IT-2000D
(DOS version)

Technical Reference

Manual

(Version 1.00)

April 1998

Casio Computer Co., Ltd.

Table of Contents

Preface

The IT-2000 Technical Reference Manual (hereinafter referred to as this document) is provided to assist the user in developing programs to run on the Casio IT-2000 (hereinafter referred to as this terminal or IT-2000 or HT in this manual). Microsoft C/C++ Ver.7.0 or later, and the manuals supplied with it, is required to develop programs for this terminal.

Read Chapter 1 of this manual in its entirety to understand the features of this terminal.

Important notices to user

The information contained in this document may be modified without a prior notice.

Casio Computer Co., Ltd. shall not be liable for any outcome that result from the use of this document and the terminal.

Copyright notice

The contents of this document are protected by the Copyright Law of Japan.

This document may not be reproduced or transferred in part or in whole, in any form without permission from Casio Computer Co., Ltd.

About MS-DOS 6.22

The MS-DOS copyright is the proprietary of Microsoft and is protected by the United States Copyright Law and International Treaty provisions.

The MS-DOS software shall not be modified, reverse-engineered, decompiled, or disassembled. Any form of reproduction is also absolutely prohibited.

About trademarks

- AT and IBM PC/AT are registered trademarks of International Business Machines Corporation in the United States.
- MS, MS-DOS, Microsoft C/C++, Visual C ++, Visual Basic, and MS-Windows are registered trademarks of Microsoft Corporation in the United States.

1. Overview

1.1 Features of System

1.1.1 Development Concept

The IT-2000 is a data collection terminal for business use. After years of refinement Casio Computer Co., Ltd. has developed its hand-held type terminals so that they yield high speed and a high functionality in comparison to general personal computers. This allows improved efficiency in software development.

It has adopted the IBM PC/AT architecture and incorporated an IBM PC/AT compatible BIOS. It uses MS-DOS Ver. 6.22 for its OS. This has drastically improved the software development environment and compatibility to IBM PC/AT family applications.

The adoption of a power-saving type 32-bit CPU, the Intel 80486GX, allows the terminal to operate continuously for eight hours (when the backlight is off).

1.1.2 Hardware

- Global IBM PC/AT architecture standard is adopted.
- Compact design: 85 (W) x 196 (L) x 30 (H) mm, 430 g (approx.)
- Uses a 32-bit CPU (Intel 80486 GX) for 25 MHz high-speed operation.
- High-resolution (192 x 384 pixels), large-size liquid crystal touch panel.
- Supports various interfaces, including RS-232C (8-pin, 14-pin), IR, and PC card.
- High environmental adaptability: Operation temperature at between -5 and 50℃, water splash proof capability conforms to the IPXII standard, etc.
- Uses a small-size, large capacity lithium-ion battery pack as the main battery.
- Incorporates a large capacity flash ROM drive as the user drive.

1.1.3 Software

- MS-DOS Ver. 6.22 as the operating system.
- IBM PC/AT-compatible BIOS makes it easy to develop user application programs.
- Uses APM 1.1, for advanced low-power consumption capability.

- PC card slot conforms to PCMCIA Release 2.1 supporting various PC cards.
- Implements IrDA 1.1 protocol for high-speed infrared communication.
- System menu makes it easy to maintain the IT-2000 and install user application programs.
- Provides various development support tools including C-language libraries and communication utilities for developing business application programs.

1.1.4 Basic Specifications

IT-2000

Architecture		
	IBM PC/AT architecture	
External dimensions and weight		
	Dimensions	: 85 (W) x 196 (L) x 30 (H) mm
	Weight	: 430 g (approx.)
CPU		
	Intel 80486GX(32-bit)	
Memory		
	DRAM	: 4 MB
	F-ROM	: 0/4/8/12/16/24 MB (depending on the model)
	MASK ROM	: 8 MB, Windows file (IT-2000W only)
	BIOS ROM	: 1 MB (BIOS section: 256 KB, Drive C image: 768 KB)
Display and input		
	LCD panel	: 192 x 384 dots (FSTN semi-transparent LCD), with EL backlight
	Touch panel	: Analog, 192 x 384 dots
Interface		
	8-pin	: RS-232C
	14-pin	: RS-232C
	IrDA	: Standards 1.0/1.1
	PC Card	: PCMCIA Release 2.1
Power supply		
	Main battery	: Lithium-ion battery pack (x 1)
	Sub-battery	: Lithium-vanadium battery (x 1), lithium battery (x 1)
	Operating hours	: 8 hours (if backlight off)
	Backup period	: 2 weeks
Environment conditions		
	Temperature	: Operation -5 to 50 ℃
		: Storage -10 to 55 ℃
Reliability		
	Water-splash proof	: Conforms to IPXII standard
Software		
	BIOS	: IBM PC/AT compatible
	OS	: MS-DOS Version 6.22
	F-ROM	: NAND flash file system
	Basic functions	: Suspend/Resume, Auto Power OFF, Auto Backlight OFF, Auto Backlight ON/OFF with light intensity detection, Auto Power ON with timer/ring signal/detection of mounted I/O Box

1.1.5 Model Name

The following IT-2000s of MS-DOS version will be available. For price of each model, please consult with your local Casio representative.

Model	RAM	Flash ROM	Remark
IT-2000D10	4 Mbytes	-	
IT-2000D20	4 Mbytes	4 Mbytes	
IT-2000D30	4 Mbytes	8 Mbytes	
IT-2000D60	4 Mbytes	24 Mbytes	

1.2 System Configuration

1.2.1 Hardware Block Diagram
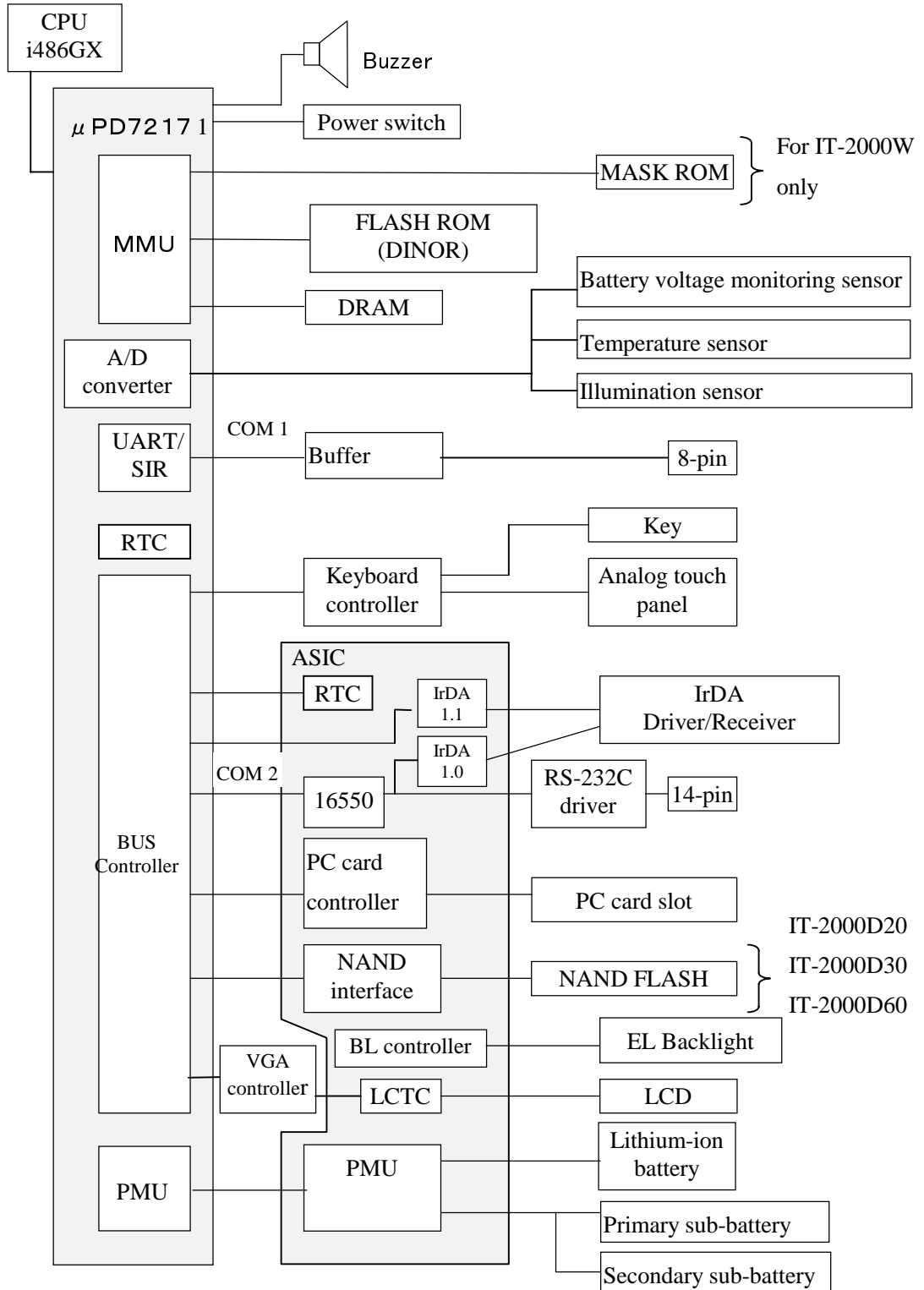
1.2.2   Supported Software

The software used with this terminal can be divided into two categories: the system software that includes the BIOS, OS, and device drivers and the user software such as the development tools. The system software is stored on the DINOR FLASH ROM (1 MB), and the user software is supported from the SDK CD-ROM (version 4.0) supplied by Casio at free of charge. The following paragraphs describe the software.

BIOS

The BIOS program is stored in the DINOR FLASH ROM. 256 KB of DINOR FLASH ROM is allocated specifically as the BIOS storage area.
The BIOS of this terminal consists of the standard PC/AT BIOS section, PEN BIOS for supporting the touch panel, extension BIOS for supporting devices inherent to this terminal, and APM BIOS for attain the low-power consumption capability.

MS-DOS Main Part

The main part of the MS-DOS Ver. 6.22 is stored in drive (C:).
In drive (C:) 768 KB of memory area in the DINOR FLASH ROM (1 MB) is allocated. Because of the capacity limitation, only the essential MS-DOS files are stored in drive (C:). Therefore, if using an MS-DOS file that is not included in the main part, copy it from the Backup CD-ROM (title on CD-ROM: MS-DOS version 6.22 Software) to the F-ROM drive (D:) or RAM disk (A:).
For information about each MS-DOS file refer to an MS-DOS manual, commonly available at book stores.

Device Drivers and System Files

These files must be loaded via CONFIG.SYS or AUTOEXEC.BAT at boot-up. These files are all stored in drive (C:).

File name	Storage location	Description
SYSDRV.SYS	Basic drive (C:)	System driver
TIME.SYS	Basic drive (C:)	Clock control driver
CS.EXE, etc.	Basic drive (C:)	PC card driver
CASIOAPM.COM	Basic drive (C:)	Touch-panel enabler
ENDATA.COM	Basic drive (C:)	ATA card-related data
CKRAMDSK.EXE	Basic drive (C:)	RAM disk checker
CKRAMDSK.DAT	Basic drive (C:)	
CALIB.EXE	Basic drive (C:)	Calibration
SYSMENU.EXE	Basic drive (C:)	System Menu
HWWMAN.EXE	Basic drive (C:)	Hardware window manager
KEYPAD.EXE	Basic drive (C:)	Keypad
KEYPAD.DAT		

TFORMAT.EXE	Basic drive (C:)	F-ROM drive formatter

Utilities

For information about the utilities refer to Chapter 8 "Utility".

File name	Storage location	Description
CAL.EXE	SDK	Calendar utility
CALC.EXE	SDK	Calculator utility
CLOCK.EXE	SDK	Clock utility
CHKBATT.EXE	SDK	Power status indication utility
XY.EXE	Basic drive (C:)	XY utility
FLINK.EXE	Basic drive (C:)	FLINK utility
LCDREV.EXE	SDK	Reverse video utility

Development Tool Libraries

File name	Storage location	Description
SLIBSYSD.LIB	SDK	System library
MLIBSYSD.LIB		
LLIBSYSD.LIB		
SYSLIB.H		
SLIBPAD.LIB	SDK	Keypad library
MLIBPAD.LIB		
LLIBPAD.LIB		
PADLIB.H		
KEYPADP.EXE	SDK	Keypad for PC simulation
HWWMANP.EXE		
SLIBOBRD.LIB	SDK	OBR library
MLIBOBRD.LIB		
LLIBOBRD.LIB		
OBRLIB.H		
COM2KEY.EXE	Basic drive (C:)	COM < -- > Key for DEBUG
PENMOUSE.COM	SDK	Pen mouse driver
PMON.COM	Drive (C:)	Switching DOZE mode on/off
PMOFF.COM		

1.3   Precautions

● If reading the internal clock with INT21h the significant data should include and be limited to the seconds digits. On this terminal the time is read directly from the RTC so that the correct time can be attained at any moment, even during extended continuous use. As a result the 1/100 of a second digit is ignored. (refer to Chapter 6.3 "Clock Control Driver")

● If it is necessary to reboot the system from an application, use the dedicated system library. However, the reboot operation that uses INT19h of the BIOS I/F does not work.

● Many commercial PC programs use a VGA screen (80 (H) x 25 (V)). If these programs are run on this terminal (24 (H) x 24 (V)) part of the message may not be displayed on the screen.

● Writing to a PC card should always be performed by terminating the write action through the flash-out process. Otherwise, if system operation is suspended while writing to an SRAM card or ATA card, the data on the card may be damaged.
To activate this flash-out process use the "_dos_commit()" function of Visual C/C++ or Commit Function(68h) of DOS.

● Layout your screen display in such a manner that dark characters lie on a white background. With LCDs a white background provides the most legible displays. If intermediate colors (half tones) are desired, use the following two colors (tones).

Note:
Due to technical reasons the display of the B/W LCD may change to reverse video if an application program developed by the user on a PC is executed without modification on this terminal. To restore the normal display use the Reverse Video Utility (refer to Chapter 8.8 "Reverse Video Utility").

● Key input operation is disabled for about one second after the Power has been turned on. This is not a malfunction. This occurs because the monitoring timer starts operating the moment the Power switch is turned on and does not allow key input for about one second until this timer expires. Thus, key input is not possible.

● If an LB1 event (low main battery voltage) occurs, the alarm buzzer starts sounding and system operation is suspended in about 10 minutes. If the alarm buzzer starts sounding, terminate the current operation as soon as possible and recharge the main battery.

- This system will not execute an alarm indication for an LB2 event (low sub-battery voltage) or LB3 event (low SRAM card battery voltage). Therefore, the application program side must acquire the alarm status via the system library and display an appropriate alarm message.

- If the volume of the buzzer is set to zero by the System Menu or system library, the LB1 (low main battery voltage) alarm will not be heard. Also, other sounds issued by the system will be inaudible.

- If the system is booted from a PC card and if a large-size program that resides on the card is called from AUTOEXEC.BAT file, an error may result. To avoid this problem refer to Chapter 4.2 "How to Write CONFIG.SYS and AUTOEXEC.BAT".

- The time limits that can be set for the Auto Power OFF (APO) function are 0 minute, 1 minute and 30 seconds, 2 minutes and 30 seconds, up to a maximum of 15 minutes and 30 seconds. This timer has an error of +/-23 seconds.

- Do not open the battery compartment lid while the power is on. If it is opened accidentally, an emergency alarm sounds. In case such the event occurs, close the lid at once.
  When you change the main battery, be sure to switch off the power before opening the lid.

- An SRAM card formatted with the DT-9000 (a Casio handy terminal) cannot be used or formatted with the IT-2000.

- If the battery pack is installed for the first time after purchase, or if it is installed after the main unit has not been used for a long period of time, install the battery and wait approximately eight seconds before turning the power on. This must be done because it takes approximately eight seconds until sufficient power can be raised for the emergency process. And, during this interval the power cannot be turned on even if the Power switch is turned on.

- If the power is turned on for the first time after purchase and there is no installed application, the System Menu will always appear. Before using applications call this System menu to install them. (refer to Chapter 7.4.3 "Operation Check on IT-2000 (Using COM2KEY/XT)")

- The backlight is turned off by means of the ABO (Auto Backlight OFF) function. However, it is turned off 1.3 seconds after the setup time. This is because the system has 1.3 seconds of monitoring time before the internal timer is started.

- Do not input "^P" from the DOS prompt. If it is input, "^P" requests DOS to redirect console output to printer. However, the IT-2000 does not have the printer being installed, it will enter into wait mode.

For more information about the system library refer to Chapter 7.6.2 "System Library".
Also, refer to Chapter 2.2.4 "Battery Voltage Monitoring Process" for information about the low-battery voltage notification function.

2.   Basic Software

2.1   Overview

2.1.1   Software Configuration

The following diagram shows the software configuration of the IT-2000.
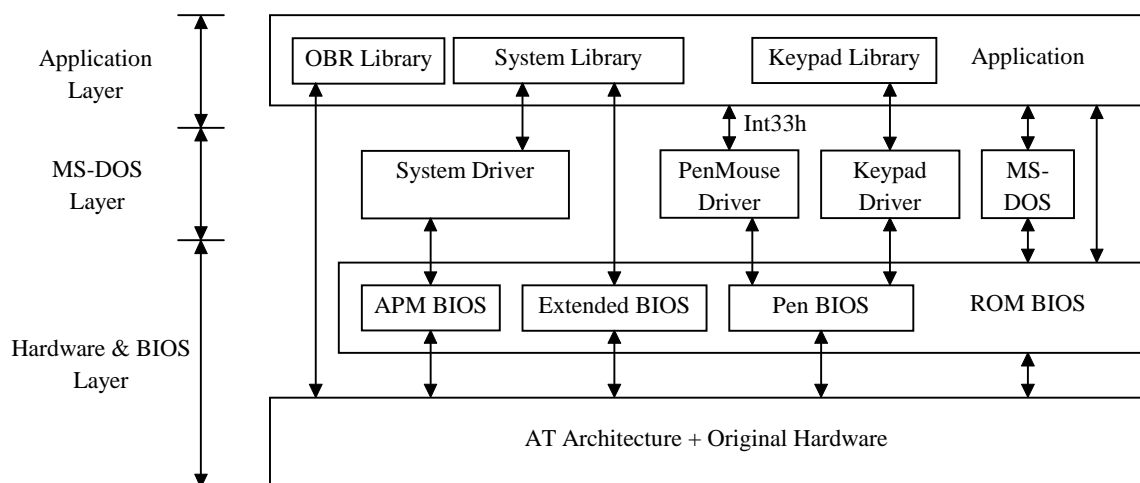
Fig. 2.1

Note: The PenMouse driver and Keypad driver cannot co-exist on the system.

2.1.2 Memory Map

The memory map of the IT-2000 is as follows.

Extended Memory	
	100000h
ROM BIOS	0F0000h
NAND DISK BIOS/VGA BIOS	0E0000h
Memory Mapped Disk I/F	0DC000h
PC Card I/F	0D8000h
EMS Windows 16 KB x 4	0C8000h
Reserved	0C0000h
Video Buffer 128 KB	0A0000h
System RAM 640 KB	000000h

Fig. 2.2

2.1.3   Drive Configuration

The drive configuration differs for each model as described in the following:

● **If F-ROM drive is supported**

Drive A: [Read and Write]	RAM disk
	This drive is prepared for use after the RAM disk size is specified from the System Menu. The contents of this RAM disk will not be erased through a boot process or by pressing the RESET switch.
Drive C: [Read Only]	Basic drive (DINOR FLASH ROM)
	This drive starts up MS-DOS. The main body of MS-DOS and maintenance programs such as the System Menu, etc., are stored in this drive.
Drive D: [Read and Write]	F-ROM drive
	Application programs are stored on this drive. The drive size (storage capacity) differs depending on the model.
Drive E: [Read Only]	Drive for Windows files (IT-2000W only)
	A ROM that stores Windows files is assigned to the drive E: on Windows models. This is a reserved drive on DOS models. In this case note that if this drive is accessed , an INT24h error will occur.
Drive F: [Read Only]	Drive for booting a card
	This read-only drive functions only while a card is being booted. For information about the mechanism of booting a card refer to Chapter 4.3 "Card Boot".
Drive G: [Read and Write]	PC card drive
	This drive is required if the application program accesses the PC card. This drive is prepared for use by loading the PC card driver via CONFIG.SYS.

Note:

The drive letter of each drive is reserved. Therefore, these drive letters are not changed even if the RAM disk is not used.

The drive configuration described in "If F-ROM drive is not supported" on page 18 will be used only if the drive D: is unformatted or is not recognized by the system for some reason.

However, this will rarely occur because the drive D: has been formatted at the factory.

● **If F-ROM drive is not supported**

Drive A: [Read and Write]	RAM disk
	This drive is readied for use after the RAM disk size is specified from the System Menu. The contents of this RAM disk will not be erased through a boot process or by pressing the RESET switch.
Drive C: [Read Only]	Basic drive (DINOR FLASH ROM)
	This drive is used to start MS-DOS. In this drive not only the main body of MS-DOS but also the maintenance programs such as the System Menu, etc., are stored. This is a read-only drive.
Drive D: [---]	Reserved drive
	This is a reserved drive. If this drive is accessed, an INT21h error will result.
Drive E: [Read Only]	Drive for booting a card
	This is a read-only drive that functions only while a card is booted. For information about the mechanism of booting a card refer to Chapter 4.3 "Card Boot".
Drive F: [Read and Write]	PC card drive
	This drive is required if the application program accesses the PC card. This drive is prepared for use by loading the PC card driver via CONFIG.SYS.

2.2 Basic System Operation

2.2.1 Overview

Basic operation of this system on the terminal consists of the suspend/resume process and boot process operated by means of the Power switch and RESET switch, as shown in the following diagram.
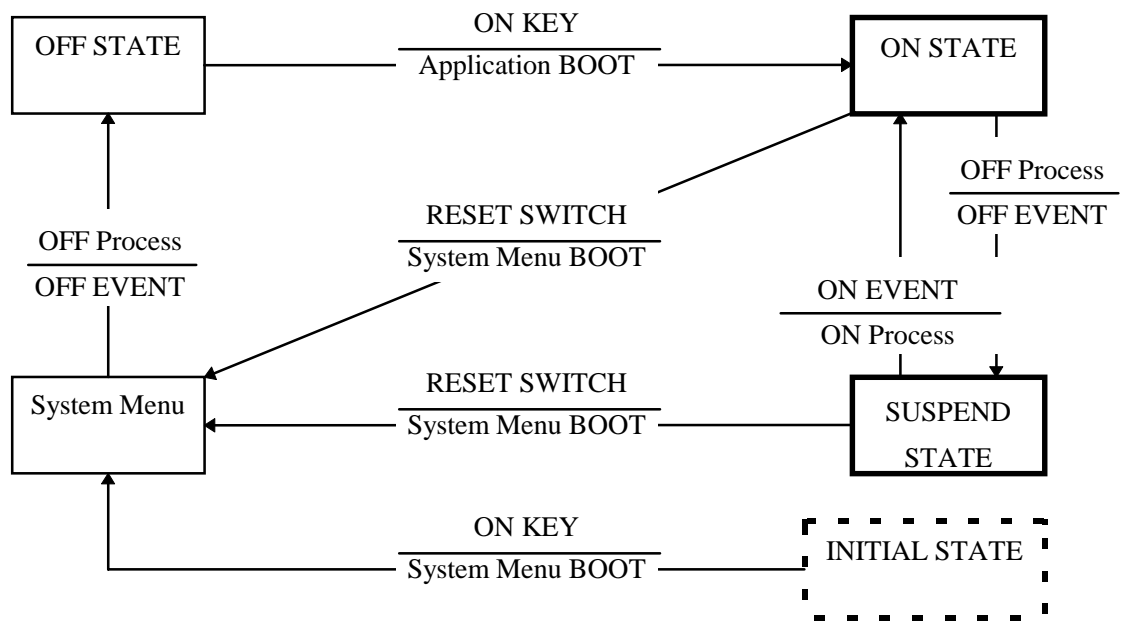
Fig. 2.3

During normal operation the system status will move between the ON state and the OFF state, shown in the above diagram, by pressing the power key.

The SUSPEND state is a state from which the previous state can be returned to at any time. The process of returning from the SUSPEND state to the ON state is called the resume process.

The RESET switch is used to either re-start the system or to initiate the System Menu, which is the maintenance program. Press this RESET switch to start hardware initialization followed by initiation of the System Menu. This process is called the System Menu boot process.

If an OFF event occurs while the System Menu is operating, the system shifts to the OFF state.
If the ON key is pressed in the OFF state, the boot process is executed again and an appropriate application program will be loaded. This process is called the application boot process.

The following table summarizes the power-on processes provided for this terminal.

System Menu boot process	Always executes CONFIG.SYS and AUTOEXEC.BAT located in drive (C:) for starting up the MS-DOS.
Application boot process	Searches for CONFIG.SYS and AUTOEXEC.BAT prepared by the user and starts up MS-DOS from the drive where they are located.
Resume process	Restores the memory conditions that existed before the power was turned off and continues operating according to the conditions.

2.2.2   Power ON Process

Overview

The ON process is provided to make the system ready for use (ON state). The actual process varies depending on the settings at that point in time and the last OFF factor (the cause of the OFF action).

ON factors:
- Pressing the Power switch
- Pressing the RESET switch
- Power ON alarm
- Reception of RING signal
- Mounting on the I/O Box

OFF factors:
- Pressing the Power switch
- Pressing the RESET switch
- Auto Power OFF (APO)
- Power OFF by software
- Auto Power OFF due to lower battery voltage
- Emergency Power OFF due to lower battery voltage

Note:

For more information power OFF factors refer to Chapter 2.2.3 "Power OFF Process".

This ON process is divided into two processes: the "Resume process" for continuing the previous process and the "Boot process" for re-loading MS-DOS. The Boot process can be further broken into the "Application boot" and the "System Menu boot" processes.

Fig. 2.4

```
                              ┌──────────────┐         ┌────────────────────────────┐
                              │ Boot Process │─────────│  Application Boot Process  │
┌─────────────┐              └──────────────┘         └────────────────────────────┘
│ ON Process  │◁
└─────────────┘              ┌────────────────┐       ┌────────────────────────────┐
                              │ Resume Process │       │ System Menu Boot Process   │
                              └────────────────┘       └────────────────────────────┘
```
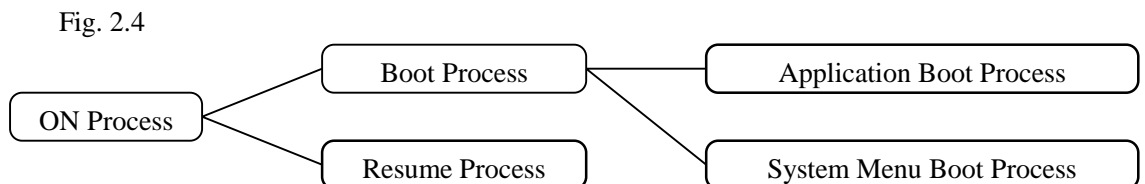
- **Application Boot Process**

Searches CONFIG.SYS and AUTOEXEC.BAT files according to the priority given to each drive and, if these files are found, sets the drive where these files are located as the current drive.
(refer to "Application Boot Process" on the next page).

