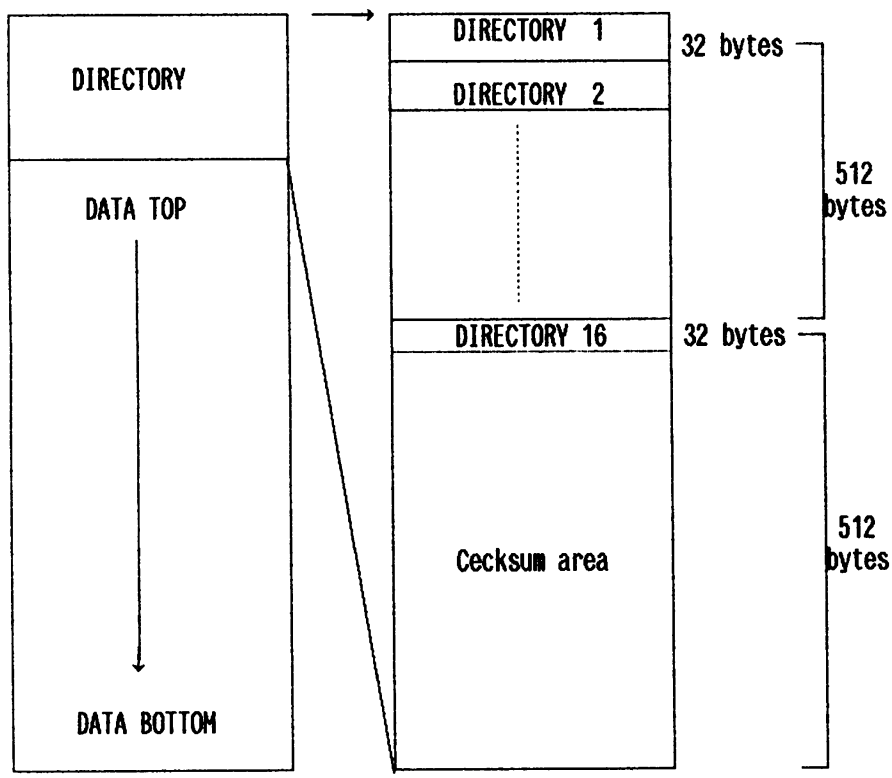


Fig. 3.8.2 Internal RAM Disk Format

(4) Miscellaneous considerations

(a) Since the internal RAM disk, in principle, is assigned to the background bank (bank Ø) of the application program, bank switching is required for a ROM-based program to access it directly. When accessing it via CP/M, however, the program need not be aware of bank switching because the OS takes care of it.

(b) The location of the internal RAM disk in memory varies depending on the size of the user BIOS and the internal RAM disk.

3.8.3.4 External RAM disk

(1) General

The nonintelligent-type external RAM disk is an optional drive connected to the PINE system bus. Its functions include automatic update of address registers after memory accesses and close mode operation that decreases power consumption. See Section 5.4, "External RAM Disk" for details.

(2) Drive name and capacity

(a) Drive name
A:

(b) Capacity
Capacity: 128K bytes
Track: Ø - 7
Sector: Ø - 63
Number of directory entries: 32

(3) Format

Figure 3.8.3 shows the external RAM disk format.

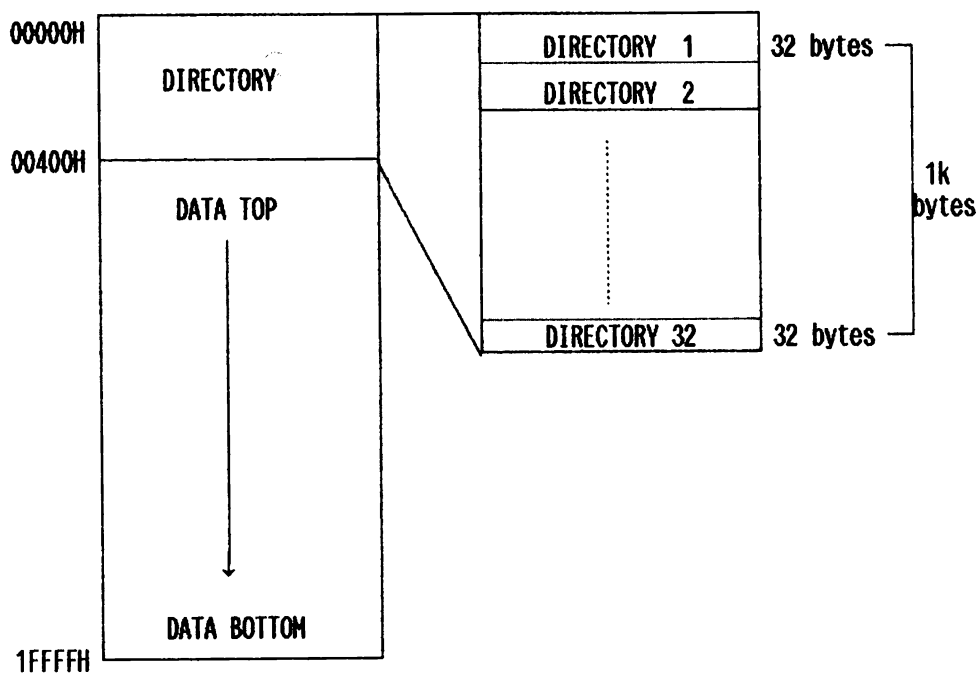


Fig. 3.8.3 External RAM Disk Format

(a) Directory area

The directory area can store a maximum of 32 entries of 32 bytes. This area is 1K bytes long and always reserved.

(b) Data area

The data area starts at address 00100H.

(4) Miscellaneous considerations

(a) When the external RAM disk is established at system initialization time, the OS automatically sets the internal RAM disk size to 0K byte.

(b) The external RAM disk connection status is determined at power-on. The system initialization is performed automatically when the PINE is switched on for the first time after the external RAM disk is connected or disconnected.

(c) Both the internal and external RAM disks are assigned to the same logical drive, i.e., drive A:. The two RAM disks cannot be used at the same time.

(d) The contents of the external RAM disk are not guaranteed if the reset button is pressed during a write to the external RAM disk. The system displays a

```
RAM DISK FORMAT (Y/N)
```

message in response to a reset made during a write so that the user can determine whether or not to format the external RAM disk.

3.8.3.5 RAM cartridge

(1) General

The PINE RAM cartridge is an optional external storage that allow both reads and writes. It is connected to the PINE main unit through the cartridge interface.

The RAM cartridge incorporates static RAM chips. Data on the static RAM chips can be accessed through I/O instructions issued from the PINE main unit. The RAM cartridge is battery-backed up and its data is sustained for up to three years even it is held removed from the main unit.

(2) Drive name and capacity

(a) Drive name

```
I:
```

(b) Capacity

```
Capacity: 16K bytes (The OS also supports 32K- and 64K-
          byte configuration.)
Track: 0 - 1 (for 16K-byte configuration)
       0 - 3 (for 32K-byte configuration)
       0 - 7 (for 64K-byte configuration)
Sector: 0 - 63
Number of directory entries: 16
```

(3) Format

The RAM cartridge has the format shown in Figure 3.8.4.

The formats of the directory, checksum, and data areas are the same as those of the internal RAM disk.

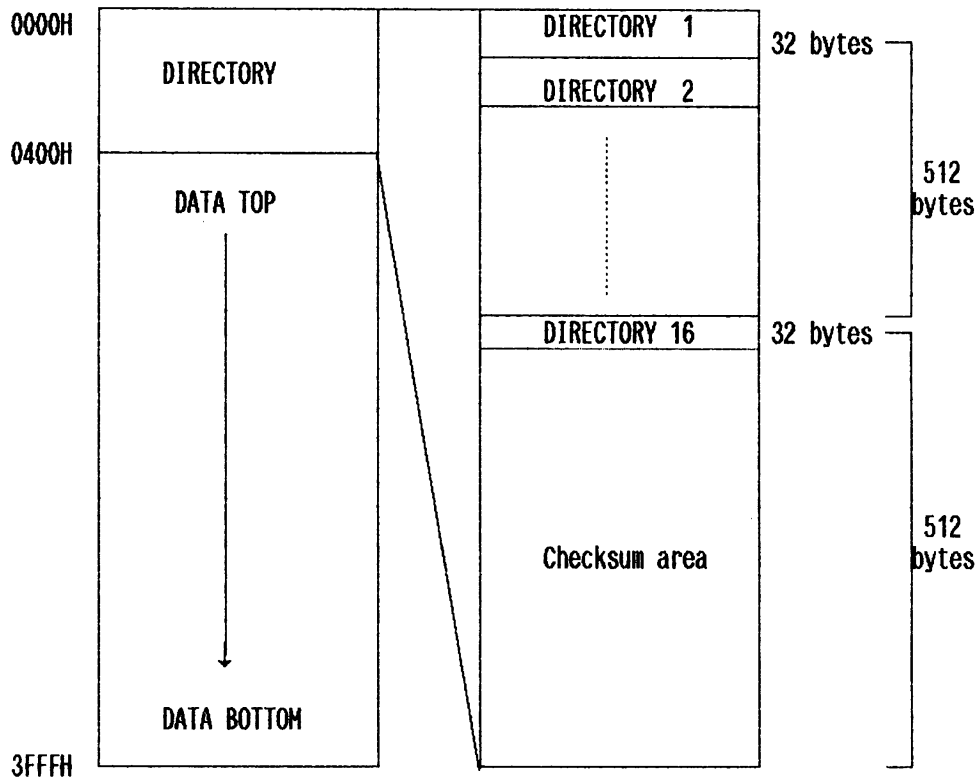


Fig. 3.8.4 RAM Cartridge Format

Note: The end address 7FFFH and ØFFFFH are for 32K-byte and 64K-byte configurations, respectively.

(4) Miscellaneous considerations

(a) The OS automatically identifies the RAM cartridge capacity when the PINE is powered on.

(b) RAM cartridges can be formatted by:
1) Selecting the item for RAM cartridge formatting from the system display.
2) Calling RAMCRFMT in the OS ROM jump table.

3.8.3.6 Microcassette

(1) General

The PINE microcassette is an optional external storage that allows both reads and writes. It is connected to the PINE main unit through the cartridge interface.

Tape data is also directed to the buzzer in the main unit or an external loudspeaker. Tape can be manipulated manually or by using programs. Manual manipulation is via system display. (Manual manipulation is not possible when an item keyboard is installed.) The MIOS routines are used to manipulate tape through programs.

Microcassettes (MCT) are used in almost the same way as disk drives except that they support only sequential files. Each MCT has a directory file at its beginning which enables fast access to files. See Section 3.7, "MTOS/MIOS Operations" for details.

(2) Drive name and capacity

(a) Drive name
H: (Fixed)

(b) Capacity

The capacity of an MCT varies depending on the tape length. The concepts of track and sector are not used for MCTs. The number of directory entries is 12.

(3) Format

See Section 3.7, "MTOS/MIOS Operations" for details.

(4) Miscellaneous considerations

(a) Mount

A mount refers to reading in the tape directory into RAM (RAM directory). Once a mount is carried out, any directory update (caused by file creation, for example) is managed in the RAM directory. A mount operation must be performed whenever a file is accessed. (The OS supports the automatic mount feature.)

(b) Remove

A remove refers to writing the RAM directory into the tape directory and must be performed at the end of file processing. If tape is removed without performing a remove operation, the tape data is not guaranteed. Moreover, the contents of the tape that is mounted next may be destroyed.

(c) Tape mount/remove

The PINE microcassette is provided with a LED for tape mount/remove identification. That LED is turned off when a tape is mounted and turned on when the tape is removed. Tape must be mounted or removed by the user after confirming that the LED is on.

(d) Tape access

Data on an MCT is blocked and deblocked in 256 byte units. The portion of tape from which files are deleted cannot be reused to store new files. Only one file at a time can be accessed for a read or write. See Section 3.7, "MTOS/MIOS Operations" for details.

(e) Additional BDOS routines

The following BDOS (MTOS) routines are provided for microcassette processing in addition to the standard BDOS routines:

1) Verify
2) Remove
3) Mount
4) Read tape ID
5) Make tape directory

(f) BIOS routine for MCT processing

The BIOS routine provided for MCT processing is MCMTX.

3.8.3.7 External disk (floppy disk)

(1) General

External disks such as 5.25 inch and 3.5 inch floppy and compact disks can be connected to the PINE via a serial interface. Data on external disks are read or written in 1K byte units.

The following disk units are connectable:

1) TF-20
2) TF-15 (single or dual drives)
3) PF-10

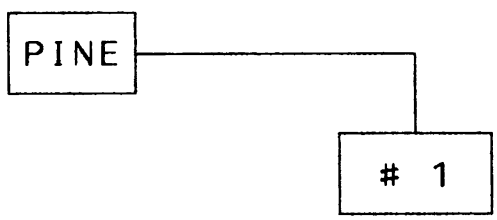
(2) Drive name and capacity

(a) Drive name

Drives D:, E:, F:, and G: are assigned to external disks. The association between a drive and a disk is determined by the DIP switches in that disk drive unit. Refer to manuals related to individual external disks for the procedure to assign a drive to a disk.

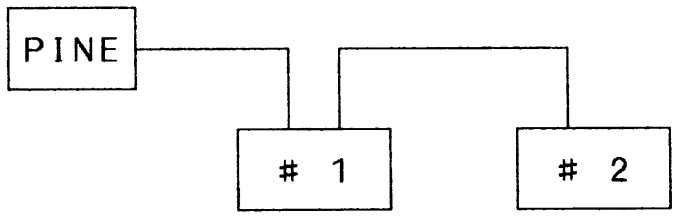
The PINE supports the following drive assignments:

1) When connecting an external disk



Model name	Drive name	Capacity
TF-20	D, E	320 KB x 2
TF-15 (single)	D	320 KB x 1
TF-15 (dual)	D, E	320 KB x 2
PF-10	D	320 KB x 1

2) When connecting two external disks



# 1		# 2	
Model name	Drive name	Model name	Drive name
TF-20	D, E	TF-20	F, G
TF-20	D, E	TF-15 (single)	F
TF-20	D, E	TF-15 (dual)	F, G
TF-20	D, E	PF-10	F
TF-15 (single)	D	TF-20	F, G
TF-15 (single)	D	TF-15 (single)	F
TF-15 (single)	D	TF-15 (dual)	F, G
TF-15 (single)	D	PF-10	F
TF-15 (dual)	D, E	TF-20	F, G
TF-15 (dual)	D, E	TF-15 (single)	F
TF-15 (dual)	D, E	TF-15 (dual)	F, G
TF-15 (dual)	D, E	PF-10	F
PF-10	D	TF-20	F, G
PF-10	D	TF-15 (single)	F
PF-10	D	TF-15 (dual)	F, G
PF-10	D	PF-10	F

(b) Capacity

Capacity/drive:	320K bytes
Number of tracks/drive:	80
Number of sectors/track:	16
Storage capacity/sector:	256 bytes
Number of directory entries:	64
Storage available for user:	278K bytes

(c) Format

See Section 5.3, "Terminal Floppy."

(4) Miscellaneous considerations

(a) Disk drives and serial communication

A external disk drive is connected to the PINE through a serial interface. Although the PINE supports three types of interfaces, RS-232C, SIO, and cartridge SIO for serial communication, only one of them can be used at a time.

The OS supports automatic switching of the serial port. This function, for example, enables access to a disk while the printer (connected through the RS-232C) is being used.

(b) Communication mode

The communication parameters passed between the PINE and external disks are as follows:

Bit rate:	38,400 bps
Data length:	8 bits
Stop bits:	1 bit
Parity bit:	None
Protocol:	EPSP (Epson Serial Communication Protocol)
Control line:	Not used

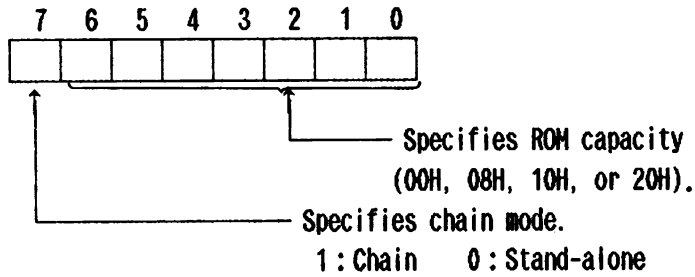
See Section 5.3, "Terminal Floppy" for more information.

(c) Buffer flushing

Disk units TF-20, TF-15 and PF-10 block or deblock data on tape to improve read/write efficiency. This leads to different processing speeds for the PINE and disk units. For example, when the PINE writes a record onto a floppy disk, the record may not be immediately written on the disk. To prevent the write data that is left unprocessed in the main unit buffer from being lost at the end of processing, the PINE OS automatically issues a flush buffer command to the disk unit at every warm boot or power-off.

3.8.4 Disk-related Work Areas

QT_ROM_CP1 (ØEF97H) 1 byte
- ROM capsule 1 (drive B:) capacity



QT_ROM_CP2 (ØEF98H) 1 byte
- ROM capsule 2 (drive C:) capacity
The format is the same as that of QT_ROM_CP1.

QT_ROM_CR1 (ØEF99H) 1 byte
- ROM cartridge 1 (drive J:) capacity
The format is the same as that of QT_ROM_CP1.

QT_ROM_CR2 (ØEF9AH) 1 byte
- ROM cartridge 2 (drive K:) capacity
The format is the same as that of QT_ROM_CP1.

QT_RAM_CR (ØEF9BH) 1 byte
- RAM cartridge capacity
= ØØH: No RAM cartridge installed.
1ØH: 16K-byte RAM cartridge
2ØH: 32K-byte RAM cartridge (not yet provided)
4ØH: 64K-byte RAM cartridge (not yet provided)

QT_RAM_IN (ØEF9CH) 1 byte
- Internal RAM disk capacity
ØØH ≤ QT_RAM_IN ≤ 23H

QT_RAM_EX (ØEF9DH) 1 byte
- External RAM disk capacity
= ØØH: No external RAM disk installed.
4ØH: 64K-byte external RAM disk (not yet provided)
8ØH: 128K-byte external RAM disk

DR_ROM_CP1 (ØEF9EH) 1 byte
- Number of directory entries for ROM capsule 1

DR_ROM_CP2 (ØEF9FH) 1 byte
- Number of directory entries for ROM capsule 2

DR_ROM_CR1 (ØEFAØH) 1 byte
- Number of directory entries for ROM cartridge 1

DR_ROM_CR2 (ØEFA1H) 1 byte
- Number of directory entries for ROM cartridge 2

AD_ROM_CP1 (ØEFA2H) 2 bytes
- ROM capsule 1 header address

AD_ROM_CP2 (ØEFA4H) 2 bytes
- ROM capsule 2 header address

AD_ROM_CR1 (ØEFA6H) 2 bytes
- ROM cartridge 1 header address
Always set to ØØØØH.

AD_ROM_CR2 (ØEFA8H) 2 bytes
- ROM cartridge 2 header address
Always set to ØØØØH.

AD_RAM_IN (ØEFAAH) 2 bytes
- Internal RAM disk header address

DISKTBL (ØF0FFH) 11 bytes
- Logical/physical drives association table
See 3.8.2, "Logical to Physical Drive Assignment."

ROMCPNO1 (ØF10AH) 1 byte
- ROM capsule 1 drive code
See 3.8.2, "Logical to Physical Drive Assignment."

ROMCPNO2 (ØF10BH) 1 byte
- ROM capsule 2 drive code
See 3.8.2, "Logical to Physical Drive Assignment."

DISKROV (ØF10CH) 2 bytes
- Disk R/O vector
See 3.8.2, "Logical to Physical Drive Assignment."

FTSTAB (ØF10EH) 22 bytes
- Jump vector for first disk select
The jump vector used when BIOS SELDSK is called for the first time to select a disk.

READTAB (ØF124H) 22 bytes
- Jump vector used during a disk read
Control is passed to the requested read routine through this jump vector during execution of a BIOS READ call.

WRTTAB (ØF13AH) 22 bytes
- Jump vector used during a disk write
Control is passed to the requested write routine through this jump vector during execution of a BIOS WRITE call.

DPBASE (ØF150H)
- Disk parameter header (DPH)
See a CP/M manual for the format of this header.

DPE0 16 bytes
- DPH for RAM disk

DPE1 16 bytes
- DPH for ROM capsule 1

DPE2 16 bytes
- DPH for ROM capsule 2

DPE3 16 bytes
- DPH for FDD (External floppy disk drive) 1

DPE4 16 bytes
- DPH for FDD 2

```
DPE5   16 bytes
- DPH for FDD 3

DPE6   16 bytes
- DPH for FDD 4

DPE7   16 bytes (microcassette)
- DPH for MCT

DPE8   16 bytes
- DPH for ROM cartridge 1

DPE9   16 bytes
- DPH for ROM cartridge 2

DPE1Ø  16 bytes
- DPH for RAM cartridge

DPBØ (ØF2ØØH)  15 bytes
- DPB (Disk Parameter Block) for RAM disk

DPB1 (ØF2ØFH)  15 bytes
- DPB for ROM capsule 1

DPB2 (ØF21EH)  15 bytes
- DPB for ROM capsule 2

DPB3 (ØF22DH)  15 bytes
- DPB for FDD

DPB7 (ØF2CFH)  15 bytes
- DPB for MCT

DPB8 (ØF24BH)  15 bytes
- DPB for ROM cartridge 1

DPB9 (ØF25AH)  15 bytes
- DPB for ROM cartridge 2

DPB1Ø (ØF269H)  15 bytes
- DPB for RAM cartridge

DIRBUF (ØF5FBH)  128 bytes
- Directory access buffer

ALVØ (ØF67BH)        32 bytes: RAM disk allocation area
CSVØ                  Ø byte: RAM disk checksum area

ALV1 (ØF69BH)        9 bytes: ROM capsule 1 allocation area
CSV1                  Ø byte: ROM capsule 1 checksum area

ALV2 (ØF6A4H)        9 bytes: ROM capsule 2 allocation area
CSV2                  Ø byte: ROM capsule 2 checksum area

ALV3 (ØF6ADH)        18 bytes: FDD 1 allocation area
CSV3                  16 bytes: FDD 1 checksum area

ALV4 (ØF6CFH)        18 bytes: FDD 2 allocation area
CSV4                  16 bytes: FDD 2 checksum area

ALV5 (ØF6F1H)        18 bytes: FDD 3 allocation area
CSV5                  16 bytes: FDD 3 checksum area
```

ALV6 (ØF713H)	18 bytes: FDD 4 allocation area
CSV6	16 bytes: FDD 4 checksum area
ALV7 (ØF735H)	8 bytes: MCT allocation area
CSV7	16 bytes: MCT checksum area
ALV8 (ØF74DH)	8 bytes: RAM cartridge allocation area
CSV8	4 bytes: RAM cartridge checksum area
ALV9 (ØF759H)	9 bytes: ROM cartridge 1 allocation area
CSV9	Ø byte: ROM cartridge 1 checksum area
ALV1Ø (ØF761H)	9 bytes: ROM cartridge 2 allocation area
CAV1Ø	Ø byte: ROM cartridge 2 checksum area
SYSFCB (ØF88DH)	36 bytes: system FCB (File Control Block) area
SYSDMA (ØF8B1H)	128 bytes: system DMA (Direct Memory Access) area

3.9 I/O Byte

3.9.1 General

The I/O byte is used to associate physical devices with four logical devices LST:, PUN:, RDR:, and CON:. It is located at address ØØØ3H as with the ordinary CP/M.

BIOS routines making use of the I/O byte are CONST, CONIN, CONOUT, LIST, PUNCH, and READER.

3.9.2 Modifying the I/O Byte

The contents of the I/O byte can be modified by:

(1) Using STAT.
(2) Using BDOS.
(3) Directly rewriting location ØØØ3H.

3.9.3 I/O Byte Contents

Table 3.9.1 shows the bit assignments of the I/O byte.

Logical device	LST :	PUN :	RDR :	CON :	
Bit position	7, 6	5, 4	3, 2	1, 0	
I/O / Bit values	Output	Output	Output	Output	Input
00	SIO	———	Keyboard	RS-232C	Keyboard
01	*Cartridge	LCD	———	*LCD	*Keyboard
10	RS-232C	*RS-232C	*RS-232C	LCD	RS-232C
11	Centronics	———	———	RS-232C	RS-232C

Table 3.9.1 I/O byte Bit Assignments

(1) The devices identified by * are defaults.

(2) The default device for LST: is determined by the state of DIP switches, bits 5 and 6.

```
DIP switches
Bits 8 - 7      Not used.
     6 - 5      = ØØ: SIO
                  Ø1: Cartridge
                  1Ø: RS-232C
                  11: Centronics
     4 - 1      Keyboard selection
```

See Section 7.2, "Dip Switches" for details.

(3) When the PUN: or RDR: fields are set to the values indicated by "---" in the list

```
PUN: ---  The device is always in the ready state and outputs no
          data when it is driven.
RDR: ---  The device is always in the ready state and returns 1AH
          (CTRL/Z) in response to reque.
```

(4) The I/O byte is initialized at cold boot time.

CHAPTER 4 CP/M II (EXTENDED CP/M) CONTENTS

4.1 User BIOS II-368
4.1.1 General II-368
4.1.2 User BIOS II-368
4.1.3 User BIOS Area II-370

4.2 Jump Tables II-381
4.2.1 General II-381
4.2.2 Organization II-382
4.2.3 Resident Jump Table II-385
4.2.4 OS ROM Jump Table II-397

4.3 Hooks II-429
4.3.1 General II-429
4.3.2 Hook Structures II-429
4.3.3 Alarm Hooks II-433
4.3.4 Interrupt Hooks II-437
4.3.5 TIMDAT Hooks II-441
4.3.6 BIOS Hook II-442

4.4 Bank Switching II-449
4.4.1 General II-449
4.4.2 Bank Structure II-449
4.4.3 Utility Routines II-453
4.4.4 Work Areas Associated with Bank Switching II-456

4.5 Resident Processing II-457
4.5.1 General II-457
4.5.2 How to Specify and Cancel Resident II-457
4.5.3 Processing Flows II-459
4.5.4 Miscellaneous Considerations II-461

4.6 Executing a ROM based Program II-461
4.6.1 General II-461
4.6.2 Establishing the Execution Environment II-461
4.6.3 Processing Flow II-464
4.6.4 Use and Programming Notes II-465

4.7 Interrupts II-472
4.7.1 General II-472
4.7.2 Interrupt Vector II-472
4.7.3 Interrupt Control II-473
4.7.4 7508 Interrupts II-489
4.7.5 ART Interrupts II-498
4.7.6 OVF Interrupts II-498
4.7.7 ICF/EXT Interrupts II-498

CHAPTER 4 CP/M II (Extended CP/M)

This chapter describes the extended facilities of the PINE CP/M. The topics covered in this chapter include the following:

- User BIOS
- Jump tables
- Hook processing
- Bank management
- Resident facility
- ROM program execution
- Interrupt handling

For the basic BDOS and BIOS functions, see Chapter 3, "CP/M I (BDOS, BIOS).

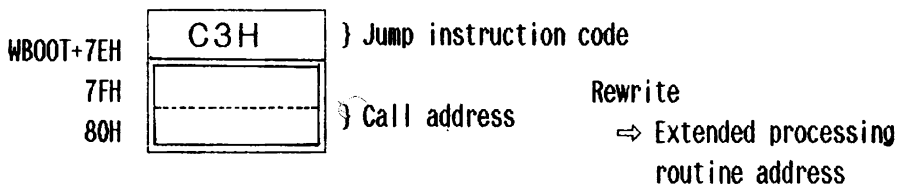
4.1 User BIOS

4.1.1 General

PINE OS provides a BIOS entry, named User BIOS, which allows the user to expand or add user-supplied BIOS routines to BIOS. It also reserves a user BIOS area for use by the user BIOS routines. The user BIOS area may be used by not only the user BIOS routines but also by common machine-language routines (e.g., barcode reader programs) and extended hook processing routines (e.g., alarm time updating routines). This section explains how to use the user BIOS functions and the user BIOS area.

4.1.2 User BIOS

4.1.2.1 Location of the entry to the user BIOS
The PINE has two BIOS entries: RBIOS1 and RBIOS2. Application programs which are to expand the user BIOS capability must rewrite the USERBIOS call address in RBIOS1.



The location of WBOOT is stored in addresses 0001H and 0002H.)

Application programs must observe the following procedure when calling USERBIOS.

(1) Calling USERBIOS from a load-and-go program
Find the entry address of WBOOT in addresses 0001H and 0002H and call WBOOT + 7EH as when calling other BIOS functions.
(2) Calling USERBIOS from a ROM-based program
Since the RBIOS1 area is likely to be used as a background bank when a ROM-based program is running, it is necessary to switch to bank 0 with a CALLXX (0FFAEH) before calling USERBIOS. The address of USERBIOS can be obtained in the same way as in paragraph (1). See Section 4.2, "Jump Tables" for the use of the CALLXX instruction.

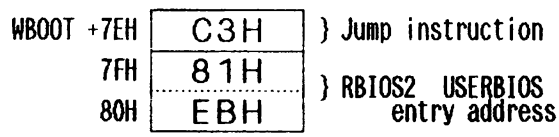
4.1.2.2 Programming notes

Care must be exercised with the following when using user BIOS:
(1) Extending BIOS functions using the user BIOS area
Observe the precautions about the user BIOS area given in 4.1.3 when inserting data pertaining to the extended user BIOS function into the user BIOS area.
(2) Location of user BIOS
The user BIOS is located on bank Ø (RAM). When user BIOS is called, however, the active bank remains to be the one on which the calling program resides. This means that the active bank is either bank 1 or 2 when user BIOS is called from a ROM-based program and bank Ø when it is called from a load-and-go program. When calling extended user BIOS from a ROM-based program, switch to bank Ø before calling user BIOS.

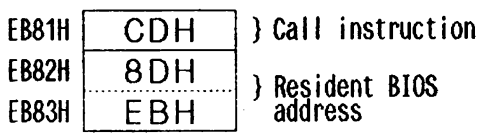
4.1.2.3 Extended user BIOS processing

Note the following when constructing extended user BIOS processing routines:
(1) Every user BIOS routine must end with a RET instruction.
(2) The user stack remains active when user BIOS is called. If necessary, the extended user BIOS processing routine must reserve and use a stack for itself.
(3) For user BIOS to remain in memory after termination of a program, the extended portion of user BIOS must be placed in the user BIOS area. If it is unnecessary to retain user BIOS, reset the user BIOS call address to the old value.

Example 1: Resetting RBIOS1 USERBIOS



Example 2: Resetting RBIOS2 USERBIOS



4.1.2.4 Initializing user BIOS

PINE OS initializes the user BIOS entry (USERBIOS) as well as the system BIOS entries when reset or system initialize processing is carried out. As initialized, USERBIOS executes only a RET instruction.

4.1.3 User BIOS Area

4.1.3.1 Location of the user BIOS area

The user BIOS area is reserved between the internal RAM disk unit and the item key table. Figure 4.1.1 shows the location of the user BIOS area in memory.

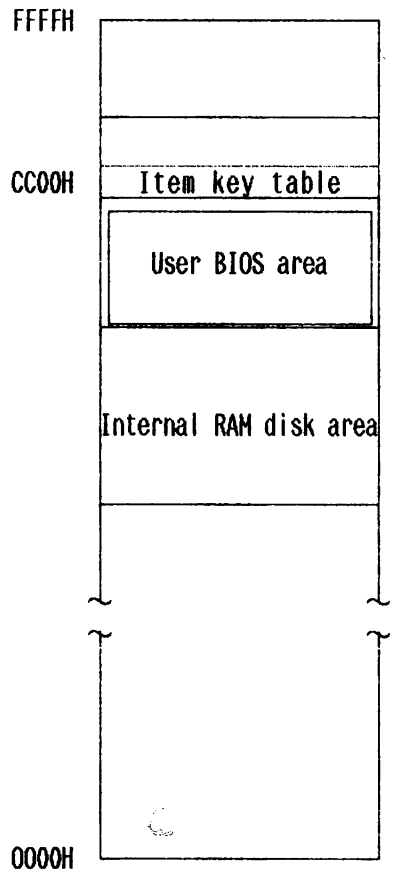


Fig. 4.1.1 User BIOS Area

4.1.3.2 Reserving the user BIOS area

(1) Outline
The user BIOS area can be reserved in one of the following ways:
a) Reserve at system initialize time.
b) Reserve using the CONFIG utility.
c) Reserve using a user-supplied program.

See Section 2.2, "System Initialize" for method a) and the CONFIG manual for method b). Here, the user BIOS reservation procedure using a user-supplied program is introduced.
(2) Reserving the user BIOS area
Follow the flowchart shown in Figure 4.1.2 to reserve space for the user BIOS area. The user BIOS area must be reserved in 256-byte increments.

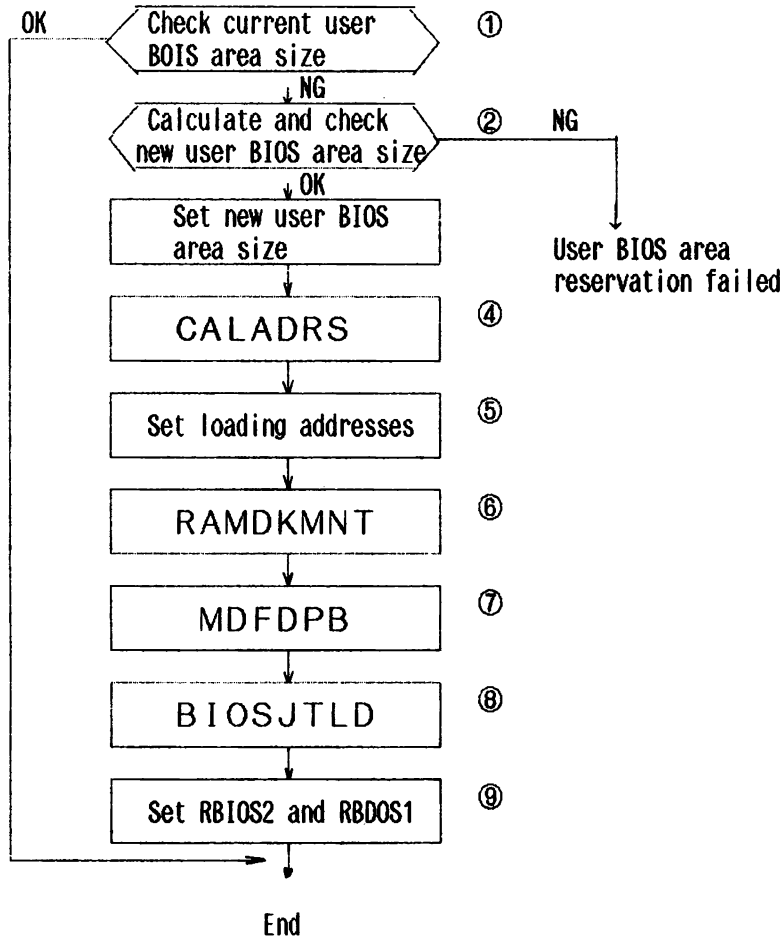


Fig. 4.1.2 Reserving the User BIOS Area

Step 1: Check the current user BIOS area size.
Check the size of the current user BIOS area for validity. The current BIOS area size is stored in USERBIOS (ØEF2DH).

Step 2:Calculate the size of the new user BIOS area.
Check the size of the required user BIOS area and the internal RAM disk to see whether their sum is less than 35.5K bytes.
The user BIOS area cannot be reserved if the sum is greater than 35.5K bytes. The size of the current internal RAM disk is given in 1K bytes in SIZRAM (ØEF2CH).

Step 3: Set the size of the new user BIOS area.
Load the size in 256-byte units of the new user BIOS area into USERBIOS (ØEF2DH).

Step 4: Call CALADRS.
CALADRS calculates the size of CP/M. CALADRS is cataloged in the jump table on OS ROM and its address is ØØ18H. Use the BIOS CALLX function to call CALADRS. See Section 4.2, "Jump Tables" for CALADRS and Section 3.4, "BIOS Details" for CALLX.

Step 5: Set the loading addresses.
Loads the CP/M loading addresses that are returned by CALADRS into the system area.

(BC) → TOPRAM (ØEF94H)
(DE) → BIlLAD (ØEF26H)
(IX) → BDSLAD (ØEF24H)
(IY) → CCPLAD (ØEF22H)

Step 6: Call RAMDKMNT.
RAMDKMNT checks for the presence of a RAM disk. RAMDKMNT is cataloged in the jump table on OS ROM and its address is ØØ1EH. Use the BIOS CALLX function to call RAMDKMNT.

Step 7: Call MDFDDB.
Modify the contents of the disk parameter block. MDFDPB is cataloged in the jump table on OS ROM and its address is ØØ21H. Use the CALLX function to call MDFDPB.

Step 8: Call BIOSJTLD.
BIOSJTLD loads the contents of the BIOS jump table. BIOSJTLD is cataloged in the jump table on OS ROM and its address is ØØ1BH. Use the BIOS CALLX function to call BIOSJTLD.

Step 9: Set RBIOS1 and RBDOS1.
Set the entry address of RBIOS1 and RBDOS1.

BIlLAD (ØEF26H) + 3 → (ØØØ1H)
BDSLAD (ØEF24H) + 6 → (ØØØ6H)

(3) Programming notes

1. BIOS in RBIOS2 must be used when reserving the user BIOS area. This is because RBIOS1 is relocated as the user BIOS area expands or shrinks.
2. Once the user BIOS area is reserved, do not use RBDOS1 (at address ØØØ5H) until WBOOT is invoked. This is because once the user BIOS area expands or shrinks, RBDOS1 is not loaded until WBOOT is invoked. Use RBDOS2 when using BDOS.
3. The sum of the sizes of the user BIOS area and internal RAM disk must not exceed 35.5K bytes. Otherwise, a system initialize will occur when the RESET switch is pressed (this does not hold true at power-on time because in such a case no size check is performed).

(4) Reference
The table below lists the work areas that are affected when the user BIOS area size is updated.

SIZRAM (ØEF2CH) 1 byte
Contains the size in 1K-byte units of the internal RAM disk.
Ø ≤ SIZRAM ≤ 35, SIZRAM ≠ 1

USERBIOS (ØEF2DH) 1 byte
Contains the size in 256-byte units of the user BIOS area.
Ø ≤ SIZRAM ≤ 142,
(SIZRAM x 4) + USERBIOS ≤ 142.

CCPLAD (ØEF22H) 2 bytes
Contains the CCP loading address.

BDSLAD (ØEF24H) 2 bytes
Contains the RBDOS1 loading address.

BIlLAD (ØEF26H) 2 bytes
Contains the RBIOS1 loading address.

TOPRAM (ØEF94H) 2 bytes
Contains the user BIOS area starting address. Set to CCØØH if no user BIOS area is reserved.

```
                        ;       ************************************************
                        ;               CHANGE RAM DISK & USER BIOS SIZE PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is changing RAM disk
                        ;               and User BIOS size.
                        ;
                        ;       <> assemble condition <>
                        ;
                        ;       .Z80
                        ;
                        ;       <> loading address <>
                        ;
                        ;       .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
                        ;       BIOS entry
                        ;
EB03                    WBOOT           EQU     0EB03H          ; Warm Boot entry
EB0C                    CONOUT          EQU     WBOOT   +09H    ; Console out entry
EB69                    CALLX           EQU     WBOOT   +66H    ; Call extra entry
                        ;
                        ;       System area
                        ;
EF94                    TOPRAM          EQU     0EF94H          ; Top of User BIOS
EF26                    BIlLAD          EQU     0EF26H          ; RBIOS1 loading addr
EF24                    BDSLAD          EQU     0EF24H          ; RBDOS1 loading addr
EF22                    CCPLAD          EQU     0EF22H          ; CCP loading addr
EF9D                    QTRAMEX         EQU     0EF9DH          ; Quantity of external RAM disk
EF9C                    QTRAMIN         EQU     0EF9CH          ; Quantity of internal RAM disk
EF2D                    USERBIOS        EQU     0EF2DH          ; Size of User BIOS area
F77A                    YSIZERAM        EQU     0F77AH          ; Size of RAM disk
EF2C                    SIZRAM          EQU     0EF2CH          ; Size of internal RAM disk
F52E                    DISBNK          EQU     0F52EH          ; Destination bank for CALLX
                        ;
                        ;       Bank 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)
                        ;
                        ;       User BIOS area
                        ;
CBF0                    UB_HEAD         EQU     0CBF0H          ; Top addr of User BIOS area's header
CBFB                    UB_OVWRITE      EQU     UB_HEAD +11     ;  Over write flag
CBFC                    UB_RELEASE      EQU     UB_HEAD +12     ;  Release address
                        ;
                        ;
0001                    BIOSENTRY       EQU     00001H          ; CP/M BIOS entry addr
0006                    BDOSENTRY       EQU     00006H          ; CP/M BDOS entry addr
                        ;
                        ;       OS ROM jump table
                        ;
0018                    CALADRS         EQU     00018H          ; Calculate loading addr
001E                    RAMDKMNT        EQU     0001EH          ; RAM disk mount check
0021                    MDFYDPB         EQU     00021H          ; Modify disk parameter block
001B                    BIOSJTLD        EQU     0001BH          ; BIOS jump table load
                        ;
                        ;       New RAM disk size and User BIOSsize
                        ;
001E                    SRAMDISK        EQU     30
0004                    SUSERBIOS       EQU     4
                        ;
008E                    MAXSIZE         EQU     142             ; 35.5 KB * 4
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This program is changing size as following.
                        ;                       RAM disk  size --> 30 kbytes
                        ;                       User BIOS size --> 4  kbytes
                        ;
0100                    MAIN:
0100    31 1000                 LD      SP,1000H
0103    06 1E                   LD      B,SRAMDISK
0105    0E 10                   LD      C,SUSERBIOS*4
0107    CD 01A4                 CALL    CHNGSZ
                        ;
010A    CD 0110                 CALL    MESSAGE
010D    C3 EB03                 JP      WBOOT
                        ;
                        ;
                        ;
                        ;       ************************************************
                        ;               RETURN MESSAGE DISPLAY
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               A  : Message parameter
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0110                    MESSAGE:
0110    21 0128                 LD      HL,MSGTBL       ; Message table top addr.
0113    87                      ADD     A,A             ; Get target message.
0114    4E                      LD      C,A             ;  A*4      --> C
```

```
0115    06 00                   LD      B,0             ;  0        --> B
0117    09                      ADD     HL,BC           ;  HL + BC --> HL
0118    5E                      LD      E,(HL)          ;  (HL)     --> HL
0119    23                      INC     HL              ; HL is top addr of message,
011A    56                      LD      D,(HL)          ;
011B    EB                      EX      DE,HL           ;
                        ;
011C                    MSGLOOP:
011C    4E                      LD      C,(HL)          ; Get message.
011D    0D                      DEC     C               ; Data is 0?
011E    0C                      INC     C               ;
011F    C8                      RET     Z               ; Yes.
                        ;
0120    E5                      PUSH    HL              ;
0121    CD EB0C                 CALL    CONOUT          ; Display message.
0124    E1                      POP     HL              ;
0125    23                      INC     HL              ; Pointer update.
0126    18 F4                   JR      MSGLOOP         ; Loop until find 0.
                        ;
                        ;       Message table
                        ;
0128                    MSGTBL:
0128    0130                    DW      MSG1            ;
012A    0152                    DW      MSG2            ;
012C    0165                    DW      MSG3            ;
012E    0188                    DW      MSG4            ;
                        ;
                        ;       Message data
                        ;
0130                    MSG1:
0130    43 68 61 6E             DB      'Changing size is normaly ending',0DH,0AH,00H
0134    67 69 6E 67
0138    20 73 69 7A
013C    65 20 69 73
0140    20 6E 6F 72
0144    6D 61 6C 79
0148    20 65 6E 64
014C    69 6E 67 0D
0150    0A 00
0152
0152    50 61 72 61             DB      'Parameter error.',0DH,0AH,00H
0156    6D 65 74 65
015A    72 20 65 72
015E    72 6F 72 2E
0162    0D 0A 00
0165
0165    55 73 65 72             DB      'User BIOS area cannot destroyed.',0DH,0AH,00H
0169    20 42 49 4F
016D    53 20 61 72
0171    65 61 20 63
0175    61 6E 6E 6F
0179    74 20 64 65
017D    73 74 72 6F
0181    79 65 64 2E
0185    0D 0A 00
0188                    MSG4:
0188    4F 76 65 72             DB      'Overwrite User BIOS area.',0DH,0AH,00H
018C    77 72 69 74
0190    65 20 55 73
0194    65 72 20 42
0198    49 4F 53 20
019C    61 72 65 61
01A0    2E 0D 0A 00
                        ;
                        ;
                        ;       ************************************************
                        ;               SIZE CHANGING UTILITY
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               B   : New RAM disk  size (unit 1  kbytes)
                        ;               C   : New User BIOS size (unit 256 bytes)
                        ;       <> return parameter <>
                        ;               A   : Return information
                        ;                       =00H -- Normal return
                        ;                       =01H -- Entry parameter is size over
                        ;                       =02H -- Cannot getting User Bios area
                        ;                       =03H -- Overwrite User BIOS area
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;               If new User BIOS size doesn't equal
                        ;               to 0, the programor data in old User
                        ;               BIOS area will by destroyed.
                        ;               If old User BIOS area inhibits over-
                        ;               write by other program, this routine
                        ;               will go back by error return code.
                        ;
01A4                    CHNGSZ:
01A4    3A EF9D                 LD      A,(QTRAMEX)     ; Get quantity of external RAM disk.
01A7    B7                      OR      A               ; Size is 0?
01A8    28 02                   JR      Z,IN_RAM        ; Yes,
01AA    06 00                   LD      B,00H           ; Change new RAM disk size to 0.
01AC                    IN_RAM:
01AC    78                      LD      A,B             ; Check area size.
01AD    87                      ADD     A,A             ;  Calculate by unit 256 bytes.
01AE    87                      ADD     A,A             ;
01AF    81                      ADD     A,C             ;  Add RAM disk size and User BIOS size.
01B0    FE 8F                   CP      MAXSIZE+1       ;  New size is OK?
01B2    3E 01                   LD      A,01H           ;   Return parameter.
01B4    D0                      RET     NC              ;  Return if size over.
                        ;
01B5    AF                      XOR     A               ; Set return parameter.
```

```
01B6    32 024F                 LD      (RET_PRM),A     ;  0 --> RET_PRM
01B9    3A EF2D                 LD      A,(USERBIOS)    ; Get User BIOS area size.
01BC    B7                      OR      A               ; No User BIOS area?
01BD    28 16                   JR      Z,CHNG_UB       ;  Yes.
                        ;
01BF    CD 021D                 CALL    CHK_HEAD        ; Check User BIOS header.
01C2    20 11                   JR      NZ,CHNG_UB      ;  No User BIOS header.
01C4    3E 03                   LD      A,03H           ; User BIOS already exists.
01C6    32 024F                 LD      (RET_PRM),A     ;  3 --> RET_PRM
01C9    CD 0239                 CALL    RELEAS          ; User BIOS area release.
01CC    30 07                   JR      NC,CHNG_UB      ;  Release OK.
01CE    3E 02                   LD      A,02H           ; Set error return parameter.
01D0    32 024F                 LD      (RET_PRM).A     ;  2 --> RET_PRM
01D3    18 04                   JR      CHNG_RAM        ;
                        ;
01D5                    CHNG_UB:
01D5    79                      LD      A,C             ; Change User BIOS area size.
01D6    32 EF2D                 LD      (USERBIOS),A    ;  New User BIOS area size.
                        ;
01D9                    CHNG_RAM:
01D9    78                      LD      A,B             ; Change internal RAM disk size.
01DA    32 EF2C                 LD      (SIZRAM),A      ;  New internal RAM disk size.
                        ;
01DD    3E FF                   LD      A,SYSBANK       ; Set desination bank.
01DF    32 F52E                 LD      (DISBNK),A      ;  FFH --> DISBNK
01E2    DD 21 0018              LD      IX,CALADRS      ; Calculate loading addr.
01E6    CD EB69                 CALL    CALLX           ;
                        ;
01E9    ED 43 EF94              LD      (TOPRAM),BC     ; Set new loading addrs.
01ED    ED 53 EF26              LD      (BIlLAD),DE     ;
01F1    DD 22 EF24              LD      (BDSLAD),IX     ;
01F5    FD 22 EF22              LD      (CCPLAD),IY     ;
01F9    13                      INC     DE              ; Set BIOS entry.
01FA    13                      INC     DE              ;
01FB    13                      INC     DE              ;
01FC    ED 53 0001              LD      (BIOSENTRY),DE  ;
                        ;
0200    DD 21 001E              LD      IX,RAMDKMNT     ; RAM disk mount check.
0204    3A EF2C                 LD      A,(SIZRAM)      ;  Entry parameter. (RAM disk size)
0207    B7                      OR      A               ;  No format.
0208    CD EB69                 CALL    CALLX           ;
                        ;
020B    DD 21 0021              LD      IX,MDFYDPB      ; Modify disk parameter block.
020F    CD EB69                 CALL    CALLX           ;
0212    DD 21 001B              LD      IX,BIOSJTLD     ; BIOS jump table loading.
0216    CD EB69                 CALL    CALLX           ;
                        ;
0219    3A 024F                 LD      A,(RET_PRM)     ; Restore return parameter. (0 or 2)
021C    C9                      RET                     ;
                        ;
                        ;
                        ;       ************************************************
                        ;               CHECK USER BIOS HEADER
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Check of User BIOS header
                        ;                1. First 2 bytes of header is 'UB'?
                        ;                2. Check sum of header is OK?
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               Z-flag : Return information.
                        ;                        =1 : Header exists.
                        ;                        =0 : Header doesn't exists.
                        ;       <> preserved registers <>
                        ;               BC,DE,HL
                        ;
                        ;       CAUTION :
                        ;
021D                    CHK_HEAD:
021D    3A CBF0                 LD      A,(UB_HEAD)     ; User BIOS header check.
0220    FE 55                   CP      'U'             ;  Header is 'UB'?
0222    C0                      RET     NZ              ;  No.
0223    3A CBF1                 LD      A,(UB_HEAD+1)   ;
0226    FE 42                   CP      'B'             ;
0228    C0                      RET     NZ              ;  No.
                        ;
0229    E5                      PUSH    HL              ; Save registers.
022A    C5                      PUSH    BC              ;
022B    21 CBF0                 LD      HL,UB_HEAD      ; Sum check.
022E    06 10                   LD      B,16            ;  Header size.
0230    AF                      XOR     A               ;
                        ;
0231                    CHK_SUM:
0231    86                      ADD     A,(HL)          ;  Add header data.
0232    23                      INC     HL              ;  Next address.
0233    10 FC                   DJNZ    CHK_SUM         ;  Loop 16 times.
                        ;                               ; If result isn't 0,
0235    B7                      OR      A               ;  then non-exist User BIOS.
0236    C1                      POP     BC              ; Restore registers.
0237    E1                      POP     HL              ;
0238    C9                      RET                     ;
                        ;
                        ;       ************************************************
                        ;               USER BIOS AREA OVERWITE CHECK
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Check overwite flag in User BIOS area.
                        ;               If overwrite OK, then call release routine
                        ;               and clear header.
                        ;               if overwrite NG, then return with CY on.
                        ;                1. Using SETERR and RSTERR
                        ;                2. Replacing BDOS error vector
```

```
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               CY-flag : Return information
                        ;                         =0 -- Nrmal return
                        ;                         =1 -- Cannot overwrite
                        ;       <> preserved registers <>
                        ;               BC,DE,HL
                        ;
                        ;
0239                    RELEAS:
0239    3A CBFB                 LD      A,(UB_OVWRITE)  ; Check over write flag.
023C    B7                      OR      A               ;  Cannot over write?
023D    37                      SCF                     ;
023E    C8                      RET     Z               ; Yes. then return with carry on.
                        ;
023F    E5                      PUSH    HL              ; Save registers.
0240    D5                      PUSH    DE              ;
0241    C5                      PUSH    BC              ;
0242    21 024A                 LD      HL,REL_RET      ; Set return address.
0245    E5                      PUSH    HL              ;
0246    2A CBFC                 LD      HL,(UB_RELEASE) ; Get release routine addr.
0249    E9                      JP      (HL)            ; Go release routine!
024A                    REL_RET:
024A    B7                      OR      A               ; Carry off.
024B    C1                      POP     BC              ; Restore registers.
024C    D1                      POP     DE              ;
024D    E1                      POP     HL              ;
024E    C9                      RET                     ;
                        ;
                        ;       Work area
                        ;
024F                    RET_PRM:
024F                    DS      1               ; Return parameter area.
                        END
```

4.1.3.3 Using the user BIOS area

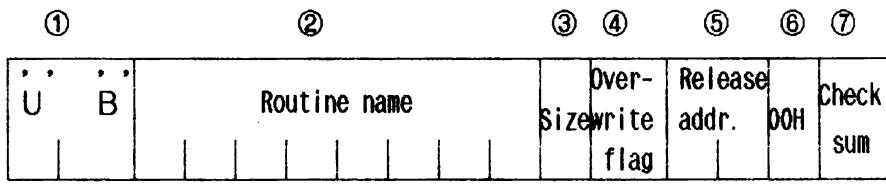
(1) Outline

The user BIOS area cannot be used by more than one program at a time. To identify which program is currently using the user BIOS area, a header is provided at the end of the user BIOS area. Application programs which are to use programs or data in the user BIOS area can identify the program or data that they are going to use by examining this header. Furthermore, when loading a program or data into the user BIOS area, they can check whether the user BIOS area is being used by another program by examining the header.

The contents of the user BIOS area remain unchanged until a system initialize is executed or its size is altered.

(2) Header

The format of the header is shown below.



The header is 16 bytes long and lies between ØCBFØH through ØCBFFH.

(a) Header fields

Header ID (ØCBFØH - ØCBF1H):
Identifies the header area. This field is set to 'UB' (ASCII).
Routine name (ØCBF2H - ØCBF9H):
Contains the name of the routine loaded in the user BIOS area. The routine name may be any string of ASCII codes.
Size (ØCBFAH):
Is a binary number indicating the size in 256-byte units of the routine loaded in the user BIOS area.
Overwrite flag (ØCBFBH):
Indicates whether the current routine can be overwritten by a new routine.
= ØØH: Overwrite is disabled.
= Nonzero: Overwrite is enabled.
Release address (ØCBFCH - ØCBFDH):
Contains the address of the routine which releases the area for the existing routine to load a new routine. The release routine is enabled only when the overwrite flag is nonzero. The release address must be within the user BIOS area. The release processing routine must end with a RET instruction.
ØCBFEH:
Not used. Always set to ØØH.
Checksum (ØCBFFH):
Contains the checksum byte which is calculated by successively subtracting the value of the 15 header bytes from ØØH.

(b) Overwrite flag

The overwrite flag must be set to ØØH for routines which, once loaded, must reside in the the user BIOS area. Such routines can be deleted from the the user BIOS area only by the program that called them or by system initialize processing.

The overwrite flag must be set to a nonzero value for routines which --
Modify the the user BIOS area when loaded,
Can restore the the user BIOS area into the original state, and
Can be overwritten by a new routine.

(c) Release processing

The user BIOS routine in the the user BIOS area must save the contents of the system area into the user BIOS area before making any attempt to alter the system area. The release processing routine restores the contents of the saved user BIOS area into the system area to establish the original system state when the user BIOS routine was loaded, and initializes the header to zeros. The release processing routine clears the header to zeros even if there is no need to restore the system area.

The release processing routine must fit in the 256 byte area at the end of the user BIOS area (ØCBØØH through ØCBFFH). It must end with a RET instruction.

(3) Loading a user BIOS routine

The application program which is to load a user BIOS routine into the user BIOS area must make checks shown in the figure below to verify whether the the user BIOS area is available for the user BIOS routine.

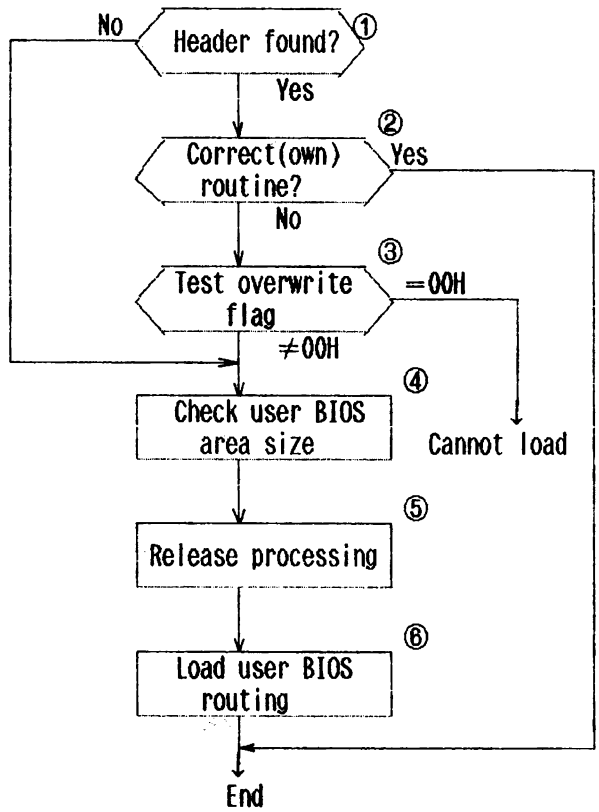


Fig. 4.1.3 Loading a User BIOS Routine

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.

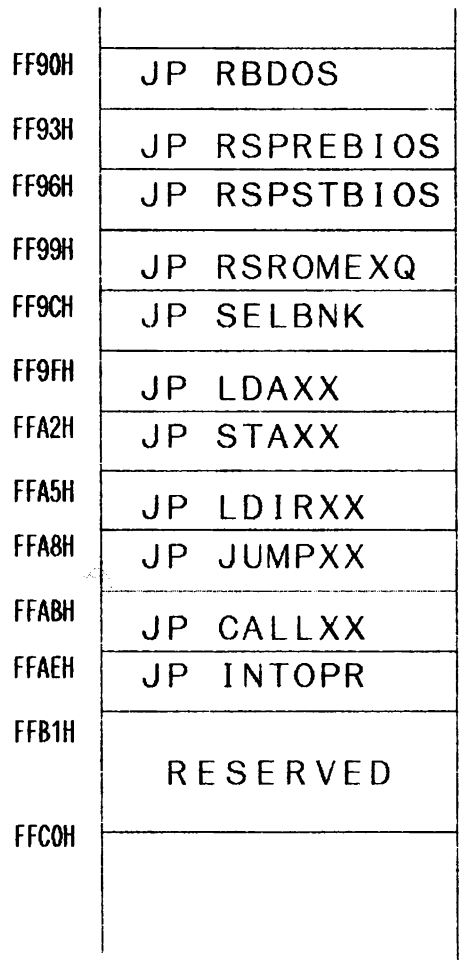


Figure 4.2.1 Resident Jump Table Organization

(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.

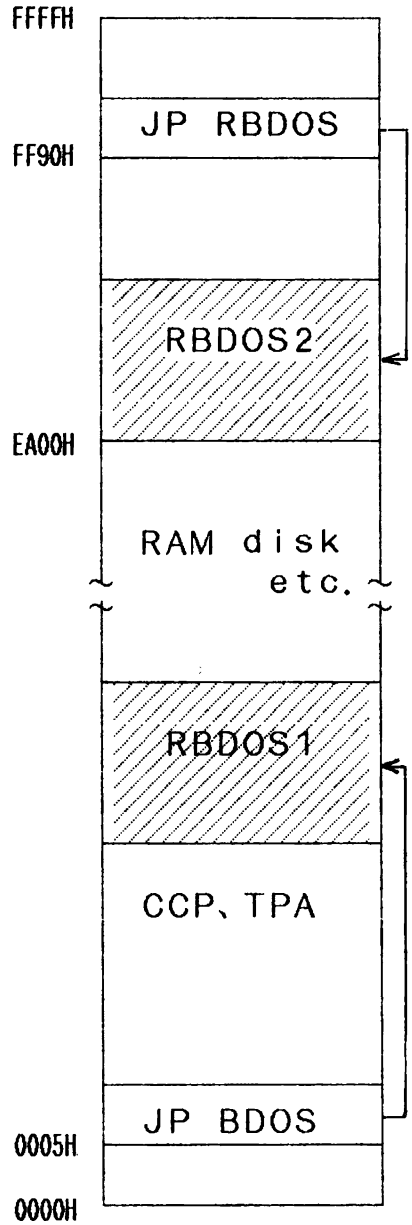


Figure 4.2.3 Relationship between RBDOS1 and RBDOS2

(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.

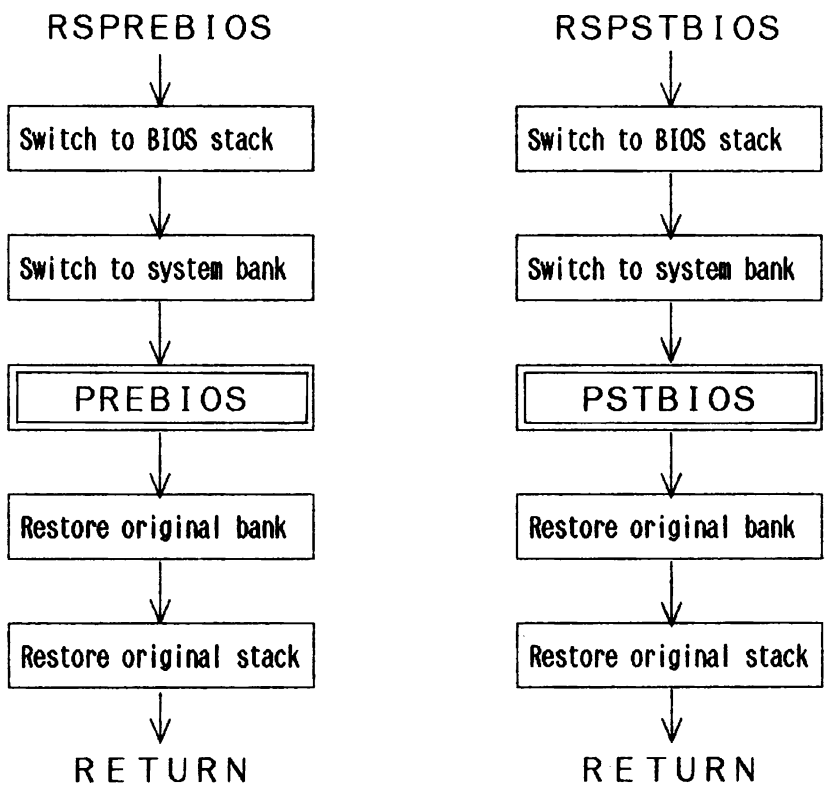


Figure 4.2.4 RSPREBIOS/RSPSTBIOS Processing Flow

PREBIOS and PSTBIOS processing flowcharts are shown below.

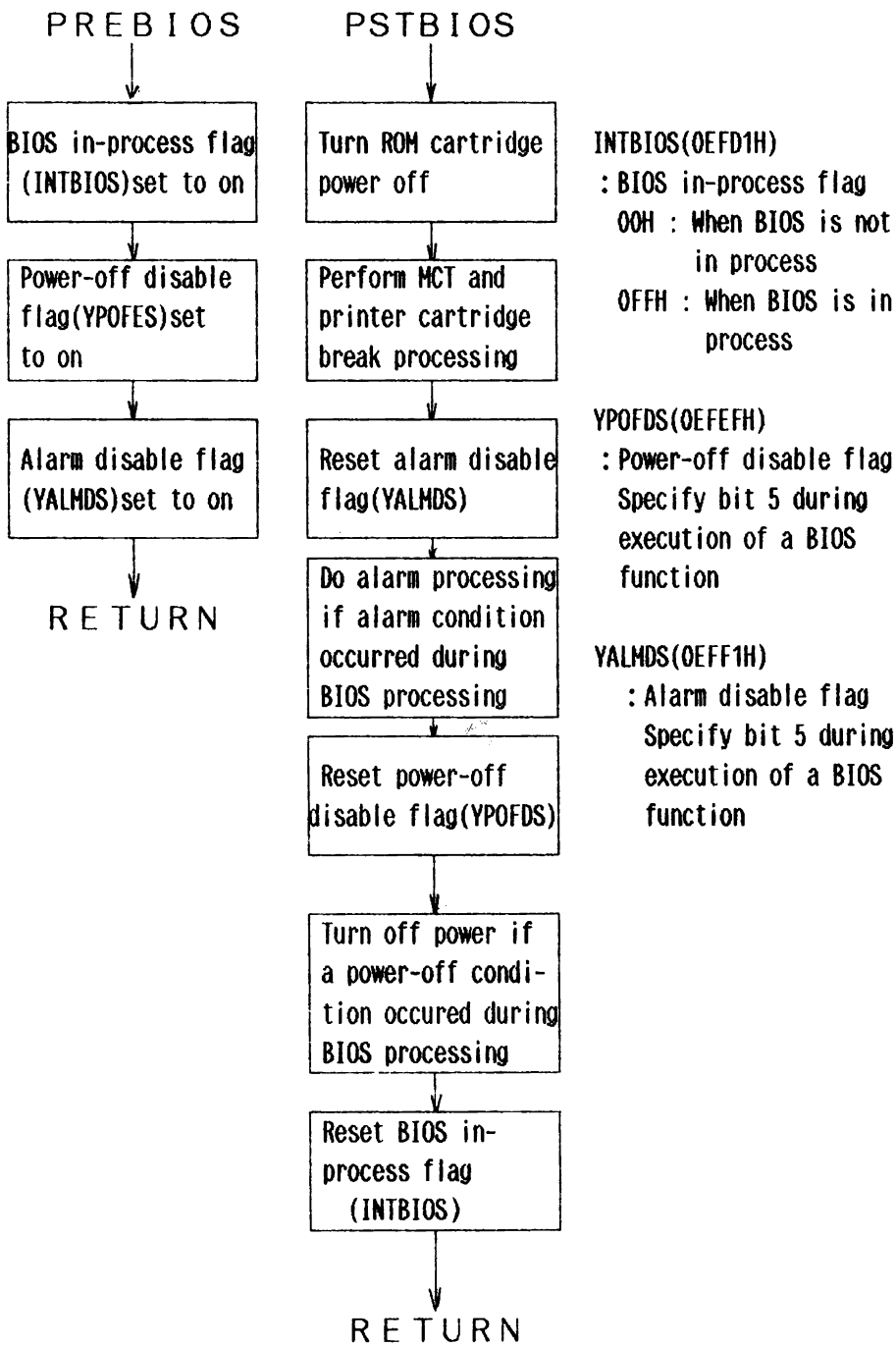


Figure 4.2.5 PREBIOS/PSTBIOS Processing Flow

(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.

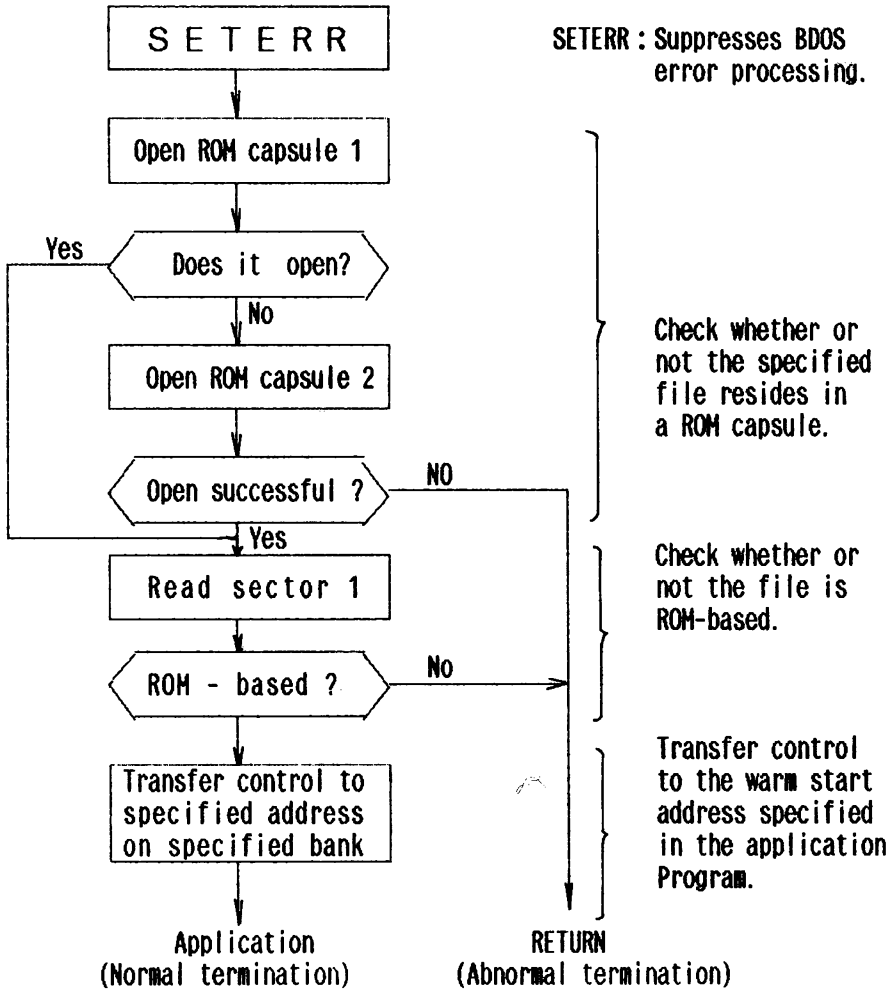


Figure 4.2.6 RSROMEXQ Processing Flow

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

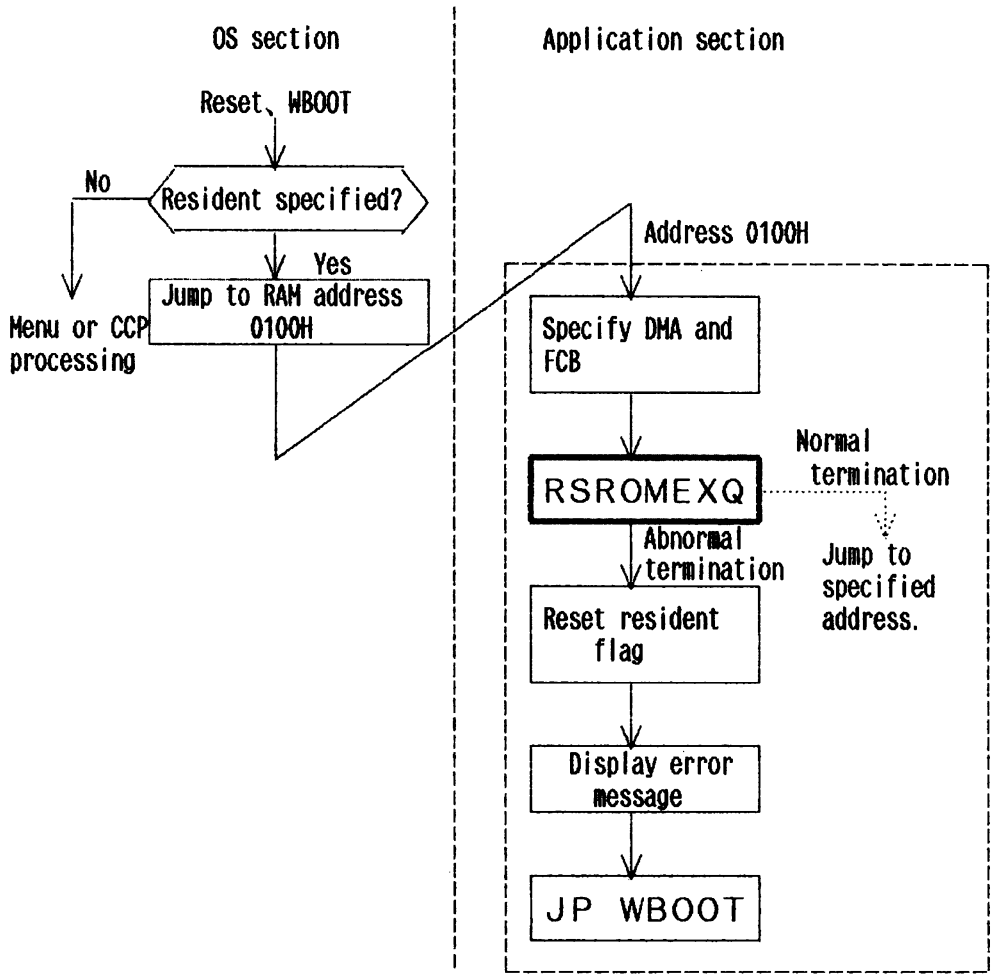


Figure 4.2.7 ROM-based Resident Program Processing Flow

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

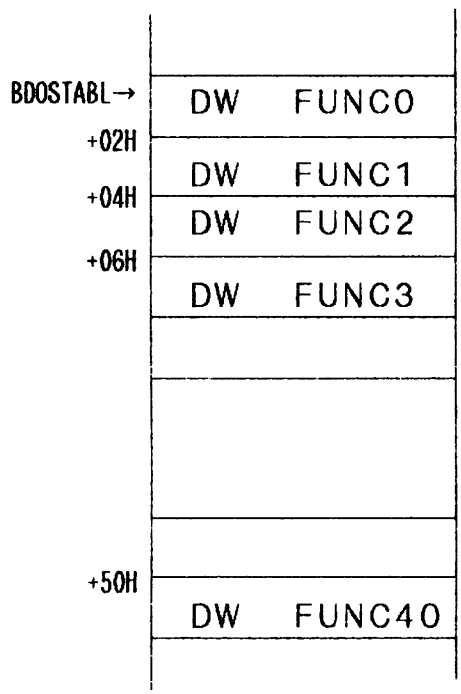
The OS ROM jump table includes 33 routines. This subsection describes the specifications for individual routines.

(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.

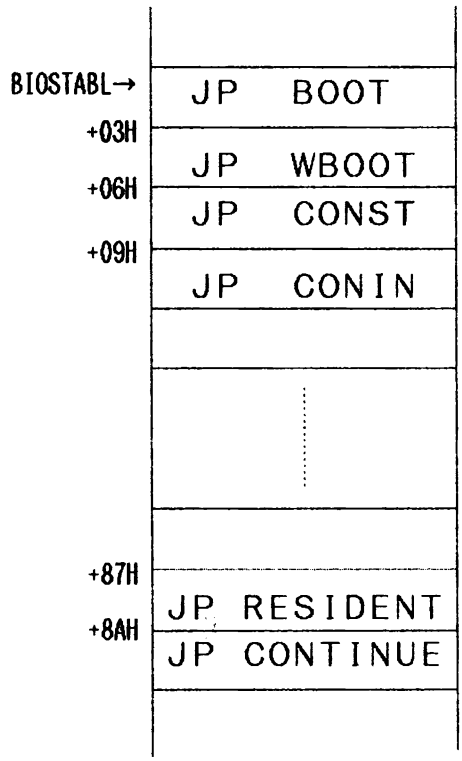


(2) BIOSTABL (0006H)

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.

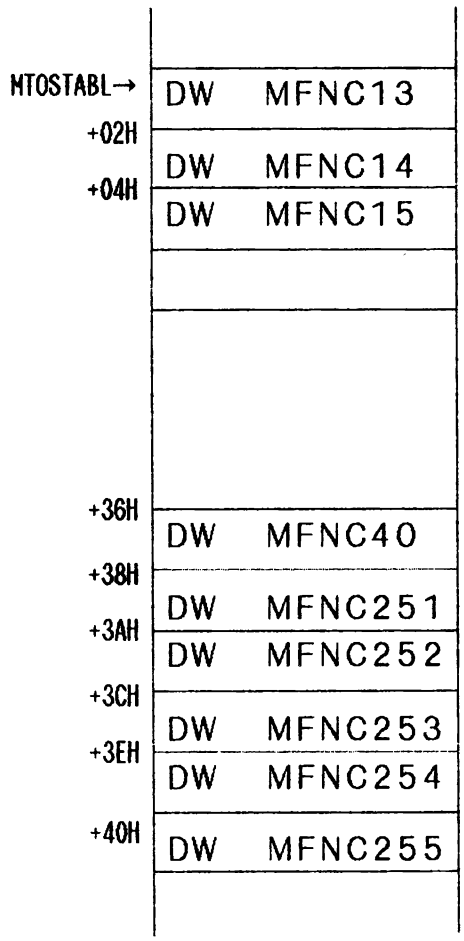


(3) MTOSTABL (0009H)

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.

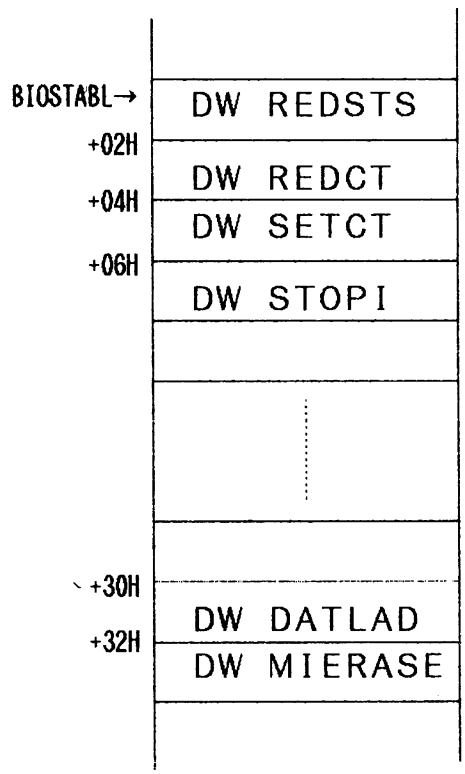


(4) MIOSTABL (ØØØCH)

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 BI1LAD (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)

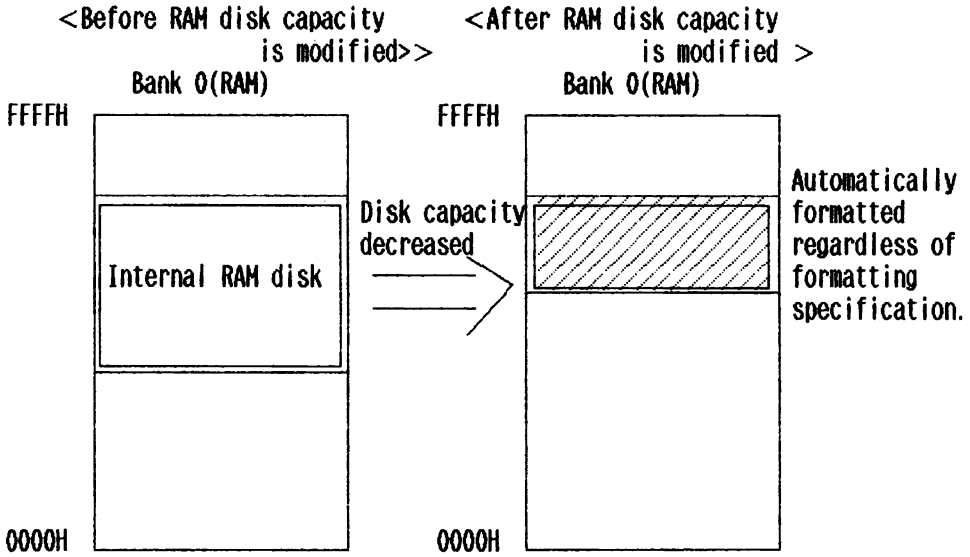
Explanation:

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.

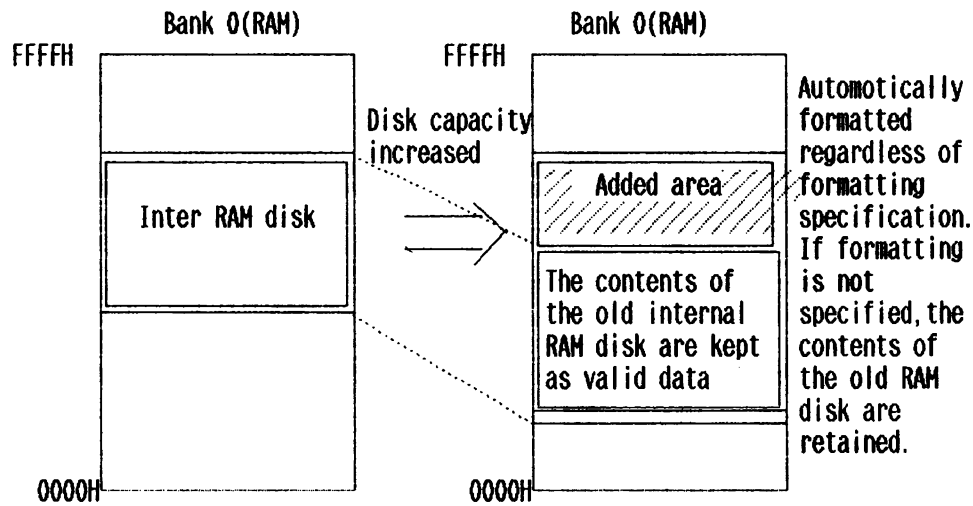
Notes:

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
```

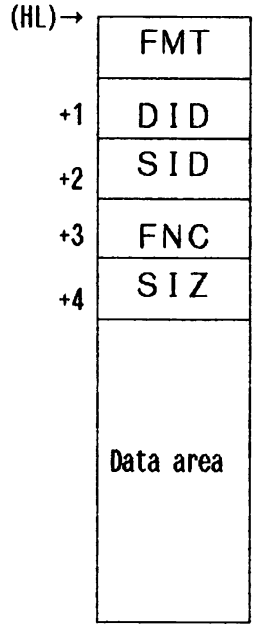
Return parameters:

```
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 06                   LD      B,6             ;Display 16 byte data on each line
                        ;                               ;therefore it takes 6 line to list out
                        ;                               ;128 byte data
01BE                    PRDT00:
01BE    C5                      PUSH    BC
01BF    06 10                   LD      B,16
01C1    CD 01E6                 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 terminater of string.
                        ;
                        ;       on exit  ; none
                        ;
                        ;       Registers are not preserved.
                        ;
01DC                    DSPMSG:
01DC    7E                      LD      A,(HL)          ;Display data
01DD    B7                      OR      A               ;Check terminater
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                    DSPDT1(
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 0289              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 (ØØ39H)

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 Øs) is returned.