- **System Menu Boot Process**

  Pressing the RESET switch sets the drive C: as the current drive, and MS-DOS is loaded from that drive. As a result, the System Menu that includes the maintenance program will be initiated (refer to "System Menu Boot Process" on this page).

- **Resume Process**

  This process restores the conditions that existed before the power was most recently turned off. Any application program that was running at that point in time can be continued.

  The contents of the above listed processes will be described in the following sections.

Application Boot Process

The application boot process is used to initiate application programs that have been installed in the system by the user. The main system will search for CONFIG.SYS and AUTOEXEC.BAT files according to the priority given below.

The system assigns the first drive on which the files are found as the current drive, and boots MS-DOS from the drive C. Consequently, if the CONFIG.SYS and AUTOEXEC.BAT files created by the user are located on one drive, MS-DOS will be booted from the drive assigned as the current drive. Under factory defaults it is apparent that the CONFIG.SYS and AUTOEXEC.BAT files created by the user cannot be found. If this occurs, therefore, the CONFIG.SYS and AUTOEXEC.BAT files located in the drive C: are selected and the System Menu will be initiated.

> **Priority of the drives:**
>
> If the F-ROM drive is provided
>
>   [Card drive (F:)] -> [RAM drive (A:)] -> [F-ROM drive (D:)] -> [Basic drive (C:)]
>
> If the F-ROM drive is not provided
>
>   [Card drive (E:)] -> [RAM drive (A:)] -> [Basic drive (C:)]

Note:

The RAM disk (A:) is valid for use only if setup is made through the System Menu.

System Menu Boot Process

The System Menu boot process is used to initiate the System Menu, which is nothing but a maintenance program for this terminal system. The System Menu boot process will be executed only if the RESET switch at the rear of the main unit is pressed.

If, in addition, a power OFF factor is encountered during the execution of the System Menu, the next boot process will be the Application boot process.

Note:

- The RESET switch can be used not only for initiating the System Menu but also as the forced restart witch when the user application program under development hangs. However, note that if the RESET switch is pressed while the disk is being written to, the data may be corrupted. Therefore, the RESET switch should be pressed only while the power is off.
- Clock data or information on the RAM disk will not be lost if the RESET switch is pressed.

Resume Process

When the power is turned on the resume function resumes system operation under the conditions that existed the last time the power was tuned off. Application programs are continued as soon as the power is resumed.

Setup of Resume Process ON/OFF

The default settings have been made so that every OFF factor encountered during the operation of an application program is the objective of the resume process. However, these default settings can be modified so that the system reacts differently to OFF factors by means of the system library. For example, according to the default settings, pressing the Power switch will suspend and resume the execution of an application program. However, it is also possible to simply reboot the system with the Power switch without activating the resume function if such a setup is made. However, note that this setup is not permanent. The resume process is replaced by the boot process once only right after the system library is called.

ON Factors

Various ON factors used to turn on the system are explained below.

- Pressing the Power switch

  If the Power switch is pressed while the system is off, the system power can be tuned on. When the power is turned on the system operation sequence proceeds as described in "Relationship between OFF Factors and ON Processes" on page 24.
- Pressing the RESET switch

  Press the RESET switch to turn on the system power. In this case the System Menu will always be initiated.

This terminal has the Auto Power ON function which automatically starts the system. This Auto Power ON function can operate in one of the following three ways:

- **Auto Power ON function (only affects the resume process) activated by alarm**

  The system power can be turned on (resumed) at the specified time by means of an alarm. However, this will not function if the next start-up method is set to the boot process in the system library.

- **Auto Power ON function activated by the RING signal**

  This function can be used if a modem is connected to the 14-pin expansion interface. In this case the system power can be turned on by the detection of the RING signal from the modem. Remember that Power ON by means of the RING signal is prohibited by default. Execute this function using the system library to enable the Power ON function to be activated by the RING signal. System operation after the power is turned on follows the sequence described in " Relationship between OFF Factors and ON Processes" on this page.

- **Auto Power ON activated by mounting on the I/O Box**

  The system power can be automatically turned on as soon as this terminal is mounted on the I/O Box. However, this function is effective only if power is supplied to the I/O Box. This function is enabled by default, however, it can be disabled using the system library. System operation after the power is turned on proceeds according to the sequence described in "Relationship between OFF Factors and ON Processes".

Relationship between OFF Factors and ON Processes

As described in the above overviews, the ON process (the Boot process or Resume process) will run differently depending on the last OFF factor (what caused the OFF) and the conditions that existed when the power was turned OFF. The following table shows the relationship between the OFF factors and the ON processes that take place the next time the power is turned on.

OFF factor	If an application is running	If the System Menu is on
Power switch	Resume process or Application boot process (see note below.)	Application boot process
Auto Power OFF		
Software OFF		
Low battery voltage (LB1)		
Low battery voltage (LB0)	Resume process	
RESET switch pressed	System menu boot process	System menu boot process

Note:

Depends on whether the resume function is enabled or disabled. With this setup the next boot process can be designated as the Application boot process.

2.2.3 Power OFF Process

Overview

Turns off the system power. However, the power to all the devices is not turned off and some can be used for storing the information required for the next resume operation. This process is called the suspend process and the state of the system while off is called the suspend state.

The suspend process can be divided into two categories: one is the normal suspend process which is the usual off method and the other is the critical suspend process to execute the emergency escape process for protecting the system from drops or bumps. Either of these suspend processes will be selected depending on the OFF factor, as described later.

Normal Suspend Process

If the Power switch is held down for more than one second while system is on, the system power will be turned off. The process that takes place at this time is the normal suspend process. Before this suspend process is executed, the application currently running is informed of the suspend request (OFF factor) by the system. Then the system stores the information required for resumption and turns off the power.

Hereinafter the suspend process (or OFF process) refers to the normal suspend process.

For information about the method used by each application to detect the occurrence of an OFF factor (suspend event), refer to Appendix C Acquisition of Suspend/Resume Event and Power Status.

Critical Suspend

This is a suspend process that takes place in an emergency. Since this critical suspend process should achieve its escape process with very little power in the system, only essential information can be retained.

The system will not inform the application currently running of the fact that it is critically suspended. However, the application will be informed of the fact that it was critically suspended at resumption.

For information about the method used by each application to receive this information, refer to Appendix C Acquisition of Suspend/Resume Event and Power Status.

OFF Factors

The OFF factors refer to various causes that make the system enter the OFF state (suspend state), as follows:

OFF factor	Description	Suspend
Power switch	System operation can be suspended by holding down the Power switch for more than a second. (see note)	Normal
Auto Power OFF (APO)	System operation automatically shifts to the suspend state if key or touch panel operation is not performed for a specified period of time. The duration until Auto Power OFF occurs can be set and modified through the System Menu or system library.	Normal
Software Power OFF	The system can be made to enter the suspend state by calling the system library from the application program.	Normal
Power OFF due to time-out of low battery voltage (LB1) alarm	The system will issue an alarm (buzzer) if the remaining battery voltage falls below the low main battery voltage alarm level. If this occurs, recharge the battery or replace it within ten minutes. If the battery is not replaced or recharged the system automatically shifts to the suspend state to protect the data.	Normal
If main battery voltage falls to an inoperable level (LB0)	If the terminal is used while the LB1 alarm, mentioned above, is sounding, the main battery voltage may reach the LB0 level. If this occurs the system will execute the critical suspend process and forcibly turn off the power. Therefore, if the LB1 alarm sounds, recharge or replace the battery as soon as possible.	Critical
Power OFF due to RESET switch pressed	Press the RESET switch to forcibly turn off the system power. If this is attempted to initiate the System Menu, it is strongly recommended to complete the application running at present then turn off the system power with the Power switch before hand.	Restart

For more information about LB0 and LB1, refer to Chapter 2.2.4, "Battery Voltage Monitoring Process".

For information about the system library refer to Chapter 7.6.2. "System Library".

For information about the System Menu refer to Chapter 3 "System Menu".

For information about the method used by each application to acquire a power ON/OFF event, refer to Appendix C Acquisition of Suspend/Resume Event and Power Status.

Note:

Hold down the Power switch for more than one second until the power is off. This is done to prevent the power from accidentally being turned off by the user. In addition, key input will not be enabled for approximately one second after the Power switch has been pressed.

This occurs because the monitoring timer starts operating the moment the Power switch is pressed and does not allow key input for about one second until this timer expires.

After this interval, key input becomes possible.

2.2.4   Battery Voltage Monitoring Process

This terminal uses a main battery (lithium-ion battery pack) for driving the main unit, and a primary sub-battery (lithium battery) and a secondary sub-battery (lithium-vanadium battery) for backup. Application programs can acquire the status of these batteries through the APM BIOS or system library. Refer to Appendix C Acquisition of Suspend/Resume Event and Power Status.

Battery Operation Scheme
The following diagram shows how each battery operates within the system.
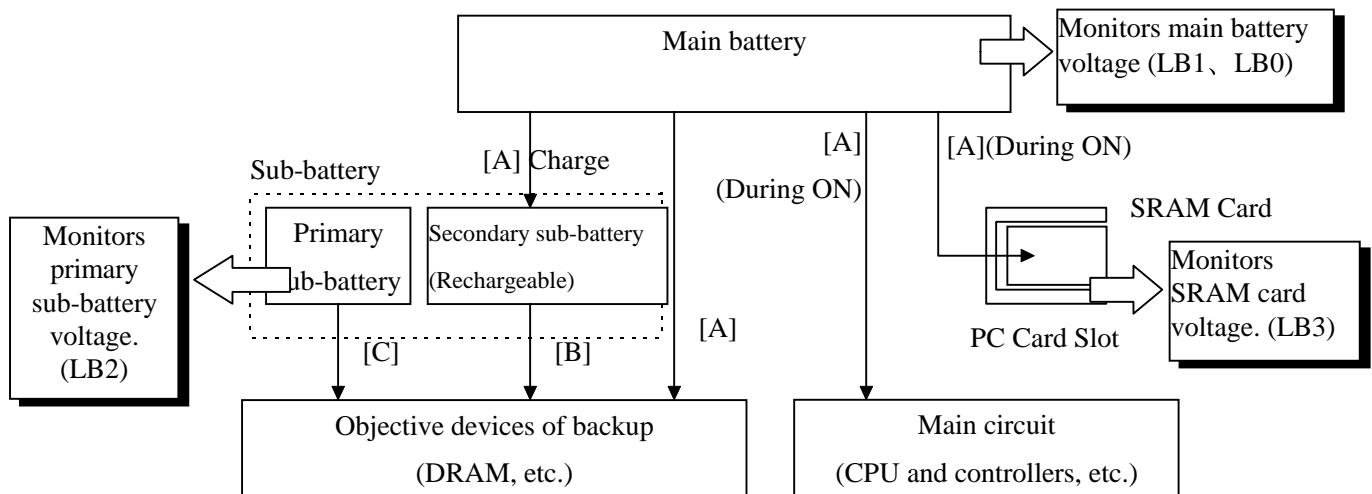
Fig. 2.5

[A] This is the power supply route where the fully charged main battery is installed.
While the power is on, the main battery supplies power to all the devices, including the main circuit, PC card slot and DRAM, and, at the same time, it charges the secondary sub-battery.
In the suspend state, it stops the supply of power to the main circuit and PC card, but continues to supply power to the DRAM and charge the secondary sub-battery. In this route neither the primary nor the secondary sub-batteries are used.

[B] This is a power supply route operating where the main battery is absent or not fully charged. The DRAM is back-upped by the voltage of the secondary sub-battery. The primary sub-battery is not used.

[C] This power supply route operates if the main battery and secondary sub-batteries are not fully charged. The DRAM is backed-up by the voltage of the primary sub-battery. If the voltage of this primary sub-battery falls below the limit level, an LB2 event occurs.

Low Voltage Level

The IT-2000 continuously monitors the voltage of the main battery, the primary sub-battery, and the SRAM card battery. This allows an application program to determine through the system library if the voltage of each battery reaches a warning level.

The following table summarizes the low battery voltage warning levels, which application programs can acquire through the system library.

Name	Abbreviation	Objective battery	Description
Low main battery voltage warning level	LB1	Main battery	Indicates that the main battery voltage has reached a limit level that requires a warning to be issued. The system sounds the buzzer to issue an alarm. If this occurs, the user must replace the main battery within ten minutes. If the battery is not changed within ten minutes, the system automatically executes the suspend process.
Low sub-battery voltage warning level	LB2	Sub-battery	Indicates that the sub-battery voltage has reached a limit level that requires a warning to be issued. Since the system does not issue an alarm, the application program must execute a warning by acquiring the status from the system library. The sub-battery must be replaced according to the procedure described later.
Low SRAM card battery voltage warning level	LB3	SRAM card battery	Indicates that the SRAM card battery voltage has reached a limit level that requires a warning to be issued. Since the system does not issue an alarm, the application program side must execute a warning by acquiring the status from the system library. The SRAM card battery must be replaced according to the procedure described later.

There is also a main battery inoperable level (LB0). This is the status of the main battery when its voltage falls below LB1. If this happens, the system executes an emergency power off (critical suspend). Therefore, this level is also referred to as the emergency escape process level.

This status cannot be acquired from the application side, since the system turns off the power as soon as the voltage reaches LB0.

Main Battery Voltage Monitoring

If the main battery voltage reaches LB1, the system issues a warning buzzer. If this warning buzzer sounds, either start recharging the battery or replace it with a fully charged battery as soon as possible.

If one of these measures is not taken within ten minutes, the system will forcibly turn off the power for safety. The following diagram shows the main battery voltage against the time axis.
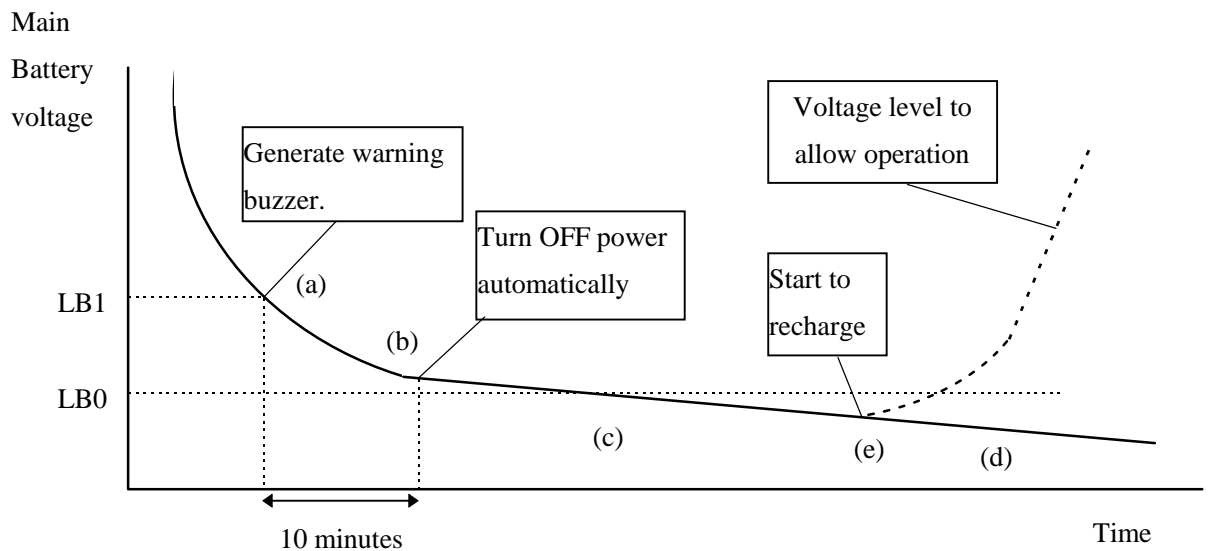
Fig. 2.6

(a) If the main battery voltage reaches LB1, the low battery voltage warning alarm sounds.

(b) Unless the main battery is either replaced or recharged within ten minutes, the system power is automatically turned off to protect the data.

(c) If the main battery voltage falls further and reaches LB0, the system automatically shuts off the power to the main unit (critical suspend).

(d) If the main battery voltage drops below LB0, the main unit power cannot be turned on even if the Power switch is pressed.

(e) If the main unit is mounted on the I/O Box or connected to the AC adaptor, charging of the battery is initiated and the main battery voltage will gradually increase.

(f) Once the main battery voltage has been recharged to an operable level, it is possible to turn on the power to the main unit.

For information about the method used to replace the main battery refer to Chapter 2.2.6 "
How to Replace or Recharge Batteries".

Sub-battery Voltage Monitoring

The sub-batteries are used for system backup while the main battery is being replaced. The sub-batteries consists of two units: the primary sub-battery (button-type lithium battery) and secondary sub-battery (button-type lithium-vanadium battery). The secondary sub-battery is recharged by the voltage of the main battery.

While the fully charged main battery is installed , the entire system is backed-up by the main battery, and the secondary sub-battery is charged by the voltage of the main battery. If the main battery is removed, the job of system backup shifts to the secondary sub-battery. If the secondary sub-battery voltage drops below the required level while the main battery is removed, the backup job shifts to the primary sub-battery (refer to "Battery Operation Scheme" on page 28.).

Application programs are permitted, through the system library, to monitor this primary sub-battery voltage and determine if it is lower than the warning level (LB2). However the system side will not issue a warning about the low voltage level (LB2) of the primary sub-battery. Therefore, the application program must monitor the primary sub-battery voltage via the system library and inform the user that it must be replaced.

For information about the method used to replace the sub-battery refer to Chapter 2.2.6, "How to Replace or Recharge Batteries".

SRAM Card Battery Voltage Monitoring

This function monitors the SRAM card battery voltage. Application programs are permitted, through the system library, to monitor this voltage and determine if it is lower than the warning level (LB3). However, the system side will not issue a warning about the low voltage level (LB3) of the SRAM card battery. Therefore, the application program must monitor the SRAM card battery voltage via the system library and inform the user that it must be replaced.

For information about the method used to replace the SRAM card battery refer to Chapter 2.2.6 "
How to Replace or Recharge Batteries".

Acquiring Power Status through APM BIOS

This terminal has APM 1.1 installed. This makes it possible for application programs to obtain information, such as the percentage of battery voltage remaining or the connector status, via the APM BIOS. For more information refer to Appendix C Acquisition of Suspend/Resume Event and Power Status.

Acquiring Power Status through Battery Status Acquisition Utility

With the battery status acquisition utility the user can be advised of the current remaining voltage of the main battery, sub-battery status, or connector status in real time. For more information refer to Chapter 8.7 "Remaining Battery Voltage Display Utility".

2.2.5 Low Consumption Current Process

This terminal has (1) the APM BIOS installed to provide a low-power consumption capability. It works in combination with POWER.EXE from Microsoft Corporation. The low-power consumption capability is further enhanced by the use of unique power management functions, including (2) Auto Power OFF (APO) function, (3) Auto Backlight OFF (ABO) function, and (4) DOZE/RUN transition function.

Advanced Power Management Process (APM)

The APM process, which is an interface between the hardware and application programs, has been developed by the Intel Corporation and Microsoft Corporation for power control purposes. APM consists of four layers. The layers include hardware, APM BIOS, APM Driver, and the application, as shown below. With respect to the PC card which is a removable device, the APM functions are provided from the specific APM driver (CS_APM.EXE).

Applications Layer	APM-aware Applications	PC Card APM Driver (CS_APM.EXE)
OS Layer	APM Driver (POWER.EXE)	
BIOS Layer	APM BIOS	
Hardware Layer	APM BIOS Controlled Device	PC Card (Add-in Device)
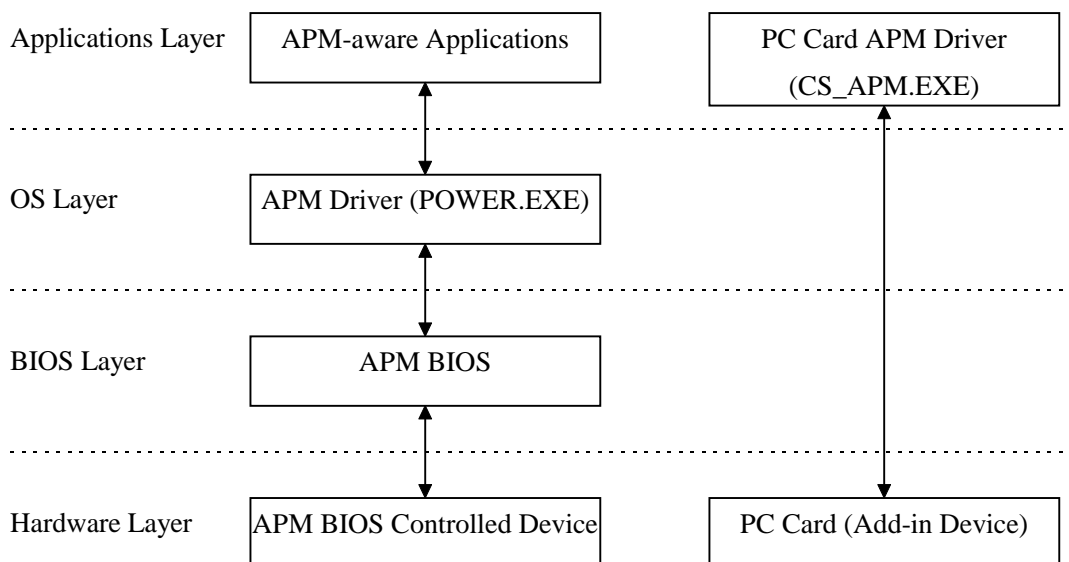
Fig. 2.7

Basically, APM functions in the following two ways:

● APM BIOS, which is in the background, controls the power conditions of each device.

● Applications can call the APM BIOS functions to obtain or control the power conditions. An application that uses the APM BIOS function is called an APM-aware Application. If an application acquires information related to power conditions via the system library (refer to Chapter 7.6.2 "System Library"), APM BIOS is actually called within the system library. It is also possible to directly call APM BIOS from applications. For more information refer to an APM BIOS manual published separately by a third party.

Auto Power OFF Function (APO)

This function automatically shifts the system to the OFF state (suspend state) if no event has taken place for a specified period of time from the touch panel, the keyboard, COM1, or a file.

This time interval has been set to one minute by default. It can be modified using the System Menu or system library.

About the activity

Any access to the touch panel, key, COM1, or file that causes results in Auto Power OFF is defined as "an activity", and it is said that "an activity occurs" if one of these devices is accessed. In other words, the Auto Power OFF function can be said to have shifted the system to the suspend state if no activity has occurred for a specified period of time.

The term "activity" is also used in the later description of the ABO function, but it has a different meaning.

Activity monitored by APO:

- Touch panel input
- Key input
- Access to files
- Access to COM1

Auto Backlight OFF Function (ABO)

This function automatically turns off the backlight if no access to the touch panel or keys has been attempted for a specified period of time. This time interval has been set to twenty seconds by default. It can be modified using the System Menu or system library. Touch panel or key sensing is performed by the keyboard controller. This keyboard controller not only processes key input or touch panel input, but it also simultaneously detects activity while executing various background processes. Consequently, the limit value set as the Auto Backlight OFF time will not be accurate down to the seconds. The accuracy of this setup value is ± 10 percent.

Activity monitored by ABO:

- Touch panel input
- Key input

DOZE/RUN Transit Function

On this terminal the system will reduce the clock speed of the built-in CPU if no activity (access to the touch panel, keys, COM1, or file) has occurred for a specified period of time (four seconds). The state in which the CPU clock speed has been reduced is called the "DOZE state" and the state in which the CPU is operating at full speed is called the "RUN state". If an activity occurs in the DOZE state, the system returns to the RUN state. The DOZE/RUN transit function automatically switches between the DOZE and RUN states.
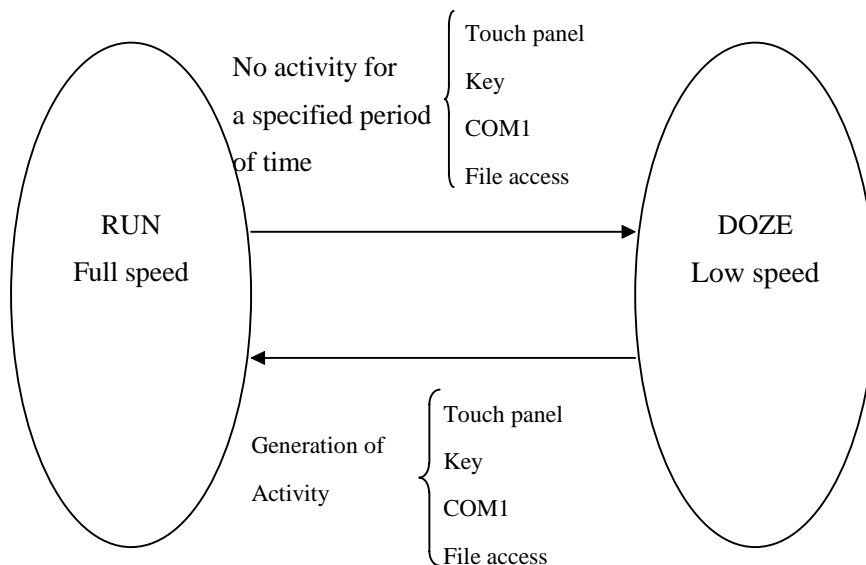
Fig. 2.8

Usually, application programs do not have to anxious about the RUN/DOZE state.

The user may tolerate the operation speed since the system shifts to the RUN state whenever the user attempts an action.

However, the clock speed is quickly reduced and CPU operation is slow if high-speed processing is attempted intentionally or if system operation continues without user action (e.g. in a long calculation).

In order to avoid this, disable the power management function by means of the system library (refer to Chapter 7.6.2 "System Library".).

Activity causing RUN/DOZE transition:
- Touch panel input
- Key input
- Access to files
- Access to COM1

Note:

If the power management function is disabled by the system library, the Auto Power OFF function (APO) is also disabled. This is because both the power management function and Auto Power OFF function use the same activity processing routine.

2.2.6 How to Replace or Recharge Batteries

Replacement of Batteries

The method used to replace the main battery, sub-battery, and SRAM card battery are explained here. Failure to observe the correct battery replacement procedure may result in a loss of data. Always observe the following steps in battery replacement.

Main battery replacement
- Hold down the Power switch for more than one second to turn off the main unit power.
- Make sure that two sub-batteries are installed, then open the battery compartment lid.
- Replace the fully charged main battery, the close the battery compartment lid.

Note:
Make sure that both sub-batteries are installed. If either of the sub-batteries is not installed, the data may be lost.
Do not open the battery compartment lid while the power is on. If it is opened accidentally, an emergency alarm sounds. In case such the event occurs, close the lid at once.

Sub-battery replacement
- Hold down the Power switch for more than one second to turn off the main unit power.
- Make sure that the fully charged main battery is installed.
- Replace the primary sub-battery (button-type lithium battery) with a new one.

Note:
Make sure that the main battery is installed. If the primary sub-battery is removed without the main battery being in place, data will be lost.
The secondary sub-battery (button-type lithium-vanadium battery) does not have to be replaced.