(2Ø) XUSRSCRN (ØØØ3CH)

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 (ØF29ØH) and those starting at LWORKBF (ØF2B7H).
2. Output the VRAM starting address to I/O ports. (ØPØ8H)
3. Output the Y-direction offset to I/O ports. (ØPØ9H)

(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.

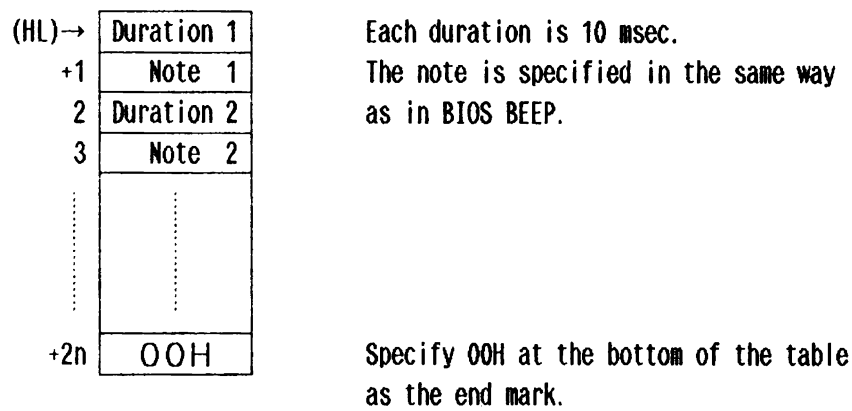
(24) MELODY (0048H)

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 (Fl)
= 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 (ØØ6ØH)

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
= Ø : 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 8ØØØH 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     ;Carlage 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     ;Head 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      RCD06           ;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
                        ;       RET
0191    C9
                        ;
                        ;*****************
                        ;*   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
01B6                    PRMCTDT00:
01B6    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 0226         ;       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
```

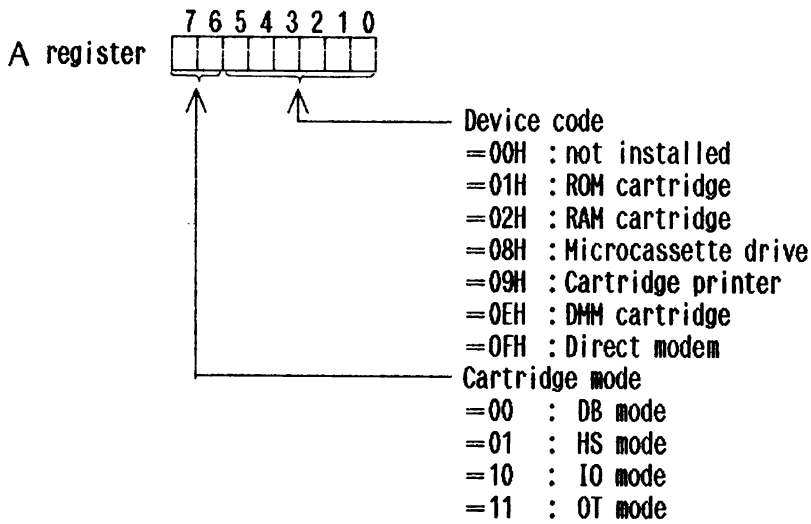
(33) CHKMOD (ØØ63H)

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.

4.3 Hooks

4.3.1 General

The PINE is provided with some hooks to facilitate the application program to extend the capabilities of the interrupt and BIOS functions. The PINE hooks are:

1. Alarm hook
2. Interrupt hook
3. BIOS hook

This section describes and show the use of the PINE hooks.

4.3.2 Hook Structures

4.3.2.1 Hook jump table

The hook jump table is located between ØFFCØH and ØFFEFH. Its structure is shown below.

FFC0H	ALMHK0	JP EF1FH
FFC3H	ALMHK1	JP EF1FH
FFC6H	ALMHK2	JP EF1FH
FFC9H	ALMHK3	JP EF1FH
FFCCH	ALMHK4	JP EF1FH
FFCFH	RESERVED	JP EF1FH
FFD2H	HK8251	JP EF1FH
FFD5H	ICFHOOK	JP EF1FH
FFD8H	OVFHOOK	JP EF1FH
FFDBH	EXTHOOK	JP EF1FH
FFDEH	TMDT83	JP EF1FH
FFE1H	TMDT85	JP EF1FH
FFE4H	TMDT86	JP EF1FH
FFE7H	BIOSHK	JP EF1FH
FFEAH	RESERVED	JP EF1FH
FFEDH	RESERVED	JP EF1FH

ØEF1F contains a RET instruction.

4.3.2.2 Hook types

The PINE hooks are divided into the following types:
(1) Alarm/wake hooks (5)
    ALMHKØ   (Alarm hook Ø)
    ALMHK1   (Alarm hook 1)
    ALMHK2   (Alarm hook 2)
    ALMHK3   (Alarm hook 3)
    ALMHK4   (Alarm hook 4)
(2) Interrupt hooks (4)
    HK8251   (RxRDY interrupt hook)
    ICFHOOK   (ICF interrupt hook)
    OVFHOOK   (OVF interrupt hook)
    EXTHOOK   (EXT interrupt hook)
(3) BIOS hooks (4)
    TMDT83 (TIMDAT hook)
    TMDT85 (TIMDAT hook)
    TMDT86 (TIMDAT hook)
    BIOSHK (BIOS hook)
(4) Reserved (3)

HK8251 is not supported by OS version 1.Ø.

4.3.2.3 Hook structure

The PINE executes its program modules while switching four banks. Depending on which bank the PINE is currently running, the hook processing mode is divided into the following two types:
1. Hooks in which control is transferred to the hook address after bank Ø (RAM) has been selected.
2. Hooks in which control is transferred to the hook address without switching the active bank (system bank).

(1) Hooks which run without switching to bank Ø (RAM)
The active bank is unpredictable when an interrupt occurs. For this reason, interrupt hook processing proceeds as follows:
1. Switch to bank Ø (RAM).
2. Cause the CPU to jump to the specified hook address.
3. Switch to the original bank.

Hooks which run in this mode include OVFHOOK, ICFHOOK, and EXTHOOK.

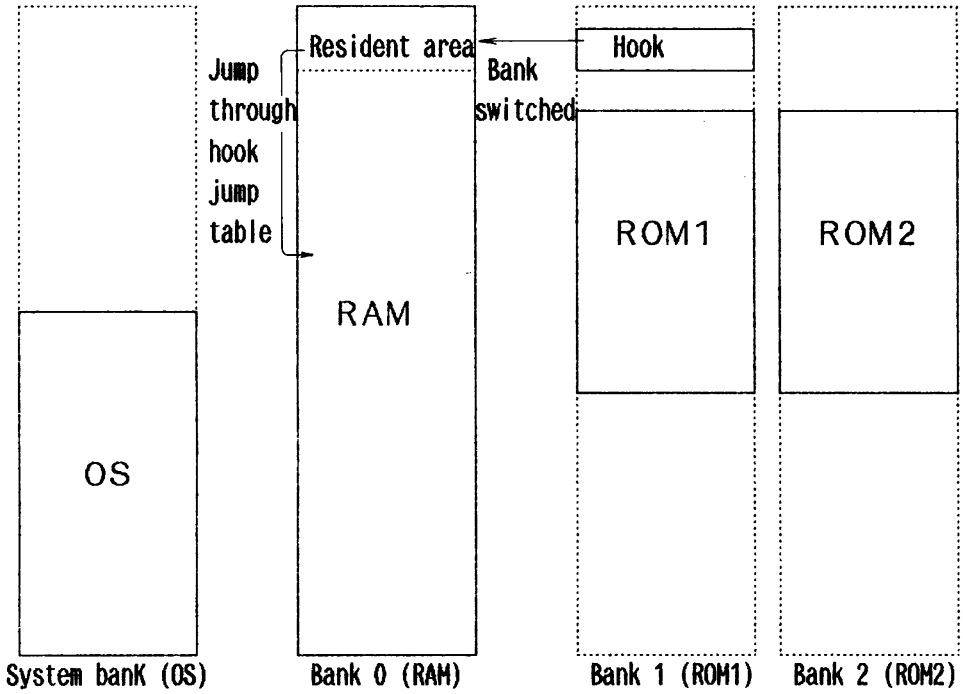


Fig. 4.3.1 Hooks Which Switch Banks

(2) Hooks which run without switching banks.
Hooks which are entered during execution of a module on OS ROM do not switch banks. When rewriting this type of hook, therefore, it is necessary to specify an address not lower than 8000H. Hooks of this type include: ALMHK0, ALMHK1, ALMHK2, ALMHK3, ALMHK4, HK8251, TMDT83, TMDT85, TMDT86, and BIOSHK.

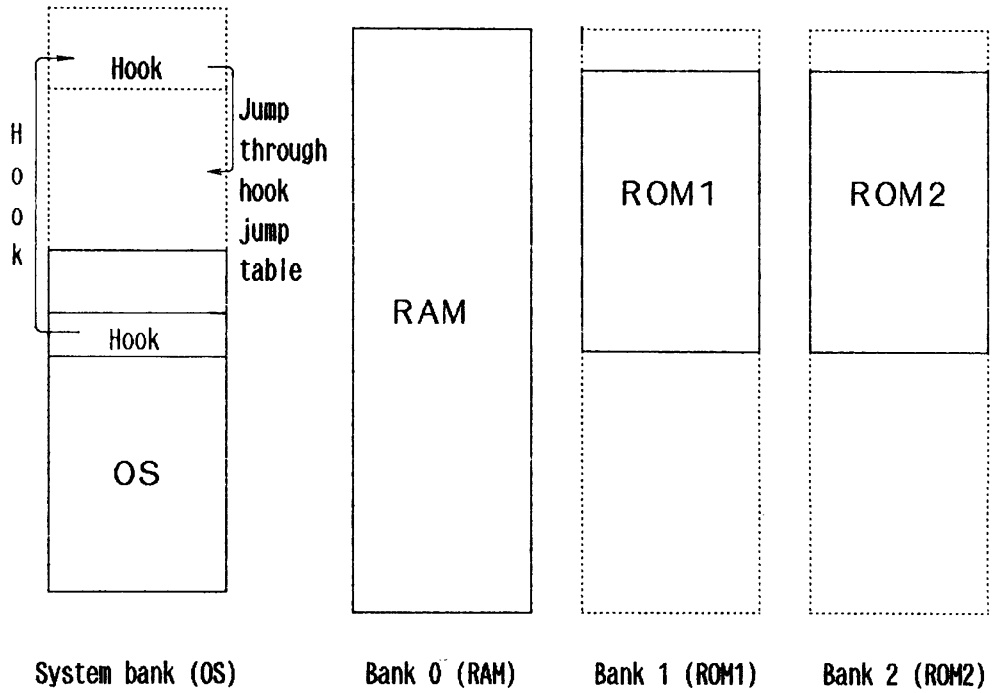


Fig. 4.3.2 Hooks Which Do Not Switch Banks

4.3.2.4 Programming notes

Since hooks are embedded in interrupt and BIOS processing modules, the following programming notes must be taken:
(1) Do not call BIOS/BDOS from a hook.
BIOS and BDOS establish their own stack area at the beginning of execution. Any call to BIOS or BDOS from a hook may destroy the stack area, causing a program hangup. To use a BIOS or BDOS function from a hook, call directly the BIOS or BDOS function on the OS ROM.
(2) Set up a stack area.
A stack overflow condition may occur if the called hook uses the stack established by the calling module. To prevent stack overflows, the hook processing modules must establish their own stack area. The stack area for hooks which do not switch banks must be located in a RAM area not lower than 8000H.
(3) Registers must be saved by the calling module.
The modules that receive control from a hook save the contents of only the required registers. The hook processing modules, therefore, must save all registers that are to be used during their execution. Failure to take this measure when implementing ICFHOOK or EXTHOOK may cause a program hangup on return from the interrupt processing module.

(4) Maintain the interrupt state.
Some interrupt modules run in the DI state and prevent subsequent interrupts. Accordingly, the hook processing modules must keep track of the interrupt state by themselves. The interrupt state when hooks are entered are as follows:

ALMHKØ	DI state
ALMHKl	DI state
ALMHK2	EI state, 75Ø8 disabled
ALMHK3	EI state, 75Ø8 disabled
ALMHK4	EI state
HK8251	DI state
ICFHOOK	DI state
OVFHOOK	DI state
EXTHOOK	DI state
TMDT83	EI state, 75Ø8 disabled
TMDT85	EI state, 75Ø8 disabled
TMDT86	EI state, 75Ø8 disabled
BIOSHK	EI state

(5) Disable interrupts when rewriting a hook.
Disable interrupts when rewriting the hook jump table. Particularly, be sure to disable interrupts when rewriting address data, one byte at a time.
(6)
The hook jump table is initialized during system initialize and reset.

4.3.3 Alarm Hooks

The PINE has five hooks for alarm processing. These hooks are used for automatic alarm/wake data update processing. The locations of the hooks are listed below.

Alarm hook Ø: Within the module that is activated when an alarm occurs in the power-off state and immediately before the alarm screen is displayed.

Alarm hook 1: Within the module that is activated when an alarm occurs in the power-off state and immediately after the alarm screen disappears.

Alarm hook 2: Within the module that is activated when an alarm occurs in the power-on state and immediately before the alarm screen is displayed.

Alarm hook 3: Within the module that is activated when an alarm occurs in the power-on state and immediately after the alarm screen disappears.

Alarm hook 4: Within the module that is activated when an alarm occurs and immediately before the module is exited due to a condition such as the depression of the ESC key.

The relationships between the alarm processing modules are shown in Figures. 4.3.3 through 4.3.5.

The alarm/wake update processing module, for example, sets the next alarm or wake time using alarm hook Ø and alarm hook 2 when an alarm or wake condition occurs.

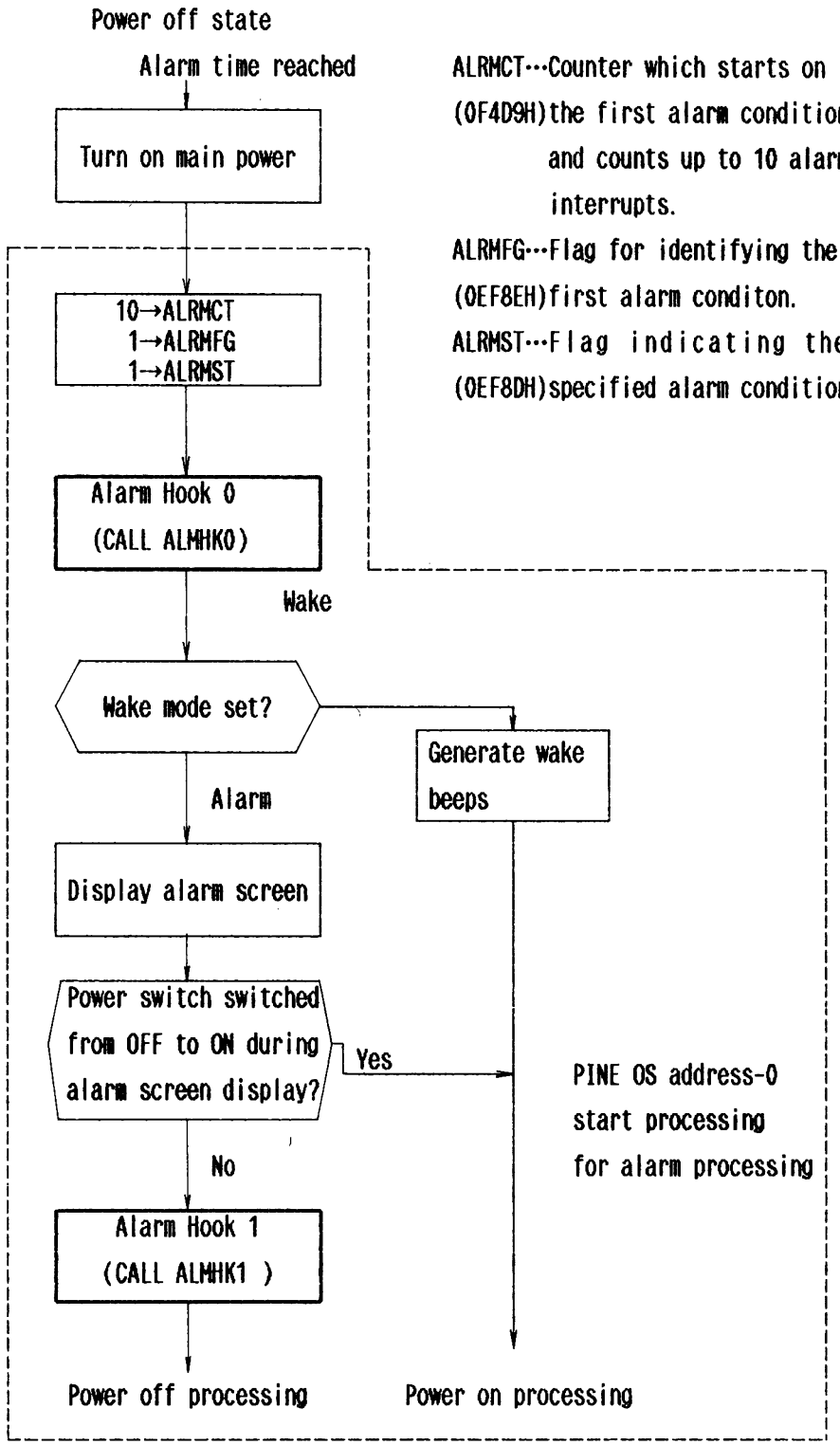


Fig. 4.3.3 Alarm Processing in the Power-off State

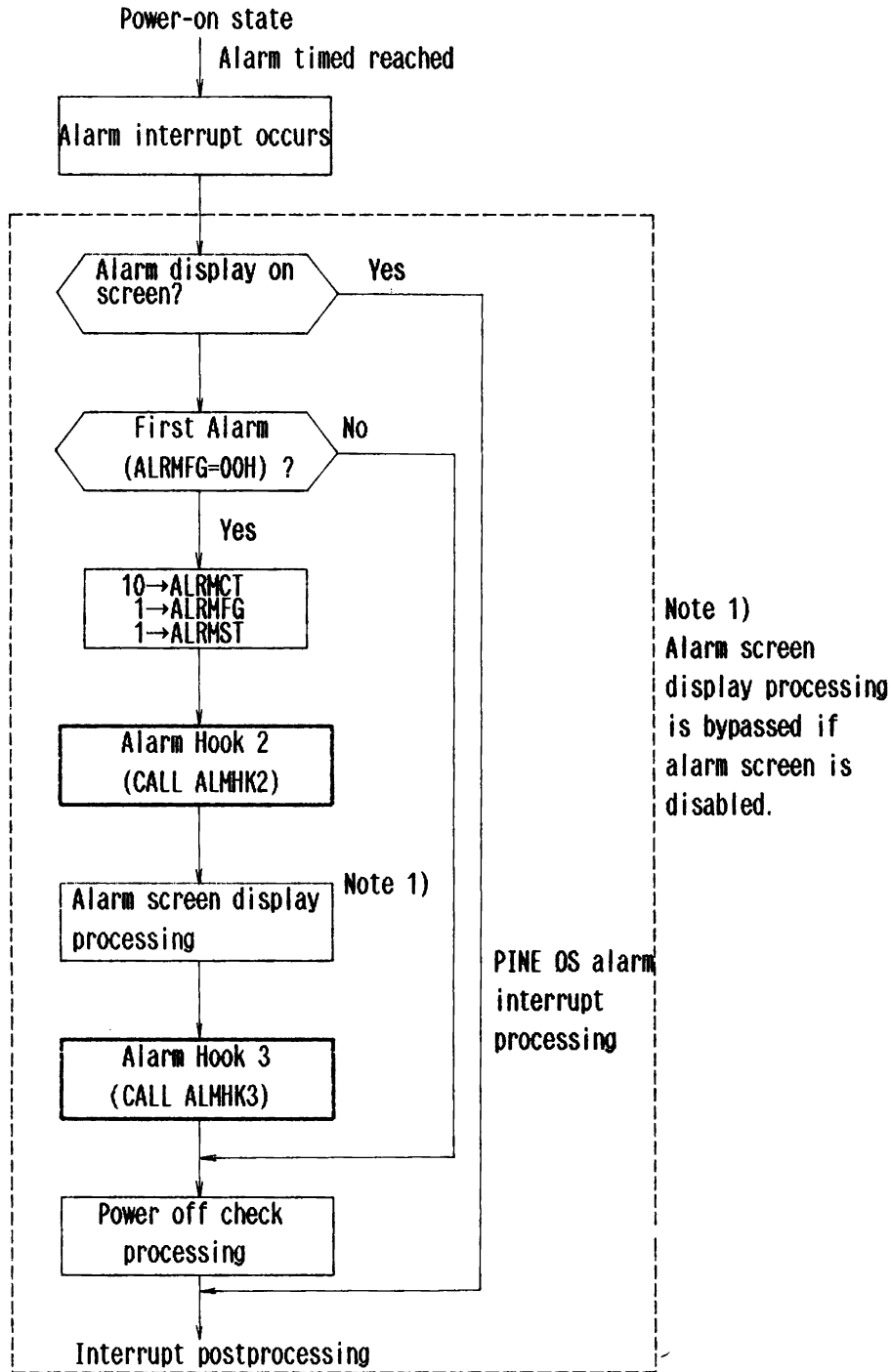
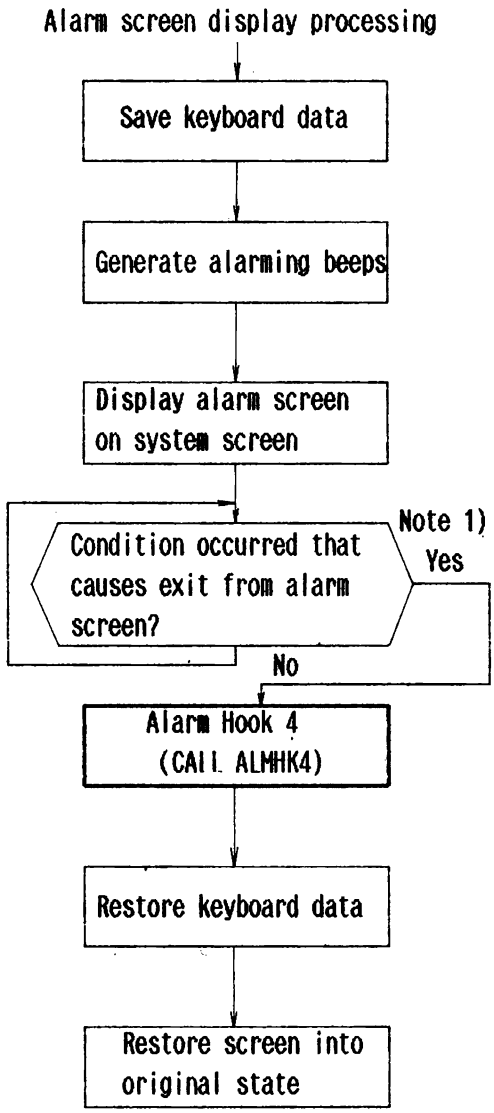


Fig. 4.3.4 Alarm Processing in the Power-on State



Note 1) Conditions that causes the alarm screen processing to be exited include:
- ESC is typed.
- 50 seconds have elapsed.
- Power switch is turned off.
- Batty is discharged.

Fig. 4.3,5 Alarm Screen Display Processing

4.3.4 Interrupt Hooks

The PINE has four hooks for interrupt processing. The locations of the hooks are listed below.

HK8251: Within the ART interrupt processing module and immediately before data is received.
ICFHOOK: Within the ICF interrupt processing module
OVFHOOK: Within the OVF interrupt processing module and immediately before blink processing.
EXTHOOK: Within the EXT interrupt processing module

HK8251 is not supported by OS Export version 1.Ø.

4.3.4.1 HK8251

HK8251 is located on OS ROM and provides extended ART interrupt processing. Figure 4.3.6 shows the relationship of HK8251 to interrupt processing.

As seen from the figure, HK8251 just returns control doing nothing when an interrupt occurs unless the RSIOX OPEN function has been executed.

HK8251 is not supported by OS version 1.Ø.

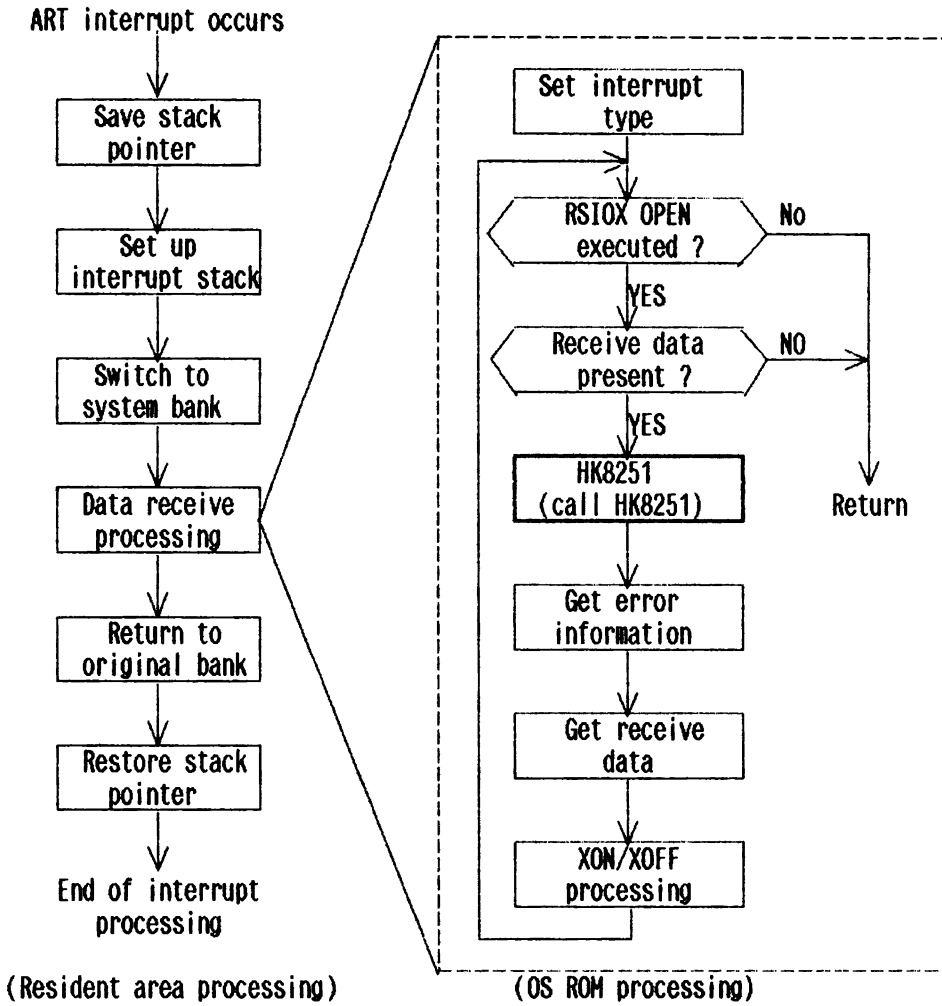


Fig. 4.3.6 ART Interrupt Processing

(2) ICFHOOK, OVFHOOK, and EXTHOOK
During ICF, OVF, and EXT interrupt processing, the active bank is switched to bank Ø (RAM) before the pertinent hook is called. After this, the bank is returned to the original bank.

Figure 4.3.7 shows the relationship of ICFHOOk, OVFHOOk, and EXTHOOK to the interrupt processing modules.

Since the minimum required size of stack area is reserved during ICF, OVF, and EXT processing, it will be necessary to reserve a larger stack area when interrupt processing is to be expanded. Since neither ICF nor EXT interrupt processing modules reset the interrupt signal, it must be reset by the corresponding hook.

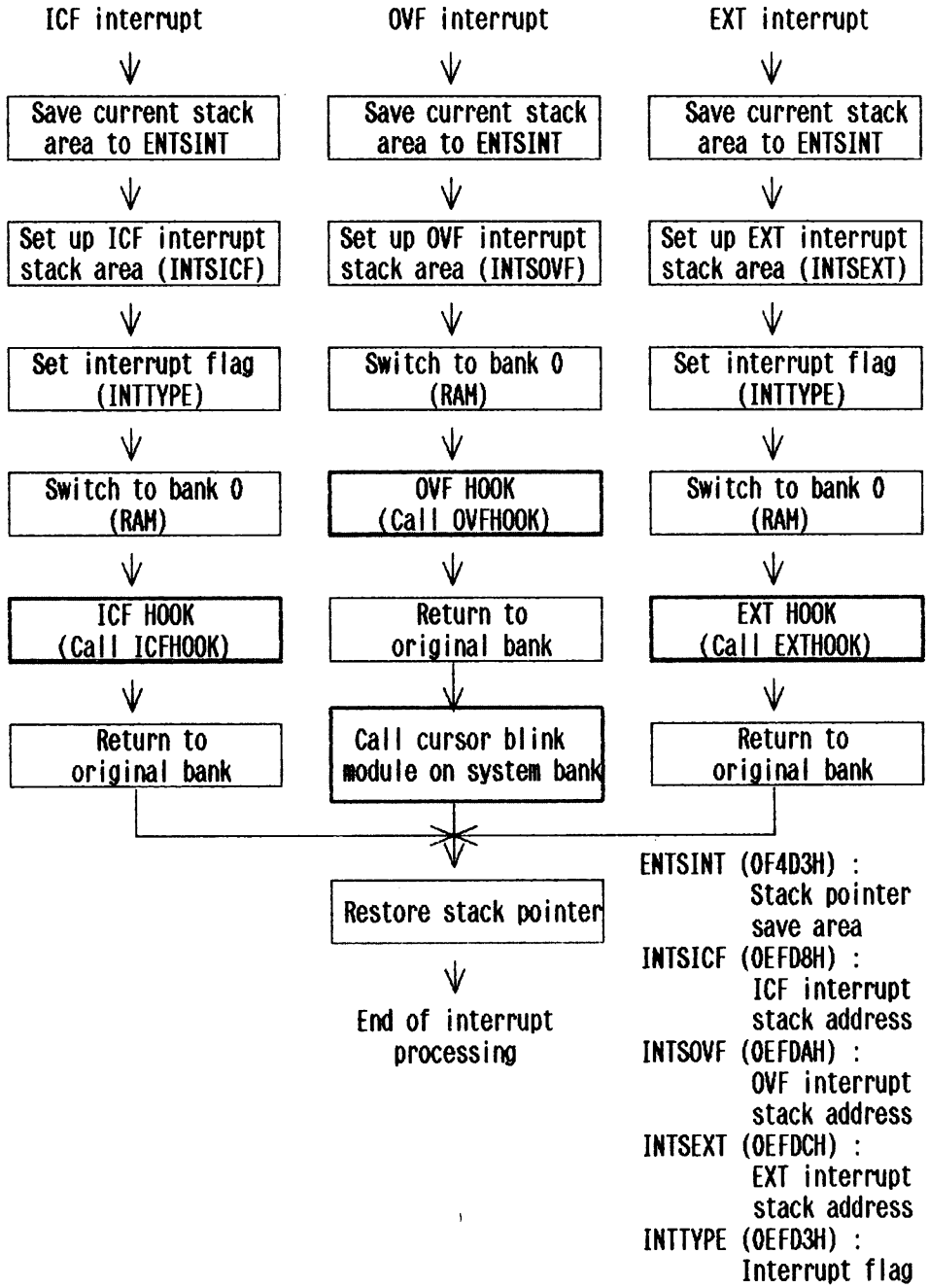


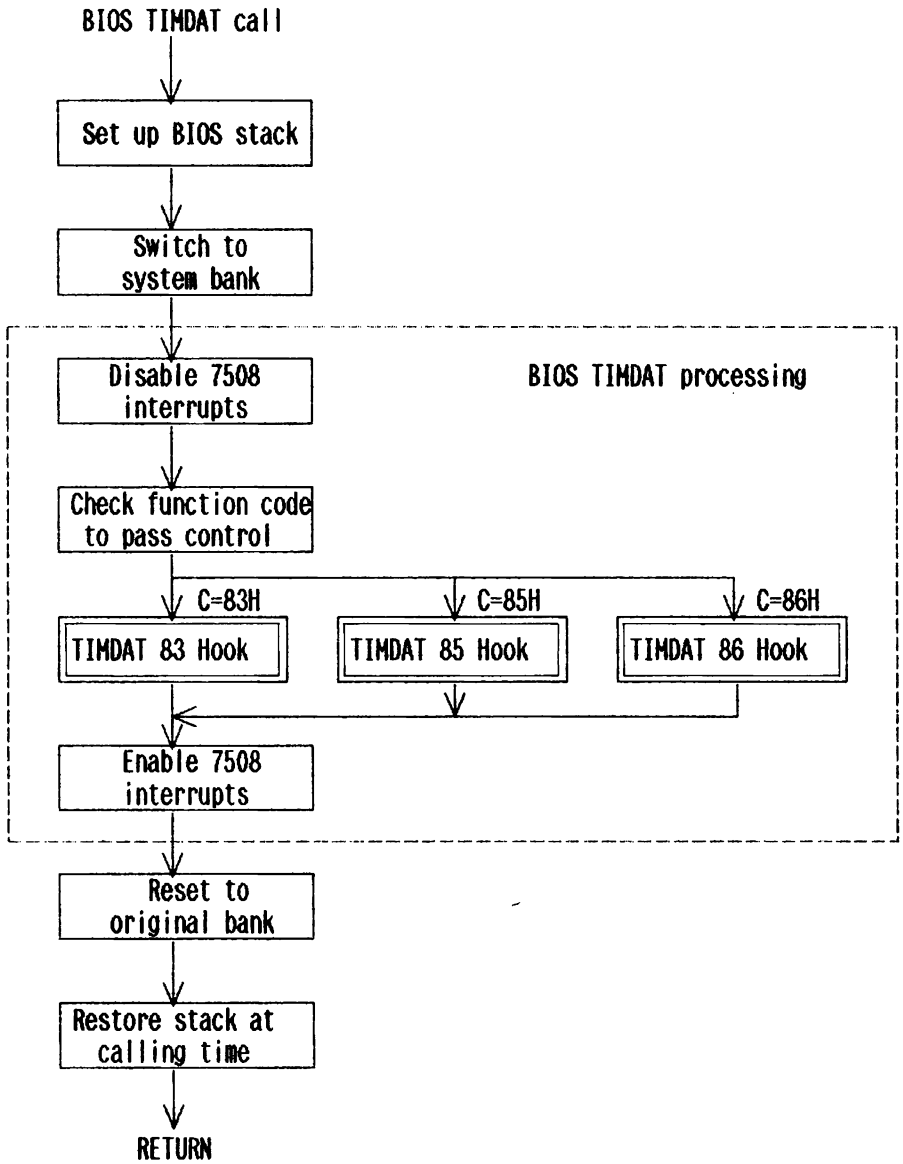
Fig. 4.3.7 ICF/OVF/EXT Interrupt Processing

4.3.5 TIMDAT Hooks

The PINE has the following three hooks for extended BIOS TIMDAT functions.:

TMDT83: Called during BIOS TIMDAT function with C set to 83H.
TMDT85: Called during BIOS TIMDAT function with C set to 85H.
TMDT86: Called during BIOS TIMDAT function with C set to 86H.

These hooks are used to extend the TIMDAT functions. Only the DE registers are preserved when TIMDAT is called.



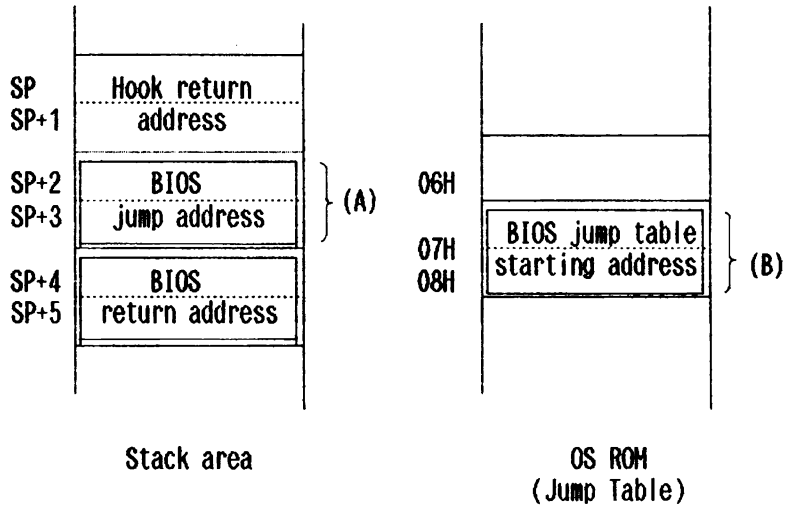
(TIMDAT processing proper is indicated by broken lines.)

Fig. 4.3.8 TIMDAT Processing

4.3.6 BIOS Hook

BIOSHK is taken when BIOS is called by BDOS or by an application program and provides extended BIOS functions.

BIOSHK processing must proceed as follows:
(1) Calculate the BIOS function number.



The logical BIOS number can be calculated using the equation:

$$\{(A) - (B)\}/3$$

where (A) is the BIOS jump address and (B) is the BIOS jump table starting address. For example, the result of the above computation for WBOOT is Ø1H and that for CONST is Ø2H.

(2) Standard BIOS function processing
If the result of step (1) is found to be a standard BIOS call, BIOSHK immediately returns control after restoring the registers into the state before BIOSHK was entered.
(3) Extended BIOS function processing
If the result of step (1) is found to be an extended BIOS call, BIOSHK transfers control to the application program for the extended service. When the system BIOS service is to be executed after the extended processing is completed, BIOSHK restores the registers into the state before BIOSHK was entered and returns control to the calling module. When bypassing the system BIOS service after the extended BIOS processing, BIOSHK returns control directly to the point at the BIOS return address that is stored in the stack area shown in paragraph (1). This allows the system to return control to the program that called BIOS after executing PSTBIOS (including bank switching and stack swapping).

See 3.3.3, "BIOS Hook" for use examples of BIOS hooks.

```
                        ;       ************************************************
                        ;               HOOK SAMPLE PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is testing hook.
                        ;
                        ;       <> assemble condition <>
                        ;
                        .Z80
                        ;
                        ;       <> loading address <>
                        ;
                        .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
                        ;       BIOS entry
                        ;
EB03                    WBOOT           EQU     0EB03H          ; Warm Boot 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
EB87                    RESIDENT        EQU     WBOOT   +84H    ; Resudent entry
                        ;
                        ;       System area
                        ;
EF94                    TOPRAM          EQU     0EF94H          ; Top of User BIOS
EF2D                    USERBIOS        EQU     0EF2DH          ; Size of User BIOS area
F52E                    DISBNK          EQU     0F52EH          ; Destination bank for CALLX
F53E                    RZIER           EQU     0F53EH          ; Interrupt enable register
                        ;
FFC0                    HOOKTBL         EQU     0FFC0H          ; Hook table top address
                        ;
                        ;          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)
                        ;
                        ;
CBF0                    UB_HEAD         EQU     0CBF0H          ; Top addr of User BIOS area's header
CBFB                    UB_OVWRITE      EQU     UB_HEAD +11     ;  Over write flag
CBFC                    UB_RELEASE      EQU     UB_HEAD +12     ;  Release address
                        ;
                        ;       OS ROM jump table
                        ;
0006                    BIOSJT          EQU     00006H
003C                    XUSRSCRN        EQU     0003CH
                        ;
                        ;       Resident jump table
                        ;
FF9C                    SELBNK          EQU     0FF9CH
                        ;
000D                    CR              EQU     0DH
000A                    LF              EQU     0AH
0012                    CLS             EQU     12H
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Using system hook jump program.
                        ;
                        ;       CAUTION :
                        ;               This program uses User BIOS area.
                        ;               Usually, you must check that another
                        ;               already used this area. But this program
                        ;               doesn't check it.
                        ;
                        ;               This program doesn't do resetting hook.
                        ;               So, you wish to stop test hook, you must
                        ;               push reset bottom.
                        ;
0100                    MAIN:
0100    C3 03AD                 JP      START           ; Programm start
                        ;
                        ;       *****   Output message data
                        ;
0103                    MSGX;
0103    2A 2A 2A 20             DB      '*** User BIOS size error ***',CR,LF
0107    55 73 65 72
010B    20 42 49 4F
010F    53 20 73 69
0113    7A 65 20 65
0117    72 72 6F 72
011B    20 2A 2A 2A
011F    0D 0A
0121    20 20 53 65             DB      '  Set User BIOS size to 2 k bytes!!'
0125    74 20 55 73
0129    65 72 20 42
012D    49 4F 53 20
0131    73 69 7A 65
0135    20 74 6F 20
0139    32 20 6B 20
013D    62 79 74 65
0141    73 21 21
0144    0D 0A                   DB      CR,LF
0146    00                      DB      00H
                        ;
0147                    MSG0:
0147    12 00                   DB      CLS,00H
                        ;
```

```
0149                            MSG1:
0149    0D 0A                           DB      CR,LF
014B    2A 2A 2A 20                     DB      '*** Hook test program ***'
014F    48 6F 6F 6B
0153    20 74 65 73
0157    74 20 70 72
015B    6F 67 72 61
015F    6D 20 2A 2A
0163    2A
0164    0D 0A                           DB      CR,LF
0166    20 20 20 31                     DB      '   1   -- ALMHKX  test',CR,LF
016A    20 20 20 2D
016E    2D 20 41 4C
0172    4D 48 4B 58
0176    20 20 74 65
017A    73 74 0D 0A
017E    20 20 20 32                     DB      '   2,3 -- Reserve    ',CR,LF
0182    2C 33 20 2D
0186    2D 20 52 65
018A    73 65 72 76
018E    65 20 20 20
0192    20 20 0D 0A
0196    20 20 20 33                     DB      '   3   -- HK8251  test',CR,LF
019A    20 20 20 2D
019E    2D 20 48 4B
01A2    38 32 35 31
01A6    20 20 74 65
01AA    73 74 0D 0A
01AE    20 20 20 34                     DB      '   4   -- ICFHOOK test',CR,LF
01B2    20 20 20 2D
01B6    2D 20 49 43
01BA    46 48 4F 4F
01BE    4B 20 74 65
01C2    73 74 0D 0A
01C6    20 20 20 35                     DB      '   5   -- OVFHOOK test',CR,LF
01CA    20 20 20 2D
01CE    2D 20 4F 56
01D2    46 48 4F 4F
01D6    4B 20 74 65
01DA    73 74 0D 0A
01DE    20 20 20 36                     DB      '   6   -- EXTHOOK test',CR,LF
01E2    20 20 20 2D
01E6    2D 20 45 58
01EA    54 48 4F 4F
01EE    4B 20 74 65
01F2    73 74 0D 0A
01F6    20 20 20 37                     DB      '   7   -- TIMDAT  test',CR,LF
01FA    20 20 20 2D
01FE    2D 20 54 49
0202    4D 44 41 54
0206    20 20 74 65
020A    73 74 0D 0A
020E    20 53 65 6C                     DB      ' Select one of 1 to 7 -- '
0212    65 63 74 20
0216    6F 6E 65 20
021A    6F 66 20 31
021E    20 74 6F 20
0222    37 20 2D 2D
0226    20
0227    00                              DB      00H
                                ;
0228                            MSG3:
0228    0D 0A                           DB      CR,LF
022A    0D 0A                           DB      CR,LF
022C    2A 2A 2A 20                     DB      '***  Alarm hook test  ***'
0230    20 41 6C 61
0234    72 6D 20 68
0238    6F 6F 6B 20
023C    74 65 73 74
0240    20 20 2A 2A
0244    2A
0245    0D 0A                           DB      CR,LF
0247    20 20 20 31                     DB      '   1 -- ALMHK1',CR,LF
024B    20 2D 2D 20
024F    41 4C 4D 48
0253    4B 31 0D 0A
0257    20 20 20 32                     DB      '   2 -- ALMHK2',CR,LF
025B    20 2D 2D 20
025F    41 4C 4D 48
0263    4B 32 0D 0A
0267    20 20 20 33                     DB      '   3 -- ALMHK3',CR,LF
026B    20 2D 2D 20
026F    41 4C 4D 48
0273    4B 33 0D 0A
0277    20 20 20 34                     DB      '   4 -- ALMHK4',CR,LF
027B    20 2D 2D 20
027F    41 4C 4D 48
0283    4B 34 0D 0A
0287    20 20 20 35                     DB      '   5 -- ALMHK5',CR,LF
028B    20 2D 2D 20
028F    41 4C 4D 48
0293    4B 35 0D 0A
0297    20 53 65 6C                     DB      ' Select one of 1 to 5 -- '
029B    65 63 74 20
029F    6F 6E 65 20
02A3    6F 66 20 31
02A7    20 74 6F 20
02AB    35 20 2D 2D
02AF    20
02B0    00                              DB      00H
                                ;
02B1                            MSG4:
02B1    0D 0A                           DB      CR,LF
02B3    0D 0A                           DB      CR,LF
02B5    2A 2A 2A 20                     DB      '***  Interrupt hook test  ***'
02B9    20 49 6E 74
02BD    65 72 72 75
```

```
02C1  70 74 20 68
02C5  6F 6F 6B 20
02C9  74 65 73 74
02CD  20 20 2A 2A
02D1  2A
02D2  0D 0A                     DB      CR,LF
02D4  20 57 68 65               DB      ' When interrupt occur,',CR,LF
02D8  6E 20 69 6E
02DC  74 65 72 72
02E0  75 70 74 20
02E4  6F 63 63 75
02E8  72 2C 0D 0A
02EC  20 20 70 72               DB      '  print message "interrupt".',CR,LF
02F0  69 6E 74 20
02F4  6D 65 73 73
02F8  61 67 65 20
02FC  22 69 6E 74
0300  65 72 72 75
0304  70 74 22 2E
0308  0D 0A
030A  00                        DB      00H
                        ;
                        MSG5:
030B                    
030B  0D 0A                     DB      CR,LF
030D  0D 0A                     DB      CR,LF
030F  2A 2A 2A 20               DB      '***  TIMDAT test  ***'
0313  20 54 49 4D
0317  44 41 54 20
031B  74 65 73 74
031F  20 20 2A 2A
0323  2A
0324  0D 0A                     DB      CR,LF
0326  20 57 68 65               DB      ' When timdat routine call,',CR,LF
032A  6E 20 74 69
032E  6D 64 61 74
0332  20 72 6F 75
0336  74 69 6E 65
033A  20 63 61 6C
033E  6C 2C 0D 0A
0342  20 20 70 72               DB      '  print message "TIMDAT".',CR,LF
0346  69 6E 74 20
034A  6D 65 73 73
034E  61 67 65 20
0352  22 54 49 4D
0356  44 41 54 22
035A  2E 0D 0A
035D  20 20 20 31               DB      '   1 -- TIMDAT83',CR,LF
0361  20 2D 2D 20
0365  54 49 4D 44
0369  41 54 38 33
036D  0D 0A
036F  20 20 20 32               DB          2 -- TIMDAT85',CR,LF
0373  20 2D 2D 20
0377  54 49 4D 44
037B  41 54 38 35
037F  0D 0A
0381  20 20 20 33               DB      '   3 -- TIMDAT86',CR,LF
0385  20 2D 2D 20
0389  54 49 4D 44
038D  41 54 38 36
0391  0D 0A
0393  20 53 65 6C               DB      ' Select one of 1 to 3 -- '
0397  65 63 74 20
039B  6F 6E 65 20
039F  6F 66 20 31
03A3  20 74 6F 20
03A7  33 20 2D 2D
03AB  20
03AC  00                        DB      00H
                        ;
03AD                    START:
03AD  3A EF2D                   LD      A,(USERBIOS)
03B0  FE 08                     CP      08H             ; If User BIOS is smaller than 8 k,
03B2  DA 056C                   JP      C,SZERROR       ;  then error
                        ;
03B5  21 0448                   LD      HL,HOOK         ; Copy programm data for hook
03B8  ED 5B EF94                LD      DE,(TOPRAM)     ;  from hook to topram (userbios)
03BC  01 0124                   LD      BC,HOOKBTM-HOOK
03BF  F3                        DI                      ; Interrupt disable
03C0  ED B0                     LDIR
03C2  FB                        EI
                        ;
03C3  21 0147                   LD      HL,MSG0         ; Clear screen
03C6  CD 0581                   CALL    DSPMSG
                        ;
03C9                    PTOP1:
03C9  31 2000                   LD      SP,2000H        ; Set stack pointer
                        ;
03CC  0E 01                     LD      C,01H           ; Set resident flag
03CE  CD EB87                   CALL    RESIDENT        ;
                        ;
03D1  21 0149                   LD      HL,MSG1         ; Message (alarm, interrupt, timdat)
03D4  CD 0581                   CALL    DSPMSG
                        ;
03D7                    SELECT2:
03D7  CD 0575                   CALL    CONINF
03DA  CA 058D                   JP      Z,STOPEND       ; If stop, then end
03DD  FE 31                     CP      '1'
03DF  38 F6                     JR      C,SELECT2       ; If select error, then retry
03E1  28 08                     JR      Z,ALARM         ; Alarm select
03E3  FE 36                     CP      '6'
03E5  28 2A                     JR      Z,TIMDAT        ; TIMDAT select
03E7  30 EE                     JR      NC,SELECT2      ; If select error, then retry
03E9  18 18                     JR      INTERRUPT       ; else interrupt
                        ;
                        ;       *****   Alarm hook test *****
                        ;
```

```
03EB                            ALARM:
03EB    21 0228                         LD      HL,MSG3         ; Message (alarm hook 1 -- 5)
03EE    CD 0581                         CALL    DSPMSG
                                ;
03F1                            SELECT3:
03F1    CD 0575                         CALL    CONINF
03F4    CA 058D                         JP      Z,STOPEND       ; If stop, then end
03F7    FE 31                           CP      '1'
03F9    38 F6                           JR      C,SELECT3       ; If select error, then retry
03FB    FE 36                           CP      '6'
03FD    30 F2                           JR      NC,SELECT3      ; If select error, then retry

03FF    D6 31                           SUB     '1'             ; A reg. is 0..4
0401    18 26                           JR      SETHOOK         ; Goto hook setting process
                                ;
                                ;       *****   Interrupt hook test     *****
                                ;
0403                            INTERRUPT:
0403    F5                              PUSH    AF              ; Save selecting number
0404    21 02B1                         LD      HL,MSG4
0407    CD 0581                         CALL    DSPMSG          ; Interrupt hook select message
040A    F1                              POP     AF
040B    D6 32                           SUB     '2'
040D    C6 05                           ADD     A,05H           ; A reg. is 5..9
040F    18 1A                           JR      SETHOOK         ; Goto hook setting process
                                ;
                                ;       ***** TIMDAT hook test  *****
                                ;
0411                            TIMDAT:
0411    21 030B                         LD      HL,MSG5
0414    CD 0581                         CALL    DSPMSG          ; Select TIMDAT message
                                ;
0417                            SELECT4:
0417    CD 0575                         CALL    CONINF
041A    CA 058D                         JP      Z,STOPEND       ; If stop, then end
                                ;
041D    FE 31                           CP      '1'
041F    38 F6                           JR      C,SELECT4       ; If select error, then retry
0421    FE 34                           CP      '4'
0423    30 F2                           JR      NC,SELECT4      ; If select error, then retry
                                ;
0425    D6 31                           SUB     '1'
0427    C6 0A                           ADD     A,10            ; A reg, is 10..12
0429    18 00                           JR      SETHOOK         ; Goto hook setting process
                                ;
                                ;       *****   Set hook data for each select  ****
                                ;               A -- Hook logical number (0 -- 12)
042B                            SETHOOK:
042B    4F                              LD      C,A             ; Get hook data addr
042C    87                              ADD     A,A
042D    57                              LD      D,A
042E    81                              ADD     A,C
042F    4F                              LD      C,A
0430    06 00                           LD      B,00H
0432    21 FFC0                         LD      HL,HOOKTBL
0435    09                              ADD     HL,BC           ; HOOKTBL + A*3 --> HL
0436    E5                              PUSH    HL              ; Push target hook table
                                ;
0437    2A EF94                         LD      HL,(TOPRAM)
043A    87                              ADD     A,A
043B    4F                              LD      C,A
043C    09                              ADD     HL,BC           ; (TOPRAM) + A*6 --> HL
043D    EB                              EX      DE,HL
                                ;
043E    E1                              POP     HL              ; Set hook jump addr
043F    F3                              DI                      ; Interrupt disable
0440    23                              INC     HL
0441    73                              LD      (HL),E
0442    23                              INC     HL
0443    72                              LD      (HL),D
0444    FB                              EI
                                ;
0445    C3 03C9                         JP      PTOP1
                                ;
                                ;       *****   User BIOS data  *****
                                ;               This part is copyed into user bios area
0448                            HOOK:
0448    E5                              PUSH    HL
0449    21 C4A4                         LD      HL,0C400h+PDATA1-HOOK
044C    18 48                           JR      HOOKSTART
                                ;
044E    E5                              PUSH    HL
044F    21 C4AD                         LD      HL,0C400H+PDATA2-HOOK
0452    18 42                           JR      HOOKSTART
                                ;
0454    E5                              PUSH    HL
0455    21 C4B6                         LD      HL,0C400H+PDATA3-HOOK
0458    18 3C                           JR      HOOKSTART
                                ;
045A    E5                              PUSH    HL
045B    21 C4BF                         LD      HL,0C400H+PDATA4-HOOK
045E    18 36                           JR      HOOKSTART
                                ;
0460    E5                              PUSH    HL
0461    21 C4C8                         LD      HL,0C400H+PDATA5-HOOK
0464    18 30                           JR      HOOKSTART
                                ;
0466    E5                              PUSH    HL
0467    21 C4D1                         LD      HL,0C400H+PDATA6-HOOK
046A    18 2A                           JR      HOOKSTART
                                ;
046C    E5                              PUSH    HL
046D    21 C4DA                         LD      HL,0C400H+PDATA7-HOOK
0470    18 24                           JR      HOOKSTART
                                ;
0472    E5                              PUSH    HL
0473    21 C4E3                         LD      HL,0C400H+PDATA8-HOOK
```

```
0476    18 1E                   JR      HOOKSTART
                        ;
0478    E5                      PUSH    HL
0479    21 C4ED                 LD      HL,0C400H+PDATA9-HOOK
047C    18 18                   JR      HOOKSTART
                        ;
047E    E5                      PUSH    HL
047F    21 C4F7                 LD      HL,0C400H+PDATA10-HOOK
0482    18 12                   JR      HOOKSTART
                        ;
0484    E5                      PUSH    HL
0485    21 C501                 LD      HL,0C400H+PDATA11-HOOK
0488    18 0C                   JR      HOOKSTART
                        ;
048A    E5                      PUSH    HL
048B    21 C50C                 LD      HL,0C400H+PDATA12-HOOK
048E    18 06                   JR      HOOKSTART
                        ;
0490    E5                      PUSH    HL
0491    21 C517                 LD      HL,0C400H+PDATA13-HOOK
0494    18 00                   JR      HOOKSTART
                        ;
                        ;
0496                    HOOKSTART:
0496    ED 73 C522              LD      (0C400H+SAVESP-HOOK),SP
                                                ; Set stack pointer
049A    31 CAEF                 LD      SP,0CAEFH
                        ;
049D    E5                      PUSH    HL              ; Save all registers
049E    D5                      PUSH    DE
049F    C5                      PUSH    BC
04A0    F5                      PUSH    AF
                        ;
04A1    3A F53E                 LD      A,(RZIER)       ; Interrupt disable
04A4    F5                      PUSH    AF
04A5    AF                      XOR     A
04A6    D3 04                   OUT     (4),A
                        ;
04A8    0E FF                   LD      C,0FFH
04AA    CD FF9C                 CALL    SELBNK          ; Select OS bank
04AD    C5                      PUSH    BC              ; Save old bank information
                        ;
                                                        ; Print message
04AE    E5                      PUSH    HL
04AF    CD 003C                 CALL    XUSRSCRN        ; Change to user screen
04B2    E1                      POP     HL
                        ;
04B3    CD C482                 CALL    0C400h+XCONOUT-HOOK
                        ;
04B6    C1                      POP     BC
04B7    CD FF9C                 CALL    SELBNK          ; Recover old bank
                        ;
04BA    F1                      POP     AF              ; Recover interrupt register
04BB    32 F53E                 LD      (RZIER),A
04BE    D3 04                   OUT     (4),A
                        ;
04C0    F1                      POP     AF              ; Restore all registers
04C1    C1                      POP     BC
04C2    D1                      POP     DE
04C3    E1                      POP     HL
                        ;
04C4    ED 7B C522              LD      SP,(0C400H+SAVESP-HOOK)
                        ;                               ; Restore stack pointer
04C8    E1                      POP     HL
04C9    C9                      RET
                        ;
04CA                    XCONOUT:
04CA    1E 0C                   LD      E,0CH           ; Conout function
04CC    18 02                   JR      XPRINT0
                        ;
04CE                    XPRINT:
04CE    1E 0F                   LD      E,0FH           ; List function
                        ;
04D0                    XPRINT0:
04D0    4E                      LD      C,(HL)          ; DATA --> C
04D1    79                      LD      A,C
04D2    B7                      OR      A
04D3    C8                      RET     Z               ; Data end?
                        ;
04D4    CD C492                 CALL    0C400H+ROMBIOS-HOOK
                                                        ; Call ROM BIOS
04D7    23                      INC     HL              ; Pointer update
04D8    18 F6                   JR      XPRINT0
                        ;
                        ;       *****   Select target rom bios  *****
                        ;               E -- Function number
04DA                    ROMBIOS:
04DA    E5                      PUSH    HL              ; Save all registers
04DB    D5                      PUSH    DE
04DC    C5                      PUSH    BC
                        ;
04DD    2A 0007                 LD      HL,(BIOSJT+1)   ; Get ROM BIOS jump addr
04E0    16 00                   LD      D,00H
04E2    19                      ADD     HL,DE
04E3    11 C4A0                 LD      DE,0C400H+RETADDR-HOOK
04E6    D5                      PUSH    DE              ; Push return addr
04E7    E9                      JP      (HL)            ; Go !!
                        ;
04E8                    RETADDR:
04E8    C1                      POP     BC              ; Restore all registers
04E9    D1                      POP     DE
04EA    E1                      POP     HL
04EB    C9                      RET
                        ;
                        ;
04EC                    PDATA1:
04EC    41 4C 41 52             DB      'ALARM1',CR,LF,0
```

```
04F0    4D 31 0D 0A
04F4    00
04F5                            PDATA2:
04F5    41 4C 41 52                     DB      'ALARM2',CR,LF,0
04F9    4D 32 0D 0A
04FD    00
04FE                            PDATA3:
04FE    41 4C 41 52                     DB      'ALARM3',CR,LF,0
0502    4D 33 0D 0A
0506    00
0507                            PDATA4:
0507    41 4C 41 52                     DB      'ALARM4',CR,LF,0
050B    4D 34 0D 0A
050F    00
0510                            PDATA5:
0510    41 4C 41 52                     DB      'ALARM5',CR,LF,0
0514    4D 35 0D 0A
0518    00
0519                            PDATA6:
0519    54 4D 48 4F                     DB      'TMHOOK',CR,LF,0
051D    4F 4B 0D 0A
0521    00
0522                            PDATA7:
0522    48 4B 38 32                     DB      'HK8251',0ah,0dh,0
0526    35 31 0A 0D
052A    00
052B                            PDATA8:
052B    49 43 46 48                     DB      'ICFHOOK',0ah,0dh,0
052F    4F 4F 4B 0A
0533    0D 00
0535                            PDATA9:
0535    4F 56 46 48                     DB      'OVFHOOK',0ah,0dh,0
0539    4F 4F 4B 0A
053D    0D 00
053F                            PDATA10:
053F    45 58 54 48                     DB      'EXTHOOK',CR,LF,0
0543    4F 4F 4B 0D
0547    0A 00
0549                            PDATA11:
0549    54 49 4D 44                     DB      'TIMDAT83',CR,LF,0
054D    41 54 38 33
0551    0D 0A 00
0554                            PDATA12:
0554    54 49 4D 44                     DB      'TIMDAT85',CR,LF,0
0558    41 54 38 35
055C    0D 0A 00
055F                            PDATA13:
055F    54 49 4D 44                     DB      'TIMDAT86',CR,LF,0
0563    41 54 38 36
0567    0D 0A 00