SRAM card battery replacement
- Make a backup of the SRAM card on the F-ROM drive or on some other device.
- Remove the SRAM card from the PC card slot of the main unit.
- Replace the battery of the SRAM card.
- Insert the SRAM card into the PC card slot.
- If the data has been lost, format (refer to Chapter 2.3.6 "PC Card") the SRAM card then restore the data on it from the backup device.

Note:

The SRAM card is supplied power by the main battery when it is installed in the main unit. This means that the SRAM card can be used normally as long as it is in the slot, even if the voltage of the card battery is zero.

In this case, however, the data on the SRAM card will be lost when the card is removed from the main unit slot. Since the Casio SRAM card is provided with two batteries, the data will not be lost for a short while even if one of them is removed. However, it is recommended that the SRAM card battery be replaced only after making a backup of the data to avoid accidental loss.

Main Battery Recharge

The main battery can be recharged using either of the following methods:

● Recharging with the charger

　According to the "Main battery replacement" procedure described on the previous page, remove the main battery and place it on the charger.

● Recharging with the AC adaptor

　While keeping the main battery to be recharged in the main unit, insert the AC adaptor plug in the charging jack located on the side of the main unit. This starts the recharging operation.

● Recharging with the I/O Box

　If the main unit is mounted on the I/O Box while the main battery to be recharged is in the main unit, the charging operation starts.

2.3 Supported Devices

2.3.1 Display Unit

Hardware Configuration

LCD	FSTN semi-transparent liquid crystal display
Resolution	192 x 384 dots
Tone	B/W 16 gray scales (4 gray scales are identifiable)
Method	VGA compatible
VRAM	512 KB
RAM for hardware window	32 KB

Note:

With B/W liquid crystal displays the actual display colors will be changed to reverse video.

About the Display Screen

Since this terminal has a VGA controller, it can internally control the entire VGA (640 x 480 dots) screen. However, only the 192 x 384 dots, which corresponds to the upper left portion of the VGA screen, can be displayed.
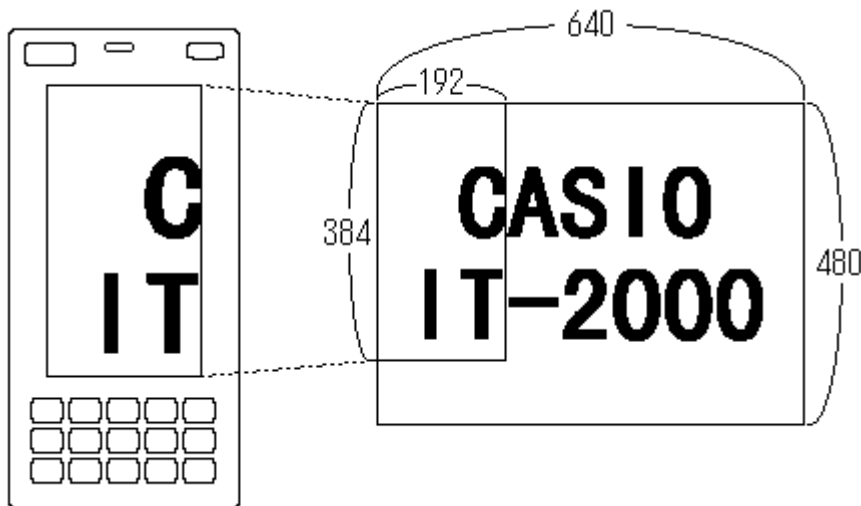
Fig. 2.9

Software Functions

Standard Video BIOS is supported. This supports the following video modes:

Mode No	Mode Type	Characters	Resolution	Colors	Memory Segment
00h	Text	40 x 25	320 x 200	16	B800h
01h	Text	40 x 25	320 x 200	16	B800h
02h	Text	80 x 25	640 x 200	16	B800h
03h	Text	80 x 25	640 x 200	16	B800h
04h	Graphics		320 x 200	4	B800h
05h	Graphics		320 x 200	4	B800h
06h	Graphics		640 x 200	2	B800h
07h	Text	80 x 25	640 x 350	2	B000h
0Dh	Graphics		320 x 200	16	A000h
0Eh	Graphics		640 x 200	16	A000h
10h	Graphics		640 x 350	16	A000h
11h	Graphics		640 x 480	2	A000h
12h	Graphics		640 x 480	16	A000h

Hardware Window

The hardware window provides the superimpose function for the VGA controller.

With this hardware window a pop-up screen can be displayed without affecting the operation of the application program. This hardware window is used in the keypad driver and various utility programs.

Contrast Adjustment

The contrast of the liquid crystal display automatically compensates for temperature changes.

The user can adjust the offset value (refer to Chapter 5 "Keyboard Controller") for the automatically adjusted contrast in the following ways.

● Press the 8 key after the Fn key to increase the contrast offset one step.

● Press the 9 key after the Fn key to decrease the contrast offset one step.

● Call the system library to increase/decrease the contrast offset.

2.3.2   EL Backlight

Overview

This terminal has the following functions to control the backlight. For more information refer to Chapter 5, "Keyboard Controller".

● Manual Backlight ON/OFF function

● Auto Backlight OFF function (ABO)

● Auto Backlight Control function (ABC)

Manual Backlight ON/OFF Function

The backlight can be turned on and off with the following methods.

● Press the 7 key after the Fn key to turn on or off the backlight.

● Call the system library to turn on or off the backlight.

Auto Backlight OFF Function

This function automatically turns off the backlight when no key or touch panel input has been occurred in the specified period of time. The time interval until the backlight is automatically turned off can be set with the System Menu or the system library.

Auto Backlight Control Function

This function detects the intensity of ambient light and automatically turns on or off the backlight accordingly. This function is set to off by default, however, it can be set to on using the System Menu or system library. For more information about the system library refer to Chapter 7.6.2 " System Library".

2.3.3 Touch Panel

Hardware Configuration

Method   : Analog type touch panel
Resolution   : 192 x 384 dots

Software Function

To enable application programs to acquire touch panel coordinates, the following two pieces of software are provided:

● PENMOUSE.COM

With this PENMOUSE.COM application programs can acquire touch panel input through the mouse I/F. (refer to Chapter 6.5 "PenMouse Driver".)

● KEYPAD.EXE

With this keypad driver application programs can perform character input through the touch panel. However, it cannot be used concurrently with PENMOUSE.COM. (refer to Chapter 6.4 "Keypad Driver / Hardware Window Manager".)

2.3.4 Disk

Types of Disk

Type	Drive name	Capacity
RAM disk	A	0 to 1920 Kbytes
Basic drive	C	768 Kbytes
F-ROM disk	D	0, 4, 8, 12, 16 or 24 Mbytes
PC card	G or F	SRAM card, ATA card

Note:

The drive name of the PC card varies for each model. For more information refer to Chapter 2.1.3 "Drive Configuration".

RAM Disk

Part of the main RAM can be assigned on the RAM disk using System Menu.

The contents of the RAM disk will not be erased if the Power switch is turned on and off, since they are backed-up by the main battery and the sub-batteries. The contents of the RAM disk are not affected by pressing the RESET switch either. Since this RAM disk permits the use of INT13h, it can be used as the built-in fixed disk. Its drive name is A.

Note:

Since the RAM disk shares part of the main memory installed in the main unit, a large-size RAM disk may affect the operation of application programs.

Basic Drive

Part of the DINOR FLASH ROM is used as the basic drive. It cannot be written to.

Disk capacity : 768 KB

Since the basic drive supports the INT13h (Read Only) interrupt, it can be used as the built-in fixed drive. Its drive name is C.

F-ROM Drive

The F-ROM drive is supported as a disk for which both read and write operations are possible (only for models with the F-ROM drive). Various disk capacities are supported for each model:

Disk capacity: 0 (models without F-ROM), 4M, 8M, 12M, 16M or 24 MB

To format the F-ROM drive use the System Menu. For information about the formatting method refer to Chapter 3 "System Menu". In this process the System Menu will call TFORMAT.EXE from drive (C:) to format the F-ROM drive.

For more information about the TFORMAT.EXE operation refer to Appendix A TFORMAT.

Since this F-ROM drive supports the INT13h interrupt, it can be used as the built-in fixed drive. Its drive name is D.

PC Card Drive

If either an SRAM card or ATA F-ROM card is inserted in the PC card slot, it can be used as the drive G (Drive F for models without the F-ROM drive). If the ATA F-ROM card is inserted in the card slot, the system can boot up according to the CONFIG.SYS/AUTOEXEC.BAT files included on this card. This start-up method is called "card boot".

For more information about card boot refer to Chapter 4.3 "Card Boot".

2.3.5 Serial Communication

Available Interfaces

Port	I/O Address	Name	Uses	Remark
COM1	3F8h-3FFh	8-pin serial I/F	Connection with a barcode reader or PC	
COM2	2F8h-2FFh	14-pin serial I/F	Connection with an expansion I/F device	Can be switched via the system library.
		IrDA 1.0	Communication with an I/O Box or between two IT2000s	
COM3	3E8h-3EFh	(Modem card)	Modem card	If a modem card is used.
COM4	2E8h-2EFh	IrDA 1.1	Communication with an I/O Box or between two IT2000s	Direct control not possible

COM1

This is a COM port for RS-232C communication. This port can be used after turning on the power to the 8-pin serial I/F via the system library. The 8-pin serial I/F is located on the side panel of the main unit.

Signal		Pin 1. SD
		Pin 2. RD
		Pin 3. RS
		Pin 4. CS
		Pin 5. Vcc
		Pin 6. GND
		Pin 7. ER
		Pin 8. DR
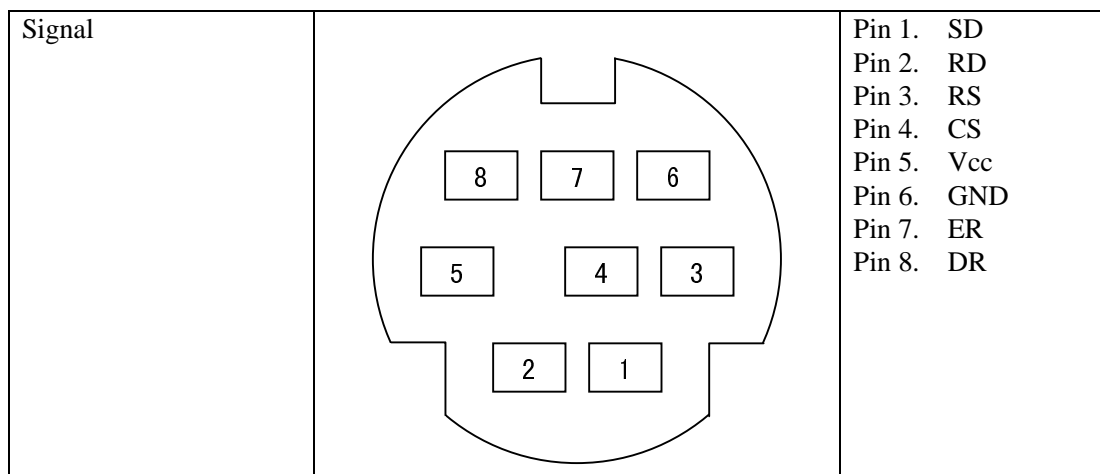
Fig. 2.10

COM2

Either the 14-pin serial I/F or IrDA 1.0 can be assigned to this COM2 port depending on the system library setup. Both the 14-pin serial I/F and IrDA 1.0 can be used as a normal RS-232C interface. By default, the COM2 channel is not assigned to either device. Therefore, always use the system library to designate either the 14-pin serial I/F or IrDA, then turn on the power. The 14-pin serial I/F is located on the rear of the panel.

		Pin 1. GND
	8 9 10 11 12 13 14	Pin 2. GND
		Pin 3. N.C.
		Pin 4. SD
	1 2 3 4 5 6 7	Pin 5. RD
		Pin 6. RS
		Pin 7. ER
		Pin 8. CS
		Pin 9. CI
		Pin 10. DR
		Pin 11. CD
		Pin 12. EXTSW
		Pin 13. VH
		Pin 14. VH
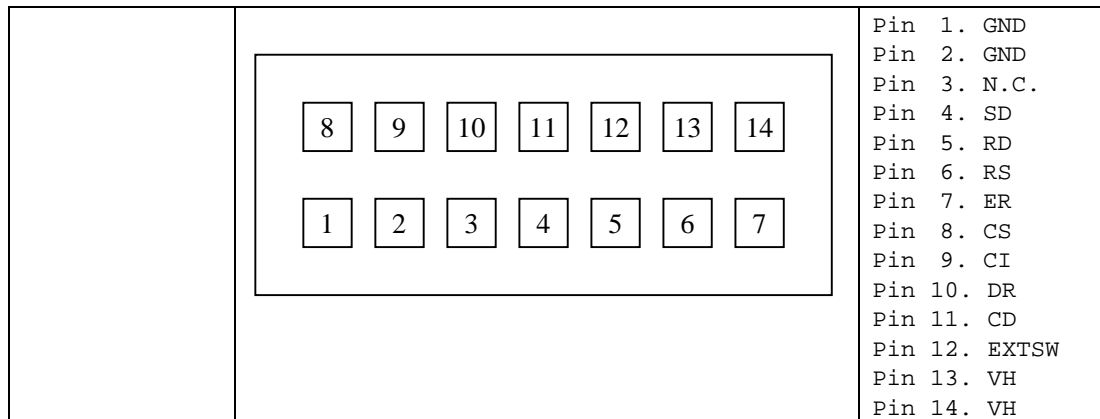
Fig. 2.11

COM3

A modem card, if one is inserted in the PC card slot, can be used as the COM3 port.
(refer to Chapter 2.3.6 "PC Card".)

COM4

The COM4 port is dedicated for IrDA 1.1. It is used internally by the FLINK utility. Therefore, it cannot be directly controlled by application programs.

2.3.6　PC Card

Hardware Overview

Standard	Conforms to PCMCIA Release 2.1
Register compatibility	Has register compatibility with Intel 82365SL Step
Slot	1 slot TYPE II
Power supply	Vcc : 5V (not operable at 3.3V)
Card lock switch	Has a card lock switch

Recommended PC Cards

Type	Model name
SRAM card	DT-635MC (256 KB) DT-636MC (512 KB) DT-637MC (　1 MB)
ATA Flash ROM card	DT-9031FMC (2.5 MB) DT-9032FMC (5　 MB) DT-9033FMC (10 MB) DT-9034FMC (20 MB)

How to Format SRAM Card and ATA F-ROM Card

To format, call FORMAT.COM in the basic drive (C:). Then, in the DOS prompt screen that appears, execute the following command to format the SRAM card or ATA F-ROM card. FORMAT.COM can also be called as a child process.

　　　FORMAT G:

COM Port of Modem Card

COM Port	COM3
IRQ	11
I/O Address	3E8h to 3EFh

Notes:

● This port is not applicable for a 3.3V card, CardBus, or a ZV port.

● Neither turn off the power nor remove the card while accessing the card. If this is done, system operation becomes unstable.

● Before using each type of PC card the PC card driver should be installed by means of the CONFIG.SYS file. For information about the method used to write CONFIG.SYS refer to Chapter 4.2 "How to Write CONFIG.SYS/AUTOEXEC.BAT".

● If the PC card is inserted in the slot and the card is locked, a card recognition sound (buzzer) will be issued. Since the card is locked, a short period may be required until the recognition sound is actually issued. Therefore, it is necessary to confirm this recognition sound in advance if accessing to the card. An error may occur if the card is accessed before the recognition sound is issued.

Card Lock Switch

The IT-2000 has a card lock switch to prevent accidental removal of the card. Any card can be made usable only after it has been inserted in the slot and the switch has been locked properly. However, since some types of cards do not allow this card lock switch to be closed, a library routine to disable this switch is supported. For more information refer to Chapter 7.6.2 "System Library".

2.3.7 Clock Timer

Clock BIOS

00h to 07h of the INT1Ah function are compatible with the IBM PC/AT.

Since INT1Ah can be called in the C language, an alarm operation using the clock can be set with the system library.

Alarm

If an alarm operation is set using the INT1Ah or system library, it is possible to cause an INT4Ah interrupt at the specified time to issue the alarm. Normally a buzzer sounds if an INT4Ah occurs, however, the application program side can hook this interrupt and perform its unique alarm process. It is also possible to automatically turn on the power at the specified alarm time by means of the system library (refer to Chapter 7.6.2 "System Library").

2.3.8 Buzzer

This terminal is provided with a buzzer function that is compatible, via an appropriate interface, with the IBM PC. The application side can sound this buzzer by controlling the I/O port assigned to 61h. It is also possible to modify the sound frequency by controlling channel 2 of the timer.

For information about the method used to modify the frequency refer to the hardware manual of the PC/AT compatible machine.

Use of Buzzer From the System

The IT-2000 system uses the buzzer in the following cases:

- At power on (boot)
- If the power is turned off by the Power switch.
- If the PC card is inserted/removed
- If a key input is accepted (for matrix key and keypad). Enable/disable can be set with the system library. (refer to Chapter 7.6.2 "System Library")
- If the key buffer is full
- At a low battery voltage (LB1)
- If an alarm interrupt (INT4Ah) occurs
- If the battery cover is opened while the power is on.
- At a hardware anomaly
- For calibration and System Menu operation

Setting Volume of Buzzer

The buzzer volume can be set with System Menu or from the system library.

The volume can be set to one of the four levels: OFF/Small/Medium/Large.

For more information about System Menu and the system library refer to Chapter 3 "System Menu" and Chapter 7.6.2 "System Library" respectively.

2.3.9   Barcode Reader

Overview

The IT-2000 supports the following two Casio OBR (Optical Barcode Reader) models:

DT-9650BCR (Pen scanner)

DT-9656BCR (CCD scanner)

Connect the OBR to the COM1 (8-pin) port of this terminal, and set up the interface as follows.

Synchronization	asynchronous
Baud rate	9600 bps
Data bit	8 bits
Parity bit	none
Stop bit	1 bit

For communication between the OBR and this terminal use the OBR library. The various settings such as an objective readout codes can be set up by transmitting the set up commands from this terminal to the OBR.

Notes:

● The OBR power is controlled by the OBR library function.

● Before connecting the OBR to this terminal, turn off the main power.

● Every OBR can write the current setup values in the EEPROM built into each OBR.
  This ensures that the setup data is retained even when the power is off.

For more information about the OBR library, refer to Chapter 7.6.4 "OBR Library".

2.3.10 Infrared Communication (IR)

The infrared communication function of this terminal supports the protocol of IrDA 1.0
(see note below) and IrDA 1.1 standards. IrDA 1.0 can be used as the COM port for a general
RS-232C. IrDA 1.1 can provide communication at a maximum rate of 4 Mbps by means of
the dedicated utility (FLINK utility).

IrDA 1.0

Item	Specification	Remark
Synchronization	asynchronous	Conforms to IrDA1.0
Baud Rate	115.2 Kbps max.	
COM Port	COM2	

IrDA 1.1

Synchronization	Frame synchronization	Conforms to IrDA1.1 (see note below)
Baud Rate	4 Mbps max.	
COM Port	COM4	Cannot be controlled directly from the application.

Note:

The distance between the two ports must not be more than 60 cm (or 23.6 inches) apart.

2.3.11 Keys

Hardware Overview

Key configuration	5 (column) x 3 (row) keys
IRQ	IRQ1
Key repeat function	available
Simultaneous pressing of multiple keys	not available
Roll-over function	not available

Key Layout

See the following key layout.

Fig. 2.12

Fn key

The "Fn" key should be used in combination with the numeric key. Hold down the "Fn" key and press a numeric key.

Fn -> 0	Function key F10
Fn -> 1 to 6	Function key F1 to F6
Fn -> 7	Backlight on/off
Fn -> 8	Increase the contrast
Fn -> 9	Decrease the contrast

For more information refer to Chapter 5 "Keyboard Controller".

2.3.12 Sensors

The IT-2000 has the following three types of built-in sensors:

Illumination sensor	Attached to the upper section of this terminal and used to sense the ambient light intensity. It is used for the Auto Backlight Control (ABC) function. It cannot be controlled directly from the application. (For more information about the system library refer to Chapter 5 "Keyboard Controller".)
Temperature sensor	Attached to the inside of the main unit and used to detect the ambient temperature. It is used for Automatic Brightness Adjustment (ABA) of the liquid crystal display. It cannot be controlled directly from the application. (For more information about the system library refer to Chapter 5 "Keyboard Controller".)
Battery voltage level sensor	Detects the voltage levels of the main battery, sub-batteries, and card battery. It is used by the system to take action against low battery voltages. The system manages low voltage through this battery electric potential sensor. Applications can acquire the information from this battery voltage level sensor via the system library or APM BIOS. (Refer to Chapter 2.2.4 "Battery Voltage Monitoring Process".)

3.   System Menu

3.1   Overview

The system menu is a program and used to perform various setups (system clock, contrast of liquid crystal display, etc.) and implement (downloading) application programs, all of which are necessary to use this terminal.

To start up the system menu press the reset switch located at the back of the main unit.

When the reset switch is released a short beep will sound and, after a short while, a screen as shown in Fig. 3.1 will appear.

The calibration (touch panel adjustment) program is initiated first and it must be executed before entering to the system menu selection stage. If this terminal is used for the first time or if the touch screen is out of line, adjust the touch panel using this calibration program.

(For information about adjusting the touch panel refer to Chapter 3.9 "Touch Panel Calibration")

If the " 1" key is pressed the system menu will be initiated as shown in Fig. 3.2.
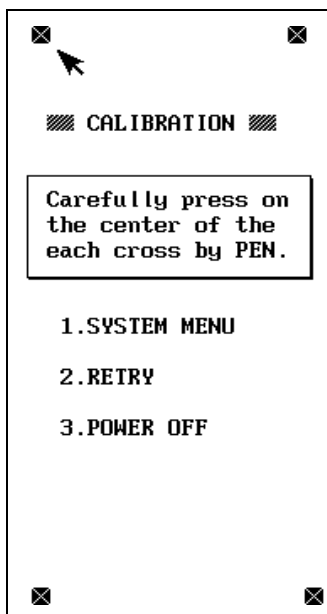
Fig. 3.1
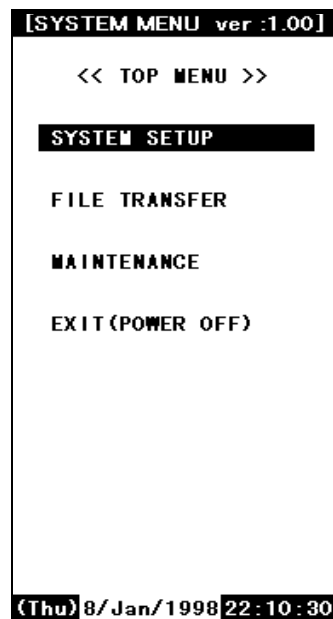
Fig. 3.2

3.2 Basic Operation

In the system menu a common set of key operations are used. The following list shows the keys that can be used in the system menu.

Current Condition	Key Operation	Operation Process
Line cursor is on	8	Moves the line selection cursor up one line.
	2	Moves the line selection cursor down one line.
	CLR	Moves the line selection cursor to the upper menu area, if it is located in the lower menu area.
	RET	Confirms and executes the currently selected menu item.
Others	0 to 9	Selection of an item or entry of a numeric value.
	RET	Confirms the currently selected execution item.
	CLR	Cancels the currently selected execution item.

If "FILE TRANSFER" or "MAINTENANCE" is selected for the first time after the system menu is initiated, the operator is required to enter a password for system security purposes. For information about password entry refer to Chapter 3.17 "Password Entry".

3.3 List of Functions

Command Screen		Description
SYSTEM SETUP	Key Click Sound	Switch ON or OFF the key click sound.
	Buzzer Volume	Set volume of buzzer.
	LCD Contrast	Adjust the brightness of contrast.
	Auto Backlight	Set the control of auto backlight.
	Auto Power OFF	Set auto power off.
	Calibration	Adjust the calibration on touch panel.
FILE TRANSFER (requires password)	Ymodem Batch	Start up the YMODEM utility.
	FLINK (IrDA)	Start up the FLINK utility.
MAINTENANCE (requires password)	Set Date/Time	Set date and time.
	MS-DOS Command	Set the command entry mode.
	RAM Disk Size	Change the size of RAM DISK.
	Format Disk	Format on user disk.
	Default Setting	Start up the system initialization.
EXIT (power off)		

For information about each function in the list above refer to the following pages.

3.4 Key Click Sound Setup

Function

Sets the key click sound ON and OFF. If it is set to ON, a key click sound is heard when a key is pressed or when the keypad is touched. It does not sound if it is set to OFF.

```
[Key Click Sound]
Key Click Sound :
    1:ON / 0:OFF
```

Fig. 3.3

Operation

Select ON/OFF with the " 0" or " 1 " key, then confirm the selection with the "RET" key.

Key Operation	Function
0 key	Sets the key click sound to OFF.
1 key	Sets the key click sound to ON.
. (decimal) key	Toggles to ON and OFF of the key click sound.
RET key	Confirms the current setup and exits the current operation.
CLR key	Cancels the setup and exits the current operation.
Others	Invalid.

3.5   Buzzer Volume Setup

Function

Sets the volume of the buzzer (beep). One of the four levels (OFF/Small/Medium/Large) can be selected.

```
[Buzzer Volume]

Buzzer Volume
    0:OFF
    1:SMALL
    2:MIDDLE
    3:LARGE
```
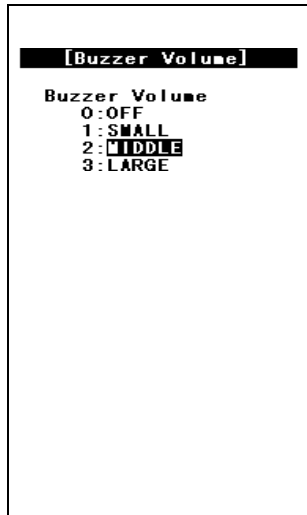
Fig. 3.4

Operation

Make a selection with a key, " 0" to " 3 ", and confirm the selection with the " RET" key.

Key Operation	Function
0 to 3 keys	Selects the corresponding number.
. (decimal) key	Toggles between two selections.
RET key	Confirms the currently selected setup and exits this operation.
CLR key	Cancels the currently selected setup and exits this operation.
Others	Invalid.

3.6   Contrast Adjustment

Function

Adjusts the contrast of the liquid crystal display.

```
[LCD Contrast]

LCD Contrast :

    8 : UP
    2 : DOWN
```
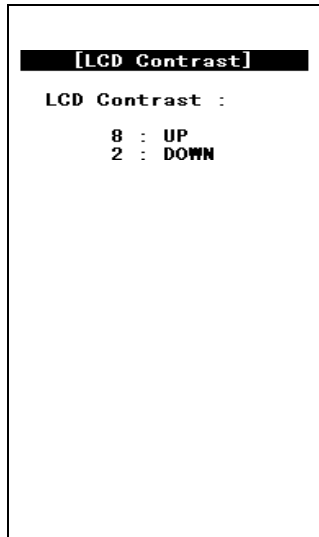
Fig. 3.5

Operation

Press the "8 " key to increase the contrast or press the " 2" key to decrease the contrast.

Press the " RET" key to confirm the setting.

Key Operation	Function
8 key	Increase the contrast.
2 key	Decrease the contrast.
RET key	Confirms the currently selected contrast setup and exits this operation.
CLR key	Cancels the currently selected contrast setup.
Others	Invalid.

Note:

Depending on whether the parameters are being modified, the CLR key activates differently.

For example, if the CLR key is pressed while a parameter is being changed, that parameter will be reset to the previous value.

However, if the CLR key is pressed while no parameter is being changed, the setup process will be aborted and exited at that point.

3.7   Auto Backlight Setup

Function

Sets the auto backlight control ON or OFF (refer to Chapter 5 "Keyboard Controller").

[Auto Backlight]

Auto Backlight :
    1:ON / 0:OFF
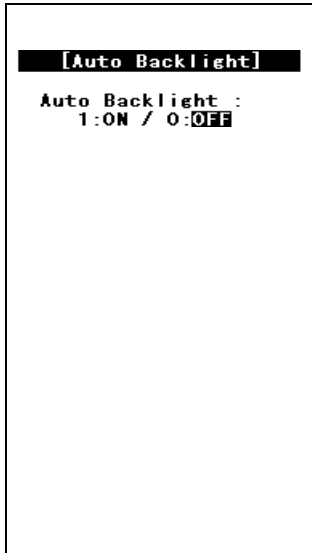
Fig. 3.6

Operation

Select ON/OFF with the " 0" or " 1" key, then confirm the selection with the "RET" key.

Key Operation	Function
0 key	Turns the auto backlight control to OFF.
1 key	Sets the auto backlight control to ON.
. (decimal) key	Toggles to ON and OFF of the auto backlight control.
RET key	Confirms the current setup and exits this operation.
CLR key	Cancels the current setup and exits this operation.
Others	Invalid.

3.8 Auto Power OFF Setup

Function

Sets the time-out period of the auto power off function (APO) (refer to Chapter 2.2.3 "Power OFF Process"). This time-out period is the interval between when no key entry or no entry on the touch panel is made and when the power of system is shut off automatically.

```
[Auto Power OFF]

APO Time : 01min 30sec

        8 : UP
        2 : DOWN
```
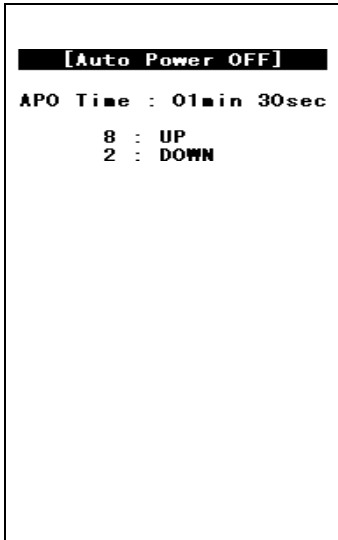
Fig.3.7

Operation

Set the APO time out period with the " 2" or " 8 " key, then confirms the setting with the " RET" key.

Key Operation	Function
8 key	Increase the APO timeout period.
2 key	Decrease the APO timeout period. If "DISABLE" appears, the APO function is disabled.
RET key	Confirms the current setup and exits this operation.
CLR key	Cancels the current setup and exits this operation.
Others	Invalid.

3.9   Touch Panel Calibration

Function

Adjusts the calibration of touch panel. If an inconsistency is noted between the target position
and the position actually touched on the touch panel, correct it by performing this calibration
adjustment.
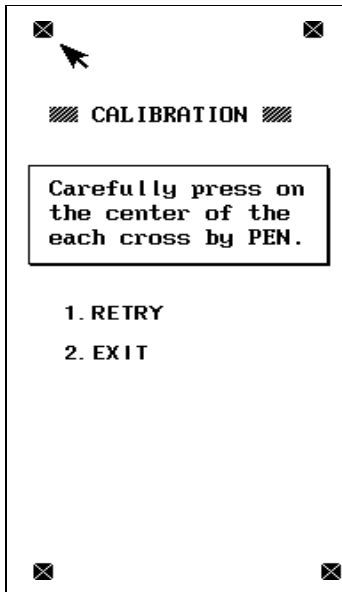
Fig. 3.8

Operation

Adjustment of the calibration :

- First make sure that the arrow points to ▨ in the upper left corner of the screen, then touch the center of this ▨ with the stylus provided.
- When the buzzer sounds, release the stylus from the touch panel.
- After the ▨ in the upper left corner disappears and the arrow moves to the (x) in the upper right corner, touch it in the same way.
- Do the same for the s in the lower left and lower right corners.
- When all four ▨s are touched by the stylus, the touch panel calibration is completed.
  If any improper operation has been done, press the "1" key to perform the touch panel

  calibration again.

- If the "2" key is pressed after the four positions have been touched, the calibration

  adjustment result takes effect and the menu screen is restored. However, if the " 2 " key is

  pressed before finishing on the fourth position, the adjustment process performed so far will

  be canceled.

Note:

If an ▨ mark does not disappear and the arrow does not move to the next position even if the ▨ mark has been touched by the stylus, an incorrect position was likely touched. Touch the correct position.

Key Operation	Function
1 key	Adjusts the touch panel calibration starting from the beginning.
2 key	Returns to the menu screen.
Others	Invalid

3.10   YMODEM Utility

Function

Used to achieve a file transfer via the COM cable.

Communication can be established either between an AT-compatible machine (PC) and an
IT-2000 (main unit), referred to as "PC-to-HT communication". A dedicated 9-pin DSUB-8-pin
cross-type cable (DT-9689AX) is required to connect both the terminals. This utility does not have
functions to allow communication between HT and HT. Use the FLINK function for the HT-to-HT
communication.

Fig. 3.9

Fig. 3.10
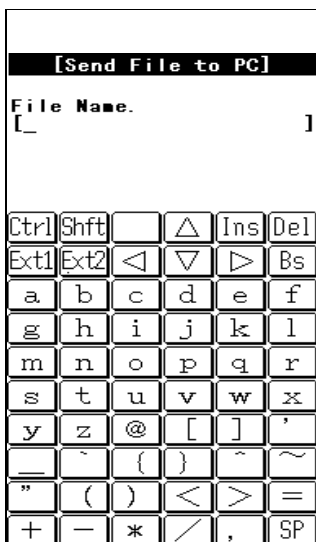
Fig. 3.11
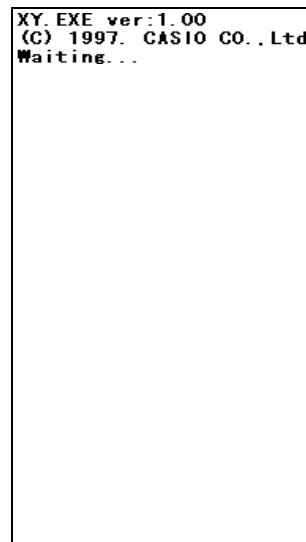
Note:

When the cable comes off while the communication takes place:

If the connection cable is accidentally unplugged while communication between the IT-2000 and PC is taking place, a communication error results and communication is interrupted. In this case the communication software on the PC will display an error message and interrupt transmission/reception, however, some data may remain in the transmission buffer. If an attempt is made to restart communication in this condition, the XY utility may receive illegal packets, hampering normal communication. If this occurs, terminate the communication software on the PC side then restart it to restore normal communication.

About time stamping of files:

This utility supports the function to exchange time stamp information between the transmitted file and received file. The time stamp information to be exchanged will be processed assuming that it is Greenwich standard time. In contrast, the time used by the IT-2000 is the local time, and the time stamp of IT-2000 files are accordingly controlled based on the local time.

The XY utility, for file transmission/reception by means of the YMODEM protocol, will convert a time stamp in Greenwich standard time to a time stamp in local time, or vice versa. This time conversion is achieved according to the environment variable, TZ. In communication between two IT-2000 terminals, if, for example, TZ of the transmission side is "JST+5", the time stamp of a file to be transmitted will have five hours added. In this case the reception side will create a file by subtracting five hours from the time stamp of the received file.

If the environment variable TZ is not set, this time conversion is not performed.

The time stamp made at XMODEM communication uses the system time of the reception side.

Transmission side   Reception side

IT-2000(TZ=none)	12:00 →	±0	→ 12:00 →	±0	→ 12:00	IT-2000(TZ=none)
IT-2000(TZ=GMT)	12:00 →	±0	→ 12:00 →	±0	→ 12:00	IT-2000(TZ=GMT)
IT-2000(TZ=JST+5)	12:00 →	+5	→ 17:00 →	-5	→ 12:00	IT-2000(TZ=JST+5)
IT-2000(TZ=JST+5)	12:00 →	+5	→ 17:00 →	?	→ ??:??	PC
PC	12:00 →	?	→ ??:?? →	-5	→ (??-5):??	IT-2000(TZ=JST+5)

About key input during communication

Do not press any key during communication, otherwise file transmission/reception may be hampered.

Operations

(1) SEND FILE TO HT (one file transmission from IT-2000 to IT-2000)
This function may be available in future (as of now, not available). It is not allowed to use the function. If the file transmission between IT-2000s is needed, FLINK utility may be used (refer to Chapter 3.11 "FLINK Command".).

(2) SEND FILE TO PC (one file transmission from IT-2000 to PC)
This function is used to copy an optional file from an IT-2000 to PC. To do this, use commercial terminal emulation software on the PC side. The destination directory of this copy should be specified by the terminal emulation software on the PC side.

- Connect one end of the serial cable (cross-type) to the 8-pin COM port of the IT-2000 and connect the other end to the COM port of the PC.
- Select "SEND FILE TO PC" on the transmission side.
- On the PC side initiate the terminal emulation software to prepare for download. Select a baud rate of 9600 bps, and specify the YMODEM Batch protocol.
- When the file name input screen appears on the IT-2000 side, specify the transmitted file with its full path name (including the drive name), then press the "RET" key.
- Pressing the "RET" key starts file transfer. When the "Normal End" message is displayed, file transmission has been completed.
- If the "CLR" key is pressed during file transfer, transfer will be interrupted. It will take about 10 seconds for communication to completely stop.

(3) SEND ALL TO HT (transfer all files in the user drive of IT-2000 to IT-2000)
This function may be available in future (right now, not available). It is not allowed to use the function. If the file transmission between IT-2000s is needed, FLINK utility may be used (refer to Chapter 3.11 "FLINK Command".).

(4) RECEIVE FILES (file reception)
The function is used to receive one file from the PC.
For information about the method used to receive a file from the IT-2000, refer to the "SEND FILE TO HT" description. On the PC side commercial terminal emulation software must be used. In this operation the copy destination directory cannot be specified.

- Connect one end of the serial cable (cross-type) to the 8-pin COM port of the IT-2000 and connect the other end to the COM port of the PC.
- Move the cursor to " RECEIVE FILES ", then press the " RET" key to prepare for reception.
- Start upload with the terminal emulation software on the PC side. Select a baud rate of 9600 bps, and specify the YMODEM Batch protocol.

- When the "Normal End" message is displayed on the IT-2000 side, file reception has been completed. For information about the copy destination directory refer to the following table.
- If the "CLR" key is pressed during communication, file reception will be interrupted. It will take about 10 seconds for communication to completely stop.

The destination drive/directory will vary depending on whether the destination side has an FROM drive (D:) and/or RAM disk (A:). The following table shows the possible destination drive/directory for copy purposes.

FROM drive (D:)	RAM disk (A:)	Copy destination drive/directory
Installed	Installed	FROM drive (D:)
	Not installed	FROM drive (D:)
Not installed	Installed	RAM disk (A:)
	Not installed	Error

3.11   FLINK Command

Function

Files can be transferred by infrared communication (IR). This can be implemented either as PC-to-HT (AT-compatible machine to IT-2000) communication or as HT-to-HT (between two IT-2000 terminals) communication.

To perform PC-to-HT communication an I/O Box and a PC-side communication utility "LMDOS.EXE (for DOS)" is required.

Fig. 3.12
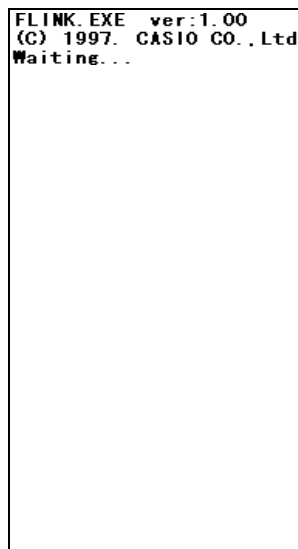
Fig. 3.13
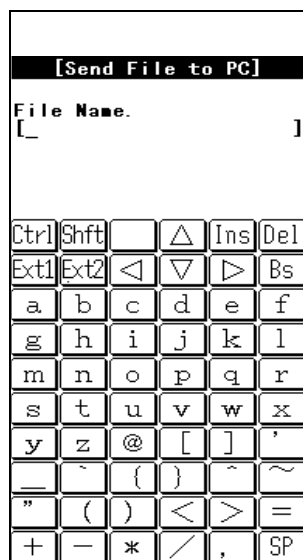
Fig.3.14

Note:

If the identical file name exists on the reception side, this command overwrites the existing file.

At this time, the system does not unconditionally overwrite the existing file but creates a temporary file on the reception-side disk and attempts the overwrite after file transmission has been competed. This protects file data even if transmission of the file fails.

Therefore, if the identical file name exists on the reception side, the reception-side disk must have a space large enough for the transmitted file. If there is insufficient space, either delete unnecessary files in advance on the reception side or use the file delete command (on page 204) to delete them.

Operation

SEND FILE (One file transmission from IT-2000 to IT-2000)

This function is used to copy one file from one IT-2000 to another IT-2000. This file will be copied to a destination directory that has a name that is identical to the source directory.

- Place the two IT-2000 units so that their IR windows face each other.
- Select "SEND FILE TO HT" at the transmission side.
- Select "REMOTE SERVER" at the reception side to prepare for reception.
- If the file name input screen appears at the transmission side, specify the transmitted file by its full pathname (including the drive name), then press the " RET" key.
- Press the"RET" key to start file transfer. If the "Normal End" message is displayed, file transmission has been completed.

Note:

If the [CLR] key is pressed during file transfer, transfer will be interrupted. It will take about 10 seconds for communication to completely stop.

SEND ALL to HT (Transfer of all files in the F-ROM drive of IT-2000 to IT-2000)

This function is used to mirror-copy the F-ROM drive. All files existing on the F-ROM drive of the copy source side are copied to the F-ROM drive of the destination side. Since this process does not attempt either file deletion or formatting on the copy destination side, it is necessary to confirm in advance that the F-ROM drive of the destination side has sufficient free space.

- Place the two IT-2000 units so that their IR windows face each other.
- Select "REMOTE SERVER" on the reception side to prepare for reception.
- On the transmission side move the cursor to "SEND ALL TO HT" and press the " RET" key. File transfer begins.
- If the "Normal End" message is displayed, file transmission has been completed.

Note:

If the "CLR" key is pressed during file transfer, transfer will be interrupted. It will take about 10 seconds for communication to completely stop.

REMOTE SERVER (remote server mode)

The remote server mode is used by the system which assigns the right of issuing a transmission request to the partner side and enters the wait state for a request from the partner.

To facilitate communication between two IT-2000 terminals, set the reception side to this mode.

For HT-to-PC communication set the IT-2000 side to this mode and perform the entire operation on the PC side.

● Move the cursor to "REMOTE SERVER" and press the " RET" key.

● If the "Hit Any Key!" message appears, file transmission has been completed.

Note:

If the "CLR" key is pressed during file transfer, transfer will be interrupted. It will take about 10 seconds for communication to completely stop.

About communication with PC

To achieve communication between a PC and IT-2000 it is necessary to prepare the I/O Box and the PC-side communication utility "LMDOS.EXE(for DOS)" or "LMWIN.EXE(for Windows)". The following procedure shows the steps required for communication with a PC.

● Connect the I/O Box and PC using a communication cable. Turn on the I/O Box power.

● Mount the IT-2000 on the I/O Box.

● Select "REMOTE SERVER" on the IT-2000 side to enter the wait state.

● On the PC side initiate the PC-side communication utility, LMDOS.EXE.

● Operate the PC-side communication utility to perform reception or transmission. For information about the operation of the PC-side communication utility refer to the "IT-2000 Upload/Download Utility Manual".

● If the "Hit Any Key!" message appears on the IT-2000 side, file transmission has been completed.

Note:

If the "CLR" key is pressed during file transfer, the transfer will be interrupted. It will take about 10 seconds for communication to completely stop.

3.12   System Date/Time Setup

Function

This is used to set (modify) the date and time of the built-in timer in the IT-2000 unit.

```
[Set Date/Time]

Date : 1998/01/08
Time : 10:10
```
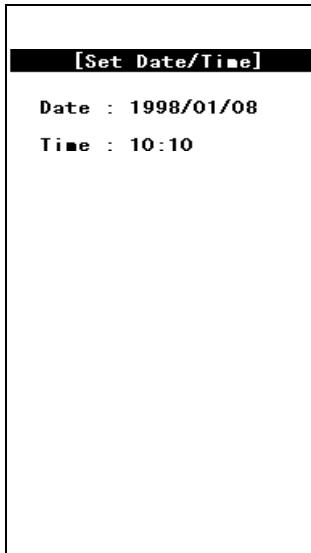
Fig. 3.15

Operation

Enter in the following order: year -> month -> day -> hour -> minute. Press a numeric key and the corresponding number will appear in the cursor position. Press the " RET" key to advance to the next setting. If the "RET" key is pressed without making a numeric entry, the cursor will advance to the next setting without affecting the previous value. If the " RET" key is pressed when the cursor is positioned on the minute setting, the current setup is confirmed.

Note that the seconds can not be specifically set. When the date and time is modified, the seconds will be set to 0. The year can be set to between 1980 to 2099. If the entered value includes an invalid number, the setup operation will result in an error when the entire entry has been completed. If this occurs, reenter from the beginning.

Key Operation	Function
0 to 9 keys	Enters the corresponding digit in the cursor position.
RET key	Moves to the next input item. When the cursor is in the minute setting, the current setup is confirmed.
CLR key	Cancels the currently selected setting and exits this operation.
Others	Invalid.

Operations on the touch panel are not permitted.

3.13   Command Prompt

Function

This is the MS DOS command prompt screen. An appropriate DOS command can be inputted through the keypad.

This DOS command prompt is the result of calling COMMAND.COM as a child process from the system menu. Consequently, if the EXIT command is entered, operation returns to the system menu.

```
Microsoft(R) MS-DOS(R) V
         (C)Copyrigh

C:\>_
```

```
Ctrl Shft      △  Ins Del
Ext1 Ext2 ◁  ▽  ▷  Bs
 a   b   c   d   e   f
 g   h   i   j   k   l
 m   n   o   p   q   r
 s   t   u   v   w   x
 y   z   @   [   ]   '
 _   `   {   }   ^   ~
 "   (   )   <   >   =
 +   −   *   /   ,   SP
```
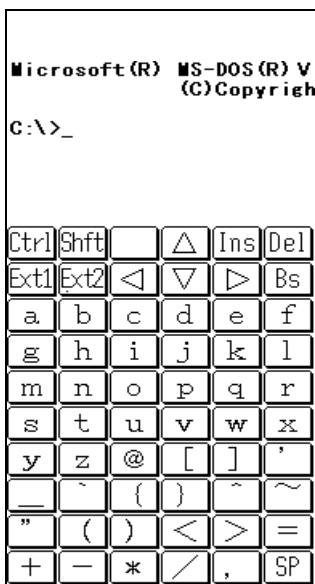
Fig. 3.16

3.14 RAM Disk Size Change

Function

This screen is used to set the RAM DISK size (capacity). The setting will become valid after the system has rebooted.

```
┌─────────────────────────┐
│  [Ram Disk Size]        │
│                         │
│ size : 0000 K bytes     │
│                         │
│      8 : UP             │
│      2 : DOWN           │
│                         │
│                         │
│                         │
│                         │
│                         │
│                         │
│                         │
│                         │
└─────────────────────────┘
```
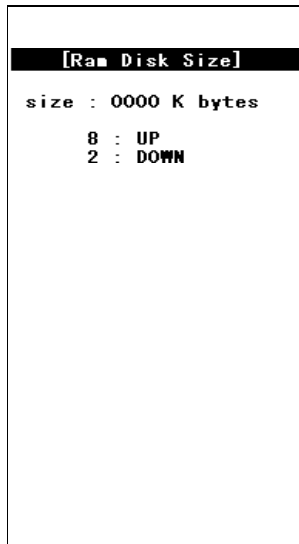
 Fig. 3.17

Operation

Setting up the RAM disk

● Adjust the RAM disk size with the "8" and "2" keys.

● Confirm the setup with the " RET" key.

● When the "Hit Any Key... " message is displayed, press any key other than the "Fn" key.

● The IT-2000 is turned off. After making sure that it turns off, press the reset switch to turn on the main unit again.

● After the IT-2000 is turned on again, the format confirmation screen, as shown below, will be displayed during system start-up. Then press the "1" key. This properly formats the RAM disk. After formatting the RAM disk is usable.

```
┌──────────────────────────────┐
│     RamDisk is broken.        │
│   Format?    YES:1/NO:0       │
└──────────────────────────────┘
```

Key Operation	Function
8 key	Increases the RAM disk size.
2 key	Decreases the RAM disk size.
RET key	Confirms the currently selected RAM disk size and exits this operation.
CLR key	Cancels the currently selected RAM disk size.

1 key	Formats the RAM disk (Format confirmation screen).
0 key	Aborts formatting of the RAM disk.
Others	Invalid.

Operations with the touch panel are not permitted.

3.15 Disk Format

Function

Formats the RAM disk or user drive.

Fig. 3.18

Operation

In the screen shown above, use the "2" or "8" key to select whether the RAM disk or user drive is to be formatted, then press the "RET" key.

This makes the following screen appear. In this screen press the "1" key to move the cursor onto "YES" and press the "RET" key to start formatting. If either the "RET" key is pressed while the cursor is on "NO", or "CLR" key is pressed while the cursor is on "YES", the formatting operation will be canceled.
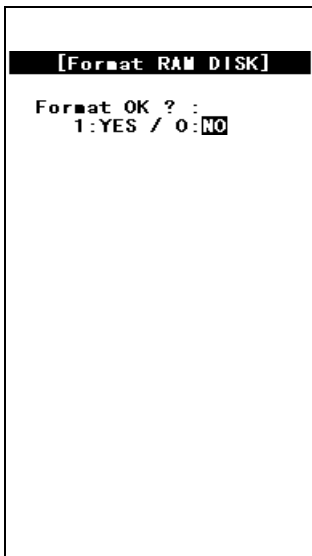
Fig. 3.19
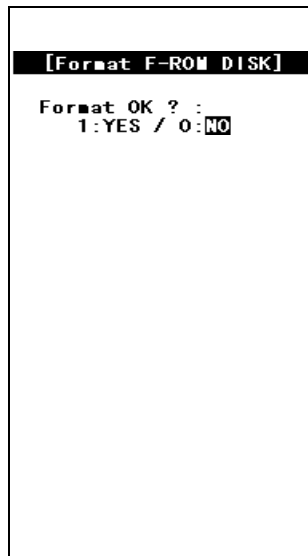
Fig. 3.20

Key Operation	Function
2 or 8 key	Selects the objective item (drive selection screen).
0 key	Does not perform formatting (formatting start screen).
1 key	Starts formatting (formatting start screen).
. (decimal) key	Toggles YES and NO options of formatting.
RET key	Confirms the current setting.
CLR key	Cancels the current setting.
Others	Invalid.

3.16 System Initialization

Function

Sets all the system setups to their defaults.

```
[Default Setting]

Set BIOS Default ? :
     1:YES / 0:NO
```
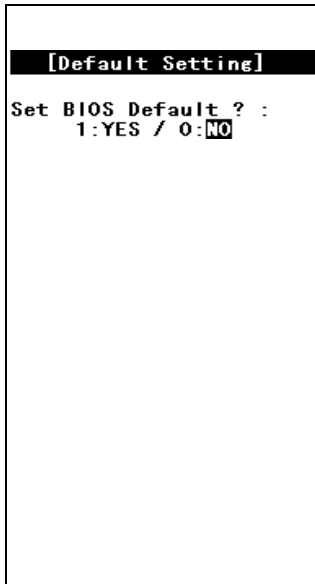
Fig. 3.21

Operation

The following table shows the available key operations.

Key Operation	Function
0 key	Does not initialize the system.
1 key	Initializes the system.
. (decimal) key	Toggles YES and NO options of initialization.
RET key	Confirms the current setting.
CLR key	Cancels the current setting and exits this operation.
Others	Invalid.

3.17 Password Entry

Function

When "FILE TRANSFER" or "MAINTENANCE" is selected for the first time after the system menu is initiated, the operator is requested to enter a password.
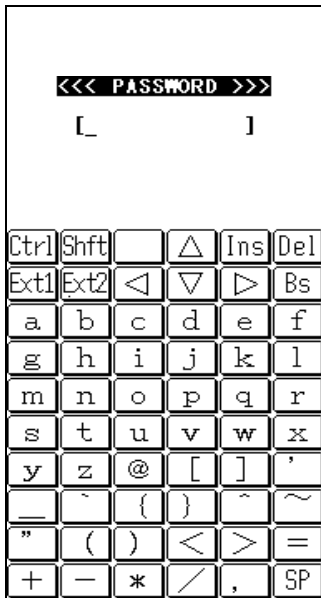
Fig. 3.22

Operation

With the keypad enter "system" (lowercase letter), then press the "RET" key. If the "CLR" key is pressed without entering a character, the password entry operation will be canceled. If the "CLR" key is pressed with characters having been entered, the characters entered so far will be canceled, and the password entry operation must be performed again.

This password will, if it is accepted once, be valid and will not have to be entered again unless the system menu is re-started.

Key	Function
RET key	Confirms the entry.
CLR key	Either clears or cancels the entered characters.
Others	Inputted as a character comprising the password.

Touch Panel	Function
BS key	Clears one character entered.
Arrow key INS key DEL key SP key	Invalid.
Others	Inputted as a character comprising the password.

4. MS-DOS

4.1 Overview

If a personal computer is booted-up with a floppy disk in the drive, first an attempt will be made to read MS-DOS from the floppy disk, and if a copy of MS-DOS does not reside on the floppy it is loaded from the hard disk (C:). However, this method cannot be used on this terminal since its basic drive (C:), which corresponds to the hard disk of a PC, is defined as a read-only device.

The MS-DOS on the boot drive (C:) can be loaded initially provided that no PC card is inserted in the slot, but, in this case, it is not possible to add the start-up code for user programs to the AUTOEXEC.BAT file. This problem is solved on the terminal as follows.

- At boot-up this terminal searches each drive to locate the CONFIG.SYS and AUTOEXEC.BAT files and sets it as the current drive, then MS-DOS is loaded into the main memory. As a result, the CONFIG.SYS and AUTOEXEC.BAT files in the current drive can be processed through MS-DOS.
- The CONFIG.SYS and AUTOEXEC.BAT files will be searched in the following order:
  [PC card drive] -> [RAM disk] -> [F-ROM drive] -> [Basic drive]
- The CONFIG.SYS and AUTOEXEC.BAT files on the basic drive will be executed only if the RESET button is pressed. As a result, the System Menu, which is the maintenance program for this terminal, will be initiated.