056A                            ;
056A    0000                    SAVESP:
                                        DW      0000H
                                ;
056C                            HOOKBTM:
                                ;
                                ;
                                ;
                                ;
056C                            SZERROR:
056C    21 0103                         LD      HL,MSGX         ; Error message display
056F    CD 0581                         CALL    DSPMSG
0572    C3 058D                         JP      STOPEND
                                ;
                                ;       *****   Console input routine   *****
                                ;
0575                            CONINF:
0575    CD EB09                         CALL    CONIN           ; Console in
0578    FE 03                           CP      03H
057A    F5                              PUSH    AF              ; If stop, then z-flag on
057B    4F                              LD      C,A
057C    CD EB0C                         CALL    CONOUT          ; Display inputing char
057F    F1                              POP     AF
0580    C9                              RET
                                ;
                                ;       *****   Console output routine  *****
                                ;               IN :    HL -- Conout message top addr.
0581                            DSPMSG:
0581    4E                              LD      C,(HL)
0582    79                              LD      A,C
0583    B7                              OR      A               ; If data is 0,
0584    C8                              RET     Z               ;  then end of data.
                                ;
0585    E5                              PUSH    HL
0586    CD EB0C                         CALL    CONOUT          ; Console output
0589    E1                              POP     HL
058A    23                              INC     HL
058B    18 F4                           JR      DSPMSG
                                ;
                                ;       *****   Ending process  *****
                                ;
058D                            STOPEND
058D    0E 00                           LD      C,00H           ; Reset resident flag
058F    CD EB87                         CALL    RESIDENT
0592    C3 EB03                         JP      WBOOT           ;
                                ;
                                        END
```

4.4 Bank Switching

4.4.1 General

The PINE is provided with four banks of memory which are managed by the PINE CP/M operating system. Normal application programs can access the PINE memory without being aware of these memory banks. There are some advanced application programs which need to control the memory banks directly. For such application programs, this section describes the facilities to control the memory banks and shows how to use those facilities.

4.4.2 Bank Structure

4.4.2.1 Bank types

The PINE memory is divided into the following four banks:

(1) System bank: Accommodates the OS ROM and the higher portion of the RAM.
(2) Bank Ø: RAM.
(3) Bank 1: Application program ROM and portion of RAM.
(4) Bank 2: Application program ROM and portion of RAM.

The four memory banks provide 64K bytes of RAM and 96K bytes (maximum) of ROM.

Banks 1 and 2 are available in three configurations depending on the capacity of the ROM area (8K, 16K, and 32K bytes). PINE OS establishes the memory bank configuration at power-on time by reading the capacity of the ROM area on banks 1 and 2 from their header area. The ROM capacity for bank 1 is 32K bytes (for BASIC interpreter) as standard. This ROM area may be replaced by an application ROM. Figure 4.4.1 shows the PINE memory space.

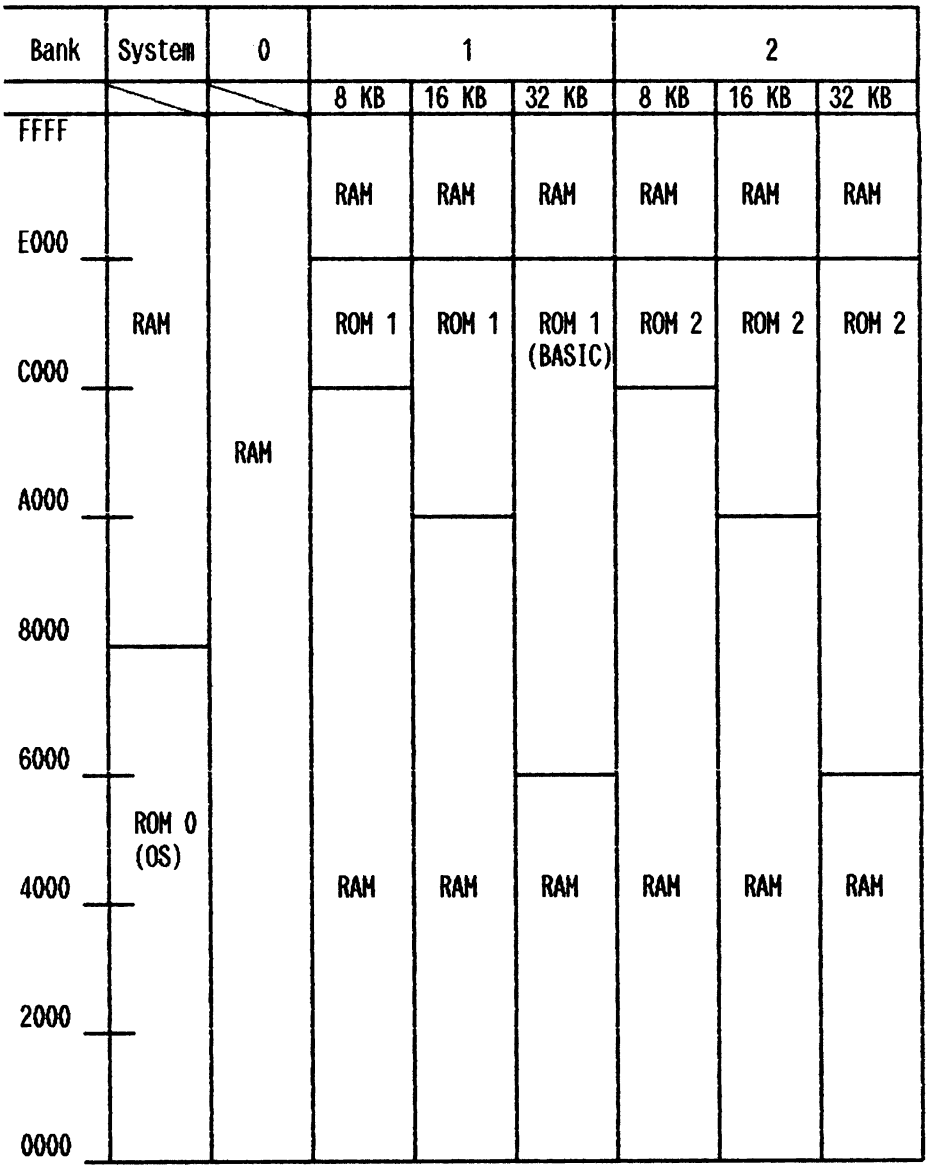
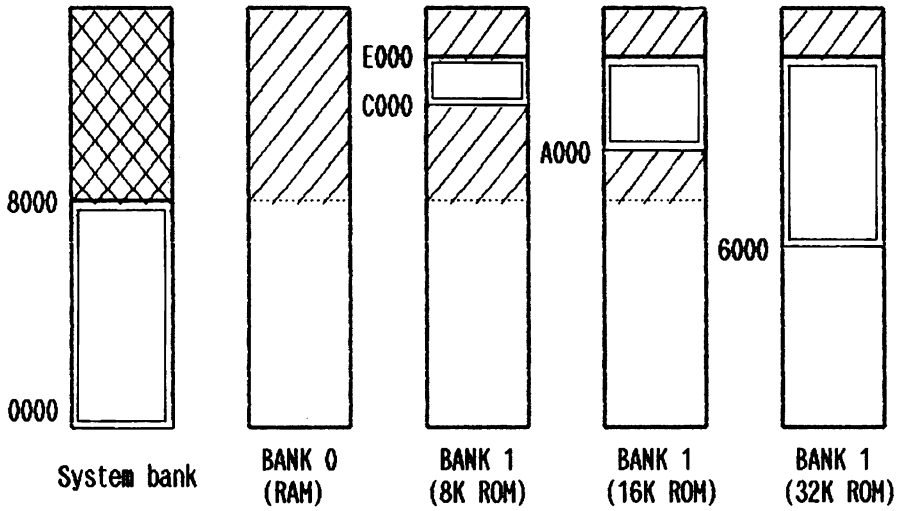


Fig. 4.4.1 PINE Memory Space

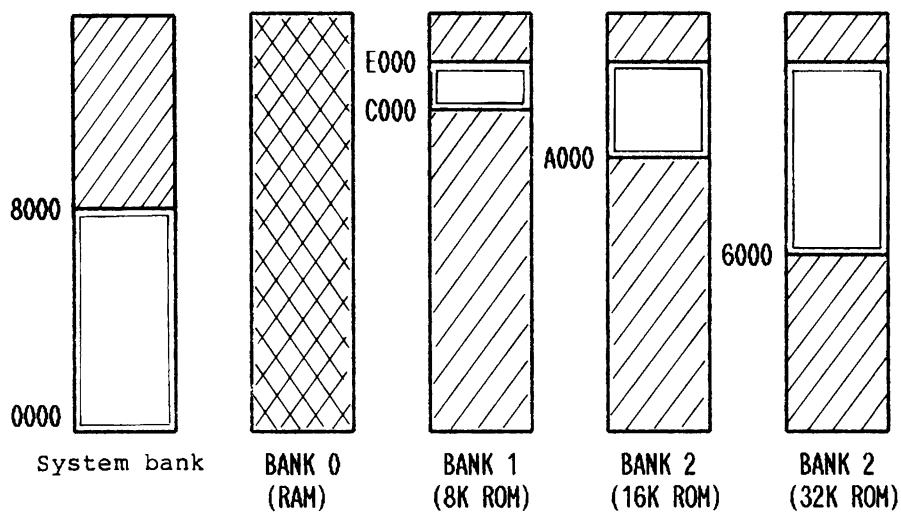
4.4.2.2 Common RAM areas

The following areas can be used as common area when banks are switched:
(1) Switching from system bank to bank Ø, 1, or 2



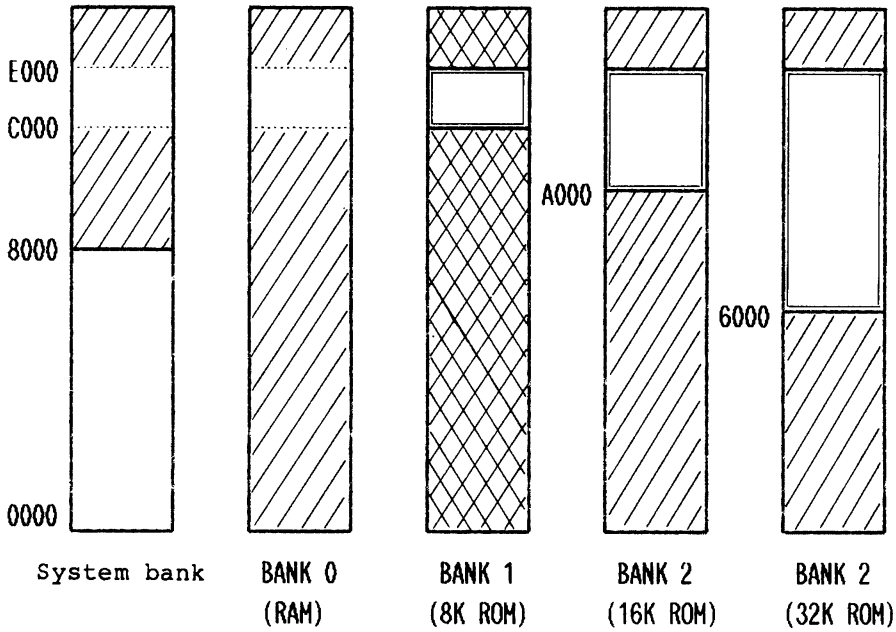
Hatched areas in the above figure are the common RAM areas which remain valid when banks are switched. Areas enclosed in double boxes are made up of ROM. Bank 2 has the same configuration as bank 1.

(2) Switching from bank Ø to system bank, bank 1 or 2



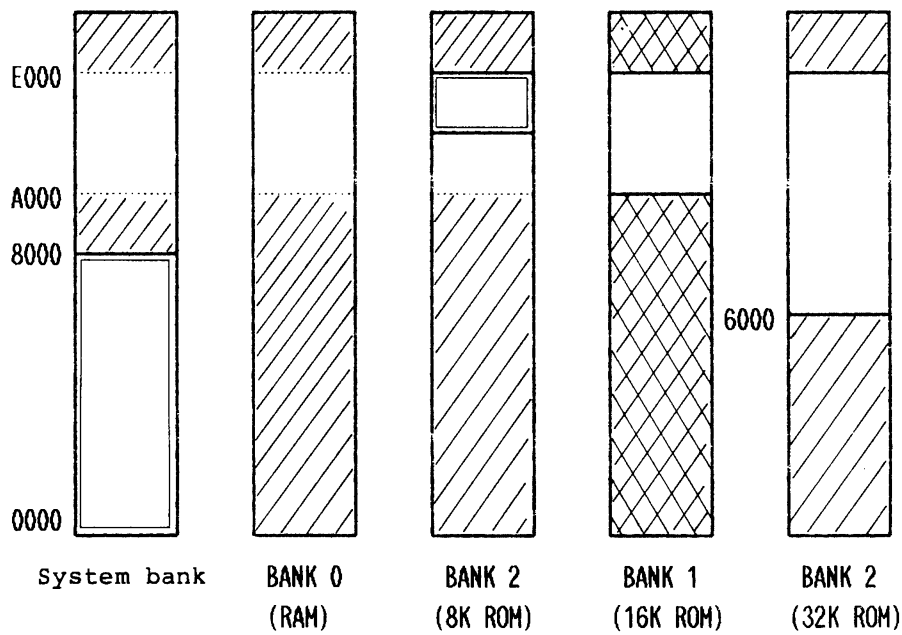
Hatched areas in the above figure are the common RAM areas which remain valid when banks are switched. Bank 2 has the same configuration as bank 1.

(3) Switching from bank 1 to system bank, bank Ø or 2 (8K ROM)



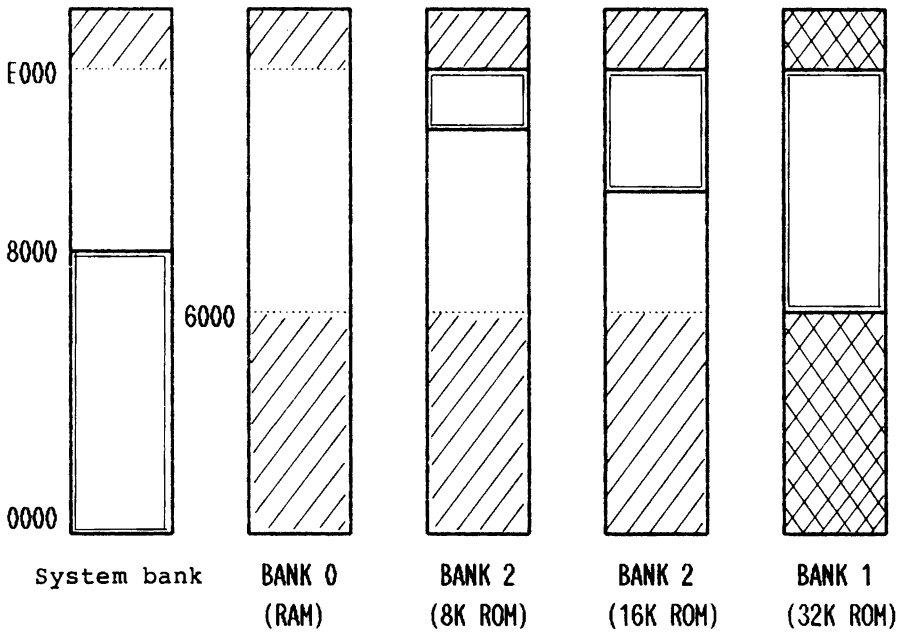
Hatched areas in the above figure are the common RAM areas which remain valid when banks are switched. On bank 2 (8K ROM), the area excluding the ROM area can be used as a common area.

(4) Switching from bank 1 to system bank, bank Ø or 2 (16K ROM)



Hatched areas in the above figure are the common RAM areas which remain valid when banks are switched. On bank 2 (16K ROM), the area excluding the ROM area can be used as a common area.

(5) Switching from bank 1 to system bank, bank Ø or 2 (32K ROM)



Hatched areas in the above figure are the common RAM areas which remain valid when banks are switched. On bank 2 (32K ROM), the area excluding the ROM area can be used as a common area.

4.4.2.3 Programming notes

(1) Banks 1 and 2 have different memory configurations depending on the capacity of the ROM devices in the ROM capsule. PINE OS establishes the memory bank configuration at power-on time by reading the capacity of the ROM area on banks 1 and 2. More specifically, the OS searches for the ROM header and, if one is found, read the ROM size from the header. If no header is found, the OS regards the bank as having no ROM. See the description of ROM capsules in Section 3.8, "Disk Storage" for the ROM header.

(2) The OS switches banks based on the bank information (BNKRGS, BNKRG1, BNKRG2, and BNKRG3) that are examined in paragraph (1).

4.4.3 Utility Routines

PINE OS supplies the following two types of utility routines for bank management:
1. Bank subroutines using BIOS functions
2. Bank subroutines which controls banks directly.

The application program must select one of these subroutine types depending on the purpose of its bank control.

4.4.3.1 Bank switching using BIOS

Bank switching using BIOS is used by ordinary application programs to exchange data between banks. PINE OS provides six BIOS functions for bank switching. Their functions are summarized in the table below.

The application program need not consider the stack area to be used during bank switching because the OS reserves it by itself. See Section 3.4, "BIOS Details" for details of the six BIOS functions.

Name	Address	Description
LOADX	WBOOT + 5AH or ØEB5DH	Reads one byte from the given address on the given bank.
STORX	WBOOT + 5DH or ØEB60H	Writes one byte into the given address on the given bank.
LDIRX	WBOOT + 60H or ØEB63H	Writes the given number of bytes at the given address on the given bank into the bank Ø area at the given address.
JUMPX	WBOOT + 63H or ØEB66H	Causes the CPU to jump to the given address on the given bank.
CALLX	WBOOT + 66H or ØEB69H	Calls the module at the given address on the given bank.
MEMORY	WBOOT + 4EH or ØEB51H	Gets the current bank information.

4.4.3.2 Bank switching using direct bank control

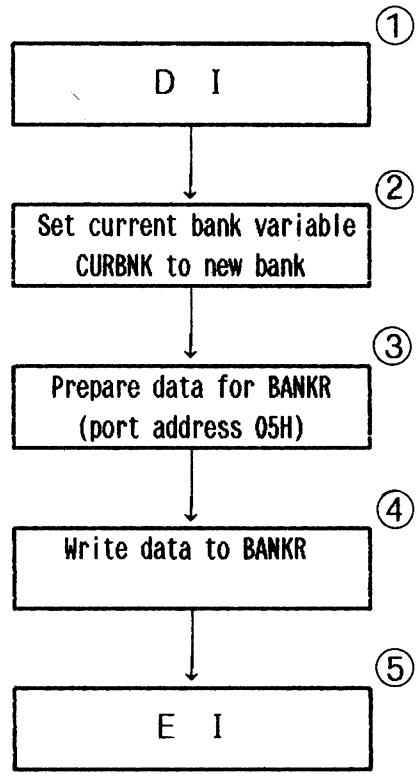
Bank switching using direct bank control is used by special application programs (extending BIOS or interrupt handling functions) which need to exercise direct control over banks. PINE OS provides six bank switching utility routines. Their functions are summarized in the table below.

When using these routines, the application program must reserve a stack area at a location in which it is accessible to the new bank. See Section 4.2, "Jump Tables" for details on the bank switching utility routines.

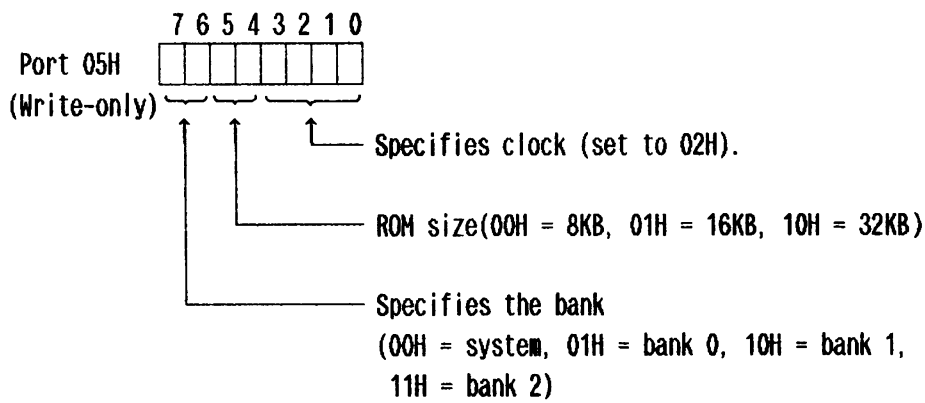
Name	Address	Description
SELBNK	ØFF9CH	Switches to the given bank.
LDAXX	ØFF9FH	Reads one byte from the given address on the given bank.
STAXX	ØFFA2H	Writes one byte into the given address on the given bank.
LDIRXX	ØFFA5H	Writes the given number of bytes at the given address on the given bank into the bank Ø area at the given address.
JUMPXX	ØFFA8H	Causes the CPU to jump to the given address on the given bank.
CALLXX	ØFFABH	Calls the module at the given address on the given bank.

4.4.3.3 Programming notes

The application program must use the following steps to switch banks using no OS routine:



BANKR (Bank Register) is a write-only register assigned to I/O port address Ø5H. Its format is shown below.



Step 1: Execute a DI instruction.
Disables interrupts during bank switching.
Step 2: Set current bank variable CURBNK to new bank.
Update the current bank information in CURBNK (ØF534H) with new bank information.
Step 3: Prepare data for BANKR.
Load system memory area RZBANKR (ØF53DH) for BANKR (PØ5H).
RZBANKR and BANKR contain the same data.
Step 4: Write data to BANKR.
Output new bank data into BANKR (PØ5H). The bank data must be identical to that set up in step 3.
Step 5: Execute an EI instruction.
Enables the CPU for interrupts.

4.4.4 Work Areas Associated with Bank Switching

OLDBNK (ØF52CH) 1 byte
Save area loaded with bank information on entry to BIOS.
= ØFFH: System bank
= ØØH: Bank Ø (RAM)
= Ø1H: Bank 1
= Ø2H: Bank 2

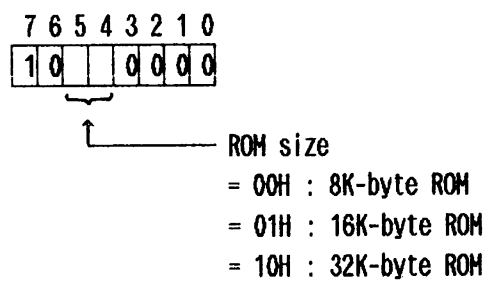
DISBNK (ØF52EH) 1 byte
Parameter area for JUMP or CALL bank switching routine.
= ØFFH: System bank
= ØØH: Bank Ø (RAM)
= Ø1H: Bank 1
= Ø2H: Bank 2

BNKRGS (ØF53ØH) 1 byte
System bank area. Set to ØØH.

BNKRG1 (ØF531H) 1 byte
Bank Ø area. Set to 4ØH.

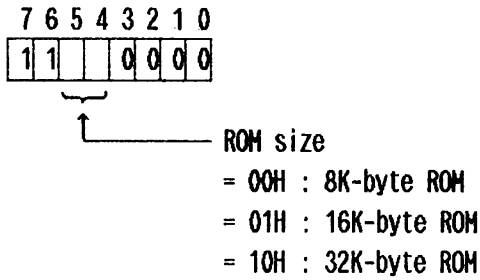
RZBANKR (ØF53DH) 1 byte
System area for holding output data to RANKR (PØ5H).
Has the same format as BANKR.

BNKRG2 (ØF532H) 1 byte
Bank 1 area



8K-byte ROM is assumed when no ROM is installed.

BNKRG3 (ØF533H) 1 byte
Bank 2 area



8K-byte ROM is assumed when no ROM is installed.

CVRBNK (ØF534H) 1 byte
Contains the bank number of the current bank.
= ØFFH: System bank
= ØØH: Bank Ø (RAM)
= Ø1H: Bank 1
= Ø2H: Bank 2

4.5 Resident Processing

4.5.1 General

When resident processing is specified by an application program, the current program in RAM is preserved and when a power-on condition is generated in the restart mode (by the wake function or other factor), control is transferred immediately to address 1ØØH on bank Ø.

This function allows the PINE to warm-start an application program at a power-on time without passing through the menu processing or CCP, making it a turn-key system.

When this function is activated, the auto start or wake string is passed to the application program through the keyboard buffer. The application program can treat the string as if it were entered from the keyboard at warm-start time.

4.5.2 How to Specify and Cancel Resident

4.5.2.1 Specifying resident

Resident must be specified within the application program. The method of specifying resident processing differs between ROM-based and load-and-go programs.

Resident can be specified by setting the resident flag RESEXQ (ØEF28H) in the system area in one of the following ways:
1. Calling the BIOS RESIDENT function (WBOOT + 84H).
2. Writing Ø1H directly into RESEXQ (ØEF28H).

(1) Actions required of a load-and-go program for resident processing
A load-and-go program, when started, is loaded into RAM (bank Ø) at address 1ØØH, after which control is transferred to address 1ØØH. To specify resident processing, the load-and-go program must take the following steps:

1. Set the resident flag.
2. If necessary, rewrite the instruction at address 1ØØH to a jump to the point of warm start.

Subsequently, when power-on, reset, or WBOOT processing is initiated, the system transfers control directly to address 1ØØH in RAM.

(2) Actions required of a ROM-based program for resident processing
When a ROM-based program is started, the first sector of the program is loaded into RAM (bank Ø) at address 1ØØH, after which control is transferred to the specified program start address. The ROM-based program which is to specify resident processing must load necessary data in advance into the first 9Ø bytes of the ROM program in the specified format (see Section 4.61, "Executing a ROM Program" for the format of the 9Ø-byte data). The ROM-based program need only set the resident flag.

Subsequently, when power-on, reset, or WBOOT processing is initiated, the system transfers control directly to address 1ØØH in RAM. The 9Ø bytes from address 1ØØH contains the data that was read when the ROM-based program was started for the first time. The data consists of routines for checking for a program for which resident is specified, for transferring control to the given address, and for handling errors.

4.5.2.2 Canceling resident processing

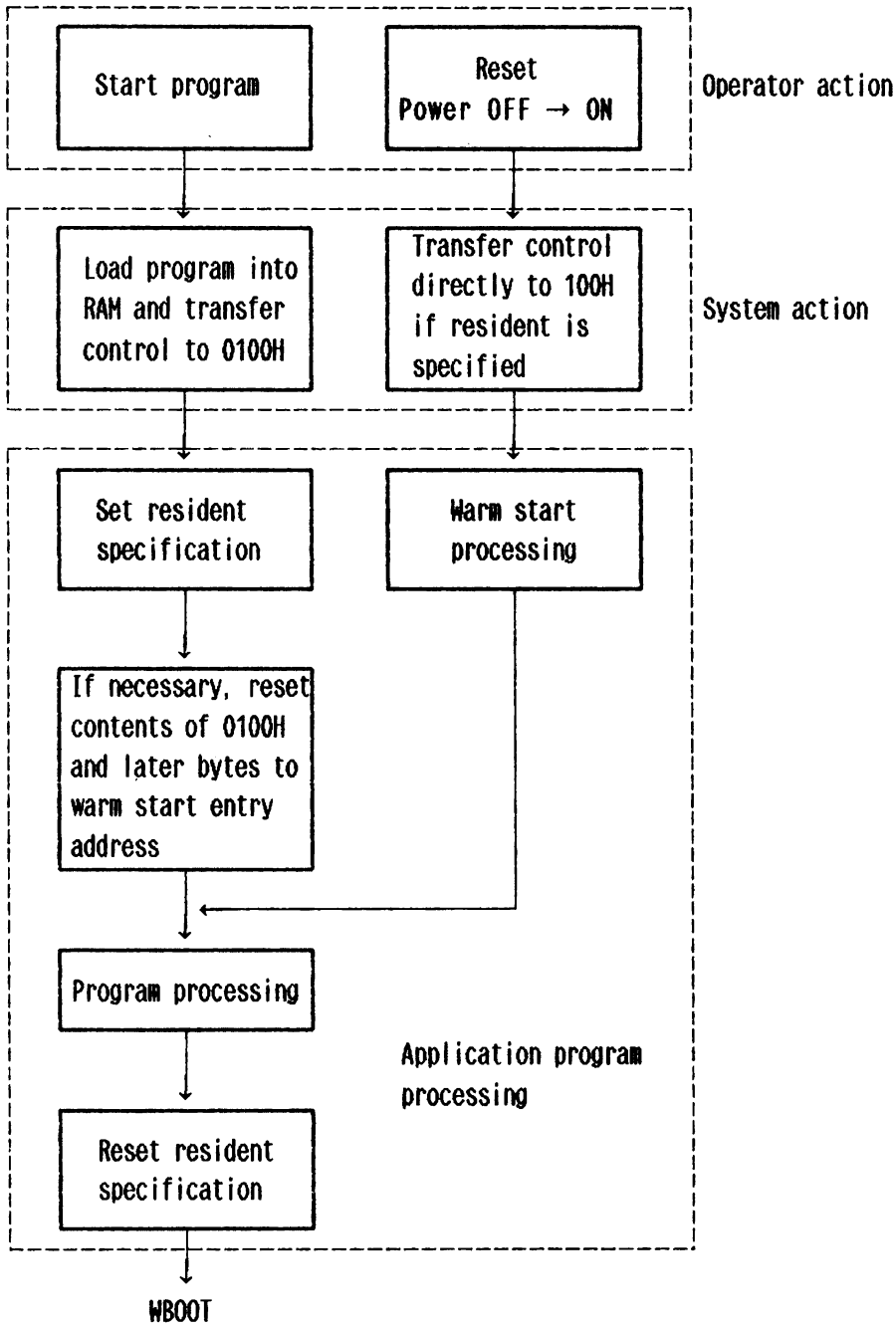
Resident processing must be canceled by the application program. Resident can be canceled by:

1. Calling the BIOS RESIDENT function (WBOOT + 84H).
2. Writing ØØH directly into RESEXQ (ØEF28H).

The resident flag RESEXQ is not affected by resets; it is cleared to zero only by System Initialize. This means that control cannot exit the application program if the resident flag remains set.

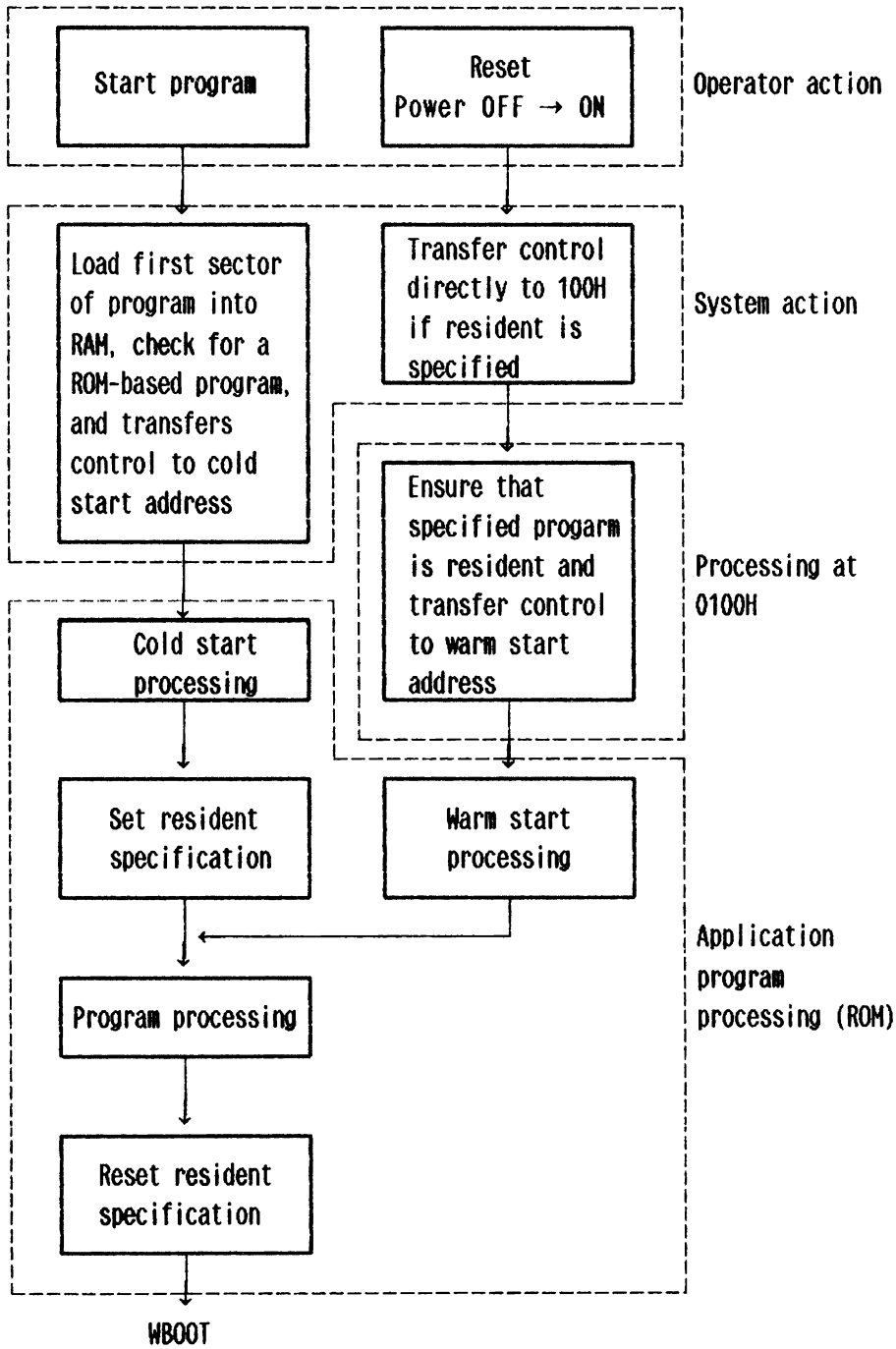
4.5.3 Processing Flows

The flowcharts below show the flows of processing of the system and application programs for which resident is specified.
(1) Load-and-go program



(2) ROM-based program

ROM-based programs to be executed with resident specified must begin with a 90-byte data area in the specified format. See Section 4.6, "Executing a ROM Program" for the 90-byte data format.



4.5.4 Miscellaneous Considerations

4.5.4.1 Programming notes

(1) Before terminating (calling WBOOT) a program for which resident is specified, cancel the resident specification. Otherwise, the program can be terminated by no means but system initialize.
(2) When specifying resident processing for a ROM-based program, be sure to define predetermined data in the first 90-byte are of the program in advance. Otherwise, warm start processing will fail.

4.5.4.2 Related Work Area

RESEXQ (0EF28H) 1 byte
Resident flag.
= 00H: Resident not specified.
= Nonzero: Resident specified.
The initial value is 00H. This area is initialized only by system initialize.

4.6 Executing a ROM-based Program

4.6.1 General

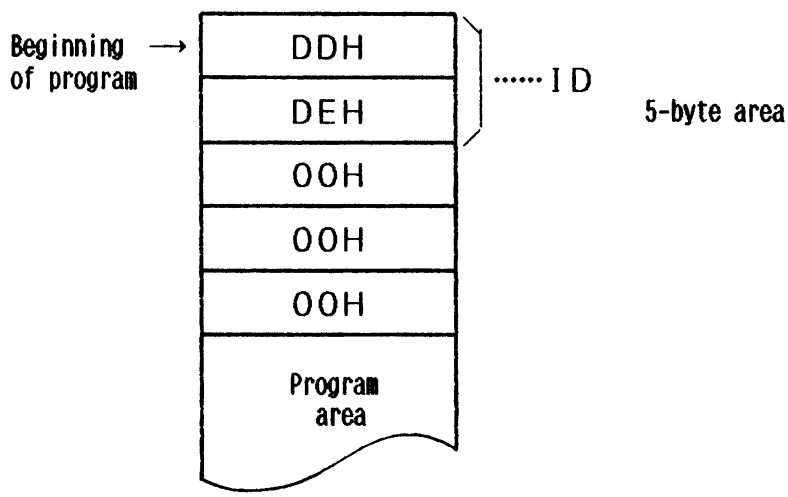
Although the PINE can load application programs from ROM capsules into RAM memory at address 100H for execution as under standard CP/M, it can also execute the programs as stored in the ROM to increase memory efficiency and to save electric power. This section explains the principle of ROM-based program execution, method of establishing the execution environment, and programming notes.

4.6.2 Establishing the Execution Environment

ROM-based programs, which are executed immediately in a ROM capsule, must be given an identification at their beginning to distinguish themselves from ROM programs, called load-and-go ROM programs, which are loaded into RAM memory for execution.

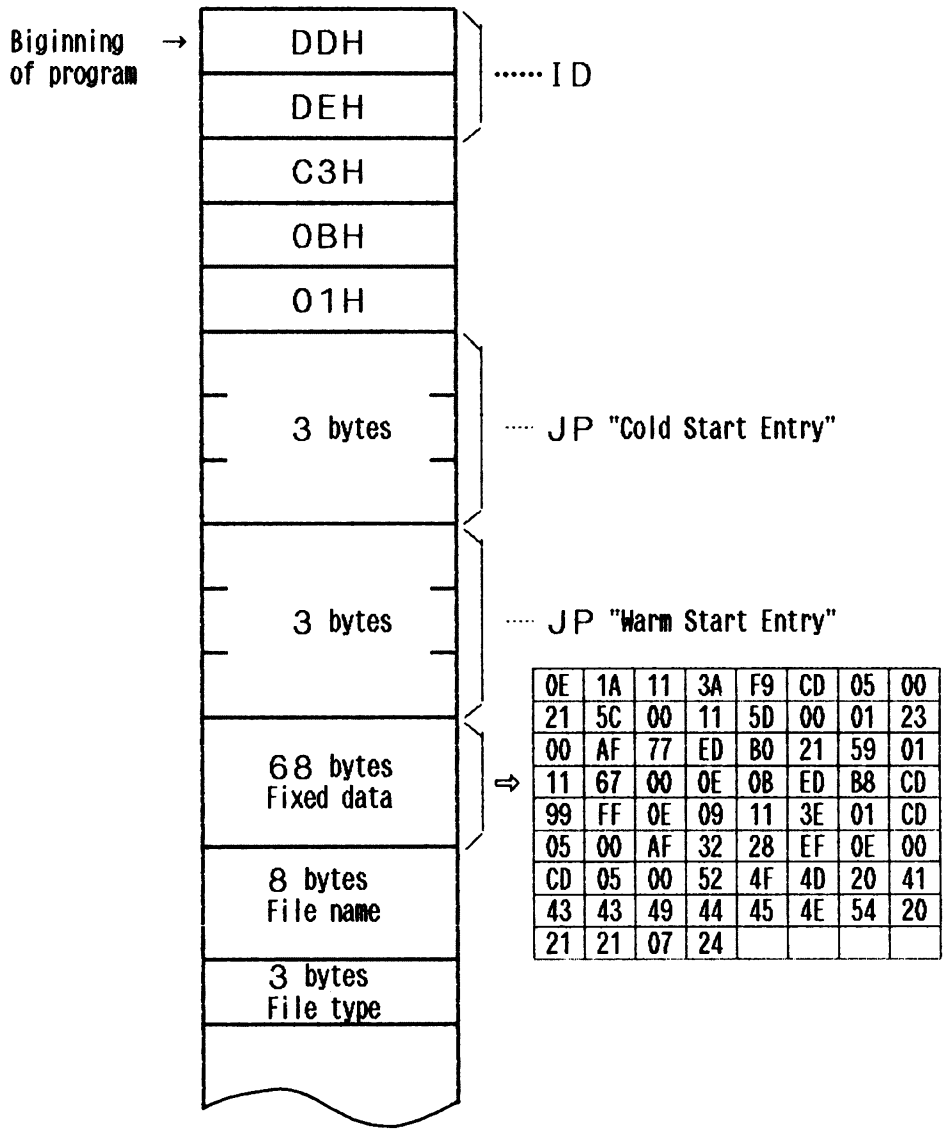
4.6.2.1 When resident is not specified

A ROM-based program without resident specification must begin with a 5-byte ID data as shown in the figure below. The system identifies a ROM-based program by examining this field.



4.6.2.2 When resident is specified

A ROM-based program with resident specified must begin with 90 bytes of information in the format shown below.



0E	1A	11	3A	F9	CD	05	00
21	5C	00	11	5D	00	01	23
00	AF	77	ED	B0	21	59	01
11	67	00	0E	0B	ED	B8	CD
99	FF	0E	09	11	3E	01	CD
05	00	AF	32	28	EF	0E	00
CD	05	00	52	4F	4D	20	41
43	43	49	44	45	4E	54	20
21	21	07	24				

Offset	Bytes	Name	Description
00H-01H	2	ID	ROM-based program identifier
02H-04H	3	Jump instruction for resident processing	Jump instruction for resident processing (NOP when resident is not specified)
05H-07H	3	Cold Start Entry	Program cold start address (Jump to ROM cold start address)
08H-0AH	3	Warm Start Entry	Program hot start address (Jump to ROM hot start address)
0BH-4EH	68	Resident processing	Instructions for resident processing. Resident processing consists of: 1. Setting up DMA and FCB. 2. Executing RSROMEXQ. 3. Abnormal termination processing.
4FH-56H	8	File name	ROM-based program name (must be the same as that in the ROM directory)
57H-59H	3	FILE TYPE	ROM-based program type (must be the same as that in the ROM directory)

4.6.3 Processing Flow

Figure 4.6.1 shows the flow of processing up to the point where a ROM-based program receives control.

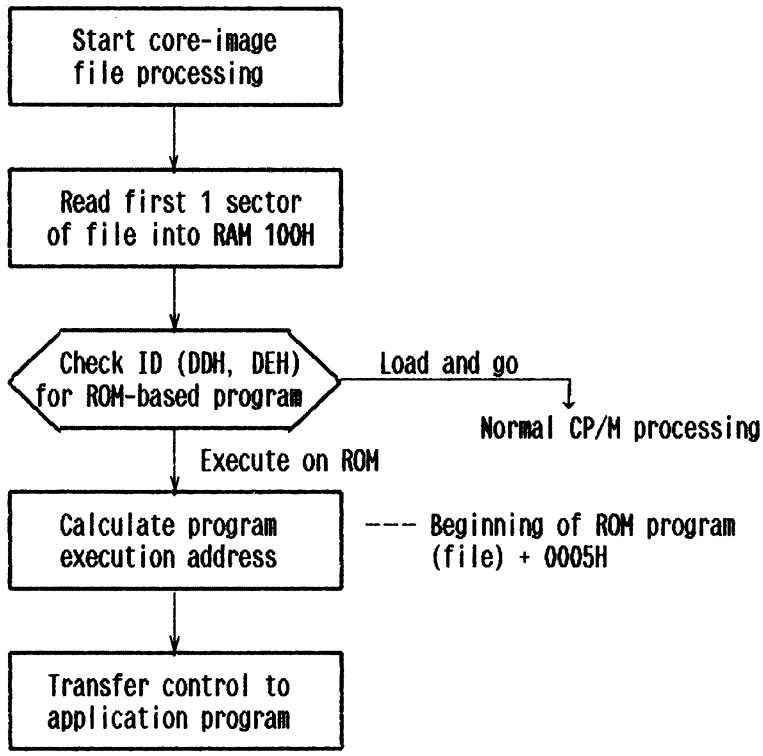


Fig. 4.6.1 Starting a ROM-based Program

4.6.4 Use and Programming Notes

This subsection describes the use and programming notes to be taken when programming ROM-based programs.

4.6.4.1 Program starting address

Load-and-go ROM programs are loaded into RAM memory at location 100H for execution. Such programs must be allocated RAM memory starting at address 100H. ROM-based programs can execute directly in a ROM capsule. The starting address of such programs must be determined by taking into account the header and directory areas.

The program starting address differs depending on the size of the ROM-based program and the type of the PROM (27256, 27128, or 2764) in which it is to be stored.

A ROM-capsule must contain header and directory areas. Since the size of the directory area of a program file varies with the number and size of the programs that the file contains (though its header area is fixed at 32 bytes), it is necessary to calculate the size of the directory area to be reserved before actually loading the programs.

Figure 4.6.2 shows the relationship between the ROMs and PINE addresses.

Note that the physical addresses of 27128 and 27256 ROMs do not match the PINE logical addresses.

When programming a ROM program, use the optional PROM WRITEr cartridge or PROMFORM utility program.

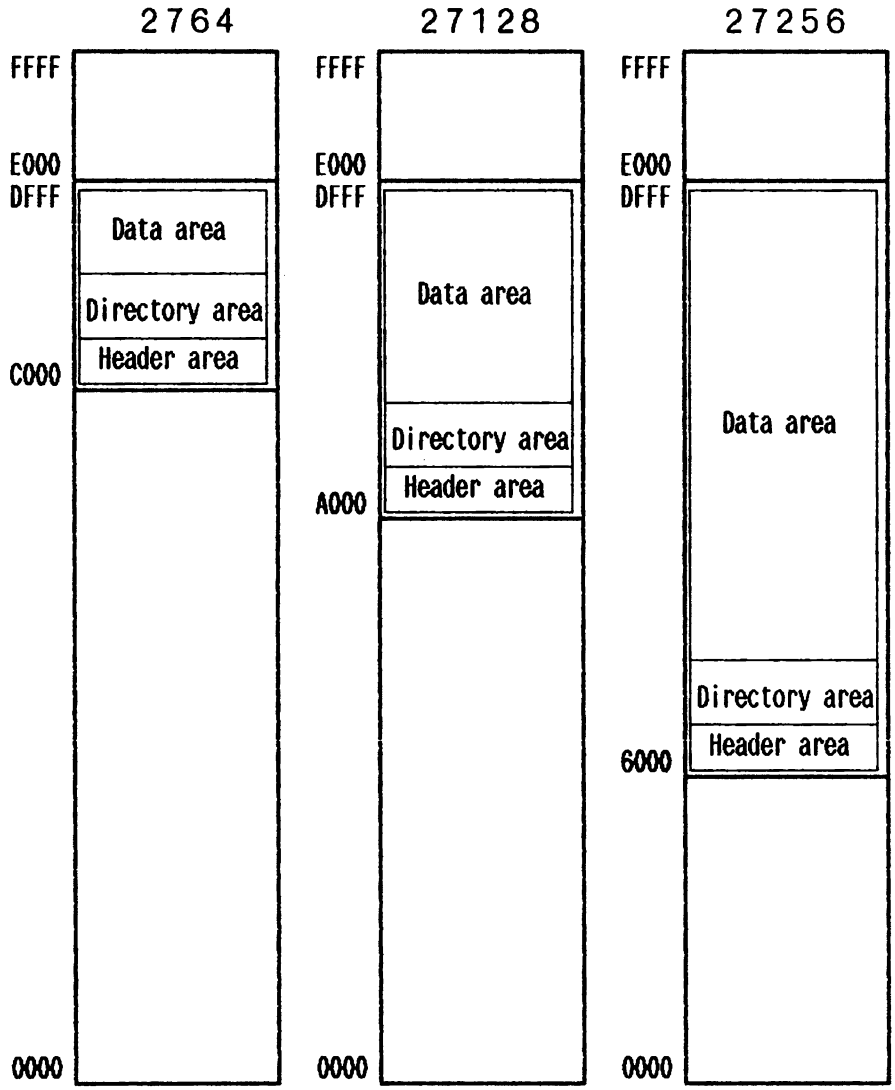


Fig. 4.6.2 Relationship between Logical and ROM Addresses
(ROM area is enclosed in double boxes)

(1) When implementing a single ROM-based program in a ROM
The program starting address of a ROM-based program consisting of a single program is listed below.

ROM	2764	27128	27256
Starting address	0C080H	0A080H	6080H

A ROM-based program of this type must begin with a 32-byte header and a 32 bytes x 3 directory area. Accordingly, the program starting address must be 80H bytes away from the beginning of the ROM. See Section 7.3, "Guide for Programming ROMs" for the structure of the header and directory areas.

(2) When implementing more than one ROM-based program in a ROM
When implementing more than one ROM-based program in a ROM, it is necessary to calculate in advance the size of the directory area to be reserved since the size of a portion of the directory area varies with the number of programs and ROM capacity.

One directory entry can handle a file of up to 16K bytes. Two or more entries are used to handle files larger than 16K bytes. For example, two directory entries are required for a ROM-based program larger than 16K bytes and one directory entry for one which is smaller than 16K bytes.

Since the directory area (including the header area) is reserved in 128-byte increments, the size of space necessary for the header and directory areas is obtained from the formula:

$$[\{((n1/16384 + 1) + (n2/16384 + 1) + \ldots)/4\} \times 4 + 1] \times 32 \text{ b}$$

Number of directory entries

Total number of directory entries

For 128-byte boundary alignment

(16K = 16,384)

where n1, n2, ... denote program sizes.

The program starting address is obtained by adding the above result to the starting address of the ROM.

To find the starting address of the second and subsequent programs, follow the steps given below.

Since CP/M manages files in 1K-byte units, the starting address of the second or subsequent program is obtained by adding the starting address of the preceding program plus

$$[\{(n-1)/1024\} + 1] \times 1024 \quad \text{(bytes)}$$

where n is the size of the preceding program.

Figure 4.6.3 shows the memory map for the following ROM programs implemented in a 27256 ROM:

File name	Records	File size	Directory entries	Starting address
FILE1.COM	30	4K bytes	1	6100H
FILE2.COM	150	19K bytes	2	7100H
FILE3.COM	50	7K bytes	1	BD00H

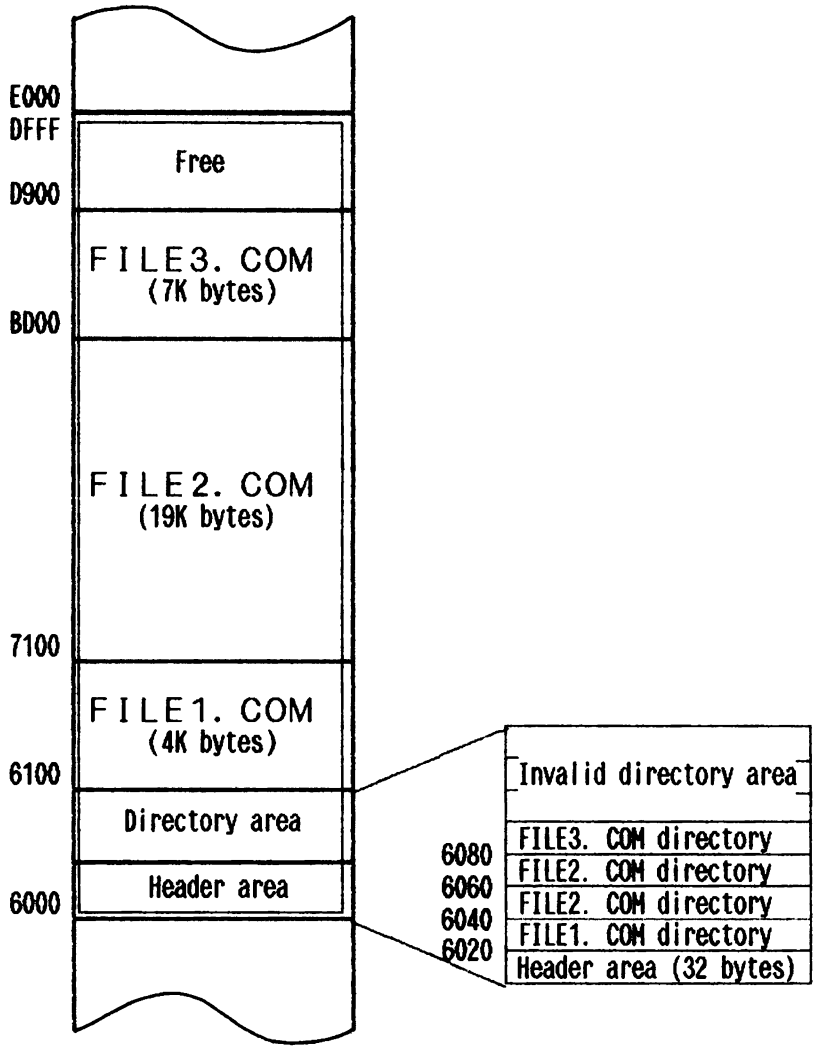


Fig. 4.6.3 Sample ROM Memory Map (for PINE)

(2) ROM-based Program Work Areas

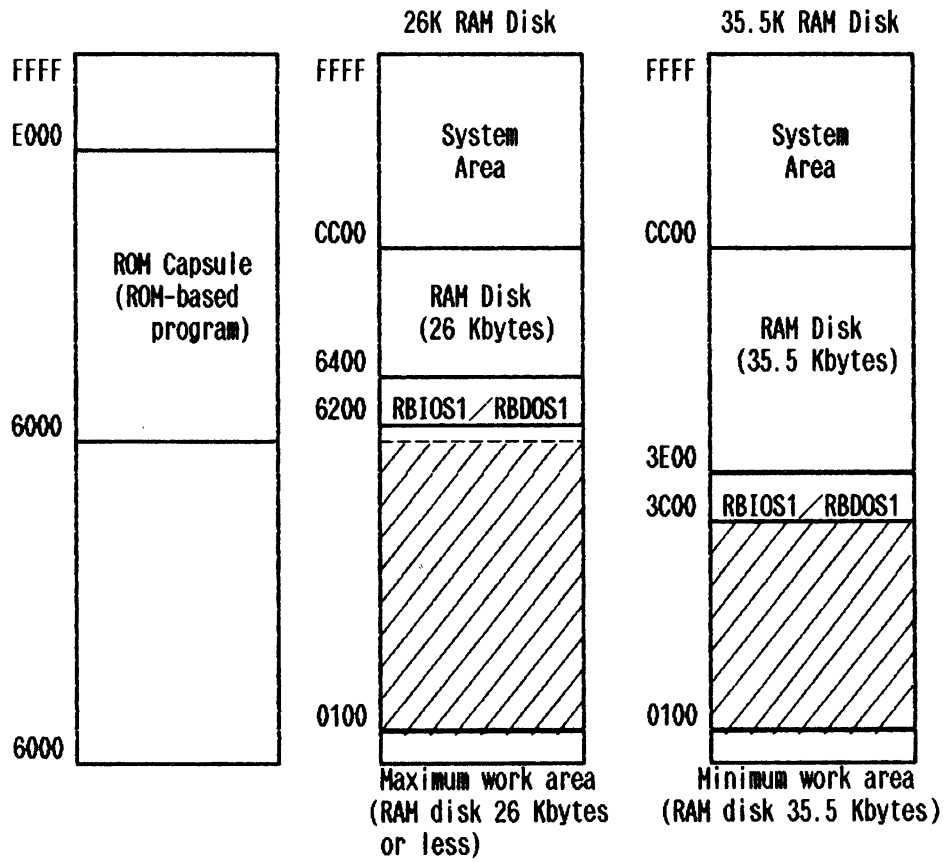
ROM-based programs need a work area which is different from that used by load-and-go programs. Whereas load-and-go programs can use the area following their program area as a work area, ROM-based programs must use an area higher than 100H (15AH or higher when Resident is specified). The upper limit of the work area for ROM-based programs is obtained as follows:

min{(ROM Starting Address), (RBDOS1 Starting Address)}

The ROM starting address is:
- 6000H for 27256
- 0A000H for 27128
- 0C000H for 2764

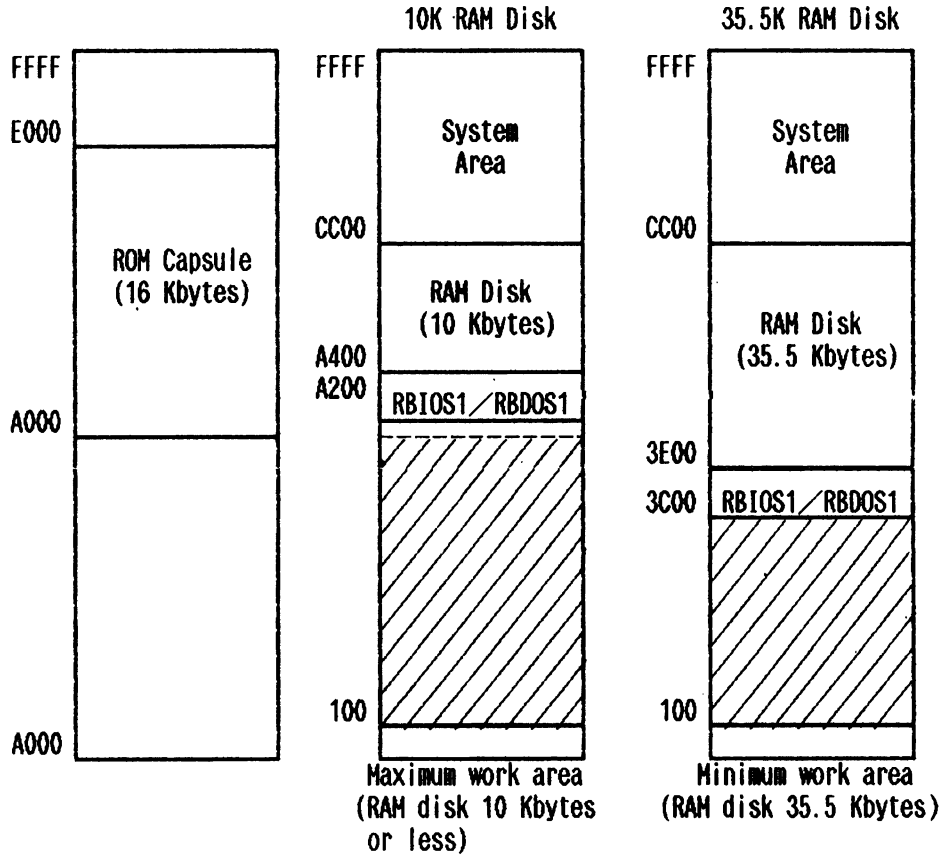
The RBDOS1 staring address is stored in addresses 0006H and 0007H.

a) When 27256 ROM is used



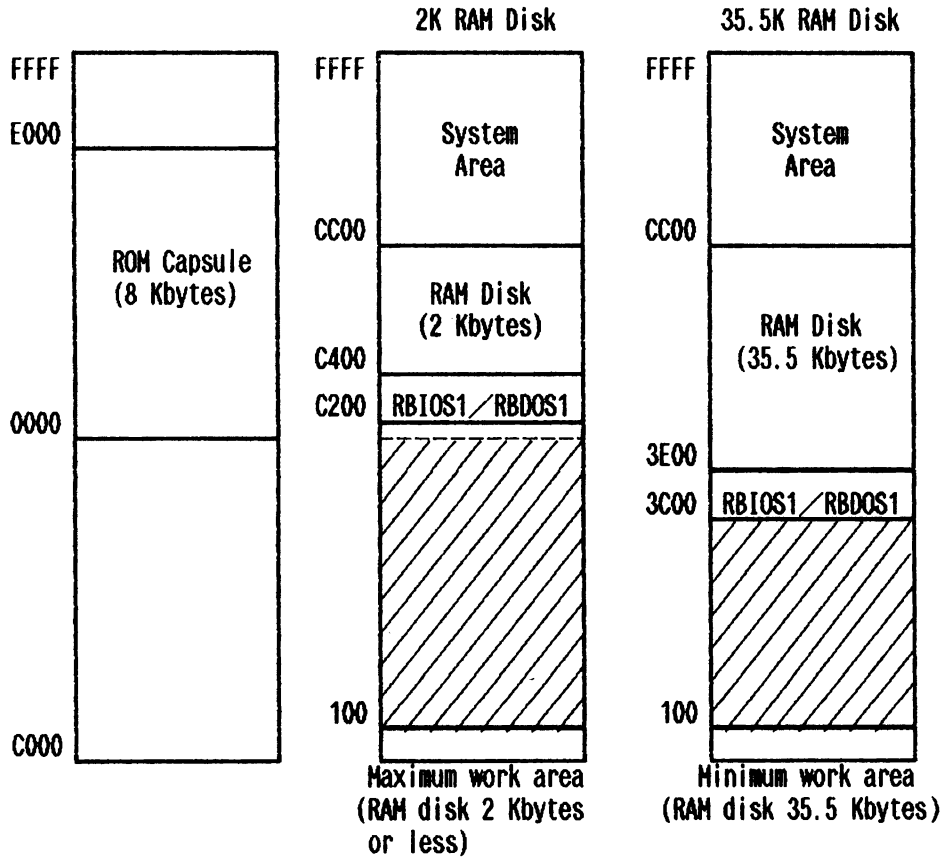
The hatched area is available as the work area without bank switching. The area starting at 6000H is made available through bank switching.

b) When 27128 ROM is used



The hatched area is available as the work area without bank switching. The area starting at address 0A000H is made available through bank switching.

c) When 2764 ROM is used



The hatched area is available as the work area without bank switching. The area starting at address 0C000H is made available through bank switching.

(3) BDOS/BIOS Calls from ROM-based Programs
Load-and-go programs call address 0005H through 0007H when making a BDOS call and address 0000H through 0002H when making a BIOS call. For ROM-based programs, however, the addresses of BDOS and BIOS may be located in the background memory of the ROM capsule (because their location varies with the capacity of the installed RAM disk). When calling BDOS and BIOS, therefore, ROM-based programs must call BDOS and BIOS entry points (RBDOS2 and RBIOS2) in the resident area.

```
RBDOS2 entry address:     0FF90H    (JP RBDOS2)

RBIOS2 BOOT address:      0EB00H
RBIOS2 WBOOT address:     0EB03H
RBIOS2 CONST address:     0EB06H

          |                  |

RBIOS2 CONTINUE address: 0EB8AH
```

(4) ROM-based program ROMs
The PINE supports two formats, P- and M-formats, for ROM capsules. ROM-based programs must be programmed in the P-format. Load-and-go ROM programs may be programmed either in the P- or M-format. See Section 7.3, "Guide for Programming ROMs" for instructions for programming ROMs.

4.7 Interrupts

4.7.1 General

PINE supports five types of interrupts. When an interrupt occurs, it takes in and sets up the corresponding interrupt vector and transfers control to the interrupt processing routine.

The interrupts that the PINE supports include:
1. 7508 (4-bit CPU)
2. ART (RxRDY)
3. Alarm time
4. ICF (Input Capture)
5. OVF (FRC Overflow)
6. EXT (External)

The sources of 7508 interrupts are:
1. Keyboard
2. Power switch
3. Alarm time
4. Power fail
5. 1-second interrupt

4.7.2 Interrupt Vector

The PINE handles five types of interrupts. Their interrupt vector table is located in the area between 0FFF0H and 0FFFFH.

Interrupts are given priorities; the PINE accepts interrupts of the highest priority first.

Table 7.4.1 Interrupt Vector Table

Priority	Interrupt source	Vector	RAM address
1 (highest)	7508 (4-bit CPU)	0F0H	0FFF0H and 0FFF1H
2	ART (RxRDY)	0F2H	0FFF2H and 0FFF3H
3	ICF (Input Capture)	0F4H	0FFF4H and 0FFF5H
4	OVF (FRC Overflow)	0F6H	0FFF6H and 0FFF7H
5	EXT (External interrupt)	0F8H	0FFF8H and 0FFF9H

The interrupt sources and their reset conditions are listed in Table 4.7.2.

Table 4.7.2 Interrupt Sources and Reset Conditions

Interrupt	Interrupt source	Reset conditions
75Ø8	* Keyboard input * Lapse of 1 second * Alarm time * Power switch on or off * Power fail voltage	Returns a reply to the 75Ø8.
ART (RxRDY)	* ART RxRDY set.	Reads the receive data register ARTDIR (P14H).
ICF	* Change in state of the input signal from barcode reader or cassette drive	Reads the input capture register (PØ3H).
OVF	* FRC (Free Running Counter) overflows. FRC is a 16-bit counter running at a period of 1Ø6.7 ms.	Issues a Reset OVF command (sets CMDR (PØ2H) bit 2 to 1).
EXT	* Interrupt signal from external device (via system bus)	Returns a reply to the external device (via system bus).

4.7.3 Interrupt Control

The PINE takes various actions to control interrupts.

4.7.3.1 Setting up the interrupt mode and vector data

When an address ØØØØH start occurs (started by a system initialize, reset, or power-on), the PINE specifies the mode 2 interrupt and loads the I register with ØFFH (the higher order 2 digits of the interrupt vector table address).

4.7.3.2 Loading the interrupt vector table

The PINE loads interrupt data from OS ROM into addresses ØFFFØH through ØFFFFH when a reset or system initialize occurs.

4.7.3.3 Interrupt control (disabling and enabling)

(1) Interrupt control by the system
The PINE system disables and enables interrupts as follows:

Table 4.7.3 Interrupt Control

Interrupt / Processing	7508	7508			ART	OVF	ICF	EXT	Remarks
		key	1sec	Alarm					
System Initialize	○	○	○	×	×	○	×	×	This state continues.
Reset	○	○	○	—	×	○	×	×	
Restart Pw ON	○	—	—	—	×	○	×	×	
Continue Pw ON	—	—	—	—	—	—	—	—	
7508 interrupt processing	×	—	—	—	—	—	—	—	Enabling the CPU for interrupt of 7508 during alarm screen display
ART interrupt processing	×	—	—	—	×	×	×	×	Continues processing in DI state
OVF interrupt processing	×	—	—	—	×	×	×	×	
ICF interrupt processing	×	—	—	—	×	×	×	×	
EXT interrupt processing	×	—	—	—	×	×	×	×	
MCT processing	×	—	—	—	—	×	—	—	
FDD processing	×	—	—	—	×	×	×	×	Continues processing in DI state.
BEEP processing	—	△*	—	—	—*	×	—*	—*	*

* indicates that the interrupt disable/enable state can be changed during processing.
○··· Enabled ×··· Disabled —··· No change
△··· Disables all interrupts except STOP key interrupts.

(2) Disabling or enabling interrupts
The PINE CPU can be disabled or enabled for interrupts by:
1. Using the BIOS MASKI function.
2. Rewriting the Interrupt Enable Register (IER) (PØ4H) directly.

See Section 3.4, "BIOS Details" for the use of the BIOS MASKI function.

The figure below shows the steps for rewriting the IER directly.

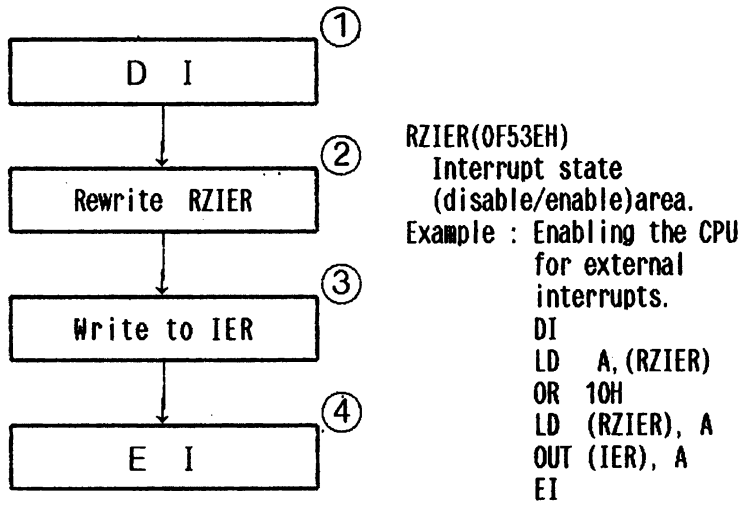
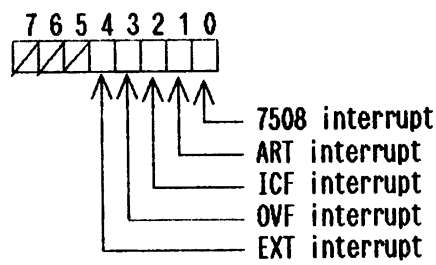


Fig. 4.7.4 Rewriting IER

Related work area:
RZIER (ØF53EH) 1 byte
Contains the interrupt disable/enable status.



1 refers to enabling and a Ø to disabling interrupts.
The format of the RZIER is identical to that of the IER.

4.7.3.4 Interrupt processing time

Table 4.7.5 lists the times required for the PINE to process interrupts.

Table 4.7.5 Interrupt Processing Times

Interrupt	Number of machine states	Time	Remarks
7508 (key in)	13789	3747μs	
7508 (1 sec)	14141	3842μs	
7508 (Alarm)	14087	3828μs	Processing time when alarm display processing is disabled
7508 (Pw sw on)	13837	3760μs	
7508 (Pw sw off)	175852	47786μs	Processing time when power-off processing is disabled
7508 (Power fail)	13891	3775μs	Processing time when power-off processing is disabled
ART	1623	441μs	
OVF	1122 (4504)	305μs (1224μs)	Number in parentheses is the processing time during reverse video cursor processing
ICF	686	186μs	
EXT	698	190μs	

Note: The reason that the interrupt processing times during power-off processing are longer than usual is that the PINE waits for approximately 40 ms after issuing a keyboard clear command to the 7508.

4.7.3.5 Interrupt processing

(1) Location where interrupt processing is performed
It is unpredictable on which bank the PINE is running when an interrupt occurs. To keep track of the location of PINE execution when an interrupt occurred, the PINE places the entry portion of the interrupt processing in the resident area (ØE000H through ØFFFFH). When an interrupt occurs, the banks are switched in the entry portion and the actual interrupt processing is carried out in the main interrupt processing routine on the OS ROM.

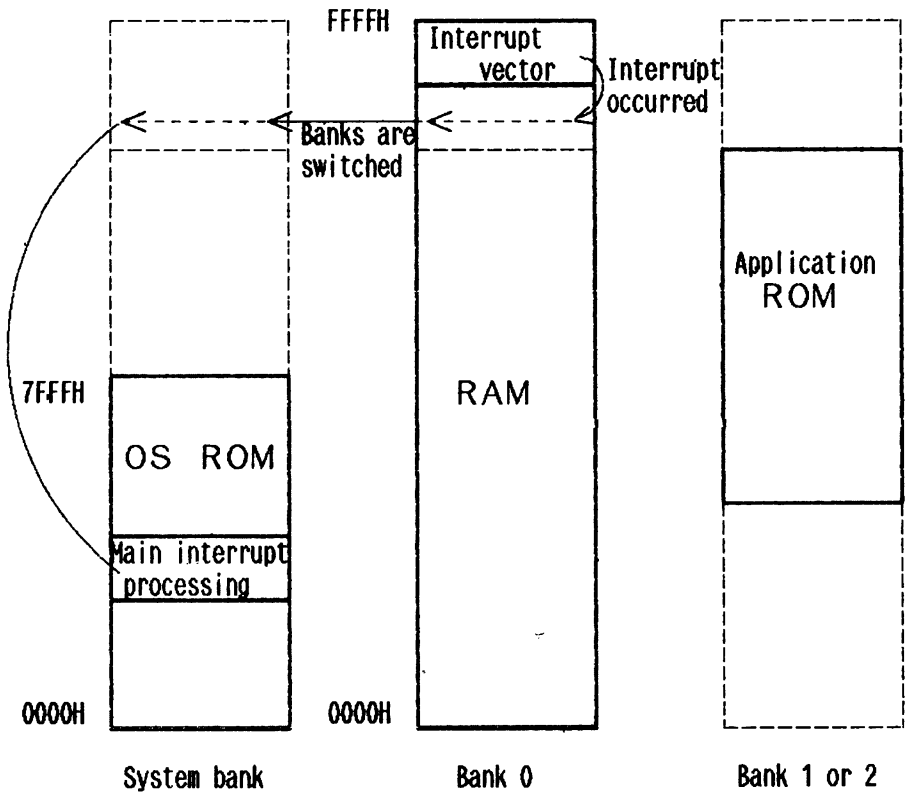


Fig. 4.7.6 Interrupt Processing When an Interrupt Occurred

(2) Relationship between Interrupt processing and BIOS
There are some types of interrupts which, when generated during BIOS processing, prevent the PINE from continuing program execution on return from interrupt processing or from performing successful power-off processing in the continue mode if it processes the interrupt immediately.

To solve this problem, the PINE sets the BIOS in-process flag on entry to BIOS (PREBIOS) and, if an interrupt occurs when this flag is on, causes the interrupt processing routine only to set the interrupt flag to memorize the occurrence of an interrupt. At the end of BIOS processing after the return from the interrupt processing, the PINE tests the interrupt flag and, if it is found to be set, performs the actual interrupt processing at the exit of BIOS (PSTBIOS).

BIOS functions which contain loops (e.g., CONIN, RSIN, and RSOUT) also test the interrupt flag as PSTBIOS does and invoke necessary interrupt processing accordingly.

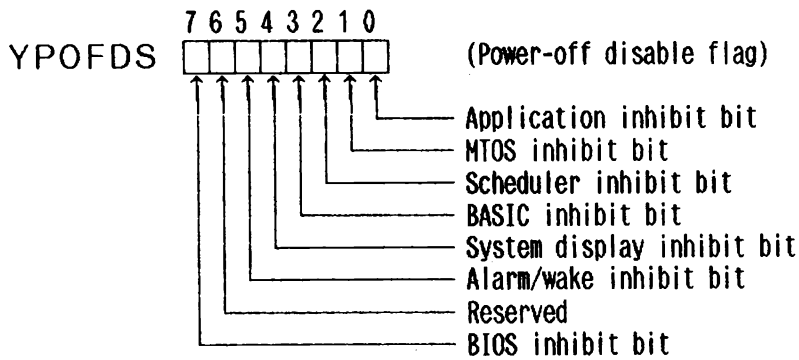
See also Section 3.3, "BIOS Operations" for details on PREBIOS and PSTBIOS.

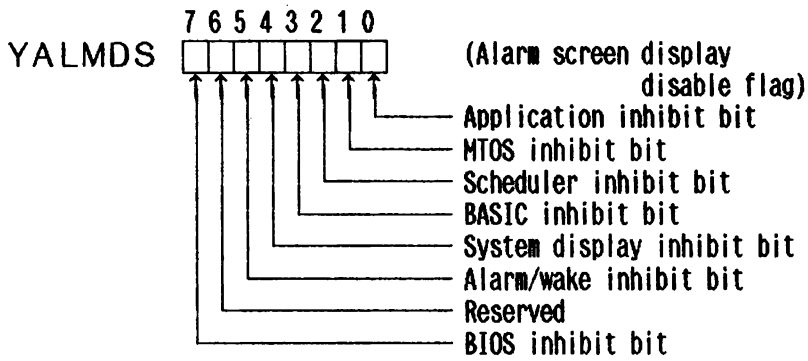
(3) Inhibiting interrupts
The PINE can inhibit the system from performing interrupt processing under program control. Interrupt processing that can be inhibited in this way includes:
1. Power-off processing and power fail processing
2. Alarm/wake processing

"Disabling interrupts under program control" means that the PINE accepts interrupt requests but inhibits the execution of the corresponding interrupt processing.

This facility is used by PREBIOS and PSTBIOS. Application programs can use this facility to temporarily inhibit power-off or alarm processing while they are taking specific actions.

a) How to inhibit interrupt processing
PINE OS uses the YPOFDS (ØEFEFH) and YALMDS (ØEFF1H) flag areas for inhibiting power-off and alarm/wake processing. The format of these areas are shown below.





Setting the corresponding bit to 1 inhibits power-off or alarm screen display processing.

During interrupt processing, the system copies the values of YPOFDS and YALMDS into YPOFST (ØEFFØH) and YALMST (ØEFF2H). If YPOFST or YALMST contains ØØH, the system performs power-off or alarm screen display processing; if both YPOFST and YALMST contains nonzero values, the system does nothing and terminates the interrupt processing.

b) Procedures for inhibiting interrupt processing
Figures 4.7.7 and 4.7.8 show the procedures for inhibiting interrupt processing. Application programs can manipulate only bit Ø of YPOFDS and YALMDS and must not use the other bits.

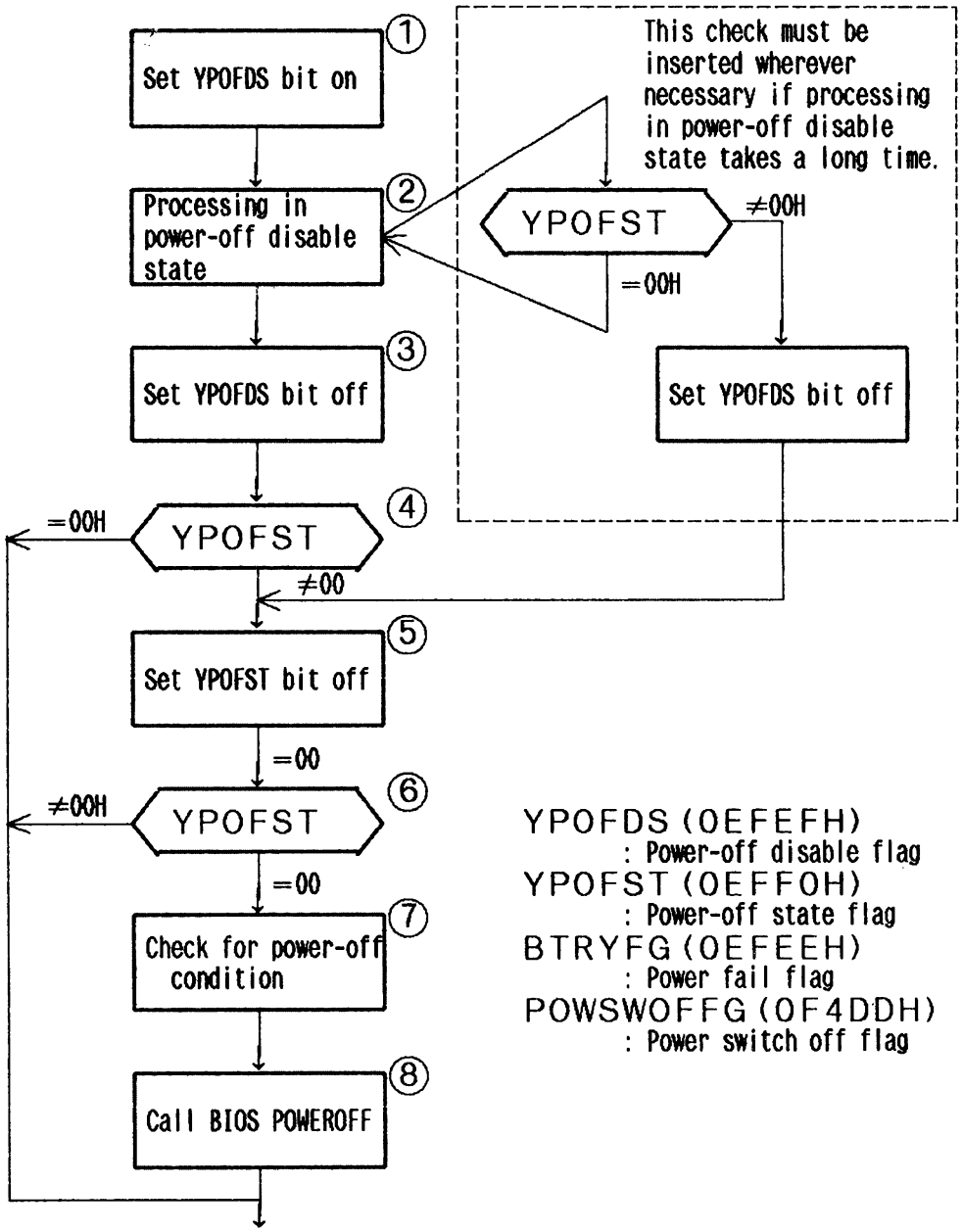


Fig. 4.7.7 Procedure for Inhibiting Power-off Processing

Step 1: Set YPOFDS bit on.
Set the specified YPOFDS (ØEFEFH) bit to 1. The application program must use bit Ø.

Step 2: Processing in power-off disable state.
Perform the processing to be executed in the power-off disabled state.
A power-off check routine must be inserted wherever necessary if the processing in power-off disable state takes a long time.

Step 3: Set YPOFDS bit off.
Set the YPOFDS bit that is set to 1 in step 1 to Ø.

Step 4: Check YPOFST.
Tests the value of YPOFST (ØEFFØH).
A nonzero value in YPOFST indicates that a power-off interrupt occurred while power-off processing was inhibited.

Step 5: Set YPOFST bit off.
Set the YPOFST bit that is set to 1 in step 1 to Ø.

Step 6: Check YPOFST.
Test the value of YPOFST (ØEFFØH).
A nonzero value in YPOFST indicates that power-off processing is inhibited by another module.

Step 7: Check for power-off condition.
Check the power-off conditions.
A nonzero in BTRYFG (ØEFEEH) indicates a continue mode power-off condition. If BTRYFG contains ØØH, the system examines PWSWOFFG (ØF4DDH). A Ø2H in PWSWOFFG indicates the restart condition and the other values indicate the continue-mode power-off condition.

Step 8: Call BIOS POWEROFF.
Call BIOS POWEROFF specifying the power-off condition found in step 7 specified.

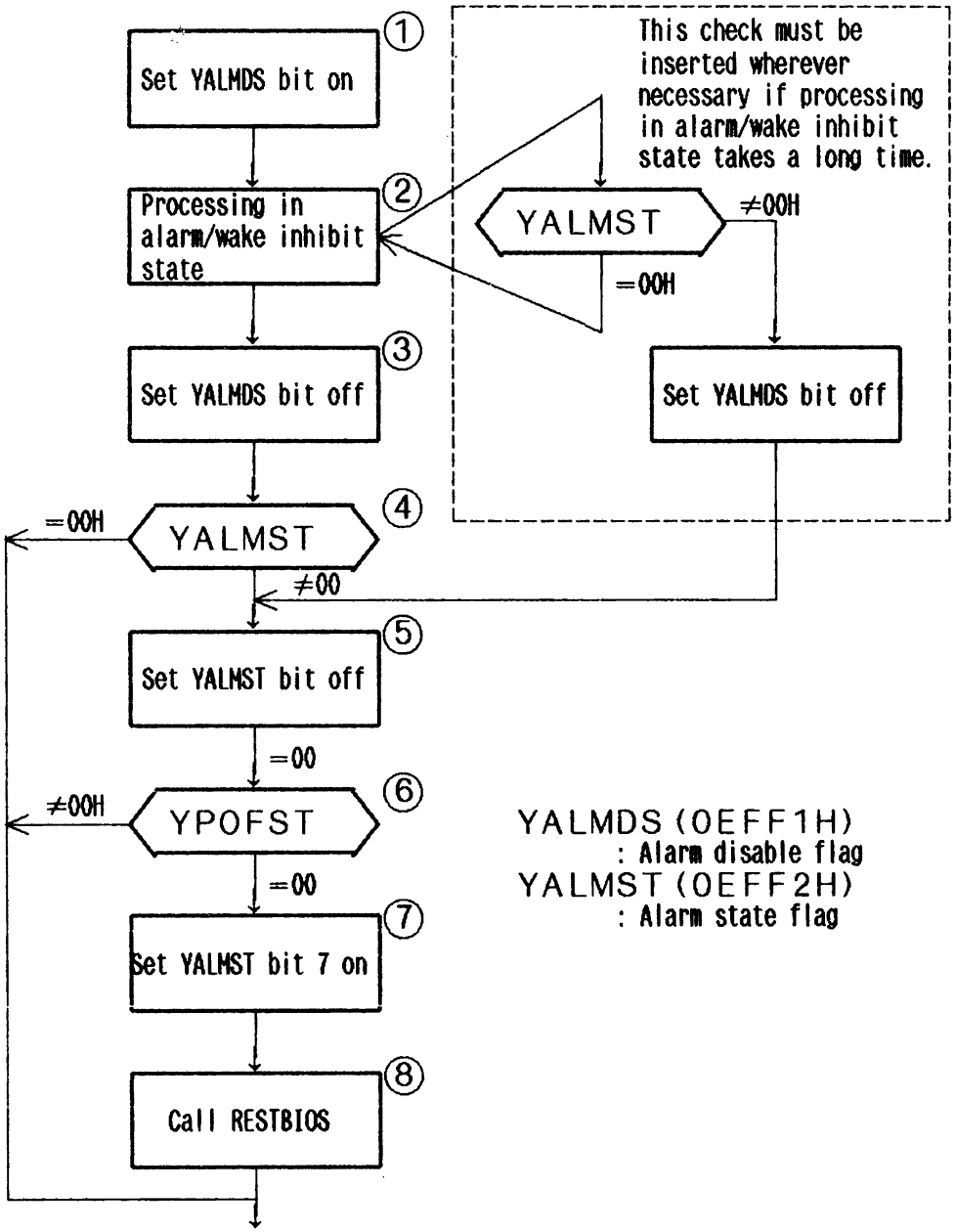


Fig. 4.7.8 Procedure for Inhibiting Alarm Screen Display Processing

Step 1: Set YALMDS bit on.
Set the specified YALMDS (ØEFF1H) bit to 1. The application program must use bit Ø.
Step 2: Processing in alarm/wake inhibit state.
Perform the processing to be executed in the alarm/wake screen display inhibit state.
A alarm/wake check routine must be inserted wherever necessary if the processing in alarm/wake inhibit state takes a long time.
Step 3: Set YALMDS bit off.
Set the YALMDS bit that is set to 1 in step 1 to Ø.
Step 4: Check YALMST.
Tests the value of YALMST (ØEFF2H).
A nonzero value in YALMST indicates that an alarm interrupt occurred while alarm/wake processing was inhibited.
Step 5: Set YALMST bit off.
Set the YALMST bit that is set to 1 in step 1 to Ø.
Step 6: Check YALMST.
Test the value of YALMST (ØEFF2H).
A nonzero value in YALMST indicates that alarm/wake screen display processing is inhibited by another module.
Step 7: Set YALMST bit 7 on.
Set bit 7 of YALMST (ØEFF2H) to 1. This enables alarm/wake screen to be displayed by PSTBIOS.
Step 8: Call RSPSTBIOS.
Call RSPSTBIOS (ØFF96H) to display the alarm/wake screen.

c) Programming notes

1. While the system is waiting for keyboard data with BIOS CONIN, it unconditionally executes power switch off, power fail, and alarm/wake processing even if they are inhibited. This also applies to BIOS which is waiting for send or receive data or for a printer ready signal (except during hardcopy processing).
2. Checks must be made for power-off or alarm/wake conditions if processing in the power-off or alarm/wake processing disabled state is to take a long time.

```
                        ;       ************************************************
                        ;               POWER OFF & ALARM CONTROL PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program shows how to control
                        ;               power off & alarm interrupt.
                        ;
                        ;       <> 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
EB7E                    POWEROFF        EQU     WBOOT   +7BH    ; Power off entry
                        ;
                        ;       System area
                        ;
EFEF                    YPOFDS          EQU     0EFEFH          ; Power off disable flag.
EFF0                    YPOFST          EQU     0EFF0H          ; Power off status.
EFF1                    YALMDS          EQU     0EFF1H          ; Alarm disable flag.
EFF2                    YALMST          EQU     0EFF2H          ; Alarm status.
EFEE                    BTRYFG          EQU     0EFEEH          ; Power fail status.
EFF0                    PWSWOFFG        EQU     0EFF0H          ; Power sw. off status.
                        ;
                        ;         RAM jump table
                        ;
FF96                    RSPSTBIOS       EQU     0FF96H          ; Post BIOS execute.
                        ;
0003                    STOP            EQU     03H             ; STOP code
000A                    LF              EQU     0AH             ; Line feed
000D                    CR              EQU     0DH             ; Carriage return
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This program is setting power off & alarm
                        ;               alarm disable, and if key inputed, do power
                        ;               off or alarm.
                        ;
0100                    MAIN:
0100    31 1000                 LD      SP,1000H        ; Set stack pointer.
                        ;
0103    CD 0121                 CALL    DISABLE         ; Interrupt disable.
                        ;
0106                    MAIN10:
0106    3A 0237                 LD      A,(PEND)        ; Stop key check.
0109    B7                      OR      A               ; Stop key pressed?
010A    C2 011B                 JP      NZ,MAIN20       ; Yes.
                        ;
010D    76                      HALT                    ; Wait until interrupt happened.
010E    CD 0144                 CALL    CHKINT          ; Check interrupt status.
0111    28 F3                   JR      Z,MAIN10        ; Neither power off nor alarm
                        ;
0113    F5                      PUSH    AF              ; Save interrupt information.
0114    CD 0185                 CALL    KEYIN           ; Message display and key in.
0117    F1                      POP     AF              ; Restore interrupt information.
                        ;
0118    CD 019F                 CALL    OKINT           ; Interrupt execute.
                        ;
011B                    MAIN20:
011B    CD 0132                 CALL    ENABLE          ; Interrupt enable.
011E    C3 EB03                 JP      WBOOT           ; End.
                        ;
                        ;       ************************************************
                        ;               DISABLE POWER OFF & ALARM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Disable the following system function.
                        ;                1. Power off execute.
                        ;                2. Alarm screen display.
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0121                    DISABLE:
0121    3A EFEF                 LD      A,(YPOFDS)      ; Set power off disable.
0124    F6 01                   OR      01H             ; Bit 0 is application bit.
0126    32 EFEF                 LD      (YPOFDS),A      ;
                        ;
0129    3A EFF1                 LD      A,(YALMDS)      ; Set alarm disable.
012C    F6 01                   OR      01H             ; Bit 0 is application bit.
012E    32 EFF1                 LD      (YALMDS),A      ;
0131    C9                      RET                     ;
                        ;
```

```
                        ;       ************************************************
                        ;               ENABLE POWER OFF & ALARM INTERRUPT
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0132                    ENABLE:
0132    21 EFEF                 LD      HL,YPOFDS       ; Reset my disable bit.
0135    CB 86                   RES     0,(HL)          ;

0137    21 EFF1                 LD      HL,YALMDS       ; Reset my disable bit.
013A    CB 86                   RES     0,(HL)          ;

013C    CD 0144                 CALL    CHKINT          ; Check interrupt happened.
013F    C8                      RET     Z               ; No interrupt.
0140    CD 019F                 CALL    OKINT           ; Interrupt execute.
                        ;
0143    C9                      RET                     ;
                        ;
                        ;       ************************************************
                        ;               CHECK POWER OFF & ALARM INTERRUPT
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Check power off & alarm interrupt occurred.
                        ;               If occurred, set the information to return
                        ;               code.
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               Z-flag  : Return information.
                        ;                       =0 -- Both interrupt not occurred.
                        ;                       =1 -- Interrupr occurred
                        ;               A       : Interrupt type
                        ;                 bit 0 : Alarm interrupt.
                        ;                 bit 1 : Power off interrupt.
                        ;                       ( 1=occurr, 0=not occurr)
                        ;
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;               If STOP key is pressed, then sets
                        ;               PEND flag.
                        ;
0144                    CHKINT:
0144    CD EB06                 CALL    CONST           ; Key in check.
0147    3C                      INC     A               ; No inputed key?
0148    20 0A                   JR      NZ,CHK10        ; Yes.

014A    CD EB09                 CALL    CONIN           ; Get key code.
014D    FE 03                   CP      STOP            ; STOP key?
014F    20 03                   JR      NZ,CHK10        ; No.
                        ;
0151    32 0237                 LD      (PEND),A        ; Set program end flag.
                        ;
0154                    CHK10:
0154    0E 00                   LD      C,00H           ; Clear return information.

0156    3A EFF0                 LD      A,(YPOFST)      ; Check power off status.
0159    B7                      OR      A               ; Power off occurred?
015A    28 10                   JR      Z,CHK20         ; No.

015C    21 EFEF                 LD      HL,YPOFDS       ; Reset my disable bit.
015F    CB 86                   RES     0,(HL)          ;
                        ;
0161    E6 FE                   AND     11111110B       ; Reset my status bit.
0163    32 EFF0                 LD      (YPOFST),A      ;
0166    20 04                   JR      NZ,CHK20        ; Disable by other.
                        ;
0168    79                      LD      A,C             ; Set Power-off- go bit.
0169    F6 02                   OR      02H             ;
016B    4F                      LD      C,A             ;
                        ;
016C                    CHK20:
016C    3A EFF2                 LD      A,(YALMST)      ; Check alarm status.
016F    B7                      OR      A               ; Alarm occurred?
0170    28 10                   JR      Z,CHK40         ; No.