Since the main part of MS-DOS is always loaded from the basic drive (C:) in this case, it is not necessary to install MS-DOS and COMMAND.COM on the user drive.

Fig. 4.1

```
                        ┌─────────────────┐
                        │  LOADING MS-DOS │
                        └─────────────────┘
                                 │
         YES                     ◇                    NO
      ┌──────────────  RESET BUTTON was pressed?  ──────────────┐
      │                                                          │
┌──────────────────────────────┐          ┌──────────────────────────────────────┐
│ Specifies the basic drive as  │          │ Specifies the drive on which          │
│ the current drive.            │          │ CONFIG.SYS and AUTOEXEC.BAT reside    │
│                               │          │ as the current drive.                 │
└──────────────────────────────┘          └──────────────────────────────────────┘
      │                                                          │
      └──────────────────────────┬───────────────────────────────┘
                        ┌──────────────────────┐
                        │     Load MS-DOS.      │
                        └──────────────────────┘
                                 │
                        ┌──────────────────────────────┐
                        │ Executes CONFIG.SYS and        │
                        │ AUTOEXEC.BAT in the current    │
                        │ drive.                         │
                        └──────────────────────────────┘
```
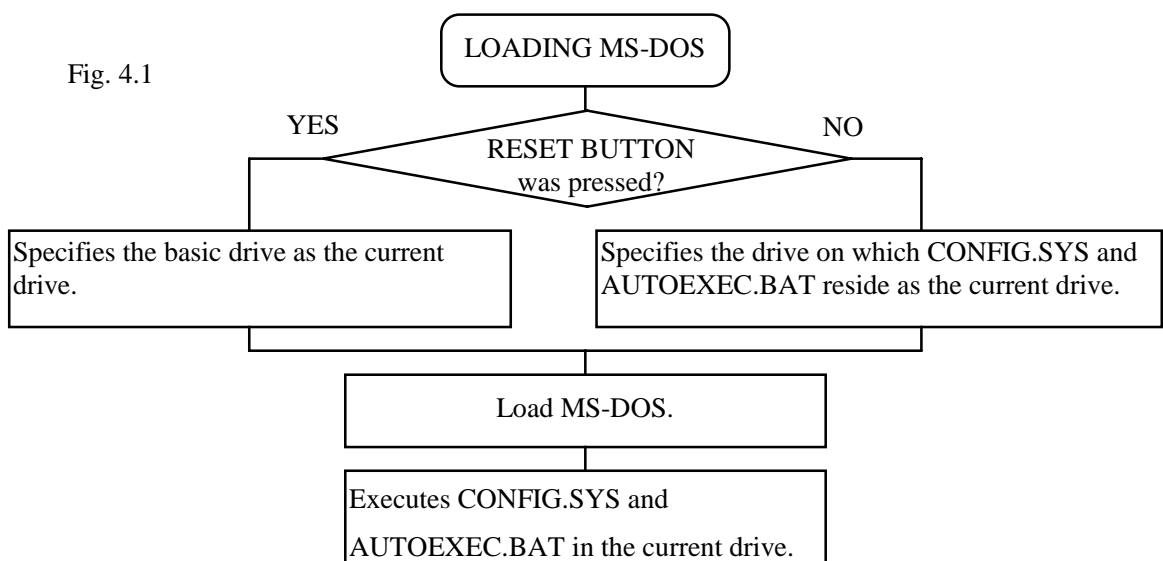
As described above, if the system power is turned on without an application installed (i.e. the conditions just after purchase), the CONFIG.SYS and AUTOEXEC.BAT files locating on the basic drive will be executed automatically. This inevitably initiates the System Menu (maintenance program). Therefore, if not only CONFIG.SYS and AUTOEXEC.BAT, but also an application program are installed on the user drive, it is possible for the application program to be automatically initiated from the user drive.

Example 1

In the following example MS-DOS is loaded from the RAM disk which has been designated as the current drive, since the system finds the CONFIG.SYS and AUTOEXEC.BAT first in the RAM disk.
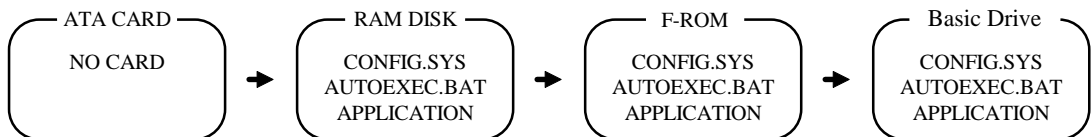
Fig. 4.2

Example 2

In the following example the RAM disk contains only CONFIG.SYS. As a result, MS-DOS is loaded from the F-ROM drive designated as the current drive.
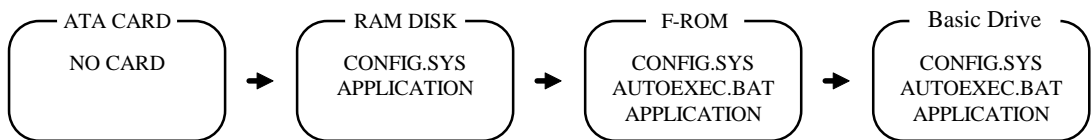
Fig. 4.3

Example 3

The following example shows a case where there is no F-ROM drive. The search order is also the same in this case. However in this case, CONFIG.SYS and AUTOEXEC.BAT in the basic drive will be executed, and System Menu will be initiated.
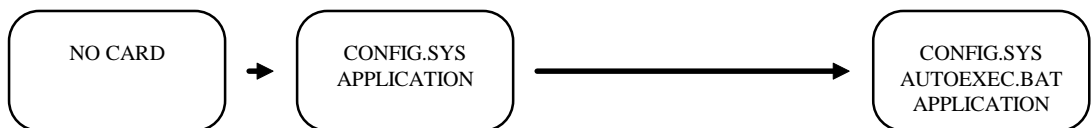
Fig. 4.4

4.2   How to Write CONFIG.SYS and AUTOEXEC.BAT

This section explains how to write the CONFIG.SYS and AUTOEXEC.BAT files mentioned in the previous section. A basic explanation of the CONFIG.SYS and AUTOEXEC.BAT is not given here. For further information about these files refer to the MS-DOS manual or appropriate technical documents. Observe the following points if writing a CONFIG.SYS file.

● The System Driver (SYSDRV.SYS) is required to operate this terminal.
   Always include a line through which to load the System Driver in the CONFIG.SYS.
● As described above, MS-DOS, which is in the basic drive, is always loaded.
   Consequently, C:\COMMAND.COM is used as the command interpreter. Therefore, set a path to COMMAND.COM to be reloaded in CONFIG.SYS using the SHELL command.
● Within CONFIG.SYS the MENU command can be used. Note however, that no power off command is included in the MENU selection screen. This is to prevent the power from being accidentally turned off while loading the drivers. The Power switch is also disabled until the CASIOAPM.COM program is initiated from AUTOEXEC.BAT, etc. In other words, the MENU command should only be used in the application program development processes.

Example of CONFIG.SYS
The following example shows a typical CONFIG.SYS file script. Since this example assumes that the system is booted from either the RAM disk or NAND F-ROM drive, it is necessary to partially modify it if booting up from the ATA card. For information about booting from the ATA card refer to Chapter 4.3, "Card Boot".

1	FILES=30	Not required
2	BUFFERS=20	Not required
3	DOS=HIGH, NOUMB	Required (1)
4	DEVICE=C:\SYSDRV.SYS	Required (2)
5	DEVICE=C:\HIMEM.SYS /M:2	Required (3)
6	DEVICE=C:\POWER.EXE	Required (4)
7	DEVICE=C:\TIME.SYS	Required (4)
8	DEVICE=C:\EMM386.EXE FRAME=C800 X=C000-C7FF X=D800-DFFF I=C800-D7FF	Required (5)
9	SHELL=C:\COMMAND.COM C:\ /P /E:1024	Required
10	DEVICE=C:\CARDSOFT\SS365SL.EXE /SKT=1	Required (6)
11	DEVICE=C:\CARDSOFT\CS.EXE /POLL:1	Required (6)
12	DEVICE=C:\CARDSOFT\CSALLOC.EXE	Required (6)
13	DEVICE=C:\CARDSOFT\ATADRV.EXE /S:1	Required (6)
14	DEVICE=C:\CARDSOFT\MTSRAM.EXE	Required (6)
15	DEVICE=C:\CARDSOFT\MTDDRV.EXE	Required (6)
16	DEVICE-C:\CARDSOFT\MTDAPM.SYS	Required (6)
17	DEVICE=C:\CARDSOFT\CARDID.EXE	Required (6)
18	INSTALL=C:\CARDSOFT\CS_APM.EXE	Required (6)

Note:

1 `DOS=HIGH,NOUMB`

This specifies that the main part of DOS is to be loaded in the HMA and, consequently, the UMB (Upper Memory Block) is not used. This terminal does not support a memory space for UMB if the EMS memory is to be used. Therefore, always specify NOUMB when using the EMS.

2 `DEVICE=C:\SYSDRV.SYS`

This driver is required to operate this terminal. Always install it before all other drivers.

3 `DEVICE=C:\HIMEM.SYS /M:2`

Never fail to specify the "/M:2" option.

4 `DEVICE=C:\POWER.EXE`

  `DEVICE=C:\TIME.SYS`

This driver is required to enable the APM function. TIME.SYS must follow immediately after POWER.EXE.

5 `DEVICE=C:\EMM386. EXE FRAME=C800 X=C000-C7FF X=D800-DFFF I=C800-D7FF`

Always specify the above options if using the EMS. Options other than the X option can be eliminated if the EMS is not used.

6 `DEVICE=C:\CARDSOFT\SS365SL.EXE /SKT=1`

  `DEVICE=C:\CARDSOFT\CS.EXE /POLL:1`

  `DEVICE=C:\CARDSOFT\CSALLOC.EXE`

  `DEVICE=C:\CARDSOFT\ATADRV.EXE /S:1`

  `DEVICE=C:\CARDSOFT\MTSRAM.EXE`

  `DEVICE=C:\CARDSOFT\MTDDRV.EXE`

  `DEVICE=C:\CARDSOFT\MTDAPM.SYS`

  `DEVICE=C:\CARDSOFT\CARDID.EXE`

  `INSTALL=C:\CARDSOFT\CS_APM.EXE`

This driver is required if the PC card driver is used. However, if the SRAM card is not used, the lines following ATADRV.EXE can be modified as follows. This saves a memory space as large as that used for the SRAM card driver. For more information refer to Appendix B, "PC Card Driver".

```
DEVICE=C:\CARDSOFT\ATADRV.EXE /D:1
DEVICE=C:\CARDSOFT\MTDAPM.SYS
DEVICE=C:\CARDSOFT\CARDID.EXE
INSTALL=C:\CARDSOFT\CS_APM.EXE
```

Example of AUTOEXEC.BAT

The following example shows a typical AUTOEXEC.BAT script. Since this example assumes that the system is booted from either the RAM disk or the NAND F-ROM drive, it is necessary to partially modify it if booting up from the ATA card. For information about booting from the ATA card refer to Chapter 4.3, "Card Boot".

1: `C:\ENDATA` 2: `C:\CASIOAPM` 3: (Environment variables setup and application call, etc.)	Required (1) Required (2) Optional

Note:

1 `C:\ENDATA`

Disables the card boot function in the BIOS. For more information refer to Chapter 4.3, "Card Boot".

2 `C:\CASIOAPM`

Enables the touch panel and Power switch operations. The touch panel and Power switch operations cannot be used until this program has been executed. This program only needs to be called once when booting the system.

4.3   Card Boot

Basically the "card boot" operation boots MS-DOS from the ATA card, just like it is booted from a floppy disk. For this terminal the boot operation looks the same as this case. However, this terminal uses a boot process greatly different from a general card boot so that the MS-DOS in the drive C is always loaded, in such a way that MS-DOS not residing in the card is booted.

Usually, in order to access the ATA card, a specific card driver is required. This card driver should be registered as a MS-DOS block device for the MS-DOS and added as a new drive to the system. Then the user can read from and write to the disk via the added drive by this device driver.
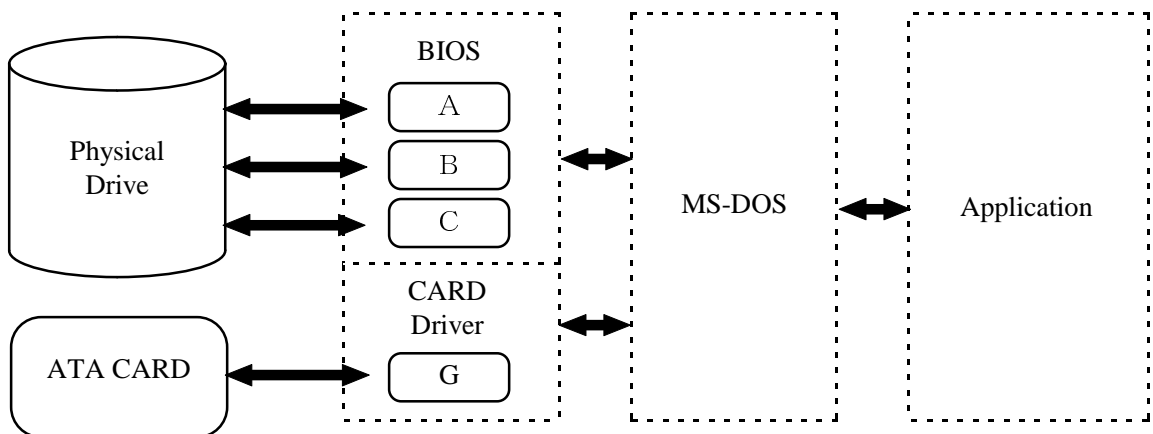
Fig. 4.5

However, in order to achieve a card boot, readout from the ATA card must be enabled before MS-DOS is loaded into the main memory. To solve this conflict the terminal has a function in its BIOS that can directly read the data from the ATA card. This function is assigned to the drive F (drive E for models without an F-ROM) and the ATA card looks, from MS-DOS, like a physical drive. As a result, when BIOS recognizes the presence of an ATA card during the boot process, it will search for CONFIG.SYS and AUTOEXEC.BAT in the ATA card prior to loading MS-DOS. If these files are found, the BIOS will load MS-DOS into main memory and shift control to MS-DOS after designating the drive F as the current drive. Subsequently, MS-DOS will execute the CONFIG.SYS and AUTOEXEC.BAT files in the current drive (drive F). This completes the load process.

The mechanism determining which drive is specified as the one to be used by an application that accesses the card is explained below. The drive G (drive F for models without F-ROM), which is a drive specifically reserved for applications, will be enabled by CARDID.EXE which is loaded into the main memory. It is loaded when CARDID.EXE is loaded and when both the drive F and drive G are being enabled. If this is the case, note that if an attempt is made to access the drive F,

the drive G, which is currently enabled, will be disabled.

This problem arises from the fact that the hardware conditions established by initialization with CARDID.EXE are lost since access to the drive F was executed by means of specific codes included in the BIOS. To avoid this problem, these specific codes in BIOS should be disabled. ENDATA.COM is used to do this. If ENDATA.COM is executed with the two drives mentioned above enabled, the specific codes (program) in BIOS are disabled, and the drive G can be retained as the only valid card drive. Below are example CONFIG.SYS and AUTOEXEC.BAT scripts used to boot a card.

Example of CONFIG.SYS

```
FILES=30
BUFFERS=20
DOS=HIGH,NOUMB
DEVICE=C:\SYSDRV.SYS
DEVICE=C:\HIMEM.SYS /M:2
DEVICE=C:\POWER.EXE
DEVICE=C:\TIME.SYS
DEVICE=C:\EMM386.EXE FRAME=C800 X=C000-C7FF X=D800-DFFF I=C800-D7FF
SHELL=C:\COMMAND.COMC:\ /P /E:1024
DEVICE=C:\CARDSOFT\SS365SL.EXE /SKT=1
DEVICE=C:\CARDSOFT\CS.EXE /POLL:1
DEVICE=C:\CARDSOFT\CSALLOC.EXE
DEVICE=C:\CARDSOFT\ATADRV.EXE /S:1
DEVICE=C:\CARDSOFT\MTSRAM.EXE
DEVICE=C:\CARDSOFT\MTDDRV.EXE
DEVICE=C:\CARDSOFT\MTDAPM.SYS
```

Example of AUTOEXEC.BAT

```
@ECHO OFF
C:
CD \
C:\CARDSOFT\CARDID.EXE
C:\ENDATA.COM
C:\CARDSOFT\CS_APM.EXE
PROMPT $p$g
PATH C:\
C:\CASIOAPM.COM
```

For the moment concentrate on the positions of CARDID.EXE and ENDATA.COM. CARDID.EXE can be registered as a device driver. In fact, this CARDID.EXE is registered as a device driver in

CONFIG.SYS which resides on the drive C. However, CARDID.EXE cannot be registered as a device driver at a card boot. If this CARDID.EXE is registered as a device driver, two drives may be enabled concurrently if MS-DOS executes CONFIG.SYS. In addition, if ENDATA.COM is called with the INSTALL command, the drive G is enabled exclusively. However, since MS-DOS is operating under the assumption that the Drive F is the current drive, an access error with the drive F, which does not actually exist, occurs because the AUTOEXEC.BAT file has been opened.

Then how about calling ENDATA.COM from AUTOEXEC.BAT? It is apparent that this is also not successful. Although two drives are enabled by executing CONFIG.SYS, the drive G having been enabled by CARDID.EXE is disabled when MS-DOS accesses the drive F to execute the AUTOEXEC.BAT file.

Next, the problem where a large program cannot be directly initiated from AUTOEXEC.BAT is explained. The explanation discusses the restrictions that apply to a card boot. This can be the situation when an attempt is made to read AUTOEXEC.BAT from the drive F while it is being disabled. COMMAND.COM consists of two independent parts called the resident part and non-resident part. The non-resident part will be overwritten by a large application program if it is loaded into the main memory. The resident part checks if the non-resident part has been destroyed at the termination of an application program, and will, if it is found to have been destroyed, reload the non-resident part again from the disk. In this case, accessing the drive F would not cause an error since the COMMAND.COM file to be read at this time was designated by the SHELL command in the CONFIG.SYS file. However, an error will result when an attempt is made by the reloaded COMMAND.COM file to open the AUTOEXEC.BAT file in order to continue its process. This problem can be avoided by shifting control priority from the AUTOEXEC.BAT file to another appropriate batch file in the drive G.

Example of AUTOEXEC.BAT

```
@ECHO OFF
C:\CARDSOFT\CARDID.EXE
C:\ENDATA.COM
----
G:
Other.bat
```

In the above example the current drive is moved to the drive G, and the Other bat file in the drive G is called. Since execution of the Other .bat file is performed under the assumption that the drive G is the current drive, no problem occurs if an attempt is made to open the same batch file in the course of reloading the non-resident part. But, it is prohibited to use a CALL statement to invoke the Other.bat file from AUTOEXEC.BAT. This will cause an error when control is returned to the AUTOEXEC.BAT file.

5.   Keyboard Controller

5.1   Overview

This terminal is equipped with a sub-CPU dedicated to controlling the keyboard, touch panel, backlight, and various sensors. This chapter describes major tasks assigned to this sub-CPU.
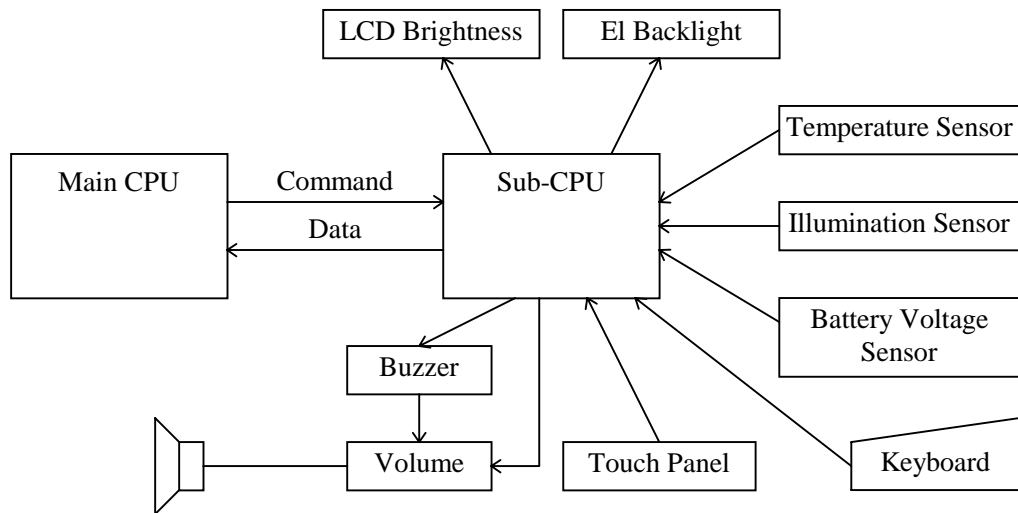
Fig. 5.1

5.2   Keyboard Control

The keyboard control of this terminal is compatible with the IBM PC/AT. The keyboard controller senses if a key has been pressed and sends a MAKE or BREAK code to the main CPU.
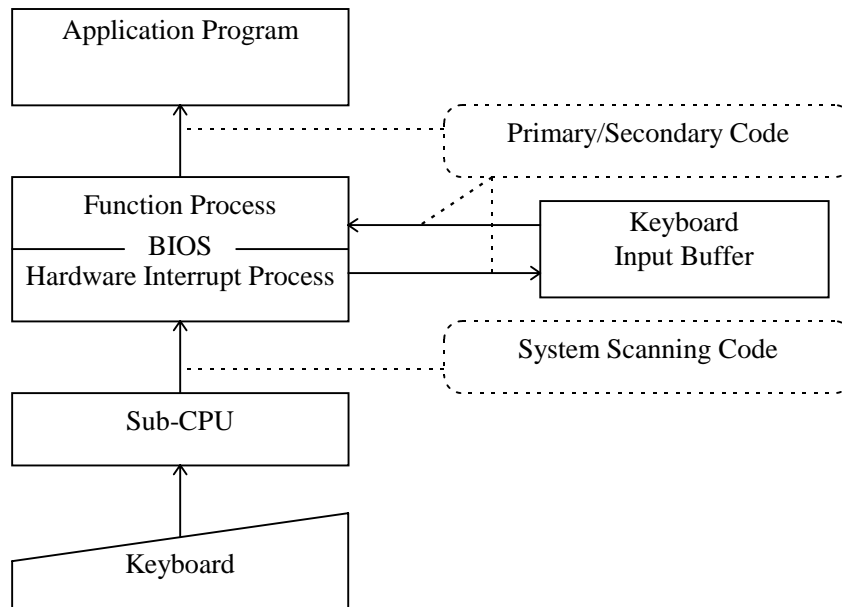
Fig. 5.2

System Scanning Code

Each keyboard scanning code generated from the keyboard main unit will be converted to the keyboard system scanning code through the controller.

     MAKE code   : Code generated when the corresponding key is pressed.
     BREAK code   : Code generated when the corresponding key is released.

Primary/Secondary Code

A code generated if an INT09h interrupt occurs will be converted to a primary code and a secondary code through the BIOS and set in the key buffer. They can be acquired from the application program by calling INT16h.

Primary code

Basically a character code (refer to the code table) is assigned to each key, except that 00h is assigned to function keys (Fn+ 0 to Fn+ 6), which must be recognized together with a secondary code as a set.

Secondary code

Basically a system scan code is assigned to each key, however, for some keys, different codes will be assigned depending on the Fn key.

Code Table

The following diagram shows the relationship between the keyboard keys and primary codes.
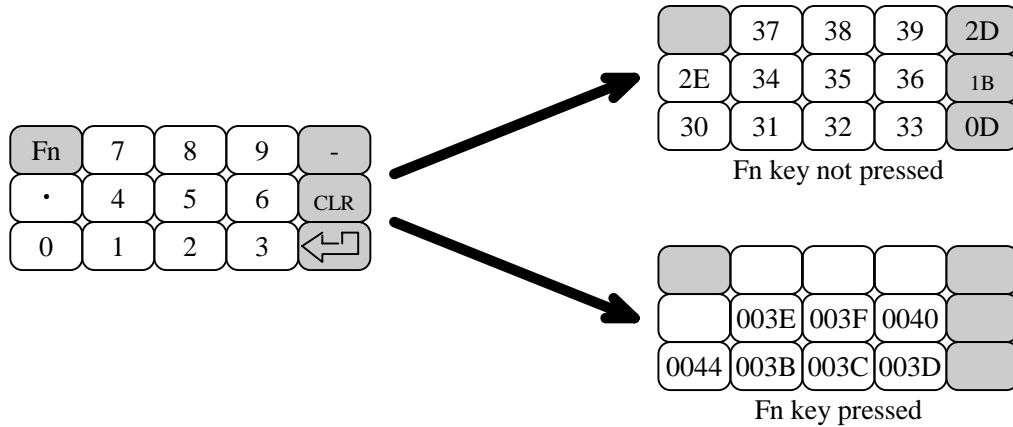
Fig. 5.3

Fn key

The Fn key is used to generate a system scan code for the function key if it is pressed together with a numeric key. For example, Fn + 1 generates a system scan code for the F1 key, and Fn + 0 generates a system scan code for the F10 key. However, Fn + 7 to 9 will not generate a system scan code that corresponds to any function keys because they have already been assigned to the following internal functions to be executed internally.

Operation	Function
Fn + 7	Toggles the backlight on and off.
Fn + 8	Increases the LCD screen contrast by one increment.
Fn + 9	Decreases the LCD screen contrast by one increment.

5.3　Touch Panel Control Function

The keyboard controller has incorporated a program for acquiring the touch coordinates of the touch panel. This program compensates these acquired coordinates with the values obtained through calibration so that correct coordinate values can be calculated. The calculated coordinates will be passed to a ROM-resident program called PEN BIOS when mouse interrupt occurs.

The following diagram shows an operational flow until the coordinates acquired by the keyboard controller are passed to the application program as a mouse event.
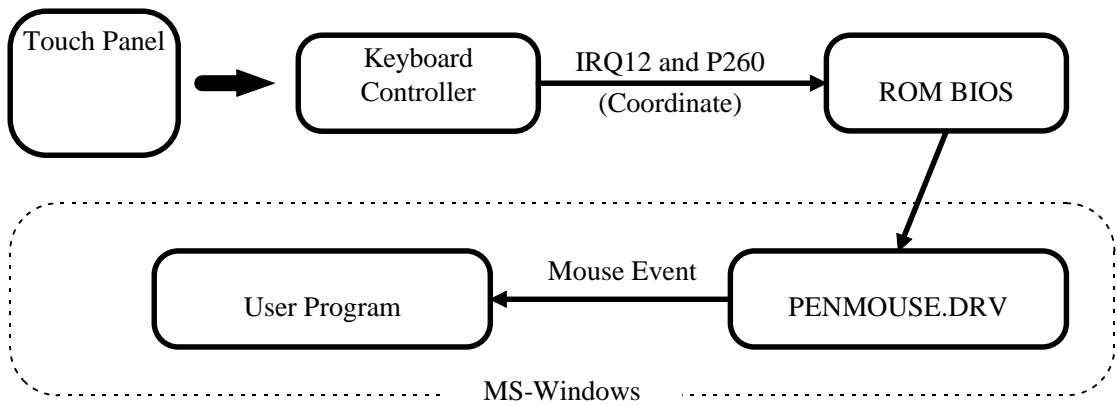
Fig. 5.4

5.4   Sensor Control

This terminal has the following three types of sensors installed to serve as dedicated devices for handy terminal.