0172    21 EFF1                 LD      HL,YALMDS       ; Reset my disable bit.
0175    CB 86                   RES     0,(HL)          ;
                        ;
0177    E6 FE                   AND     11111110B       ; Reset my status bit.
0179    32 EFF2                 LD      (YALMST),A      ;
017C    20 04                   JR      NZ,CHK40        ; Disable by other.
                        ;
017E    79                      LD      A,C             ; Set alarm-go bit.
017F    F6 01                   OR      01H             ;
0181    4F                      LD      C,A             ;
                        ;
0182                    CHK40:
0182    79                      LD      A,C             ; Set return information.
0183    B7                      OR      A               ;
0184    C9                      RET                     ;
```

```
                        ;       ************************************************
                        ;               STATUS DISPLAY & KEY IN
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               If power off or alarm occurred, then
                        ;               display message & wait until key inputed.
                        ;
                        ;       <> entry parameter <>
                        ;               A  : Interrupt type.
                        ;                  bit 0 : alarm
                        ;                  bit 1 : power off
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
0185                    KEYIN:
0185    0F              ;               RRCA                    ; Alarm bit --> CY
0186    F5              ;               PUSH    AF              ; Save interrupt status.
0187    21 01D3         ;               LD      HL,MSG01        ; Alarm happen message.
018A    DC 01C7         ;               CALL    C,DSPMSG        ; Display if alarm occurred.
018D    F1              ;               POP     AF              ; Restore interrupt status.
018E    0F              ;               RRCA                    ; Power off bit --> CY
018F    21 01EF         ;               LD      HL,MSG02        ; Power off happen message.
0192    DC 01C7         ;               CALL    C,DSPMSG        ; Display if power off occurred.
                        ;
0195    21 021A         ;               LD      HL,MSG03        ; Key in message
0198    CD 01C7         ;               CALL    DSPMSG          ;
                        ;
019B    CD EB09         ;               CALL    CONIN           ; Input any key.
019E    C9              ;               RET                     ;
                        ;
                        ;       ************************************************
                        ;               POWER OFF OR ALARM EXECUTE
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Power off or alarm execute in
                        ;               this routine.
                        ;
                        ;       <> entry parameter <>
                        ;               A   : Interrupt type.
                        ;                  bit 0 : alarm
                        ;                  bit 1 : power off
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
019F                    OKINT:
019F    F5              ;               PUSH    AF              ; Save interrupt information.
01A0    CB 4F           ;               BIT     1,A             ; Power off?
01A2    28 13           ;               JR      Z,OK20          ; No.
                        ;
01A4    0E 00           ;               LD      C,00H           ; Set continue power off mode.
01A6    3A EFEE         ;               LD      A,(BTRYFG)      ; Power fail check.
01A9    B7              ;               OR      A               ; Power fail?
01AA    20 08           ;               JR      NZ,OK10         ; Yes.
                        ;
01AC    3A EFF0         ;               LD      A,(PWSWOFFG)    ; Power off check.
01AF    FE 02           ;               CP      02H             ; Continue power off?
01B1    28 01           ;               JR      Z,OK10          ; Yes.
                        ;
01B3    0C              ;               INC     C               ; Set restart power off.
                        ;
01B4                    OK10:
01B4    CD EB7E         ;               CALL    POWEROFF        ; Go power off.
                        ;
01B7                    OK20:
01B7    F1              ;               POP     AF              ; Restore interrupt information.
01B8    CB 47           ;               BIT     0,A             ; Alarm?
01BA    C8              ;               RET     Z               ; No.
                        ;
01BB    3A EFF2         ;               LD      A,(YALMST)      ; Set BIOS bit
01BE    F6 80           ;               OR      10000000B       ;
01C0    32 EFF2         ;               LD      (YALMST),A      ;
01C3    CD FF96         ;               CALL    RSPSTBIOS       ; Go alarm.
                        ;
01C6    C9              ;               RET                     ;
                        ;
                        ;       ************************************************
                        ;               DISPLAY MESSAGE UNTIL FIND 00H
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               HL : Message top address.
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
01C7                    DSPMSG:
01C7    4E              ;               LD      C,(HL)          ; Get message data.
01C8    0C              ;               INC     C               ;
01C9    0D              ;               DEC     C               ; End of message?
01CA    C8              ;               RET     Z               ; Yes.
                        ;
01CB    E5              ;               PUSH    HL              ; Save pointer.
```

```
01CC    CD EB0C                 CALL    CONOUT          ; Message display.
01CF    E1                      POP     HL              ; Restore pointer.
01D0    23                      INC     HL              ; Poiner update.
01D1    18 F4                   JR      DSPMSG          ; Loop
                        ;
                        ;       Message and work area
                        ;
01D3                    MSG01:
01D3    41 6C 61 72             DB      'Alarm interrupt occurred.'
01D7    6D 20 69 6E
01DB    74 65 72 72
01DF    75 70 74 20
01E3    6F 63 63 75
01E7    72 72 65 64
01EB    2E
01EC    0D 0A 00                DB      CR,LF,00H
                        ;
01EF                    MSG02:
01EF    50 6F 77 65             DB      'Power switch off or Power fail occurred.'
01F3    72 20 73 77
01F7    69 74 63 68
01FB    20 6F 66 66
01FF    20 6F 72 20
0203    50 6F 77 65
0207    72 20 66 61
020B    69 6C 20 6F
020F    63 63 75 72
0213    72 65 64 2E
0217    0D 0A 00                DB      CR,LF,00H
                        ;
021A                    MSG03:
021A    50 72 65 73             DB      'Press any key to continue.'
021E    73 20 61 6E
0222    79 20 6B 65
0226    79 20 74 6F
022A    20 63 6F 6E
022E    74 69 6E 75
0232    65 2E
0234    0D 0A 00                DB      CR,LF,00H
                        ;
                        ;
0237                    PEND:
0237    00                      DB      00H             ; Program end flag.
                        ;
                                END
```

4.7.3.6 Extending interrupt processing

Interrupt processing can be extended by:
1. Using interrupt hooks.
2. Rewriting the interrupt vector.
3. Checking the interrupt state.

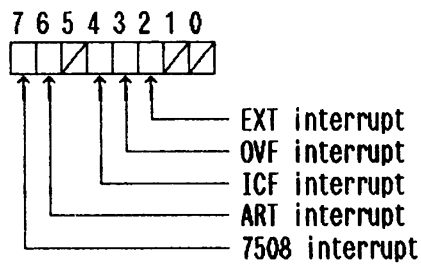
(1) Extending interrupt processing using interrupt hooks.
The interrupt hooks that the PINE supports include ICFHOOK, OVFHOOK, EXTHOOK, and HK8251 (OS kana V2.Ø (in Japan) only). See Section 4.3, "Hooks" for the location and the user of these hooks.

(2) Extending interrupt processing by rewriting the interrupt vector
User-supplied interrupt processing can be executed by rewriting the interrupt jump vector (ØFFFØH through ØFFFFH). The user must exercise care with the following when rewriting the interrupt jump vector:
1. Rewrite the jump vector after disabling the CPU for interrupts (DI state).
2. Place the new interrupt processing routine in the resident area between ØE000H through ØFFFFH, that is, the new vector entry must point to somewhere between ØE000H and ØFFFFH.
3. The interrupt processing routine must reserve its own stack area.
4. The interrupt processing routine must not use BDOS or BIOS.
5. On exit, restore the registers into the original state established when it is entered (in other words, all registers must be preserved).

(3) Checking the interrupt state
With this method, the interrupt processing routine does not perform interrupt processing immediately but checks for interrupts frequently. The occurrence of interrupts can be tested by examining the following:
1. Interrupt flag INTTYPE (ØEFD3H)
2. Interrupt Status Register (ISR) (PØ4H)

See Part I, "Firmware" for ISR.

INTTYPE (ØEFD3H) 1 byte
Interrupt flags



When an interrupt occurs, the corresponding bit position is set to 1.

4.7.4 7508 Interrupts

4.7.4.1 Outline

When an interrupt is generated by the 7508, the interrupt processing routine reads the 7508 status through the serial communication interface, and transfers control to the corresponding routine that services the interrupt. The routines that service 7508 interrupts include:

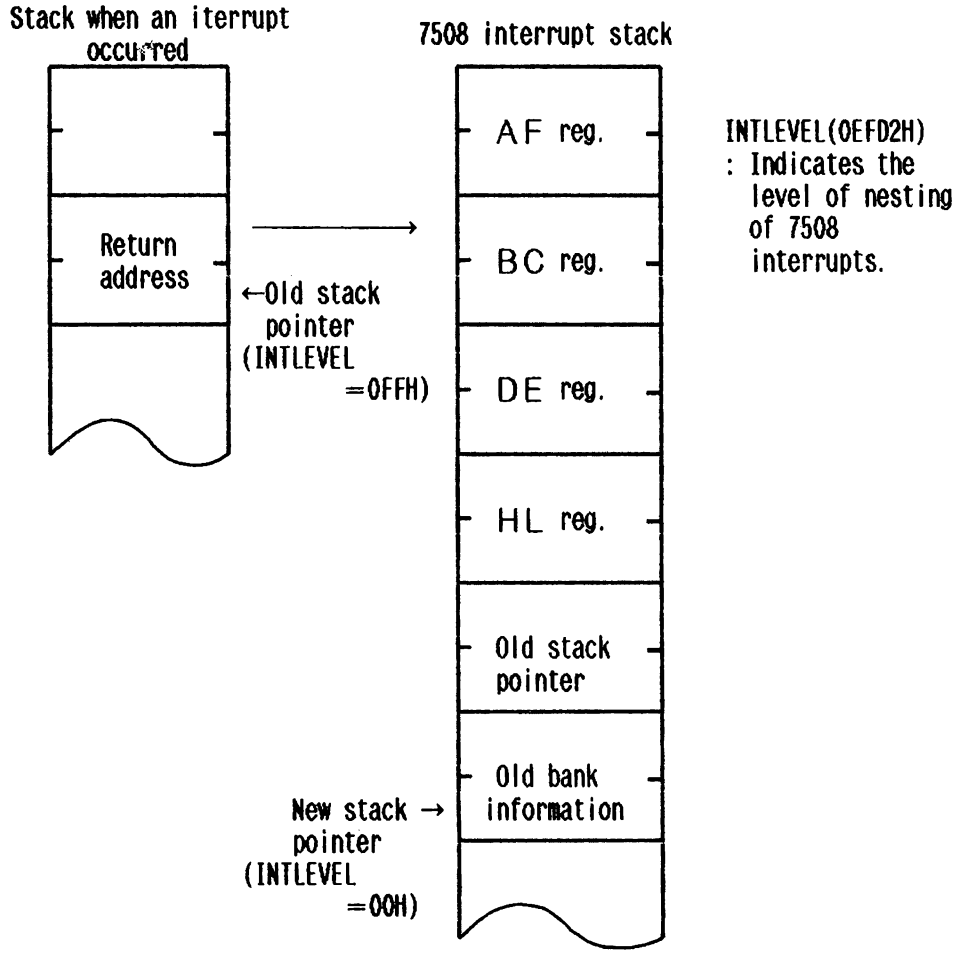
- Key interrupt
- 1-second interrupt
- Power fail interrupt
- Alarm interrupt
- Power switch interrupt

7508 interrupt processing allows multiple 7508 interrupts so that the PINE can accept key inputs while displaying the alarm screen.

4.7.4.2 Multiple interrupt processing

Provisions are made in 7508 interrupt processing so that multiple 7508 interrupts can be honored.

Multiple interrupt processing is accomplished using the interrupt level work area INTLEVEL (0EFD2H) and stacks. The stacks used for multiple interrupt processing are illustrated below.



The stack status in the above figure shows the one when a 7508 interrupt occurred. If another 7508 interrupt occurs in this condition, the interrupt processing routine does not switch the stacks, but saves the contents of 7508 interrupt stack, from AF register to old bank information, and increments INTLEVEL by 1.

Fig. 4.7.9 Multiple 7508 Interrupt Processing

4.7.4.3 Types of 75Ø8 interrupts

There are five types of 75Ø8 interrupts which can be identified by reading the 75Ø8 status.

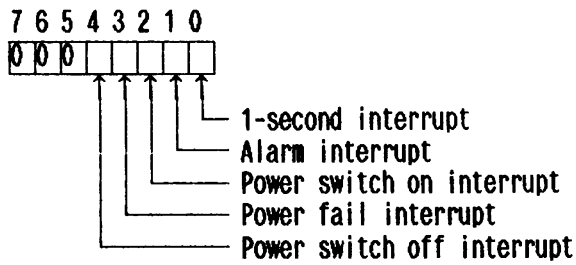
When an interrupt is generated by the 75Ø8, the system issues a Status Read command (Ø2H) to the 75Ø8 to read the 75Ø8 status. A status value ØBFH or smaller indicates a hard code generated by a key interrupt and a status value ØCØH or greater indicates a power switch, 1-second, alarm, or power fail interrupt. The system stores this status in STS75Ø8 (ØF4D6H).

Power switch interrupts are further divided into power switch on and off interrupts. After loading the status of the interrupt (power switch on/off, 1-second, alarm, or power fail interrupt), the system transfers control to the corresponding interrupt servicing routine. The interrupt servicing routine to be started is determined from the status information in INTFG (ØF4D5H) and the table TBL75Ø8 at ØEFDEH.

The status of 1-second and alarm interrupt processing is stored in FG75Ø8 at ØF4D7H.

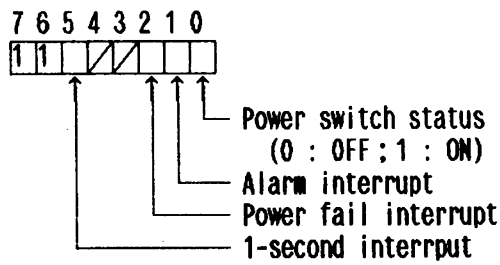
The work areas related to 75Ø8 interrupt processing are given below.

INTFG (ØF4D5H) 1 byte
Indicates the type of 75Ø8 interrupt processing performed by the system. INTFG is set to ØØH when key interrupt processing is performed.



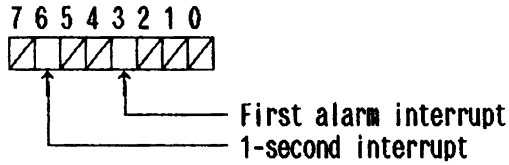
A 1 in a bit position indicates that the corresponding interrupt occurred.

STS75Ø8 (ØF4D6H) 1 byte
Contains the status read by the system from the 75Ø8 when a 75Ø8 interrupt occurred. Values greater than or equal to ØØH and smaller than or equal to ØBFH identify key interrupts and represent keys' hard codes.
When STS75Ø8 contains a value greater than or equal to ØCØH:



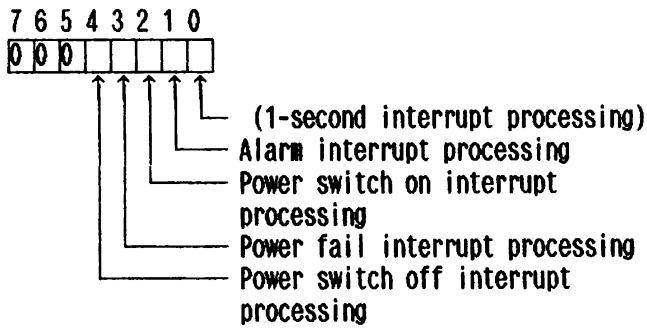
A 1 in a bit position indicates that the corresponding interrupt occurred.

FG7508 (ØF4D7H) 1 byte
Is the 7508 interrupt processing flag.



A 1 in a bit position indicates that the corresponding interrupt occurred.

TBL7508 (ØEFDEH) 16 bytes
Is the table used to determine the 7508 interrupt processing to be serviced. Each byte has the following format:



A 1 in a bit position indicates that the corresponding interrupt occurred.

The meaning of the table entries are listed below.

Address	7508 status	PW switch status	initial value
EFDEH	C0H or E0H	OFF	00H
DF		ON	10H
E0	C1H or E1H	OFF	04H
E1		ON	00H
E2	C2H or E2H	OFF	02H
E3		ON	12H
E4	C3H or E3H	OFF	04H
E5		ON	02H
E6	C4H or E4H	OFF	08H
E7		ON	10H
E8	C5H or E5H	OFF	08H
E9		ON	08H
EA	C6H or E6H	OFF	08H
EB		ON	10H
EC	C7H or E7H	OFF	08H
ED		ON	08H

The presence or absence of the 1-second interrupt processing routine can be determined by software after rewriting the table.

4.7.4.4 Key interrupt processing

When an interrupt is generated by the 7508, the system issues a Status Read command (02H) to the 7508 to read the 7508 status. A status value 0BFH or smaller indicates that a key interrupt occurred and it is processed as such.

When a key interrupt is generated by the 7508, the system interprets the status data as a hard code which indicates the position of the key on the keyboard matrix. The key interrupt processing routine places the key's hard code into the key buffer (KBUF). If the key buffer is full, the key code is discarded.

If the hard code received from the 7508 is the one for the STOP key (10H for the standard keyboard and 0B6H for the item keyboard), the system performs special processing different from that for ordinary keys.

See Section 3.5, "Keyboard" for the key buffer and hard codes.

(1) STOP key special processing
Interrupt processing proceeds as follows when the system receives a hard code for the STOP key from the 7508:

a) Sends a Keyboard Clear command to the 7508 to clear the 7-byte buffer of the 7508, and waits for approximately 40 ms (key scan time). If an item keyboard is installed or the STOP and CTRL/STOP keys are disabled (specified in ISYSREG (0F01EH)), the system does no special processing but carries out normal key interrupt processing.
b) Reads the 7508 buffer to check whether the current key is pressed with the SHIFT key.
c) Loads 0FFH into the BRKFLG (0F019H) flag that indicates that the STOP key has been pressed.

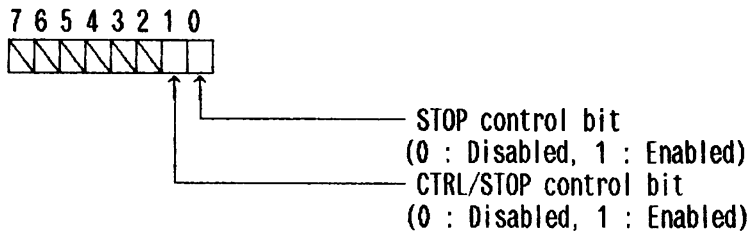
d) If the CTRL key is also pressed, the system loads 01H into CSTOPFLG (0F01AH), which is a flag indicating that the CTRL and STOP keys have been pressed simultaneously, CSTOPMCT (0F01BH), and CSTOPPRN (0F309H). The system bypasses this step if an item keyboard is installed or the CTRL and STOP keys are disabled.

e) Checks the state of the switch key that is currently being pressed to determine the keyboard state.

f) Clears the key buffer (KBUF) and loads the hard code (10H) for the STOP key into the key buffer. The system places 10H into the key buffer also when an item keyboard is installed.

The work areas related to key interrupt processing are described below.

ISYSREG (0F01EH) 1 byte
Is the system function flag.



BRKFLG (0F019H) 1 byte
Indicates whether the STOP key has been pressed.
= 00H: The STOP key has not been pressed.
= 0FFH: The STOP key has been pressed.

CSTOPFLG (0F01AH) 1 byte
Indicates whether the CTRL/STOP keys have been pressed.
= 00H: The CTRL/STOP keys have not been pressed.
= 01H: The CTRL/STOP keys have been pressed.

CSTOPMCT (0F01BH) 1 byte
Flag for terminating microcassette processing.
= 00H: The CTRL/STOP keys have not been pressed.
= 01H: The CTRL/STOP keys have been pressed.

CSTOPPRN (0F309H) 1 byte
Flag for terminating cartridge printer.
= 00H: The CTRL/STOP keys have not been pressed.
= 01H: The CTRL/STOP keys have been pressed.

Note: BRKFLG and CSTOPFLG are reset to 00H when BIOS CONIN or CONST is called and CSTOPMCT and CSTOPPRN are reset by PSTBIOS.

4.7.4.5 1-second interrupt processing

1-second interrupt processing proceeds as follows:

a) Increments 16-bit timer TIMER0 (0EF8FH) by 1.
b) Decrements 16-bit timer TIMER1 (0EF91H) by 1.
c) Processes timer function TMFUNC (0F313H).
   When TMFUNC (0F313H) contains a nonzero value, the interrupt processing routine decrements TMSEC (0F4DAH) by 1. If the TMSEC reaches 00H as the result of the decrement, the interrupt processing routine sets TMFUNC to 00H and TMGLAG (0F314H) to 0FFH.
d) Counts down the ROM cartridge power-off time.
e) Counts down the alarm repeat time.

The areas referred to in steps a), b), and c) are 1-second counters available for application programs and those referred to in steps d) and e) are counters available for system control. TIMER0 is also used by the system to monitor the auto power-off time.

(1) Uses of TIMER0 and TIMER1

TIMER0 and TIMER1 are 16-bit 1-second counters and incremented or decremented by 1 every time a 1-second interrupt occurs. These counters can be used as reference-only counters (not overwritten) to measure processing time.

(2) Use of TMFUNC

TMFUNC is a flag area available for the user to specify the execution of the timer function for checking for the lapse of a specified time. Figure 4.7.10 shows how to use this flag.

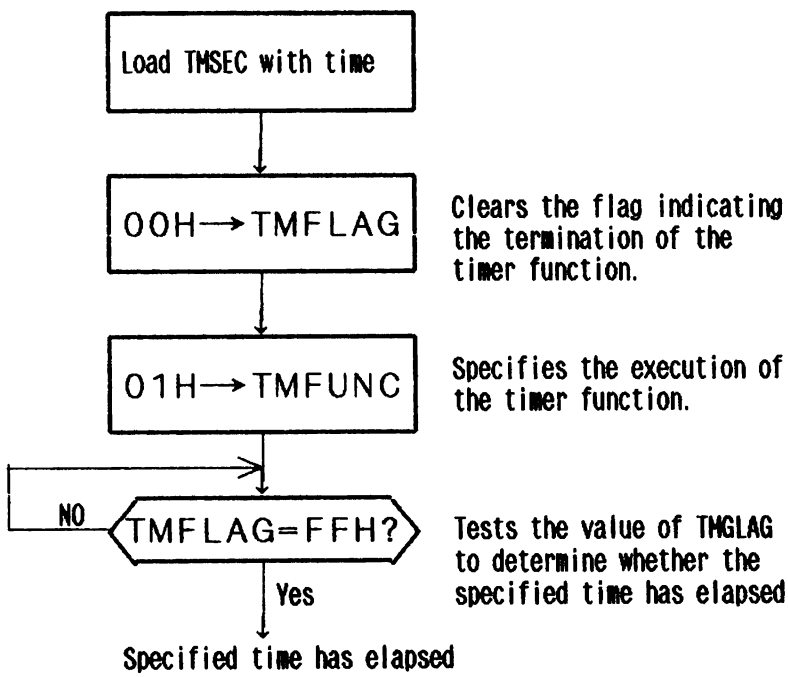


Fig. 4.7.10 Using TMFUNC

The work areas related to 1-second interrupt processing are described below.

TIMER0 (0EF8FH) 2 bytes
16-bit timer incremented by 1 every time a 1-second interrupt occurs.

TIMER1 (0EF91H) 2 bytes
16-bit timer decremented by 1 every time a 1-second interrupt occurs.

TMFUNC (0F313H) 1 byte
Timer function flag.
= 00H: The timer function is not specified.
= Nonzero: The timer function is specified.

TMFLAG (0F314H) 1 byte
Timer flag.
When TMFUNC contains a nonzero value, TMSEC is decremented by 1 every 1-second interrupt and TMFLAG is set to 0FFH when TMSEC reaches 0000H.

TMSEC (0F4DAH) 2 bytes
Timer counter.
When TMFUNC contains a zero value, TMSEC is decremented by 1 every 1-second interrupt and when TMSEC reaches 0000H, TMFLAG is set to 0FFH and TMFUNC to 00H.

4.7.4.6 Alarm interrupt processing

An alarm interrupt occurs when the time specified by BIOS TIMDAT with a Set Alarm/Wake is reached. Alarm interrupts are distinguished from wake interrupts by software; on hardware, only alarm interrupts can occur.

Although the alarm time can be specified to the precision of month, day, day of the week, hour, minute, and second, the minimum unit is 10 seconds. This is because the 7508 recognizes the arrival of an alarm time by a coincidence between the counter value and the specified time in the precision up to 10 seconds. Accordingly, once the set alarm time is reached, a maximum of 10 alarm interrupts can occur. PINE OS takes one of the 10 interrupts as a true alarm interrupt.

PINE OS places hooks in the alarm interrupt processing routine so that the application programs can extend the capability of the alarm interrupt processing routine. See Section 4.3, "Hooks" for details on alarm hooks.

The display of the alarm screen can be disabled by software during alarm processing using ALRMDS (0EFF1H).

4.7.4.7 Power switch on interrupt processing

A power switch on interrupt occurs when the power switch is switched from the OFF to ON position. The power switch on processing routine only sets the power switch on flag PWSWONFG (0F4DCH). See Section 2.4, "Power-on" for power-on processing started when the main power is off.

The work area related to power switch on interrupt processing is described below.

PWSWONFG (ØF4DCH) 1 byte
Power switch on flag.
= ØØH: No power switch on interrupt has occurred.
= ØFFH: A power switch on interrupt has occurred.
Reset to ØØH when power switch on processing is started with the main power switch in the OFF position.

4.7.4.8 Power fail interrupt processing

A power fail interrupt occurs when the main battery voltage falls below a certain level. The threshold level is 4.8 volts for NiCd batteries and 4.Ø volts for Mn dry batteries.

When a power fail interrupt occurs, the corresponding interrupt processing routine sets the power fail flag BTRYFG (ØEFEEH) to alert that the battery voltage has been fallen. See Section 2.6, "Power Fail" for details on the power fail screen.

Once a power fail interrupt occurs, the same interrupt occurs every one second. If main power is not shut down within 5Ø seconds after the first power fail interrupt, the 75Ø8 forces main power to be shut down.

The work area related to power fail interrupt processing is described below.

BTRYFG (ØEFEEH) 1 byte
Power fail interrupt flag.
= ØØH: No power fail interrupt has occurred.
= ØFFH: A power fail interrupt has occurred.
Reset to ØØH by a power-on start sequence.

4.7.4.9 Power switch off interrupt processing

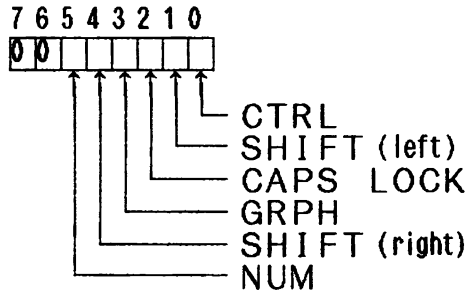
A power switch off interrupt occurs when the power switch is switched from the ON to OFF position. The power switch off processing routine tests the current key shift state to determine whether the CTRL key has been pressed, places the result into power switch off flag PWSWOFFG (ØF4DDH), then turns off power.

The key shift state that determines the continue or restart mode power-off can be altered. A continue mode power-off occurs when the value of CNTNKEY (ØEF2AH) match the key shift state when the power switch off interrupt occurred. See Section 2.5, "Power-off" for details on power off processing.

The work areas related to power switch off interrupt processing are described below.

PWSWOFFG (ØF4DDH) 1 byte
Power switch off flag.
= ØØH: No power switch off interrupt has occurred.
= Ø1H: A restart mode power switch off interrupt occurred.
= Ø2H: A continue mode power switch off interrupt has occurred.

CNTNKEY (ØEF2AH) 1 byte
Continue key shift state flag.



A 1 in a bit position indicates that the corresponding shift key is pressed.
Only bit Ø (CTRL) is valid when an item keyboard is installed.
Only continue mode power switch off interrupts are enabled when CNTNKEY contains ØØH, irrespective of the key shift state.
This area is initialized to Ø1H.

4.7.5 ART Interrupts

An ART interrupt occurs when the RxRDY pin in the ART block is set. ART interrupt processing proceeds as follows (see also Section 3.4, "BIOS Details" since ART interrupt processing is closely related to BIOS RSIOX processing):

a) Does nothing if the RSIOX OPEN function has not been performed.
b) Checks for errors (framing, receive overrun, parity, and receive buffer overflow errors).
c) Reads the received data into the receive buffer.
d) If XON/XOFF control is specified, checks the number of the received bytes in the receive buffer and sends an XOFF code if it exceeds the 3/4 of the buffer size.

4.7.6 OVF Interrupts

An OVF interrupt occurs when an FRC (Free Running Counter) overflow occurs. The FRC is a 16-bit counter running on the 616.6 kHz clock, so an OVF interrupt occurs every 1Ø6.7 msec or so.

PINE OS uses OVF interrupts for controlling cursor blinking. THe cursor turns on and off every 5ØØ msec or so. The blinking interval can be changed by rewriting BLNKTIME (ØEFBAH).

BLNKTIME (ØEFBAH) 1 byte
Specifies the cursor blink time in 1ØØ ms units. This area is initialized to Ø4H.

4.7.7 ICF/EXT Interrupts

ICF interrupts occur as the state of the input to the barcode reader or external cassette drive changes whereas EXT interrupts are generated by the interrupt signal from external devices (sent via the system bus).

When an ICF or EXT interrupt occurs, the PINE does nothing but sets interrupt flag INTTYPE at ØEFD3H. Hooks are provided for extension of ICF/EXT interrupt processing. See Section 4.3, "Hooks" for details.

CHAPTER 5 I/O OPERATIONS

5.1 Cartridges Interface II-502
5.1.1 General II-502
5.1.2 Modes and Mode Setting II-502
5.1.3 ROM Cartridge II-506
5.1.4 RAM Cartridge II-511
5.1.5 Microcassette Drive II-515
5.1.6 Cartridge Printer II-532
5.1.7 Modem/DMM Cartridge II-536

5.2 Serial Interfaces II-561
5.2.1 General II-561
5.2.2 Setting Up Serial Interface II-561
5.2.3 System Support II-563

5.3 Disk Drive Unit II-566
5.3.1 General II-566
5.3.2 EPSON Serial Communication Protocol (EPSP) II-566
5.3.3 Use II-571

5.4 External RAM Disk II-572
5.4.1 General II-572
5.4.2 Hardware General II-572
5.4.3 Structure II-573
5.4.4 Use II-576
5.4.5 System Support II-579

5.5 Barcode Reader and External Cassettes II-582
5.5.1 General II-582
5.5.2 Structure II-582
5.5.3 Use II-583
5.5.4 System Support II-585

5.6 ROM Capsules II-586
5.6.1 General II-586
5.6.2 Structure II-586
5.6.3 Use II-588
5.6.4 System Support II-589

5.7 Printer Interface II-590
5.7.1 General II-590
5.7.2 Structure II-590
5.7.3 Use II-591
5.7.4 System Support II-591

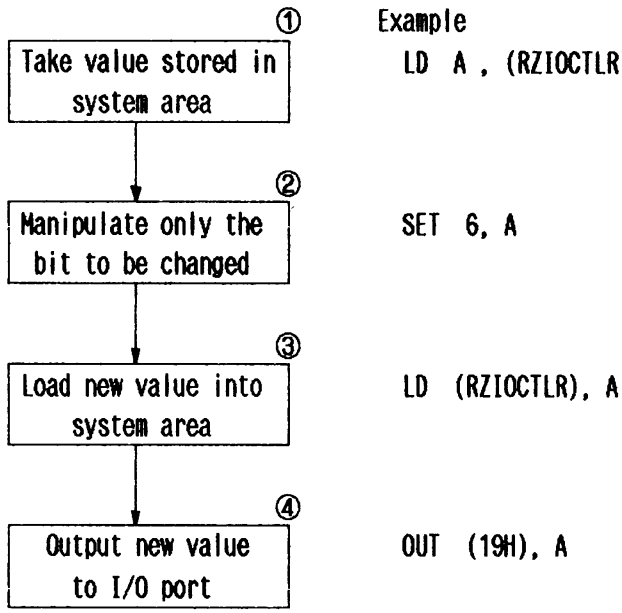
5.8 Printer II-592
5.8.1 General II-592
5.8.2 System Support II-592
5.8.3 Printer Types II-594

CHAPTER 5 I/O OPERATIONS

This chapter describes PINE I/O devices such as cartridges and serial interfaces.

This chapter also covers I/O registers. Points to be noted when rewriting I/O registers are shown below.

PINE OS stores the current contents of the output registers in the system area. An application program can write to an I/O register through the following procedure:



The following registers must be written to by this procedure:

I/O address	Register name	System area address	Variable name	Remarks
P00H	CTLR1	0F001H	RZCTLR1	
P04H	IER	0F53EH	RZIER	Disable interrupt during writes.
P05H	BANKR	0F53DH	RZBANKR	
P15H	ARTMR	0F003H	RZARTMR	
P16H	ARTCR	0F004H	RZARTCR	
P18H	SWR	0F005H	RZSWR	
P19H	IOCTLR	0F006H	RZIOCTLR	

The PINE provides the following I/O registers for display processing:

I/O address	Register name	System area address	Variable name	Remarks
P08H	VADR	0F295H	LSCRVRAM+1	The value must be 80H or higher.
P09H	YOFF	0F2A0H	LVRAMYOFF	DSPFLAG indicates bit 7, and LVRAMYOFF bits 6 - 0.
P05H		0EFB6H	DSPFLAG	

The PINE I/O map is as follows:

Table 5.1.1 I/O Map

I/O port	Description
P00H - P0FH	Reserved by the system.
P10H - P13H	Reserved by the system for expansion cartridges.
P14H - P1FH	Reserved by the system.
P20H - P8FH	Not used.
P90H - P94H	Reserved by the system for external RAM disks.
P95H - PFFH	Not used.

See Part I, "Firmware" for details of individual registers.

5.1 Cartridges Interface

5.1.1 General

Optional devices such as microcassette, ROM, and RAM cartridges can be connected to the PINE cartridge interface. The cartridge interface runs in different modes depending on the type of device connected. It automatically switches to the correct mode for a given device.

This section describes how PINE OS controls cartridges. See Part I Section 4.1, "Cartridge Interface" for cartridge interface.

When using expansion cartridges, the user must not assign the following device codes because they are assigned to the cartridges supported by the system as standard:

DB mode ØlH (ROM cartridge)
Ø2H (RAM cartridge)
HS mode Ø8H (Microcassette Drive)
Ø9H (Cartridge printer)

5.1.2 Modes and Mode Setting

The PINE cartridge interface operates in four modes: HS, IO, DB, and OT. The operation mode is switched depending on the cartridge type. Cartridges are assigned a device code identifying the cartridge type.

The relationship between the modes and the cartridges that can be used on the PINE is shown in Table 5.1.2.

Table 5.1.2 Cartridges and Modes

Cartridge	Mode	Remarks
ROM cartridge	DB	
RAM cartridge	DB	
Microcassette Drive	HS	
Cartridge printer	HS	
Modem cartridge	IO	Not supported by OS.
DMM cartridge	IO	Not supported by OS.
PROM WRITER cartridge	IO	Not supported by OS.

PINE OS checks for the presence or absence of a cartridge at power-on or reset time and selects the cartridge mode automatically.

The cartridge mode is determined by the CSEL signal (IOSTR (P16H), bit 6) and the device code (P13H, bits 7 - 4). The mode and the device code are loaded into the system area CRGDEV (ØF53FH).

Figure 5.1.3 illustrates the mode setting flow.

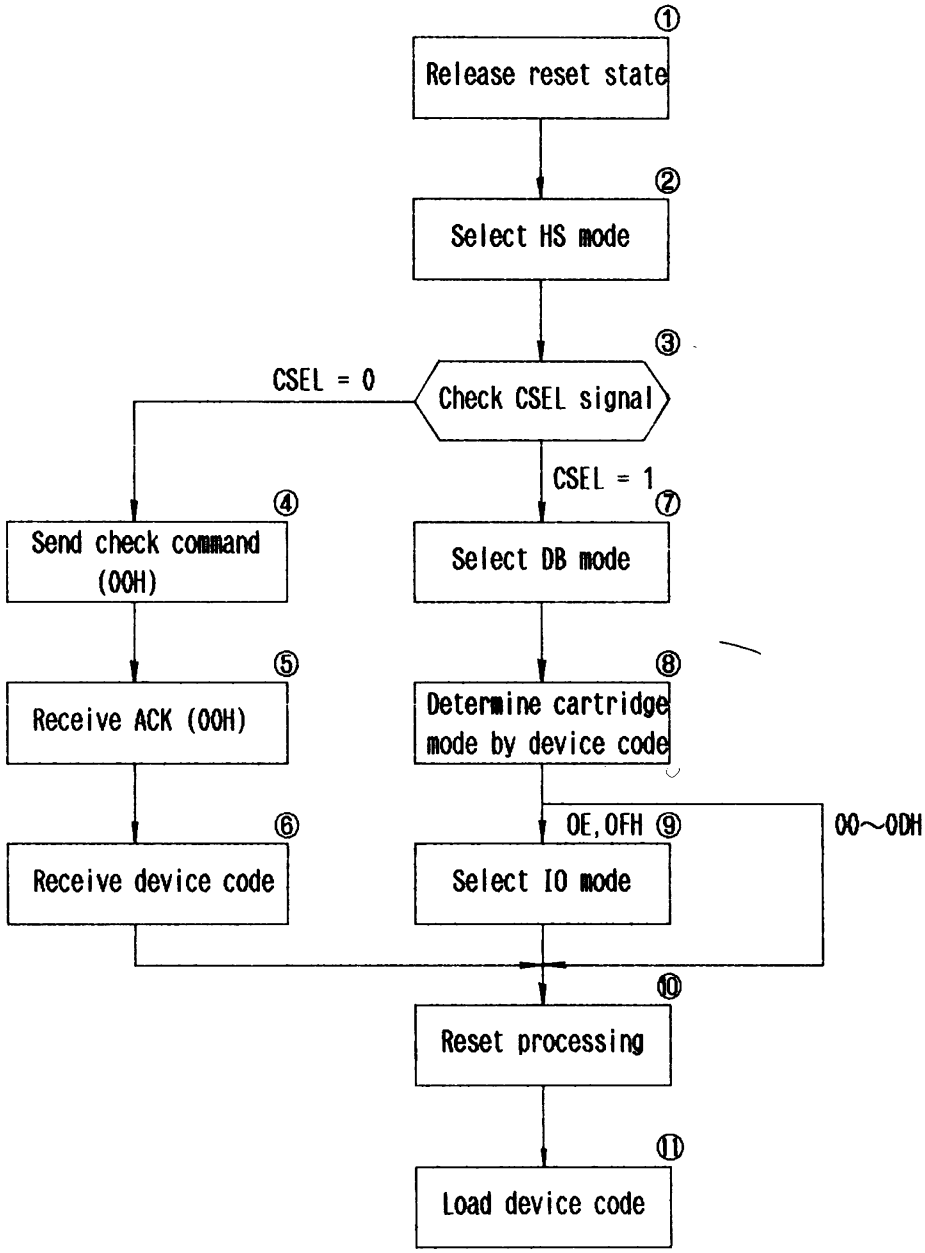


Fig. 5.1.3 Mode Setting Flow

Step 1: Release reset state.
Release the cartridge reset signal ($\overline{CRS}$) to check the identiry cartridge option. The system sets the IOCTLR $\overline{CRS}$ (P19H, bit3) high and waits approximately 10 msec.

Step 2: Select HS mode.
Set the cartridge mode to HS using the SWR CSW1,0 (P18H, bits 1,0).

Step 3: Check CSEL signal.
Check the CSEL signal to determine whether or not the cartridge mode is HS. 0 in the IOSTR CSEL (P16H, bit 6) indicates HS mode 1 indicates any other mode.

Steps 4 through 6: Identify HS mode.
Identify the HS mode option in the option following sequence:
1) Send 00H (Option Check command) to the cartridge.
2) Receive 00H (ACK) from the cartridge.
3) Receive a device code from the cartridge.
If no ACK is received from the cartridge, send the Option Check command again.

Step 7: Select DB mode.
Set the cartridge mode to DB.

Step 8: Determine cartridge mode by device code.
Read the cartridge device code from I/O port address 13H and determine the cartridge mode (other than the HS mode). The cartridge mode is identified by the high-order bits of the device code.

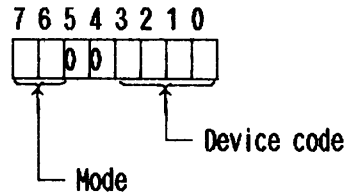
High-order bits = 00H - 0DH: DB mode
= 0EH - 0FH: IO mode

Step 9: Select IO mode.
If the device code is 0EH or 0FH, set the cartridge mode to IO.

Step 10: Reset processing.
Perform reset processing. Specify reset again based on the value of CRSTPTN (0EFC1H).

Step 11: Load device code.
Load the cartridge mode and device code into CRGDEV (0F53FH).

CRGDEV (ØF53FH) 1 byte
Cartridge device code

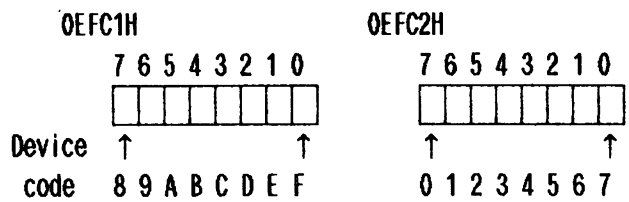


Bits		Mode
7	6	
0	0	DB mode
0	1	HS mode
1	0	IO mode
1	1	(OT mode)

Bits				Cartridge
3	2	1	0	
0	0	0	0	No cartridge
0	0	0	1	ROM cartridge
0	0	1	0	RAM cartridge
1	0	0	0	Microcassette drive
1	0	0	1	Cartridge printer
1	1	1	0	(DMM cartridge)
1	1	1	1	(Modem cartridge)

The OS does not support the OT mode, DMM, or Modem cartridge. The user can specify codes other than the device codes listed above.

CRSTPTN (ØEFC1H) 2 bytes
Specifies the cartridge reset signal ($\overline{CRS}$) to be generated after the cartridge is checked.
Each bit of CRSTPTN corresponds to a cartridge device.



A Ø indicates that the $\overline{CRS}$ is set low. A 1 indicates that $\overline{CRS}$ is set high.
The initial values are 41H and 40H.

5.1.3 ROM Cartridge

5.1.3.1 Outline

The PINE ROM cartridge is an optional read-only external storage which connects to the PINE main unit via the PINE cartridge interface. The ROM cartridge can contain up to two mask ROMs or 2764-, 27128-, or 27256-compatible EPROMs, which are installed as capsules.

The ROM cartridge runs in the DB mode. Its device code is Ø1H.

5.1.3.2 Structure

(1) ROM cartridge I/O address space

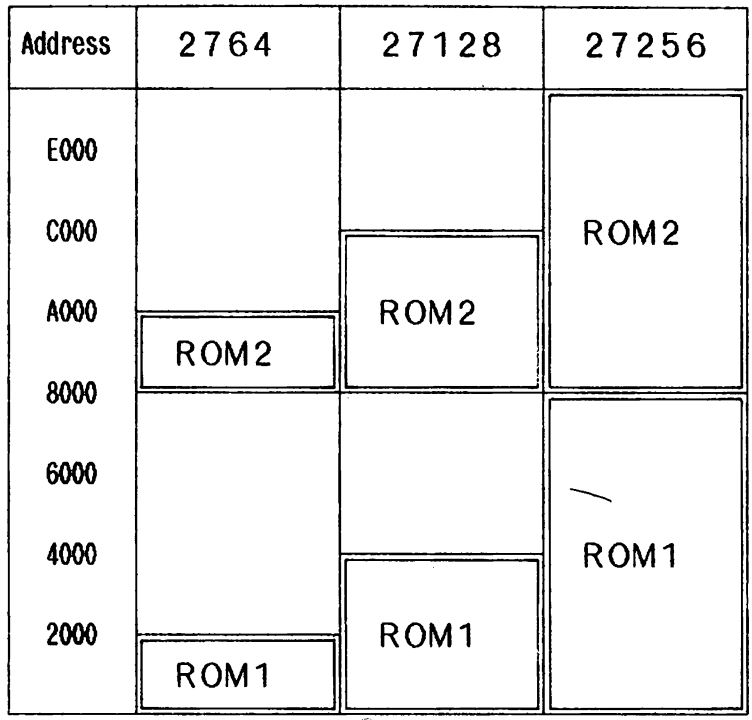
R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Remarks
Read	P10H	Not used (access inhibited)									
	P11H										
	P12H	CDBIR2	8bit data								
	P13H										Used to identify device
Write	P10H	CDBOR0	8bits data (ROM lower-order address)								
	P11H	CDBOR1	8bits data (ROM higer-order address)								
	P12H	Not used (access inhibited)									
	P13H	CDBOR3	Don't care							PWON	

CDBIR2: Loaded with the ROM data from the specified address of the ROM cartridge.
CDBORØ: Specifies the lower address of the ROM cartridge ROM.
CDBOR1: Specifies the higher address of the ROM cartridge ROM.
CDBOR3: PWON -- Specifies whether or not the ROM cartridge power is on.
= Ø: ROM cartridge power off.
= 1: ROM cartridge power on.

Note: The ROM cartridge is accessible 6Ø msec. after power-on.

(2) ROM memory map
Figure 5.1.4 shows the memory map of ROM installed in the ROM cartridge. A ROM cartridge can contain up to two 64K-, 128K-, or 256K-bit ROMs.

Fig. 5.1.4 ROM memory map



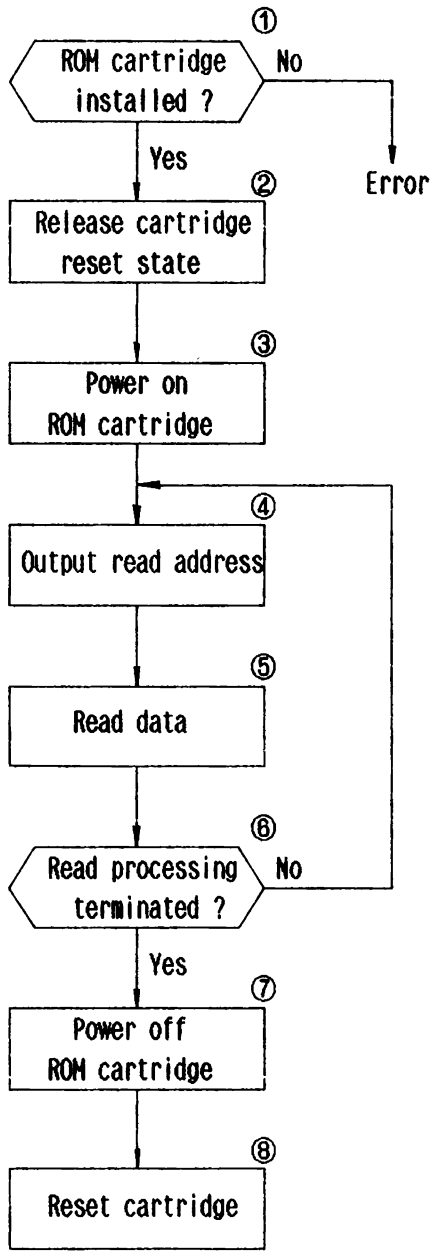
(3) Addresses on 256K-bit ROM
In 64K-bit or 128K-bit ROMs, logical ROM addresses have a one-to-one correspondence with physical ROM addresses. Therefore, data are written into ROM sequentially from lower to higher addresses. In 256K-bit ROMs, however, the first half of the logical address space is mapped into the second half of the physical address space, and the second half of the logical address space is mapped into the first half of the physical address space (see the figure below).

Address on ROM cartridge	Physical ROM address
0000H (8000H)	4000H
3FFFH (0BFFFH)	7FFFH
4000H (0C000H)	0000H
7FFFH (0FFFFH)	3FFFH

Addresses enclosed in parentheses are for ROM2.

5.1.3.3 Use

Figure 5.1.4 shows the ROM cartridge processing flow.



01H in CRGDEV (0F53FH) indicates that a ROM cartridge is installed.

Wait approximately 60msec after power-on.

Fig. 5.1.5 ROM Cartridge Processing Flow

Step 1: ROM cartridge installed?
Check whether or not a ROM cartridge is installed. CRGDEV (0F53FH) identifies the presence or absence of a ROM cartridge.

CRGDEV = 01H: ROM cartridge installed.
= Others: No ROM cartridge installed.

Step 2: Release cartridge reset state.
Reset the cartridge reset signal ($\overline{CRS}$) by setting IOCTLR (P19H), bit 6 ($\overline{CRS}$) to 1. The system waits approximately 10 ms. before actual reset.

Step 3: Power on ROM cartridge.
Power on the ROM cartridge by setting P13H, bit0 (PWON) to 1. The system waits approximately 60 ms. before actual power-on.

Steps 4 through 6: Read ROM data
Read ROM data from the specified address, following the steps below.
1) Write the lower address (8 bits) to P10H,
2) Write the higher address (8 bits) to P11H, and
3) Read ROM data from P12H.

Step 7: Power off ROM cartridge
Power off the ROM cartridge by setting P13H, bit0 (PWON) to 0. The system waits approximately 60 ms. before actual power-off.

Step 8: Reset cartridge
Reset the ROM cartridge by setitng IOCTLR (P19H), bit 6 ($\overline{CRS}$) to 0. The system waits approximately 10 ms. before actual reset.

5.1.3.4 System support

PINE OS supports the ROM cartridge as a read-only disk drive. See Section 3.8, "Disk Storage" for further information.

(1) Installation check
The OS checks whether or not a ROM cartridge is installed at power-on or reset time. Since the OS supports three types of ROM (8 KB, 16 KB, and 32 KB), it also identifies the ROM capacity by the following procedure:

- Read the header area (23 bytes from 0000H) of ROM cartridge ROM1 and check for a format mismatch.
- If no mismatch is found, calculate the ROM1 capacity from the ROM capacity specified in the header area.
- If a mismatch is found, ROM1 is not installed.
- Read and check the header area (23 bytes from 8000H) of ROM2 in the same way as with ROM1.

5.1.3.5 Miscellaneous

(1) Power control
PINE OS controls ROM cartridge power as follows:

Power-on: Turns power on if the ROM cartridge power is off when data is to be read.
Power-off: Turns power off if no data is read within two seconds after the last data was read.

Time is measured using the 7508 1-second interrupt. Power-off processing is performed by PSTBIOS.

(2) Reset ($\overline{CRS}$) signal control
Since setting or releasing of the cartridge reset state requires a wait period, a considerable amount of time would be wasted if they had to be performed at every data read. To avoid this, the OS controls resetting of the ROM cartridge in the same way as it controls ROM cartridge power.

5.1.4 RAM Cartridge

5.1.4.1 Outline

The PINE RAM cartridge is an optional external storage for read and write. It is connects to the PINE main unit via the cartridge interface.

The RAM cartridge can incorporate up to two 64K-bit static RAMs. Data on the static RAMs can be accessed through I/O instructions from the PINE main unit. The RAM cartridge is backed up by a battery and its contents are retained for up to three years, even if it is removed from the main unit.

The RAM cartridge runs in the DB mode. Its device code is Ø2H.

5.1.4.2 Structure

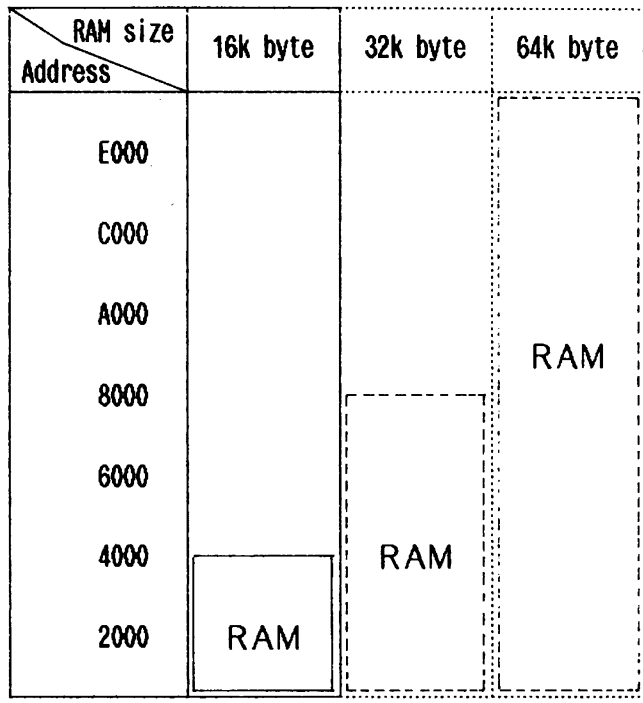
(1) RAM cartridge I/O address space

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Remarks
Read	P10H	Not used (access inhibited)									
	P11H										
	P12H	CDBIR2	8bit data								
	P13H										Used to identify device.
Write	P10H	CDBOR0	8bit data (RAM lower-order address)								
	P11H	CDBOR1	8bit data (RAM higher-order address)								
	P12H	CDBOR2	8bit data								
	P13H	Not used (access inhibited)									

CDBIR2: Loaded with the read data from the specified address of the RAM cartridge.
CDBORØ: Specifies the lower address of the RAM cartridge RAM.
CDBOR1: Specifies the higher address of the RAM cartridge RAM.
CDBOR2: Loaded with the write data from the specified address of the RAM cartridge.

(2) RAM memory map
Figure 5.1.6 shows the memory map of RAM installed in the RAM cartridge. PINE OS supports 16 KB, 32 KB, and 64 KB RAM cartridges.

Fig. 5.1.6 RAM memory map



5.1.4.3 Use

Figure 5.1.7 shows the RAM cartridge processing flow.

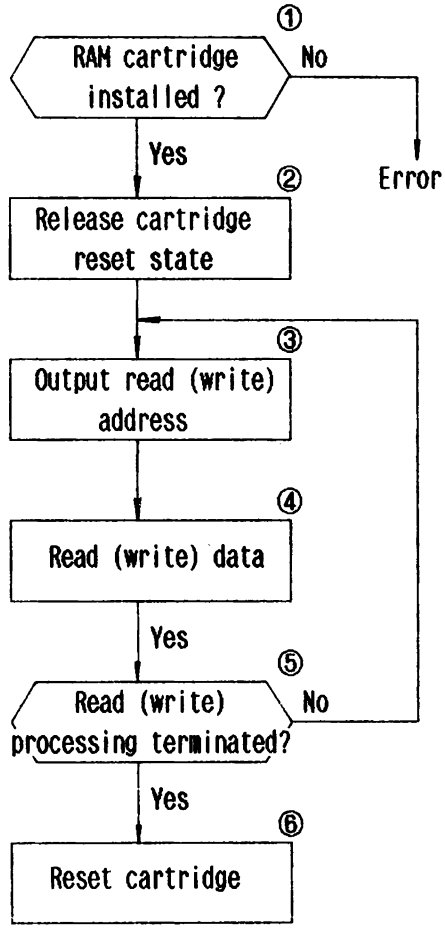


Fig. 5.1.7 RAM Cartridge Processing Flow

Note: System actions in each step are the same as for the ROM cartridge. See Subsection 5.1.3.

5.1.4.4 System support

PINE OS supports the RAM cartridge as a read/write disk drive. See Section 3.8, "Disk Storage" for further information.

(1) Installation check
The OS checks whether or not a RAM cartridge is installed at power-on or reset time. Since the OS supports three types of RAM (16 KB, 32 KB, and 64 KB), it also identifies the RAM capacity by the following procedure:

- Rewrite address 4000H of the RAM cartridge. If the attempt fails, identify the RAM cartridge capacity as 16 KB.
- Rewirte address 8000H of the RAM cartridge. If the attempt fails, idnetify the RAM cartridge capacity as 32 KB.
- If the rewrite of address 8000H was successful, identify the RAM capacity as 64 KB.

However, only 16 KB RAM cartridges are available as products.

5.1.4.5 Miscellaneous

(1) Reset ($\overline{CRS}$) signal control
Since setting or releasing of the cartridge reset state requires a wait period, a considerable amount of time would be wasted if they had to be performed at every data read. To avoid this, the PINE OS controls resetting of the RAM cartridge as follows:

Releasing reset state: Release the reset state if the RAM cartridge is reset when data is to be read or written.
Setting reset state: Reset the cartridge if no data is read or written within two seconds after the last read or write.

Time is measured using the 7508 1-second interrupt. Reset state setting processing is performed by PSTBIOS.

5.1.5 Microcassette Drive

5.1.5.1 Outline

The PINE microcassette drive (MCT) ia an optional external storage for read and write. It connects to the PINE main unit via the PINE cartridge interface.

The microcassette drive is an intelligent cartridge option incorporating CMOS MCU 6301V1 for controlling. It runs in the HS mode. device code is 08H.

5.1.5.2 Structure

(1) Microcassette drive I/O address space

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Flag state changed by
Read	10H	CHSIR	8bit command/data								IBF reset
	11H	CHSSR						CAB1	IBF	OBF	
	12H	Not used (access inhibited)									
	13H										
Write	10H	CHSOR	8bit data								OBF set
	11H	CHSOR	8bit command								OBF set
	12H	Not used (access inhibited)									
	13H										

CHSIR: Cartridge HS mode input register
(CAB1=0 identifies data entry. CAB1=1 identifies command entry.)
CHSSR: Cartridge HS mode status register
CHSOR: Cartridge HS mode output register.

5.1.5.3 Use

The microcassette drive interfaces to the PINE main unit through commands, data, and the ACK signal.

The PINE main unit controls the maicrocassette drive by sending commands or commands and parameters (data) to the microcassette drive via the HS interface. An ACK signal, or data and an ACK signal, corresponding to the commands is returned to the main unit during or after operation.

The following routines are provided to manipulate command and data transmission between the main unit and the HS mode cartridge:

HSRST
HSSCOM
HSSDAT
HSRCV
HSSBL
HSRBL
HSBRK

See Section 4.2, "Jump Tables" for details.

(1) Protocol

The PINE uses the protocol shown in Figure 5.1.8. Every communication, as viewed from the main unit, starts with transmission of a command (for data transmission), followed by reception of an ACK signal (for data reception). During tape reads, however, data is received before an ACK signal. If these procedures are not followed, a protocol error occurs during processing of commands other than the break or cold start commands.

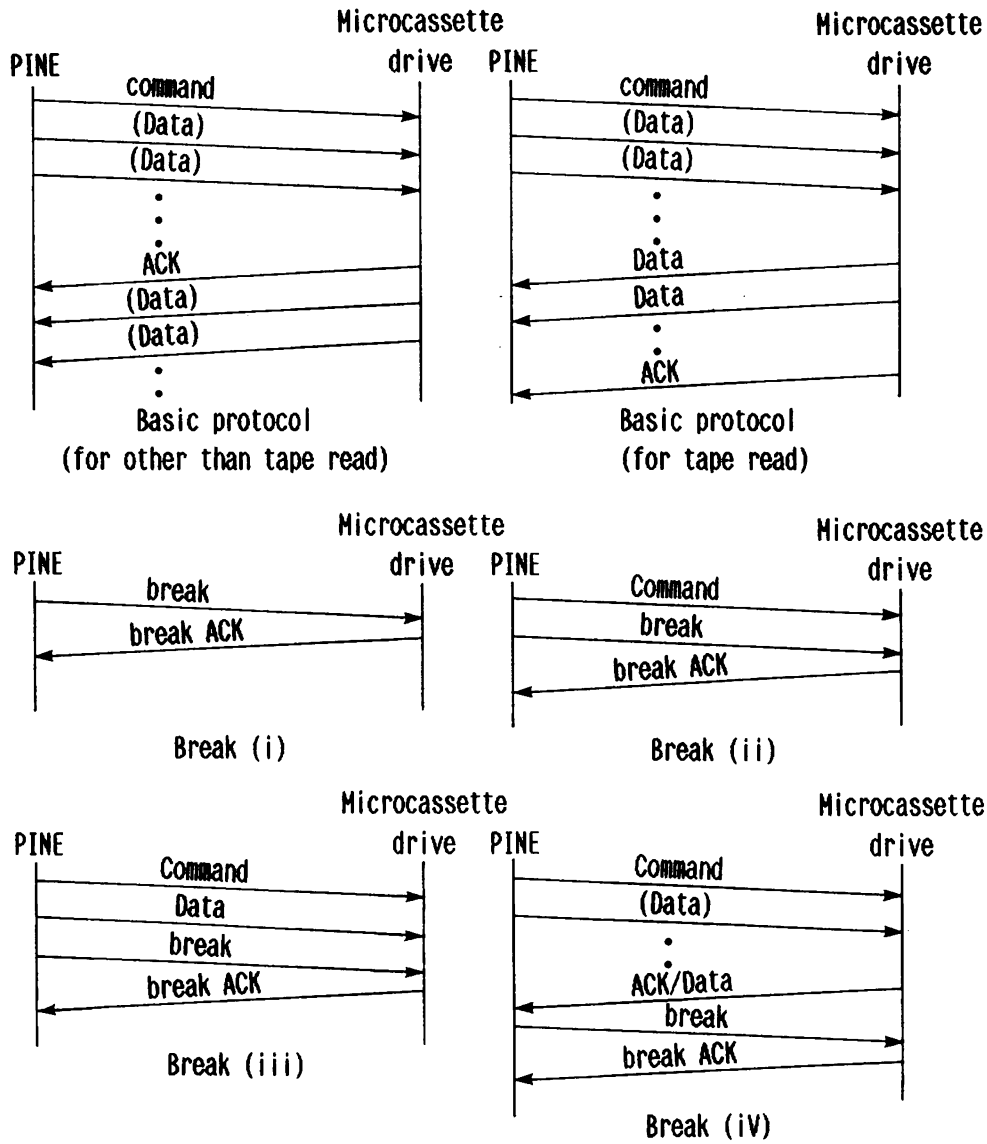


Fig. 5.1.8 Protocol

(2) Command directory

Code	Function	Page
00H	Read drive code of installed option	II-518
01H	Read monitor data	II-518
02H	Write monitor data	II-519
03H	Spare	II-519
0BH	Cold start	II-519
0FH	Break	II-519
40H	Read microcassette status	II-519
41H	Turn head on	II-520
42H	Turn head off	II-520
43H	Rewind n count	II-520
44H	Fast feed n count	II-521
45H	Rewind	II-521
46H	Fast feed	II-521
47H	Slow rewind	II-522
48H	Slow wind	II-522
49H	Turn on erase and record head and wind slow	II-522
4AH	Stop capstan motor	II-522
4BH	Read write protect tab status	II-523
4CH	Read counter	II-523
4DH	Set counter	II-523
4EH	Spare	II-523
4FH	Spare	II-523
50H	Verify (CRC check)	II-524
51H	Write in nonstop mode	II-524
52H	Write in stop mode	II-525
53H	Read in nonstop mode	II-525
54H	Read in stop mode	II-526
55H	Set rewind count	II-526
56H	Release rewind count	II-526
57H	Set interblock gap length	II-526
58H	Set bit rate	II-527
59H	Enable/disable CRC check	II-527
5AH	Turn on/off tape removable LED	II-528
5BH	Send internal-RAM data to main unit	II-528
5CH	Write data to internal RAM	II-528
5DH	Spare	II-528
5EH	Spare	II-528
5FH	Self diagnosis	II-529

(3) Command details
The following pages list the functions and parameters of the microcassette commands. Send parameters are sent from the main unit to the microcassette drive ACK and receive parameters are returned from the microcassette drive to the main unit.

Return codes shown in the ACK field have the following meanings:

ACK code	Value(hex)	Description
RCD00	00	Normal termination
RCD01	01	Cold start executed
RCD02	02	Break executed
RCD03	03	Command error
RCD04	04	Protocol error
RCD05	05	FIFO full (First in First out buffer)
RCD06	06	FIFO empty
RCD07	41	Head error (time over)
RCD08	42	Tape stop error or no tape error
RCD09	43	Write protect error
RCD10	44	Data error (time over)
RCD11	45	CRC error
RCD12	46	(not used)
RCD13	47	Protected area encountered
RCD14	07	Monitor write inhibited

(a) Read device code of Installed Option
- Commadn code: ØØH
- Send parameter: None
- ACK: RCDØØ
- Receive parameter: Device code (Ø8H) of the microcassette drive

Function:
Reads the device code of the optional device installed. No IRQ1 interrupts are accepted during processing.

(b) Monitor commands
1) Read Monitor Data
- Commadn code: Ø1H
- Send parameters:
  SP1: Address at which read is to start (low-order)
  SP2: Address at which read is to start (high-order)
  SP3: Read n data bytes ($1 \leq n \leq 4Ø$)
- ACK: RCDØØ
- Receive parameters:
  RP1: Data 1
  .
  .
  .
  .
  RPn: Data n

Function:
Reads the number of bytes specified in SP3 (maximum of 4Ø bytes) from the address specified in SP1 and SP2, and sends them to the main unit.

*1 During execution of processing, the sending of commands other than break and cold start, results in protocol error and abnormal end (RCD04).
In the case of break, processing is halted, and break ACK (RCD02) is returned. In the case of cold start, the following processes are executed, and a return made to the initialization state (reset state).
1. Motor stop
2. Head OFF
3. Reception of ACK (RCD01)

2) Write Monitor Data
- Commadn code: 02H
- Send parameters:
SP1: Address at which write is to start (low-order)
SP2: Address at which write is to start (high-order)
SP3: Write n data bytes (n=0: 256 bytes, 1: 1 byte, ...)
SP4: Write data 1
.
.
SPn+3: Write data n
- ACK: RCD00
RCD14 (An attempt was made to write to an address outside the range of 80H - 0FFH.)
- Receive parameter: None
Function:
Writes data SP4 through SPn+3 to the address specified in SP1 and SP2. An error occurs if an attempt is made to write to an address outside the range of 80H to 0FFH.
For commands accepted during execution refer to 1) Read Monitor Data *1.

3) Spare
- Commadn code: 03H
- Send parameter: None
- ACK: RCD03
- Receive parameter: None

(c) Cold Start
- Command code: 0BH
- Send parameter: None
- ACK: RCD01
- Receive parameter: None
Function:
Initializes the microcassette drive and the MCU6301.

(d) Break
- Command code: 0FH
- Send parameter: None
- ACK: RCD02
- Receive parameter: None
Function:
Terminates microcassette drive processing and puts the device into the command mode.

(e) Microcassette commands
1) Read Microcassette Status
- Command code: 40;H
- Send parameter: None
- ACK: RCD00

- Receive parameter:
    RP1: Microcassette status (MCSTS)
        Bit 7: Head on (Ø: No, 1: Yes)
        Bit 6: Motor rotate (Ø: No, 1: Yes)
        Bit 5: Fast feed backward (REW) (Ø: No, 1: Yes)
        Bit 4: Fast feed forward (FF) (Ø: No, 1: Yes)
        Bit 3: Slow feed forward (play) (Ø: No, 1: Yes)
        Bit 2: Slow feed backward (Ø: No, 1: Yes)
        Bit 1: Erase (Ø: No, 1: Yes)
        Bit Ø: Feed forward enable flag set (Ø: No, 1: Yes)

Function:
Reads the microcassette status.
For commands accepted during execution refer to (b) Monitor Command.

2) Turn Head On
- Command code: 41H
- Send parameter: None
- ACK: RCDØØ,
      RCDØ7 (Head error: Time over)
- Receive parameter: None

Function:
When the head is off the tape, this command drives the head motor and places the read/write head on the tape. When the head is already on the tape, the command does nothing but returns an ACK.

An head error occurs if the specified head state is not established within 1.6 seconds after this command is issued. No interrupts are accepted untill the operation is completed. The command sets the polarity of the input edge. The default setting is from positive (H) to negative (L). The speaker audio output is held off while the head motor is running.

3) Turn Head Off
- Command code: 42H
- Send parameter: None
- ACK: RCDØØ
      RCDØ7 (Head error: Time over)
- Receive parameter: None

Function:
This command performs the same function as the Turn Head On command except that it turns off the head if it is on, and it does nothing but return an ACK if the head is already off.

4) Rewind n Counts
- Command code: 43H
- Send parameters:
    SP1: Counter value (high-order)
    SP2: Counter value (low-order)
    SP3: Head on/off (Ø: OFF, Nonzero: ON)
- ACK: RCDØØ
      RCDØ7 (Head error)
      RCDØ8 (Tape stop or no tape error)
      RCD13
- Receive parameter: None

Function:
Moves the tape backward the number of tape counts specified in SP1 and SP2. If the capstan motor is rotaing forward and the head is on when the command is received, the command rewinds the tape after stoppig the motor and turning off the head.

If a nonzero value is specified in SP3, the command turns the head on after finisning rewinding.

A no-tape error occurs after approximately one second if no tape is mounted. A tape stop error occurs if the tape end is reached before tape is rewound by n counts.
For commands accepted during execution refer to (b) M-nitor Command *1.

5) Fast Feed n Counts
- Command code: 44H
- Send parameters:
  SP1: Counter value (high-order)
  SP2: Counter value (low-order)
  SP3: Head on/off (Ø: OFF, Nonzero: ON)
- ACK: RCDØØ
  RCDØ7 (Head error)
  RCDØ8 (Tape stop error)
  RCD13
- Receive parameter: None

Function:
Moves the tape forward the number of tape counts specified in SP1 and SP2 at a high speed.

If the head is on and the capstan motor is rotating backward when this command is issued, the command feeds tape after stopping the motor and turning off the head. If a nonzero value is specified in SP3, the command turns the head on after finisning the operation.

Even when no tape is mounted, the command executes as if one were and terminates normally. A tape stop error occurs if the tape end is reached before the tape is wound by n counts.
For commands accepted during execution refer to (b) Monitor Command *1.

6) Rewind
- Command code: 45H
- Send parameter: None
- ACK: RCDØØ
  RCDØ7 (Head error)
- Receive parameter: None

Function:
Rotates the motor in the reverse direction (rewind). If the motor is rotating forward or the head is on when this command is received, the command stops the motor or turns off the head before rewinding.

7) Fast Feed
- Command code: 46H
- Send parameter: None
- ACK: RCDØØ
  RCDØ7 (Head error)
- Receive parameter: None

Function:
Rotates the motor forward at a high speed. If the motor is rotating backward or the head is on when this command is received, the command stops the motor or turns off the head before rotating the motor fast forward.

The command drives the motor normally even if no tape is mounted but automatically stops it when the tape end is reached.

8) Slow Rewind
- Command code: 47H
- Send parameter: None
- ACK: RCDØØ
  RCDØ7 (Head error)
- Receive parameter: None
Function:
Rotates the motor in the reverse direction (rewind) at a low speed. If the motor is rotating forward or the head is on when this command is received, the command stops the motor or turns off the head before rotating the motor in reverse.

The command automatically stops the motor approximately one second after a no tape error or tape stop error is detected.

9) Slow Wind
- Command code: 48H
- Send parameter: None
- ACK: RCDØØ
- Receive parameter: None
Function:
Rotates the motor forward at a low speed. If the motor is rotating backward when this command is received, the command stops the motor before rotaing it forward.

When the tape end is reached, the command automatically stops the motor and turns off the head after approximately one second. When the head is on, the read signal is transmitted from the head output line.

1Ø) Turn On Erase or Record Head and Wind Slow
- Command code: 49H
- Send parameter: None
- ACK: RCDØØ
- Receive parameter: None
Function:
If the microcassette write protect tab is broken, or if a cassette has not been inserted RCDØ9 is immediately returned. If the microcassette write protect tab is not broken or the cassette holder is open, if the motor is rotating backward when this command is received, the command stops the motor and rotates it forward.

Data already on the tape will be erased if the read/write head is on. Data will be written on the tape if signals are output to the record head.

It automatically stops the motor and turns off the head approximately one second after the tape end is reached.

11) Stop Capstan Motor
- Command code: 4AH
- Send parameter: None
- ACK: RCDØØ
- Receive parameter: None
Function:
Stops the motor and, Ø.6 seconds later, stops the counter. This action does not change the head position.

12) Read Write-Protect Tab Status
- Command code: 4BH
- Send parameter: None
- ACK: RCD00
- Receive parameter:
    RP1: Write-protect tab status
        00H: Write protected (tab removed)
        08H: Write enabled

Function:
Used to check whether the current microcassette tape is write protected.

13) Read Counter
- Command code: 4CH
- Send parameter: None
- ACK: RCD00
- Receive parameters:
    RP1: Counter value (high-order)
    RP2: Counter value (low-order)

Function:
Reads the current counter value.

14) Set Counter
- Command code: 4DH
- Send parameters:
    SP1: Counter value (high-order)
    SP2: Counter value (low-order)
- ACK: RCD00
- Receive parameter: None

Function:
Sets the counter to the value specified in SP1 and SP2.

15) Spare
- Command code: 4EH
- Send parameters:
    SP1, SP2
- ACK: RCD03
- Receive parameter: None

16) Spare
- Command code: 4FH
- Send parameters:
    SP1, SP2, SP3
- ACK: RCD03
- Receive parameter: None

17) Verify (CRC check)
- Command code: 50H
- Send parameters:
    SP1: Number of data bytes to be verified (high-order)
    SP2: Numver of data bytes to be verified (low-order)
- ACK: RCD00
       RCD07
       RCD08
       RCD10
       RCD11
       RCD13
- Receive parameters: Same as for the Read in Nonstop Mode command (53H).

Function:
Reads the number of data bytes specified in SP1 and SP2 in the stop mode (stops at every block end) starting at the current tape position, checks for CRC, and sends the whole data to the main unit.

The read signal is sent to the speaker during verify processing. For commands accepted during execution refer to (b) Monitor Command *1.

18) Write in Nonstop Mode
Command code: 51H
- Send parameters:
    SP1: Data length (high-order)
    SP2: Data length (low-order)
    SP3: Data 1
    SP4: Data 2
      .
      .
    SPn+2: Data n
- ACK: RCD00
       RCD07
       RCD08
       RCD09
       RCD13
- Receive parameters:
    RP1: Block start counter (high-order)
    RP2: Block start counter (low-order)

Function:
Writes the number of data bytes specified in SP1 and SP2 onto tape. Data blocks shorter than 40 bytes cannot be written. Data bytes are sent at intervals of up to 5 ms/byte.

When the same data is to be written n times, it must be sent from the main unit to the microcassette drive n times. Since the microcassette writes the received data as is, without checking, the main unit must send the correct data, checing block numbers.

The tape does not stop after the last write is completed. It does stop when an error occurs.

The head is turned off when an RCD08, RCD09, or RCD13 error occurs.

An ACK is returned from the microcassette before RP1 and RP2. If the command is sent before the sending of specified byte data to the microcassette drive has been completed, a protocol error occurs. However break and cold start commands are accepted and respective processing carried out.

/w.fastio.com

19) Write in Stop Mode
- Command code: 52H
- Send parameters:
  SP1: Data length (high-order)
  SP2: Data length (low-order)
  SP3: Data 1
  SP4: Data 2
  •
  •
  SPn: Data n-2
- ACK: RCD00
  RCD07
  RCD08
  RCD09
  RCD13
- Receive parameters:
  RP1: Block start counter (high-order)
  RP2: Block start counter (low-order)

Function:
This command performs the same function as Write in Nonstop Mode except that it stops the tape after writing data.

20) Read in Nonstop Mode
- Command code: 53H
- Send parameters:
  SP1: Data length (high-order)
  SP2: Data length (low-order)
- ACK: RCD00
  RCD05
  RCD07
  RCD08
  RCD10
  RCD11
  RCD13
- Receive parameters:
  RP1: Counter value set when a block synchronous bit is detected (high-order)
  RP2: Counter value set when a block synchronous bit is detected (low-order)
  RP3: Data 1
  •
  •
  RPn: Data n-2

Function:
Reads the number of data bytes specified in SP1 and SP2, starting at the current tape position.

An FIFO full error occurs if the FIFO becomes full before reading is completed. If the specified number of data bytes are read with no errors, the command terminates reading without stopping the motor. If an error occurs during processing, it stops the motor and turns off the head. Data bytes are sent at intervals of up to 5 ms per byte.

The read signal is sent to the loudspeaker during processing. An ACK is sent to the main unit after transmission of data is completed.
For commands accepted during execution refer to (b) Monitor Command *1.

21) Read in Stop Mode
- Command code: 54H
- Send parameters:
    SP1: Data length (high-order)
    SP2: Data length (low-order)
- ACK: RCD00
       RCD05
       RCD07
       RCD08
       RCD10
       RCD11
       RCD13
- Receive parameters:
    RP1: Counter value set when a block synchronous bit is detected (high-order)
    RP2: Counter value set when a block synchronous bit is detected (low-order)
    RP3: Data 1
     .
     .
    RPn: Data n-2

Function:
This command has the same function as Read in Nonstop Mode except that it stops the motor after reading data.

22) Set Rewind Count
- Command code: 55H
- Send parameters:
    SP1: Counter value (high-order)
    SP2: Counter value (low-order)
- ACK: RCD00
- Receive parameter: None

Function:
Sets the maximum count by which the tape can be rewound from the current position as well as the flag for enabling the rewind count. When the flag is set, the tape stops and the head turns off automatically if the rewind count reaches 0.

23) Reset Rewind Count
- Command code: 56H
- Send parameter: None
- ACK: RCD00
- Receive parameter: None

Function:
- Resets the rewind count flag.

24) Set Interblock Gap Length
- Command code: 57H
- Send parameters:
    SP1: Short interblock gap length SBIGIN
    SP2: Long interblock gap length LBIGIN
- ACK: RCD00
- Receive parameter: None

Function: Sets the short and long interblock gap lengths.

25) Set Bit Rate
- Command code: 58H
- Send parameters:
  SP1: 1 bit half period (high-order) B1TIME
  SP2: 1 bit half period (low-order)
  SP3: Ø bit half period (high-order) BØTIME
  SP4: Ø bit half period (low-order)
  SP5: Bit threshold level (high-order) MCTTHR
  SP6: Bit threshold level (low-order)
  SP7: Noise threshold level (high-order) MCTØTH
  SP8: Noise threshold level (low-order)
- ACK: RCDØØ
- Receive parameter: None

Function:
Sets the half period of 1 and Ø bits during writes, the bit threshold level during reads, and the noise threshold level.

The half periods of 1 and Ø bits are obtained by multiplying the MCU6301V1 machine cycle by (SP1, SP2) and (SP3, SP4), respectively. Since the machine cycle for the PINE is 1 μs, each parameter value equals the corresponding half period. The default values are 500 μs and 250 μs, respectively. The upper and lower limits are as follows:

	Upper limit (μs)	Lower limit (μs)
1 bit	750	500
Ø bit	375	250

The bit threshold level defaults to 750 μs. When redefining the bit threshold level with this command, set it to the arithmetic or geometrical mean valueo f the periods of the 1 and Ø bits.

It is desirable that the noise threshold level be set to the Ø bit half period. The default value is 250 μs.

26) Enable/Disable CRC Checking
- Command code: 59H
- Send parameters:
  SP1: Ø (Disable), Nonzero (Enable)
  SP2: CRC multinomial expression (high-order)
  SP3: CRC multinomial expression (low-order)
- ACK: RCDØØ
- Receive parameter: None

Function:
Suppresses generation of a CRC check bit during reads when SP1 is set to Ø. Any CRC multinomial expression can be specified when SP1 is a nonzero value. The default value is CRC-CCITT. Once an expression is specified, it is not restored to the default value until a cold start is performed.

SP2 and SP3 must be sent even when CRC checking is disabled in SP1.

27) Turn ON/OFF Tape Removable LED
- Command code: 5AH
- Send parameter:
  SP1: Ø (OFF), Nonzero (ON)
- ACK: RCDØØ
- Receive parameter: None

Function
Turns off the green LED when SP1 is Ø and turns it on when SP1 is 1. When this LED is on, it indicates that a microcassette drive is loaded, but not mounted, in the PINE ccartridge space.

28) Send Internal-RAM Data to Main Unit
- Command code: 5BH
- Send parameter: None
- ACK: RCDØØ
  RCDØ2
  RCDØ4
- Receive parameters:
  RP1: Data 1
  RP2: Data 2
  .
  .
  RP32: Data 32

Function:
Transfers internal-RAM data to the main unit. The data length is fixed at 32 bytes.
For commands accepted during execution refer to (b) Monitor Command *1.

29) Write Data to Internal RAM
- Command code: 5CH
- Send parameters:
  SP1: Data 1
  SP2: Data 2
  .
  .
  SP32: Data 32
- ACK: RCDØØ
  RCDØ4 (Protocol error)
  RCDØ2 (Break)
- Receive parameter: None

Function:
Writes data into internal RAM. The data length is fixed at 32 bytes.
For commands accepted during execution refer to (b) Monitor Command *1.

3Ø) Spare
- Command code: 5DH
- Send parameter: None
- ACK: RCDØ3
- Receive parameter: None

31) Spare
- Command code: 5EH
- Send parameter: None
- ACK: RCDØ3
- Receive parameter: None

32) Self Diagnosis
- Command code: 5FH
- Send parameter: None
- ACK: RCD00, RCD07, RCD08, RCD09
- Receive parameter: None

Function:
Performs self diagnosis on the microcassette drive by the following procedure:

1. Release the FF (Feed Forward enable) area.
2. Rewind tape to the tape start.
3. Reset the counter.
4. Turn on the green LED.
5. Set the FF area (30H).
6. Wait in the head-on/play mode until the drive stops.
7. Reset the FF area.
8. Turn off the green LED.
9. Feed tape forward by 200H counts.
10. Set the FF area (30H)
11. Wait in the head-on/erase mode until the drive stops.
12. Set the FF area (100H).
13. Write a block in the stop mode.
14. Set the FF area (100H).
15. Write a block in the stop mode.
16. Set the FF area (30H).
17. Wait in the head-on/erase mode until the drive stops.
18. Reset the FF area.
19. Rewind tape to the tape start.
20. Turn on the green LED.
21. Feed tape forward by 260H counts.
22. Set the FF area (30H).
23. Wait in the head-on/erase mode until the drive stops.
24. Reset the FF area.
25. Rewind tape to the tape start.
26. Reset the counter.
27. Flash the green LED on and off.

This command imediately returns an ACK when an error occurs.

Break processing

The PINE main unit can put the microcassete drive into the command mode by sending a break or cold start command to it.

A break command interurpts current command processing, and puts the microcassette drive into the command mode. A cold start command interrupts current command processing, and resotres the microcassette drive and RAM to their initial settings. The actual content of break processing differs depending on the current microcassette drive status. Break processing in individual cases is described below.

1) Command mode
When receiving a break command in this mode, the microcassette drive returns an ACK and enters the command mode again. If the motor and head are on, the microcassette drive turns them off.

Whe receiving a cold start command in this mode, the microcassette drive returns an ACK and enters the command mode again. The microcassette drive and RAM are restored to their initial settings*.

2) During data reception ** (except during writes)
A break command received during data reception is invalid; the microcassette drive does not accept the subsequent data, but instead returns an ACK and enters the command mode. If the motor and head are on, the microcassette drive turns them off.

When receiving a cold start command during data reception, the microcassette drive returns an ACK and enters the command mode. The microcassette drive and RAM are restored to their initial settings.

3) During data transmission ** (except during reads)
When receiving a break command during data transmission, the microcassette drive does not transmit the subsequent data, but instead returns an ACK and enters the command mode.

When receiving a cold start command during data transmission, the microcassette drive returns an ACK and enters the command mode. The microcassette drive and RAM are restored to their initial settings. Since an ACK has already been sent, the main unit must read another ACK.

*: "Initial settings" refers to the following:
Power, head, LEDs (green, red), capstan motor -- OFF
FIFO -- Empty
RAM data = ROM data
Work area -- Cleared
**: "Transmission" and "reception" are from the viewpoint of the microcassette drive.

4) Between termination of command processing and transmission of an ACK
When receiving a break command in this state, the microcassette drive returns an ACK and enters the command mode. Any data to be transmitted is ignored. If the motor and head are on, the microcassette drive turns them off.

When receiving a cold start command in this state, the microcassette drive returns an ACK and enters the command mode again. The microcassette drive and RAM are restored to their initial settings.

5) During command processing
When receiving a break command, the microcassette drive operates differently depending on the command being currently executed.
- When the Fast feed n count, read, or write commands are in execution:
The microcassette drive terminates processing, returns an ACK, and enters the command mode. If the motor and head are on, it turns them off.
- When a command other than the above is in execution:
The microcassette drive terminates prodessing and accepts the break command (the subsequent operation is the same as described in 4) above.)

Note: After execution of a break command, the register contents are retained but FIFO is emptied.

5.1.5.4 System support

PINE OS supports the microcassette drive as a disk drive for read and write. See Section 3.7, "MTOS/MIOS Operations" for further information.

- Break processing
Upon receiving a request to terminate microcassette processing, PINE OS completes the current operation, and then interrupts processing by sending a break command to the microcassette drive.

Microcassette processing is interrupted when:
- The CTRL/STOP keys are pressed.
- The power switch is turned off or a power failure occurs.
- An alarm or wake condition occurs.

5.1.6 Cartridge Printer

5.1.6.1 Outline

The PINE cartridge printer is a mechanical dot printer which connects to the PINE main unit via the PINE cartridge interface. The cartridge printer employs an M-164 micro-dot printer, which can print 40 characters (6-by-8 font) per line. The cartridge printer is furnished with a print buffer of 2 K bytes (8 lines)

The cartridge printer runs in the HS mode. Its device code is 09H.

5.1.6.2 Structure

(1) Block diagram

Figure 5.1.9 shows the block diagram for the cartridge printer.

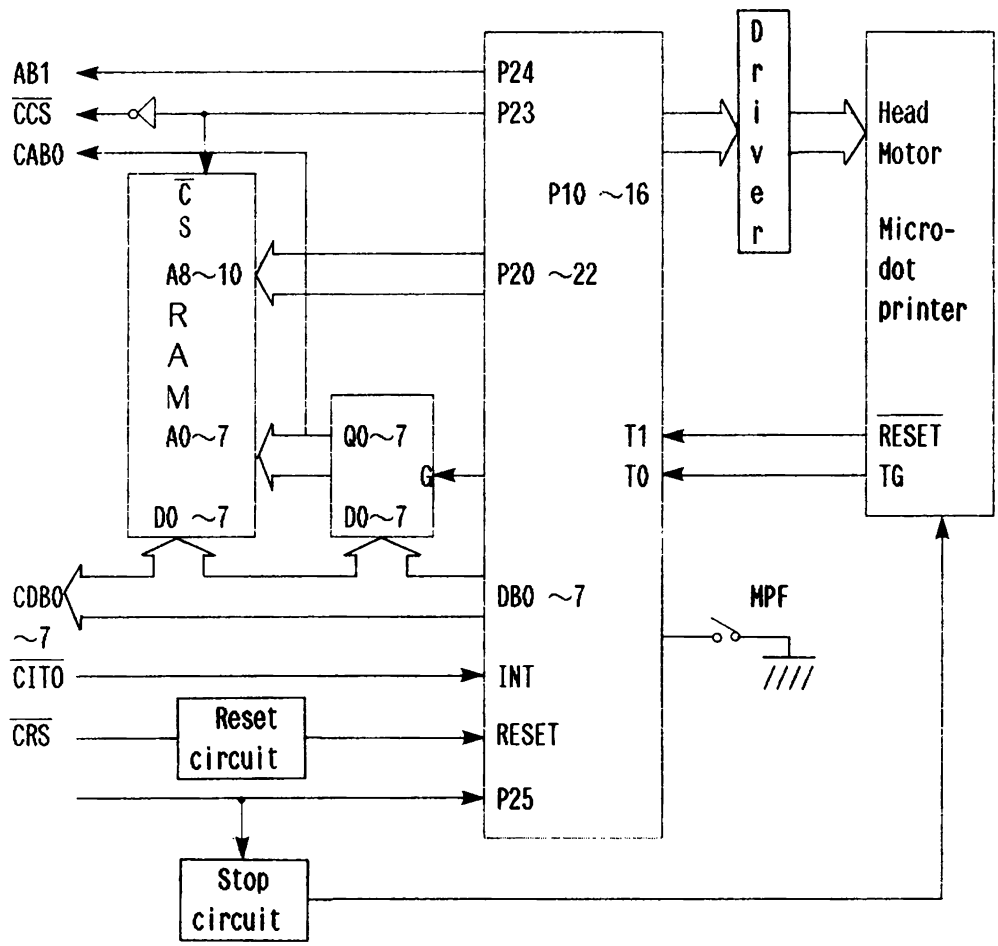


Fig. 5.1.9 Cartridge Printer Block Diagram

(2) Cartridge printer I/O address space

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Flag state changed by
Read	P10H	CHSIR	8bits data								IBF reset
	P11H	CHSSR						CAB1	IBF	OBF	
	P12H	Non used (access inhibited)									
	P13H										
Write	P10H	CHSOR	8bits data								OBF set
	P11H	CHSOR	8bits command								OBF set
	P12H	Non used (access inhibited)									
	P13H										

CHSIR: Cartridge HS mode input register
CHSSR: Cartridge HS mode status register
(CAB1=0: Printer ready, CAB1=1: Printer busy)
CHSOR: Cartridge HS mode output register

(3) Power supply
The cartridge printer draws power from the PINE main unit via a connector. To protect the main unit from heavy voltage fluctuations during printing, the cartridge printer is desingned to stop immediately when the voltage drops below approximately 3.5 V. If the printer stops for this reason, it accepts nothing but an ID check or break command.

5.1.6.3 Use

The PINE main unit determines whether of not to send data to the printer by examining CAB1 (printer status flag).

Figure 5.1.10 shows the cartridge printer processing flow.

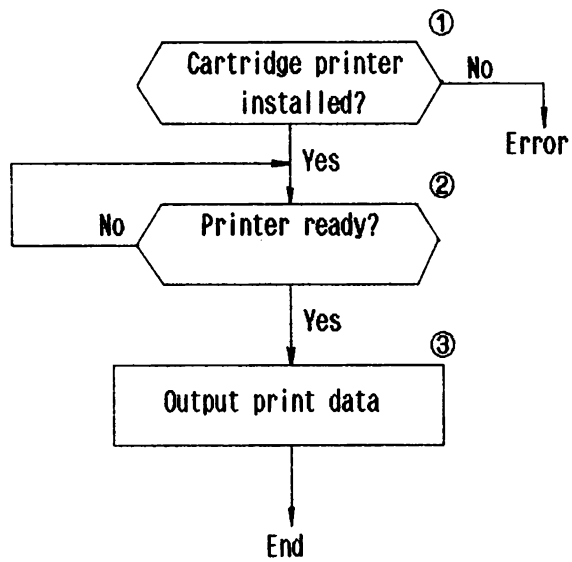


Fig. 5.1.10 Cartridge Printer Processing Flow

Step 1: Cartridge printer installed?
Check whether or not a cartridge printer is installed.
CRGDEV (ØF53FH) indicates the presence or absence of a cartridge printer.
CRGDEV = 49H: Cartridge printer installed.
= Other than 49H: No cartridge printer installed.

Step 2: Printer ready?
Determine whether or not the cartridge printer is ready by examining bits 2 (CABl) and Ø (OBF) of CHSSR (PllH).
CABl = Ø: Ready
= 1: Busy
OBF = Ø: Output buffer empty
= 1: Output buffer full
Output to the printer is allowed only when both CABl and OBF are Ø.

Step 3: Output print data.
Output print data to CHSOR (PlØH).

5.1.6.4 System support

PINE OS supports the cartridge printer as a list output device using the BIOS LIST, LISTST, and SCRNDUMP functions. See Section 3.4, "BIOS Details" for individual functions.

(1) Break processing
Upon receiving a request to terminate cartridge printer processing, PINE OS immediately terminates procsssing by by sending a break command to the cartridge printer.

Cartridge printer processing is terminated when:
- The CTRL/STOP keys are pressed.
- The power switch is turned off or a power failure occurs.

(2) Commands
The cartridge printer accepts ID check and break commands sent from the PINE main unit.

1) ID check (ØØH)
On receiving the command ØØH from the main unit, the cartridge printer returns ØØH as an ACK, followed by the device code Ø9H.

2) Break (ØFH)
On receiving the command ØFH from the main unit, the cartridge printer returns Ø2H as an ACK and initializes all states except the universal character mode.

(3) Control codes
Table 5.1.11 lists the control codes for the cartridge printer.

Table 5.1.11 Printer Control Codes

Control code	Description
STX (Ø2H)	Enable double height mode.
ETX (Ø3H)	Disable double height mode.
BS (Ø8H)	Back space.
LF (ØAH)	Line feed.
CR (ØDH)	Carriage return.
SØ (ØEH)	Enable double width mode with auto reset.
DC4 (14H)	Disable double width mode with auto reset.
DEL (7FH)	Delete.
ESC (1BH)	Start ESC sequence.
ESC 1 (31H)	Set paper feed to 8 dots.
ESC 2 (32H)	Set paper feed to 1Ø dots.
ESC A+n (41H+n)	Set paper feed to n dots.
ESC R+n (52H+n)	Enable universal character mode.
ESC K+n1+n2 (4BH+n1+n2)	Specify single density bit image.
ESC W+Ø (57H+Ø)	Disable double width mode.
ESC W+1 (57H+1)	Enable double width mode.

5.1.7 Modem/DMM Cartridge

5.1.7.1 Outline

The PINE modem cartridge is an optional modem unit which has the direct modem, acoustic coupler, and auto dialing functions, and allows communication through a handset.

The PINE DMM cartridge is an intelligent measuring unit incorporating the A/D conversion IC and 4-bit COMS CPU.

Since PINE OS supports neither modem nor DMM cartridge, they must be interfaced to the main unit through application programs. Both modem and DMM cartridges run in the IO mode. The device code of the modem cartridge is ØFH and that of the DMM cartridge is ØEH.

5.1.7.2 Modem cartridge

(1) Outline
The PINE Modem cartridge is an optional modem unit which has the direct modem, acoustic coupler, and auto dialing functions, and allows communication through a handset.

The modem cartridge has the following features:
(a) Modem communication function
Full-duplex communication (Bell 1Ø3) at speeds up to 3ØØ bps is possible.
(b) Direct modem feature that allows direct connection to a telephone line
Acoustic coupler function is made available by connecting a speaker and microphone to the modem unit.
(c) Communication function
Audio communication using the optional handset is possible.
(d) Auto dialing and answering
Both pulse and tone dialing are possible. Auto answering is possible using a telephone-ring detection circuit.

(2) Structure
(a) Modem cartridge I/O address space

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Remarks
Read	P10H	CIOR					0	CDB2	CDB1	CDB0	
	P11H	Non used (access inhibited)									
	P12H										
	P13H										
Write	P10H	Non used (access inhibited)									
	P11H										
	P12H										
	P13H										

CIOR: Cartridge IO-mode input register
CDB2 = Ø: Carrier detected from communication line
1: Normal state
CDB1 = Ø: Indicates that data on CRXD is data received from the communication line.
1: Indicates that data on CRXD is a return code.
CDBØ = Ø: Ring detected
1: Normal state

The modem cartridge uses the following registers as well:

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Remarks
Read	P14H	ARTDIR	7 or 8 bit data								
	P16H		CAUD	CSEL							Bits 5-0 are used for other purposes.
Write	P14H	ARTDIR	7 or 8 bit data								
	P19H	IOCTLR					$\overline{CRS}$				Bits other than bit 3 are used for other purposes.

ARTDIR: ART data input register
Loaded with a return code received from the modem through RXD, or data received from the communication line.
IOSTR: IO status register
CAUD: Normally fixed at Ø or 1.
Used when monitoring signals from the communication line. In such a case, a signal to or from the communication line, which is shaped into a TTL level square wave, is present at this terminal.
CSEL: Always 1.
ARTDOR: ART data output register
Loaded with a command to the modem which is to be output to TXD, or data to the communication line.
IOCTLR: IO control register
$\overline{CRS}$: Software reset signal sent from the main unit to the modem.

Note: When using ARTDIR or ARTDOR, the serial switch must be in the cartridge·SIO mode.

(b) Communication line interface
The communication line interfaces with the modem cartridge as shown in Figure 5.1.12.

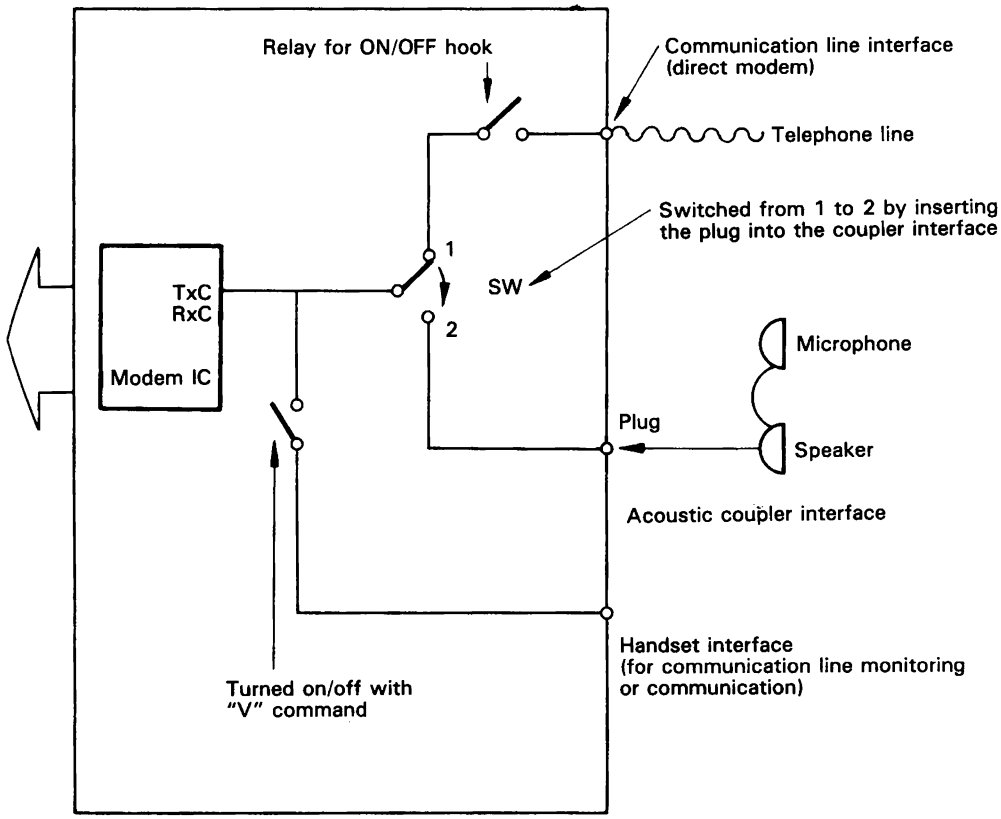


Fig. 5.1.10 Communication Line Ineterface

(3) Use
The modem cartridge can be processed using BIOS RSIOX as shown below.

(a) Modem open processing

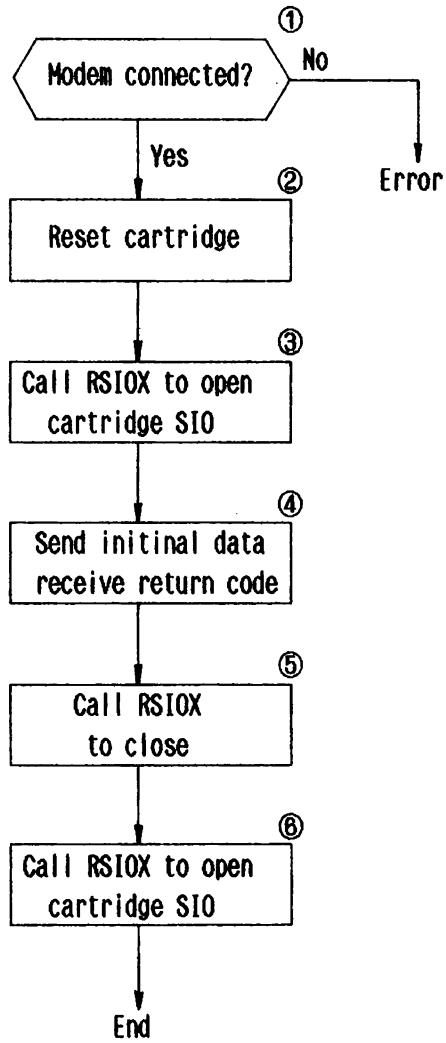
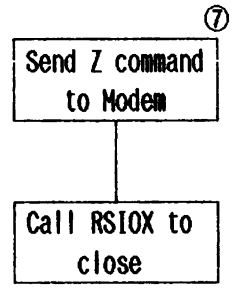


Fig. 5.1.13 Modem Open Processing

(b) Data transmission/reception processing
Transmission/reception processing of data, commands, and return codes are performed using the BIOS RSIOX GET or PUT command.

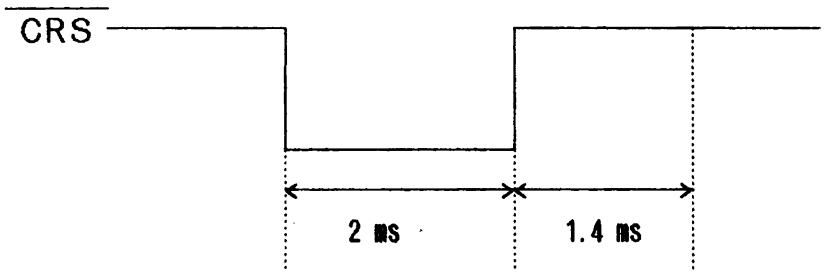
(c) Modem close processing



(d) Note
Communication via the modem is interrupted when a power-off condition occurs. That is, communication via the modem is not restored even if power is turned off in the continue mode and then turned on again (the modem remains in the halt state).

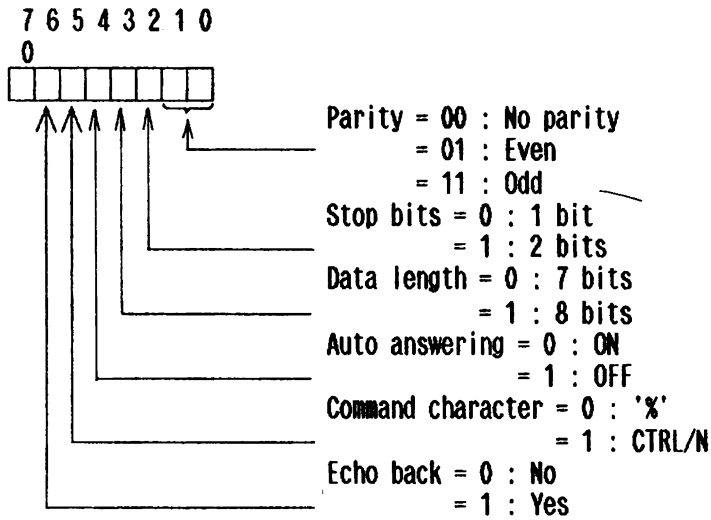
Step 1: Modem cartridge connected?
Check whether or not a modem is connected to the cartridge interface when using a modem. It can be checked with CRGDEV (ØF53FH).
CRGDEV = 8FH: Modem cartridge connected.
= Other than 8FH: Another type of cartridge connected.

Step 2: Reset cartridge.
Reset the modem to change it from the halt mode to the initial-data wait mode. It can be reset by the procedure below.
1) Set $\overline{CRS}$ to Ø and wait 2 ms.
2) Set $\overline{CRS}$ to 1 and wait 1.4 ms.



Step 3: Call RSIOX to open cartridge SIO.
Open cartridge SIO with BIOS RSIOX to send initial data to the modem. The open parameters are: cartridge SIO as I/O device, 3ØØ bps, 8 bit data, no parity, 1 stop bit.

Step 4: Send initial data.
Send initial data to the modem usnig RSIOX PUT. Initial data is 1 byte in length and has the following format:



Receive a return code using RSIOX GET. Normally, command character + 'Ø' + ØDH is returned.

Step 5: Call RSIOX to close.
After receiving a return code normally, close the cartridge SIO using BIOS RSIOX.

Step 6: Call RSIOX to reopen.
Call RSIOX OPEN to transfar data between the modem and the communication line. Open parameters must be specified in the same communication format specified in the initial data that was sent to the modem in step 4. The communication speed is fixed at 300 bps. After the cartridge SIO is opened, communication with the modem through BIOS RSIOX commands is possible.

Step 7: Send Z command to modem.
When communication with the modem is completed, send character code + 'Z' + ØDH to the modem to put it into the halt state and save power. After this, the entire procedure must be followed again to communicate with the modem.

(4) Modem commands and return codes

The modem cartridge enters the active mode (operable mode) when modem open processing is completed. After the modem cartridge is activated, application programs can control it by sending it commands.

(a) Command character <CC>

A command character is always required at the beginning of a command line. Two types of command characters, '%' (25H) and CTRL/N (ØEH), are provided by default. Which type is to be used in a given session is specified in the initial data.

A command character is also placed at the beggining of a return code to allow application programs to identify it as a return code instead of as receive data.

When processing a command character as send data, that character must be sent twice in succession. In this way, the receiver can recognize the second command character as send data.

(b) Command line

A command line begins with a command character and ends with a carriage return (ØDH). It can contain more than one command provided that its entire length does not exceed 41 characters, including the command character and carriage return.

The following four characters are ignored in command lines:

'(' (28H)
')' (29H)
'-' (2DH)
' ' (Space) (20H)

The modem cartridge does not start execution of a command until the command line ends with a carriage return. When detecting an error in a command, the modem cartridge returns '6' to the main unit to signal a syntax error, and does not execute the commands that follow. Since commands in a command line are executed sequentially from the first one, valid commands preceding an invalid one are executed normally.

(c) Return codes
After temination of command execution, the modem cartridge sends a return code to the main unit.

A return code begins with a command character and ends with a carriage return, so that it can be distinguished from receive data. The format of the return code is as follows:

<CC> + One-digit number in ASCII code + CR

Table 5.1.14 shows the return codes and their meaning.

Table 5.1.14 Return Codes

ASCII code	HEX	Meaning
Ø	3ØH	Normal termination.
1	31H	Framing or parity error detected in input from the main unit.
2	32H	Carrier from the counterpart modem detected and communication established.
3	33H	Carrier from the counterpart turned off and the communication line disconnected (handset is ON HOOK.)
4	34H	No carrier from the counterpart was detected in the specified period of time after the communication line was established.
5	35H	Line busy.
6	36H	Syntax error detected in the command line.
7	37H	Command cancelled.
8	38H	Ring detected.

(d) Control codes
The modem cartridge uses the following three control codes:

CR (ØDH): Carriage return. Used to identify the end of a command line or return code.
BS (Ø8H): Back space. Used in command lines to make corrections.
CAN (18H): Cancel. Used to cancel command lines or terminate command execution.

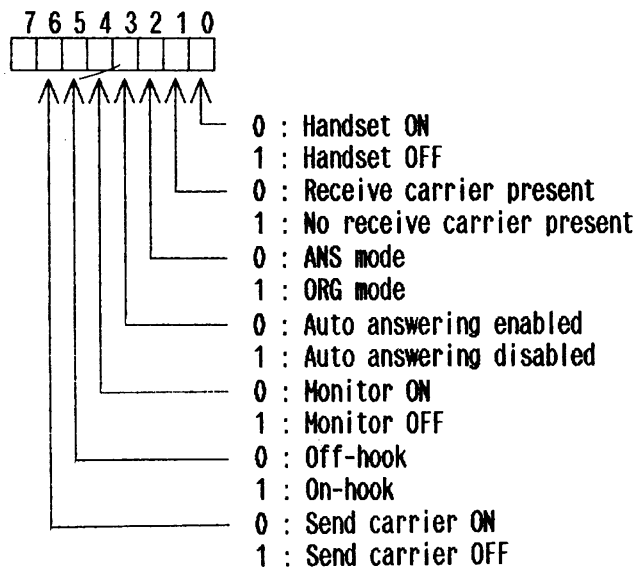
(e) Commands
Commands for the modem cartridge are listed in Table 5.1.15. Commands and parameters in the table are specified with ASCII data.

Table 5.1.15 Commands

Command	Parameter	Function
A	None.	Connects the line in the ANS (auto answering) mode.
B	None.	Turns off the send carrier.
Cn	n = Ø or 1	n = Ø: Send carrier at the space level frequency = 1: Send carrier at the mark level frequency
Ds	s = A combination of T, P, R, W, comma, Ø - 9, #, and *	Hooks off the handset and dials (auto dialing). T: Enable tone dialing. P: Enable pulse dialing. R: Switch between ORG and ANS modes. W: Dial two seconds after off-hook. Comma: Create dialing interval. Ø - 9, #, *: Phone number
Fh	h = Bit pattern whose lower-order four bits are significant	Sets up the data length, parity, and stop bits. Bit Ø = Ø: No parity = 1: Parity 1 = Ø: Even parity = 1: Odd parity 2 = Ø: 1 stop bit = 1: 2 stop bits 3 = Ø: 7-bit data = 1: 8-bit data
F?	None	Outputs the current data length, parity, and stop bit settings.
G	None	R-executes the previous command line.
H	None	Hooks on the handset.
Ln	n = Ø or 1	Makes an analog loop-back test. n = Ø: Normal operation = 1: Analog loop-back test
Mn	n = Ø or 1	Controls the monitor. n = Ø: Monitor OFF = 1: Monitor ON
Ns	s = character	Changes the command character. <CC>
O	None	Connects the line in the ORG mode.

Command	Parameter	Function
P	None	Enables pulse dialing.
R	None	Switches between ORG and ANS modes.
Sr=n	r = Ø - 3, n = Ø - 127	Rewrites the S registers. SØ: Abort time (in seconds) S1: Time corresponding to the comma sent during dialing (in seconds) S2: Interval between carrier shut-off and line disconnection (in Ø.1- second increments) S3: Number of rings after which the auto answering function is invoked
Sr?	r = Ø - 4	Outputs the S register contents. The contents of SØ through S4 are the same as those for Sr=n. See Note 1 for S4.
T	None	Enables tone dialing.
Vn	n = Ø or 1	Controls the handset. n = Ø: Handset OFF = 1: Handset ON
Z	None	Pits the modem cartridge into the power save mode.

Note 1: The S4 register has the format shown below.



Note 2: The following code is returned for the Sr? and F? commands.

<CC> + '=' + status + CR

(5) Miscellaneous
The pages that follow give points to be noted when using the modem cartridge.

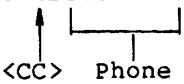
(a) Opening the modem
Open the modem by the procedure described in (3)-(a).

The modem enters the halt state when the master reset signal is input from the main unit (by power-on or reset). It exits from the halt state and waits for initial data when the modem reset signal ($\overline{CRS}$) is input.

(b) Closing the modem
Close the modem by the procedure described in (3)-(c).

When modem processing ends, send the Z command to put the modem into the halt state. Hold the modem in the halt state when it is not being used, because it consumes less power in this state.

(c) How to call
1) Calling in the ORG mode
Example: %D123-4567<CR>



When this command line is executed, the modem dials the specified phone number and waits in the ORG mode for a carrier from the counterpart.

2) Calling in the ANS mode
Example: %DR987-6543<CR>
When this command line is executed, the modem dials the specified phone number and waits in the ANS mode for a carrier from the counterpart.

3) Sending Ø with pulse dial and then dialing a number with tone dial
Example: %DPØ, T975-3124<CR>

4) Monitoring the time signal (In Japan)
Example: %M1D117W<CR>

M1 specifies that the monitor be set to ON.

5) After dialing, communicating first through the handset, and then through the modem
Example: %V1D52-2552W<CR> -- Dial after turning handset on.
%Ø<CR> ---------- Return code for V command
%Ø<CR> ---------- Return code for D command
(Communication through handset)
%O<CR> or %A<CR> --
%2<CR> ---------- Detect the carrier from the counterpart.
(Communication)

(d) Using a branch telephone
Communication is established when a number is dialed from a branch telephone and the A or O command is executed after the counterpart answered. The handset must be hooked on after command execution so that it will not puck up noise.

Communication through a branch telephone can also be established by dialing a number with the D command with W option. In this case, the line is not disconnected even when the handset is hooked on. It can be disconnected only by executing the H command and hanging up the handset.

(e) How to answer
When the S2 register is nonzero, the modem answers the phone automatically (auto answering); it counts rings to the value sepcifeid in the S2 register and executes the A command automatically. Therefore, the line is always connected in the ANS mode when the auto answering function is enabled.

The auto answering function is disabled when the S2 register is loaded with Ø. Even in this state, the modem informs the main unit of ring detection by sending it return code 8. When a ring is detected, execute the A or O command to establish communication.

(f) Handset
The handset can be turned on or off using the V command. Since whether the handset is on or off does not affect the communication function of the modem, the communication line can be monitored by turning on the handset during data transmission. However, noises caught by the handset microphone may enter the line and disturb communication.

(g) Acoustic coupler
The modem cartridge serves as an acoustic coupler when furnished with an optional acoustic coupler unit. In this case, the telephone line is disconnected from the modem.

The only differences between the coupler and modem are that 1) with the coupler, the computer sends data through the telephone mouthpiece while the modem sends it directly via the telephone line, and 2) with the coupler, the computer receives data through the telephone earpiece, while the modem gets it directly from the telephone line. The function of commands are the same for both couplers and modems except that the coupler is not provided with the dialing function.

If both the handset and coupler are connected at the same time, neither of them can be used.

5.1.7.3 DMM cartridge

(1) General
The Digital MultiMeter (DMM) cartridge is an intelligent measuring unit which incorporates an A/D conversion IC and 4-bit CMOS CPU.

The DMM cartridge transfers data and commands via the PINE cartridge interface in 600-bps asynchronous serial communication mode. The cartridge interface is set to the IO mode and used as the control signal line (Power ON/OFF, RDY).

(2) Structure
(a) DMM cartridge I/O address space

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Remarks
Read	P10H	CIOR	Output port contents							CDB0	
	P11H	Not used (access inhibited)									
	P12H										
	P13H										
Write	P10H	CIOR	0	0	0	CDB4	Don't care				
	P11H	Not used (access inhibited)									
	P12H										
	P13H										

CIOR: Cartridge IO register
CDB0: Command (Range switch command) RDY signal
= 0: Ready
= 1: Busy
CDB4: Power-on/off control signal
= 0: Power-on
1: Power-off

In addition to the above signal lines, CTXD and CRXD are used for asynchronous serial communication (600 bps, 8 data bits, no parity, 1 stop bit). In actual operation, data is sent or received via ARTDOR (P14H) or ARTDIR (P14H). The serial switch must be set to the cartridge SIO mode.

```
                        ;       ************************************************
                        ;               MODEM SAMPLE PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This sample program is for using MODEM
                        ;               cartridge.
                        ;
                        ;       <> assemble condition <>
                        ;
                                .Z80
                        ;
                        ;       <> loading address <>
                        ;
                                .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
                        ;       IO registers
0018                    ZSWR            EQU     18H             ; Switch register
0002                    ZCSW1           EQU     02H             ;  Cartridge switch 1
0001                    ZCSW0           EQU     01H             ;  Cartridge switch 2
0019                    ZIOCTLR         EQU     19H             ; IO control register
0003                    BZCRS           EQU     3               ;  CRS signal bit
                        ;
                        ;       System area
F005                    RZSWR           EQU     0F005H          ; SWR data setting area
F006                    RZIOCTLR        EQU     0F006H          ; IOCTLR data setting area
F53F                    CRGDEV          EQU     0F53FH          ; Cartridge device code
                        ;
                        ;       BIOS entry (RBIOS2)
EB03                    WBOOT           EQU     0EB03H          ; Warm Boot
EB54                    RSIOX           EQU     WBOOT   +51H    ; Serial Input/Output
EB06                    CONST           EQU     WBOOT   +03H    ; Console status
EB09                    CONIN           EQU     WBOOT   +06H    ; Console in
EB0C                    CONOUT          EQU     WBOOT   +09H    ; Console out
                        ;
                        ;       RSIOX parameter
0010                    RSOPN           EQU     10H             ; Open code
0020                    RSCLS           EQU     20H             ; Close code
0030                    RSIST           EQU     30H             ; Input status code
0040                    RSOST           EQU     40H             ; Oltput status code
0050                    RSGET           EQU     50H             ; Get code
0060                    RSPUT           EQU     60H             ; Put code
                        ;
0001                    RS232           EQU     01H             ; RS-232C using
0002                    SIO             EQU     02H             ; SIO using
0003                    CSIO            EQU     03H             ; Cartridge SIO using
                        ;
                        ;       Cartridge mode.
                        ;
0000                    DBMODE          EQU     00000000B       ; DB mode
0040                    HSMODE          EQU     01000000B       ; HS mode
0080                    IOMODE          EQU     10000000B       ; IO mode
00C0                    OTMODE          EQU     11000000B       ; OT mode
                        ;
000F                    DVMDM           EQU     00001111B       ; Device code for Modem
                        ;
000D                    CR              EQU     0DH             ; Carriage return.
000A                    LF              EQU     0AH             ; Line feed.
                        ;
0200                    RCVSZ           EQU     200H            ; Receive buffer size.
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       CAUTION :
                        ;               If you use resident BIOS, change
                        ;               RBIOS1 to RBIOS2.
                        ;               If Your program is ROM execute program,
                        ;               you must use RBIOS2.
                        ;
0100                    START:
0100    C3 01AE                 JP      PSTART
                        ;
                        ;       Message
                        ;
0103    53 74 61 72     MSG1:   DB      'Start of Modem communication.',CR,LF,00H
0107    74 20 6F 66
010B    20 4D 6F 64
010F    65 6D 20 63
0113    6F 6D 6D 75
0117    6E 69 63 61
011B    74 69 6F 6E
011F    2E 0D 0A 00
0123    4D 6F 64 65     MSG2:   DB      'Modem cartridge check.',CR,LF,00H
0127    6D 20 63 61
012B    72 74 72 69
012F    64 67 65 20
0133    63 68 65 63
0137    6B 2E 0D 0A
013B    00
013C    53 65 6E 64     MSG3:   DB      'Send initial data.',CR,LF,00H
0140    20 69 6E 69
0144    74 69 61 6C
0148    20 64 61 74
014C    61 2E 0D 0A
0150    00
0151    52 65 63 65     MSG4:   DB      'Receive answer code.',CR,LF,00H
0155    69 76 65 20
0159    61 6E 73 77
015D    65 72 20 63
0161    6F 64 65 2E
0165    0D 0A 00
```

```
0168  21 21 21 20          MSG5:   DB      '!!! Start of Modem !!!',CR,LF,00H
016C  53 74 61 72
0170  74 20 6F 66
0174  20 4D 6F 64
0178  65 6D 20 21
017C  21 21 0D 0A
0180  00
0181  21 21 21 20          MSG6:   DB      '!!! End of Modem !!!',CR,LF,00H
0185  45 6E 64 20
0189  6F 66 20 4D
018D  6F 64 65 6D
0191  20 21 21 21
0195  0D 0A 00
0198  43 6F 6D 6D          MSG7:   DB      'Communication error',CR,LF,00H
019C  75 6E 69 63
01A0  61 74 69 6F
01A4  6E 20 65 72
01A8  72 6F 72 0D
01AC  0A 00

                                ;
                                ;
                                ;       Main program
                                ;
01AE                            PSTART:
01AE  31 4000                           LD      SP,4000H        ; Set stack pointer.
01B1  21 0103                           LD      HL,MSG1         ; 'Start of Modem communication'
01B4  CD 02BC                           CALL    MSGDSP          ; Message display.

01B7  21 0123                           LD      HL,MSG2         ; 'Modem cartridge check'
01BA  CD 02BC                           CALL    MSGDSP          ; Message display.
01BD  3A F53F                           LD      A,(CRGDEV)      ; Get cartridge device code.
01C0  47                                LD      B,A
01C1  E6 3F                             AND     03FH            ; Get device code only.
01C3  FE 0F                             CP      DVMDM           ; Modem cartridge?
01C5  C2 EB03                           JP      NZ,WBOOT        ; No. then WBOOT.
01C8  78                                LD      A,B             ; Restore CRGDEV.
01C9  FE 8F                             CP      IOMODE+DVMDM    ; Already IO mode?
01CB  C4 0289                           CALL    NZ,SETMODE      ; No. then set into IO mode.
                                ;
01CE                            CRESET:
01CE  CD 029D                           CALL    CRST02          ; CRS line on and wait 10 msec.
01D1  CD 0296                           CALL    CRST01          ; CRS line off and wait 10 msec.
                                ;
01D4                            MDMOPN:
01D4  21 02F2                           LD      HL,OPNDAT       ; Copy RSIOX open parameters.
01D7  11 0305                           LD      DE,OPNPRM       ;
01DA  01 0009                           LD      BC,9            ;
01DD  ED B0                             LDIR                    ;
                                ;
01DF  21 0305                           LD      HL,OPNPRM       ; RSIOX open.
01E2  06 13                             LD      B,RSOPN+CSIO    ; Using Cartridge SIO.
01E4  CD EB54                           CALL    RSIOX           ;
01E7  B7                                OR      A               ; Already opened?
01E8  C2 026F                           JP      NZ,ERREND       ; Yes.
                                ;
01EB                            PUTINIT:
01EB  21 013C                           LD      HL,MSG3         ; 'Send initial data.
01EE  CD 02BC                           CALL    MSGDSP          ; Display message.
                                ;
01F1  21 0305                           LD      HL,OPNPRM       ; Send initial data.
01F4  3A 02FB                           LD      A,(INITDATA)    ;  Initial data --> A
01F7  4F                                LD      C,A             ;
01F8  06 60                             LD      B,RSPUT         ;  Using PUT function of RSIOX.
01FA  CD EB54                           CALL    RSIOX           ;
01FD  20 70                             JR      NZ, ERREND      ;  Error return, then retry.
                                ;
01FF                            GETANS:
01FF  21 0151                           LD      HL,MSG4         ; 'Receive answer code.'
0202  CD 02BC                           CALL    MSGDSP          ; Display message.
                                ;
0205                            GETANS1:
0205  21 0305                           LD      HL,OPNPRM       ; Receive answer code.
0208  06 50                             LD      B,RSGET         ;  Using GET function of RSIOX.
020A  CD EB54                           CALL    RSIOX           ;
020D  C2 026F                           JP      NZ,ERREND       ;  Error return, then Warm Boot.
                                ;
0210  CD 02C8                           CALL    CHARDSP         ; Display receiving character.
0213  FE 0D                             CP      CR              ; Recive char. is CR?
0215  20 EE                             JR      NZ,GETANS1      ; No. (Loop until receivig CR code.)
                                ;
0217  06 20                             LD      B,RSCLS         ; Close RSIOX.
0219  CD EB54                           CALL    RSIOX           ;
021C  21 0168                           LD      HL,MSG5         ; '!!! Start of modem !!!'
021F  CD 02BC                           CALL    MSGDSP          ; Display message.
                                ;
0222  21 02FC                           LD      HL,MDMDAT       ; Copy initial data parameter for RSIOX.
0225  11 0305                           LD      DE,OPNPRM       ;
0228  01 0009                           LD      BC,9            ;
022B  ED B0                             LDIR                    ;
022D  21 0305                           LD      HL,OPNPRM       ; Open RSIOX
0230  06 13                             LD      B,RSOPN+CSIO    ;  Using OPEN function of RSIOX.
0232  CD EB54                           CALL    RSIOX           ;
0235  B7                                OR      A               ; Error return?
0236  C2 026F                           JP      NZ,ERREND       ; Yes. (then Warm Boot)
                                ;
0239                            KEYCHK
0239  CD EB06                           CALL    CONST           ; Get console status.
023C  3C                                INC     A               ; Exist inputing key data?
023D  28 0E                             JR      Z,PUT           ; Yes.
                                ;
023F  21 0305                           LD      HL,OPNPRM       ; Get RSIOX receiving status.
0242  06 30                             LD      B,RSIST         ;  Using INSTS function of RSIOX.
0244  CD EB54                           CALL    RSIOX           ;
0247  3C                                INC     A               ; Exist receiving data?
0248  28 18                             JR      Z,GET           ; Yes.
024A  76                                HALT                    ;
024B  18 EC                             JR      KEYCHK          ; Loop to Key Check.
                                ;
```

```
024D                    PUT:
024D    CD EB09                 CALL    CONIN           ; Get inputed key code.
0250    FE 03                   CP      03H             ; STOP key?
0252    28 21                   JR      Z,PEND          ; Yes.
0254    CD 02C8                 CALL    CHARDSP         ; Display inputed key data.

0257    21 0305                 LD      HL,OPNPRM       ; Send inputed key data.
025A    4F                      LD      C,A             ;
025B    06 60                   LD      B,RSPUT         ;  Using PUT function of RSIOX.
025D    CD EB54                 CALL    RSIOX           ;
0260    18 D7                   JR      KEYCHK          ; Loop to Key Check.
                        ;
0262                    GET:
0262    21 0305                 LD      HL,OPNPRM       ; Get reciving data.
0265    06 50                   LD      B,RSGET         ;  Using GET function of RSIOX.
0267    CD EB54                 CALL    RSIOX           ;
026A    CD 02C8                 CALL    CHARDSP         ; Display receiving data.
026D    18 CA                   JR      KEYCHK          ; Loop to Key Check.
                        ;
026F                    ERREND:
026F    21 0198                 LD      HL,MSG7         ; 'Communication error.'
0272    CD 02BC                 CALL    MSGDSP          ; Display message.
0275                    PEND:
0275    21 0161                 LD      HL,MSG6         ; '!!! End of modem !!!'
0278    CD 02BC                 CALL    MSGDSP          ; Display message.
027B    21 02FE                 LD      HL,CLSCMD       ; Modem close parameter --> HL
027E    CD 02AB                 CALL    PUTDATA         ; Send modem close parameter.
                        ;
0281    06 20                   LD      B,RSCLS         ; Close RSIOX.
0283    CD EB54                 CALL    RSIOX           ;
0286    C3 EB03                 JP      WBOOT           ; Jump to Warm Boot.
                        ;
                        ;
                        ;       ************************************************
                        ;               SELECT IO MODE
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Select IO mode (Cartridge mode)
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
0289                    SETMODE:
0289    3A F005                 LD      A,(RZSWR)               ; Get switch register data.
028C    E6 FC                   AND     0FFH-ZCSW1-ZCSW0        ; Clear CSW1,0 bit.
028E    F6 01                   OR      ZCSW0                   ; Set IO mode.
0290    32 F005                 LD      (RZSWR),A               ; Store to memory.
0293    D3 18                   OUT     (ZSWR),A                ; Output to IO port.
0295    C9                      RET                             ;
                        ;
                        ;       ************************************************
                        ;               CRS LINE CONTROL SUBROUTINE
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               There are two routine, one is setting
                        ;               CRS high, and one is setting CRS low.
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
0296                    CRST01:
0296    3A F006                 LD      A,(RZIOCTLR)    ; Get IO control register data.
0299    CB DF                   SET     BZCRS,A         ; Set CRS high.
029B    18 05                   JR      CRST            ;
029D                    CRST02:
029D    3A F006                 LD      A,(RZIOCTLR)    ; Get IO control register data.
02A0    CB 9F                   RES     BZCRS,A         ; Reset CRS low.
02A2                    CRST:
02A2    32 F006                 LD      (RZIOCTLR),A    ; Store data to memory.
02A5    D3 19                   OUT     (ZIOCTLR),A     ; Out put to IO port.
02A7    CD 02DE                 CALL    WAIT10          ; Wait about 10 msec.
02AA    C9                      RET                     ;
                        ;       ************************************************
                        ;               SEND DATA TO RSIOX
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               Send data to RSIOX until finding
                        ;               00h code.
                        ;
                        ;       <> entry parameter <>
                        ;               HL : Data top address.
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;               If error happend, then stop this
                        ;               this program.
                        ;
02AB                    PUTDATA:
02AB    4E                      LD      C,(HL)          ; Get sending data. (1 byte)
02AC    0C                      INC     C               ;
02AD    0D                      DEC     C               ; Data is 00h?
02AE    C8                      RET     Z               ; Yes. (then return)
```

```
02AF    E5                      PUSH    HL              ; Save parameter.
02B0    21 0305                 LD      HL,OPNPRM       ; Send data to RSIOX.
02B3    06 60                   LD      B,RSPUT         ;  Use PUT function of RSIOX.
02B5    CD EB54                 CALL    RSIOX           ;
02B8    E1                      POP     HL              ; Restore parameter.
02B9    23                      INC     HL              ; Pointer update.
02BA    18 EF                   JR      PUTDATA         ; Loop.

                        ;***************************************************
                        ;       DISPLAY MESSAGE UNTIL FIND 00H
                        ;***************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               HL : Message data top address
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
02BC                    MSGDSP:
02BC    4E                      LD      C,(HL)          ; Get displaying data.
02BD    0C                      INC     C               ;
02BE    0D                      DEC     C               ; Getting data is 00h?
02BF    C8                      RET     Z               ; Yes. (then return)
                        ;
02C0    E5                      PUSH    HL              ; Save parameter.
02C1    CD EB0C                 CALL    CONOUT          ; Console out data.
02C4    E1                      POP     HL              ; Restore parameter.
02C5    23                      INC     HL              ; Update parameter.
02C6    18 F4                   JR      MSGDSP          ; Loop.

                        ;***************************************************
                        ;       DISPLAY A CHARACTER
                        ;***************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               A  : Character code.
                        ;          return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               All registers.
                        ;
                        ;       CAUTION :
                        ;               If character code is CR, then console
                        ;               out CR with LF.
02C8                    CHARDSP:
02C8    F5                      PUSH    AF              ; Save all registers.
02C9    C5                      PUSH    BC              ;
02CA    D5                      PUSH    DE              ;
02CB    E5                      PUSH    HL              ;
02CC    F5                      PUSH    AF              ;
02CD    4F                      LD      C,A             ; Console out data.
02CE    CD EB0C                 CALL    CONOUT          ;
02D1    F1                      POP     AF              ; Get inputed parameter,
02D2    FE 0D                   CP      CR              ; Is it CR?
02D4    0E 0A                   LD      C,LF            ; If so, then console out LF.
02D6    CC EB0C                 CALL    Z,CONOUT        ;
02D9    E1                      POP     HL              ; Restore all registers.
02DA    D1                      POP     DE              ;
02DB    C1                      POP     BC              ;
02DC    F1                      POP     AF              ;
02DD    C9                      RET                     ;

                        ;***************************************************
                        ;       WAIT ABOUT 100 MILI SECOND
                        ;***************************************************
                        ;
                        ;       NOTE :
                        ;
                        ;       <> entry parameter <>
                        ;               NON
                        ;       <> return parameter <>
                        ;               NON
                        ;       <> preserved registers <>
                        ;               NON
                        ;
                        ;       CAUTION :
                        ;
02DE                    WAIT10:
02DE    01 0002                 LD      BC,2            ;
02E1    F5                      PUSH    AF              ;
02E2                    WT10:
02E2    3E E6                   LD      A,230           ;
02E4                    WT20:
02E4    3D                      DEC     A               ;
02E5    20 FD                   JR      NZ,WT20         ;
                        ;
02E7    0B                      DEC     BC              ;
02E8    78                      LD      A,B             ;
02E9    B1                      OR      C               ;
02EA    20 F6                   JR      NZ,WT10         ;
02EC    F1                      POP     AF              ;
02ED    C9                      RET                     ;

                        ;       Constant data & Work area
                        ;
02EE                    CLSCMD:
02EE    25 5A 0D 00             DB      '%Z',CR,00H     ; Modem close command data.
                        ;
                        ;       RSIOX first open parameter.
                        ;
```

```
02F2                    OPNDAT:
02F2    030E                    DW      RCVBUF          ; Receive buffer top address.
02F4    0200                    DW      RCVSZ           ; Receive buffer size.
02F6    06                      DB      006H            ; Baud rate. (300 bps)
02F7    03                      DB      003H            ; Bit length. (8 bits/character)
02F8    00                      DB      000H            ; Parity. (non parity)
02F9    01                      DB      001H            ; Stop bit. (1 stop bit)
02FA    FF                      DB      0FFH            ; Special parameter.
                        ;
                        ;       Modem initial data.
                        ;               (8 bits, non parity, 1 stop bit)
02FB                    INITDATA:
02FB    58                      DB      01011000B
                        ;
                        ;       RSIOX second open parameter.
                        ;               (This parameters are matched with initial data.)
02FC                    MDMDAT:
02FC    030E                    DW      RCVBUF          ; Receive buffer top address.
02FE    0200                    DW      RCVSZ           ; Receive buffer size.
0300    06                      DB      006H            ; Baud rate.
0301    03                      DB      003H            ; Bit length.
0302    00                      DB      000H            ; Parity.
0303    01                      DB      001H            ; Stop bit.
0304    FF                      DB      0FFH            ; Special parameter.
                        ;
                        ;       RSIOX parameter area (for calling & return)
                        ;
0305                    OPNPRM:
0305                            DS      9
                        ;
                        ;       Receiving buffer area.
                        ;
030E                    RCVBUF:
030E                            DS      RCVSZ

                                END
```

(3) Use
The DMM cartridge is used following the steps shown below using BIOS RSIOX:

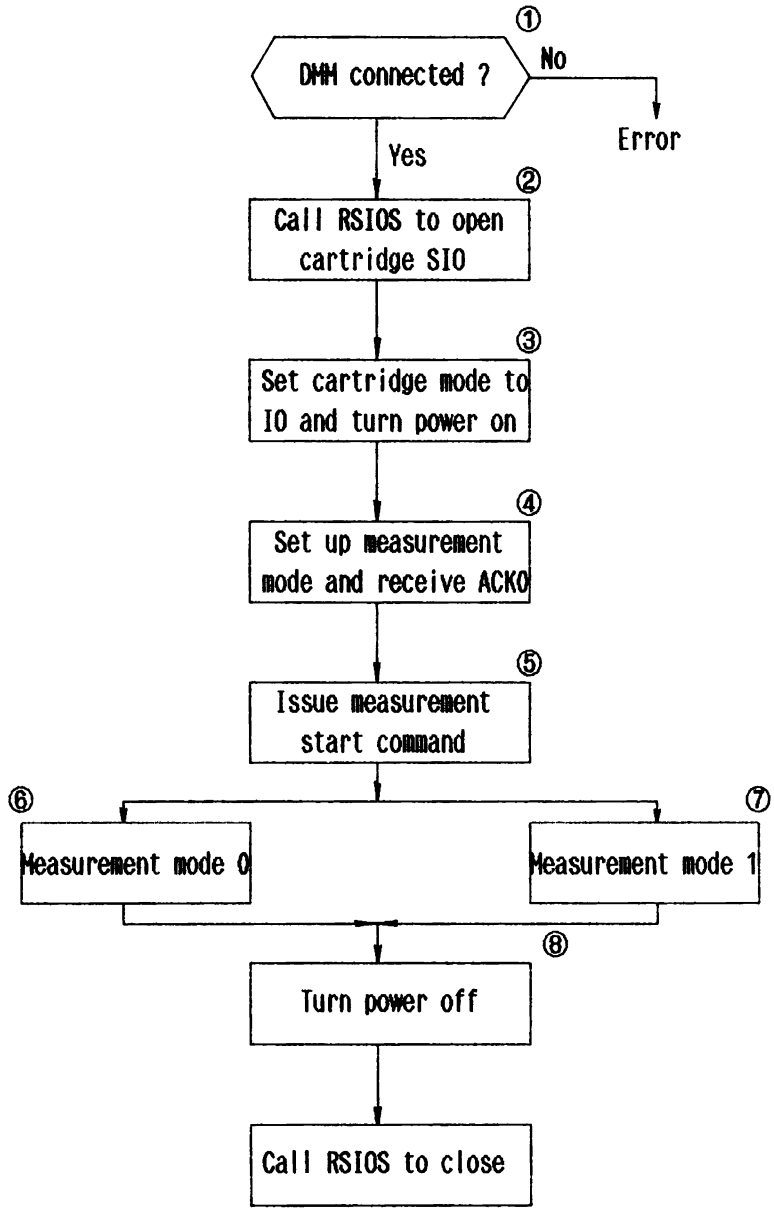
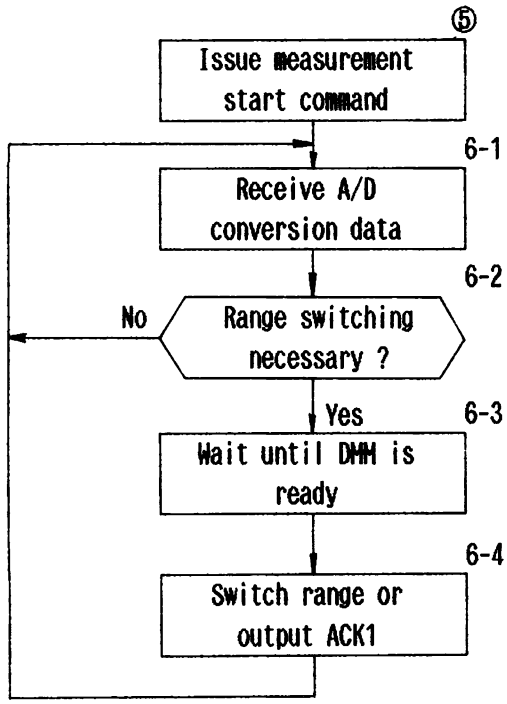
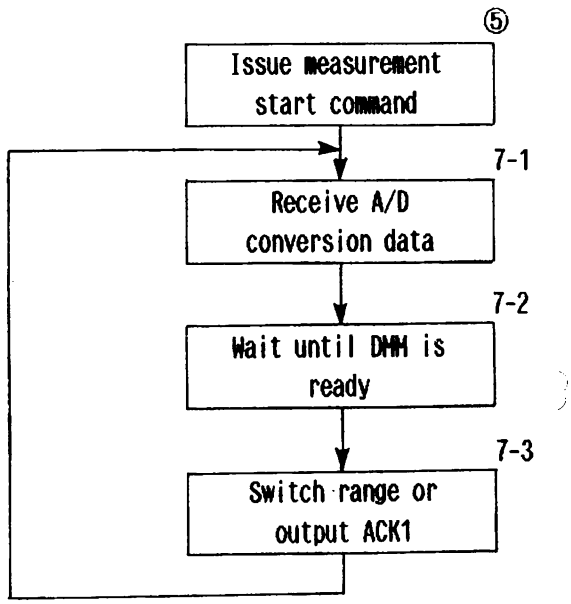


Fig. 5.1.16 DMM Processing Procedure

The measurement procedure in measurement mode Ø is as follows:



The measurement procedure in measurement mode 1 is shown below.



The difference between modes Ø and 1 is as follows:
In measurement mode Ø, the DMM cartridge automatically starts sending A/D conversion data if the range switch command is not sent within a specified period after the DMM cartridge is ready (approximately 1/1Ø of the sampling period).
In measurement mode 1, the DMM cartridge keeps waiting until either a Range Switch command or ACK1 is received.

Step 1: DMM cartridge connected?
Check whether or not a DMM cartridge is connected to the cartridge interface. This can be checked with CRGDEV (ØF53FH).
CRGDEV, bits 3 - Ø
= ØEH: DMM cartridge connected
= Other than ØEH: Another type of cartridge connected.

Step 2: Call RSIOX to open cartridge SIO.
Open cartridge SIO with BIOS RSIOX. The open parameters are: cartridge SIO as I/O device, 6ØØ bps, 8 data bits, no parity, 1 stop bit.

Step 3: Set cartridge to IO and turn power on.
Set the cartridge mode to IO through the following procedure:

```
LD   A, (RZSWR)
AND  ØFCH
OR   Ø1H
LD   (RZSWR), A
OUT  (18H), A
```

RZSWR (ØFØØ5H) is the area that stores the SWR (P19H) output status.

After setting up the IO mode, turn power on by writing ØØH to P1ØH. After power-on, the system waits approximately three seconds to allow for DMM reset processing.

Step 4: Set up measurement mode and receive ACKØ.
After power-on, send a measurement mode command to the DMM using RSIOX PUT. The measurement command has the following format:

7	6	5	4	3	2	1	0
1	m2	m1	m0	n3	n2	n1	n0

m2 : Selects measurement mode
= 0 : Measurement mode 0
1 : Measurement mode 1
m1 : Don't care
m0 : Selects voltage/resistance
= 0 : Measure voltage
1 : Measure resistance
n3 : Selects DCV/ACV or ohm/LP(Low Power)-Ohm
= 0 : DCV or ohm
1 : ACV or LP-ohm
n2~n0 : Selects range

n2 n1 no Measurement mode	000	001	010	011	100	101	100
DCV	AUTO		300mV	3V	30V	300V	500V
ACV	AUTO			3V	30V	300V	350V
Ω	AUTO		300Ω	3kΩ	30KΩ	300 KΩ	3000 KΩ
LPΩ	AUTO			3KΩ	30KΩ	300 KΩ	3000 KΩ

After sending the measurement command, receive ACKØ using RSIOX GET.
The values in the measurement command, except that of ml, are returned unchanged as ACKØ.

ml = Ø: 5Ø Hz sampling rate
Approx. 5 times/second
= 1: 6Ø Hz sampling rate
Approx. 4 times/second

Step 5: Send measurement start command.
Send a measurement start command to the DMM using RSIOX PUT. The data to be sent the DMM may be of any value. On receiving a data byte, the DMM starts measurement in the mode specified in step 4.

Step 6-1 or 7-1: Receive A/D conversion data.
Receive A/D conversion data using RSIOX GET. A/D conversion data is three bytes in length. See (4) for details.

Step 6-2: Range switching necessary?
If the measurement mode is other than AUTO, determine in the application program whether or not range switching is necessary, according to the results of 6-1 or 7-1.

Step 6-3 or 7-2: Wait until DMM is ready.
Examine P1ØH, bit Ø, and identify whether or not the DMM is ready. If not (P1ØH, bit Ø = 1), wait until it gets ready.

Step 6-4 or 7-3: Switch range or send ACK1.
Send a range switch command or send ACK1 to the DMM using RSIOX PUT. Send a range switch command when switching the range, or ACK1 when not. See (4) for the range switch command and ACK1.

Step 8: Turn power off.
Turn off DMM power when measurement is completed. Power is turned off by writing Ø1ØH to P1ØH.

Note:
The DMM is automatically turned off if a power-off signal is received while the DMM is measuring voltage or resistance. Even if power is turned off in the continue mode, the DMM remains offline when power is turned on again. That is, DMM processing DMM cannot be continued. In such a case, follow the above procedure starting at step 1.

If measurement is found to be unsuccessful, carry out power-off processing in step 8 and begin agaom frp, step 1. When turning on the DMM immediately after turning it off, be srue to wait at least three seconds after power-off before switching power on again.

(4) Command, ACK, conversion data

(a) Range switch command

7	6	5	4	3	2	1	0
0	0	0	0	n3	n2	n1	n0

n3 : Selects DCV/ACV or ohm/LP-Ohm.

= 0 : DCV or ohm

1 : ACV or LP-ohm

n2 - n0 : Selects range

n2 n1 n0 Measurement mode	000	001	010	011	100	101	110
DCV	AUTO		300mV	3V	30V	300V	500V
ACV	AUTO			3V	30V	300V	350V
Ω	AUTO		300Ω	3KΩ	30KΩ	300KΩ	3000KΩ
LPΩ	AUTO			3KΩ	30KΩ	300KΩ	3000KΩ

(b) ACK1

7	6	5	4	3	2	1	0
1	*	*	*	*	*	*	*

An asterisk (*) stands for 'Don't care'.

(c) A/D conversion data
A/D conversion data consists of three bytes and has the following format:

First byte

7	6	5	4	3	2	1	0
1	1	1	1	0	0	0	0
D	D	D	D	D	D	D	D
2	2	1	0	3	2	1	0

BCD data, tens place

BCD data, ones place

B : Battery low bit
= 1 : Battery low
0 : Battery OK
Replace the battery when voltage gets low.

Second byte

7	6	5	4	3	2	1	0
0	S	3	3	2	2	2	2
		D	D	D	D	D	D
		1	0	3	2	1	0

BCD data, thousands place

BCD data, hundreds place

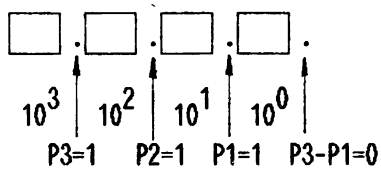
K : K units (during measurement of voltage or resistance)
= 1 : K-ohm
= 0 : Not K-ohm units
M : M units (during measurement of voltage or resistance)
= 1 : mV units
= 0 : Not mV units
A : Auto range
= 1 : Auto range
= 0 Manual range

Third byte

7	6	5	4	3	2	1	0
A	M	K	B	0	P1	P2	P3

Decimal point

S : Sign bit
= 1 : Minus
= 0 : plus
P1 - P3 : Decimal point



(5) Miscellaneous
(a) Auto range
When Auto range is specified, the DMM cartridge automatically selects the range appropriate to the measured value.

The DMM cartridge selects the 1-level higher or lower range as follows:
Higher range: When the count exceeds 1999
Example: If the resistance exceeds 19.99K ohms when measured at the 30K-ohm range, the 300K-ohm range is automatically selected.
Lower range: When the count is below 179
Example: If the resistance is less than 1.79K ohms when measured at the 30K-ohm range, the 3K-ohm range is automatically selected.

(b) Power-off
The DMM cartridge must always be turned off when it is not to be used. Otherwise, its battery will be quickly be drained.

(c) Battery low
When the voltage of the DMM battery falls within the range 1.32V - 0.8V through 1.32V + 0.8V, the A/D conversion data 13 bits are set to 1. In this case, the battery must be replaced.

(d) Changing the reset time
AC noise can be offset by setting the reset time according to the power (AC) frequency (50Hz/60Hz) of the area where the DMM is used.

The reset time can be set by turning the switch placed under the lid of the battery box.

ACK0, bit 5 is loaded with the currently specified reset time. PINE software can identify the specified reset time.

5.2 Serial Interfaces

5.2.1 General

The PINE supports RS-232C, SIO, and cartridge SIO as serial communication devices. In principle, these serial devices are switched by a serial controller and cannot be used simultaneously.

5.2.2 Setting Up Serial Interface

5.2.2.1 Switching serial modes

The serial interface modes (RS-232C, SIO, and cartridge SIO) can be switched through SWR (P18H), bits 3 and 2.

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Remarks
Write	18H	SWR					SSW1	SSW0			Bits other than 3 and 2 are used for other purposes

Serial mode	SSW1	SSW2	R×D	T×D	Devices
0	0	0	Cartridge SIO	Cartridge SIO	
1	0	1	SIO	SIO	
2	1	0	RS-232C	RS-232C	Printer, acoustic coupler
3	1	1	RS-232C	SIO	

5.2.2.2 Setting up parameters

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Remarks
Read	14H	ARTDIR	7 or 8 bit data								
	15H	ARTSR	RDSR		FE	OE	PE	Tx empty	Rx RDY	Tx RDY	
	16H	IOSTR			RCTS	RCD	RXD	SIN			Bits other than 5 through 2 are used for other purposes
Write	00H	CTLR1	BRG3	BRG2	BRG1	BRG0					Bits other than 3 through 0 are used by purposes
	14H	ARTDOR	7 or 8 bit data								
	15H	ARTMR	STOP		EVEN	PEN		DATA			
	16H	ARTCR			RRTS	ER	SBRK	RXE	RDTR	TXE	
	19H	IOCTLR						SOUT			Bits other than 2 are used for other purpose

See FIRMWARE, CHAPTER 2, "I/O REGISTERS" for register contents.

5.2.3 System Support

5.2.3.1 Switching serial interfaces

The PINE supports three types of serial interface configurations: SIO for FDDs, RS-2232C or SIO for list devices, and RS-232C, SIO, or cartridge SIO for by the user. PINE OS automatically controls the serial switch to allow these configurations to operate independently.

Figure 5.2.1 illustrates the serial interface configurations.

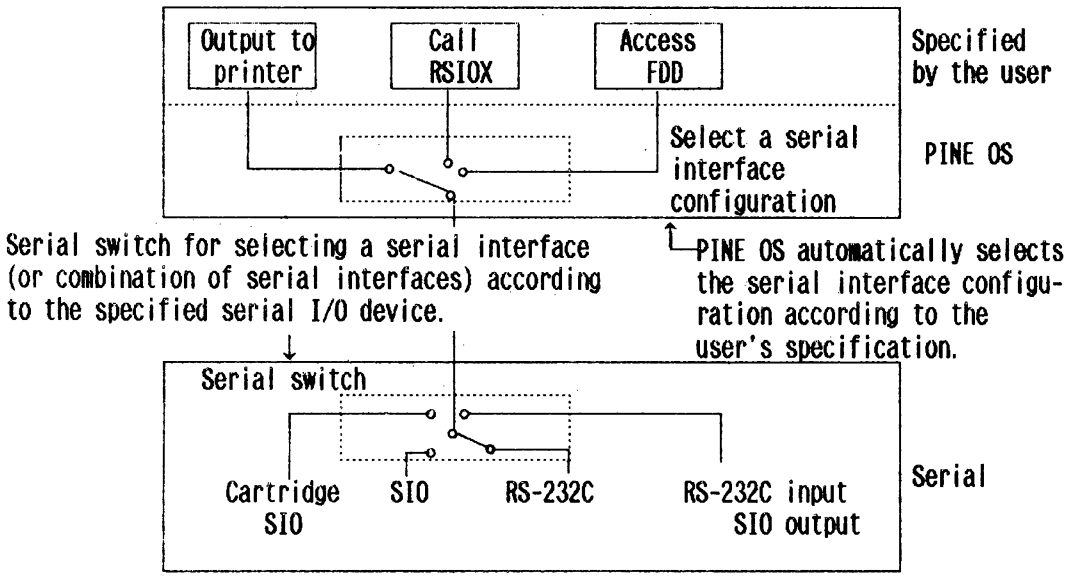


Fig. 5.2.1 Serial Interface Configurations

The OS selects a serial interface configuration as follows:

1) Loads the current configuration into SRMODE (ØF33CH).
2) Checks whether or not the the serial interface configuration to be used matches the current serial interface configuration.
3) If they match, performs the subsequent processing without changing the current configuration.
   If not, performs the subsequent processing after changing to the newthe configuration.

The parameters for changing the serial interface configuration are stored in the 15-byte area, SRTABL (ØF279H).

SRMODE (ØF33CH) 1 byte
Indicates the current serial interface configuration.
= ØFFH: Not used.
= ØØH: Used by the system (list device).
= Ø1H: Used by the user (RSIOX).
= Ø2H: Used by FDD (floppy disk drive).
The initial value is ØFFH. It is initialized by a warm boot.

SRTABL (ØF279H) ØFH bytes
Serial interface configuration switching table

Name	Length	Contents
SYSCTLR1	1	CTLR1 for LIST
SYSARTMR	1	ARTMR for LIST
SYSSWR	1	SWR for LIST
SYSARTCR	1	ARTCR for LIST
SYSSOUT	1	SOUT for LIST
RS2CTLR1	1	CTLR1 for RSIOX
RS2ARTMR	1	ARTMR for RSIOX
RS2SWR	1	SWR for RSIOX
RS2ARTCR	1	ARTCR for RSIOX
RS2SOUT	1	SOUT for RSIOX
HSCTLR1	1	CTLR1 for FDD
HSARTMR	1	ARTMR for FDD
HSSWR	1	SWR for FDD
HSARTCR	1	ARTCR for FDD
HSSOUT	1	SOUT for FDD

- This table is divided into 5-byte units. Each entry in the table corresponds to I/O register CTLR1, ARTMR, SWR, ARTCR, or IOCTLR.
- Bits not associated with serial interfaces must be set to Ø.
- RS2CTLR1 through RS2SOUT are set when RSIOX OPEN is executed.

5.2.3.2 How serial interfaces are used

Although the PINE is furnished with three types of serial interfaces for serial communication, only one is available at a time. Consequently, the system selects the serial interface to be used according to the module to be executed.

The modules that use the seial interfaces are divided into the following groups:
1) System modules
Modules that communicate with the device selected through the I/O byte (PUN:, RDR:, or CON:). (RS-232C)
2) User modules
Modules that use serial interfaces. (RS-232C, SIO, or cartridge SIO)
3) Disk and list device modules
Modules that use terminal floppy disks. (SIO)
System modules that use list devices. (RS-232C, SIO)

Since these devices operate independently, the I/O mode must be specified separately for each module.

Table 5.2.2 shows how one module can use a serial interface while another module is using it.

Table 5.2.2 Interrelationships among Serial Interfaces and Modules

Module to use serial device \ Module currently using serial device		DISK/LST module	System module	User module			
				RS-232C	SIO	Cartridge SIO	RS-232C input SIO output
DISK/LST module		○	○	○	○	○	○
System module (PUN:, RDR:, CON:)		○	○	×	×	×	×
User module	RS-232C	○	△	○	×	×	×
	SIO	○	×	×	○	×	×
	Cartridge SIO	○	×	×	×	○	×
	RS-232C input SIO output	○	×	×	×	×	○

○ : Availabe
↑ : Available only in the current mode
× : Not available

Programming notes

1) Relationship between the system and user modules and DISK or LST devices
System modules resets the parameters for the DISK or LST device (RS-232C or SIO) before accessing it when it is being used by a user module. Accordingly, the validity of the data received after the active serial device has been switched to the DISK or LST device that is being used by a system or user module. When the system or user module attempts to access the DISK or LST device again, the device is automatically restored into the previous state.

2) Relationship between system and user modules
When one module is using a serial device, another module cannot use even a different serial device. When a serial device is used by two or more modules:
   1. While a user modlue is using the serial device, a system module can use it in the mode set up for the user module.
   2. An error will be generated if a user module attempts to access a serial device which is being used by a system module. To use the same serial device, the user module must once close the serial device and clear the mode established by the system module.

5.3 Disk Drive Unit

5.3.1 General

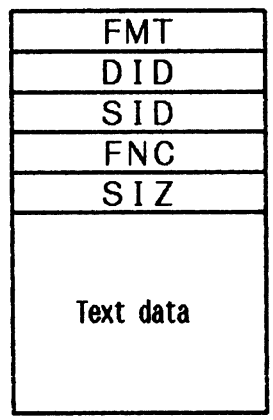
The PINE can connect to external floppy disk drive units via the SIO interface. Data is read from or written onto a floppy disk in 1K-byte units.

Disk drive units TF-20, TF-15, and PF-10 are available. Communication between the PINE and the FDD (floppy disk drive unit) is established in accordance with the EPSON Serial Communication Protocol (EPSP).

5.3.2 EPSON Serial Communication Protocol (EPSP)

Data is transferred between the PINE and the FDD according to a protocol called EPSP (EPSON Serial Communication Protocol).

EPSP has the general format shown below.



FMT: Identifies the header block format.
- 00H: Indicates message transmission from the main unit (PINE).
- 01H: Indicates message transmission from the FDD.

DID: Destination device ID.
- 31H: First FDD (Drive D: or E:)
- 32H: Second FDD (Drive F: or G:)

SID: Source device ID
This field contains 23H if the message (command) is from the PINE.

FNC: Command for FDD.

SIZ: Indicates the text data length. The value in this field is the length of the actual text data minus 1.

Text data returned from the FDD ends with a return code.

Tables 5.3.1 and 5.3.2 show the floppy disk drive commands and return codes, respectively.

Table 5.3.1 Commands

No.	Command name	FNC	Function
1	RESET	ØDH	Reset terminal floppy
2	READ	77H	Read disk directly
3	WRITE	78H	Write disk directly
4	WRITEHST	79H	Flush buffer
5	COPY	7AH	Copy volume
6			
7	FORMAT	7CH	Format disk

Table 5.3.2 Return Codes

Return code	Meaning
ØØH	Normal termination
ØFAH	BDOS error (read error)
ØFBH	BDOS error (write error)
ØFCH	BDOS error (drive select error)
ØFDH	BDOS error (write protect error)
ØFEH	

(1) RESET
Function: Resets the disk drive floppy.
Send data:

< Send data >

+0	00H	(FMT)
1	SS	(DID)
2	MM	(SID)
3	0DH	(FNC)
4	00H	(SIZ)
5	00H	

< Receive data >

+0	01H	(FMT)
1	MM	(DID)
2	SS	(SID)
3	0DH	(FNC)
4	00H	(SIZ)
5	Return code	

Description:
The RESET command causes the FDD to initialize itself and wait for an ENQ block. The FDD sends return code ØØH to the main unit.

(2) READ
Function: Reads a disk directly.
Send data:

< Send data >

+0	00H	(FMT)
1	SS	(DID)
2	MM	(SID)
3	77H	(FNC)
4	02H	(SIZ)
5	Drive code	
6	Track number	
7	Sector number	

Drive code = 1 or 2
Track number = 0 - 39
Sector number = 1 - 64

< Receive data >

+00H	01H	(FMT)
01H	MM	(DID)
02H	SS	(SID)
03H	77H	(FNC)
04H	80H	(SIZ)
05H	Read data	
06H	Read data	
		128 bytes
85H	Read data	
86H	Return code	

Description:
The READ command causes the FDD to transfer to the system the data (128 bytes) read from the specified sector on the specified logical track of the disk as well as a return code.

(3) WRITE
Function: Writes a disk directly.
Send data:

< Send data >

+00H	00H	(FMT)
01H	SS	(DID)
02H	MM	(SID)
03H	78H	(FNC)
04H	83H	(SIZ)
05H	Drive code	
06H	Track number	
07H	Sector number	
08H	Write type	
09H	Write data	
0AH	Write data	
	⋮	
+88H	Write data	

128 bytes

< Receive data >

01H	(FMT)
MM	(DID)
SS	(SID)
78H	(FNC)
00H	(SIZ)
Return code	

Drive code = 1 or 2
Track number = 0 - 39
Sector number = 1 - 64
Write type = 0 - 2

Description:
The WRITE command causes the FDD to write the specified data (128 bytes) to the location on the disk addressed by the specified logical track and sector numbers.

Write type = 00H: Standard write (The FDD blocks data before a write.)
= 01H: Flush buffer (The FDD immediately writes data on the FDD without blocking.)
= 02H: Sequential write (The FDD simply blocks data for a high-speed write.)

(4) WRITEHST
Function: Flushes the buffer.
Send data:

< Send data >

+00H	00H	(FMT)
01H	SS	(DID)
02H	MM	(SID)
03H	79H	(FNC)
04H	00H	(SIZ)
05H	00H	

< Receive data >

+00H	01H	(FMT)
01H	MM	(DID)
02H	SS	(SID)
03H	79H	(FNC)
04H	00H	(SIZ)
05H	Return code	

Description:
The WRITEHST command causes the FDD to flush the contents of the 1K-byte host buffer (filled by the WRITE command) onto the disk.

(5) COPY
Function: Copies a volume.
Send data:

< Send data >

+00H	00H	(FMT)
01H	SS	(DID)
02H	MM	(SID)
03H	7AH	(FNC)
04H	00H	(SIZ)
05H	Drive code	

< Receive data >

01H	(FMT)
MM	(DID)
SS	(SID)
7AH	(FNC)
02H	(SIZ)
Track number of the currently copying track (high-order byte)	
Track number of the currently copying track (low-order byte)	
Return code	

Track number of the currently copying track = Ø - 39
ØFFFFH: End
Description:
The COPY command causes the FDD to copy the entire disk in the specified drive onto another disk in the same drive unit. This command is not available if the unit has only one drive.

(6) FORMAT
Function: Format disk
Send data:

< Send data >

+00H	00H	(FMT)
01H	SS	(DID)
02H	MM	(SID)
03H	7CH	(FNC)
04H	00H	(SIZ)
05H	Drive code	

Drive code = 0 or 1

< Receive data >

+00H	01H
01H	MM
02H	SS
03H	7CH
04H	02H
05H	Track number of the currently formatting track (high-order byte)
06H	Track number of the currently formattng track(low-order byte)
07H	Return code

Track number of the currently formatting track = Ø - 39
ØFFFFH: End
Description:
The FORMAT command causes the FDD to format two tracks and return the corresponding track number (logical number) and a return code to the system. The FDD continues formatting in two track units and sets the logical track number in the return message to ØFFFFH when it completes formatting.

5.3.3 Use

In PINE OS, data is read from or written to the FDD through BIOS. However, the user can communicate directly with the FDD through application programs.

Use the following system-supplied utilities when communicating directly with the FDD through application programs:

- EPSPSND (Utility for sending EPSP data)
- EPSPRCV (Utility for receiving EPSP data)

See Section 4.2, "Jump Tables" for the use of these utilities.

5.4 External RAM Disk

5.4.1 General

The PINE external RAM disk is an optional external read/write storage device. It connects to the PINE main unit via the system bus interface.

The external RAM disk can contain up to 128K bytes of RAM and a 1M-bit CMOS mask ROM or EPROM. ROM is installed as a capsule and can be replaced from the exterior of the PINE.

In operation mode (i.e, when PINE power is on), the external RAM disk draws power from the battery in the PINE main unit. In non-operation mode (when PINE power is off), it draws backup power from the battery in the external RAM disk.

The external RAM disk is always attached to the PINE main unit. Its contents cannot be guaranteed if it is removed from the main unit.

5.4.2 Hardware General

5.4.2.1 Block diagram

The block diagram of the external RAM disk is shown in Figure 5.4.1.

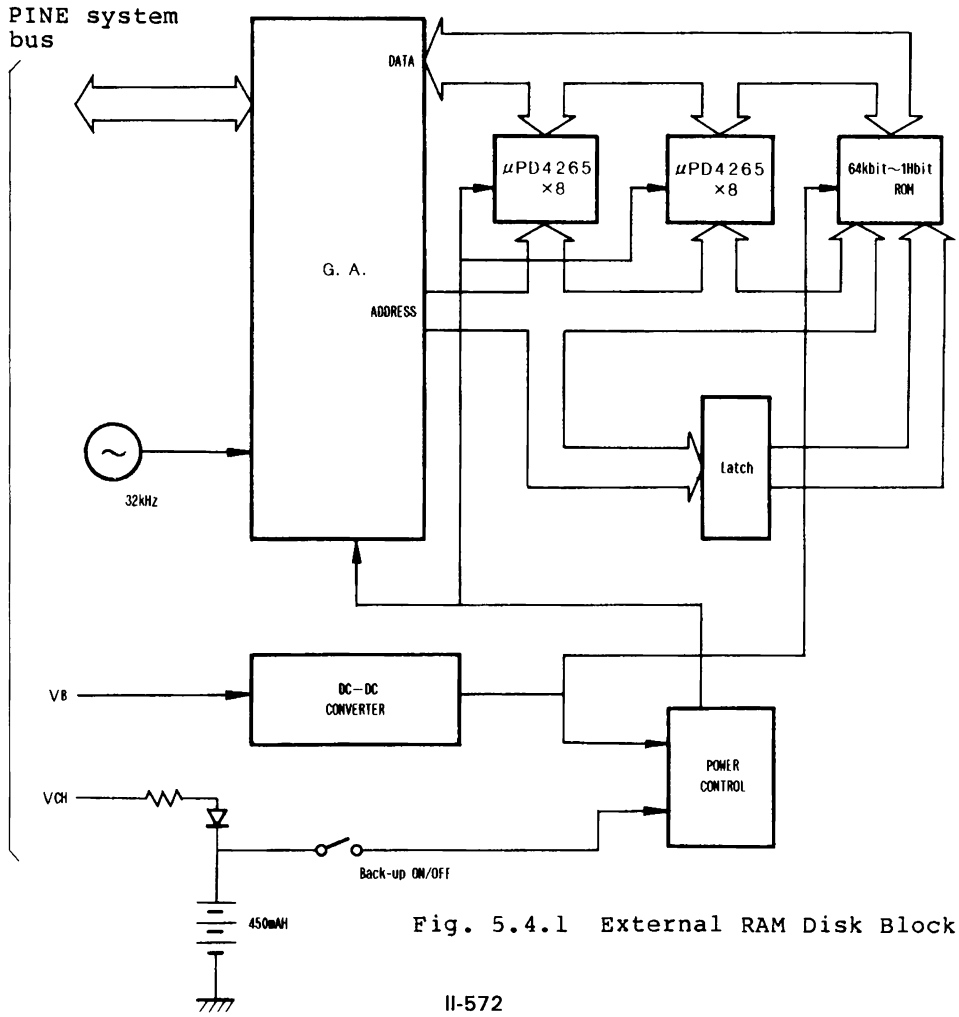


Fig. 5.4.1 External RAM Disk Block Diagram

5.4.2.2 Interface with the main unit

The external RAM disk connects to the main unit via the following signal lines out of the system bus:

Signal name	IN/OUT	Number of pins	Logic	Description
DB7 - Ø	IN/OUT	8	Positive	Data bus
AB7 - Ø	IN	8	Positive	Address bus
$\overline{MRQ}$	IN	1	Negative	Z-8Ø memory request signal
$\overline{IORQ}$	IN	1	Negative	Z-8Ø I/O request signal
$\overline{RD}$	IN	1	Negative	Z-8Ø read signal
$\overline{WR}$	IN	1	Negative	Z-8Ø write signal
$\overline{WAIT}$	OUT	1	Negative	Z-8Ø wait signal
$\overline{HLTA}$	IN	1	Negative	Z-8Ø halt signal
$\overline{M1}$	IN	1	Negative	Z-8Ø M1 signal
$\overline{CLK}$	IN	1	Negative	System clock reverse signal
DW	IN	1	-	DRAM refresh control signal
DCAS	IN	1	-	DRAM refresh control signal
$\overline{RS}$	IN	1	Negative	System reset signal
PON	IN	1	Positive	Power-on signal (turns on the DD converter.)
OFF	IN	1	Positive	Off signal
VCH	IN	1	-	Battery charge power
VBK	IN/OUT	1	-	Backup power
VB1	IN	1	-	Main battery power
GND	-	1	-	Signal ground

5.4.3 Structure

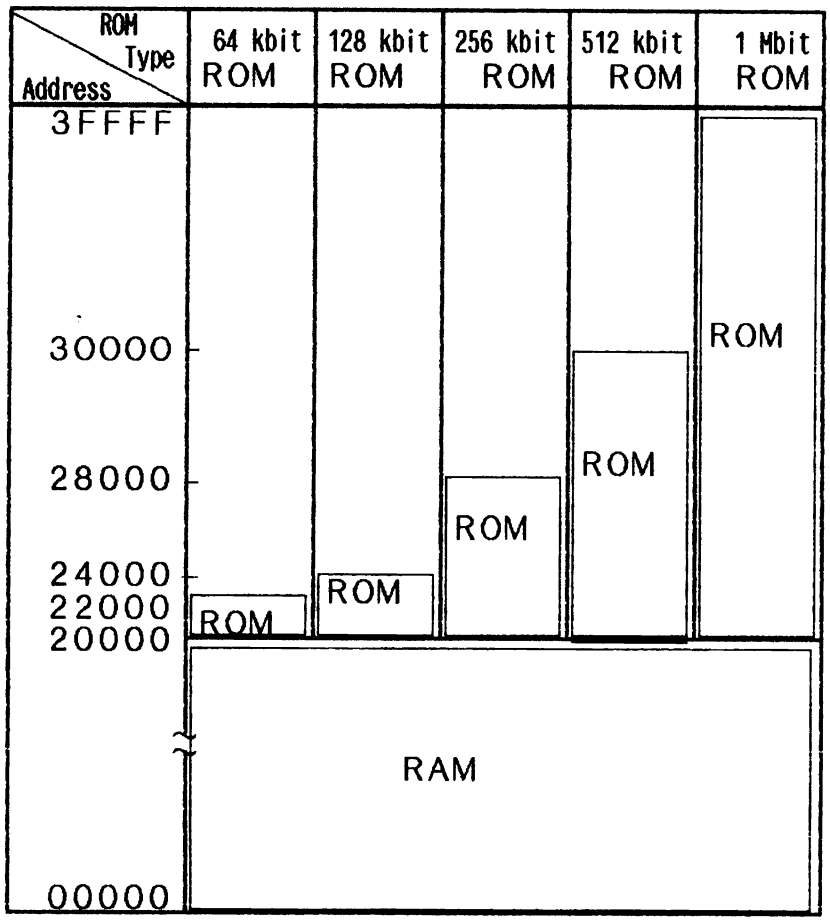
5.4.3.1 External RAM disk I/O address map

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Remarks
Read	90H										
	91H										
	92H										
	93H	EXTIR	8 bit data								
	94H	EXTSR	EXT						(OPN)	(WP)	
Write	90H	EXTAR	A7	A6	A5	A4	A3	A2	A1	A0	
	91H	EXTAR	A15	A14	A13	A12	A11	A10	A9	A8	
	92H	EXTAR	×	×	×	×	×	A18	A17	A16	
	93H	EXTOR	8 bit data								
	94H	EXTCR	×	×	×	×	×	×	OPN	WP	

The x mark stands for "Don't care."

EXTIR: External RAM disk input register
EXTSR: External RAM disk status register
OPN = 1: RAM opened
= Ø: RAM closed
WP = 1: Write protected
= Ø: Write enabled
EXT = 1: No external RAM disk installed
= Ø: External RAM disk installed
EXTAR: External RAM disk address register
EXTOR: External RAM disk output register
EXTCR: External RAM disk command register (WP and OPN have the same meaning as for EXTSR.)

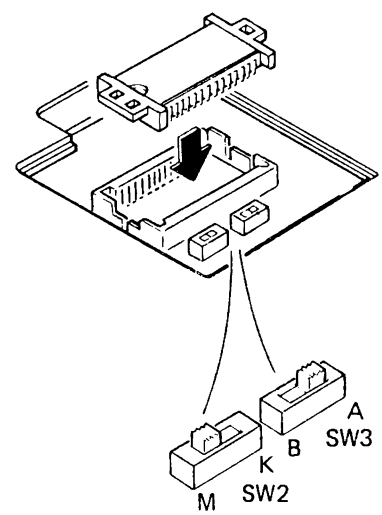
5.4.3.2 External RAM disk memory map



5.4.3.3 Miscellaneous

(1) Switch settings
The external RAM disk has two switches, SW2 and SW3, in the box on its rear. These switches are used to select ROM capacities.

ROM type	SW2	SW3
64K bit (e.g., 27C64)	K	B
128K bit	K	B
256K bit (e.g., 27C256)	K	B
512K bit	M	B
1M bit (e.g., HN62301)	M	A



(2) Addresses in 256K-bit ROM
In ROMs excluding 256K-bit ROMs, logical ROM addresses have a one-to-one correspondence with physical ROM addresses. Therefore, data are written into ROM sequentially from lower to higher addresses. In 256K-bit ROMs, however, the first half of the logical address space is mapped into the second half of the physical address space, and the second half of the logical address space is mapped into the first half of the physical address space (see the figure below).

Address on external RAM disk	Physical ROM address
20000H	4000H
23FFFH	7FFFH
24000H	0000H
27FFFH	3FFFH

(3) Backup battery ON/OFF switch
The external RAM disk is furnished with a switch for turning on or off the connection with a backup battery. When this switch is on, the external RAM disk is always supplied with a small quantity of power from the battery.

(4) Power
When the PINE main unit power is on, PINE battery power is regulated by the DC to DC convertor in the external RAM disk unit and applied to the circuit. When the main unit power is off, power is applied to the circuit from the backup battery in the external unit.

The battery in the external RAM disk is charged at the same time as the NiCd battery in the main unit. Note that charging of the battery in the external RAM disk is not guaranteed if the battery in the main unit is an Mn battery and used without being connected to an AC adaptor.

When the main unit power is off, the main unit and the external RAM disk are each supplied with power from their own backup batteries, respectively. If, however, the voltage of either battery drops below a specified value, that battery is supplied with power from the other battery. Therefore, both the main unit and the optional RAM disk are guaranteed a similar backup time.

5.4.4 Use

Memory (RAM or ROM) can be accessed by loading the desired address into EXTAR (P90H - P92H) and reading or writing the data in EXTIR (P93H) or EXTOR (P93H), respectively.

The address counter is automatically incremented after each read or write. Only the lowest 8 bits of the address counter are incremented; the higher bits remain unchanged. Therefore, once an address is specified, up to 256 contiguous data bytes can be read from or written into RAM or ROM, without respecifying the address.

5.4.4.1 Reading/writing RAM

Figure 5.4.2 shows the procedure for reading/writing RAM.

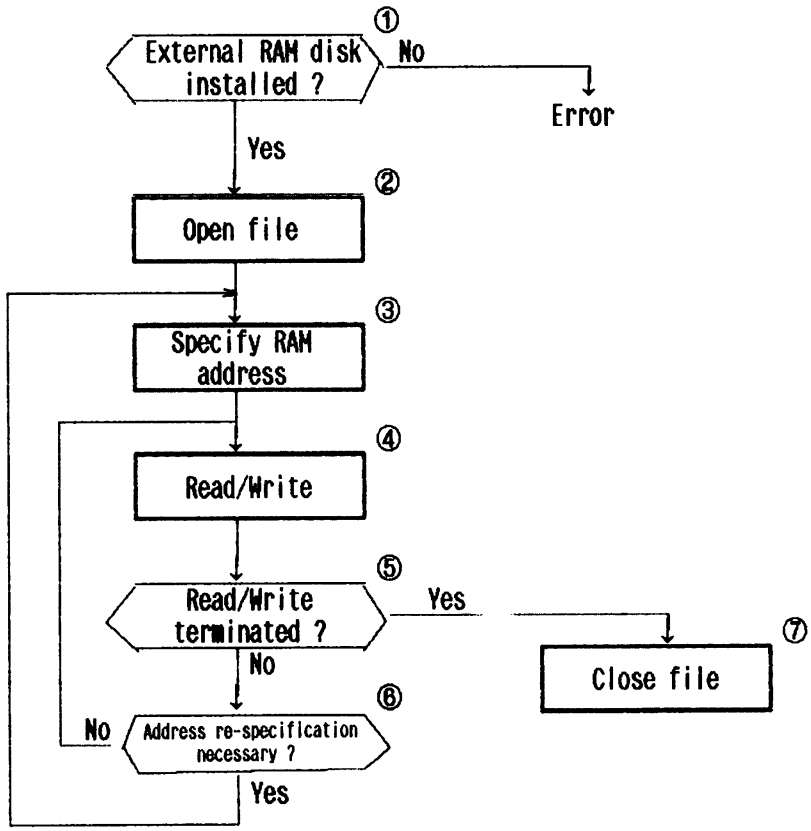


Fig. 5.4.2 Procedure for Reading/writing RAM

Step 1: External RAM disk installed?
Check whether or not an external RAM disk is installed by reading I/O port address 94H. The MSB indicates the presence or absence of an external RAM disk.
MSB = 1: External RAM disk installed.
= Ø: No external RAM disk installed.

Step 2: Open file.
Write 1 to I/O port address 93H, bit 1 to open the external RAM disk for read or write. Cancel write protection of RAM, if necessary.

Step 3: Specify RAM address.
Load I/O port addresses 9ØH through 92H with the address to be read or written. The RAM address may be in the range of ØØØØØH through 1FFFFH.

Step 4: Read or write.
Access the I/O port address 93H for read or write.

Steps 5 and 6: Loop.
Determine whether or not read/write processing has been terminated in the application program. During sequential read/write, the external RAM disk auto increment function allows up to 256 bytes of data to be read/written without address re-specification.

Step 7: Close file.
When read/write processing is completed, set I/O port address 93H, bit 1 to Ø to close the external RAM disk. Write-protect the RAM, if necessary.

5.4.4.2 Reading ROM

Figure 5.4.3 illustrates the procedure for reading ROM.

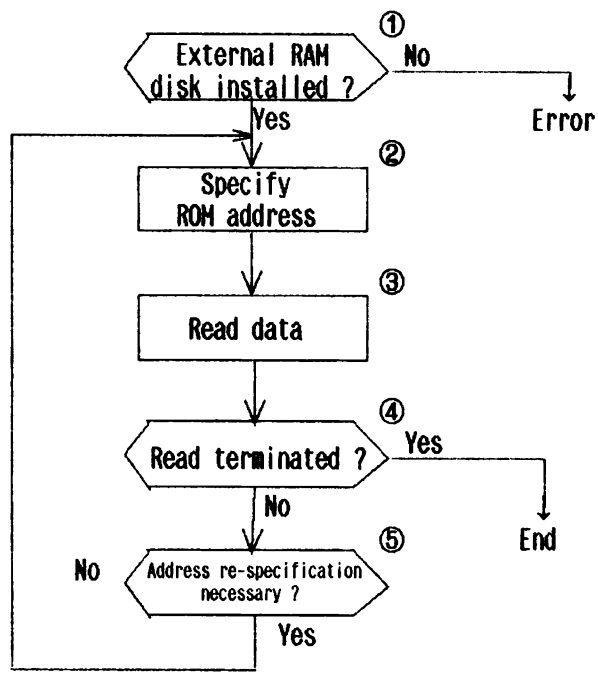


Fig. 5.4.3 Procedure for Reading ROM

Step 1: External RAM disk installed?
Check whether or not an external RAM disk is installed by reading I/O port address 94H. The MSB identifies the presence or absence of an external RAM disk.
MSB = 1: External RAM disk installed.
= Ø: No external RAM disk installed.

Step 2: Specify ROM address.
Load I/O port addresses 9ØH through 92H with the address from which data is to be read. The ROM address must be 20ØØØH or up.

Step 3: Read data.
Access I/O port address 93H to read data.

Steps 4 and 5: Loop.
Determine whether or not read processing has been terminated in the application program. During sequential read, the external RAM disk auto increment function allows up to 256 bytes of data to be read without address re-specification.

5.4.4.3 Notes

(1) RAM open/close processing
To save power, the external RAM disk must be opened only when it is being accessed. While RAM is closed, it is refreshing itself and cannot be accessed.

ROM need not be opened for access.

(2) RAM write protection
When RAM is write protected, a write operation causes no actual write to RAM and does not increment the address register. However, RAM can be read, in which case the address register is incremented accordingly. Reading of ROM is not affected by write protection, either.

5.4.5 System Support

PINE OS supports the RAM portion of the external RAM disk as a read/write disk drive. See Section 3.8, "Disk Storage" for detailed information.

The OS does not support the ROM portion of the external RAM disk. Application programs must read data from this area for themselves.

5.4.5.1 Installation check

The OS checks for presence or absence of an external RAM disk at power-on or reset time. It examines the MSB of EXTSR (P94H) and, if an external RAM disk is found to be installed, treats the RAM in the external RAM disk as a RAM disk.

5.4.5.2 Opening/closing RAM

The OS opens and closes the RAM in the external RAM disk before and after a read or write. Therefore, the RAM is always closed after the OS performs a read/write operation.

5.4.5.3 RAM write protection

The OS automatically cancels write protection of RAM when reading/writing the RAM in the external RAM disk. Therefore, the RAM is not write protected after the OS performs a read/write operation.

5.4.5.4 Resetting during write to RAM

Write operation is not guaranteed if the RESET button is pressed during a write to the external RAM disk. The OS displays a "RAM DISK FORMAT" message to warn against possible destruction of RAM contents.