Sensor type	Purpose of Use
Temperature sensor	Detects the temperature inside the main unit. This result will be used to automatically compensate the LCD brightness.
Illumination sensor	Detects the ambient light intensity to automatically turn on and off the backlight. This function is called the Auto Backlight Control (ABC) function, and it can be enabled or disabled with the System Menu or application programs.
Remaining battery voltage sensor	Used to acquire the remaining battery voltage. Application programs can obtain this value via the APM BIOS.

5.5   Backlight Control

This terminal has incorporated two types of automatic backlight control functions: ABO (Auto Backlight OFF) and ABC (Auto Backlight Control). The ABO function is used to turn off the backlight if no key or touch panel input has been made for a given period of time, and the ABC function is used to automatically turn on and off the backlight depending on the intensity of the ambient light. These operations are performed by the keyboard controller.

ABC (Auto Backlight Control)

The ABC function automatically turns on or off the backlight by detecting the ambient light intensity. Every second it determines the amount of light received by the illuminance sensor and automatically turns on or off the backlight depending on whether the amount of light is less than the given amount or more than the given amount.
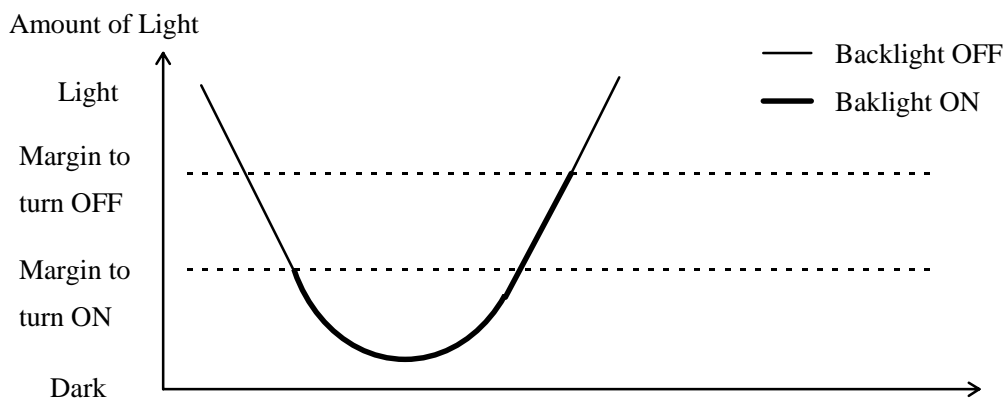
Fig. 5.5

In the above graph, the marginal light amount across which the backlight is turned ON is at a setting less than if the backlight is OFF. If these two levels are identical, the backlight will flicker if it detects a small variation in the incident light on the illumination sensor. To avoid this problem an appropriate hysterisis is provided.

Transition of Backlight Control Methods

The concept of ABC lies in automating user operations. However, automatic control depends on the illumination sensor. It cannot be perfect because various types of light, sunlight or room light for example, may be incident to the sensor. Consequently, this requires manual ON/OFF control even if under ABC control. This leads to a further problem wherein the user may forget to turn it on or off. To avoid these problems this system employs the following rules for transition between ABC, manual operation (ON function/OFF function), and ABO.
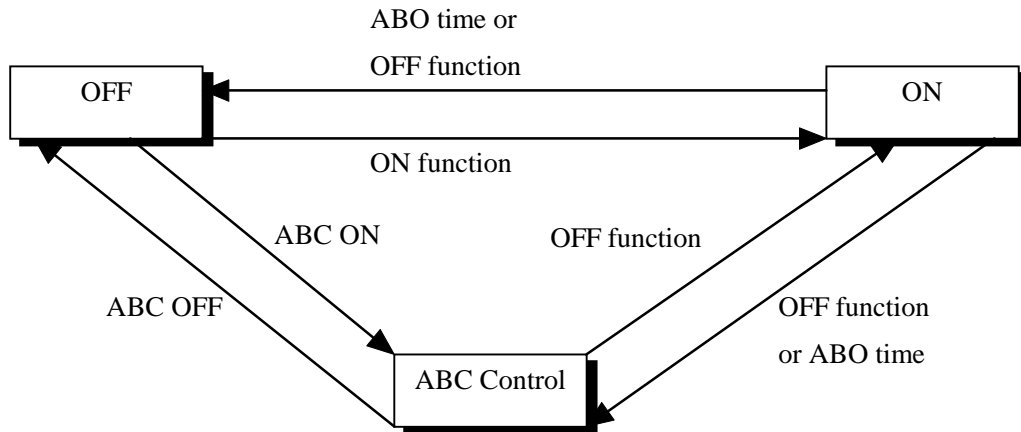
Fig. 5.6

		Press F7 key	ABO time-up	ABC Enable	ABC Disable	Becomes dark	Becomes light
1	ABC disabled Backlight ON state	→ 2	→ 2	→ 3 or 4 *1	---	Ignore	Ignore
2	ABC disabled Backlight OFF state	→ 1	---	→ 3 or 4 *1	---	Ignore	Ignore
3	ABC enabled Backlight ON state	→ 6	Ignore	Ignore	→ 2	---	→ 4
4	ABC enabled Backlight OFF state	→ 5	---	Ignore	→ 2	→ 3	---
5	ABC temporarily disabled Backlight ON state	→ 4	→ 3 *3	Ignore	→ 2	→ 3 *4	---
6	ABC temporarily disabled Backlight OFF state	→ 3	---	Ignore	→ 2	---	→ 4 *4

Note:

*1:   The backlight turns ON or OFF depending on the current light intensity.

*2:   ABO event does not occur during ABC. However, if the terminal is left in a dark place,
        the APO (Auto Power OFF) function will turn off the backlight.

*3:   Since the backlight is presently ON, the normal state can be restored by jumping to step [3].

*4:   Cancels the sole condition of "ABC temporarily disabled".

6. Drivers

6.1   Overview

The following drivers are supplied for this terminal. Install them as required for operation.

Name	File name	Purpose
System driver	SYSDRV.SYS	Driver required to operate the system. This driver must be installed.
Clock control driver	TIME.SYS	Executes the process that restores the clock condition at a resume-boot in cooperation with POWER.EXE. This driver must be installed.
Hardware window manager	HWWMAN.EXE	Driver that controls the hardware window. It is called from the keypad driver.
Keypad driver	KEYPAD.EXE	Driver that adds the keypad function to the system. This driver is called from applications via the keypad library.
PenMouse driver	PENMOUSE.COM	Driver to simulate the Microsoft mouse driver operation on the touch panel.

For information about the drivers associated with MS-DOS refer to an MS-DOS reference manual or other technical reference documents published separately by third party.

6.2　System Driver

6.2.1　Function

The system driver (SYSDRV.SYS) must be installed because it executes critical processes in this terminal. The system driver mainly performs the following processes.

● **LB1 monitoring and warning**

Monitors the main battery conditions and sounds a warning buzzer if an LB1 event is detected.

It also forcibly turns off the system, if the battery voltage has not recovered within ten minutes of the buzzer sounding.

● **Alarm notification**

When alarm (INT4Ah) occurs, the driver will hook the interrupt and ring the buzzer. And, the driver will notify to the user.

● **Adjustment of the number of display lines**

On a general VGA screen twenty five lines (if video mode=03h) of text are displayed. However, on this terminal, it is limited to twenty four lines because of the screen size.

To make display possible the system driver modifies the number of allowable lines to twenty four. The number of display columns has not been modified.

6.2.2　Startup Method

This driver is loaded by defining the DEVICE statement in the CONFIG.SYS file. SYSDRV.SYS is stored in the basic drive (C:).

Format
```
DEVICE=C:\SYSDRV.SYS
```

Start option
None

Note:

SYSDRV.SYS must be loaded before any device drivers.

6.3   Clock Control Driver

6.3.1   Function

This driver adjusts the system time on this terminal. This driver must be installed.

On a general PC a timer interrupt occurs every 55 msec to update the clock tick counter, which is one of the BIOS system variables, and the clock overflow counter. The clock tick counter is incremented each time the timer interrupt occurs and read out from the real-time clock (RTC) when the PC power is turned on, and disappears when the power is off. However, in the case of a handheld terminal, since the suspend/resume state is frequently cycled, the clock tick counter is initialized only once, at the initial boot. Therefore, the clock time may be slightly off if the terminal is operated for a long period of time. To avoid this problem the terminal uses this driver to control the clock in cooperation with POWER.EXE so that the time can be directly read from the RTC. This ensures that the correct time can always be obtained, irrespective of the length of operation. However, since the time is read from the RTC in seconds, the 1/100 of a seconds digit will be ignored if the time is read using INT21h.

The relationship between the clock control driver and application programs is shown in the following diagram.
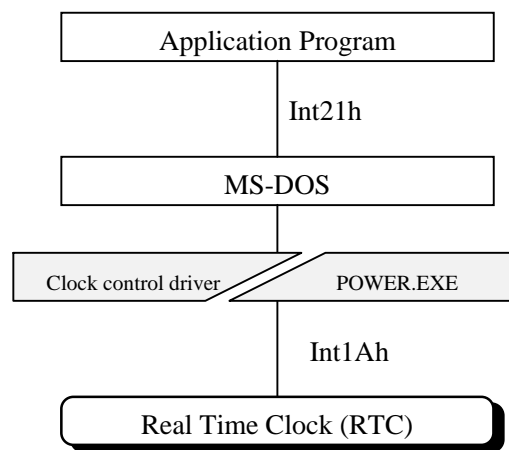
Fig. 6.1

6.3.2 Startup Method

This driver is loaded by defining the DEVICE statement in the CONFIG.SYS file. TIME.SYS is stored in the basic drive (C:).

Format
```
DEVICE=C:\TIME.SYS
```

Start option
None

Note:
TIME.SYS must be loaded immediately after POWER.EXE.

6.4　Keypad Driver/Hardware Window Manager

6.4.1　Function

The keypad driver (**KEYPAD.EXE**) is used to add the keypad function to the system. Application programs can use the keypad by calling the keypad driver functions via the keypad library (refer to Chapter 7.6.3 "Keypad Library").

This keypad driver internally calls the hardware window manager that enables the use of the hardware window. Therefore, the use of the keypad driver requires the residence of the hardware window manager.

The keypad driver is also used by some utilities (refer to Chapter 8 "Utility".) supported for this terminal. Therefore, before executing an application program or utility that uses the keypad driver, make it reside in the main memory.

The relationship between the keypad driver/hardware window manager and application programs is shown by the following diagram.

```
          ┌─────────────────────────────┐
          │     Application Program     │
          └─────────────────────────────┘
                         │
          ┌─────────────────────────────┐
          │        Keypad Driver        │
          └─────────────────────────────┘
               │                 │
        ┌────────────┐   ┌──────────────────────────┐
        │  PEN BIOS  │   │  Hardware Window Manager  │
        └────────────┘   └──────────────────────────┘
               │                 │
        ┌────────────┐   ┌──────────────────────────┐
        │ Touch Panel│   │     Hardware Windows     │
        └────────────┘   └──────────────────────────┘
```
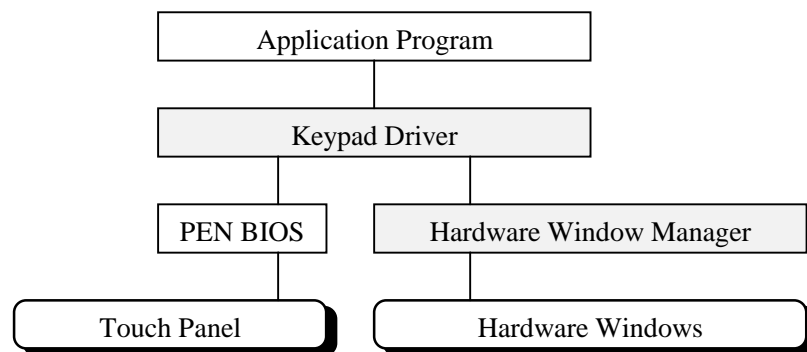
Fig. 6.2

6.4.2 Startup Method

Format

```
HWWMAN [Option]
KEYPAD [Option]
```

Start option

```
/R   Cancels the residence.
```

To make each driver resident in the main memory, make the following specification at the DOS prompt. Always install the hardware window manager first. These drivers are stored in the basic drive (C:).

```
C:\>HWWMAN
C:\>KEYPAD
```

The residency of these drivers can be released by specifying as follow. The keypad driver must be released first.

```
C:\>KEYPAD/R
C:\>HWWMAN/R
```

Note:

The keypad driver uses 2 pages (32 KB) of EMS memory space. Before using the keypad driver insert a line specifying the use of EMS memory in CONFIG.SYS.

6.5 PenMouse Driver

6.5.1 Overview

The PenMouse driver (PENMOUSE.DRV) simulates the operation of the mouse driver (INT33h) specific to the personal computer using inputs received from the touch panel. The PenMouse driver makes it possible to run on the IT-2000 terminal an application that was designed for use with a mouse driver on the personal computer.

However, perfect simulation cannot be achieved because of the physical difference between the mouse and touch panel. For example, no touch panel operation can simulate a right mouse button click. However, application developers do not have to be particularly concerned with this difference. This is because a right mouse button click can be recognized as a "Pen UP" state.

Some of the functions described in the following "Dummy Function" written next to them. These functions have not been supported due to the difference between the mouse and touch panel, as mentioned above, or because the functions are not associated with an actual operation.

The PenMouse driver is designed to ignore a function call which is not supported, instead of returning an error to the function call.

This should provide a more flexible development environment for application developers.

The relationship between the PenMouse driver and application programs is shown by the following diagram.
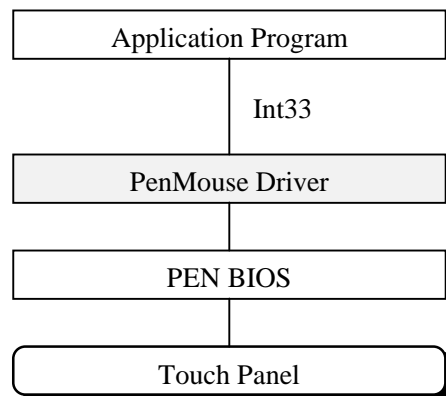
Fig. 6.3

6.5.2 Startup Method

The PenMouse driver is supplied as an SDK. Before use, copy it to the F-ROM drive (D:) or RAM disk (A:). To load the PenMouse driver, make the following specification at the DOS prompt.

Format:
```
PENMOUSE [Option]
```

Start option:
```
/R   Cancels the residence.
```

6.5.3 Functions

Page	Function	Page	Function
103	Initialize	107	Set Mouse Resolution
103	Display Cursor	107	Set Erase Cursor Range
103	Erase Cursor	107	Replace User Handler
103	Read Cursor Position	107	Get Mouse Information Size
104	Set Cursor Position	107	Get Mouse Information
104	Read Pushed Counter and Position	108	Restore Mouse Information
104	Read Popped Counter and Position	108	Set Interrupt Subroutine
105	Set X Range	108	Get Interrupt Subroutine
105	Set Y Range	108	Set Mouse Distance
105	Set Cursor Form	108	Get Mouse Distance
105	Set Text Cursor	108	Set CRT Page Number
105	Read Mouse Distance	109	Get CRT Page Number
106	Set User Handler	109	Disable Mouse Driver
106	Enable Write Pen Emulation	109	Enable Mouse Driver
106	Disable Write Pen Emulation	109	Software Reset

The functions supported by this driver are summarized below.

Function	Description

Initialize	Returns information as to whether the mouse can be used and sets the mouse function to the initial conditions.
	To start using the mouse this function must be called.
	
	Input:
	     AX = 0
	
	Output:
	     AX = Use of mouse permitted/not permitted
	          = 0 Use of mouse not permitted
	          = -1 Use of mouse permitted
	     BX = Number of mouse buttons
	
	Note:
	The position of the cursor is first set to the center of the screen if this function is executed on a PC. However, it is positioned at the top left corner of the screen (0,0) on this terminal.

Display Cursor
(Dummy Function)
	Displays a cursor on the screen. This is a dummy function.
	
	Input:
	     AX = 1
	Output:
	     None

Erase Cursor
(Dummy Function)
	Removes the cursor from the screen. This is a dummy function.
	
	Input:
	     AX = 2
	Output:
	     None

Read Cursor Position	Reads the information about the current cursor position and whether the mouse button is pressed.
	
	Input:
	     AX = 3
	
	Output:
	     BX = Button status
	       b0=1 The left button is pressed.
	       b1=1 The right button is pressed.
	     CX = X coordinate of the cursor position
	     DX = Y coordinate of the cursor position

Set Cursor Position	Places the cursor at the specified position
	
	Input:
	    AX = 4
	    CX = X coordinate of the cursor position
	    DX = Y coordinate of the cursor position
	Output:
	    None

Read Pushed Counter and Position	Reads the number of times the specified mouse button is pressed, the cursor position when the specified button was last pressed, and the current button status. If this function is called, the number of times pressed is initialized to 0.
	
	Input:
	    AX = 5
	    BX = Specified button
	        0 Left button
	        1 Right button
	
	Output:
	    AX = Current button status
	        b0=1 The left button is pressed.
	        b1=1 The right button is pressed.
	    BX = Number of times the specified button is pressed
	    CX = X coordinate of cursor when the specified button was last
	        pressed.
	    DX = Y coordinate of cursor when the specified button was last
	        pressed.

Read Popped Counter and Position	Reads the number of times the specified mouse button is released (popped), the cursor position when the specified button was last released, and the current button status. If this function is called, the number of times released is initialized to 0.
	
	Input:
	    AX = 6
	    BX = Specified button
	        0 Left button
	        1 Right button
	Output:
	    AX = Current button status
	        b0=1 The left button is pressed.
	        b1=1 The right button is pressed.
	    BX = Number of times the specified button is released
	    CX = X coordinate of cursor when the specified button was last
	        released.
	    DX = Y coordinate of cursor when the specified button was last
	        released.

Set X Range	Determines the range of cursor movement on the screen (maximum and minimum in the X direction).
	
	Input:
	    AX = 7
	    CX = X coordinate of the left limit of cursor movement
	    DX = X coordinate of the right limit of cursor movement
	
	Output:
	    None

Set Y Range	Determines the range of cursor movement on the screen (maximum and minimum in the Y direction). On this terminal, if the user makes a touch outside the range set by this function, the limit value is acquired.
	
	Input:
	    AX = 8
	    CX = Y coordinate of the upper limit of cursor movement
	    DX = Y coordinate of the lower limit of cursor movement
	
	Output:
	    None

Set Cursor Form (Dummy Function)	Sets up also the shape of the graphic cursor. This is a dummy function.
	
	Input:
	    AX = 9
	
	Output:
	    None

Set Text Cursor (Dummy Function)	Sets up the type and shape of the text cursor. This is a dummy function.
	
	Input:
	    AX = 10
	
	Output:
	    None

Read Mouse Distance	Reads the distance from the current mouse position to the position it was in when this function was last called.
	
	Input:
	    AX = 11
	
	Output:
	    CX = Distance of mouse movement in the X direction
	    DX = Distance of mouse movement in the Y direction

Set User Handler	Sets up the interrupt condition required to call the corresponding subroutine in the application program as well as the subroutine to be called. If an interrupt occurs, the following values are stored in the register.
	
	Input:
	      AX = 12
	      CX = Interrupt mask
	          b0=1 Cursor position is modified.
	          b1=1 Left button is pressed.
	          b2=1 Left button is released.
	          b3=1 Right button is pressed.
	          b4=1 Right button is released.
	      ES:DX = Interrupt address
	
	Output:
	      None
	
	If an interrupt occurs, the following values are stored in the register.
	      AX = Mouse status
	          b0=1 Cursor is moved.
	          b1=1 Left button is pressed.
	          b2=1 Left button is released.
	          b3=1 Right button is pressed.
	          b4=1 Right button is released.
	      BX = Button status
	          b0=1 Left button is pressed.
	          b1=1 Right button is pressed.
	      CX = X coordinate of the cursor position
	      DX = Y coordinate of the cursor position
	      SI = Distance of mouse movement in the X direction
	      DI = Distance of mouse movement in the Y direction

Enable Write Pen Emulation (Dummy Function)	Enables the emulation of write pen with the mouse. This is a dummy function.
	
	Input:
	      AX = 13
	
	Output:
	      None

Disable Write Pen Emulation (Dummy Function)	Disables the emulation of write pen with the mouse. This is a dummy function.
	
	Input:
	      AX = 14
	
	Output:
	      None

Set Mouse Resolution (Dummy Function)	Sets up the factor by which cursor movement on the screen reflects actual mouse displacement. This is a dummy function.
	
	Input:
	    AX = 15
	Output:
	    None

Set Erase Cursor Range (Dummy Function)	Sets up the range within which the cursor is off the screen. This is a dummy function.
	
	Input:
	    AX = 16
	Output:
	    None

Replace User Handler	Sets up the interrupt condition required to call the corresponding subroutine in the application program as well as the subroutine to be called.
	The previous interrupt mask and interrupt address can be acquired.
	The information acquired when the interrupt occurs is the same as the information acquired with "Set User Handler" (AX=12).
	
	Input:
	    AX = 20
	    CX = Interrupt mask
	        b0=1 Cursor position is modified.
	        b1=1 Left button is pressed.
	        b2=1 Left button is released.
	        b3=1 Right button is pressed.
	        b4=1 Right button is released.
	    ES:DX = Interrupt address
	
	Output:
	    CX = Original interrupt mask
	    ES:DX = Original interrupt address

Get Mouse Information Size (Dummy Function)	Returns the buffer size required to save the current mouse driver conditions. This is a dummy function.
	
	Input:
	    AX = 21
	Output:
	    None

Get Mouse Information (Dummy Function)	Saves the mouse driver conditions in the allocated buffer. This is a dummy function.
	
	Input:
	    AX = 22
	Output:
	    None

Restore Mouse Information (Dummy Function)	Restores the mouse driver conditions that have been saved by "Get Mouse Information". This is a dummy function.
	
	Input:
	    AX = 23
	Output:
	    None

Set Interrupt Subroutine (Dummy Function)	Sets up the interrupt condition required to call the corresponding subroutine in the application program as well as the subroutine to be called.
	This function is the same as "Set User Handler" (AX=12), except that a set of key strokes can be included in the interrupt mask.
	This is a dummy function.
	
	Input:
	    AX = 24
	Output:
	    None

Get Interrupt Subroutine (Dummy Function)	Acquires the address of the alternate interrupt subroutine that currently corresponds to the specified interrupt mask. This is a dummy function.
	
	Input:
	    AX = 25
	Output:
	    None

Set Mouse Distance (Dummy Function)	Sets up the mouse sensitivity. This is a dummy function.
	
	Input:
	    AX = 26
	Output:
	    None

Get Mouse Distance (Dummy Function)	Acquires the currently set mouse sensitivity. This is a dummy function.
	
	Input:
	    AX = 27
	Output:
	    None

Set CRT Page Number (Dummy Function)	Sets the CRT page number in which to display the cursor. This is a dummy function.
	
	Input:
	    AX = 29
	Output:
	    None

Get CRT Page Number (Dummy Function)	Acquires the CRT page number in which to display the cursor. This is a dummy function. **Input:** AX = 30 **Output:** None
Disable Mouse Driver (Dummy Function)	Restores all the interrupt vectors used by the mouse driver, except INT31h, to the values they had before the mouse driver was installed. This is a dummy function. **Input:** AX = 31 **Output:** None
Enable Mouse Driver (Dummy Function)	Again sets up the values of all interrupt vectors used by the mouse driver. This is a dummy function. **Input:** AX = 32 **Output:** None
Software Reset	This function is the same as "Initialize" (AX=0) except that the mouse hardware is not initialized. **Input:** AX = 33 **Output:** None

7. Application Development

7.1   Overview

This terminal uses the IBM PC/AT architecture. The actual display size is 192 (H) x 384 (V) pixels, internally with the area of 640 (H) x 480 (V) pixels, is supported. Therefore, if the user develops an application that makes use of the upper left 192 (H) x 384 (V) region of the display, a dedicated application program will run on this terminal. Also, since the numeric keys generate the same keycodes as the IBM PC/AT machine, there is no need to discriminate between this terminal and the development machine in terms of the standard input/output operations.

There are two methods for developing user applications that use the touch panel. One method is to use the mouse (INT 33h) for development. Application programs developed using this method can be operated with a dedicated mouse driver on this terminal. The other method is to use the keypad library.

The keypad is a set of dedicated driver and library used for characters, such as alphabets, that cannot be entered using the numeric keys. Since the programs for implementing each method cannot reside in the terminal concurrently, select one of them depending on the nature of the application program to be developed.

Applications that use the COM1 port (8-pin) can be programmed in the same way they are for IBM PC/AT machines except that they must include the power control functions. On this terminal, the power to the COM port is default-set to off so that the power consumption is reduced to a minimum. Therefore, application programs that use the COM port must turn on the power to the COM port in advance using the system library.

7.2   Notes on Developing Application

- Any program that uses the COM port must turn on the power to it in advance using the system library. The power to the COM port remains on once it has been turned on, or until it is turned off by the system library or until the RESET button is pressed. Therefore, do not forget to turn off the power to the COM port when it is no longer required. This power is automatically turned off during the suspend state, but power is restored to it if system operation is resumed. Accordingly, the application program side does not have to be aware of the power condition.

- This terminal has an operation mode called the DOZE mode. It is one of the unique IT-2000 power management functions. In this mode power consumption is restricted by operating the CPU at a low speed. Usually, the CPU exits this mode and operates at a high speed if the COM port is accessed. However, if the system remains for a long period of time in the wait state for a reception interrupt from the COM port, it will automatically shift to the DOZE mode. If data is received in the DOZE mode, it may not be received because the shift to the high-speed mode will not be achieved successfully. To avoid this problem create programs that use the COM port in which a FIFO data buffering method is always used.

- If a program is running on MS-DOS, data may not always be written in the physical disk each time the file write function is called. MS-DOS will hold the write data in memory until a given amount of data is accumulated. Do not turn the power off and on or remove and insert the card if this occurs. If this event occurs, the programmer should create an application which calls the COMMIT command form MS-DOS after attempting a write to the disk.
  This COMMIT function can also be called using the _dos_commit () function of Microsoft-C.