```
                        ************************************************
                                1 M BIT ROM READ PROGRAM
                        ************************************************

                        NOTE :
                                This sample program is reading 1 Mbit ROM,
                                and displaying the data.

                        <> assemble condition <>

                                .Z80

                        <> loading address <>

                                .PHASE  100H
                        ;
                        ;       <> constant values <>
                        ;
                        ;       BIOS entry
                        ;
EB03                    WBOOT           EQU     0EB03H          ; Warm Boot entry
EB09                    CONIN           EQU     WBOOT   +06H    ; Console input entry
EB0C                    CONOUT          EQU     WBOOT   +09H    ; Console out entry
                        ;
                        ;       System area
                        ;
F294                    LSCRVRAM        EQU     0F294H          ; VRAM top address.
                        ;
0003                    STOP            EQU     03H
000D                    CR              EQU     0DH
000A                    LF              EQU     0AH
000C                    CLS             EQU     0CH
001B                    ESC             EQU     1BH
0020                    SPACE           EQU     20H
                        ;
0090                    P90             EQU     90H             ; I/O port 91H
0091                    P91             EQU     91H             ; I/O port 91H
0092                    P92             EQU     92H             ; I/O port 92H
0093                    P93             EQU     93H             ; I/O port 93H
0094                    P94             EQU     94H             ; I/O port 94H
                        ;
                        ;       ************************************************
                        ;               MAIN PROGRAM
                        ;       ************************************************
                        ;
                        ;       NOTE :
                        ;               This program is reading 1 Mbit ROM,
                        ;               and displaying kanji font.
                        ;
0100                    MAIN:
0100    31 1000                 LD      SP,1000H        ; Set stack pointer.

0103    CD 0177                 CALL    CHKROM          ; Connect external RAM disk?
0106    DA EB03                 JP      C,WBOOT         ; No.

0109    CD 0188                 CALL    CUSROFF         ; Cursor off.
                        ;
010C                    MAIN10:
010C    0E 0C                   LD      C,CLS           ; Clear screen & home.
010E    CD EB0C                 CALL    CONOUT          ;

0111    2A F294                 LD      HL,(LSCRVRAM)   ; VRAM top addr --> HL
0114    06 04                   LD      B,4             ; Loop counter (4 lines)
                        ;
0116                    MAIN20:
0116    E5                      PUSH    HL              ; Save registers
0117    C5                      PUSH    BC              ;
0118    06 0F                   LD      B,15            ; Loop counter (15 characters)
011A                    MAIN30:
011A    E5                      PUSH    HL              ; Save registers
011B    C5                      PUSH    BC              ;
011C    06 10                   LD      B,16            ; Loop counter (16 dot lines)
                        ;
011E                    MAIN40:
011E    CD 014C                 CALL    READROM         ; Read 1 Mbit ROM.
0121    CD 017B                 CALL    WRTVRAM         ; Write the data to VRAM directly.
                        ;
0124    23                      INC     HL              ; VRAM pointer increment.
0125    CD 014C                 CALL    READROM         ; Read 1 Mbit ROM.
0128    CD 017B                 CALL    WRTVRAM         ; Write the data to VRAM directly.
                        ;
012B    11 001F                 LD      DE,31           ; Get next dot line address in VRAM.
012E    19                      ADD     HL,DE           ;
012F    10 ED                   DJNZ    MAIN40          ; Loop.

0131    C1                      POP     BC              ; Restore registers.
0132    E1                      POP     HL              ;
0133    23                      INC     HL              ; Get nect column address in VRAM.
0134    23                      INC     HL              ;
0135    10 E3                   DJNZ    MAIN30          ;
                        ;
0137    C1                      POP     BC              ; Restore registers
0138    E1                      POP     HL              ;
0139    11 0200                 LD      DE,32*16        ; Get next line address in VRAM.
013C    19                      ADD     HL,DE           ;
013D    10 D7                   DJNZ    MAIN20          ; Loop.
                        ;
013F    CD EB09                 CALL    CONIN           ; Key input wait.
0142    FE 03                   CP      STOP            ; STOP key?
0144    20 C6                   JR      NZ,MAIN10       ; No.
                        ;
0146    CD 017D                 CALL    CUSRON          ; Cursor on.
0149    C3 EB03                 JP      WBOOT           ; End.
                        ;
```

```
                        ;***************************************************
                        ;       READ DATA FROM 1 M BIT ROM
                        ;***************************************************
                        ;
                        ;NOTE :
                        ;       Read a data from 1 Mbit ROM.
                        ;       This routine uses the function of auto
                        ;       increment.
                        ;
                        ;<> entry parameter <>
                        ;       NON
                        ;<> return parameter <>
                        ;       A  : Read data.
                        ;   preserved registers <>
                        ;       NON
                        ;
                        ;       CAUTION
014C                    READROM:
014C  3A 0193                   LD      A,(P90DT)       ; Last 8 bits data.
014F  3C                        INC     A               ; 256 bytes read?
0150  32 0193                   LD      (P90DT),A       ;  Set the new address.
0153  20 1F                     JR      NZ,READ50       ; No.
                        ;
0155  3A 0194                   LD      A,(P91DT)       ; Middle 8 bits data.
0158  3C                        INC     A               ; 256*256 bytes read?
0159  32 0194                   LD      (P91DT),A       ;  Set the new address.
015C  20 07                     JR      NZ,READ40       ; No.
                        ;
015E  3A 0195                   LD      A,(P92DT)       ; Top 3 bits data.
0161  3C                        INC     A               ; Count up.
0162  32 0195                   LD      (P92DT),A       ; Set the new ROM address.
                        ;
0165                    READ40:
0165  3A 0193                   LD      A,(P90DT)       ; I/O port output for setting next address.
0168  D3 90                     OUT     (P90),A         ;  Last 8 bits
016A  3A 0194                   LD      A,(P91DT)       ;
016D  D3 91                     OUT     (P91),A         ;  Middle 8 bits.
016F  3A 0195                   LD      A,(P92DT)       ;
0172  D3 92                     OUT     (P92),A         ;  Top 8 bits.
                        ;
0174                    READ50:
0174  DB 93                     IN      A,(P93)         ; Read ROM data.
0176  C9                       RET                     ;
                        ;
                        ;***************************************************
                        ;       CHECK CONNECTING EXTERNAL RAM DISK
                        ;***************************************************
                        ;
                        ;NOTE :
                        ;
                        ;<> entry parameter <>
                        ;       NON
                        ;<> return parameter <>
                        ;       CY : Return information.
                        ;          =0 -- Connected RAM disk.
                        ;          =1 -- Not connected RAM disk.
                        ;<> preserved registers <>
                        ;       NON
                        ;
                        ;CAUTION :
                        ;
0177                    CHKROM:
0177  DB 94                     IN      A,(P94)         ; Get external status.
0179  17                       RLA                     ; MSB --> CY
017A  C9                       RET                     ;
                        ;
                        ;       WRITE VRAM AT THE HL ADDRESS.
                        ;
017B                    WRTVRAM:
017B  77                        LD      (HL),A
017C  C9                        RET
                        ;
                        ;       CURSOR ON
                        ;
017D                    CUSRON:
017D  0E 1B                     LD      C,ESC
017F  CD EB0C                   CALL    CONOUT
0182  0E 33                     LD      C,'3'
0184  CD EB0C                   CALL    CONOUT
0187  C9                        RET
                        ;
                        ;       CURSOR OFF
                        ;
0188                    CUSROFF:
0188  0E 1B                     LD      C,ESC
018A  CD EB0C                   CALL    CONOUT
018D  0E 32                     LD      C,'2'
018F  CD EB0C                   CALL    CONOUT
0192  C9                        RET
                        ;
0193                    P90DT:
0193  FF                        DB      0FFH
0194                    P91DT:
0194  FF                        DB      0FFH
0195                    P92DT:
0195  01                        DB      001H
                        ;
                                END
```

5.5 Barcode Reader and External Cassettes

5.5.1 General

The PINE is furnished with barcode reader and external cassette interfaces. The barcode reader interface has +5V power output and the barcode reader input (BCRD). The external cassette interface handles the motor drive signal (RMT) for cassettes with a remote terminal, the record signal (MIC) to cassettes, and the earphone signal (EAR) from cassettes.

Neither barcode reader nor external cassette is supported by PINE OS.

5.5.2 Structure

5.5.2.1 Barcode reader/external cassette I/O address map

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Remarks
Read	P02H	ICRL.B	ICR on state transition of the signal (low-order byte)								
	P03H	ICRH.B	ICR on state transition of the signal (high-order byte)								
	P04H	ISR						ICF			Bits other than bit 2 are used for other purposes.
	P05H	STR							BCRD	EAR	Bits other than bits 1 and 0 are used for other purposes.
Write	P00H	CTLR1					SW BCR	BCR1	BCR0	SL BCR	Bits other than bits 3 through 0 are used for other purposes.
	P02H	CTLR2	Don't Care						RMT	MIC	
	P04H	IER						IER2			Bits other than bit 2 are used for other purposes.

ICRL.B, ICRH.B: 16 bit data latched from FRC to ICR by transition in the state of the signal from the barcode reader or external cassette.

ISR (Interrupt Status Register):
ICF: Interrupt signal generated when the FRC value is latched to ICR by transition in the state of the signal from the barcode reader or external cassette.

STR (Status Register):
BCRD: Barcode reader input signal
EAR: External cassette EAR input signal

CTLR1 (Control Register 1)
SWBCR: Turns on/off power (+ 5V) to the barcode reader.
= Ø: Power off.
= 1: Power on.
BCR1, BCRØ: Selects the latch trigger polarity.
= ØØ: Trigger inhibited.
= Ø1: Falling trigger (_┌┐_)
= 1Ø: Rising trigger (_┐└_)
= 11: Rising and falling triggers (_┐┌┐_)
SLBCR: Selects the trigger signal that is to latch the FRC value to ICR.
= Ø: External cassette EAR input signal
= 1: Barcode reader input signal
CTLR2 (Control Register 2)
RMT: External cassette remote ON/OFF signal
= Ø: Remote OFF
= 1: Remote ON
MIC: External cassette write signal (output to microphone)
IER (Interrupt Enable Register)
IER2: Controls INT2 (ICF) interrupts.
= Ø: Interrupts disabled.
= 1: Interrupts enabled.

5.5.3 Use

This subsection discusses the method of determining the wave form of the input signal from the barcode reader or external cassette, as well as the method of sending signals to the external cassette.

5.5.3.1 Selecting the barcode reader or external cassette

The barcode reader or external cassette is initialized in the following steps:

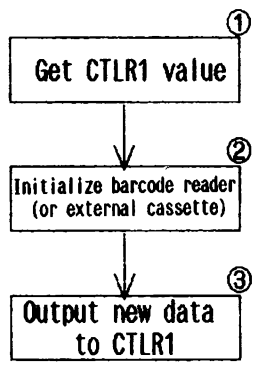


Fig. 5.5.1 Initialize Procedure

Step 1: Get CTLR1 value.
Read the current CTLR1 (PØØH) value from the system area RZCTLR1 (ØFØØ1H).
Step 2: Initialize barcode reader (or external cassette).
Manipulate the value read in step (1) as follows:
Bits 7 - 4: Do not change.
Bits 3 - Ø: Specify whether power to the barcode reader is to be turned on or off, the trigger polarity, and the trigger signal.
Step 3: Output data to CTLR1.
Load the value specified in step (2) into RZCTLR1 (ØFØØ1H) and output it to CTLR1 (PØØH).

5.5.3.2 Data input

Whether or not data is input from the barcode reader or external cassette is determined by measuring the input signal 1 or Ø state time. The time can be measured by the procedure shown in Figure 5.5.2 (the procedure for measuring the time of the mark signal).

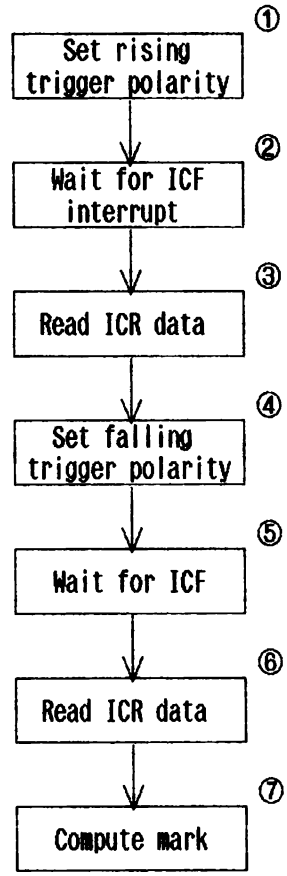


Fig. 5.5.2 Time Measuring Procedure

Step 1: Set rising trigger polarity.
Set CTLR1 (P00H), bits 2 and 1 to 1 and 0, respectively, using the procedure shown in Figure 5.5.1. Make a dummy read or ICRH.B (P03H), if necessary, and reset the ICF interrupt.
Step 2: Wait for ICF interrupt.
Read the value of ISR (P04H) to check for an ICF interrupt. ISR, bit 2 identifies the presence or absence of an ICF interrupt.
ISR (P04H), bit 2 = 0: No ICF interrupt generated.
= 1: ICF interrupt generated.
If no ICF interrupt has been generated, repeat this until one is generated.
Step 3: Read ICR data.
Read ICRL.B (P02H) and ICRL.B (P03H) and get the FRC value at input signal rise time.
Step 4: Set falling trigger polarity.
Set CTLR1 (P00H), bits 2 and 1 to 0 and 1, respectively, using the procedure shown in Figure 5.5.1.
Steps 5 and 6:
Same as steps (2) - (3).
Step 7: Compute mark signal time.
Compute the mark signal time from the formula
(T2 - T1)/614.6 ms.
where T1 and T2 are the values obtained in steps (3) and (6), respectively. If T2 -1 T1 <= 0, add 65536 to the result to get a positive value.

Note: Since FRC is set to approximately 106.6 ms. frequency, time exceeding one frequency (more than 106.6 ms.) may elapse while waiting for an ICF interrupt in step (2). Therefore, an application program must include frequency count check and timeout processing routines in its wait processing routine. The frequency count can be checked using the OVF interrupt.

5.5.3.3 Data output (output to MIC of the external cassette)

Data can be output to the MIC terminal of the external cassette by setting CTLR2 (P02H), bit 0 to 1 or 0 and outputting mark or space signals.

The REMOTE terminal can also be controlled through CTLR2 (P02H), bit 1.

5.5.4 System Support

PINE OS supports neither the barcode reader nor the external cassette. For the ICF interrupt, however, a hook is provided in the corresponding interrupt processing routine to allow expansion of ICF interrupt processing.

The external cassette is supported by BASIC.

5.6 ROM Capsules

5.6.1 General

The PINE can house up to two capsule ROMs in memory, in addition to OS ROM (32K bytes) and RAM (64K bytes). ROM capsules may be 8K-, 16K-, or 32K-byte mask ROMs. Data in these capsules can be accessed immediately through bank switching.

5.6.2 Structure

5.6.2.1 ROM capsule I/O address map

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Remarks
Read	P05H	STR	BANK3	BANK2	BANK1	BANK0	Bits 3 through 0 are used for other purposes.				
Write	P05H	BANKR	BANK3	BANK2	BANK1	BANK0	Bits 3 through 0 are used for other purposes.				

STR (Status Register)
BANK3 - Ø: Indicate the current bank status.
BANKR (Bank Register)
BANK3 - Ø: Select a bank.

5.6.2.2 Memory map

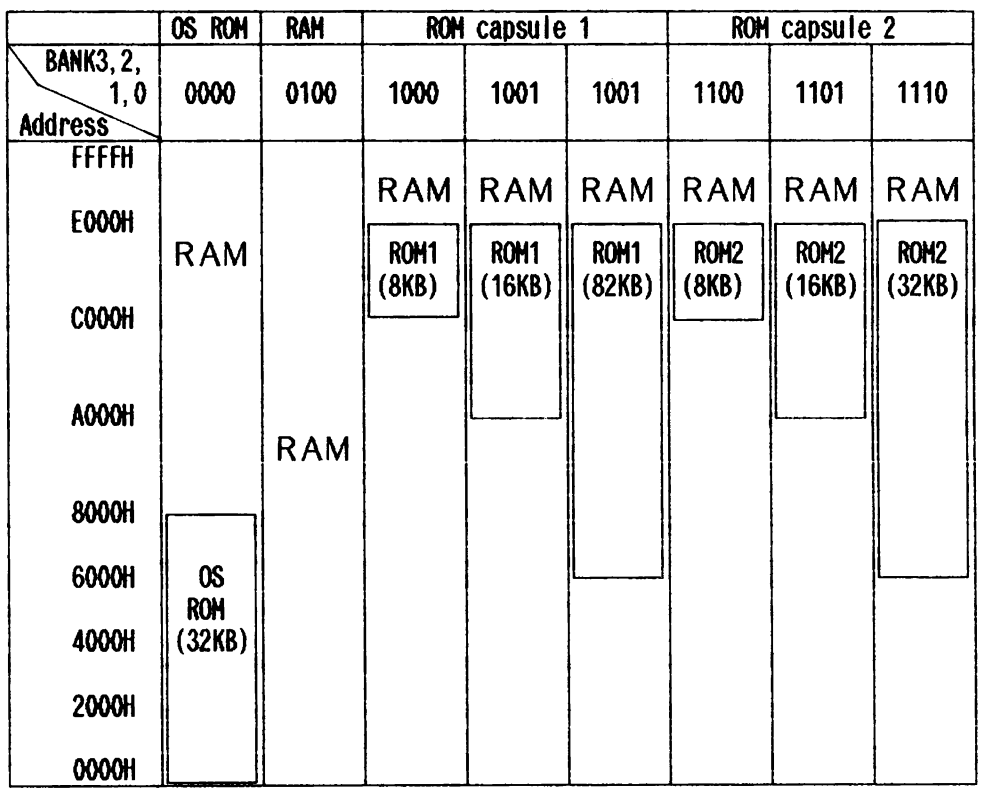


Fig. 5.6.1 PINE ROM Capsule Memory Map

5.6.2.3 ROM addresses

In ROMs 1 and 2 in the ROM capsule, the relationship between logical and physical addresses differs depending on the ROM capacity.

(1) Addresses in 64K-bit ROM
In 64K-bit ROMs, logical ROM addresses have a one-to-one correspondence with physical ROM addresses.

Memory address	Physical ROM address
0C000H	0000H
|	|
0DFFFH	1FFFH

(2) Addresses in 128K-bit ROM
In 128K-bit ROMs, the first half of the logical address space is mapped into the econd half of the physical address space, and the second half of the logical address space is mapped into the first half of the physical address space (see the figure below).

Memory address	Physical ROM address
0A000H	2000H
|	|
0BFFFH	3FFFH
0C000H	0000H
|	|
0DFFFH	1FFFH

(3) Addresses in 256K-bit ROM
In 256K-bit ROMs, the first half of the logical address space is mapped into the econd half of the physical address space, and the second half of the logical address space is mapped into the first half of the physical address space (see the figure below).

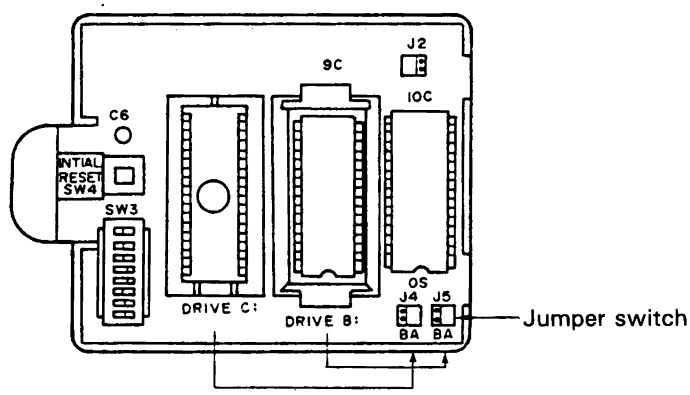
Memory address	Physical ROM address
6000H	6000H
|	|
7FFFH	7FFFH
8000H	0000H
|	|
DFFFH	5FFFH

5.6.2.4 ROM switch jumper

When installing 64K-, 128K-, or 256K-bit ROM devices in ROM capsules, change the settings for J4 and J5 according to the ROM type.

Table 5.6.2 ROM Capacity and Jumpers

J4 and J5	ROM type
A	128K bit (PROM)
B	64K bit
B	128k bit (Mask ROM)
B	256K bit



Jumper J4 is for ROM capsule 1 (drive C) use and Jumper J5 is for ROM capsule 2 (drive B) use.

5.6.3 Use

Figure 5.6.1 shows the procedure for reading data in ROM capsule through bank switching.

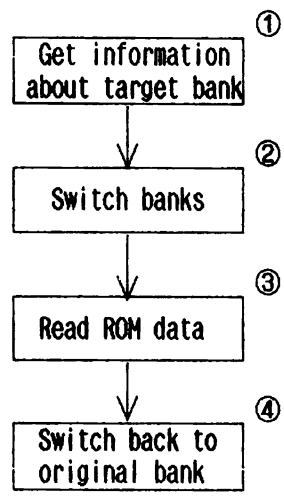


Fig. 5.6.1 Procedure for Reading ROM Data

See Section 4.4, "Bank Switching" for steps (1), (2), and (4).

5.6.4 System Support

PINE OS supports ROM capsules as read-only disks or executable programs. See Section 3.8, "Disk Storage" for the method of supporting ROM capsules as disks. See Section 4.6, "Executing a ROM Program" for the executing ROM programs.

5.6.4.1 Installation check

PINE OS checks whether or not a ROM capsule is installed at power-on or reset time.

At power-on or reset time, PINE OS determines whether or not a ROM capsule is installed by checking for a ROM header.

5.7 Centronics Interface

5.7.1 General

The PINE employs the Centronics interface which allows the PINE to print on an ordinary terminal printer (MP, RP, or FP) without processing print data.

5.7.2 Structure

5.7.2.1 Centronics interface I/O address map

R/W	I/O address	Register name	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0	Remarks
Read	16H	IOSTR							PERR	PBSY	Bits 7 through 2 are used for other purposes
Write	17H	PDR	8-bit data								
	19H	IOCTLR							$\overline{\text{PINI}}$	PSTB	Bits 7 through 2 are used for other purposes

IOSTR (I/O Status Register)
PERR: Printer error signal
= 0: Normal state.
= 1: Error occurred.
PBSY: Printer busy signal
= 0: Printer ready.
= 1: Printer busy.
PDR (Printer Data Register)
Used to output data to the printer. Loaded with 0FFH when not in use.
IOCTLR (I/O Control Register)
$\overline{\text{PINI}}$: Printer initial reset signal
PSTB: Data strobe signal

Note: The printer can be reset by holding the *PINI signal in the 0 state for more than 50 us.

5.7.3 Use

Figure 5.7.1 shows the procedure for using the Centronics interface.

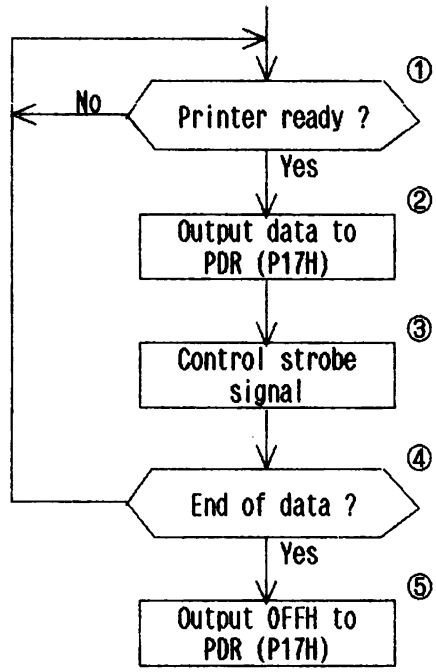


Fig. 5.7.1 Data Transmission Procedure

Step 1: Printer ready?
Check whether or not the printer is ready by examining PBSY and PERR (IOSTR, bits Ø and 1). Ø in both bits indicates that the printer is ready.
Step 2: Output data.
Write to PDR (P17H) the data to be output to the printer.
Step 3: Control strobe signal.
Generate the strobe signal to cause the printer to read data. To do this, set PSTB (IOCTLR, bit Ø) to 1 and then restore it to Ø.
Steps 4 and 5: Termination processing.
When outputting data continuously, return to step (1). When data output processing is completed, send ØFFH to PDR (P17H) to save power.

5.7.4 System Support

In PINE OS, the Centronics interface is manipulated through the LIST and LISTST functions. See Section 3.4, "BIOS Details" for BIOS functions.

5.7.4.1 Centronics interface initial signal

The PINE initially resets the printer at power-on or reset time by sending an initial signal to the Centronics interface. Even if power is turned off in the continue mode during printing, the printer is reset the next time power is turned on; print operation cannot be continued, once power is turned off.

5.8 Printer

5.8.1 General

PINE OS supports printers incorporating the Centronics or RS-232C interface. Any type of printer connects to the PINE Centronics, RS-232C, or SIO interface. The cartridge printer, which is one of the PINE cartridge options, is always connected to the cartridge interface.

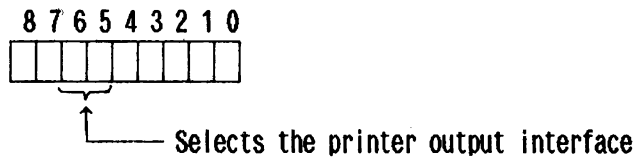
See Section 5.1, "Cartridges" for the cartridge printer, and Section 5.7, "Centronics Interface" for the Centronics interface. For the RS-232C and SIO interfaces, see Section 5.2, "Serial Interfaces."

The following pages discuss printers in general.

5.8.2 System Support

5.8.2.1 Selecting printer interface

The PINE can connect to a printer via four types of interfaces: Centronics, RS-232C, SIO, and cartridge SIO. The interface to which the printer is to be connected is determined by bits 5 and 6 of the DIP switch on the PINE main unit rear panel.

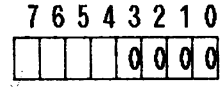


Bit 6	Bit 5	Output interface
OFF	OFF	SIO
OFF	ON	Cartridge
ON	OFF	RS-232C
ON	ON	Printer

5.8.2.2 Printer interface parameters

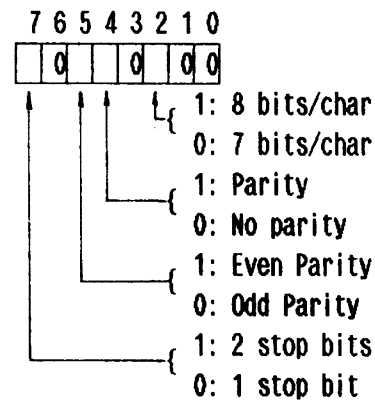
The bit rate must be specified when outputting print data to the RS-232C or SIO. In PINE OS, the initial settings are 4800 bps, 8 data bits, no parity, and 2 stop bits. These parameter values can be changed by rewriting the following areas:

SYSCTLR1 (0F279H) 1 byte
Specifies the bit rate.



bit 7	bit 6	bit 5	bit 4	Transmission speed	bit 7	bit 6	bit 5	bit 4	Transmission speed	bit 7	bit 6	bit 5	bit 4	Transmission speed
0	0	0	0	110	0	1	0	1	2400	1	0	0	0	1200
0	0	0	1	150	0	1	1	0	4800	1	0	0	1	75
0	0	1	0	300	0	1	1	1	9600	1	1	×	×	200
0	0	1	1	600	1	0	1	0	19200					
0	1	0	0	1200	1	0	1	1	38400					

SYSARTMR (0F27AH) 1 byte
Specifies the send and receive parameters.



5.8.2.3 Output to the printer

Data is output to the printer through the BIOS LIST and LISTST functions. See Section 3.4, "BIOS Details" for further information.

5.8.2.4 Screen dump

In PINE OS, the data on the LCD screen can be output on the printer in bit image by pressing the CTRL/PF5 keys or by calling the BIOS SCRNDUMP function.

5.8.2.5 International character set specification processing

PINE OS automatically specify the international character set (ESC + 'R') for the printer.

The international character set specification is output when data is sent to the printer for the first time after:
- The LST: field of the I/O byte is changed,
- A warm boot is performed,
- Power is turned on, or
- The display character generator is changed.

5.8.3 Printer Types

Table 5.8.1 lists the printers available for the PINE.

Table 5.8.1 Available Printers

Printer type	PINE graphics character	PINE screen dump	Connectable interface
MX-80II, III	$ *	$ **	Centronics (RS-232C, SIO)
MX-100II, III	x	o	Centronics (RS-232C, SIO)
MX-80, 100	o	o	Centronics (RS-232C, SIO)
FX-80, 100	Down load	o	Centronics (RS-232C, SIO)
LQ1500	x	o	Centronics
SQ2000	x	o	Centronics
DX-100	x	x	Centronics ***
C-40	x	o	RS-232C, SIO
P-40P	o	o	Centronics
P-40S	o	o	RS-232C, SIO
P-80	o	o	RS-232C, SIO
Cartridge printer	o	o	Cartridge

*: The $ marks indicate that the corresponding functions are available if PROM for the HC-20 is used.
**: The screen dump function is not available for the MP-80 I.
***: The optional printer adaptor must be used.
Notes:
1) Characters E0H, E1H, and 7FH cannot be printed on any of the above printers.
2) The user is advised to use the Centronics interface for ordinary terminal printers (MP, FP, and RP).

CHAPTER 6 MEMORY CONTENTS

6.1 Memory Maps II-596
6.1.1 General II-596
6.1.2 RAM Memory Map II-596

6.2 System Area Details II-600
6.2.1 General II-600
6.2.2 System Area I (RSYSAR1) II-601
6.2.3 System Area II (RSYSAR2) II-613
6.2.4 System Area III (RSYSAR3) II-639
6.2.5 System Area IV (RSYSAR4) II-647
6.2.6 System Area V (RSYSAR5) II-675

CHAPTER 6 MEMORY

6.1 Memory Map

6.1.1 General

The PINE has 64K bytes of RAM, 32K bytes of OS ROM and two 32K-byte ROM capsules. Each memory can be accessed through bank switching.

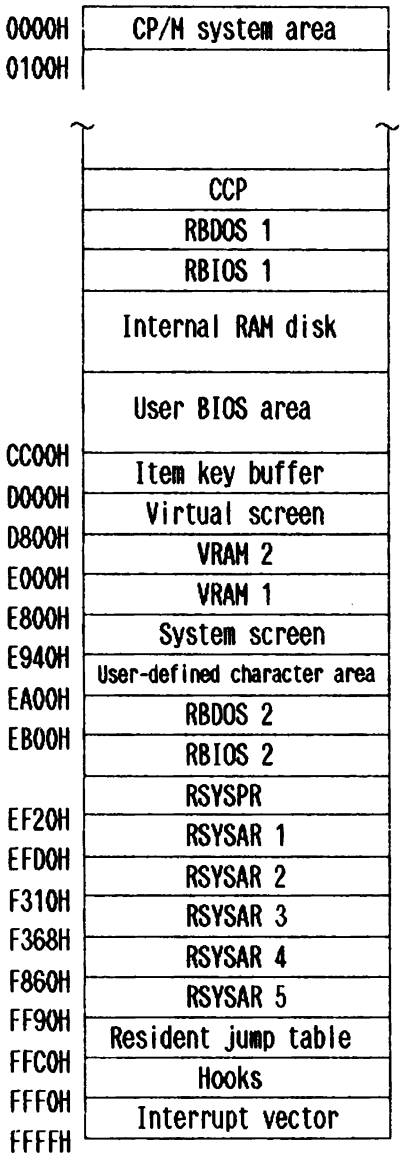
The area located at ØE000H to ØFFFFH (8K bytes) is not affected by bank switching. This area plays an important role in system operations.

See Section 4.4, "Bank Switching" for detailed information concerning bank switching and memory configuration.

6.1.2 RAM Memory Map

Figure 6.1.1 shows the PINE RAM (bank Ø) memory map.

System areas RSYSAR1 to RSYSAR5 are detailed in Section 6.2.



The size of these areas is variable (maximum combined value = 35.5KB).

Figure 6.1.1 RAM Memory Map

CP/M System Area (at 0000H) 100H bytes
- Loaded with jump addresses to BDOS and BIOS functions, or the I/O byte.

CCP (variable address) 800H bytes
- Loaded with a CCP program for processing by CCP.
- Supports built-in commands such as ERA, DIR and REN.
- The starting address of this area is affected by the RAM disk and user BIOS area size.

RBDOS1 (variable address) 100H bytes
- Loaded with the BDOS entry addresses for load-and-go programs.
- When calling any BDOS function in load-and-go programs, call address 0005H just as in standard CP/M application programs.
- See Section 3.2, "BDOS Operations."

RBIOS1 (variable address) 100H bytes
- Loaded with the BIOS entry addresses for load-and-go programs.
- When calling any BIOS function in load-and-go programs, call the BIOS address computed from the WBOOT entry address located at 0001H to 0002H just as in standard CP/M application programs.
- See Section 3.3, "BIOS Operations."

Internal RAM Disk (variable address) Variable length
- This area can be used as if it were on disk.
- The internal RAM disk size is specified in 1K-byte units. The total combined size of the user BIOS area and this area can be set at up to 35.5 KB.
- When the external RAM disk is connected, the size of the internal RAM disk is automatically set to 0 KB.

User BIOS Area (variable address) Variable length
- Used by the user to store a program or data in RAM.
- The user BIOS area is specified in 256 byte units. The total combined size of the internal RAM disk and this area can be set at up to 35.5 KB.
- See 4.1, "User BIOS."

Item Key Buffer (at 0CC00H) 400H bytes
- Stores the table for item function keys.
- This table is used to redefine item function keys when the item keyboard is installed. Key strings can be specified as up to 15 characters in length.
- See 3.5.4, "Item Keyboard."

Virtual Screen (at 0D000H) 800H bytes
- Stores character data for the virtual screen (user screen).
- The virtual screen size can be specified as up to 25 lines x 80 characters.
- See 3.6.2, "Screen Modes."

VRAM2 (at 0D800H) 800H bytes
- VRAM area used for the system screen.
- See 3.6.2, "Screen Modes."

VRAM1 (at 0E000H) 800H bytes
- VRAM area used for the virtual screen (user screen).
- See 3.6.2, "Screen Modes."

System Screen (at ØE8ØØH) 14ØH bytes
- Stores character data for the system screen.
- The system screen size is fixed at 8 lines x 4Ø characters.
- See 3.6.2, "Screen Modes."

User-defined Character Area (at ØE94ØH) CØH bytes
- Stores user-defined characters.
- One character (6 x 8-dot font) is made up of 6 bytes. The user can define up to 32 character fonts in EØH through FFH.
- See 3.6.6, "Character Generator."

RBDOS2 (at ØEAØØH) 1ØØH bytes
- Loaded with the BDOS entry addresses for ROM-based programs.
- When calling any BDOS function in ROM-based programs, call 'JP RBDOS' (ØFF9ØH).
- See Section 3.2, "BDOS Operations."

RBIOS2 (at ØEBØØH), RSYSPR (at ØEBØØH) 52ØH bytes

RBIOS2
- Loaded with the BIOS entry addresses for ROM-based programs.
- When calling BIOS functions in ROM-based programs, use these fixed BIOS entry addresses.
- See Section 3.3, "BIOS Operations."

RSYSPR
- Stores system resident programs such as bank switching and interrupt processing.

RSYSAR1 (at ØEF2ØH) BØH bytes
- A work area used by the system.
- This area is initialized at system initialize time.
- See 6.2.2, "System Area I (RSYSAR1)."

RSYSAR2 (at ØEFDØH) 34ØH bytes
- A work area used by the system.
- This area is initialized at system initialize or reset time.
- See 6.2.3, "System Area II (RSYSAR2)."

RSYSAR3 (at ØF31ØH) 58H bytes
- A work area used by the system.
- This area is initialized at system initialize, reset, or WBOOT time.
- See 6.2.4, "System Area III (RSYSAR3)."

RSYSAR4 (at ØF368H) 4F8H bytes
- A work area used by the system.
- Loaded with parameters that need not be initialized.
- See 6.2.5, "System Area IV (RSYSAR4)."

RSYSAR5 (at ØF86ØH) 738H bytes
- A work area used by the system.
- This area is used as a buffer or stack area.
- See 6.2.6, "System Area V (RSYSAR5)."

Resident Jump Table (at ØFF9ØH) 3ØH bytes
- Loaded with the jump table to OS ROM programs to make system utilities accessible to the user.
- See 4.2, "Jump Tables."

Hook (at ØFFCØH) 3ØH bytes
- Loaded with the hook jump table for extended OS interrupt and BIOS processing.
- See 4.3, "Hooks."

Interrupt Vector (at ØFFFØH) 1ØH bytes
- Loaded with an interrupt vector for five levels of interrupts.
- See 4.7, "Interrupts."

6.2 System Area Details

6.2.1 General

This subsection describes individual system areas used by the PINE OS.
The system areas are described in terms of the following items:

Address:
Indicates the address of each parameter. If a parameter is spread over two or more bytes, its starting address is indicated. For a stack area, its ending address is indicated according to the characteristics of the area.

Variable name:
Indicates each parameter name used by the system. Specifying two or more names for this item indicates that the same area is concurrently used by two or more programs which have different names.

Number of bytes:
Indicates the size of the specified area in bytes. The size is represented in hexadecimal units.

Type:
Indicates the characteristics of the specified area.
- o ------- The user can modify system operation by rewriting the area.
- # ------- Variable area. The value of this area is changed when the status is changed. The user can use this area to check the system status.
- x ------- Work area. This area is used as a work area when the system is in operation.

Initial Value:
Indicates the initial default value of the specified area.

Description:
An explanation of the specified area and the purpose for which it is used.

Reference:
Indicates the section in which the specified area is described in detail. A section number enclosed in parentheses indicates that the section has no description of the area, but is related to the area.

6.2.2 System Area I (RSYSAR1)

This area is initialized only at system initialize time.

RETADD (at ØEF1FH) Ø1H byte Type: x
- The initial value is ØC9H.
- Indicates the address of the called hook. The address contains the RET instruction.
- The hook call address points to RETADD after the hook routine is completed.
- See 4.3, "Hooks."

RBITOP (at ØEF2ØH) Ø2H bytes

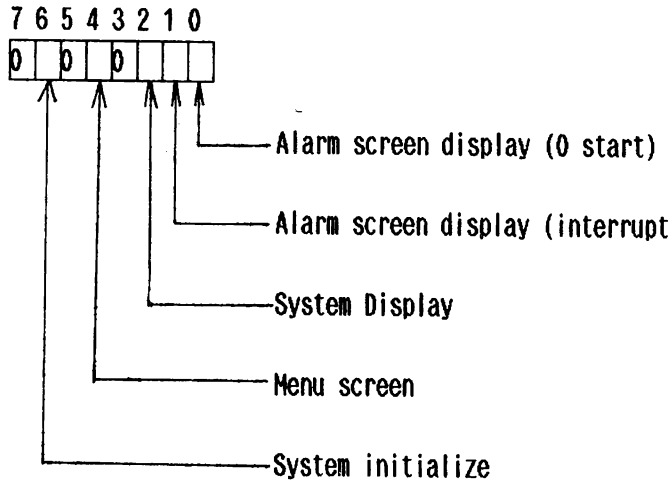
CCPLAD (at ØEF22H) Ø2H bytes Type: #
- The initial value is 5AØØH.
- Loaded with the CCP load starting address used when relocating the CCP program into RAM.
- This value is affected by the size of the internal RAM disk and user BIOS area.
- The initial value is specified in relation to an internal RAM disk size of 26 KB and a user BIOS area size of Ø KB.

BDSLAD (at ØEF24H) Ø2H bytes Type: #
- The initial value is 62ØØH.
- Loaded with the load starting address used when loading the BDOS program using RBDOS1 into RAM.
- This value is affected by the internal RAM disk and user BIOS area sizes. The value must satisfy the following formula:
  (BDSLAD) .= (CCPLAD) + Ø8ØØH
- The initial value is specified in relation to an internal RAM disk size of 26 KB and a user BIOS area size of Ø KB.
- See 3.2, "BDOS Operations."

BI1LAD (at ØEF26H) Ø2H bytes Type: #
- The initial value is 63ØØH.
- Indicates the starting address in RAM of the BIOS entry points to RBIOS1.
- This value is affected by the size of the internal RAM disk and user BIOS area. This value must satisfy the following formula:
  (BI1LAD) = (BDSLAD) + 1ØØH
- The initial value is specified in relation to an internal RAM disk size of 26 KB and a user BIOS area size of Ø KB.
- See 3.3, "BIOS Operations."

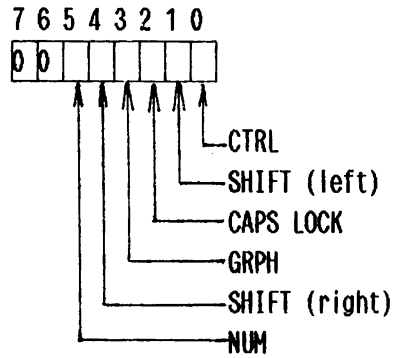
RESEXQ (at ØEF28H) Ø1H byte Type: o
- The initial value is ØØH.
- This is the flag used to specify resident processing.
  RESEXQ = ØØH : Non-Resident
  = other than ØØH: Resident
- This flag is set and reset using BIOS RESIDENT.
- See 4.5, "Resident Processing."

MODEFG (at ØEF29H) Ø1H byte Type: x
- The initial value is ØØH.
- This flag indicates the status of the currently executing processing.



- See 8.1, "State Transition."

CNTNKEY (at ØEF2AH) ØlH byte Type: o
- The initial value is ØlH.
- Indicates the shift key state when turning power off in the continue mode.



(Each bit is set to 1 if the corresponding key is pressed down at power off time.)

- When the item keyboard is installed, only Bit Ø (CTRL) can be specified.
- When CNTNKEY is set to ØØH, regardless of the shift key state, the power is always turned off in the continue mode.
- When the initial value is still set, the power can be turned off in the continue mode by setting the power switch to OFF while holding down the CTRL key.

EXRAMACFLG (at ØEF2BH) ØlH byte Type: x
- The initial value is ØØH.
- Indicates whether or not the external RAM disk is accessed.

EXRAMACFLG = ØØH : Not accessed.
= Other than ØØH: Accessed.

- This area is used by the system to determine whether or not the 'RAM DISK FORMAT (Y/N)' prompt should be displayed when the RESET key is pressed during an access to the external RAM disk.

SIZRAM (at ØEF2CH) Ø1H byte Type: #
- The initial value is 1BH.
- Indicates the internal RAM disk capacity in 1K byte units.
- Application programs can modify the internal RAM disk size by changing the value of SIZRAM and then calling any system utility program according to the specified procedure.
- The procedure is the same as for USERBIOS.

USERBIOS (at ØEF2DH) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates the user BIOS area size in 256K byte units.
- Application programs can modify the user BIOS area size by changing the value of USERBIOS and then calling any system utility program according to the specified procedure.
- The procedure is described in Section 4.1, "User BIOS."
- See 4.1, "User BIOS."

RSXON (at ØEF2EH) Ø1H byte Type: o
- The initial value is 11H.
- Loaded with an XON code when XON/XOFF is specified by BIOS RSIOX.
- See (23) RSIOX in 3.4.2.

RSXOFF (at ØEF2FH) Ø1H byte Type: o
- The initial value is 13H.
- Loaded with the XOFF code when XON/XOFF is specified by BIOS RSIOX.
- See (23) RSIOX in 3.4.2.

DUPLEX (at ØEF3ØH) ØØH byte

SRSADR (at ØEF31H)
- Used by the system to specify the RSIOX open parameters for data input/output through the RS-232C interface with BIOS PUNCH, READER, or CONIN.
- See (7) PUNCH in 3.4.2.

    Ø2H bytes Type : o
    - The initial value is ØFC9BH.
    - Indicates the starting address of the receive buffer.
    - The initial value points to the beginning of COMBUF.

    Ø2H bytes Type : o
    - The initial value is ØØFØH.
    - Indicates the starting address of the receive buffer.

    Ø1H byte Type : o
    - The initial value is ØDH.
    - Specifies the transmission speed when sending or receiving data.

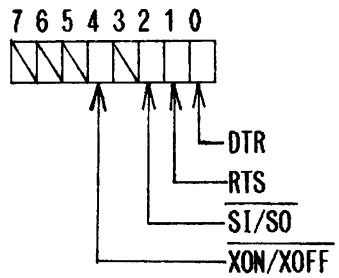
Value	Speed (BPS)	Value	Speed (BPS)
02H	110	0DH	4800
04H	150	0EH	9600
05H	200	0FH	19200
06H	300	10H	38400
08H	600	80H	75/1200
0AH	1200	81H	1200/75
0CH	2400		

Ø1H byte Type : o
- The initial value is Ø3H.
- Specifies the bit length.
  = Ø2H: 7 bits/character
  = Ø3H: 8 bits/character

Ø1H byte Type : o
- The initial value is ØØH.
- Specifies the type of parity checking.
  = ØØH: No parity
  = Ø1H: Odd parity
  = Ø3H: Even parity

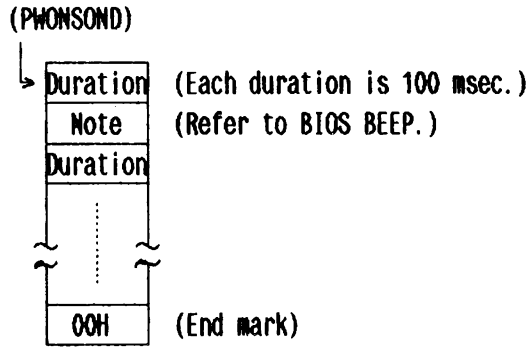
Ø1H byte Type : o
- The initial value is Ø3H.
- Specifies the number of stop bits.
  = Ø1H: 1 bit
  = Ø3H: 2 bits

Ø1H byte Type : o
- The initial value is ØFFH.
- Specifies how to control sending and receiving.



(DTR and RTS are active when the corresponding bits are set to 1, and SI/SO and XON/XOFF controls are specified when set to Ø.)

PWONSOND (at ØEF3AH) Ø2H bytes Type: o
- Indicates the starting address of the musical note table when the power is turned on in the restart mode.



- The initial value of PWONSOND points to the note table address in OS ROM, in which sound data 01H, 37H and 00H are stored.
- When rewriting the contents of the note table in application programs, set PWONSOND to point to address 8000H or higher.
- See 2.4, "Power-on."

CNTNSOND (at 0EF3CH) 02H bytes Type: o
- Indicates the starting address of the note table when power is turned on in the continue mode.
- The format of the table is described in PWONSOND (0EF3AH).
- The initial value of CNTNSOND indicates the note table address in OS ROM, in which sound data 01H, 37H and 00H are stored.
- When rewriting the contents of the note table in application programs, set CNTNSOND to point to address 8000H or higher.
- See 2.4, "Power-on."

ALRMSOND (at 0EF3EH) 02H bytes Type: o
- Indicates the starting address of the note table when the alarm/wake screen is displayed.
- The format of the table is described in PWONSOND (0EF3AH).
- The initial value of ALRMSOND indicates the note table address in OS ROM, in which sound data 01H, 25H, 01H, 29H and 00H are stored three times in succession.
- When rewriting the contents of the note table in application programs, set ALRMSOND to point to address 8000H or higher.
- See 2.9, "Alarm/Wake Function."

ATSHUTOFF (at 0EF40H) 01H byte Type: o
- The initial value is 05H.
- Specifies, in minutes, the time until power is automatically turned off when the system is waiting for entry from the keyboard (BIOS CONIN).
- ATSHUTOFF = 00H indicates that the auto power off function is disabled.
- This area is used by the system only to identify whether the auto power off function is disabled or enabled. The actual time until the auto power off is specified by ATSOTIME (0EF41H).
- See 2.8, "Auto Power-off."

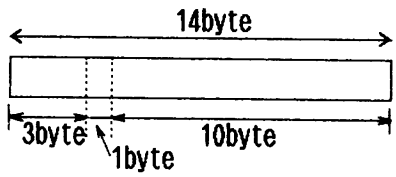
ATSOTIME (at ØEF41H) Ø2H bytes Type: o
- The initial value is Ø12CH.
- Specifies, in seconds, the time until power is automatically turned off when the system has been waiting for entry from the keyboard (BIOS CONIN).
- The value of ATSOTIME must satisfy the following formula:
  ATSOTIME = ATSHUTOFF x 6Ø
- The system measures the time specified in ATSOTIME using the 1-second interrupt.
- See 2.8, "Auto Power-off."

ALRMSTDS (at ØEF43H) Ø1H byte Type: o
- The initial value is ØØH.
- ALRMSTDS is the flag used to disable the setting of alarm/wake times during system display processing.
  = ØØH : Setting of alarm/wake times is enabled.
  = Other than ØØH: Setting of alarm/wake times is disabled.
- See 2.11, "System Display."

MENUFG (at ØEF44H) Ø1H byte Type: o
- The initial value is ØØH.
- This is the flag used to specify whether or not the menu screen is to be displayed.
  = ØØH : Menu screen is to be displayed.
  = Other than ØØH: No menu screen is to be displayed.
- See 2.1Ø, "Menu Function."

MENUDRV (at ØEF45H) ØBH bytes Type: o
- Specifies the drive for which the menu screen is displayed.
- MENUDRV is specified within the range of 'A' to 'K' in ASCII notation. All unspecified byte fields must be filled with space codes (2ØH).
- The initial value is specified as 'CBA ' when the standard keyboard is installed and 'ABCDHIJK ' when the item keyboard is installed.
- See 2.1Ø, "Menu Function."

FTYPETBL (at ØEF5ØH) 38H bytes Type: o
- This table is used to specify the files which are to be displayed on the menu screen.
- Up to 4 types of files can be specified using the file extension. Each FTYPETBL entry consists of 14 bytes.



File extension | Space (20H) | COM file name corresponding to the specified file extension, and name of the drive in which the file resides

1 Space (20H)
2 COM file name corresponding to the specified file extension, and name of the drive in which the file resides

- The initial contents of FTYPETBL are shown below.

```
'COM␣␣␣␣␣␣␣␣␣␣␣␣'
'BAS B:BASIC␣␣␣'
'␣␣␣␣␣␣␣␣␣␣␣␣␣␣'
'␣␣␣␣␣␣␣␣␣␣␣␣␣␣'
```

- See 2.10, "Menu Function."

FTYPEND (at 0EF88H) 01H byte

ALRMDS (at 0EF89H) 01H byte

ALRMTP (at 0EF8AH) 01H byte Type: #
- The initial value is 00H.
- This flag indicates whether or not an alarm/wake time is set.
  = 00H: No alarm/wake time is set.
  = 01H: Alarm time is set.
  = 02H: Wake time is set.
- See 2.9, "Alarm/Wake Function."

ALRMAD (at 0EF8BH) 02H bytes Type: #
- The initial value is 0F3DFH.
- Indicates the starting address of the alarm message or wake string.
- The initial value points to ALRMMSG.
- When rewriting the contents of ALRMAD in application programs, set ALRMAD to point to address 8000H or higher in RAM.
- See 2.9, "Alarm/Wake Function."

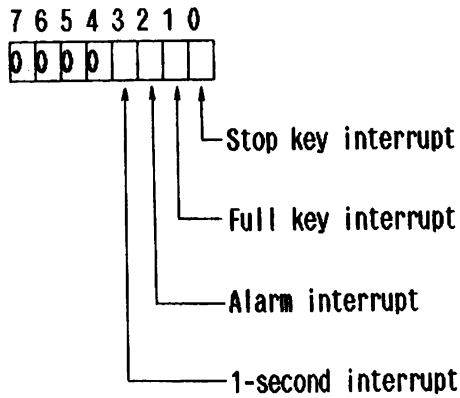
ALRMST (at 0EF8DH) 01H byte Type: #
- The initial value is 00H.
- This flag indicates whether or not an alarm/wake interrupt has occurred.
  = 00H: No alarm/wake has occurred.
  = 01H: Alarm/wake has occurred.
- ALRMST is set to 00H when an alarm/wake time is set or read by BIOS TIMDAT and 01H when an alarm/wake interrupt occurs.
- See 2.9, "Alarm/Wake Function."

ALRMFG (at 0EF8EH) 01H byte Type: x
- The initial value is 00H.
- This flag is used by the system to identify whether or not the next alarm interrupt is the first one generated for the currently-set alarm time.
  = 00H: Alarm interrupts are to be accepted.
  = 01H: Alarm interrupts are to be ignored.
- ALRMFG is set to 00H when ALRMCT (0F4D9H) is decremented to 00H by 1-second interrupts.
- See 4.3.3, "Alarm Hooks."

TIMER0 (at 0EF8FH) 02H bytes Type: #
- The initial value is 0000H.
- TIMER0 is the 16-bit, 1-second up counter.
- This counter is incremented by each time 1-second interrupt occurs.
- The value of this counter must not be modified because it is commonly used by the system.
- See 4.7.4, "7508 Interrupts."

TIMER1 (at ØEF91H) Ø2H bytes Type: #
- The initial value is ØØØØH.
- TIMER1 is a 16-bit, 1-second down counter.
- This counter is decremented each time a 1-second interrupt occurs.
- See 4.7.4, "75Ø8 Interrupts."

ISTS75Ø8 (at ØEF93H) Ø1H byte Type: #
- The initial value is ØAH.
- This flag indicates whether 7508 interrupts are disabled or enabled.



(Interrupts are enabled when the corresponding bits are set to 1, and disabled when reset to Ø.)

TOPRAM (at ØEF94H) Ø2H bytes Type: #
- The initial value is ØCCØØH.
- Indicates the starting address of the user BIOS area.
- ØCCØØH is specified when the user BIOS area size is set to Ø bytes.
- See 4.1, "User BIOS."

FORMATRAM (at ØEF96H) Ø1H byte Type: x
- The initial value is ØØH.
- This flag indicates whether or not the 'RAM DISK FORMAT (Y/N)' prompt is to be displayed.
  = ØØH: No prompt is to be displayed.
  = Nonzero: A prompt is to be displayed.
- See 2.4, "Power-on."

QT_ROM_CP1 (at ØEF97H) Ø1H byte Type: #
- Indicates the B-drive ROM capsule size in K byte units.
- The ROM size can be specified as 8K, 16K or 32K bytes.
QT_ROM_CP1 = ØØH indicates that no ROM capsule is installed.
- See 3.8, "Disk Storage."

QT_ROM_CP2 (at ØEF98H) Ø1H byte Type: #
- Indicates the C-drive ROM capsule size in K byte units.
- The ROM size can be specified as 8K, 16K or 32K bytes.
QT_ROM_CP2 = ØØH indicates that no ROM capsule is installed.
- See 3.8, "Disk Storage."

QT_ROM_CR1 (at ØEF99H) Ø1H byte Type: #
- Indicates the J-drive ROM cartridge size in K byte units.
- The ROM size can be specified as 8K, 16K or 32K bytes.
QT_ROM_CR1 = ØØH indicates that no ROM cartridge is installed.
- See 3.8, "Disk Storage."

QT_ROM_CR2 (at ØEF9AH) Ø1H byte Type: #
- Indicates the K-drive ROM cartridge size in K byte units.
- The ROM cartridge size can be specified as 8K, 16K or 32K bytes.
QT_ROM_CR2 = ØØH indicates that no ROM cartridge installed.
- See 3.8, "Disk Storage."

QT_RAM_CR (at ØEF9BH) Ø1H byte Type: #
- Indicates the I-drive RAM cartridge size in K byte units.
- The RAM size can be specified as 16K, 32K or 64K bytes.
QT_RAM_CR = ØØH indicates that no RAM cartridge is installed.
- See 3.8, "Disk Storage."

QT_RAM_IN (at ØEF9CH) Ø1H byte Type: #
- Indicates the A-drive internal RAM disk size in K byte units.
- QT_RAM_IN = ØØH indicates that no internal RAM disk is installed.
- See 3.8, "Disk Storage."

QT_RAM_EX (at ØEF9DH) Ø1H byte Type: #
- Indicates the A-drive external RAM disk size in K byte units.
- The RAM size can be specified as 64K or 128K bytes.
QT_RAM_EX = ØØH indicates that no external RAM disk is installed.
- See 3.8, "Disk Storage."

DR_ROM_CP1 (at ØEF9EH) Ø1H byte Type: #
- Indicates the number of directory entries in the B-drive ROM capsule.
- DR_ROM_CP1 is specified according to the number of directory entries in the ROM header.
- See 3.8, "Disk Storage."

DR_ROM_CP2 (at ØEF9FH) Ø1H byte Type: #
- Indicates the number of directory entries in the C-drive ROM capsule.
- DR_ROM_CP2 is specified according to the number of directory entries in the ROM header.
- See 3.8, "Disk Storage."

DR_ROM_CR1 (at ØEFAØH) Ø1H byte Type: #
- Indicates the number of directories in the J-drive ROM cartridge.
- DR_ROM_CR1 is specified according to the number of directory entries in the ROM header.
- See 3.8, "Disk Storage."

AD_ROM_CP1 (at ØEFA2H) Ø2H bytes Type: #
- Indicates the B-drive ROM capsule header starting address.
- This address is affected by the ROM size and format. ØØØØH is specified when no ROM is installed.
- See 3.8, "Disk Storage."

AD_ROM_CP2 (at ØEFA4H) Ø2H bytes Type: #
- Indicates the C-drive ROM capsule header starting address.
- This address is affected by the ROM size and format. ØØØØH is specified when no ROM is installed.

AD_ROM_CR1 (at ØEFA6H) Ø2H bytes Type: #
- The initial value is ØØØØH.
- Indicates the J-drive ROM cartridge header starting address (fixed).
- See 3.8, "Disk Storage."

AD_ROM_CR2 (at ØEFA8H) Ø2H bytes Type: #
- The initial value is 8ØØØH.
- Indicates the K-drive ROM cartridge header starting address (fixed).
- See 3.8, "Disk Storage."

AD_RAM_IN (at ØEFAAH) Ø2H bytes Type: #
- Indicates the A-drive internal RAM disk starting address.
- This address is affected by the internal RAM disk size. AD_RAM_IN = ØØØØH indicates that no internal RAM disk is installed.
- See 3.8, "Disk Storage."

LOG_ADRS (at ØEFACH) Ø3H bytes Type: x
- This work area is used by the system to compute the logical disk address from the specified track and sector numbers when a disk access is made.

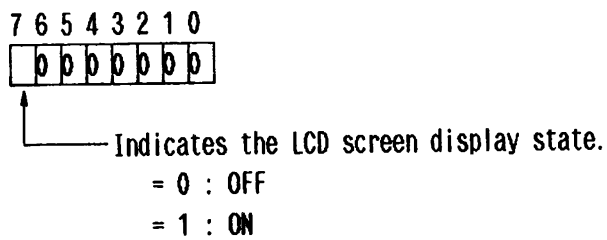
DSK_R_W (at ØEFAFH) Ø1H byte Type: x
- This work area is used by the system to determine whether to read or write at disk access time.

S_CHK_SUM (at ØEFBØH) Ø2H bytes Type: x
- This work area is used by the system to compute the checksum when a disk access is made.

QT_RAM_OLD (at ØEFB3H) Ø1H byte Type: x
- This work area is used to store the old internal RAM disk size when modifying the internal RAM disk size.

AD_RAM_OLD (at ØEFB4H) Ø2H bytes Type: x
- This work area is used to store the old internal RAM disk starting address when modifying the internal RAM disk size.

DSPFLAG (at ØEFB6H) Ø1H byte Type: #
- The initial value is ØØH.
- This flag indicates the ON/OFF state of the LCD screen.



- The LCD screen state is set to ON (DSPFLAG = 80H) when the system is activated (power is turned on).
- DSPFLAG is used when the OR operation is executed between the contents of LVRAMYOF (0F2A0H) and DSPFLAG, and the result is loaded into YOFF (P09H).
- See 3.6, "LCD Display (CONOUT Details)."

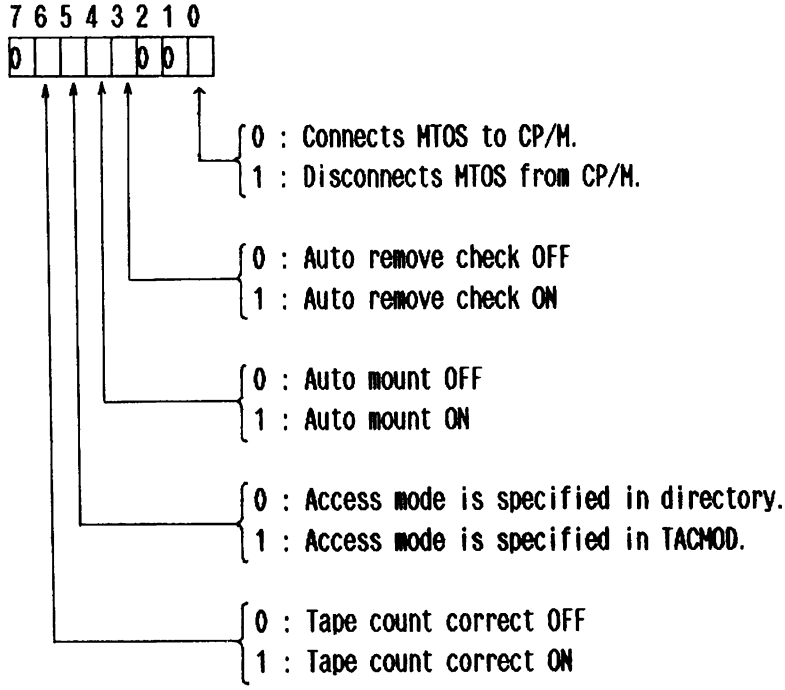
BLNKSTAT (at 0EFB7H) 01H byte Type: #
- The initial value is 00H.
- This flag indicates whether or not cursor blink processing is to be performed using the OVF interrupt.
    = 00H: No blink processing is to be performed.
    = 01H: Blink processing is to be performed.
- This flag is set to 01H when the cursor state is set to ON and blink is specified by BIOS CONOUT.
- See 3.6, "LCD Display (CONOUT Details)."

BLNKCNT (at 0EFB8H) 01H byte Type: x
- This work area is used in OVF interrupt processing to measure the cursor blink interval.

BLNKRVS (at 0EFB9H) 01H byte Type: #
- The initial value is 00H.
- This flag indicates whether or not the cursor is currently displayed on the LCD screen.
    = 00H: No cursor is currently displayed.
    = 01H: Cursor is currently displayed.
- See 3.6, "LCD Display (CONOUT Details)."

BLNKTIME (at 0EFBAH) 01H byte Type: o
- The initial value is 04H.
- Indicates the cursor blink interval in 100 ms units.
- The initial value corresponds to approximately 500 ms.
- See 4.7.6, "OVF Interrupts."

TOSCTL (at 0EFBBH) 01H byte Type: o
- The initial value is 50H.
- This flag is used to specify MTOS operations.



- Do not rewrite the value of TOSCTL directly. Instead, use IDFLG (ØFA13H) and TCNTTP (ØFA14H) to modify the value.
- See 3.7.4, "Using MTOS."

STOSCTL (at ØEFBCH) Ø1H byte

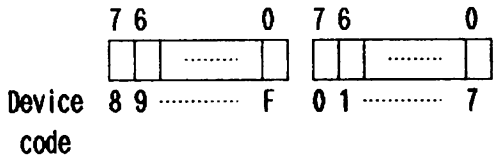
TOSSTS (at ØEFBDH) Ø1H byte Type: x
- This work area is used to indicate the MTOS state or operation during MTOS processing.
- See 3.7, "MTOS/MIOS Operations."

VERFDFLT (at ØEFBEH) Ø1H byte Type: o
- The initial value is ØØH.
- VERFDFLT is the default flag used to specify the verify mode when writing to tape.
  = ØØH : Non-verify mode
  = Other than ØØH: Verify mode
- This flag is copied into VERFFG (ØEFBFH) at warm boot time.
- See 3.7.4, "Using MTOS."

VERFFG (at ØEFBFH) Ø1H byte Type: o
- The initial value is ØØH.
- This flag is used to specify the verify mode when writing to the tape.
  = ØØH : Non-verify mode
  = Other than ØØH: Verify mode
- The initial value is loaded with the value of VERFDFLT (ØEFBEH) at warm boot time.
- See 3.7.4, "Using MTOS."

MCTDETCH (at ØEFCØH) Ø1H byte Type: o
- The initial value is ØFFH.
- If no MCT is installed at power-on time, the remove state is automatically specified.
    = ØØH: The mount/remove state is to remain unchanged.
    = FFH: The remove state is to be automatically specified.
- This flag is used to prevent the current mount state from being changed when a cartridge other than an MCT cartridge is temporarily installed.
- See 3.7, "MTOS/MIOS Operations."

CRSTPTN (at ØEFC1H) Ø2H bytes Type: o
- The initial value is 41H, 4ØH.
- Specifies the *CRS signal state corresponding to each cartridge device code after power is turned on.



Bit = Ø: Set the $\overline{CRS}$ low.
    = 1: Set the $\overline{CRS}$ high.

- The cartridge device codes are specified in bits 3-Ø of CRGDEV (ØF53FH).
- See 5.1, "Cartridges."

(Reserved) (ØEFC3H) ØD bytes

6.2.3 System Area II (RSYSAR2)

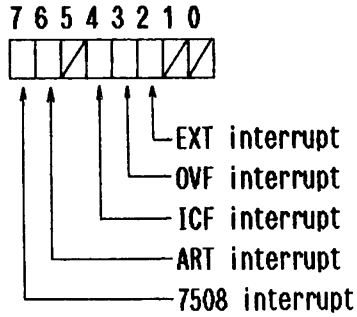
This area is initialized at system initialize or reset time.

CPMSIZ (at ØEFDØH) Ø1H byte

INTBIOS (at ØEFD1H) Ø1H byte Type: #
- The initial value is ØØH.
- This flag indicates whether or not BIOS processing is currently being executed.
    = ØØH: No BIOS processing is currently being executed.
    = ØFFH: BIOS processing is currently being executed.
- This flag is set to ØFFH by PREBIOS and reset to ØØH by PSTBIOS.
- See 3.3, "BIOS Operations."

INTLEVEL (at ØEFD2H) Ø1H byte Type: x
- The initial value is ØFFH.
- This counter indicates the 75Ø8 interrupt level.
    = ØFFH: No 75Ø8 interrupt processing performed
    = Other than ØFFH: 75Ø8 interrupt processing performed
- See 4.7.4, "75Ø8 Interrupts."

INTTYPE (at ØEFD3H) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates what type of interrupt is generated.



- Each bit is set to 1 when the corresponding interrupt is generated.
- See 4.7, "Interrupts."

INTS75Ø8 (at ØEFD4H) Ø2H bytes Type: x
- The initial value is SP75Ø8.
- Loaded with the stack pointer initial value for 75Ø8 interrupt processing.
- See 4.7, "Interrupts."

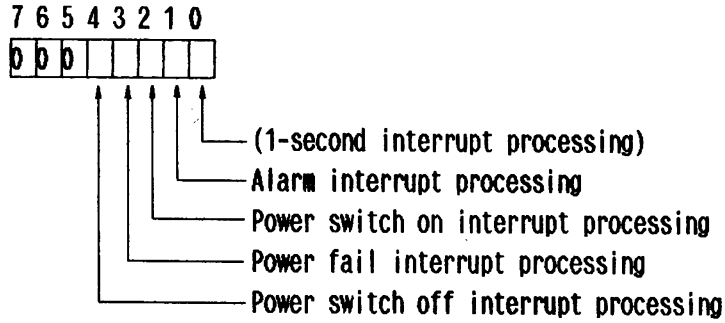
INTS8251 (at ØEFD6H) Ø2H bytes Type: x
- The initial value is SP8251.
- Loaded with the stack pointer initial value for ART interrupt processing.
- See 4.7, "Interrupts."

INTSICF (at ØEFD8H) Ø2H bytes Type: x
- The initial value is SPICF.
- Loaded with the stack pointer initial value for ICF interrupt processing.
- See 4.7, "Interrupts."

INTSOVF (at ØEFDAH) Ø2H bytes Type: x
- The initial value is SPOVF.
- Loaded with the stack pointer initial value for OVF interrupt processing.
- See 4.7, "Interrupts."

INTSEXT (at ØEFDCH) Ø2H bytes Type: x
- The initial value is SPEXT.
- Loaded with the stack pointer initial value for external interrupt processing.
- See 4.7, "Interrupts."

TBL75Ø8 (at ØEFDEH) 1ØH bytes Type: o
- This table is used to pass control to the pertinent interrupt servicing routine according to the type of the 75Ø8 interrupt.
- The bit assignments of each byte are shown below.



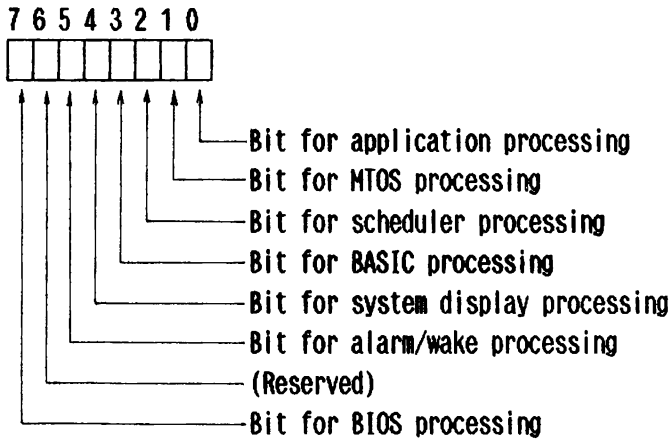
(When a bit is set to 1, the corresponding type of processing is executed.)

- The table format is shown below. When 7508 interrupt processing is performed, the appropriate type of processing routine is selected from this table by referring to the 7508 status that is read with a Status Read command and the current power switch state. Also, 1-second interrupt processing is activated by software according to the 7508 status after the correct interrupt processing is selected from this table.

Address	7508 status	Power switch state	Initial value
EFDEH	C0H or E0H	OFF	00H
DFH	〃	ON	10H
E0H	C1H or E1H	OFF	04H
E1H	〃	ON	00H
E2H	C2H or E2H	OFF	02H
E3H	〃	ON	12H
E4H	C3H or E3H	OFF	04H
E5H	〃	ON	02H
E6H	C4H or E4H	OFF	08H
E7H	〃	ON	10H
E8H	C5H or E5H	OFF	08H
E9H	〃	ON	08H
EAH	C6H or E6H	OFF	08H
EBH	〃	ON	10H
ECH	C7H or E7H	OFF	08H
EDH	〃	ON	08H

BTRYFG (at ØEFEEH) Ø1H byte Type: #
- The initial value is ØØH.
- This flag indicates whether or not a power failure has occurred.
    = ØØH: No power failure has occurred.
    = ØFFH: Power failure has occurred.
- See 4.7.4, "75Ø8 Interrupts."

YPOFDS (ØEFEFH) 1 byte Type: o
- The initial value is ØØH.
- This flag is used to disable power off processing.

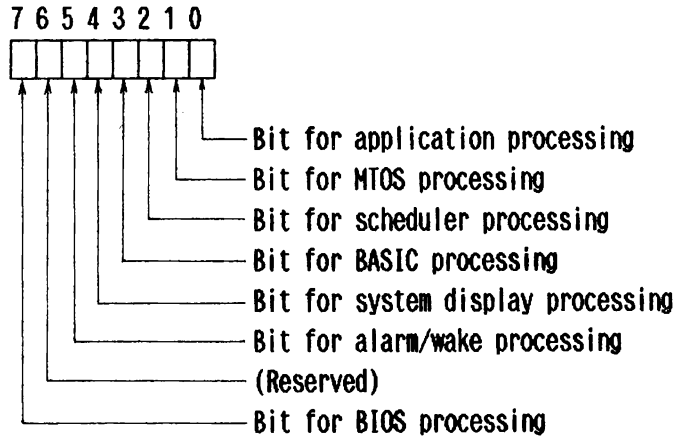


(When each bit is set to 1, the corresponding processing is disabled.)

- See 4.7.3 for the actual procedure for inhibiting each type of processing.
- See 4.7.3, "Interrupt Control."

YPOFST (ØEFFØH) Ø1H byte Type: #
- The initial value is ØØH.
- This flag indicates whether or not a power-off interrupt has occurred while power-off processing is disabled by YPOFDS (ØEFEFH).
- When a power-off interrupt occurs, the value of YPOFDS is copied into this flag.
- See 4.7.3, "Interrupt Control."

YALMDS (at ØEFF1H) Ø1H byte Type: o
- The initial value is ØØH.
- This flag is used to disable alarm screen display processing.



(When each bit is set to 1, the corresponding type of processing is disabled.)

- See 4.7.3 for the actual procedure for inhibiting each type of processing.
- See 4.7.3, "Interrupt Control."

YALMST (at ØEFF2H) Ø1H byte Type: #
- The initial value is ØØH.
- This flag indicates whether or not an alarm interrupt has occurred while alarm screen display processing is disabled by YALMDS (ØEFF1H).
- When an alarm interrupt occurs, the value of YALMDS is copied into this flag.

CONSCRN1 (at ØEFF3H) Ø8H bytes Type: o
- The initial values are 1BH, ØDØH, 19H, 5ØH, 1BH, 95H, Ø2H and ØØH.
- CONSCRN1 is the screen parameter data which is initialized at boot time.

1st byte: ESC (1BH)
2nd byte: ØDØH
3rd byte: Number of screen lines
4th byte: Number of screen columns
5th byte: ESC (1BH)
6th byte: 95H
7th byte: Scrolling mode
8th byte: ØØH

The first four bytes specify the screen size. The initial screen size is 8Ø columns by 25 lines.

The fifth to seventh bytes specify the window scrolling mode. The initial setting is for horizontal non-tracking mode.

- This area is used by BIOS CONOUT to specify the screen state.
- See 3.6.7, "Miscellaneous Considerations."

CONSCRN2 (at ØEFFBH) Ø6H bytes Type: o
- The initial values are lBH, 33H, lBH, ØD6H, ØØH and ØFFH.
- This screen parameter data is initialized at warm boot time.

1st byte: ESC (lBH)
2nd byte: Cursor display on/off
3rd byte: ESC (lBH)
4th byte: ØD6H
5th byte: Cursor type
6th byte: ØFFH

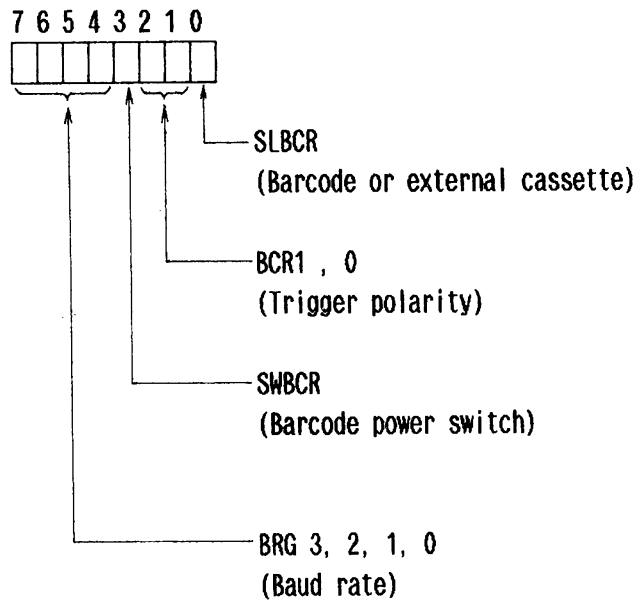
The first two bytes specify whether the cursor is to be turned on or off. The initial display mode is cursor display on.

The third to fifth bytes specify the cursor type. The initial specification is for a block-type, blinking cursor.

ØFFH identifies the end of the parameter block.

- This area is used by BIOS CONOUT to specify the screen state.
- See 3.6.7, "Miscellaneous Considerations."

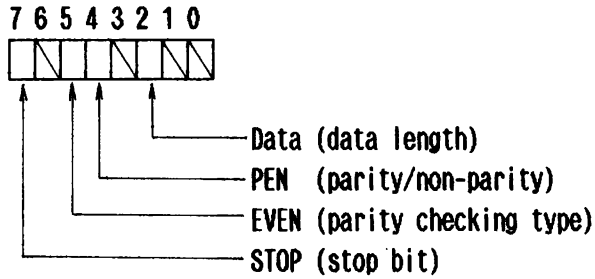
RZCTLRl (at ØF00lH) Ø1H byte Type: o
- Stores the output state of the control register 1 (PØØH).



- The bit assignments for this area are identical to those for CTLRl (PØØH).
- See Chapter 5, "I/O Operations."

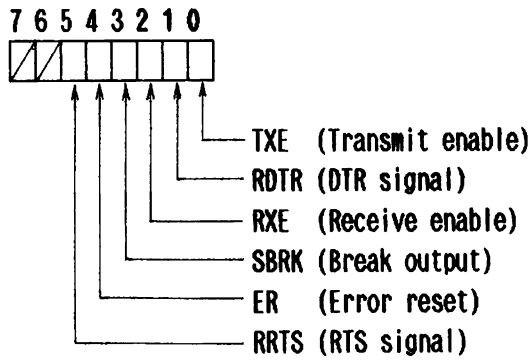
RZCMDR (at ØF002H) 1 byte

RZARTMR (at ØF003H) Ø1H byte Type: o
- Stores the output state of the ART mode register (Pl5H).



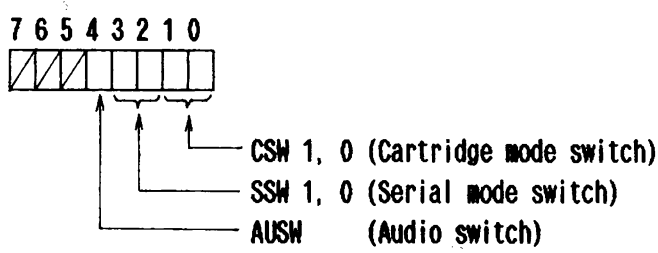
- The bit assignments for this area are identical to those for ARTMR (P15H).
- See Chapter 5, "I/O Operations."

RZARTCR (at ØFØØ4H) Ø1H byte Type: o
- Stores the output state of the ART command register (P1BH).



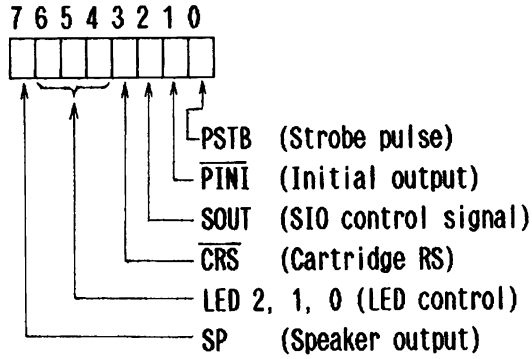
- The bit assignments for this area are identical to those for ARTCR (P16H).
- See Chapter 5, "I/O Operations."

RZSWR (at ØFØØ5H) Ø1H byte Type: o
- Stores the output state of the switch register (P18H).



- The bit assignments for this area are identical to those for SWR (P18H).
- See Chapter 5, "I/O Operations."

RZIOCTLR (at ØF006H) Ø1H byte Type: o
- Stores the output state of the I/O control register (P19H).



- The bit assignments for this area are identical to those for IOCTLR (P19H).
- See Chapter 5, "I/O Operations."

RZDBWDAT (at ØF007H) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates the current ROM cartridge power supply state.
  = ØØH: Power off state
  = Ø1H: Power on state
- See 5.1.3, "ROM Cartridge."

KEYF (at ØF008H) Ø1H byte Type: x
- The initial value is ØØH.
- KEYF is the console buffer (KEYD) status flag.
  = ØØH: No input data present
  = ØFFH: Input data present
- See 3.5, "Keyboard."

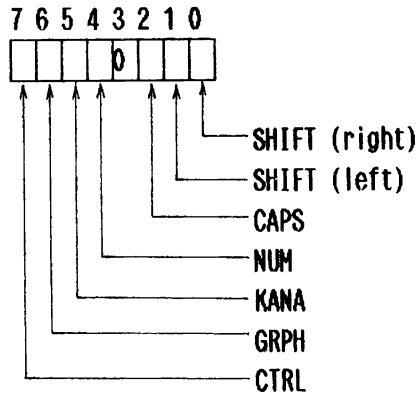
KEYD (at ØF009H) Ø1H byte Type: x
- KEYD is a 1-byte console input buffer.
- When keyed data is entered, it is loaded into this area and KEYF is set by BIOS CONST.
- See 3.5, "Keyboard."

KEYS (at ØF00AH) Ø1H byte Type: x
- The initial value is ØØH.
- Indicates the console buffer (KEYD) input key status.
  = ØØH: Non-check-mode data
  = ØFFH: Check-mode data
- See 3.5, "Keyboard."

YINTPTR (at ØFØØBH) Ø2H bytes Type: #
- The initial value is KBUF.
- YINTPTR is the key buffer (KBUF) input key code 'put' pointer.
- See 3.5, "Keyboard."

YKEYPTR (at ØFØØDH) Ø2H bytes Type: #
- The initial value is KBUF.
- YKEYPTR is the key buffer (KBUF) input key code 'get' pointer.
- See 3.5, "Keyboard."

YSHFDT (at ØFØØFH) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates the shift key state of the standard keyboard.



(Bits 5-2 are set to 1 when the corresponding modes are set to ON; other bits are set to 1 when the corresponding shift keys are pressed.)

- Bit 5 is 'don't care' in ASCII-version OS.
- See 3.5, "Keyboard."

ZSHIFT (at ØFØ1ØH) Ø2H bytes Type: x
- Loaded with the starting address of the selected key code conversion table based on the current shift state.
- See 3.5, "Keyboard."

YPFKSTR (at ØFØ12H) Ø2H bytes Type: o
- The initial value is ØF545H.
- Loaded with the starting address of the PF key table.
- When the initial value is set, YPFKSTR points to WPFKTBL.
- When modifying this address in application programs, set it to point to address 8ØØØH or higher in RAM.
- See 3.5.3, "Standard Keyboard."

YPFKPTR (at ØFØ14H) Ø2H bytes Type: x
- The initial value is ØØØØH.
- YPFKPTR is the get pointer for PF key or item key strings.
- See 3.5, "Keyboard."

YPFKCNT (at ØFØ16H) Ø1H byte Type: x
- The initial value is ØØH.
- This counter indicates the number of the characters left in the buffer containing the PF key or item key string.
- See 3.5, "Keyboard."

YPFCMFLG (at ØF Ø17H) Ø1H byte Type: o
- The initial value is ØØH.
- This flag specifies the PF key check mode.
    = ØØH: Non-check mode
    = ØFFH: Check mode
- When a PF key is pressed after YPFCMFLG is set to ØFFH, the PF key is returned, instead of the character string corresponding to the PF key.
- This flag is valid only when the standard keyboard is installed.
- See (4) CONIN in 3.4.2.

YSFCMFLG (at ØF Ø18H) Ø1H byte Type: o
- The initial value is ØØH.
- This flag specifies the special key (SHIFT/arrow, CTRL/arrow) check mode.
    = ØØH: Non-check mode
    = ØFFH: Check mode
- When a special key is pressed after YSFCMFLG is set to ØFFH, the special key system function is disabled, and instead the key code corresponding to the special key is returned.
- This flag is valid only when the standard keyboard is installed.
- See (4) CONIN in 3.4.2.

BRKFLG (at ØF Ø19H) Ø1H byte Type: #
- The initial value is ØØH.
- This flag indicates whether or not the STOP key has been pressed.
    = ØØH: STOP key not pressed.
    = ØFFH: STOP key pressed.
- This flag is reset when BIOS CONST or CONIN is called.
- See 3.5.3, "Standard Keyboard."

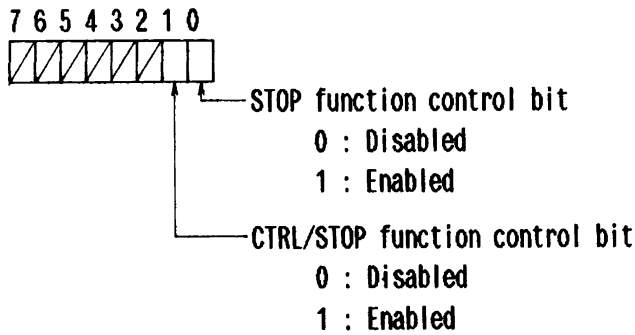
CSTOPFLG (at ØF Ø1AH) Ø1H byte Type: #
- The initial value is ØØH.
- This flag indicates whether or not the CTRL/STOP keys have been pressed.
    = ØØH: CTRL/STOP keys pressed.
    = Ø1H: CTRL/STOP keys not pressed.
- This flag is reset when BIOS CONST or CONIN is called.
- See 3.5.3, "Standard Keyboard."

CSTOPMCT (at ØF Ø1BH) Ø1H byte Type: #
- The initial value is ØØH.
- This flag indicates whether or not MCT processing is to be interrupted when the CTRL/STOP keys are pressed.
    = ØØH: MCT processing not to be interrupted.
    = Ø1H: MCT processing to be interrupted.
- This flag is reset by PSTBIOS.
- See 3.5.3, "Standard Keyboard."

IMSHFT (at ØF Ø1CH) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates the keyboard shift mode when the item keyboard is installed.
    = ØØH: Normal mode
    = 2ØH: Shift mode
- See 3.5.4, "Item Keyboard."

ITEMFLG (at ØF01DH) 01H byte Type: o
- The initial value is 00H.
- This flag is used to switch to the item mode on the standard keyboard.
    = 00H: Normal mode
    = 80H: Item mode
- When the item mode is specified, the keys corresponding to codes 40H to 7FH serve as item function keys.
- See 3.5.4, "Item Keyboard."

ISYSREG (at ØF01EH) 01H byte Type: o
- The initial value is 03H.
- This flag controls the STOP and CTRL/STOP functions when the item keyboard is installed.



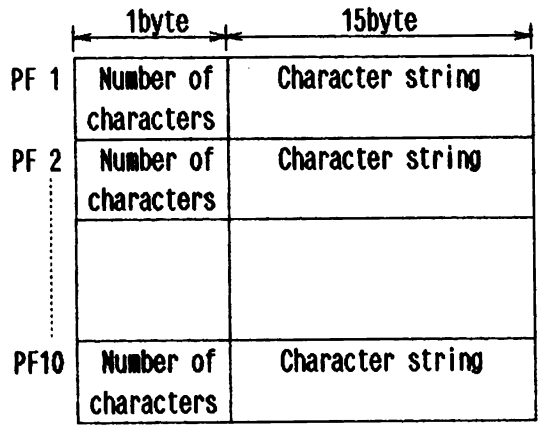
- See 3.5.4, "Item Keyboard."

DFLTRT (at ØF01FH) 0EH bytes Type: o
- Loaded with the default key codes of the arrow, SHIFT/arrow and CTRL/arrow keys.

Address	Key	Default value	Address	Key	Default value
0F01FH	→	1CH	0F026H	SHIFT /↓	FBH
0F020H	←	1DH	0F027H	SHIFT /INS	F6H
0F021H	↑	1EH	0F028H	CTRL /→	FCH
0F022H	↓	1FH	0F029H	CTRL /←	FDH
0F023H	SHIFT /→	0F8H	0F02AH	CTRL /↑	FEH
0F024H	SHIFT /←	0F9H	0F02BH	CTRL /↓	FFH
0F025H	SHIFT /↑	0FAH	0F02CH	CTRL /INS	F8H

- This area is copied into the 14-byte area starting at CHGRIGT.
- It is possible to modify key codes or functions by rewriting this area.
- See 3.5.3, "Standard Keyboard."

PFKTAB (at ØFØ2DH) AØH bytes Type: o
- PFKTAB contains the default definitions of the PF keys.



- The number of characters may be in the range of ØØH to ØFH. ØØH indicates that no character string is defined.
- This area is copied into the 16Ø-byte area starting at WPFKTBL (ØF545H).
- See 3.5.3, "Standard Keyboard."

YSUBRTN (at ØFØCDH) 1ØH bytes Type: o
- This is a jump table used when calling subroutines by using the CTRL/special keys.

Address	Key for activating subroutine	Default function
ØFØCDH	CTRL/ESC	Does nothing and return.
ØFØCFH	CTRL/PAUSE	Does nothing and return.
ØFØD1H	CTRL/HELP	System display.
ØFØD3H	CTRL/PF1	Does nothing and return.
ØFØD5H	CTRL/PF2	Does nothing and return.
ØFØD7H	CTRL/PF3	Does nothing and return.
ØFØD9H	CTRL/PF4	Does nothing and return.
ØFØDBH	CTRL/PF5	Screen dump.

- When rewriting this area, set each address to point to 8ØØØH or higher in RAM.
- See 3.5.3, "Standard Keyboard."

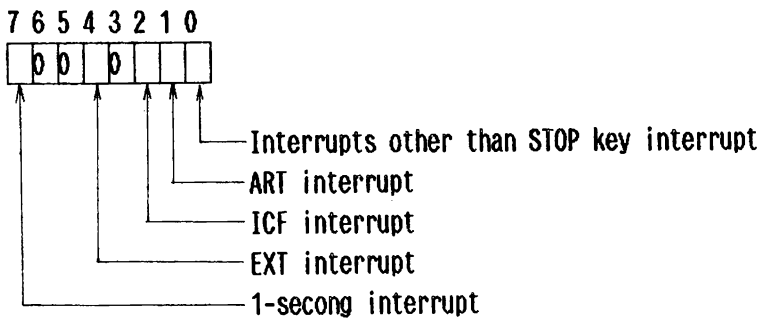
ITCDTB (at ØFØDDH) 18H bytes Type: o
- This is the redefinition table for the numeric keypad section on the item keyboard.

Address	Character on keytop	Default code	Address	Character on keytop	Default code
F0DDH	STOP	03H	F0E9H	=	3DH
DEH	△	1EH	EAH	4	34H
DFH	▽	1FH	EBH	5	35H
E0H	◀	1DH	ECH	6	36H
E1H	▶	1CH	EDH	,	2CH
E2H	*	2AH	EEH	1	31H
E3H	/	2FH	EFH	2	32H
E4H	+	2BH	F0H	3	33H
E5H	-	2DH	F1H	0	30H
E6H	7	37H	F2H	000	FFH
E7H	8	38H	F3H	.	2EH
E8H	9	39H	F4H	↵	0DH

- See 3.5.4, "Item Keyboard."

- It is possible to modify key codes by rewriting this area. ØFFH indicates that 'Ø' (3ØH) is repeated three times.

BPINTEBL (at ØFØF5H) Ø1H byte Type: o
- The initial value is Ø1H.
- This flag controls interrupts during BEEP processing.



(A 1 in a bit position disables the corresponding interrupt and a Ø has no effect.)

- OVF interrupts are always disabled during BEEP processing.
- This flag is set to avoid note fluctuations which would be caused by interrupts while generating a beep.
- See (20) BEEP in 3.4.2.

PROMPWR (at 0F0F6H) 01H byte

HSBKCD (at 0F0F7H) 01H byte Type: o
- The initial value is 0FH.
- Specifies the command code corresponding to a BREAK command for the HS-mode cartridge.

HSBKAK (at 0F0F8H) 01H byte Type: o
- The initial value is 02H.
- Specifies the ACK code corresponding to a BREAK command for the HS-mode cartridge.

AHSTWRT (at 0F0F9H) 01H byte Type: #
- The initial value is 00H.
- Indicates whether or not write data (to FDD drive D: or E:) is present.
  = 00H: No write data present
  = 31H: Write data present
- This flag is set when a Write or Read is executed on the FDD (D: or E:).
- See 3.4, "BIOS Details."

BHSTWRT (at 0F0FAH) 01H byte Type: #
- The initial value is 00H.
- Indicates whether or not write data (to FDD drive F: or G:) is present.
  = 00H: No write data pending
  = 32H: Write data pending
- This flag is set when a Write or Read is executed on the FDD (F: or G:).
- See 3.4, "BIOS Details."

ADSKOPN (at 0F0FBH) 01H byte Type: #
- The initial value is 00H.
- Indicates the FDD (D: or E:) open state.
  = 00H: Not opened
  = 0FFH: Opened
- This flag is set when BIOS SELDSK is executed on the FDD (D: or E:).
- See 3.4, "BIOS Details."

BDSKOPN (at 0F0FCH) 01H byte Type: #
- The initial value is 00H.
- Indicates the FDD (F: or G:) open state.
  = 00H: Not opened
  = 0FFH: Opened
- This flag is set when BIOS SELDSK is executed on the FDD (F: or G:).
- See 3.4, "BIOS Details."

DSKDID (at 0F0FDH) 01H byte Type: x
- The initial value is 31H.
- Indicates the EPSP DID code used when transmitting to or receiving from a FDD.
- 31H (D: or E:) or 32H (F: or G:) is valid.
- See 5.3, "Terminal Floppy (TF)."

DSKSID (at ØF0FEH) Ø1H byte Type: x
- The initial value is 23H.
- Indicates the EPSP SID code used when transmitting to or receiving from a FDD.
- The initial value (23H) indicates PINE.
- See 5.3, "Terminal Floppy (TF)."

DISKTBL (at ØF0FFH) ØBH bytes Type: o
- DISKTBL is the logical-to-physical drive assignment table.
- Loaded with the physical drive codes corresponding to the logical drive names.

Address	Logical drive name	Initial value
ØF0FFH	A:	ØØH
ØF100H	B:	Ø1H
ØF101H	C:	Ø2H
ØF102H	D:	Ø3H
ØF103H	E:	Ø4H
ØF104H	F:	Ø5H
ØF105H	G:	Ø6H
ØF106H	H:	Ø7H
ØF107H	I:	Ø8H
ØF108H	J:	Ø9H
ØF109H	K:	ØAH

- The correspondence between the physical drive codes and media is shown below.

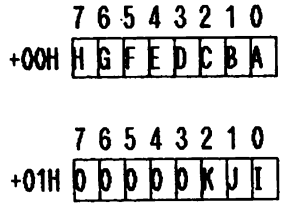
Physical drive code	Media	Physical drive code	Media
ØØH	RAM disk	Ø6H	FDD (G:)
Ø1H	ROM capsule 1	Ø7H	Microcassette
Ø2H	ROM capsule 2	Ø8H	RAM cartridge
Ø3H	FDD (D:)	Ø9H	ROM cartridge 1
Ø4H	FDD (E:)	ØAH	ROM cartridge 2
Ø5H	FDD (F:)		

- Certain points must be noted when modifying the contents of this table. Refer to Subsection 3.8.2.
- See 3.8, "Disk Storage."

ROMCPNO1 (at ØF1ØAH) Ø1H byte Type: o
- The initial value is Ø1H.
- Indicates the drive code for ROM capsule 1 (B:).
- Loaded with the ROM capsule 1 offset with respect to the beginning of DSKTBL.
- See 3.8, "Disk Storage."

ROMCPNO2 (at ØF1ØBH) Ø1H byte Type: o
- The initial value is Ø2H.
- Indicates the drive code for ROM capsule 2 (C:).
- Loaded with the ROM capsule 2 offset with respect to the beginning of DISKTBL.
- See 3.8, "Disk Storage."

DISKROV (at ØF1ØCH) Ø2H bytes Type: o
- The initial values are Ø6H, Ø6H.
- Indicates the disk read-only state.



(Each bit is set to 1 for read-only, and reset to Ø for read/write.)

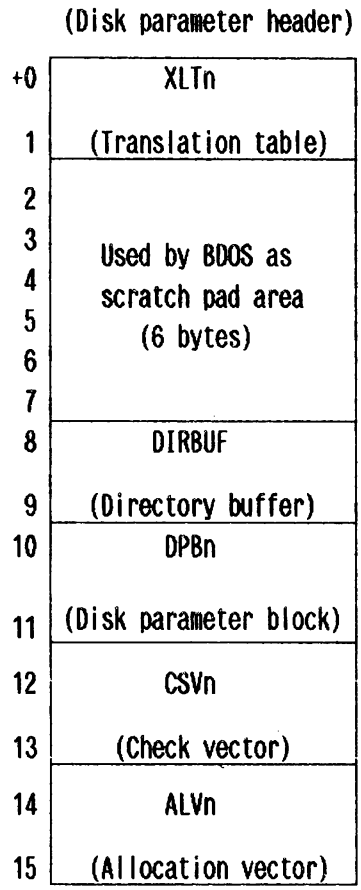
- The bits are associated with logical drives A: through K: as shown above.
- Logical drives B:, C:, J:, and K: are initially set to read-only.
- See 3.8, "Disk Storage."

FTSTAB (at ØF1ØEH) 16H bytes Type: x
- This jump table is used when BIOS SELDSK is called to select a disk for the first time.
- Loaded with the processing starting addresses of drives A: through K:. The length of each address is 2 bytes.
- See 3.4, "BIOS Details."

READTAB (at ØF124H) 16H bytes Type: x
- This the jump table is used to read a disk during execution of a BIOS READ call.
- Loaded with the processing starting addresses of drives A: through K:. The length of each address is 2 bytes.
- See 3.4, "BIOS Details."

WRTTAB (at ØF13AH) 16H bytes Type: x
- This jump table is used to write to a disk during execution of a BIOS WRITE call.
- Loaded with the processing starting addresses of logical drives A: through K:. The length of each address is 2 bytes.
- See 3.4, "BIOS Details."

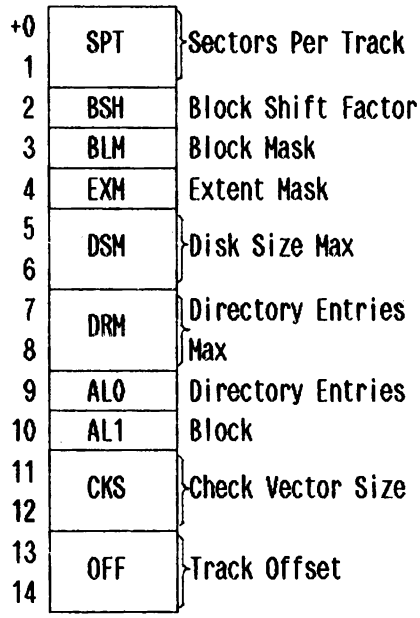
DPBASE (at ØF15ØH) BØH bytes Type: #
- DPBASE is the disk parameter header associated with each drive.
- Loaded with the 16-byte disk parameter headers corresponding to drives A: through K:.



- See 3.8, "Disk Storage."

DPBØ (at ØF2ØØH) ØFH bytes Type: #
- The disk parameter block for the RAM disk.

(Disk parameter block)



- See 3.8, "Disk Storage."

DPB1 (at ØF2ØFH) ØFH bytes Type: #
- The disk parameter block for ROM capsule 1.
- See 3.8, "Disk Storage."

DPB2 (at ØF21EH) ØFH bytes Type: #
- The disk parameter block for ROM capsule 2.
- See 3.8, "Disk Storage."

DPB3 (at ØF22DH) ØFH bytes Type: #
- The disk parameter block for the FDD.
- See 3.8, "Disk Storage."

DPB7 (at ØF23CH) ØFH bytes Type: #
- The disk parameter block for the MCT cartridge.
- See 3.8, "Disk Storage."

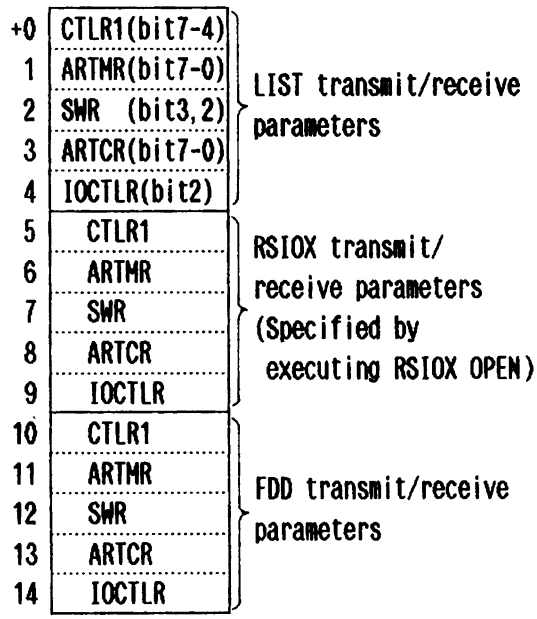
DPB8 (at ØF24BH) ØFH bytes Type: #
- The disk parameter block for the RAM cartridge.
- See 3.8, "Disk Storage."

DPB9 (at ØF25AH) ØFH bytes Type: #
- The disk parameter block for ROM cartridge 1.
- See 3.8, "Disk Storage."

DPB1Ø (at ØF269H) ØFH bytes Type: #
- The disk parameter block for ROM cartridge 2.
- See 3.8, "Disk Storage."

RSBYTE (at ØF278H) Ø1H byte

SRTABL (at ØF279H) ØFH bytes Type: o
- The initial values are 6ØH, 84H, Ø8H, 37H, Ø4H, 6ØH, 84H, Ø8H, 37H, Ø4H, BØH, Ø4H, Ø4H, 37H, and Ø4H.
- Loaded with the serial transmit/receive parameters for LIST, RSIOX and FDD.



- The bit assignments of this area are identical to those of the specified I/O register. Bits in parenthesis indicate valid bits.
- See 5.2, "Serial Interfaces."

EPTRCN (at ØF288H) Ø1H byte Type: o
- The initial value is Ø3H.
- Specifies the number of retries to be made when no response is received while sending or receiving EPSP data.
- See 5.3, "Terminal Floppy (TF)."

EPTIMO (at ØF289H) Ø1H byte Type: o
- The initial value is ØAH.
- Specifies the number of retries to be made when timeout conditions occur while receiving 1 byte of EPSP data.
- See 5.3, "Terminal Floppy (TF)."

EPETMO (at ØF28AH) Ø1H byte Type: o
- The initial value is 64H.
- Specifies the number of retries to be made when timeout conditions occur while receiving a block of EPSP data.
- See 5.3, "Terminal Floppy (TF)."

EPATMO (at ØF28BH) Ø1H byte Type: o
- The initial value is ØAH.
- Specifies the number of retries to be made when timeout conditions occur while receiving an ACK using EPSP protocol.
- See 5.3, "Terminal Floppy (TF)."

EPMODE (at ØF28CH) Ø1H byte Type: o
- The initial value is ØØH.
- Specifies the mode for EPSP data transmission/reception.
  = ØØH : Master mode (to send the header)
  = Other than ØØH: Slave mode (to omit sending the header)
- See 5.3, "Terminal Floppy (TF) "

EPETDL (at ØF28DH) Ø1H byte

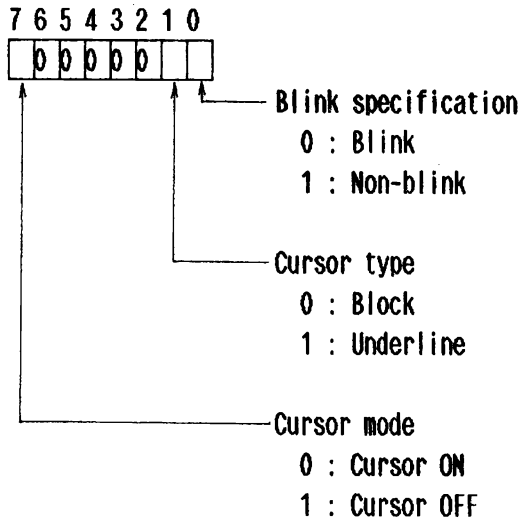
EP1BRTO (at ØF28EH) Ø2H bytes Type: o
- The initial value is 1C2FH.
- Specifies the number corresponding to the time when a timeout occurs after an EPSP data byte is received.
- The unit is approximately 13.86 μs. The initial value is equivalent to approximately 1ØØ msec.
- See 5.3, "Terminal Floppy (TF)."

LSCADDR (at ØF29ØH) Ø2H bytes Type: #
- The initial value is DØØØH.
- Indicates the starting address of the buffer for the currently displayed screen.
- See 3.6, "LCD Display (CONOUT Details)."

LSCSIZE (at ØF292H) Ø2H bytes Type: #
- The initial value is Ø8ØØH.
- Indicates the screen buffer size.
- LSCSIZE is equal to LSCSIZEX x LSCSIZEY.
- See 3.6, "LCD Display (CONOUT Details)."

LSCRVRAM (at ØF294H) Ø2H bytes Type: #
- The initial value is EØØØH.
- Indicates the starting address of the currently displayed VRAM.
- The result of LSCRVRAM + 1 (ØF295H) is loaded into VADR (PØ8H).
- See 3.6, "LCD Display (CONOUT Details)."

LCURSOR (at ØF296H) Ø1H byte Type: #
- The initial value is ØØH.
- This flag indicates the cursor state, including mode and type.



- See 3.6, "LCD Display (CONOUT Details)."

LCRVRSW (at ØF297H) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates whether or not the reverse display mode is set.
  = ØØH: Reverse display mode OFF
  = Ø1H: Reverse display mode ON
- See 3.6, "LCD Display (CONOUT Details)."

LSCCPOSX (at ØF298H) Ø1H byte Type: #
- The initial value is Ø1H.
- Indicates the horizontal position of the cursor on the screen.
  1 ≤ LSCCPOSX ≤ LSCSIZEX
- See 3.6, "LCD Display (CONOUT Details)."

LSCCPOSY (at ØF299H) Ø1H byte Type: #
- The initial value is Ø1H.
- Indicates the vertical position of the cursor on the screen.
  1 ≤ LSCCPOSY ≤ LSCSIZEY
- See 3.6, "LCD Display (CONOUT Details)."

LSCSIZEX (at ØF29AH) Ø1H byte Type: #
- The initial value is 5ØH.
- Indicates the number of columns on the screen.
- LSCSIZEX can be specified as 4Ø or 8Ø. The initial value specifies 8Ø columns in the horizontal direction.
- See 3.6, "LCD Display (CONOUT Details)."

LSCSIZEY (at ØF29BH) Ø1H byte Type: #
- The initial value is 19H.
- Indicates the number of lines on the screen.
- Any number within the range of 8 to 5Ø can be specified as LSCSIZEY. The initial value specifies 25 lines in the vertical direction. Note that LSCSIZEX x LSCSIZEY must not exceed 2ØØØ.
- See 3.6, "LCD Display (CONOUT Details)."

LWDXMIN (at ØF29CH) Ø1H byte Type: #
- The initial value is Ø1H.
- Indicates the upper left column position of the window on the screen.
- LWDXMIN can be specified as 1, 2Ø, or 4Ø.
- See 3.6, "LCD Display (CONOUT Details)."

LWDYMIN (at ØF29DH) Ø1H byte Type: #
- The initial value is Ø1H.
- Indicates the upper left line position of the window on the screen.
    1 ≤ LWDYMIN ≤ LSCSIZEY - 8
- See 3.6, "LCD Display (CONOUT Details)."

LWDCPOSX (at ØF29EH) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates the X-coordinate of the cursor relative to the window.
    = ØFDH : Inside the right margin
    = ØFEH : Inside the left margin
    = ØFFH : Outside the window
    = ØØH - 27H: Inside the window
- See 3.6, "LCD Display (CONOUT Details)."

LWDCPOSY (at ØF29FH) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates the Y-coordinate of the cursor relative to the window.
    = ØFFH : Outside the window
    = ØØH - Ø7H: Inside the window
- See 3.6, "LCD Display (CONOUT Details)."

LVRAMYOF (at ØF2AØH) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates the Y-direction offset of VRAM.
    ØØH ≤ LVRAMYOF ≤ 3FH
- LVRAMYOF is used when loading YOFF (PØ9H) with the result of the OR operation performed on DSPFLAG (ØEFB6H) and LVRAMYOF.
- See 3.6, "LCD Display (CONOUT Details)."

LWDTYPE (at ØF2A1H) Ø1H byte Type: #
- The initial value is Ø1H.
- Indicates where the window is located on the screen.
    = ØFEH: Right screen (scroll step = 4Ø)
    = ØFFH: Left screen (scroll step = 4Ø)
    = ØØH: Screen width is 4Ø
    = Ø1H: Left screen (scroll step = 2Ø)
    = Ø2H: Center screen (scroll step = 2Ø)
    = Ø3H: Right screen (scroll step = 2Ø)
- See 3.6, "LCD Display (CONOUT Details)."

LSECRETW (at ØF2A2H) Ø1H byte Type: #
- The initial value is ØØH.
- This flag indicates the secret mode on/off state.
    = ØØH: Secret mode off
    = Ø1H: Secret mode on
- See 3.6, "LCD Display (CONOUT Details)."

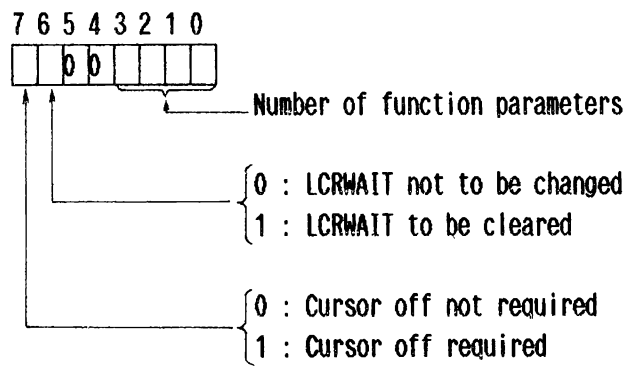
LSCROLMD (at ØF2A3H) Ø1H byte Type: #
- The initial value is Ø2H.
- This flag indicates the scroll mode which is currently specified.
  = ØØH: Tracking mode
  = Ø1H: Non-tracking mode
  = Ø2H: Horizontal non-tracking mode
- See 3.6, "LCD Display (CONOUT Details)."

LSCROLX (ØF2A4H) Ø1H byte Type: #
- The initial value is 14H.
- Indicates the number of horizontal scroll steps.
  LSCROLX = 14H or 28H
- Horizontal scrolling is available when LSCSIZEX = 8Ø.
- See 3.6, "LCD Display (CONOUT Details)."

LSCROLY (ØF2A5H) Ø1H byte Type: #
- The initial value is Ø1H.
- Indicates the number of vertical scroll steps.
  Ø1H ≤ LSCROLY ≤ Ø8H
- Vertical scrolling is available when LSCSIZEY >= Ø9H.
- See 3.6, "LCD Display (CONOUT Details)."

LCRWAIT (at ØF2A6H) Ø1H byte Type: #
- The initial value is ØØH.
- LCRWAIT is the carriage return check flag.
  = ØØH: Carriage return check mode off
  = Ø1H: Carriage return check mode on
- The carriage return check mode is turned on when the cursor moves from the end of a line to the beginning of the next line after a character is displayed by BIOS CONOUT. If CR (ØDH) is output with BIOS CONOUT immediately after this, the cursor moves to the beginning of the previous line.
- See 3.6, "LCD Display (CONOUT Details)."

LFKSTAT (at ØF2A7H) Ø1H byte Type: #
- Indicates the CONOUT function status.



- See 3.6, "LCD Display (CONOUT Details)."

LFKADDR (at ØF2A8H) Ø2H bytes Type: #
- Indicates the address of the CONOUT function processing routine.
- See 3.6, "LCD Display (CONOUT Details)."
LESCFLG (at ØF2AAH) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates whether or not an ESC sequence is received while calling CONOUT.
  = ØØH: ESC code not received
  = Ø1H: ESC code received
  = Ø2H: ESC code and function received
- See 3.6, "LCD Display (CONOUT Details)."

LESCONT (at ØF2ABH) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates the number of ESC sequence parameters received.
- See 3.6, "LCD Display (CONOUT Details)."

LESCPRM (at ØF2ACH) ØBH bytes Type: #
- Holds ESC sequence receive parameters.
- See 3.6, "LCD Display (CONOUT Details)."

LWORKBF (at ØF2B7H) 27H bytes Type: #
- The 39 bytes of data starting at LSCADDR is exchanged with the 39 bytes of data in this area when the screen mode is changed.
- Initially the system screen data is loaded in this area.
- See 3.6, "LCD Display (CONOUT Details)."
LSCMODE (at ØF2DEH) Ø1H byte Type: #
- The initial value is Ø1H.
- Indicates the currently set screen mode.
  = ØØH: System screen mode
  = Ø1H: User screen mode
- See 3.6, "LCD Display (CONOUT Details)."

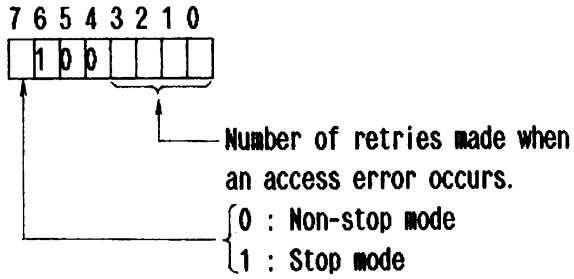
LLMARGIN (at ØF2DFH) Ø1H byte Type: #
- The initial value is Ø5H.
- Indicates the window left margin size.
  ØØH ≤ LLMARGIN ≤ ØAH
- See 3.6, "LCD Display (CONOUT Details)."

LRMARGIN (at ØF2EØH) Ø1H byte Type: #
- The initial value is 23H.
- Indicates the window right margin size.
  1EH ≤ LRMARGIN ≤ 28H
- See 3.6, "LCD Display (CONOUT Details)."

TAPMOD (at ØF2E1H) Ø1H byte Type: #
- The initial value is ØØH.
- TAPMOD is the tape mode (mount/remove) flag.
  = ØØH: Remove state
  = Ø1H: Mount state
- See 3.7, "MTOS/MIOS Operations."
MCTDCD (at ØF2E2H) Ø1H byte Type: #
- The initial value is Ø7H.
- Indicates the MCT device code.
- Used to check whether or not the drive specified in the FCB is an MCT.
- See 3.7, "MTOS/MIOS Operations."

ROMCT (at ØF2E3H) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates the MCT drive R/O status.
  = ØØH: R/W
  = Ø1H: R/O
- See 3.7, "MTOS/MIOS Operations."

DFTATR (at ØF2E4H) Ø1H byte Type: o
- The initial value is ØC2H.
- Contains the default data for TACATR (ØF7D1H), the tape access attribute flag.

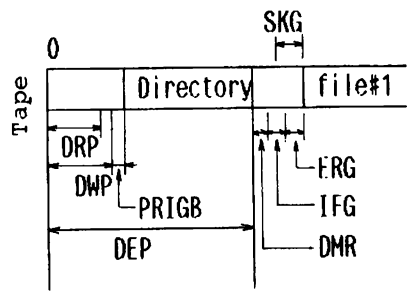


- See 3.7, "MTOS/MIOS Operations."

WRTATR (at ØF2E5H) Ø1H byte

SRDATR (at ØF2E6H) Ø1H byte Type: x
- The initial value is ØC2H.
- Is the tape access attribute flag used by the system when reading.
- The bit assignments are identical to those of DFTATR (ØF2E4H).
- See 3.7, "MTOS/MIOS Operations."

DRP (at ØF2E7H) Ø2H bytes Type: o
- This counter is initialized to ØØ3CH.
- The counter specifying the position in the MCT directory where reading is to start.



- See 3.7, "MTOS/MIOS Operations."

DWP (at ØF2E9H) Ø2H bytes Type: o
- This counter is initialized to ØØ4DH.
- The Counter specifying the position in the MCT directory where a write is to start.
- See 3.7, "MTOS/MIOS Operations."

DEP (at ØF2EBH) Ø2H bytes Type: o
- This counter is initialized to ØØD9H.
- The counter specifying the upper limit of the end of the directory area.
- See 3.7, "MTOS/MIOS Operations."

DMR (at ØF2EDH) Ø2H bytes Type: o
- The initial value is Ø068H.
- Specifies the length of the gap between the directory and the file areas on MCT.
- See 3.7, "MTOS/MIOS Operations."

IFG (at ØF2EFH) Ø2H bytes Type: o
- The initial value is Ø068H.
- Specifies the length of the gap between files on MCT.
- See 3.7, "MTOS/MIOS Operations."

SKG (at ØF2F1H) Ø2H bytes Type: o
- The initial value is Ø034H.
- Specifies the interval between seeks during an MCT file seek.
- See 3.7, "MTOS/MIOS Operations."

ERG (at ØF2F3H) Ø2H bytes Type: o
- The initial value is Ø034H.
- Specifies the length of the gap up to the point where a file write is to start.
- See 3.7, "MTOS/MIOS Operations."

LERG (at ØF2F5H) Ø2H bytes Type: o
- The initial value is Ø04EH.
- Specifies the length of the trailing gap up to the point where the MCT file is closed.
- See 3.7, "MTOS/MIOS Operations."

BKG (at ØF2F7H) Ø2H bytes

PRIGB (at ØF2F9H) Ø2H bytes

CCDFLT (at ØF2FBH) Ø1H byte Type: o
- The initial value is Ø5H.
- Loaded with the tape correction count.
- See 3.7, "MTOS/MIOS Operations."

CRCCNT (at ØF2FCH) Ø1H byte Type: #
- The initial value is Ø5H.
- Is the counter that counts the number of tape count corrections.
- This counter is decremented by one each time a file is created. When the counter reaches ØØH, the MCT rewinds the tape to the beginning and the tape count value is corrected. The correction count is specified in CCDFLT (ØF2FBH).
- See 3.7, "MTOS/MIOS Operations."

BLKWRTNO (at ØF2FDH) Ø1H byte Type: o
- The initial value is Ø3H.
- Indicates the number of times a block is to be re-written.
- The initial value indicates twice.
- See 3.7, "MTOS/MIOS Operations."

LMCTRED (at ØF2FEH) Ø4H bytes Type: o
- The initial value is 3EH, ØFFH, 18H, Ø2H.
- Indicates the error return point entered during an MCT read executed through BIOS.
- Loaded with instruction codes. Initially the codes load ØFFH into BIOS ERROR (ØF52BH).
- See 3.7, "MTOS/MIOS Operations."

LMCTWRT (at ØF3Ø2H) Ø6H bytes Type: o
- The initial value is 3EH, ØFFH, 32H, 2BH, ØF5H, ØC9H.
- Indicates the error return point entered during an MCT write executed through BIOS.
- Loaded with instruction codes. Initially the codes load ØFFH into BIOSERROR (ØF52BH).
- See 3.7, "MTOS/MIOS Operations."

ALRMTIME (at ØF3Ø8H) ØlH byte Type: o
- The initial value is 32H.
- Specifies the alarm screen display time in seconds.
- When the initial value is set, the screen automatically switches from the alarm screen mode back to the original mode after 5Ø seconds.
- See 2.9, "Alarm/Wake Function."

CSTOPPRT (at ØF3Ø9H) ØlH byte Type: #
- The initial value is ØØH.
- This flag is used to interrupt cartridge printer processing when the CTRL/STOP key is pressed.
    = ØØH: Cartridge printer processing not to be interrupted.
    = ØlH: Cartridge printer processing to be interrupted.

This flag is reset by PSTBIOS.
- See 3.5.3, "Standard Keyboard."

(Reserved) (at ØF3ØAH) Ø6H bytes

6.2.4 System Area III (RSYSAR3)

This area is initialized by system initialize, reset, restart, power-on, and WBOOT processing.

CNTNFG (at ØF31ØH) ØlH byte Type: x
- The initial value is ØFFH.
- This flag indicates whether the power-off mode.
    = ØØH: Continue mode
    = ØFFH: Restart mode
- When power is turned on, this flag is set to power-on mode.
- See 2.5, "Power-off."

FRCECNTN (at ØF311H) ØlH byte Type: o
- The initial value is ØØH.
- This flag specifies the power-off mode when an standard keyboard is installed.
    = ØØH : Restart power off mode
    = Other than ØØH: Continue power off mode
- See 2.5, "Power-off."

IFRCECNT (at ØF312H) ØlH byte Type: o
- The initial value is ØFFH.
- This flag specifies the power-off mode when an item keyboard is installed.
    = ØØH : Restart power off mode
    = Other than ØØH: Continue power off mode
- See 2.5, "Power-off."

TMFUNC (at ØF313H) Ø1H byte Type: o
- The initial value is ØØH.
- TMFUNC is the user timer function on/off flag.
    = ØØH   : Timer function OFF
    = Other than ØØH: Timer function ON
- See 4.7.4, "75Ø8 Interrupts."

TMFLAG (at ØF314H) Ø1H byte Type: o
- The initial value is ØØH.
- This flag indicates whether or not setting of the user timer function is completed.
    = ØØH: Not completed
    = ØFFH: Completed
- The value of TMSEC (ØF4DAH) is decremented by one each time a 1-second interrupt occurs. When the value reaches ØØØØH, this flag is set to ØFFH.
- See 4.7.4, "75Ø8 Interrupts."

BFLASHDT (at ØF315H) Ø1H byte

ROMEXQ (at ØF316H) Ø1H byte Type: x
- The initial value is ØØH.
- This flag is used by the system when executing a ROM-based program.
    = ØØH   : ROM-based program not executed
    = Other than ØØH: ROM-based program executed
- See 4.6, "Executing a ROM Program."

COMPCOL (at ØF317H) Ø1H byte Type: x
- The initial value is ØØH.
- A work area used by BDOS to display characters.
- See 3.2, "BDOS Operations."

STRTCOL (at ØF318H) Ø1H byte Type: x
- The initial value is ØØH.
- A work area used by BDOS to display characters.
- See 3.2, "BDOS Operations."

COLUMN (at ØF319H) Ø1H byte Type: x
- The initial value is ØØH.
- A work area used by BDOS to display a character.
- See 3.2, "BDOS Operations."

LISTCP (at ØF31AH) Ø1H byte Type: o
- The initial value is ØØH.
- This toggle switch is used by BDOS to indicate whether or not to output to the LIST device.
    = ØØH: No LIST output
    = Ø1H: LIST output
- See 3.2, "BDOS Operations."

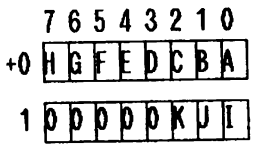
KPCHAR (at ØF31BH) Ø1H byte Type: x
- The initial value is ØØH.
- A work area used by BDOS during keyboard interrupt processing.
    = ØØH   : No keyboard interrupt occurred.
    = Other than ØØH: Key interrupt occurred (input key code).
- See 3.2, "BDOS Operations."

USRCODE (at ØF31CH) Ø1H byte Type: #
- The initial value is ØØH.
- Loaded with the currently specified user number.
- See 3.2, "BDOS Operations."

CURDSK (at ØF31DH) Ø1H byte Type: #
- The initial value is ØØH.
- Loaded with the current disk drive code.
- See 3.2, "BDOS Operations."

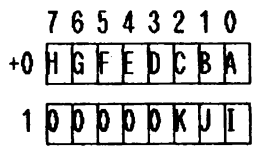
EFCB (at ØF31EH) Ø1H byte Type: x
- The initial value is ØE5H.
- Indicates the code used to search the directory for a free entry when a file is created by BDOS.
- See 3.2, "BDOS Operations."

RODSK (at ØF31FH) Ø2H bytes Type: #
- The initial value is ØØØØH.
- Loaded with the R/O disk vector.



(Each bit indicates R/O when set to 1 and R/W when Ø.)
- See 3.2, "BDOS Operations."

DLOG (at ØF321H) Ø2H bytes Type: #
- The initial value is ØØØØH.
- Loaded with the log-in disk vector value.



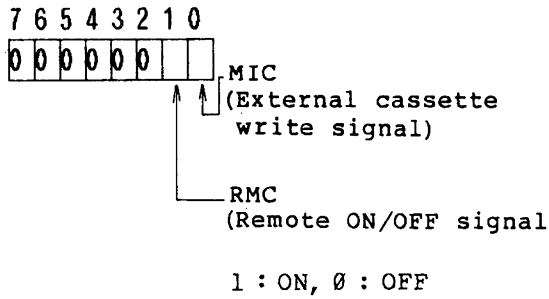
(Each bit indicates a selected drive when set to 1.)

- See 3.2, "BDOS Operations."

DMAAD (at ØF323H) Ø2H bytes Type: #
- Indicates the starting address of the DMA buffer used for BDOS processing.
- The DMA buffer is located at SYSDMA (ØF8B1H).
- See 3.2, "BDOS Operations."

USRDMA (at ØF325H) Ø2H bytes Type: #
- The initial value is ØØ8ØH.
- Indicates the starting address of the user DMA buffer used in BDOS processing.
- See 3.2, "BDOS Operations."

RZCTLR2 (at ØF327H) Ø1H byte Type: o
- The initial value is ØØH.
- This area stores the output state of control register 2 (PØ2H).



- The bit assignments for this area are identical for those of CTLR2 (PØ2H).
- See Chapter 5, "I/O Operations."

CHGRIGT (at ØF328H) ØEH bytes Type: o
- Is the table that contains the arrow, SHIFT/arrow and CTRL/arrow key code.

Address	Key	Initial value	Address	Key	Initial value
F328H	→	1CH	F32FH	SHIFT /↓	11H
29H	←	1DH	30H	SHIFT /INS	F6H
2AH	↑	1EH	31H	CTRL /→	16H
2BH	↓	1FH	32H	CTRL /←	17H
2CH	SHIFT /→	06H	33H	CTRL /↑	14H
2DH	SHIFT /←	02H	34H	CTRL /↓	15H
2EH	SHIFT /↑	10H	35H	CTRL /INS	F7H

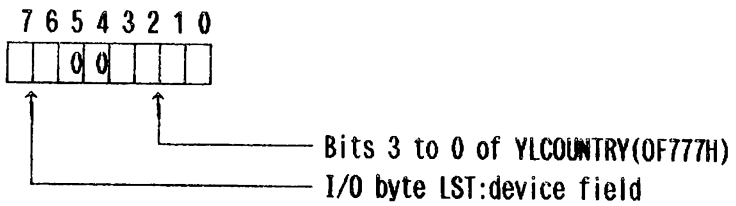
- The 14-byte data block starting at DFLTRT is copied into this area at warm boot time.
- The initial values are used for keyboard interrupt processing on the system menu screen.
- See 3.5.3, "Standard Keyboard."

RNSCOF (at ØF336H) Ø1H byte

INHCOPY (at ØF337H) Ø1H byte Type: x
- The initial value is ØØH.
- This flag indicates whether or not screen dump processing is in progress.
    = ØØH   : Not in progress
    = Other than ØØH: In progress
- See 3.4, "BIOS Details."

MDMMOD (at ØF338H) Ø1H byte Type: o
- The initial value is ØØH.
- This flag is used to disable screen dump processing via the RS-232C or SIO interface.
    = ØØH   : SCRNDUMP enabled
    = Other than ØØH: SCRNDUMP disabled
- A beep is generated as an alarm if an attempt is made to execute SCRNDUMP when screen dump processing is disabled.
- Refer to (19) SCRNDUMP in 3.4.2.

PRTFLG (at ØF339H) Ø1H byte Type: #
- The initial value is ØØH.
- This flag indicates whether or not an international character set has been indicated to the printer.
    = ØØH   : No output was received
    = Other than ØØH: Output was received
- The figure below shows the bit assignments for this area (when PRTFLG is not set to ØØH) for Export version 1.Ø.



- See (6) LIST in 3.4.2.

DISBRK (at ØF33AH) Ø1H byte Type: x
- The initial value is ØØH.
- This flag is used by the system to control the power-off and alarm screen display modes during screen dump processing.
    = ØØH   : Power-off, alarm screen displays enabled
    = Other than ØØH: Power-off, alarm screen displays disabled
- See 3.4, "BIOS Details."

RSCLSF (at ØF33BH) Ø1H byte

SRMODE (at ØF33CH) Ø1H byte Type: #
- The initial value is ØFFH.
- Indicates the current serial interface state.
    = ØFFH: Unused
    = ØØH: Used by printer (LIST)
    = Ø1H: Used by the user (RSIOX)
    = Ø2H: Used by FDD
- See 5.2, "Serial Interfaces."

ECSNCPWC (at ØF33DH) Ø1H byte Type: o
- The initial value is 5ØH.
- Specifies the time-count by which to write synchronous signals when writing synchronous patterns to the external cassette.
- This area is used in BASIC external cassette processing.

ECSNCPRC (at ØF33EH) Ø1H byte Type: o
- The initial value is 28H.
- Specifies the time-count by which to read synchronous signals which must be checked when reading synchronous patterns from the external cassette.
- This area is used by BASIC for external cassette processing.

ECKEYWRD (at ØF33FH) Ø1H byte Type: o
- The initial value is ØØH.
- Specifies the mask pattern for data protection used during a read from or write to the external cassette.
- Data is XORed with this pattern before being written to the external cassette.
- This area is used by BASIC for external cassette processing.

ECB1TIME (at ØF34ØH) Ø2H bytes Type: o
- The initial value is ØØ72H.
- Specifies the width of the waveform of 1 data bits written to the external cassette.
- The initial value corresponds to approximately 5ØØ μs.
- This value is used by BASIC for external cassette processing.

ECBØTIME (at ØF342H) Ø2H bytes Type: o
- The initial value is ØØ32H.
- Specifies the width of Ø data bits written to the external cassette.
- The initial value corresponds to approximately 25Ø μs.
- This value is used by BASIC for external cassette processing.

ECMCTTMR (at ØF344H) Ø2H bytes Type: o
- The initial value is Ø1D5H.
- Specifies the threshold value for Ø and 1 bits read from the external cassette.
- The unit is approximately 1.63 μs. The initial value corresponds to approximately 75Ø μs.
- This value is used by BASIC for external cassette processing.

ECMCTØTH (at ØF346H) Ø2H bytes Type: o
- The initial value is ØØ69H.
- Specifies the threshold value used to distinguish Ø bits read from the external cassette from noises.
- The unit is approximately 1.63 μs. The initial value corresponds to approximately 17Ø μs.
- This value is used for BASIC external cassette processing.

ECCRCPL (at ØF348H) Ø2H bytes Type: o
- The initial value is 8408H.
- Contains the mask pattern (CRC-CCITT) used during external cassette CRC.
- This pattern is used by BASIC for external cassette processing.

ECSBIGLN (at ØF34AH) Ø1H byte Type: o
- The initial value is ØAH.
- Specifies the external cassette interblock gap generated during the second write.
- This area is used by BASIC for external cassette processing.

ECLBIGLN (at ØF34BH) Ø1H byte Type: o
- The initial value is 64H.
- Specifies the external cassette interblock gap.
- This area is used by BASIC for external cassette processing.

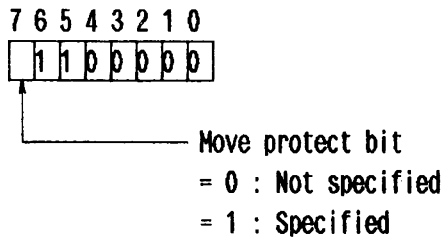
ECLFTP (at ØF34CH) Ø5H bytes

TOSDMA (at ØF351H) Ø2H bytes Type: #
- The initial value is TOSBUF (ØFB97H).
- Indicates the MCT DMA buffer starting address.
- See 3.7, "MTOS/MIOS Operations."

ALCVAD (at ØF353H) Ø2H bytes Type: #
- The initial value is ALV7 (ØF735H).
- Indicates the MCT allocation vector starting address.
- See 3.7, "MTOS/MIOS Operations."

OPNMOD (at ØF355H) Ø1H byte Type: #
- The initial value is ØØH.
- Indicates the current MCT open state.
    = ØØH: Not open
    = Ø1H: Read mode open
    = Ø2H: Write mode open
- See 3.7, "MTOS/MIOS Operations."

MISSTS (at ØF356H) Ø1H byte Type: o
- The initial value is ØEØH.
- This flag specifies the MIOS processing mode.



- When the move protect bit is set, a move protect area is established for the number of counts specified in TAPMVC(ØF35AH) during MIOS MIRDBL or MIWTBL.
- See 3.7, "MTOS/MIOS Operations."

BMDCNT (at ØF357H) Ø1H byte Type: o
- The initial value is Ø2H.
- Specifies the number of retries to be made after a block mode error occurs during a read operation.
- See 3.7, "MTOS/MIOS Operations."

RRBKMD (at ØF358H) Ø1H byte

RNIDPT (at ØF359H) Ø1H byte

TAPMVC (at ØF35AH) Ø2H bytes Type: o
- The initial value is ØØC8H.
- Indicates the default count value of the move protect area used by the system.
- See 3.7, "MTOS/MIOS Operations."

RLCGENX (at ØF35CH) Ø2H bytes Type: o
- The initial value is 5960H (In OSROM)
- Indicates the starting address of the character generator data for character codes ØØH to 1FH.
- When rewriting this address, set RLCGENX to address 8ØØØH or higher in RAM.
(This function is supported only by OS Kana V2.Ø.)
- See 3.6, "LCD Display (CONOUT Details)."

RLCGENN (at ØF35EH) Ø2H bytes Type: o
- The initial value is 5960H (In OSROM)
- Indicates the starting address of the character generator data for character codes 2ØH to 7FH.
- When rewriting this address, set RLCGENN to address 8ØØØH or higher in RAM.
(This function is supported only by OS Kana V2.Ø.)
- See 3.6, "LCD Display (CONOUT Details)."

RLCGENG (at ØF360H) Ø2H bytes Type: o
- The initial value is 5B40H (In OSROM)
- Indicates the starting address of the character generator data for character codes 8ØH to 9FH.
- When rewriting this address, set RLCGENG to address 8ØØØH or higher in RAM.
(This function is supported only by OS Kana V2.Ø.)
- See 3.6, "LCD Display (CONOUT Details)."

RLCGENK (at ØF362H) Ø2H bytes Type: o
- The initial value is 5CØØH (In OSROM)
- Indicates the starting address of the character generator data for character codes ØAØH to ØDFH.
- When rewriting this address, set RLCGENK to address 8ØØØH or higher in RAM.
(This function is supported only by OS Kana V2.Ø.)
- See 3.6, "LCD Display (CONOUT Details)."

(Reserved) (at ØF364H) Ø4H bytes

6.2.5 System Area IV (RSYSAR4)

This area is the system parameter area, which does not need to be initialized.

XUSRBIOE (at ØF368H) Ø3H bytes Type: x
- Loaded with the JP instruction to the user BIOS area starting address.
- This area is loaded according to the BIOS area size when the system is initialized.
- See 4.1, "User BIOS."

CNTNSP (at ØF36BH) Ø2H bytes Type: x
- This area saves the currently specified stack pointer before power is turned off in the continue mode.
- See 2.5, "Power-off."

CNTNILVL (at ØF36DH) Ø1H byte Type: x