- If an error occurs during a write to the disk, a fatal error handler (INT24h) will be called. This handler, which has been set as the default in MS-DOS, will always be called unless the application program intentionally replace it. This default handler requires the "R" key to be pressed for retry or the "A" key to be pressed for abort. However, since the terminal only has numeric keys, this handler cannot be used to respond to an inquiry from MS-DOS if the keypad is not displayed. To bypass this problem, the programmer should provide an INT24 handler within the application program.

7.3   Development Environment

7.3.1   Development Environment

To develop application programs a 16-bit compiler, Microsoft C/C ++ 7.0 or later, and a computer on which the compiler can run are required.

7.3.2   Application Development Library

For this terminal various libraries such as the keypad library and OBR library, which is used to enhance the efficiency of developing applications. This terminal is also provided with the libraries of controlling the IT-2000 dedicated devices such as the backlight control and device power control, etc. These libraries will control only hardware compatible to the IBM PC/AT. For example, the OBR library directly controls the COM port to communicate with an OBR, but it must internally call the system library to turn on the power to the COM port. This is because the power control function of the COM port has been customized to the handy terminal. Also, the system library merely provides an interface with the system driver and does not directly control the COM port power. The actual power supply to the COM port is controlled by the system driver.
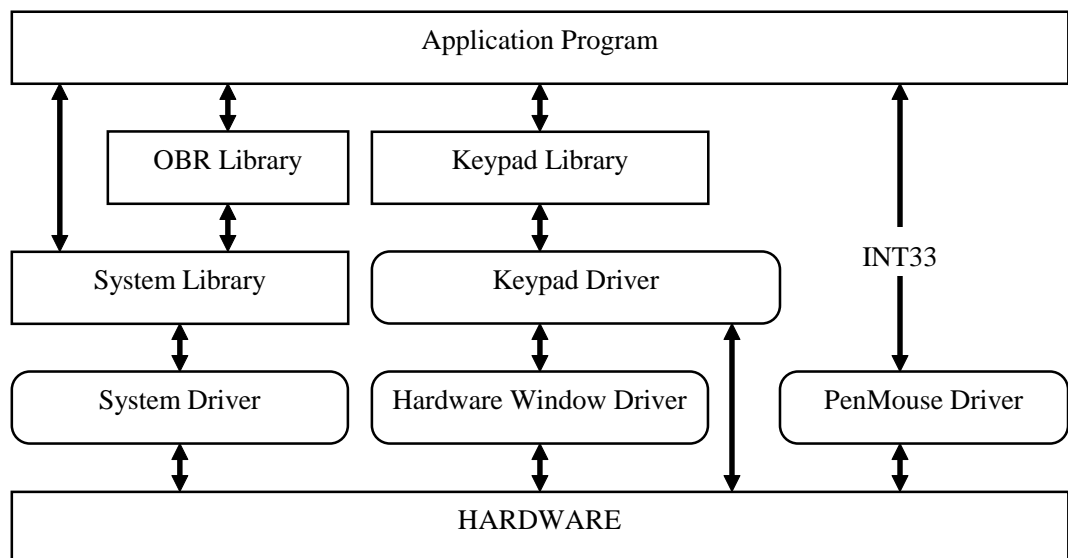
Fig. 7.1

These libraries do not have to always be used. And, in as far as standard PC/AT programming is pursued, they do not have to be used at all. The sole exception is that the COM port power must be turned on via the system library if the user wants to control the COM port directly.

7.3.3 Simulation Driver

As explained above, the libraries for this terminal only control hardware that is compatible with the IBM PC/AT. This is important to remember if application programs for the terminal are developed on a personal computer. Although each library is linked to the application program to form an executable program, they do not contain code that is specific to the hardware of the handy terminal. Consequently, if a simulation driver is used rather than one of the drivers dependent on the terminal hardware, the application program can be made to run, without modification, on the personal computer. This is the basic concept of simulation.

The shaded portion in the diagram below shows the simulation environment that has been constructed on the personal computer. However, for the mouse driver, use Microsoft's mouse driver. This is because the mouse driver on the terminal simulates the operations of the Microsoft mouse driver, while other drivers simulate the actual operations on the terminal.
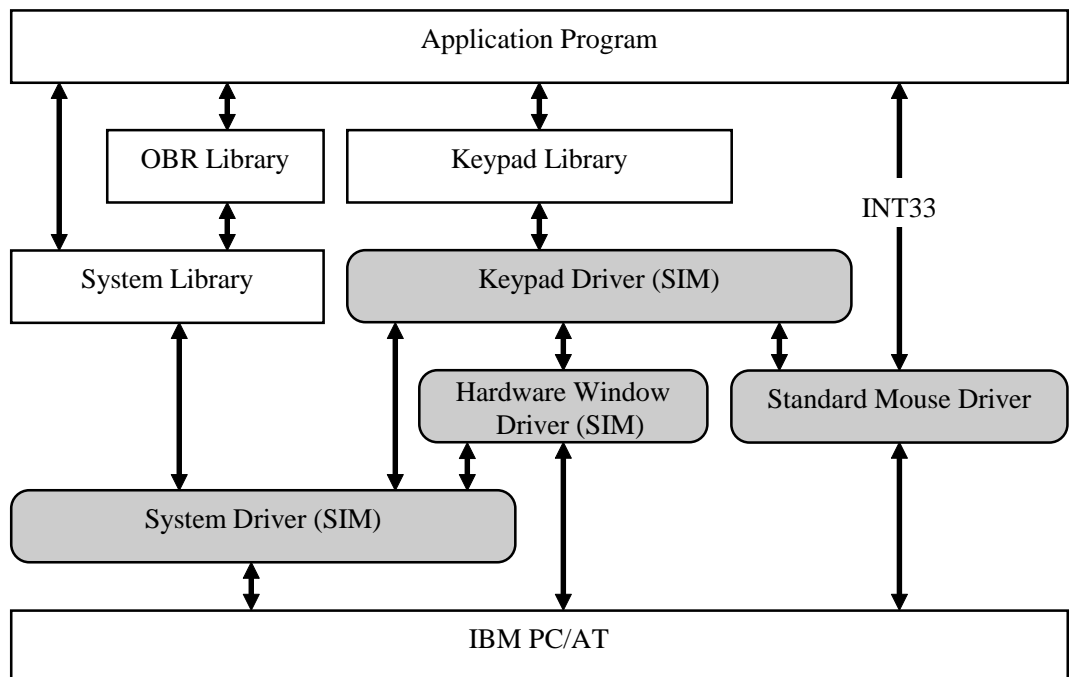
Fig. 7.2

For information about each driver refer to Chapter 7.5 "Simulation Driver".

7.4   Program Development Procedure

The following diagram shows the basic procedural flow used to develop an application program that runs on this terminal. The following paragraphs explain the details of each phase of the procedural flow.

```
+----------------------------------------+   +----------------------------------------+
|              ( Start )                 |   |                                        |
|                  |                     |   |        +-----------------+             |
|                  v                     |   |        |                 |             |
|   +------------------------------+     |   |        |    Transfer     |             |
|   | Creation/Edition of sourcecode|    |   |        |        v        |             |
|   +------------------------------+     |   |   +------------------------+           |
|                  |                     |   |   | Operation check on IT-2000 |        |
|   +------------------------------+     |   |   +------------------------+           |
|   |      Compile/Assemble        |     |   |             |                          |
|   +------------------------------+     |   |             v                          |
|                  |                     |   |          ( End )                       |
|   +------------------------------+     |   |                                        |
|   |            Link              |     |   |                                        |
|   +------------------------------+     |   |                                        |
|                  |                     |   |                                        |
|   +------------------------------+     |   |                                        |
|   |        PC simulation         |     |   |                                        |
|   +------------------------------+     |   |                                        |
|  NG              |           OK        |   |                                        |
|    <>----- Operation OK -----> --------+---+                                        |
|                                        |   |                                        |
|           < PC side >                  |   |           <IT-2000 side>               |
+----------------------------------------+   +----------------------------------------+
```
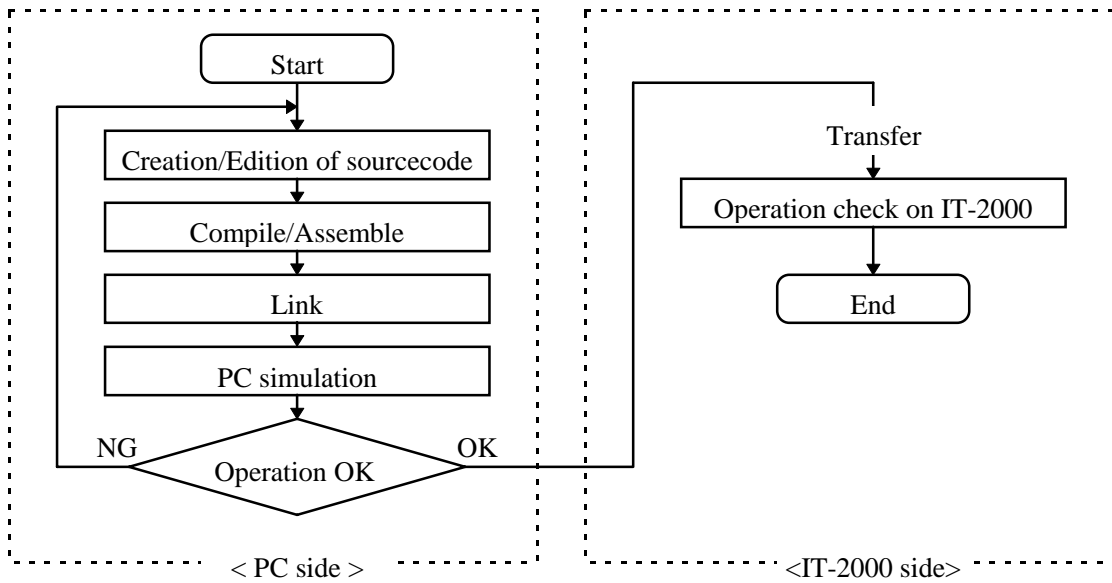
Fig. 7.3

7.4.1 Creation of Execution File

Application developers should develop programs using MS-DOS, IBM PC/AT BIOS and various application development libraries. The following sample program is used to turn on and off the backlight. With this program the backlight will be turned on or off if either "1" or "0", respectively, is entered through the numeric keypad. This program can be terminated by the input of the ESC key.

```c
#include <stdio.h>
#include "syslib.h"

void main()
{
   char ch;

   for (; ;) {
       switch (ch = getch()) {
       case '0':
       case '1':
           SYS_SetBackLight(ch-'0'); /* System Library function */
           break;
       case 0x1B:
           exit(0);
       default:
           break;
       }
   }
}                 <Test.c>
```

Next, create the execution file with the following procedure.

```
C:\SAMPLE>cl -c -Zip -Otin -Ic:\IT-2000\include test.c
Microsoft (R) C/C++ Optimizing Compiler Version 8.03
Copyright (c) Microsoft Corp 1984-1995. All rights reserved.


test.c


C:\SAMPLE>link /LI /m test,,,c:IT-2000\lib\slibsysd;


Microsoft ( R ) Segmented Executable Linker Version 5.63.2 20 Nov 29 1994
Copyright (C) Microsoft Corp 1984-1995. All rights reserved.


C:\SAMPLE>
```

This example assumes that the SDK of the IT-2000 has been installed in C:\ IT-2000. If it is installed in another directory, it is necessary to designate the location in which to store the header file and library file according to the development environment. These designation can be made using the environment variables INTCLUDE and LIB.

For more information refer to a compiler manual published separately by a third party.

7.4.2   Debugging Through Simulation

An application is debugged using the simulation driver. The configuration of simulation drivers to be loaded varies depending on the application program to be developed.

	Library or device to be used	At operation on IT-2000	When simulator is active
1	System library	SYSDRV.SYS	SYSDRVP.COM
2	Keypad library	SYSDRV.SYS HWWMANP.EXE KEYPADP.EXE	SYSDRVP.COM MOUSE.COM HWWMANP.COM KEYPADP.COM
3	Touch panel (PENMOUSE.COM)	PENMOUSE.COM	MOUSE.COM

● If the application program only uses the system library, register SYSDRV.SYS as the device driver on the terminal that performs simulation. If the simulation is performed on a personal computer, install SYSDRVP.COM on it as the resident program.

● To execute an application program on a personal computer that uses the keypad library, SYSDRVP.COM, MOUSE.COM , HWWMANP.COM, and KEYPADP.COM should be installed on it. MOUSE.COM is used to simulate touch panel operation on the personal computer.

● If an application program is programmed assuming that the mouse (INT33h) is used, install PENMOUSE.COM on the terminal and install MOUSE.COM on the personal computer.

In addition, if this application calls the system library, the environment defined in above is essential. The above sample program shown calls the system library to turn on the backlight. Therefore, in order to run this program on the personal computer, SYSDRVP.COM must reside on it.

The following shows the result after the system driver was installed as the resident simulation driver and the sample program was executed.
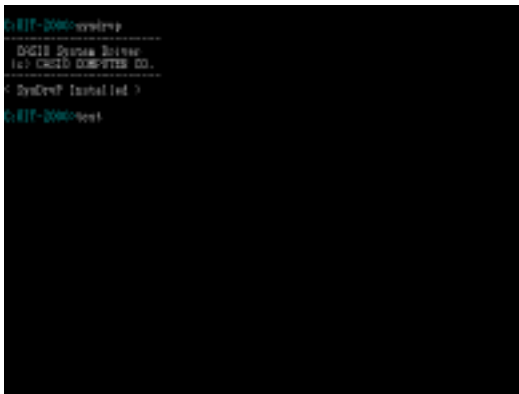
Fig. 7.4

Under this condition the program simply waits for key input. The backlight is off. To confirm this condition use the monitor function of the simulator. Under the default condition and if the F9 key is pressed, the internal status of the simulator is displayed on the far right of the screen, as shown below. The fourth line of the display indicates the backlight status, which is currently OFF.
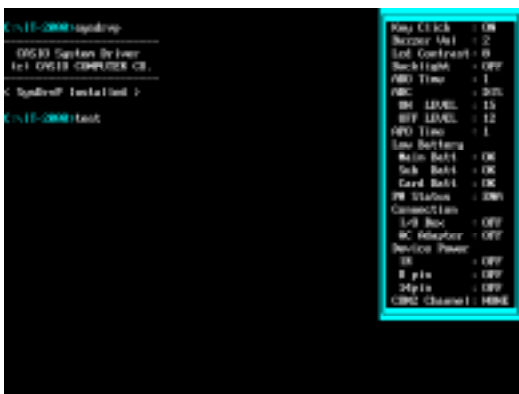
Fig. 7.5

In this condition press the "1" key. The sample program shown above is designed so that the backlight is turned on if it receives "1". Here, in order to make sure that the program operated normally, press the F9 key again to refresh the simulator status display. The result is shown below. Now, the fourth line of the display indicates that the backlight is ON.
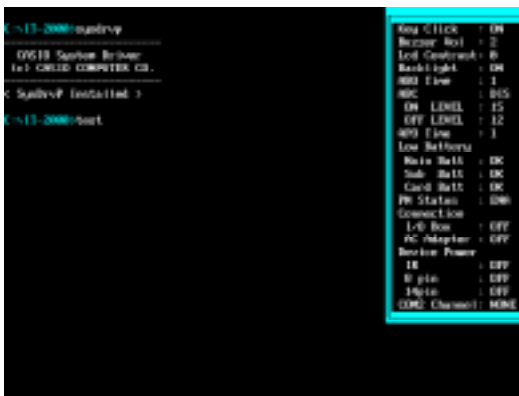
Fig. 7.6

The outline of the operation test using the simulation driver is summarized above. Debugging can of

course be performed using Microsoft's CodeView debugger.

For more information about each simulation driver refer to Chapter 7.5 "Simulation Driver".

7.4.3   Operation Check on IT-2000 (Using COM2KEY/XY)

If software coordination through simulation has been completed, it should be transferred onto the IT-2000 for operation checks. To do this use the COM2KEY utility. The COM2KEY utility will, when the COM port of a personal computer is connected with the IT-2000 via the dedicated cable (DT-9689AX), use the personal computer as a dumb terminal of the IT-2000. For more information about the COM2KEY utility refer to Chapter 8.9 "COM2KEY Utility".

A batch file (1.BAT) for initiating this COM2KEY utility is installed on the basic drive of the IT-2000. It can be initiated by using the appropriate numeric key while the IT-2000 is in the command prompt state. Since the major purpose of this utility is in assisting application development, it can be directly called from AUTOEXEC.BAT. Moreover, it can be registered as a device driver.

This is convenient for developing device drivers to be registered in CONFIG.SYS. In this case, register the COM2KEY utility before registering a developed device driver, and redirect the COM2KEY utility to the personal computer with the start-up message of the device driver.

The following is the program transfer procedure used with the COM2KEY utility.

- Connect the personal computer and IT-2000 with the dedicated cable (DT-9689AX).
- Initiate the terminal software on the personal computer side and establish communication at 9600 bps. There are no particular requirements for use of the terminal software. YMOEM/bat protocol is available.
- Initiate COM2KEY on the IT-2000. For information about the initiation method, refer to Chapter 8.9 "COM2KEY Utility".
- With the above procedure the command prompt of the IT-2000 will appear on the terminal software screen. Under this condition initiate the XY utility and perform the program transfer, as follows:

```
D:\>
D:\>xy/ry /N
```

- After the above operation has been performed the IT-2000 remains in the wait state for file reception with the YMOEM/bat protocol. Use the upload function of the terminal software to transfer the application program.
- If file transfer has been completed, the operation check of the program can be performed. Of course, this application program can be initiated with a command line displayed by the terminal software on the personal computer.

```
D:>aplic
```

● Use the debugger as required. DEBUG.COM, which is the standard MS-DOS debugger, is stored on the backup CD-ROM. If it is transferred onto the user disk by the above mentioned procedure and initiated according to the following procedure, remote debugging is made possible on the terminal software of the personal computer.

```
D:>debug aplic.exe
-r
AX=0000  BX=0000  CX=27C7  DX=0000  SP=0800  BP=0000  SI=0000  DI=0000
DS=2DEF  ES=2DEF  SS=30BC  CS=2DFF  IP=0420   NV UP EI PL NZ NA PO NC
2DFF:0420 B430          MOV   AH,30                      ;'0'
```

7.4.4 Installation of Application Program

This section describes how to install the application program, after it has been debugged, on the actual terminal. The following table summarizes IT-2000 installation required after purchase.

    (1)   Installation of main battery and sub-batteries

    (2)   Calibration

    (3)   Formatting the F-ROM drive (only for models with an F-ROM drive).

    (4)   Setting the RAM disk size and formatting it (if the RAM disk is used)

    (5)   Setting the system time

    (6)   Other various setups including the Auto Power OFF time, etc.

    (7)   Copying application programs, CONFIG.SYS, AUTOEXEC.BAT, etc.

This section mainly explains about point (7) in the above table. For information about (2) through (6), refer to Chapter 3 "System Menu".

There are three ways of installing applications in the IT-2000. Each is explained in detail below:

    ●   Installation with a PC card

    ●   Installation from a PC

    ●   Copying application program onto another IT-2000

● **Installation with a PC card**

This method is used to automatically install the application using the card boot function. To do this, first create an ATA card for card boot and store the developed application program on it. Then provide a line through which to copy the application program into the IT-2000 in the AUTOEXEC.BAT file that will be executed at card boot.

How to create a card for installation :

- Make an appropriate directory on the ATA card and copy the application program, files that are used by this application program, CONFIG.SYS, and AUTOEXEC.BAT onto this directory.
- Create CONFIG.SYS and AUTOEXEC.BAT for card boot. At the end of AUTOEXEC.BAT add a line for copying the above mentioned directory wholly onto the user disk.
- The above steps complete the creation of a card for installation.

Installation work :

- In the slot, insert the ATA card that has been created for installation and lock the card lock switch. If the terminal power is currently on, turn it off. Then press the RESET button to initiate the System Menu. Turn the Power switch to OFF and then to ON. The card boot process will take place.
- If the batch files called from AUTOEXEC.BAT have been successfully executed, installation of the application has been completed.

- **Installation from a PC**
  This method is used to directly transfer the appropriate files from the PC to the IT-2000 using the serial cable or I/O Box. For information about this method of file transfer from the PC refer to Chapter 3.10 " YMODEM Utility", or Chapter 3.11 "FLINK Command".

- **Copying application program onto another IT-2000**
  This method is used to mirror-copy the entire contents of the F-ROM drive of one IT-2000 to another IT-2000. If an application has been installed on one IT-2000 the application can be installed on another IT-2000. No accessories, such as a card or cable, are required.
  For more information refer to Chapter 3.11 "FLINK Command".

7.5   Simulation Driver

The simulation driver is used to develop on a personal computer the application programs that run on the IT-2000.

The application development libraries supported for this terminal control only the hardware that is compatible with the IBM PC/AT. This is important to remember if the application programs for the terminal are developed on a personal computer. Although each library is linked to the application program to form an executable program, they do not contain code that is specific to the hardware of the handy terminal. Consequently, if a simulation driver is used rather than one of the drivers dependent on the terminal hardware, the application program can be made to run, without modification, on the personal computer. This is the basic concept of simulation.

The shaded portion in the diagram below shows the simulation drivers to be replaced However, for the mouse driver, use Microsoft's mouse driver. This is because the mouse driver on the terminal simulates the operations of the Microsoft Mouse driver, while other drivers simulate the actual operations on the terminal.
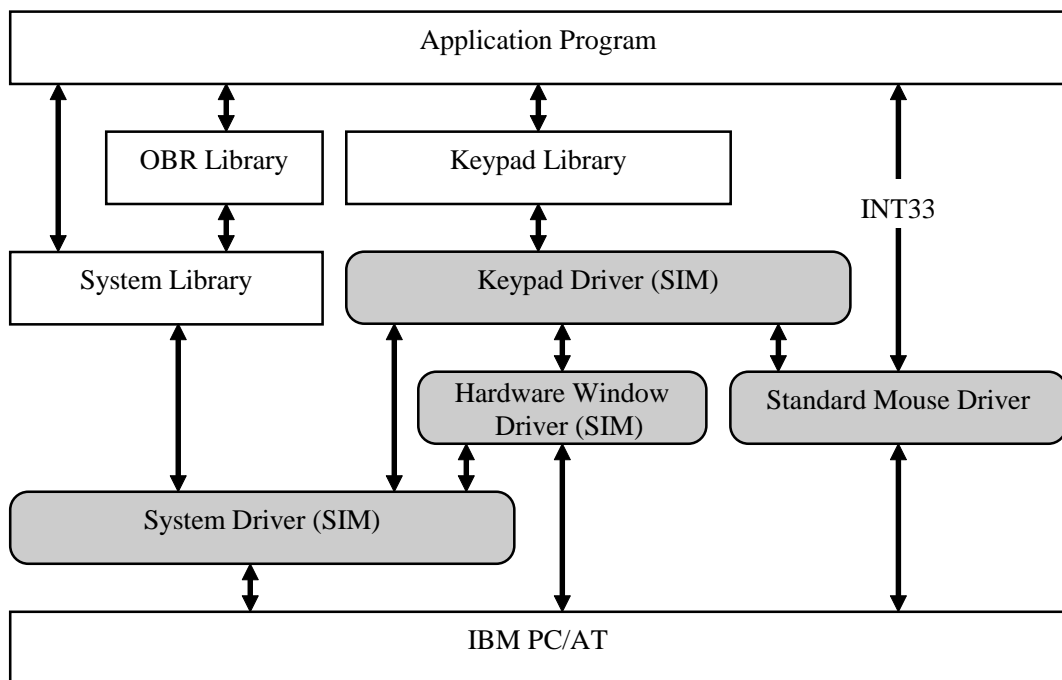
Fig. 7.7

In the following chapters, these simulation drivers are explained in detail.

7.5.1   System Driver Simulator (SYSDRVP.COM)

Overview

This simulation driver simulates the operation of the system driver on the personal computer.

It also contains the run units of various functions supported by the system library.

File name

SYSDRVP.COM

Function

Under the default condition and if the F9 key is pressed, the internal status of the simulator is displayed at the far right of the screen. The displayed internal status includes the ON/OFF condition of the backlight and the setup of the enabling/disabling key sensing sound, etc., which can be set with the system library.
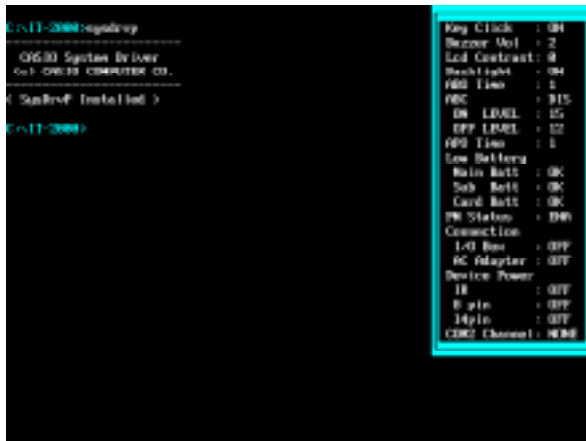
Fig. 7.8

Some of the values in Fig. 7.8 above which can be monitored are possibly changed by external operations. For example, one of them is the backlight ON/OFF by Fn+7 keys. F10 key (as default) is used to simulate the change of the status by the external operations. Pressing the F10 key will display changeable items as a pop-menu on the monitor screen. Move the cursor onto your desired item to change the status.
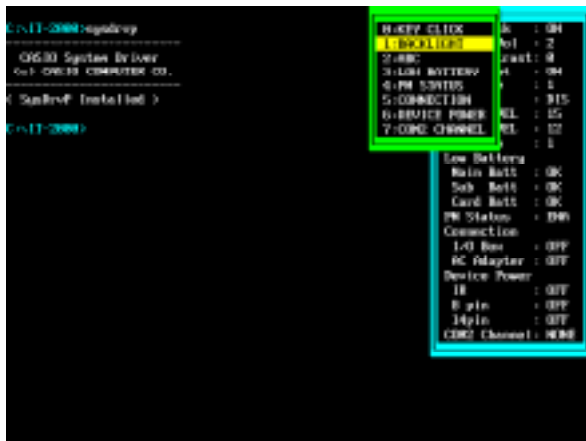
Fig. 7.9

Startup Method

This utility is included in the SDK. Since this is a memory-resident type COM execution file, it should only be used if it resides in memory.

Startup Option

Format: SYSDRVP [Options]

Options	/R	Releases residence, if it is currently resident.
	/H1=code	Specifies a hot key which displays internal status. Code of the hot key can be specified by a value in hexadecimal number which is returned with INT16 (AH=10h). When this option is omitted, the operation is the same with that of which "/H1 = 4300" is specified.
	/H2=code	Specifies a hot key which displays status change menu. Code of the hot key can be specified by a value in hexadecimal number which is returned with INT16 (AH=10h). When this option is omitted, the operation will be the same with that of which "/H2 = 4400" is specified.

Termination Codes and Messages

Termination code	Message	Description
00	NORMAL END	End normally.
01	ABNORMAL END	Parameter error.

7.5.2   Hardware Window Manager Simulator (HWWMANP.COM)

Overview

This simulator simulates only the display functions of the keypad driver on a PC. It cannot function by itself. For information about the display functions of the keypad driver see below.