- This area saves the currently specified 7508 interrupt level (INTLEVEL) before executing power-off processing in the continue mode.
- See 2.5, "Power-off."

YPWSWST (at ØF36FH) Ø1H byte Type: #
- This area is used by the system to store the current power switch state.
  - = ØØH: Power switch off state
  - = Ø1H: Power switch on state

YMAINST (at ØF37ØH) Ø1H byte Type: #
- This area is used by the system to control main CPU Z-8Ø power.
  - = ØØH: Power off command transmitted to the 7508
  - = Ø1H: Operating in power on state

ZSTARTFG (at ØF371H) Ø1H byte Type: x
- This flag is used by the system to specify Ø address start processing.
  - = ØØH: Executes warm boot processing.
  - = Ø1H: Executes power on processing.
  - = Ø2H: Executes alarm processing.
  - = Ø3H: Executes wake processing.
  - = Ø5H: Executes reset processing.
  - = Ø6H: Executes system initialize processing.
- This flag is reset by WBOOT.

CHKSUMFG (at ØF372H) Ø1H byte

CHKSUM (at ØF373H) Ø2H bytes Type: x
- This area is used to store the system area checksum result.
- At next power-on time, the checksum is calculated again to check whether or not the contents of RAM has been destroyed.
- See 2.5, "Power-off."

WKEYWORK (at ØF375H) 16H bytes Type: x
- This area is used to save the 22-byte data starting at KEYF (ØFØØ8H) when the alarm screen is displayed.
- See 2.9, "Alarm/Wake."

WKBUF (at ØF38BH) 21H bytes Type: x
- This area is used to save the 33-byte data starting at KBUF (ØF86ØH) when the alarm screen is displayed.
- See 2.9, "Alarm/Wake".

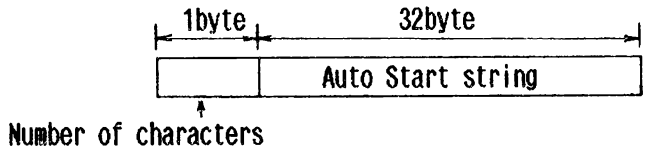
WREIOCTLR (at ØF3ACH) Ø1H byte Type: x
- This area is used to save the value of RZIOCTLR (ØFØØ6H) when the alarm screen is displayed.
- See 2.9, "Alarm/Wake."

WYSHFDT (at ØF3ADH) Ø1H byte Type: x
- This area is used to save the current YSHFDT (ØFØØFH) data to process the old keyboard data, which had been entered before the alarm screen was displayed, when exiting the alarm screen.
- See 2.9, "Alarm/Wake."

WIMSHFT (at ØF3AEH) Ø1H byte Type: x
- This area is used to save the current IMSHFT (ØF01CH) data to process the old keyboard data, which had been entered before the alarm screen was displayed, when exiting the alarm screen.
- See 2.9, "Alarm/Wake."

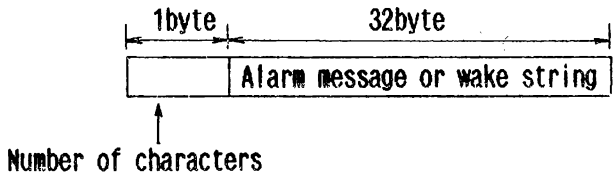
ARRWWORK (at ØF3AFH) ØEH bytes Type: x
- This area is used to save the 14-byte data starting at CHGRIGT (ØF328H) when displaying the system display screen.
- See 2.11, "System Display."

AUTOSTRT (at ØF3BDH) 22H bytes Type: o
- This area is used to specify the auto start string.



- The number of characters must be within the range of ØØH to 2ØH. ØØH indicates that no auto start string is specified.
- See 2.7, "Auto Start Function."

ALRMMSG (at ØF3DFH) 22H bytes Type: o
- This area is used to specify the alarm message or wake string.



- The number of characters is within the range of ØØH to 2ØH. ØØH indicates that neither alarm message nor wake string are specified.
- See 2.9, "Alarm/Wake Function."

STIMBUF (at ØF4Ø1H) 18H bytes Type: x
- This area is used to display the currently set date, time and day of the week on the menu and system display screens.

SSEDITWK (TIMEWK) (at ØF419H) 20H bytes Type: x
- This area is used by the system to edit the input key.
(Used during system initialize, menu and system display processing)

TPCNTWK (at ØF439H) Ø2H bytes

MFILENO (at ØF43BH) Ø1H byte Type: x
- Indicates the number of files to be displayed on the menu screen.
- See 2.1Ø, "Menu."

MLASTPAGE (at ØF43CH) Ø1H byte Type: x
- Indicates the total number of pages to be displayed on the menu screen.
- See 2.1Ø, "Menu."

MLASTFX (at ØF43DH) Ø1H byte Type: x
- Indicates the horizontal position of the last file name to be displayed on the menu screen.
- See 2.1Ø, "Menu."

MLASTFY (at ØF43EH) Ø1H byte Type: x
- Indicates the vertical position of the last file name to be displayed on the menu screen.
- See 2.1Ø, "Menu."

MFOVER (at ØF43FH) Ø1H byte Type: x
- This flag indicates that the number of files to be displayed on the menu screen exceeds 72.
- See 2.1Ø, "Menu."

MCPAGE (at ØF44ØH) Ø1H byte Type: x
- Indicates the page number of the page which is currently displayed on the menu screen.
- See 2.1Ø, "Menu."

MCFX (at ØF441H) Ø1H byte Type: x
- Indicates the horizontal position of the file name which is currently displayed on the menu screen.
- See 2.1Ø, "Menu."

MCFY (at ØF442H) Ø1H byte Type: x
- Indicates the vertical position of the file name which is currently displayed on the menu screen.
- See 2.1Ø, "Menu."

MCMNDLY (at ØF443H) Ø1H byte Type: x
- Indicates the vertical position of the command line displayed on the menu screen.
- See 2.1Ø, "Menu."

MCMNDLWK (INPWK) (at ØF444H) 29H bytes Type: x
- This area is used to edit the input data on the command line on the menu screen.
- See 2.1Ø, "Menu."

MCRSRPOS (at ØF46DH) Ø1H byte Type: x
- Indicates the horizontal position of the cursor currently displayed on the command line on the menu screen.
- See 2.1Ø, "Menu."

MCFTYPE (at ØF46EH) Ø1H byte Type: x
- Indicates the type of the file which is currently specified on the menu screen.
- See 2.1Ø, "Menu."

MVSPOSX (at ØF46FH) Ø1H byte Type: x
- Indicates the horizontal position on the virtual screen of the file which is currently specified on the menu screen.
- See 2.1Ø, "Menu."
MVSPOSY (at ØF47ØH) Ø1H byte Type: x
- Indicates the vertical position on the virtual screen of the file which is currently specified on the menu screen.
- See 2.1Ø, "Menu."

MLCFX (at ØF471H) Ø1H byte Type: x
- This area is used to record the old MCFX to restore the old cursor field to normal video when displaying a file name in reverse video on the menu screen.
- See 2.1Ø, "Menu."

MLCFY (at ØF472H) Ø1H byte Type: x
- This area is used to record the old MCFY to restore the old cursor field to normal video when displaying a file name in reverse video on the menu screen.
- See 2.1Ø, "Menu."

MTIMEBUF (at ØF473H) 18H bytes Type: x
- Used as a work area to display the header (date, time, day of the week) on the menu screen.
- See 2.1Ø, "Menu."

MCOMFNO (at ØF48BH) Ø1H byte Type: x
- This flag indicates whether or not COM is specified in FTYPETBL during menu processing.
- See 2.1Ø, "Menu."

SVMENUDR (ØF48CH) 43H bytes Type: x
- This area is used by system display to save the 67-byte data at currently specified MENUDR when modifying the menu display drive specification during menu processing.
- See 2.1Ø, "Menu."

MENUSTS (at ØF4CFH) Ø1H byte Type: x
- This flag indicates that system display has modified the menu display drive.
- See 2.1Ø, "Menu."

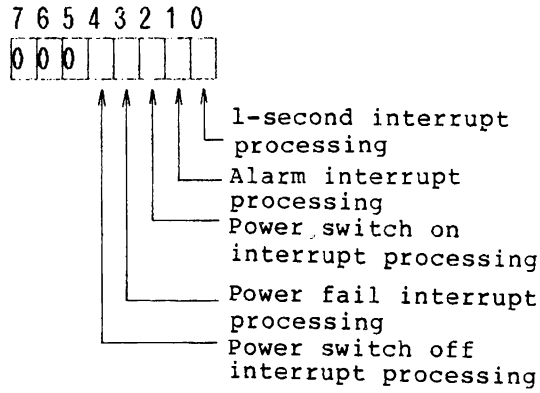
MBRCKFG (at ØF4DØH) Ø1H byte Type: x
- This flag indicates the display status when displaying a file name in reverse video on the menu screen.
- See 2.1Ø, "Menu"

ROMCT (at ØF4D1H) Ø1H byte Type: x
- This counter is used when turning off the ROM cartridge power due to a 1-second interrupt.
- Ø2H is loaded when accessing the ROM cartridge.
- See 5.1, "Cartridges."

ROMOFG (at ØF4D2H) Ø1H byte Type: x
- Indicates whether or not the ROM cartridge power has been turned off by a 1-second interrupt.
  = ØØH: No ROM cartridge power off required.
  = ØFFH: ROM cartridge power is turned off.
- See 5.1, "Cartridges."

ENTSINT (at ØF4D3H) Ø2H bytes Type: x
- This area is used during system interrupt processing to save the stack pointer when an interrupt occurs.
- See 4.7, "Interrupts."

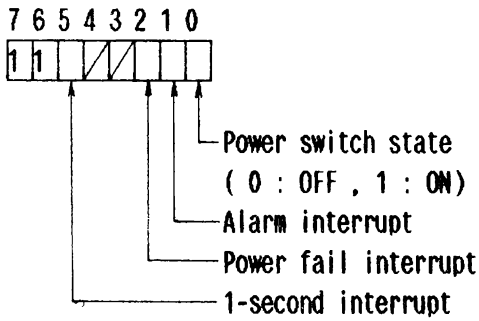
INTFG (at ØF4D5H) Ø1H byte Type: #
- Indicates the type of 75Ø8 interrupt processing initiated by the system.



(Each bit is set to 1 when the corresponding interrupt processing is started.)

- INTFG is set to ØØH when keyboard interrupt processing is executed.
- See 4.7.4, "75Ø8 Interrupts."

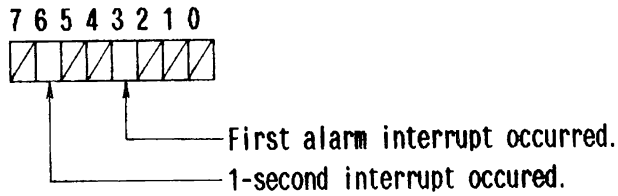
STS75Ø8 (at ØF4D6H) Ø1H byte Type: #
- This area is used by OS to store the 75Ø8 status when a 75Ø8 interrupt occurs.
- When the value of STS75Ø8 is between ØØH and ØBFH, it indicates that a keyboard interrupt has occurred and gives a hard key code.
- When STS75Ø8 is more than ØCØH, it indicates the following:



(Each bit is set to 1 when an interrupt is generated.)

- See 4.7.4, "75Ø8 Interrupts."

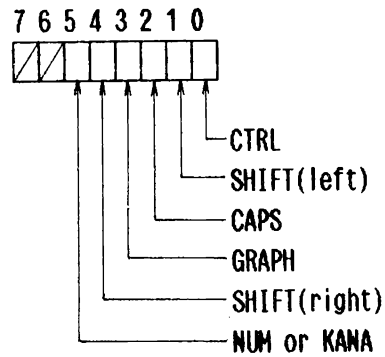
FG7508 (at 0F4D7H) 01H byte Type: #
- Indicates 1-second and alarm interrupt processing states.



(Each bit is set to 1 when the corresponding interrupt occurs.)

- See 4.7.4, "7508 Interrupts."

KEYFLG (at 0F4D8H) 01H byte Type: #
- Indicates the switch key state when a 7508 interrupt is generated by the STOP key.



(Each bit is set to 1 when the corresponding key is pressed.)

- See 4.7.4, "7508 Interrupts."

ALRMCT (at 0F4D9H) 01H byte Type: x
- This counter is used to ignore any alarm interrupts generated within 10 seconds after the first interrupt generated for the currently-set alarm time.
- 0AH is set when the first interrupt is generated.
- See 2.9, "Alarm/Wake."

TMSEC (at 0F4DAH) 02H bytes Type: o
- Specifies the user timer function time in seconds.
- The timer function is executed using TMFUNC (0F313H).
- See 4.7.4, "7508 Interrupts."

PWSWONFG (at 0F4DCH) 01H byte Type: #
- Indicates whether or not a power switch on interrupt has occurred.
  = 00H: No power switch on interrupt has occurred.
  = 0FFH: Power switch on interrupt has occurred.
- See 4.7.4, "7508 Interrupts."

PWSWOFFG (at ØF4DDH) Ø1H byte Type: #
- Indicates whether or not a power switch off interrupt has occurred.
    = ØØH: No power switch off interrupt has occurred.
    = Ø1H: Restart mode power switch off has occurred.
    = ØFFH: Continue mode power switch off has occurred.
- See 4.7.4, "75Ø8 Interrupts."

AFTER3 (at ØF4DEH) Ø1H byte Type: x
- Used to check the alarm screen display termination time.
- See 2.9, "Alarm/Wake."

WFUNCFLG (at ØF4DFH) Ø1H byte Type: x
- Loaded with the value of YPFCMFLG (ØFØ17H) when the system screen is displayed.
- See 2.11, "System Display."

INPSTOP (at ØF4EØH) Ø1H byte

ENTSP (at ØF4E1H) Ø2H bytes Type: x
- Loaded with the user stack pointer when a BDOS function is called.
- See 3.2, "BDOS Operations."

INFO (at ØF4E3H) Ø2H bytes Type: x
- A work area used by BDOS to save registers parameters.
- See 3.2, "BDOS Operations."

ARET (at ØF4E5H) Ø2H bytes Type: #
- Used to store the return code from BDOS.
- The value of ARET corresponds to the A register, and the value of ARET + 1 to the B register.
- This area is affected by BDOS functions.
- See 3.2, "BDOS Operations."

CDRMAXA (at ØF4E7H) Ø2H bytes Type: x
- Points to the area which is loaded by BDOS with the maximum number of the current disk directories.
- See 3.2, "BDOS Operations."

CURTRKA (at ØF4E9H) Ø2H bytes Type: x
- Points to the area which is loaded by BDOS with the currently specified track number.
- See 3.2, "BDOS Operations."

CURRECA (at ØF4EBH) Ø2H bytes Type: x
- Points to the area in which is loaded by BDOS with the currently specified record number.
- See 3.2, "BDOS Operations."

BUFFA (at ØF4EDH) Ø2H bytes Type: x
- Loaded by BDOS with the address of the DMA buffer used for a directory access.
- See 3.2, "BDOS Operations."

DPBADDR (at ØF4EFH) Ø2H bytes Type: x
- Loaded by BDOS with the current disk parameter block starting address.
- See 3.2, "BDOS Operations."

CHECKA (at ØF4F1H) Ø2H bytes Type: x
- Loaded by BDOS with the starting address of the area used when making software check on the directory of the current disk.
- See 3.2, "BDOS Operations."

ALLOCA (at ØF4F3H) Ø2H bytes Type: x
- Loaded by BDOS with the current disk allocation area starting address.
- See 3.2, "BDOS Operations."

SECTPT (at ØF4F5H) Ø2H bytes Type: x
- Loaded by BDOS with the number of sectors per track on the current disk.
- See 3.2, "BDOS Operations."

BLKSHF (at ØF4F7H) Ø1H byte Type: x
- Loaded by BDOS with the block shift factor value of the current disk.
- See 3.2, "BDOS Operations."

BLKMSK (at ØF4F8H) Ø1H byte Type: x
- Loaded by BDOS with the block mask value of the current disk.
- See 3.2, "BDOS Operations."

EXTMSK (at ØF4F9H) Ø1H byte Type: x
- Loaded by BDOS with the extent mask value of the current disk.
- See 3.2, "BDOS Operations."

MAXALL (at ØF4FAH) Ø2H bytes Type: x
- Loaded by BDOS with the maximum allocation number of the current disk.
- See 3.2, "BDOS Operations."

DIRMAX (at ØF4FCH) Ø2H bytes Type: x
- Loaded by BDOS with the maximum number of directory entries for the current disk.
- See 3.2, "BDOS Operations."

DIRBLK (at ØF4FEH) Ø2H bytes Type: x
- Loaded by BDOS with the bit map of the block used as the current disk directory.
- See 3.2, "BDOS Operations."

CHKSIZ (at ØF5ØØH) Ø2H bytes Type: x
- Loaded by BDOS with the current disk check vector size.
- See 3.2, "BDOS Operations."

OFFSET (at ØF5Ø2H) Ø2H bytes Type: x
- Loaded by BDOS with the current disk track offset value.
- See 3.2, "BDOS Operations."

TRANV (at ØF5Ø4H) Ø2H bytes Type: x
- Loaded by BDOS with the current disk sector conversion table starting address.
- See 3.2, "BDOS Operations."

FCB$COPIED (at ØF5Ø6H) Ø1H byte

RMF (at ØF5Ø7H) Ø1H byte Type: x
- This flag is used by BDOS to indicate the read mode.
- See 3.2, "BDOS Operations."

DIRLOC (at ØF5Ø8H) Ø1H byte Type: x
- This flag is used by BDOS to indicate whether or not rename processing has been executed.
- See 3.2, "BDOS Operations."

SEQIO (at ØF5Ø9H) Ø1H byte Type: x
- This flag is used by BDOS to indicate whether or not sequential read/write processing has been executed.
- See 3.2, "BDOS Operations."

LINFO (at ØF5ØAH) Ø1H byte Type: x
- Stores the value of the E register when a BDOS function is used.
- See 3.2, "BDOS Operations."

DMINX (at ØF5ØBH) Ø1H byte Type: x
- Used by BDOS to store the disk map position which is computed during a disk write.
- See 3.2, "BDOS Operations."

SEAR$CHL (at ØF5ØCH) Ø1H byte Type: x
- Loaded by BDOS with the number of searches to be performed.
- See 3.2, "BDOS Operations."

SERA$CHA (at ØF5ØDH) Ø2H bytes Type: x
- Loaded by BDOS with the address containing the search key.
- See 3.2, "BDOS Operations."

TINFO (at ØF5ØFH) Ø2H bytes

SINGLE (at ØF511H) Ø1H byte Type: x
- This flag is used by BDOS to indicate whether or not the maximum allocation vector (MAXALL) is 1-byte data.
- See 3.2, "BDOS Operations."

RESEL (at ØF512H) Ø1H byte Type: x
- This flag is used by BDOS to indicate whether or not the current disk must be restored into its original state.
- See 3.2, "BDOS Operations."

OLDDSK (at ØF513H) Ø1H byte Type: x
- Used by BDOS to save the currently set current disk state so that the current disk can be restored into its original state.
- See 3.2, "BDOS Operations."

FCBDSK (at ØF514H) Ø1H byte Type: x
- Used by BDOS to store the disk drive code in the FCB.
- See 3.2, "BDOS Operations."

RCOUNT (at ØF515H) Ø1H byte Type: x
- Used by BDOS to store the number of records in the FCB.
- See 3.2, "BDOS Operations."

EXTVAL (at ØF516H) Ø1H byte Type: x
- Used by BDOS to store the value of AND operation executed between the FCB extent number and the extent mask (EXTMSK).
- See 3.2, "BDOS Operations."

VRECORD (at ØF517H) Ø2H bytes Type: x
- Loaded by BDOS with the virtual record position currently being accessed.
- See 3.2, "BDOS Operations."

ARECORD (at ØF519H) Ø2H bytes Type: x
- Loaded by BDOS with the actual record position currently being accessed.
- See 3.2, "BDOS Operations."

ARECl (at ØF51BH) Ø2H bytes Type: x
- Loaded by BDOS with the actual block position currently being accessed.
- See 3.2, "BDOS Operations."

DPTR (at ØF51DH) Ø1H byte Type: x
- This pointer is used by BDOS to update the specified directory position.
- See 3.2, "BDOS Operations."

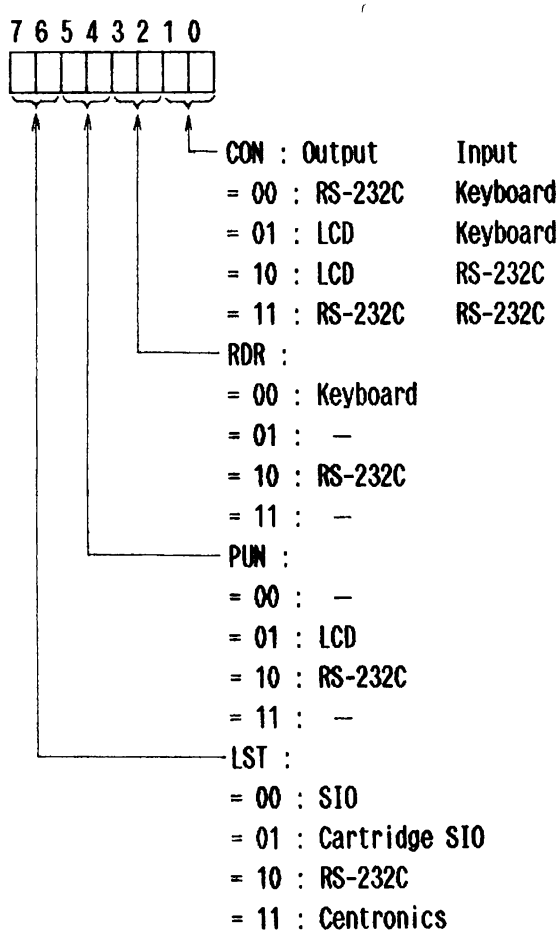
DCNT (at ØF51EH) Ø2H bytes Type: x
- This count is used by BDOS to indicate the file position in the directory.
- See 3.2, "BDOS Operations."

DREC (at ØF52ØH) Ø2H bytes

YOLDDSK (at ØF522H) Ø1H byte Type: x
- Used by BDOS to save the current disk state when a new disk is selected.
- See 3.2, "BDOS Operations."

USRSBD (at ØF523H) Ø2H bytes Type: x
- Used to save the user stack pointer during resident BDOS (RBDOS) processing.
- See 3.2, "BDOS Operations."

RIOBYTE (at ØF529H) Ø1H byte Type: #
- Used to store the current IOBYTE (the contents of address ØØØ3H) when a resident BDOS or BIOS (RBDOS or RBIOS) function is called.



- See 3.9, "I/O Byte."

ERRFLG (at ØF52AH) Ø1H byte Type: #
- This flag indicates whether or not SETERR has been executed during BDOS error recovery processing.
    = ØØH: Normal error recovery processing (message output)
    = ØFFH: Special error recovery processing (return code)
- See 3.2, "BDOS Operations."

BIOSERROR (at ØF52BH) Ø1H byte Type: #
- Loaded with the return information at the end of a BIOS disk access.
    = ØØH: Normal termination
    = Ø1H: Read error
    = Ø2H: Write error
    = Ø3H: Write protect error
    = Ø4H: Time-out error
    = Ø5H: Seek error
    = Ø6H: Break error
    = Ø7H: Power off error
    = Ø8H: Mount error
    = ØFEH: Other errors
- See 3.2.4, "BDOS Errors."

OLDBNK (at ØF52CH) Ø1H byte Type: #
- Used to store bank information when a resident BIOS (RBIOS) function is called.
    = ØFFH: System bank (OS ROM)
    = ØØH: Bank Ø (RAM)
    = Ø1H: Bank 1 (ROM capsule 1)
    = Ø2H: Bank 2 (ROM capsule 2)
- See 4.4, "Bank Switching."

LSORBNK (at ØF52DH) Ø1H byte Type: x
- Used to store bank information of the destination to which data is transferred by a LDIRXX call.

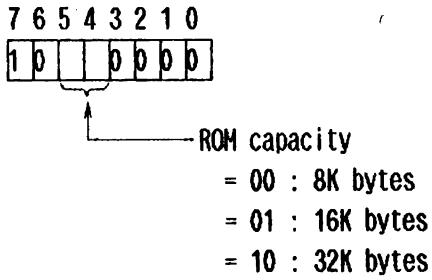
DISBNK (at ØF52EH) Ø1H byte Type: o
- Used to specify the bank when calling BIOS CALLX or JUMPX.
    = ØFFH: System bank (OS ROM)
    = ØØH: Bank Ø (RAM)
    = Ø1H: Bank 1 (ROM capsule 1)
    = Ø2H: Bank 2 (ROM capsule 2)
- See (29) CALLX in 3.4.2.

CALBNK (at ØF52FH) Ø1H byte Type: x
- Used to save bank information before bank switching when calling CALLXX.

BNKRGS (at ØF53ØH) Ø1H byte Type: #
- Bank data used in bank switch processing to switch the active bank to system bank.
- This value is usually fixed at ØØH.
- See 4.4, "Bank Switching."

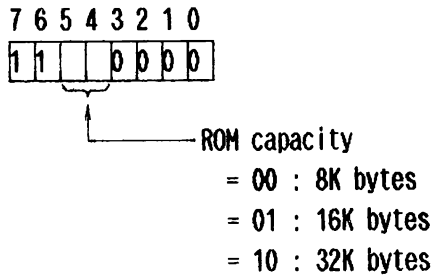
BNKRG1 (at ØF531H) Ø1H byte Type: #
- Bank data used in bank switch processing to switch the active bank to bank Ø.
- This value is usually fixed at 4ØH.
- See 4.4, "Bank Switching."

BNKRG2 (at ØF532H) Ø1H byte Type: #
- Bank data used in bank switch processing to switch the active bank to bank 1.



- When no ROM is installed, 8K bytes of ROM is assumed.
- See 4.4, "Bank Switching."

BNKRG3 (at ØF533H) Ø1H byte Type: #
- Bank data used in bank switch processing to switch the active bank to bank 2.



- When no ROM is installed, 8K bytes of ROM is assumed.
- See 4.4, "Bank Switching."

CURBNK (at ØF534H) Ø1H byte Type: #
- Loaded with information concerning the bank on which the system is currently operating.
  = ØFFH: System bank (OS ROM)
  = ØØH: Bank Ø (RAM)
  = Ø1H: Bank 1 (ROM capsule 1)
  = Ø2H: Bank 2 (ROM capsule 2)
- See 4.4, "Bank Switching."

USRSBI (at ØF535H) Ø2H bytes Type: x
- Used to save the user stack pointer during resident BIOS (RBIOS) processing.
- See 3.3, "BIOS Operations."

BIOSFN (at ØF537H) Ø2H bytes Type: x
- Stores the logical number of the user-called BIOS function during resident BIOS (RBIOS) processing.
- See 3.3, "BIOS Operations."

ROMEXQON (at ØF539H) Ø1H byte Type: x
- This flag is used to determine whether or not ROM-based processing is to be executed.
  = ØØH: ROM-based processing must not be executed.
  = Ø1H: ROM-based processing must be executed.
- See 4.6, "Executing a ROM Program."

ROMEXBNK (at ØF53AH) Ø1H byte Type: x
- Loaded with bank information when transferring control to a ROM-based program.
- See 4.6, "Executing a ROM Program."

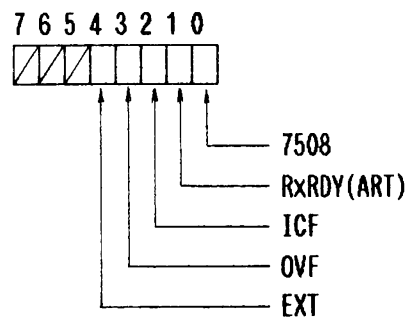
ROMEXADD (at ØF53BH) Ø2H bytes Type: x
- Loaded with the processing starting address when transferring control to a ROM-based program.
- See 4.6, "Executing a ROM Program."

RZBANKR (at ØF53DH) Ø1H byte Type: o
- Stores the bank register (PØ5H) output status.



- The bit assignments for this area are identical to those for BANKR (PØ5H).
- See 4.4, "Bank Switching."

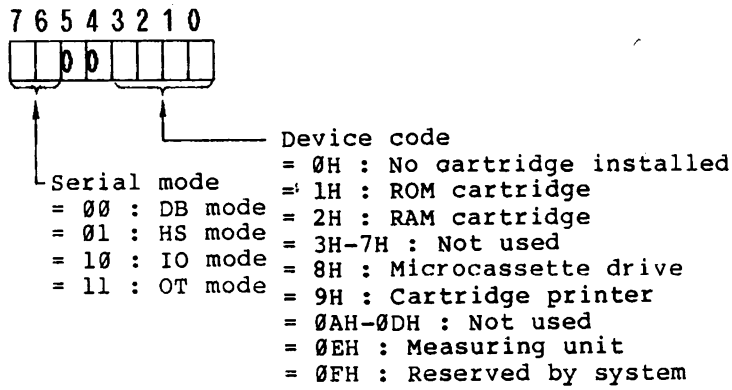
RZIER (at ØF53EH) Ø1H byte Type: o
- Stores the interrupt enable register (PØ4H) output status.



(Interrupt enabled with each bit set to 1)

- The bit assignments for this area are identical to those for IER (PØ4H).
- See 4.7, "Interrupts."

CRGDEV (at ØF53FH) Ø1H byte Type: #
- Loaded with the device code of the installed cartridge.



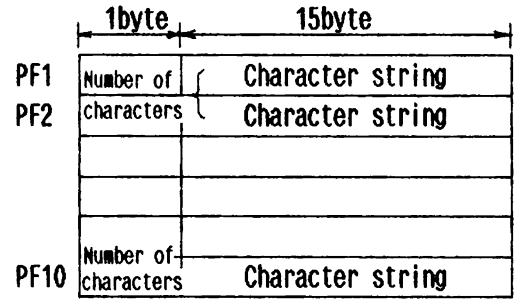
- See 5.1, "Cartridges."

SAVEIX (at ØF54ØH) Ø2H bytes Type: x
- Used to save the contents of the IX register when a BIOS function is called.
- See 3.3.3, "BIOS Hook."

SAVEIY (at ØF542H) Ø2H bytes Type: x
- Used to save the contents of the IY register when a BIOS function is called.
- See 3.3.3, "BIOS Hook."

IMSTP (at ØF544H) Ø1H byte Type: x
- Used in keyboard interrupt processing to store the input key code.
- See 3.5, "Keyboard."

WPFKTBL (at ØF545H) AØH bytes Type: o
- PF key definition table.



- The number of characters is specified within the range of ØØH to ØFH. ØØH indicates that the PF key is not defined.
- The 16Ø data bytes starting at PFKTAB (ØFØ2DH) are copied into this area at warm boot time.
- See 3.5.3, "Standard Keyboard."

SEKDSK (at ØF5E5H) Ø1H byte Type: x
- Loaded by BIOS SELDSK with the physical disk number.
- See 3.4, "BIOS Details."

SEKTRK (at ØF5E6H) Ø2H bytes Type: x
- Loaded by BIOS SETTRK with the specified track number.
- See 3.4, "BIOS Details."

SEKSEC (at ØF5E8H) Ø1H byte Type: x
- Loaded by BIOS SETSEC with the specified sector number.
- See 3.4, "BIOS Details."

HSTDSK (at ØF5E9H) Ø1H byte
HSTTRK (at ØF5EAH) Ø2H bytes
HSTSEC (at ØF5ECH) Ø1H byte
SEKHST (at ØF5EDH) Ø1H byte
HSTACT (at ØF5EEH) Ø1H byte
HSTWRT (at ØF5EFH) Ø1H byte
UNACNT (at ØF5FØH) Ø1H byte
UNADSK (at ØF5F1H) Ø1H byte
UNATRK (at ØF5F2H) Ø2H bytes
UNASEC (at ØF5F4H) Ø1H byte
ERFLAG (at ØF5F5H) Ø1H byte
RSFLAG (at ØF5F6H) Ø1H byte
READOP (at ØF5F7H) Ø1H byte
WRTYP (at ØF5F8H) Ø1H byte

DMAADR (at F5F9H) Ø2H bytes Type: #
- DMA address set at time of disk Read/Write.
- Normally points to SYSDMA (F8B1H).
- See 3.8, "Disk".

DIRBUF (at ØF5FBH) 8ØH bytes Type: #
- DMA buffer for directory access.
- This area is pointed to by the directory buffer pointer in the disk parameter header (F15ØH -)
- See 3.8, "Disk".

ALVO (at ØF67BH) 2ØH bytes Type: x
- Allocation area for RAM disk use.
- See 3.8, "Disk"

CSVO (at ØF69BH) ØØH bytes
- Checksum area for RAM disk use.
  (Label name only secured)
- See 3.8, "Disk".

ALV1 (at ØF69BH) Ø9H bytes type: x
- Allocation area for ROM capsule 1 use.
- See 3.8, "Disk"

CSV1 (at ØF6A4H) ØØH bytes
- Checksum area for ROM capsule 1 use.
  (Label name only secured)
- See 3.8, "Disk".

ALV2 (at ØF6A4H) Ø9H bytes Type: x
- Allocation area for ROM capsule 2 use.
- See 3.8, "Disk".

CSV2 (at ØF6ADH) ØØH bytes
- Checksum area for ROM capsule 2 use.
  (Label name only secured)
- See 3.8, "Disk".

ALV3 (at ØF6ADH) 12H bytes Type: x
- Allocation area for disk drive 1 use
- See 3.8, "Disk".

CSV3 (at ØF6BFH) 1ØH bytes Type: x
- Checksum area for disk drive 1 use.
- See 3.8, "Disk".

ALV4 (at ØF6CFH) 12H bytes Type: x
- Allocation area for disk drive 2 use
- See 3.8, "Disk".

CSV4 (at ØF6E1H) 1ØH bytes Type: x
- Checksum area for disk drive 2 use.
- See 3.8, "Disk".

ALV5 (at ØF6F1H) 12H bytes type: x
- Allocation area for disk drive 3 use
- See 3.8, "Disk".

CSV5 (at ØF7Ø3H) 1ØH bytes Type: x
- Checksum area for disk drive 3 use.
- See 3.8, "Disk".

ALV6 (at ØF713H) 12H bytes Type: x
- Allocation area for disk drive 4 use
- See 3.8, "Disk".

CSV6 (at ØF725H) 1ØH bytes Type: x
- Checksum area for disk drive 4 use.
- See 3.8, "Disk".

ALV7 (at ØF735H) Ø8H bytes Type: x
- Allocation area for microcassette use.
- See 3.8, "Disk".

CSV7 (at ØF73DH) 1ØH bytes Type: x
Checksum area for microcassette use.
- See 3.8, "Disk".

ALV8 (at ØF74DH) Ø8H bytes type: x
Allocation area for RAM cartridge use.
- See 3.8, "Disk".

CSV8 (at ØF755H) Ø4H bytes Type: x
Checksum area for RAM cartridge use.
- See 3.8, "Disk".

ALV9 (at ØF759H) Ø8H bytes Type: x
Allocation area for ROM cartridge 1 use.
- See 3.8, "Disk".

CSV9 (at ØF761H) ØØH bytes
Checksum area for ROM cartridge 1 use. (Label name only secured.)
- See 3.8, "Disk".

ALV1Ø (at ØF761H) Ø8H bytes Type: x
Allocation area for ROM cartridge 2 use.
- See 3.8, "Disk".

CSV1Ø (at ØF769H) ØØH bytes
Checksum area for ROM cartridge 2 use.
- See 3.8, "Disk".

HCX (at ØF769H) Ø1H byte Type: x
- During execution of screen dump, this location is the work area which stores the X coordinate for the VRAM being dumped.
- See 3.4.2 (19) SCRN DUMP

HCY (at ØF76AH) Ø1H byte
- During execution of screen dump, this location is the work area which stores the Y coordinate for the VRAM being dumped.
- See 3.4.2 (19) SCRN DUMP

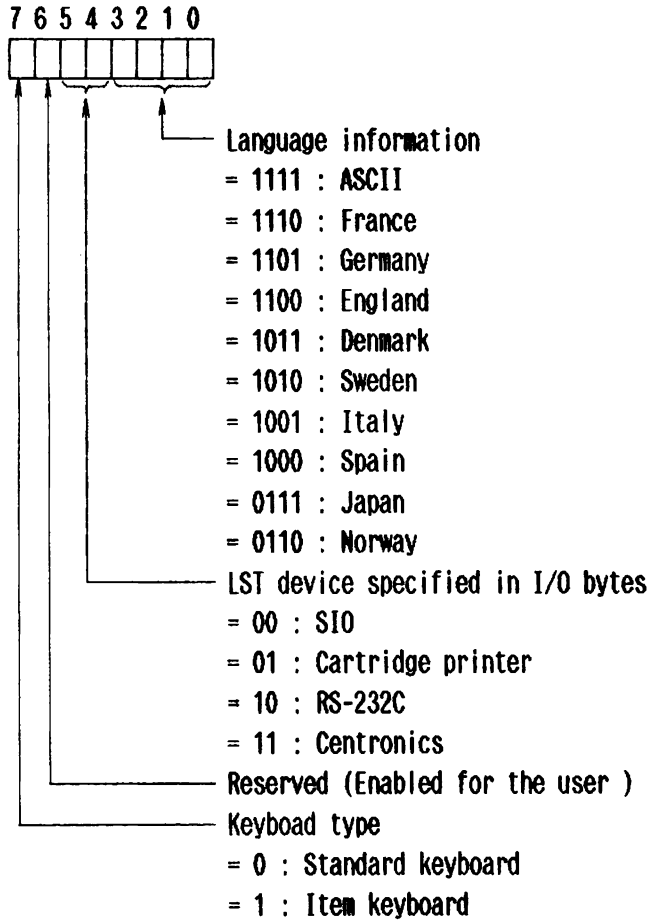
HCN (at ØF76BH) Ø1H byte

HCDATA (at ØF76CH) Ø8H bytes
- During execution of screen dump, these locations are the work area used for editing the bit image data output to the printer.
- See 3.4.2 (19) SCRN DUMP

LSTERR (at ØF773H) Ø1H byte Type: #
- Indicates whether or not screen dump processing has been interrupted by the CTRL/STOP keys.
  = ØØH: Normal termination.
  = ØFFH: Processing interrupted (CTRL/STOP is pressed or power is turned off).
- See (19) SCRNDUMP in 3.4.2.

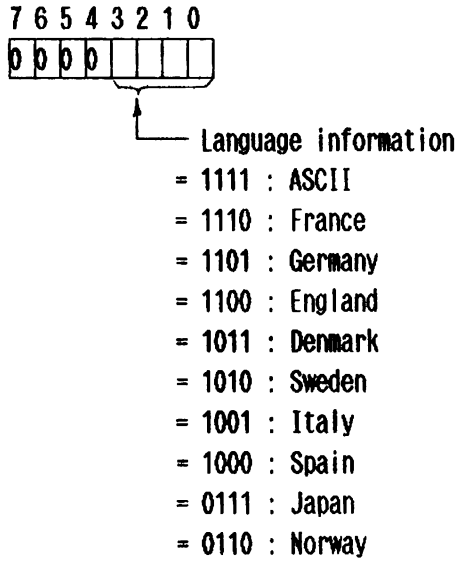
PRINTER (at ØF774H) Ø1H byte

YKCOUNTRY (at ØF775H) Ø1H byte Type: #
- Used to store the DIP switch state which is read at boot time.



- See 7.2, "DIP Switches."
- In OS Kana V1.Ø, ASCII, England and Japan are valid. The LST: (Ø1) device is assigned to LCD.
- In OS Kana V2.Ø, only Japan is valid.
- In ASCII version OS 1.Ø, all countries except Japan are valid.

YLDFLTC (at ØF776H) Ø1H byte Type: o
- Loaded with the default value for LCD screen display.



- This area is loaded with the language information which is specified with the DIP switches at boot time and copied into YLCOUNTRY (ØF777H) at WBOOT time.
- See 7.2, "DIP Switches."

YLCOUNTRY (at ØF777H) Ø1H byte Type: o
- This flag specifies the language of the characters to be displayed on the LCD screen.
- The value of YLDFLTC (ØF776H) is copied into this area at warm boot time.
- See 3.6, "LCD Display (CONOUT Details)."

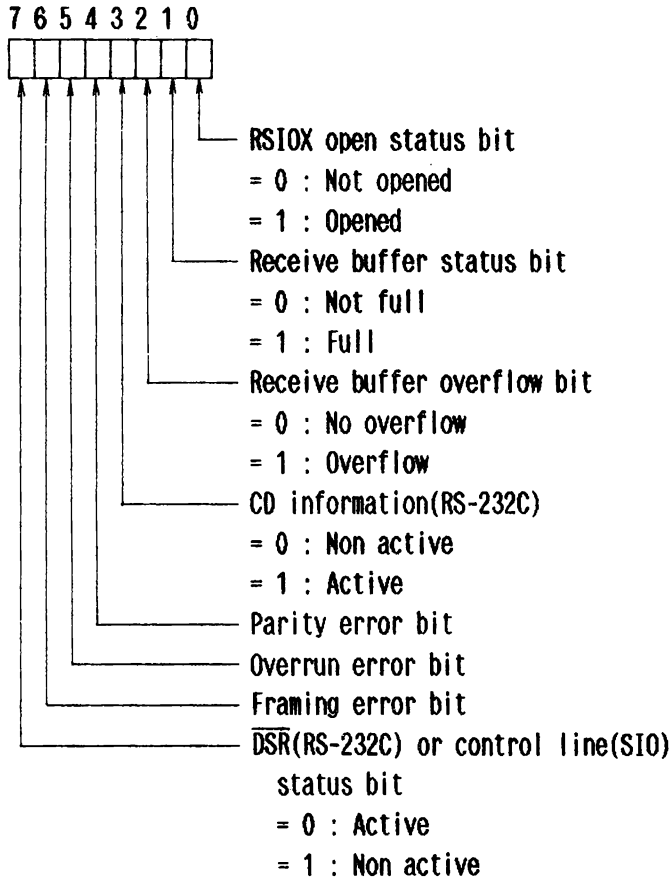
RAMSIZE (at ØF778H) Ø2H bytes

YSIZERAM (at ØF77AH) Ø1H byte Type: #
- Indicates the RAM disk size in K bytes.
- See 3.8, "Disk Storage."

TIMEND (at ØF77CH) Ø2H bytes Type: x
- Used to store the auto power off time.
- See 2.8, "Auto Power-off."

SRBADR (at ØF77EH) Ø9H bytes Type: x
- Used as the RSIOX parameter area for BIOS PUNCH or READER.
- When BIOS PUNCH or READER is first called, RSIOX copies the 9-byte data from SRSADR (ØEF31H) into this area and executes open processing using this area. Subsequently, this area is used as the RSIOX parameter.
- See 3.4, "BIOS Details."

RSPSTS (at ØF787H) Ø1H byte Type: #
- Indicates the BIOS RSIOX processing status.



(Error bits (bits 6 to 4) are set to 1 when the corresponding error occurs.)

- Bits 7 and 3 are set when RSIOX is executed and the other bits are set when an interrupt occurs.

RSPRBGP (at ØF788H) Ø2H bytes Type: #
- Pointer to the receive buffer area from which RSIOX gets data.
- See (23) RSIOX in 3.4.2.

RSPRBPP (at ØF78AH) Ø2H bytes Type: #
- Pointer to the receive buffer area to which RSIOX puts data.
- See (23) RSIOX in 3.4.2.

RSPRBAD (at ØF78CH) Ø2H bytes Type: #
- Indicates the RSIOX receive buffer starting address (parameter specified at open time).
- See (23) RSIOX in 3.4.2.

RSPRBSZ (at ØF78EH) Ø2H bytes Type: #
- Indicates the RSIOX receive buffer size (parameter specified at open time).
- See (23) RSIOX in 3.4.2.

RSPBITR (at ØF79ØH) Ø1H byte Type: #
- Indicates the transmission speed (parameter specified at open time).

Value	Speed (BPS)	Value	Speed(BPS)
Ø2H	11Ø	ØDH	48ØØ
Ø4H	15Ø	ØEH	96ØØ
Ø5H	2ØØ	ØFH	192ØØ
Ø6H	3ØØ	1ØH	384ØØ
Ø8H	6ØØ	8ØH	75/12ØØ
ØAH	12ØØ	81H	12ØØ/75
ØCH	24ØØ		

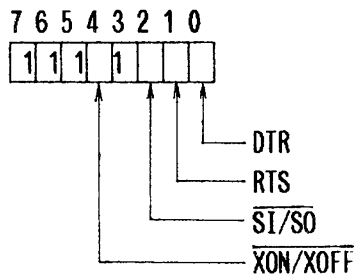
- See (23) RSIOX in 3.4.2.

RSPBITL (at ØF791H) Ø1H byte Type: #
- Indicates the bit length specified by RSIOX (parameter specified at open time).
  = Ø2H: 7 bits/character
  = Ø3H: 8 bits/character
- See (23) RSIOX in 3.4.2.

RSPPAR (at ØF792H) Ø1H byte Type: #
- Indicates the parity mode specified by RSIOX (parameter specified at open time).
  = ØØH: No parity
  = Ø1H: Odd parity
  = Ø3H: Even parity
- See (23) RSIOX in 3.4.2.

RSPSTOPB (at ØF793H) Ø1H byte Type: #
- Contains the number of stop bits specified by RSIOX (parameter specified at open time).
  = Ø1H: 1 stop bit
  = Ø3H: 2 stop bits
- See (23) RSIOX in 3.4.2.

RSPSPP (at ØF794H) Ø1H byte Type: #
- Contains the special parameter specified by RSIOX (specified at open time) status.



(DTR and RTS controls are enabled when the corresponding flag bit is set, and SI/SO and XON/XOFF controls are enabled when reset.)
- See (23) RSIOX in 3.4.2.

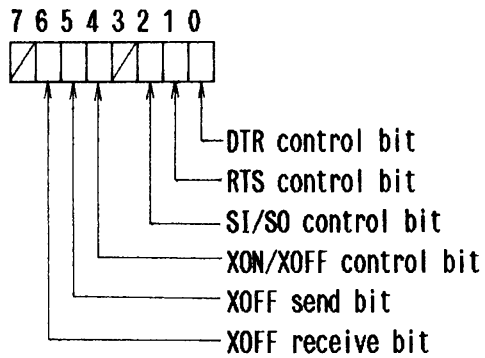
RSRBEAD (at ØF795H) Ø2H bytes Type: #
- Indicates the lower-limit address of the receive buffer used by RSIOX.
    RSRBEAD = RSPRBAD (buffer address) + RSPRBSZ (buffer size)
- See (23) RSIOX in 3.4.2.

RSXONSZ (at ØF797H) Ø2H bytes Type: #
- Indicates the number of receive data bytes when an XON code is to be sent during RSIOX XON/XOFF processing.
    RSXONSZ = RSPRBSZ (buffer size) / 4
- See (23) RSIOX in 3.4.2.

RSXOFSZ (at ØF799H) Ø2H bytes Type: #
- Indicates the number of receive data bytes when an XOFF code is to sent during RSIOX XON/XOFF processing.
    RSXOFSZ = RSPRBSZ (buffer size) - RSXONSZ (XON size)
- See (23) RSIOX in 3.4.2.

CHRMSK (at ØF79BH) Ø1H byte Type: #
- Used by RSIOX to specify the receive data mask pattern.
    = 7FH: 7 bits/character data
    = ØFFH: 8 bits/character data
- See (23) RSIOX in 3.4.2.

RSSSPP (at ØF79CH) Ø1H byte Type: #
- Used by RSIOX to indicate the transmit/receive control status.



(Each bit is set to 1 when the corresponding control is enabled.)

- See (23) RSIOX in 3.4.2.

RSPAKAD (at ØF79DH) Ø2H bytes Type: #
- Loaded by RSIOX with contents the HL register.
- See (23) RSIOX in 3.4.2.

RSRDL (at ØF79FH) Ø2H bytes Type: #
- Indicates the number of data bytes stored in the receive buffer by RSIOX.
- See (23) RSIOX in 3.4.2.

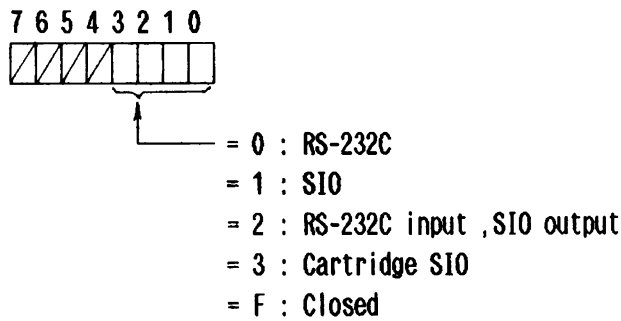
SISOCNT (at ØF7A1H) Ø2H bytes Type: #
- Indicates the number of SI and SO receive codes stored in the receive buffer when SI/SO processing is specified with RSIOX.
- See (23) RSIOX in 3.4.2.

SSXMODE (at ØF7A3H) Ø1H byte Type: #
- Indicates the SI/SO send state when SI/SO processing is specified with M.
  = ØØH: SI send state
  = 1ØH: SO send state
- See (23) RSIOX in 3.4.2.

RSXMODE (at ØF7A4H) Ø1H byte Type: #
- Indicates the SI/SO receive state when SI/SO processing is specified with RSIOX.
  = ØØH: SI receive state
  = 8ØH: SO receive state
- See (23) RSIOX in 3.4.2.

RSFDEV (at ØF7A5H) Ø1H byte Type: x
- Used to store the function code (bits 3-Ø in the B register) of RSIOX.
- See (23) RSIOX in 3.4.2.

RSODEV (at ØF7A6H) Ø1H byte Type: #
- Loaded by RSIOX with the device status (open status).



- See (23) RSIOX in 3.4.2.

RSCALSW (at ØF7A7H) Ø1H byte

RSBYTE (at ØF7A8H) Ø1H byte

EPWKTP (at ØF7A9H) Ø5H bytes Type: x
- Used to store FMT, DID, SID, FNC and SIZ when sending EPSP data.
- See 5.3, "Terminal Floppy (TF)."

EPACKC (at ØF7AEH) Ø1H byte Type: x
- Used to store the ACK or NAK to be sent to the terminal floppy when sending/receiving EPSP data.
- See 5.3, "Terminal Floppy (TF)."

EPBLCN (at ØF7AFH) Ø1H byte

EPERMD (at ØF7BØH) Ø1H byte

REGA (at ØF7B1H) Ø1H byte Type: x
- The work area used to specify the read/write mode when accessing the ROM cartridge, RAM cartridge, or external RAM disk.

RØ (at ØF7B2H) 12H bytes Type: x
- A work area used by the system when sending/receiving EPSP data.

OLDDVMD (at ØF7C4H) Ø1H byte Type: x
- Used to store the device code of the cartridge set at power off time.

BEEPBASE (at ØF7C5H) Ø1H byte Type: x
- The work area used during BIOS BEEP processing to measure the beep time.

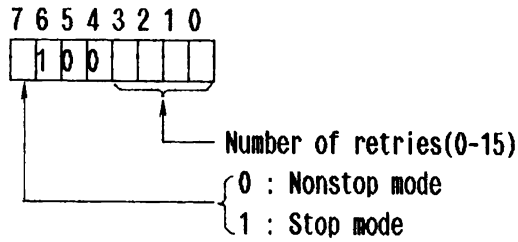
LCHRFONT (at ØF7C6H) Ø8H bytes Type: x
- The work area used during BIOS CONOUT processing to get font data from the character generator.
- See 3.6, "LCD Display (CONOUT Details)."

TOSRCD (at ØF7CEH) Ø1H byte Type: #
- Loaded with a MTOS return code.
  = ØØH: Normal termination
  = Ø1H: Not removed
  = Ø2H: Not mounted
  = Ø3H: Cannot remove
  = Ø4H: Cannot mount
  = Ø5H: Directory full
  = Ø6H: R/O file
  = Ø7H: File already exists
  = Ø8H: File not found
  = Ø9H: File already open
  = ØAH: File not open
  = ØBH: Record number error
  = ØCH: EOF reached
  = ØDH: Random access error
  = ØEH: Not used
  = ØFH: R/O drive error
  = 1ØH: Verify error
  = 11H: File number error
  = 12H: Auto remove check error
- See 3.7.5, "MTOS Functions."

IOSRCD (ØF7CFH) Ø1H byte Type: #
- Loaded with the MIOS return code.
    = ØØH: Normal termination
    = Ø1H: Head error
    = Ø2H: Motor stop error
    = Ø3H: Write protect error
    = Ø4H: Data error
    = Ø5H: CRC error
    = Ø6H: Block mode error
    = Ø7H: Tape error
    = Ø8H: Mount error
- See 3.7.6, "Using MIOS."

CURMCT (ØF7DØH) Ø1H byte Type: x
- Loaded with the MCT drive code when MCT is selected as the current drive.
- See 3.7, "MTOS/MIOS Operations."

TACATR (ØF7D1H) Ø1H byte Type: o
- Specifies the stop/nonstop mode and the number of retries during tape accessing.



- The value of DFTATR (ØF2E4H) is loaded into this area at warm boot time.
- See 3.7.4, "Using MTOS."

TACMOD (ØF7D2H) Ø1H byte Type: #
- Loaded with tape access attributes.
- TACMOD is initially set to one of the following values:
1) During file creation:
   Unconditionally set to the value of TACATR (ØF7D1H).
2) When TOSCTL (ØEFBBH), bit 5 is Ø:
   Set to the value of file attribute 1 in the RAM directory.
3) When TOSCTL (ØEFBBH), bit 5 is 1:
   Set to the value of TACATR (ØF7D1H).
- See 3.7.4, "Using MTOS."

BLKNO (ØF7D3H) Ø2H bytes Type: #
- Loaded with the current block number during MTOS processing.
- See 3.7, "MTOS/MIOS Operations."

RECNO (ØF7D5H) Ø2H bytes Type: #
- Loaded with the current record number during MTOS processing.
- See 3.7, "MTOS/MIOS Operations."

BLKREC (ØF7D7H) Ø2H bytes Type: #
- Loaded with the first record number in the TOSDMA buffer during MTOS processing.
- See 3.7, "MTOS/MIOS Operations."

RECEXT (ØF7D9H) Ø1H byte Type: #
- This flag indicates whether or not records exist in the TOSDMA buffer during MTOS processing.
    = ØØH: No record present.
    = Ø1H: First record present.
    = Ø2H: Second record present.
    = Ø3H: Two records present.
- See 3.7, "MTOS/MIOS Operations."

ORECEXT (ØF7DAH) Ø1H byte Type: x
- This flag indicate whether or not old records exist in the TOSDMA buffer.
- See 3.7, "MTOS/MIOS Operations."

DIRPNT (ØF7DBH) Ø1H byte Type: #
- Loaded with the directory file counter value of the file currently being processed by MTOS.
- See 3.7, "MTOS/MIOS Operations."

FILPNT (ØF7DCH) Ø1H byte Type: #
- Loaded with the pointer value, in the directory file, of the file that is currently being processed by MTOS.
- See 3.7, "MTOS/MIOS Operations."

SRCADD (ØF7DDH) Ø2H bytes Type: x
- Loaded with the T-FCB starting address during MTOS file search.
- See 3.7, "MTOS/MIOS Operations."

MAXBLK (ØF7DFH) Ø2H bytes Type: #
- Loaded with the maximum block number in the file that is currently being processed by MTOS.
- See 3.7, "MTOS/MIOS Operations."

MAXREC (ØF7E1H) Ø2H bytes Type: #
- Loaded with the maximum record number in the file that is currently being processed by MTOS.
- See 3.7, "MTOS/MIOS Operations."

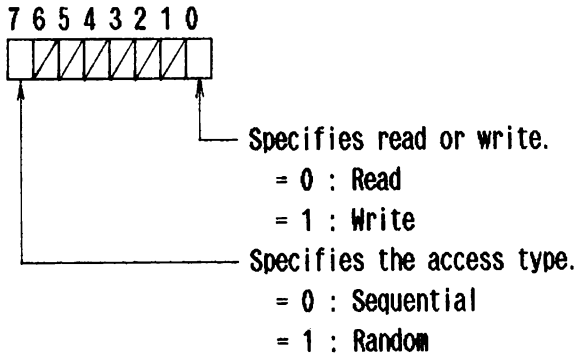
REMFLG (ØF7E3H) Ø1H byte Type: #
- This flag indicates whether or not an actual write to the tape is necessary during MTOS remove processing.
    = ØØH: Write necessary.
    = Nonzero: Write not necessary (directory is updated).
- See 3.7, "MTOS/MIOS Operations."

SKCKCT (ØF7E4H) Ø1H byte Type: x
- Loaded with the number of rewrites to be made when an error occurs during MTOS seek processing.
- See 3.7, "MTOS/MIOS Operations."

LSTCNT (ØF7E5H) Ø2H bytes Type: #
- Loaded with the last tape count during MCT processing by MTOS.
- See 3.7, "MTOS/MIOS Operations."

USEKCNT (ØF7E7H) Ø2H bytes Type: o
- Loaded with the seek counter value specified by the user using MTOS.
- IDFLG (ØFA13H), bit 6 determines whether the last free area counter value is to be obtained from the directory or USEKCNT.
     Bit 6 = Ø: Directory
           = 1: USEKCNT
- See 3.7, "MTOS/MIOS Operations."

RWMOD (ØF7E9H) Ø1H byte Type: x
- Loaded with the tape access mode (read/write) during MTOS processing.



- See 3.7, "MTOS/MIOS Operations."

ALCVEC (ØF7EAH) Ø8H bytes Type: #
- Loaded with the MTOS allocation vector.
- See 3.7, "MTOS/MIOS Operations."

REGBC (ØF7F2H) Ø2H bytes Type: x
- The work area for saving the BC register during MTOS processing.
- See 3.7, "MTOS/MIOS Operations."

REGDE (ØF7F4H) Ø2H bytes Type: x
- The work area for saving the DE register during MTOS processing.
- See 3.7, "MTOS/MIOS Operations."

BSTCNT (ØF7F6H) Ø2H bytes Type: x
- Loaded with the block starting tape count during a MIOS block read.
- See 3.7, "MTOS/MIOS Operations."
RNDNO (ØF7F8H) Ø2H bytes

RNDCNT (ØF7FAH) Ø2H bytes

MCTSVDAT (ØF7FCH) 20H bytes Type: x
- The work area for saving system data on the MCT cartridge in RAM at power-off time.
- See 3.7, "MTOS/MIOS Operations."

YIDVAL (ØF81DH) Ø1H byte Type: x

- Loaded with the offset value, in the index table, of the key that is used to change the international character set specification.
- See 3.5, "Keyboard."

(Reserved) (ØF81EH) 42H

6.2.6 System Area V (RSYSAR5)

Buffers and stack areas that do not need to be initialized are reserved in this area.

KBUF (ØF860H) 21H bytes Type: x
- The buffer for storing a key entry made through a 7508 keyboard interrupt.
- See 3.5, "Keyboard."

KBUFEND (ØF881H) Ø1H byte Type: x
- Loaded with the KBUF last position.
- See 3.5, "Keyboard."

TIMBUF (ØF882H) ØBH bytes
SYSFCB (ØF88DH) 24H bytes Type: x
- The area into which the user-specified FCB data is to be copied during BDOS or BIOS processing.
- See 3.2, "BDOS Operations."

SYSDMA (ØF8B1H) 80H bytes Type: x
- The area into which the contents of the user-specified DMA buffer are to be copied during BDOS or BIOS processing.
- See 3.2, "BDOS Operations."

PKT_TOP (ØF931H) Ø9H bytes Type: x
- The area through which EPSP system data is sent or received during EPSP data transmission.
- See 5.3, "Terminal Floppy (TF)."

SCRCH_BUF (ØF93AH) 80H bytes Type: x
- The system scratch buffer for disk accessing.
- See 3.3, "BIOS Operations."

MCTDTSV (ØF9BAH) 38H bytes

IDTOP (Ø9E2H) 35H bytes Type: #
- The area for storing tape ID field data during MTOS processing.
- See 3.7, "MTOS/MIOS Operations."
- IDTOP consists of the following fields:

TAPNAM (Ø8H bytes)
- Tape name (ASCII code)

TAPVOL (Ø2H bytes)
- Tape volume number

TAPPAS (Ø8H bytes)
- Not used

MAKMON (Ø2H bytes)
- The month in which the tape was created (ASCII code)

MAKDAT (Ø2H bytes)
- The day on which the tape was created (ASCII code)

MAKYEA (Ø2H bytes)
- The year in which the tape was created (ASCII code)

MAKHOU (Ø2H bytes)
- The hour at which the tape was created (ASCII code)

MAKMIN (Ø2H bytes)
- The minute at which the tape was created (ASCII code)

MAKSEC (Ø2H bytes)
- The second at which the tape was created (ASCII code)

REMMON (Ø2H bytes)
- The month in which the tape was last removed (ASCII code)

REMDAT (Ø2H bytes)
- The day on which the tape was last removed (ASCII code)

REMYEA (Ø2H bytes)
- The year in which the tape was last removed (ASCII code)

REMHOU (Ø2H bytes)
- The hour at which the tape was last removed (ASCII code)

REMMIN (Ø2H bytes)
- The minute at which the tape was last removed (ASCII code)

REMSEC (Ø2H bytes)
- The second at which the tape was last removed (ASCII code)

TOTMNT (Ø2H bytes)
- Total number of mounts.

TOTBNO (Ø2H bytes)
- Total number of blocks.

TOTRNO (Ø2H bytes)
- Total number of records.

TOTFNO (Ø1H byte)
- Total number of files.

IDFLG (Ø1H byte)
- Tape ID flag

TCNTTP (Ø1H byte)
- Tape TOS controller

LSTFNO (Ø2H bytes)
- File number of the last file on the tape

DIRTOP (ØFA17H) 180H bytes Type: #
- The area for storing tape directory data (RAM directory) during MTOS processing.
- This area can contain up to 12 directories, each of which consists of 32 bytes.
See 3.7, "MTOS/MIOS Operations."

TOSBUF (ØFB97H) 1Ø4H bytes Type: x
- DMA buffer used by MTOS.
- See 3.7, "MTOS/MIOS Operations."

COMBUF (ØFC9BH) FØH bytes Type: x
- System serial data receive buffer used during BIOS PUNCH or READER execution.
- See 3.4, "BIOS Details."

(Reserved) (ØFD8BH) 15H bytes

BDOSSK (ØFF8FH) 7ØH bytes Type: x
- Stack area for BDOS
- See 3.2, "BDOS Operations."

BIOSSK (ØFF1FH) 5CH bytes Type: x
- Stack area for BIOS
- See 3.3, "BIOS Operations."

SP75Ø8 (ØFEC3H) 6CH bytes Type: x
- Stack area for 75Ø8 interrupts
- See 4.7, "Interrupts."

SP8251 (ØFE57H) 2CH bytes Type: x
- Stack area for ART interrupts
- See 4.7, "Interrupts."

SPOVF (ØFE2BH) 3ØH bytes Type: x
- Stack area for OVF interrupts
- See 4.7, "Interrupts."

SPICF (ØFDFBH) 1ØH bytes Type: x
- Stack area for ICF interrupts
- See 4.7, "Interrupts."

SPEXT (ØFDEBH) 1ØH bytes Type: x
- Stack area for EXT interrupts
- See 4.7, "Interrupts."

STRTSP (ØFDDBH) 3CH bytes Type: x
- Stack area for system address Ø start processing

STKBOT (ØFD9FH) ØØH byte

CHAPTER 7 MISCELLANEOUS TOPICS CONTENTS

7.1 Versions II-679
7.1.1 General II-679
7.1.2 Version Check II-679
7.1.3 Version Differences II-680

7.2 DIP Switches II-682
7.2.1 General II-682
7.2.2 DIP Switch Bit Difinitions II-683
7.2.3 System Processing II-683

7.3 Guide for Programming ROMs II-686

CHAPTER 7 MISCELLANEOUS TOPICS

7.1 Versions

7.1.1 General

PINE OS is available in the following three versions:

- Kana V1.Ø
- Kana V2.Ø

} (Japan only)

- Export version 1.Ø

For application programs which have to run in different modes depending on the OS version, PINE OS provides a facility which allows them to refer to the version of the OS under which they are currently running.

7.1.2 Version Check

The version can be examined using one of the following methods:

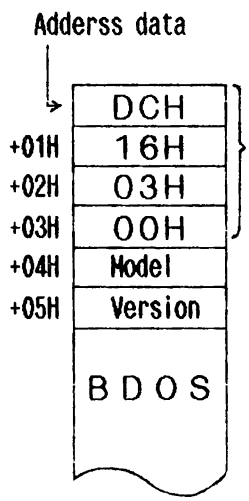
1. Checking the CP/M serial number.
2. Calling ROMID on OS ROM.

This subsection shows how to test the version using the above methods.

7.1.2.1 Checking the CP/M serial number

The model type and OS version number, and other information are stored in 6-byte CP/M BDOS fields, labeled Serial No., which are located at the beginning of RBDOS1 and RBDOS2.

To refer to Serial No. in RBDOS1, check the 6 bytes before the BDOS entry address which is stored in addresses ØØØ6H and ØØØ7H. To refer to Serial No. in RBDOS2, check the 6 bytes starting at ØEAØØH. These two fields contain the same information.



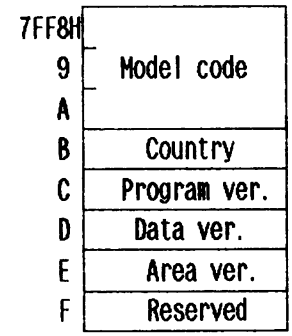
Model : ØØH : PX-8
: Ø1H : PINE

Version:
Bits 7-4 : Destination
ØØH : Export version
Ø1H : HASCI (not available for PINE)
Ø2H : ASCII (not available for PINE)
Ø3H : Kana version (Japan only)
Bits 3-Ø Version No.
Ø1H : V1.Ø
Ø2H : V2.Ø

7.1.2.2 ROMID

ROMID identifies the OS ROM installed in the PINE. It is located in an 8-byte area on OS ROM at 7FF8H. ROMID can be referenced using the LDIRX BIOS function.

The format of the ROMID field is shown below. Data in this field is all in ASCII code.



Model code : Set to 'PIN'.
Country — : 'J' : Kana version(Japan only)
'E' : Export version
'U' : USA version
(not available for PINE)
Program ver.: Identifies the OS version.
'1' : Ver.1.0
'2' : Ver.2.0
Data ver. : Identifies the version of the initial system area data.
'1' : Ver.1.0
'2' : Ver.2.0
Area ver. : Indentifies the version of the added system area.
'1' : Ver.1.0
'2' : Ver.2.0
'3' : Ver.3.0

7.1.3 Version Differences

The table on the next page summarizes the functional differences among OS Kana version V1.Ø, and V2.Ø, and Export version 1.Ø. The table below lists the OS versions and their identifications placed on the OS ROM (mask ROM).

OS version	Version ID on ROM
Kana V1.Ø (Japan only)	EPSON POSK1
Kana V2.Ø (Japan only)	EPSON POSK2
Export Version 1.Ø	EPSON POSO1

Difference	Kana V1.0	Export version 1.0	Kana V2.0	Remarks
*CRS state after cartridge installation.	Depends on the cartridge type.	Programmable by rewriting CRSTPTN(0EFC1H).	Same as in Export version 1.0.	OS cartridge check routine(CHKMOD) is made available by rewriting CRSIPTN when adding a cartridge.
Supported cartridge I/F mode	HS mode DB mode	HS mode DB mode IO mode	HS mode DB mode IO mode	IO mode is available for cartridges with device codes of 0EH and 0FH.
Emergency TF write at power-off time	No	Yes	Yes	
Provisions for modifying hook or adding hook entries	Control can be transferred to any location in the RAM area from the hook.	Control can be transferred to locations at 8000H and higher from the hook. A hook entry to BIOS is added to Kana V1.0.	Control can be trasferred to locations at 8000H and higher from the hook. A hook entry to ART-interrupt processing routine is added to Export version 1.0.	Control can be transferred to any locations in the RAM area through hook entries for ICF, OVF, and EXT interrupts.
System utility jump table.	A jump table is available for 33 utility routines.	An entry to cartridge check routine(CHKMOD) is added to the jump table for kana V1.0.	Same as in Export version 1.0.	
Power-off or alarm processing during RSIOX processing.	RSIOX is terminated before power-off or alarm processing is performed.	Power-off or alarm processing is performed during RSIOX processing(no data is lost).	Same as in Export version 1.0.	

Difference	Kana V1.0	Export version 1.0	Kana V2.0	Remarks
Alarm screen display time.	Fixed at 50 seconds.	Programmable. Defaults to 50 seconds. Display time is loaded in ALRMTIME (0F308H).	Same as in Export version 1.0.	
Alarm or wake processing after power failure.	Not inhibited.	Inhibited. Original state is restored by powering on the machine after power is restored.	Controlled by flag BTRYALM (0EFC3H). =00H : Inhibited. =Nonzero : Not inhibited.	In Export version 1.0 and kana V2.0, alarm or wake processing routine is exited if power is shut down by a power failure when it has been inhibited.
Cartridge printer support.	Not supported. CRT(01) in the LST : field is assigned to LCD.	Supported. Assingned to CRT(01) in the LST : field.	Same as in Export version 1.0.	Supported in cartridge IIS mode.
Initialization of printer version data.	After rewriting I/O byte LST : field. By sending ESC+'R'+n immediately after a warm boot.	After rewriting I/O byte LST : field. After a warm boot. After rewriting character generator. By sending ESC+'R'+n immediately after power-on	Not initialized.	In Export version 1.0, PRTFLG is cleared at power-on time. In Export version 1.0, PRTFLG contains information pertaining to LST : and display character set.
I/O device serial parameter (when RS-232C is used)	Input and output devices have different serial parameters.	Only LST: devices have different serial parameters.	Same as in Export version 1.0.	

Difference	Kana V1.0	Export version 1.0	Kana V2.0	Remarks
Character set specifications for display functions	ASCII, England , and Japan	ASCII, France, German, England Sweden, Denmark, Italy, Spain and Norway	Japan only	
Pointer to character generator table	Fixed.	Fixed.	Fonts data can be extracted from the user-supplied character generator by rewriting the area containing the starting address of the character generator table to be used. Codes 20H-7FH : RLCGENN(0F95EH) Codes 80H-9FH : RLCGENG(0F360H) Codes A0H-DFH : RLCGENK(0F362H)	

7.2 DIP Switches

7.2.1 General

PINE OS uses DIP switches to define the following:
1. Keyboard type
2. BIOS LIST device interface
3. International character set for the keyboard and display functions

The keyboard type can be identified automatically by the keyboard hardware that is actually installed, irrespective of whether the corresponding DIP switch is in the ON or OFF state.

7.2.2 DIP Switch Bit Definitions

Figure 7.2.1 shows the definition of the DIP switches. The state of bit 8 is determined by hardware depending on the keyboard actually installed, irrespective of the position of the DIP switch.

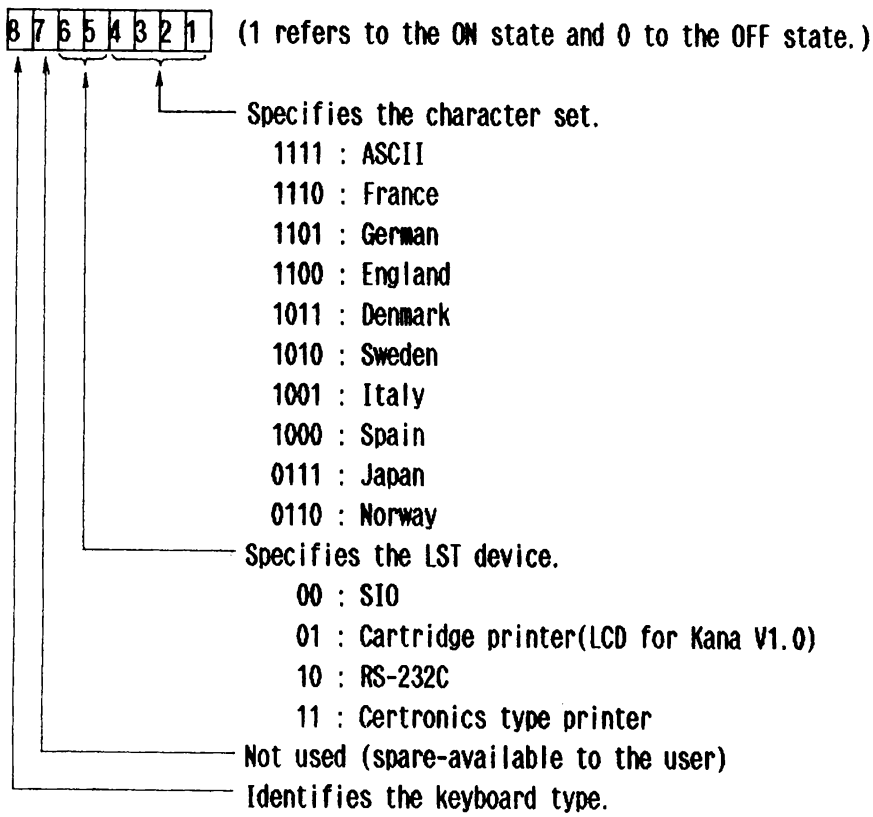


Fig. 7.2.1 DIP Switch Bit Definitions

7.2.3 System Processing

PINE OS reads in the DIP switch status during boot processing initiated by a system initialize or reset and stores it in YKCOUNTRY (ØF775H) for subsequent reference during key entry, display, and list processing.

7.2.3.1 Key entry processing

YKCOUNTRY bit 7 is referenced during key entry processing to identify the type of the keyboard installed. When the standard keyboard is used, YKCOUNTRY bits 3-Ø are used to indicate the character set employed.

7.2.3.2 Display processing

Flag areas for display processing includes YLDFLTC (ØF776H) and YLCOUNTRY (ØF777H). Bits 3-Ø of YKCOUNTRY are copied into YLDFLTC during boot processing and the contents of YLDFLTC are copied into YLCOUNTRY during a processing routine common to BOOT and WBOOT. The display function references YLCOUNTRY when converting character codes to the correct fonts of the specified character set.

7.2.3.3 List processing

The IO byte at RAM address ØØØ3H is referenced during list processing. When BOOT is started, it copies bits 5 and 4 of YKCOUNTRY into bits 7 and 6 of the IO byte. When the BDOS or BIOS list function is called, it reads in the IO byte into RIOBYTE (ØF529H). The actual list processing routine references RIOBYTE to determine the actual list driver routine.

During list processing, YKCOUNTRY is also referenced to output the international character set specification to the list device.

7.2.3.4 Programming notes

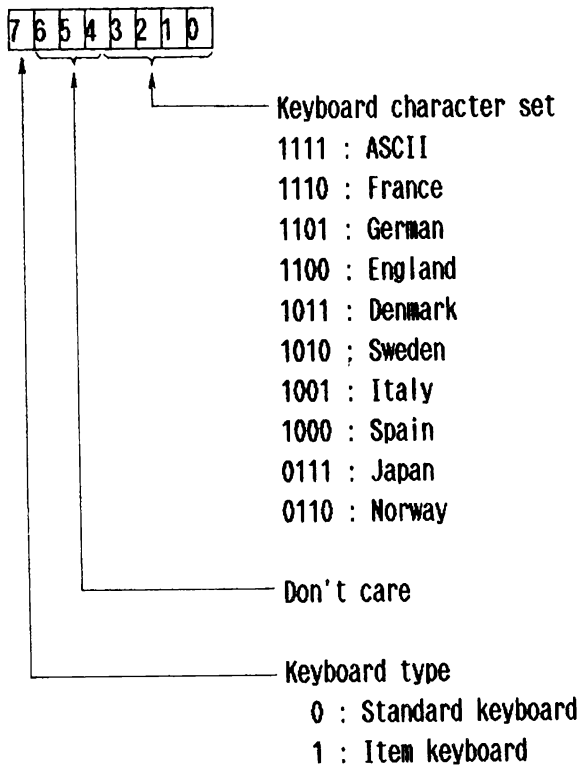
(1) The application program can check the state of the DIP switches using the READSW BIOS function.
(2) Since PINE OS reads the DIP switch status when a reset is performed, changes in the DIP switch status by the user do not take effect immediately. To make a change in the DIP switch status effective, press the RESET button.
(3) The character sets supported by the keyboard and display functions differ between OS Export and Kana versions.

Country / OS	ASCII	France	Germany	England	Denmark	Sweden	Italy	Spain	Japan	Norway
Kana V1.0	○	×	×	×	×	×	×	×	○	×
Kana V2.0	×	×	×	×	×	×	×	×	×	×
Export V1.0	○	○	○	○	○	○	○	○	×	○

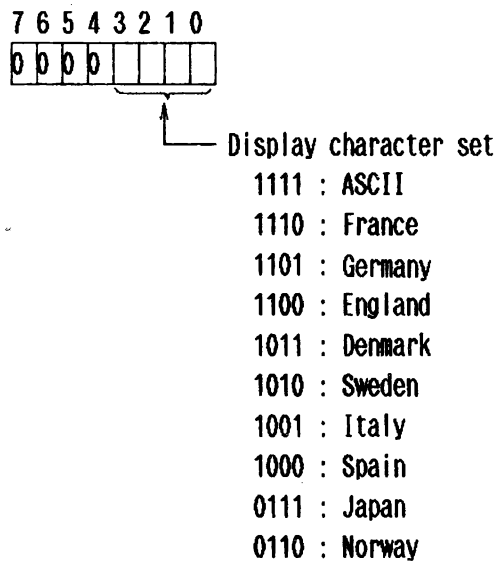
○ : Supported.
× : Not supported.

The system areas related to system processing are listed below.

YKCOUNTRY (ØF775H) 1 byte
Contains the DIP switch status read during BOOT. This flag is referenced by the key entry processing routine to identify the keyboard installed.



YLDFLTC (ØF776H) 1 byte
Contains the default display character set. This flag is loaded with the status of the DIP switches during BOOT processing.



YLCOUNTRY (ØF777H) 1 byte
Flag indicating the display character set. This area is loaded with a copy of YLDFLTC during BOOT or WBOOT processing.
The format of YLDFLTC is identical to that of YDFLTC.

RIOBYTE (ØF529H) 1 byte
Loaded with the value of IOBYTE (at ØØØ3H) when BDOS or BIOS is called.
The format of RIOBYTE is identical to that of IOBYTE. See Section 3.9, "I/O Byte" for details.

7.3 Guide for Programming ROMs

This section gives a guide for programming ROM devices for ROM capsules and cartridges.

7.3.1 Formats

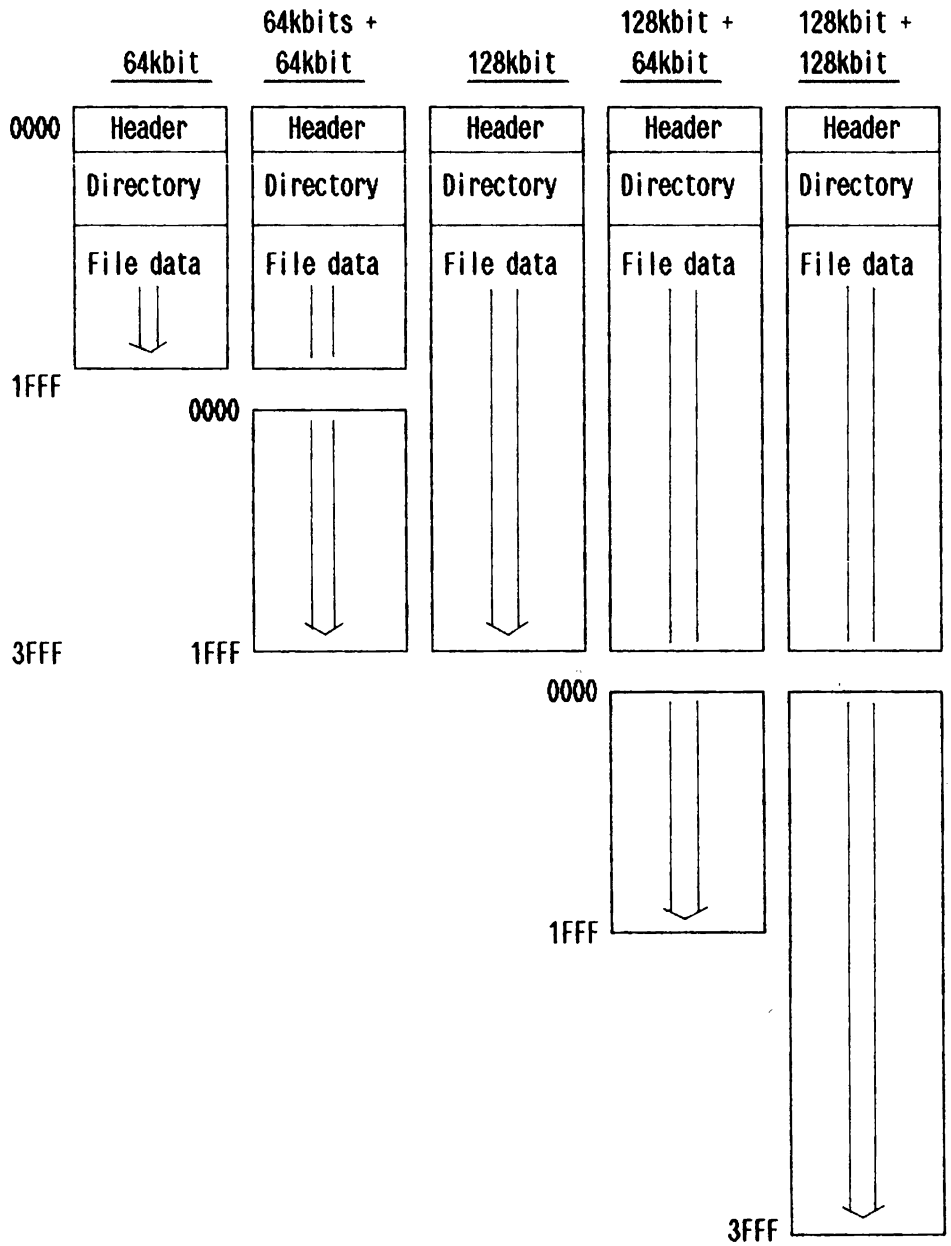
In PINE CP/M, ROMs can be programmed in two formats: M- and P-formats. The M-format is used for MAPLE and PINE ROM capsules and PINE ROM cartridges. Load-and-go programs and data files are created in this format. The P-format is dedicated to PINE ROM capsules and used to construct ROM-based programs.

The M- and P-formats differ in the location of the ROM header and directory areas. They also use different addressing schemes.

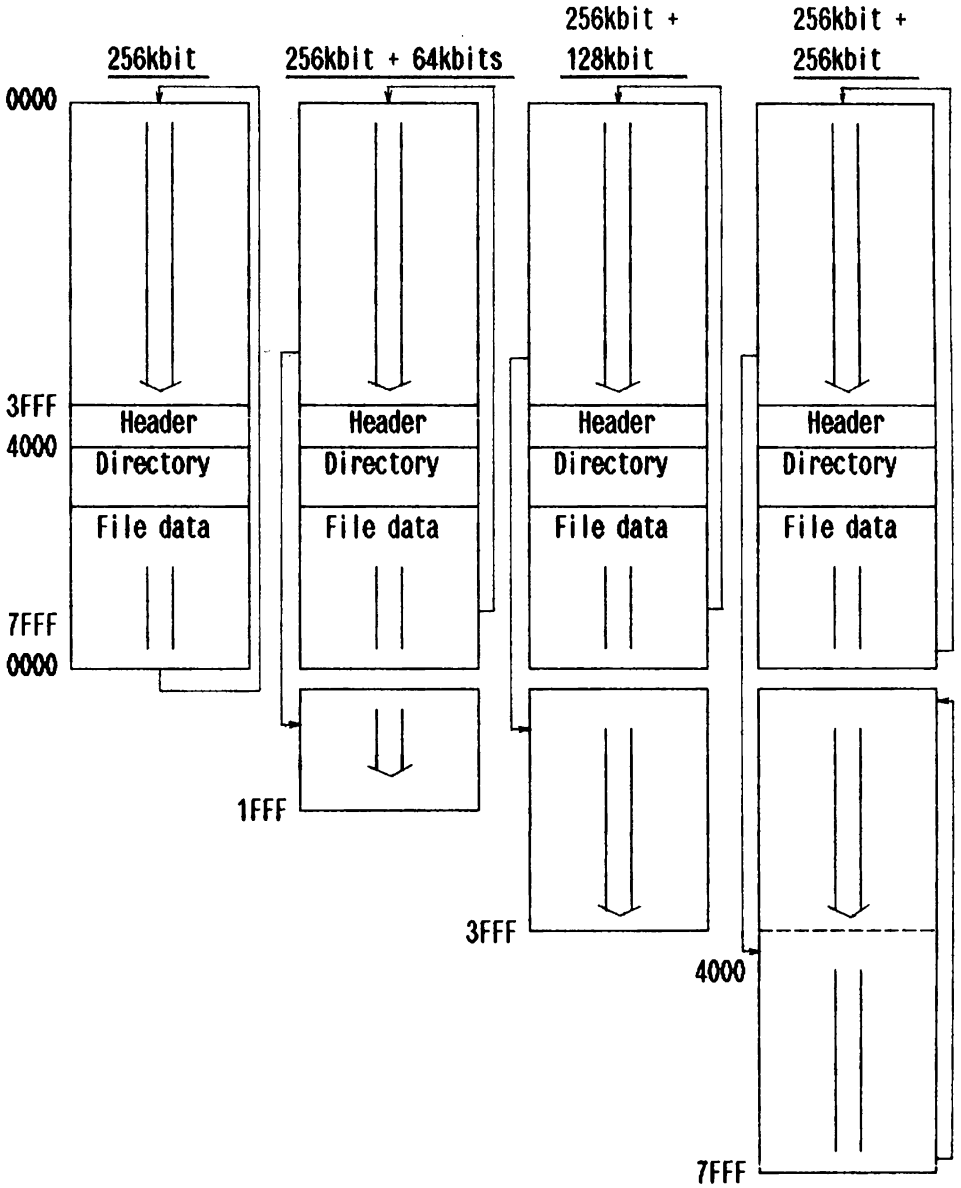
Supported ROM devices include 64K-, 128K-, and 256K-bit ROMs. Two ROM devices of the same capacity may be combined to form a single ROM cartridge or capsule.

(1) M-format ROM structure

The location of the directory for M-format PROMs differs depending on the type of the PROM device.

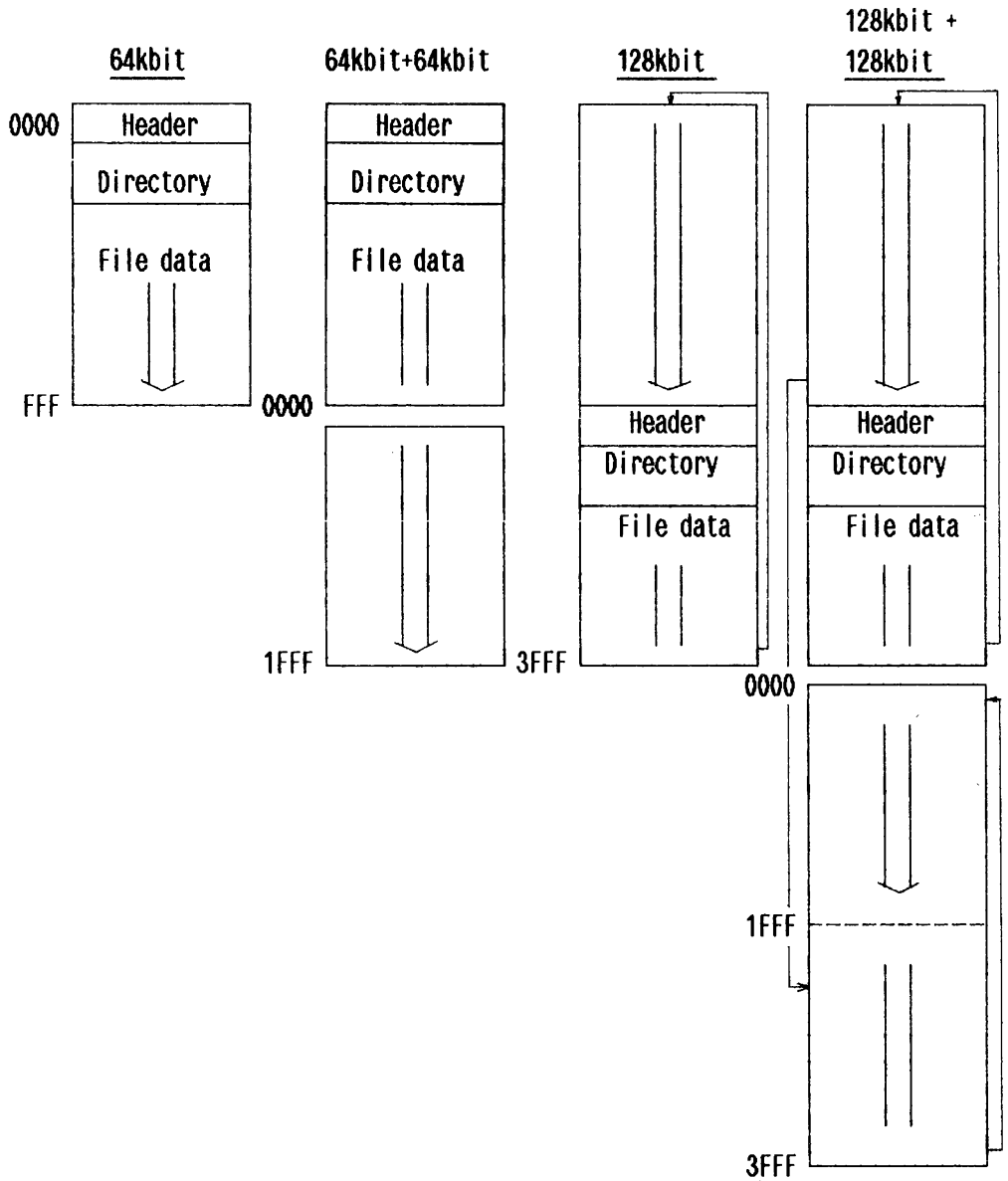


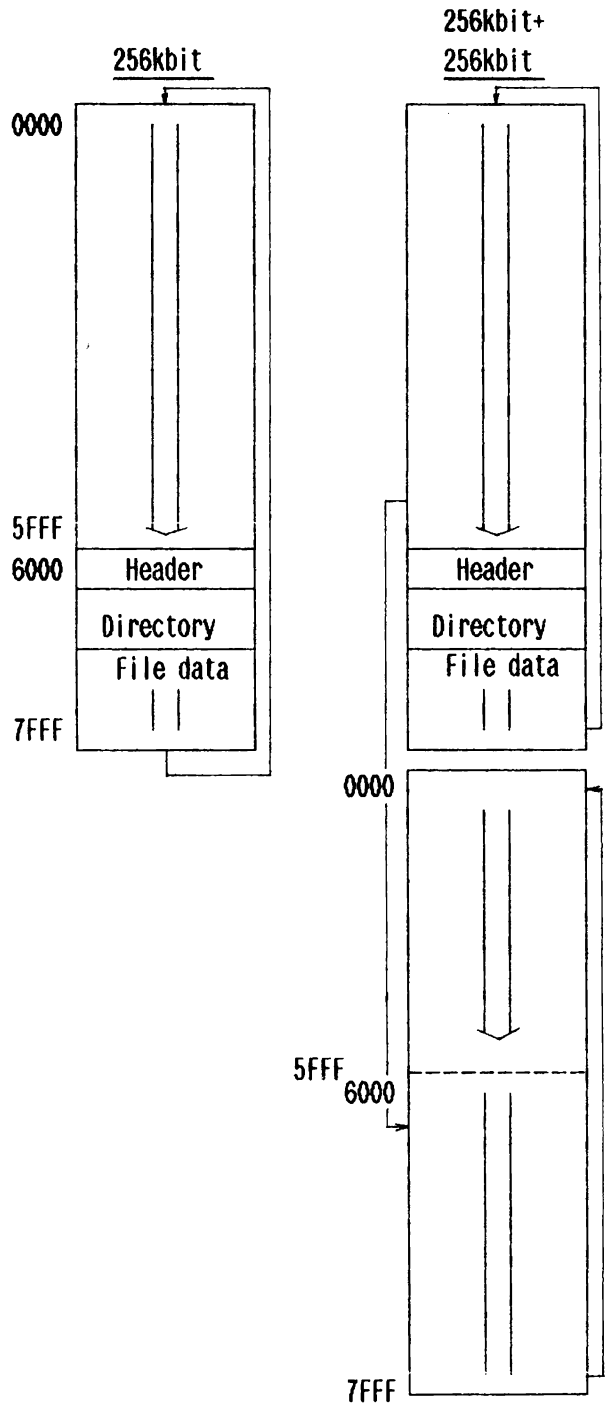
Arrows in the figure show the sequence in which ROM data is written.


256kbit
256kbit + 64kbits
256kbit + 128kbit
256kbit + 256kbit
0000
3FFF
4000
7FFF
0000
Header
Directory
File data
1FFF
3FFF
4000
7FFF

(2) P-format ROM structure

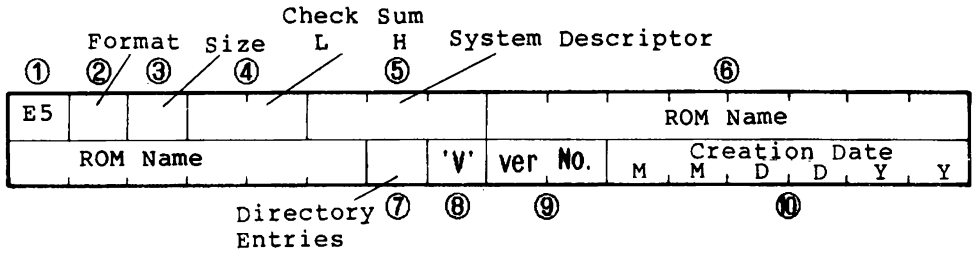
The location of the directory of P-format PROMs differs depending on the type of the PROM device.




256kbit
0000
5FFF
6000
Header
Directory
File data
7FFF
256kbit+
256kbit
Header
Directory
File data
0000
5FFF
6000
7FFF

7.3.2 Header Structure

The size of the header for a PROM file is fixed at 32 bytes. The structure of the PROM header is shown below.

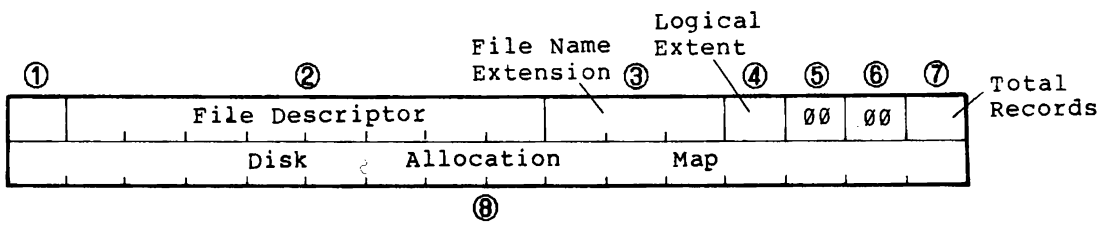


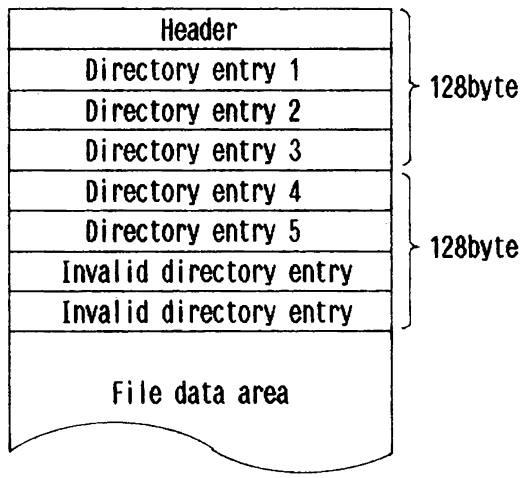
No.	Item	Size	Description
1	E5H	1	* Set to 0E5H * The system locates the beginnig of the ROM by checking this and the subsequent format fields.
2	Format	1	* Indentifies the format. 37H ···M Format 50H ···P Format
3	Capacity	1	* Indicates the capacity of the ROM in 1k bytes (binary format). The MSB is set to 1 for a dual-part PROM. 08H ··· 64k bits, single-part PROM 88H ··· 64k bits, 2-part PROM 10H ···128k bits, single-part PROM 90H ···2-part PROM with one part consisting of 128k bits. 20H ···256k bits, single-part PROM A0H ···2-part PROM with one part consisting of 256k bits
4	Check Sum	2	* 2-byte field containing the checksum of the data calculated by adding together the values of the data bytes from beginning to end of the data aera and taking the lowest order two bytes. The r lower order byte is placed in the first byteposi tion of the field. * This field is not used by the system.

No.	Item	Size	Description
5	System Name	3	* User-supplied system name. * This field is not used by the system.
6	ROM Name	14	* User-supplied system name. * This field is not used by the system.
7	Number of directories	1	* Indicates the size of the directory area in the number of directory entries. The header area is counted as one directory entry. The number of directory entries is rounded up to a multiple of 4 and stored in binary form. * This field can take values of 04H, 08H, 0CH, 10H, 14H, 18H, 1CH, and 20H. * The value of this field is calculated using the following formula : [{(Directory Entries)/4} + 1] * 4 ↑ ↑ Number of directory entries used Entry for header
8	'V'	1	* Set to 'V'.
9	Ver.No.	2	* User-defined ROM version number.
10	Date	6	* User-supplied date of ROM production in MMDDYY format. * This field is not used by the system.

7.3.3 Directory Structure

The system can manage a file of up to 16K bytes with each directory entry. For files larger than 16K bytes, the system uses two or more entries. The format of the directory entry is identical to the FCB of standard CP/M and is depicted in the figure below. The maximum number of directory entries is 31.



No.	Item	Size	Description
1			* Indicates the validity of the directory entry. 00H ··· Vaild directory entry E5H ··· Invalid directory entry * Invalid directory entries are created to ensure that the directory area ends on a 128-byte boundary. If, for example, five valid directory entries are created, counting the header as a single entry, two odd entries (64 bytes) will , dresult. Such entries are referred to as invalid directory entries(see the example below,) 
2	File name	8	* Contains a file name of up to 8 ASCII characters
3	File type	3	* Contains a file type of 3 ASCII characters.
4	Logical extent	1	* Contains the logical extent number(00H to 1FH) of this directory entry. * One directory entry can handle a file area of up to 16k bytes. Two or more entries are used for a file larger than 16h bytes. These entries are identified by numbers, called logical extent numbers, assigned to them sequentially starting at 00H.

No.	Item	Size	Description
5		1	* Set to 00H.
6		1	* Set to 00H.
7	Total records	1	* Contains the total number of records (00H to 80H)managed in this directory entry. * A record is the unit of data accessed by CP/M. Its size is 128 bytes. Since a single directory entry can handle a maximum of 16k bytes of file data, the greatest total number value for a single entry is 128(80H).
8	Disk allocation map		* Contains the disk allocation map of the ROM as a a ROM capsule or cartridge. The disk allocation map is made up of block numbers which identify the blocks allocated for the file(files are managed in 1K-byte units). The block number begins (with 1 which identifies the first 1k byte of the file. Since the disk allocation map is 16 bytes long, it can manage a maximum of 16 blocks(16k bytes) of file data. 