File name

```
HWWMANP.COM
```

Startup Method

This utility is included in the SDK. Since this is a memory-resident type COM execution file, it should only be used if it resides in memory.

Startup option

```
Format: HWWMANP [Options]
```

Options	/R	Releases residence, if it is currently resident.
	/G[n]	If the video mode is set to 12H, this function will display the hardware window in a graphic image. If the [n] option is specified, the display position shifts to the right by the specified number of dots. This specification should be made using a decimal number to indicate the number of dots by which to shift in the X direction (default is 0). The specified number of dots will be truncated to a multiple of eight. This setup should be consistent with that made by KEYPADP /G.
	/H	Displays the Help screen.

Operating Conditions

Operation of this utility requires the following driver. Before initiating this utility install it in memory.

System driver simulator (SYSDRVP.COM)

Termination Codes and Messages

Termination code	Message	Description
00	NORMAL END	Normal termination
01	ABNORMAL END	SYSDRVP is not installed. Parameter error

7.5.3　Keypad Driver Simulator (KEYPADP.COM)

Overview

This simulator will simulate the keypad driver and keypad library on a PC.

File name

```
KEYPADP.COM
```

Function

| Key input	Simulates on PC keyboard input and keypad input on the screen using the keypad library.
Coordinate input	Simulates on a PC coordinate input from the mouse using the keypad library.
Keypad display	Simulates the keypad driver on PC
Keycode input	Supports input of 2-byte hexadecimal codes.

Startup Method

This utility is included in the SDK. Since this is a memory-resident type COM execution file, it should always be installed in memory.

Startup option

```
Format: KEYPADP [Options]
```

Options	/R	Releases residence, if it is currently resident.
	/G[n]	Shifts the X-axis origin to the right by the specified number of dots. This specification should be made using a decimal number indicating the number of dots by which shift in the X direction (default is 0). This setup should be consistent with that made by HWWMANP　/G.
	/H	Displays the Help screen.

Operating Conditions

Operation of this utility requires the following drivers. Before initiating this utility always install them in memory. Observe the following order of installation.

- System driver simulator (SYSDRVP.COM)
- Hardware window manager simulator (HWWMANP.COM)

This utility uses 2 pages (32 KB) of EMS memory. The MOUSE driver must be enabled.
The KEYPAD.DAT file must be located in the same directory as KEYPADP.COM.

Termination Codes and Messages

Termination code	Message	Description
00	NORMAL END	Normal termination
01	ABNORMAL END	SYSDRVP or HWWMANP is not installed. EMS memory driver was not enabled. KEYPAD.DAT not found. Parameter error

About the Key Input
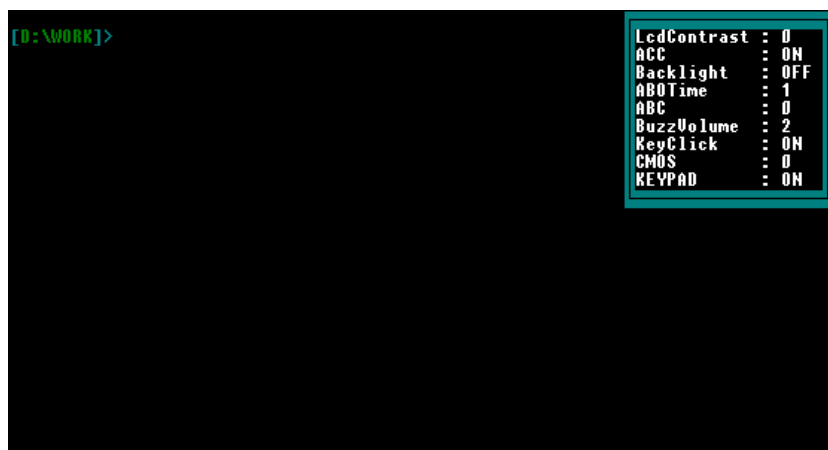
If this simulator is resident in memory, input through the keypad is permitted by clicking in the region that corresponds to each key, in addition to key input via the PC keyboard. (This is possible irrespective of whether a simulated keypad is being displayed by the G/ option.)

About Coordinate Input

If the application is designed for input of coordinates through the KEY_Read function of the keypad library, this simulator will implement the same process using the mouse pointer. In this case, if the left mouse button is clicked DOWN/RUN/UP operations will be simulated. Right mouse button clicks will be ignored.

Monitoring Keypad Display ON/OFF Status

If the F9 key is pressed, this simulator will show the keypad display ON/OFF status along with other system library information.

Fig. 7.9

Displaying Simulated Keypad

If this utility is initiated with the G/ option, a simulated keypad will be drawn in the position specified by the X coordinate provided that the video mode of the VGA BIOS (INT10h) is set to 12h (16-color graphic mode of 640 x 480 dots).
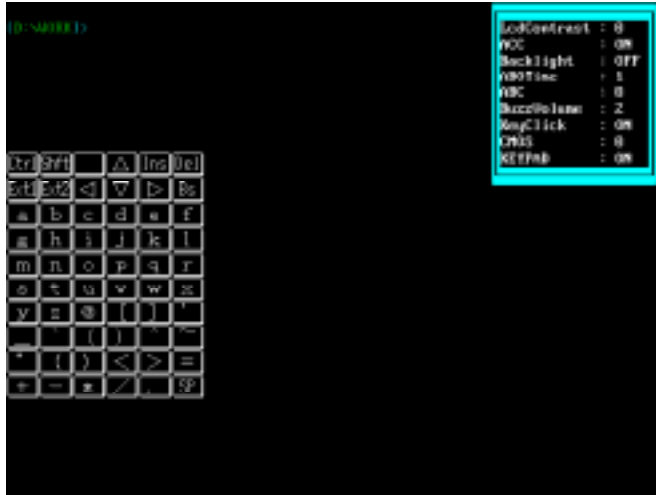
Fig. 7.10

Since this display position in the X direction can be modified with the /G option at the start of the resident utility, it is also possible to display the simulated keypad so it will not be overwritten by other displays.
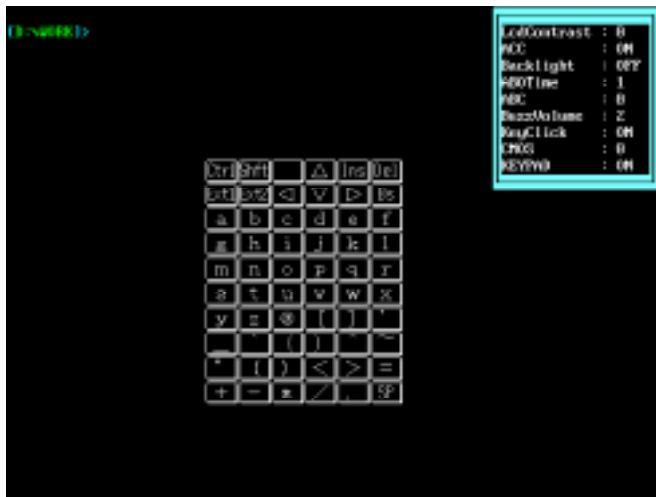
Fig. 7.11

Note:

Displaying this simulated keypad may overwrite the display results made by the application.

An identical /G option must always be specified for the simulators of the hardware window manager and the keypad driver.

About Hexadecimal Code Input

Even if the keypad is active, key input using the mouse is disabled for the simulator on the PC if the /G option is not specified. To avoid this conflict, it is possible to enter a hexadecimal code with the menu accessed from the F10 key where the keypad is active. Enter 2 bytes of normal keycode for each hexadecimal code.

7.6 Library

7.6.1 Overview

Since the IBM PC/AT architecture has been adopted in this system, all libraries including graphic library supported by Microsoft C/C++ ver. 7.0 or later versions can be used. In addition to those, the following dedicated libraries are available for the IT-2000 system.

Name of library	Description	Page
System Library	Dedicated libraries for IT-2000 and to control various devices available to the system. These libraries include back light control, contrast control, battery voltage low detection, alarm setting, etc.	131
Keypad Library	Libraries to call the functions of Keypad driver. They are used to input keys through keypad and to acquire coordinates on screen, etc.	153
OBR Library	Libraries to control the OBR functions. OBRs supported by the system are the pen type and the CCD type.	170

7.6.2   System Library

Overview

The IT-2000 has various types of devices that can be controlled by the HT-generic software which is built upon the base of PC/AT architecture. This library is used to control these dedicated devices from application programs developed with the C language.

Since this library provides lower-level programs concerning the hardware and system operations, exercise great care if integrating them into your programs.

Note About the Libraries

The system library is supplied on one of the three memory models, suitable for the application program developed by the user. Any application program that uses the system library should include **syslib.h** in the corresponding source file. Constants to be passed to the library functions and their prototypes are defined in this header file.

```
SYSLIB.H ......................  Header file for system library
SLIBSYSD.LIB .............  System library for small-memory model
MLIBSYSD.LIB ............  System library for medium-memory model
LLIBSYSD.LIB .............  System library for large-memory model
```

Note:

In application programs which perform communication incorporate **SYS_SetPMStatus()** to disable the power control capability. If **SYS_SetPMStatus()** is set to active, the CPU speed is lowered to reduce battery consumption to a minimum, which may result in a communication error.

List of Libraries

The following functions are supplied in the system library:

Function	Page	Function	Page
Acquisition of BIOS Version	132	Software Card Lock	143
Acquisition of Memory Device Size	133	Acquisition of Connector Status	144
Setting/Acquisition LCD Contrast	133	Key Click Sound ON/OFF	144
Increasing/Decreasing LCD Contrast	134	Acquisition of Key Click Sound Status	144
Switching Over COM2 Channel	135	Acquisition of Reboot Reason	145
Setting/Acquisition Reason Mask for Reboot	136	Acquisition of OFF Reason	145
Reboot Request	136	Setting Cancellation of Next Resume Process	146
Setting ABO Time	137	Acquisition of Cancellation Status of Next Resume Process	146
Acquisition of ABO Time	137	Request of Suspend (Software OFF)	147
Setting ABC status	138	Acquisition of Low Battery Voltage Status	147
Acquisition of ABC (Auto backlight Control) Status	138	Setting APO Time	147
Setting/Acquisition of ABC Threshold	139	Acquisition of APO Time	148
Backlight ON/OFF	140	Setting Status of Alarm	148
Acquisition of Backlight Status	140	Acquisition of Alarm Setting	149
Setting Buzzer Volume	141	Resetting Alarm	150
Acquisition of Buzzer Volume	141	Setting/Acquisition of Power ON Alarm	150
Acquisition of Device Power Status	142	Setting/Acquisition of Status of Power Control Function	151
Device Power ON/OFF	142	Setting Key Click Sound ON	152

Acquisition of BIOS Version

Acquires the ROM BIOS version number, which consists of the following three numbers:

major number, minor number, sub-number.

SYNTAX

    long SYS_GetBiosVersion();

INPUT

    None

OUTPUT

    b23 to b16   Major number

    b15 to b8   Minor number

    b7 to b0   Sub-number

Acquisition of Memory Device Size

If the memory device size is designated, the total capacity of the DRAM and the number of
NAND FROM chips is read. The memory device size is the total capacity of all the physically
installed devices, and not the disc capacity.

> SYNTAX
>
> > **int SYS_GetMemCapacity(int nDevice);**
>
> INPUT
>
> > nDevice = device type
> >
> > > 0   DRAM
> > >
> > > 1   NAND FROM
>
> OUTPUT
>
> > = -1   Input parameter error
> >
> > <> -1   DRAM size (by the unit of 1K)(where nDevice = 0)
> >
> > Actual installed number of NAND chips (where nDevice = 1)

Setting/Acquisition LCD Contrast

The contrast of the LCD display is affected and varied by the ambient temperature. Therefore,
this terminal automatically detects the variation of ambient temperature and determines an optimal
contrast based on the acquired data. However, it may not immediately react to rapid temperature
variations or be ideal for each specific user. With this in mind, the terminal is provided with a means
to increase or decrease the LCD contrast manually.

The contrast value returned by this function is a correction value to the optimum contrast that has
been determined by automatic calculation. The value returned will be zero if no correction is made
manually; +1 or -1 will be returned if the contrast is increased or decreased by one step, respectively.
The range of contrast values that can be set or read varies according to the ambient temperature.
This is because the range of setup values that can be set for the hardware is between 0 and 31. If, for
example, the automatically calculated value is 10, the possible correction range is between - 10 and
+21. Consequently, the range of contrast values that can be read or set is between - 31 and +31.

The practical use of this library function lies in saving or loading the contrast setup prior to using the
SYS_LcdContrastUp() or SYS_LcdContrastDown() function.

> SYNTAX
>
> > **int SYS_GetLcdContrast(int WINFAR *nValue);**
>
> INPUT
>
> > nValue = Pointer to the area where the current correction value is acquired.
>
> OUTPUT
>
> > =   0   Normal
> >
> > = -2   No response from KBC
> >
> > = -3   VxD not registered (for IT-2000W only)

SYNTAX

      `int SYS_SetLcdContrast(int nValue);`

INPUT

      nValue = Correction value to be set

OUTPUT

      = 0   Normal

      = -2   No response from KBC.

      = -3   VxD not registered (for IT-2000W only)

Note:

"WINFAR" will be treated as a far pointer only on IT-2000W models.

Increasing/Decreasing LCD Contrast

The contrast of the LCD display varies with the ambient temperature. Therefore, this terminal automatically detects the ambient temperature and determines an optimal contrast based on the acquired data. However, it may not immediately react to rapid temperature variations or be ideal for each specific user. This function is used to correct the contrast value, which has been automatically calculated by the system, to an optimal level.

The resulting contrast value adjusted using this function can be acquired via the **SYS_GetLcdContrast**() function.

SYNTAX

      `int SYS_LcdContrastUp();`

INPUT

      None

OUTPUT

      = 0   Normal

      = -2   No response from KBC

      = -3   VxD not registered (for IT-2000W only)

SYNTAX

      `int SYS_LcdContrastDown();`

INPUT

      None

OUTPUT

      = 0   Normal

      = -2   No response from KBC

      = -3   VxD not registered (for IT-2000W only)

Switching Over COM2 Channel

IR, 14-pin, or 3-pin communication interface can be selected on the COM2 port. However, since the 3-pin interface is an optional means to maintain software compatibility with other models, it is not implemented on this terminal.

SYNTAX

int SYS_GetCOM2Config();

INPUT

None

OUTPUT

= 0   Not selected (default setting at RESET)

= 1   14-pin

= 2   3-pin

= 3   IR

SYNTAX

int SYS_SetCOM2Config(int nDevice);

INPUT

nDevlce = Device to be used

0   Not used

1   14-pin

2   3-pin

3   IR

OUTPUT

= 0   Normal

= -1   Parameter error

Note:

This function is not related to the device power control. As a result, this function does not need to be restored to the "Not used" condition after the device has been used.

Setting/Acquisition of Reason Mask for Reboot

To acquire the reboot request reason, enable or disable "mounting on I/O Box" or use of the CI signal for boot-up.

SYNTAX

int SYS_GetOnEventMask();

INPUT

None

OUTPUT

b0 = 0		Enable use of ring signal
	1	Disable use of ring signal
b1 = 0		Enable use of "mounting on I/O Box"
	1	Disable use of "mounting on I/O Box"

SYNTAX

int SYS_SetOnEventMask(int nMask);

INPUT

nMask = Setting the reboot reason mask

b0 = 0		Enable use of ring signal
	1	Disable use of ring signal
b1 = 0		Enable use of "mounting on I/O Box"
	1	Disable use of "mounting on I/O Box"

OUTPUT

= 0   Normal

= -1   Parameter error

Reboot Request

This function is used to restart (reboot) the system without suspending IT-2000 operations.

SYNTAX

int SYS_Reboot(int nMode);

INPUT

nMode = Reboot type

0   Initiates the application.

1   Initiates the system menu.

OUTPUT

= 0   Normal

= -1   Parameter error

Setting ABO Time

The ABO (Auto Backlight OFF) function is used to automatically turn off the backlight if neither key entry nor touch-panel entry is permitted for a certain period of time. This function is used to set the ABO time. Enable ABO by selecting a number between 1 and 15, which corresponds to a period of between 20 seconds and 5 minutes.

SYNTAX

`int SYS_SetAboTime(int nValue);`

INPUT

nValue = ABO time

    0   Not activate ABO

    1 to 15   Activates ABO in specified number x 20 seconds.

OUTPUT

    = 0   Normal

    = -1   Parameter error

    = -3   VxD not registered (for IT-2000W only)

Note:

This function will be implemented by a software timer. Therefore, the period until the backlight is actually turned off has an error of +/- 10 % associated with it.

Acquisition of ABO Time

This function is used to read the ABO setting.

SYNTAX

`int SYS_GetAboTime();`

INPUT

None

OUTPUT

    = 0   Not activate ABO

    = 1 to 15   ABO time in units of 20 seconds

    = - 2   No response from KBC

    = - 3   VxD not registered (for IT-2000W only)

Setting ABC (Auto Backlight Control) Status

The ABC (Auto Backlight Control) function is used to sense the ambient light intensity and automatically turns ON/OFF the backlight. This function is used to enable or disable the ABC function.

SYNTAX

`int SYS_SetABC(int nOnOff);`

INPUT

nOnOff = 0   OFF

   Other than 0 ON

OUTPUT

= 0 Normal

= -1   Parameter error

= -2   No response from KBC

= -3   VxD not registered (for IT-2000W only)

Acquisition of ABC (Auto Backlight Control) Status

The ABC (Auto Backlight Control) function is used to sense the ambient light intensity and automatically turns ON/OFF the backlight. This function acquires the current setting of the ABC function.

SYNTAX

`int SYS_GetABC();`

INPUT

None

OUTPUT

0   ABC in OFF status

1   ABC in ON status

2   ABC temporarily disabled

- 2   No response from KBC

- 3   VxD not registered (for IT-2000W only)

Setting/Acquisition of ABC Threshold

The ABC (Auto Backlight Control) function is used to sense the ambient light intensity and automatically turns ON/OFF the backlight. This function is used to set marginal levels across which the backlight changes from ON to OFF or from OFF to ON.

If the readout on the AD converter falls below OnValue, the backlight turns on, and if it exceeds OffValue, the backlight turns off. If these two levels are identical or too close each other, the backlight may flicker. To avoid this problem set OnValue so that it is slightly less than OffValue.

SYNTAX

int SYS_SetThresholdOfABC(int OnValue, int OffValue);

INPUT

OnValue =   0 to 255

OffValue =   0 to 255

OUTPUT

=   0   Normal

= - 2   No response from KBC

= - 3   VxD not registered (for IT-2000W only)

SYNTAX

int SYS_GetThresholdOfABC(int *OnValue, int *OffValue);

INPUT

OnValue = Pointer to the area in which the ON threshold value is stored.

OffValue = Pointer to the area in which the OFF threshold value is stored.

OUTPUT

= 0   Normal

= -2   No response from KBC

= -3   VxD not registered (for IT-2000W only)

Backlight ON/OFF

This function is used to forcibly turn ON or OFF the backlight. If turned ON by this function, the backlight will remain on until Backlight OFF is triggered by the Backlight OFF function or ABO. If this function is activated under the ABC control, the ABC will be temporarily disabled, and will be enabled again when Backlight OFF is triggered by the Backlight OFF function or ABO.

SYNTAX

`int SYS_SetBacklight(int nOnOff);`

INPUT

nOnOff = 0   OFF

         1   ON

OUTPUT

= 0   Normal

= -2   No response from KBC

= -3   VxD not registered (for IT-2000W only)

Acquisition of Backlight Status

This function acquires the current backlight status.

SYNTAX

`int SYS_GetBacklight();`

INPUT

None

OUTPUT

=   0   Backlight OFF

=   1   Backlight ON

= -2   No response from KBC

= -3   VxD not registered (for IT-2000W only)

Setting Buzzer Volume

Sets the buzzer volume to one of four levels: Large/Medium/Small/OFF.

SYNTAX

int SYS_SetBuzzerVolume(int nVolume);

INPUT

nVolume = 0   OFF

1   Small

2   Medium

3   Large

OUTPUT

= 0   Normal

= -1   Parameter error

= -2   No response from KBC

= -3   VxD not registered (for IT-2000W only)

Acquisition of Buzzer Volume

Acquires the buzzer volume as one of four levels: Large/Medium/Small/OFF.

SYNTAX

int SYS_GetBuzzerVolume();

INPUT

None

OUTPUT

0   OFF

1   Small

2   Medium

3   Large

-2   No response from KBC

-3   VxD not registered (for IT-2000W only)

Acquisition of Device Power Status

Acquires the current power conditions (ON/OFF) of each device.

SYNTAX

int SYS_GetDevicePower(int Device);

INPUT

Device = device to be selected

2	IrDA
3	14-pin I/F
5	8-pin I/F
Other	Reserved

OUTPUT

1   Power ON

0   Power OFF

Note:

This function is used to control the power to devices of this system. Never designate parameters other than those specified on this page.

Device Power ON/OFF

Used to turn ON and OFF the power of each device.

SYNTAX

int SYS_SetDevicePower(int Device, int OnOff);

INPUT

Device = device to be selected

2	IrDA
3	14-pin I/F
5	8-pin I/F
Other	Reserved

OnOff = ON/OFF setting

0   Turns OFF.

1   Turns ON.

OUTPUT

0   Normal termination

Note:

This function is used to control the power to the devices in this system. Never designate parameters other than those specified on this page.

Software Card Lock

Sets or acquires the Lock/Unlock status of the software-type card lock switch.

This machine has a card lock mechanism that is on the card case to prevent accidental removal of the card. This mechanism has a software driver that detects the released state of this lock and executes the appropriate file closing procedure. However, some types of cards, depending on the card shape, can not be fastened by the lock switch. If this is the case, even if a card is present it will not be detected. This function is provided to handle this type of card.

To use a card for which the card lock mechanism can not be used, call this function in advance to set the software lock switch to ON. Now a card can be detected when it is inserted or removed.

SYNTAX

```
int SYS_SetCardLock(int OnOff);
```

INPUT

OnOff =   Card lock ON/OFF

0   Unlock

Other than 0   Lock

OUTPUT

0   Normal termination

Logic Circuit of Software Card-Lock
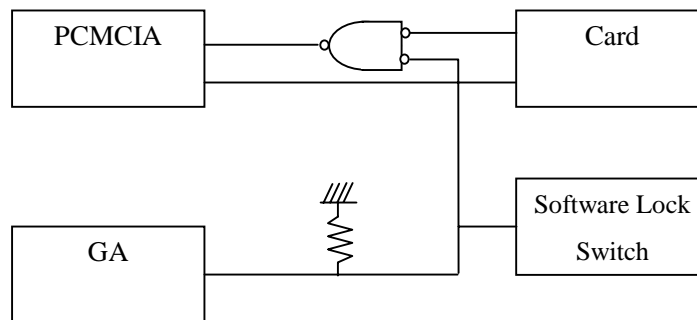
Fig. 7.12

Acquisition of Connector Status

Acquires the connection setting of the I/O Box and AC adaptor.

SYNTAX

int SYS_GetConnectorStatus(int nType);

INPUT

nType = Connector type

    0   I/O Box

    1   AC adaptor or I/O Box

OUTPUT

= 0   Normal termination

= 1   Connected

= -1   Parameter error

Key Click Sound ON/OFF

Sets the key click sound to ON or OFF.

SYNTAX

int SYS_SetKeyClck(int OnOff);

INPUT

nOnOff =   0   OFF

          Other than 0   ON

OUTPUT

=   0   Normal

= -2   No response from KBC

= -3   VxD not registered (for IT-2000W only)

Acquisition of Key Click Sound Status

Acquires the key click sound ON/OFF setting.

SYNTAX

int SYS_GetKeyClick();

INPUT

None

OUTPUT

= 0   OFF

= 1   ON

= -2   No response from KBC

= -3   VxD not registered (for IT-2000W only)

Acquisition of Reboot Reason

Used to acquire the reason the system was rebooted.

SYNTAX

int SYS_GetPowerOnFactor();

INPUT

None

OUTPUT

b0   Power key

b1   Reset button

b2   Alarm

b3   Ring signal

b4   IT-2000 is being set on I/O Box

Note :

If the reset button is pressed the system menu is initiated. This means that an application program will never acquire the status of "Reset switch being pressed" as the reboot reason.

Acquisition of Reason for OFF

Acquires the reason that the system was most recently turned OFF.

SYNTAX

int SYS_GetPowerOffFactor();

INPUT

None

OUTPUT

b0	Power key
b1	Reset key
b2	Reserved
b3	LBO
b4,b6, b7	Reserved
b5	LB1 timeout (indicates "OFF" by the condition of battery voltage low.)
b8	APO
b9	Software-triggered OFF
b10 to b15	Reserved

Note:

● If the reset switch is pressed the system menu is initiated. This means that an application program will never acquire the status of "Reset switch being pressed" as the reboot reason.

● If the system is rebooted, the reason it was set OFF will be cleared. Therefore, zero will be acquired if the reason it was set OFF is read for the first time after rebooting.

● If "Cancellation of the next resume process" is set as the reason the power was set OFF (including Power key, APO, Software-triggered OFF, etc.), the reason it was set OFF will be cleared during the reboot process.

Setting Cancellation of Next Resume Process

Sets the power-on process (Resume/Boot) for each power OFF reason. The default setting is Resume On.

SYNTAX

```
int SYS_SetResumeCondition(int nCondition);
```

INPUT

nCondition =	b0	Power key	0 = Resume On, 1 = Boot
	b1 to b7	Reserved	
	b8	APO	0 = Resume On, 1 = Boot
	b9	Software-triggered OFF	0 = Resume On, 1 = Boot
	b10 to b15	Reserved	

OUTPUT

= 0   Normal

= -1   Parameter error

Note :

With this function the power-on process can be set for each of the reasons the power is turned OFF: Power key, APO, and Software-triggered OFF. Therefore, if set to "The next power-on process is boot" from the application, it is necessary to specify all three parts with the corresponding bits.

Acquisition of Cancellation Status of Next Resume Process

Acquires the power-on process setting (Resume On/Boot) for each power-off reason.

SYNTAX

```
int SYS_GetResumeCondition();
```

INPUT

None

OUTPUT

b0	Power key	0 = Resume ON, 1 = Boot
b1 to b7	Reserved	
b8	APO	0 = Resume ON, 1 = Boot
b9	Software-triggered OFF	0 = Resume ON, 1 = Boot
b10 to b15	Reserved	

Request of Suspend (Software-triggered OFF)

Used to turn off the system with the software. If there is a need to specify the next boot-up process, complete "Setting Cancellation of Next Resume Process" beforehand, then call this function.

SYNTAX

`void SYS_PowerOff();`

INPUT

None

OUTPUT

None

Acquisition of Low Battery Voltage Status

An APM (Advanced Power Management) BIOS has been installed in this terminal. This function is used to directly refer the hardware conditions which are translated into input signals for the APM BIOS.

SYNTAX

`int SYS_GetLBStatus();`

INPUT

None

OUTPUT

b0	Reserved
b1	**LB1 event**: Main battery voltage low.
b2	**LB2 event**: Sub-battery voltage low.
b3	**LB3 event**: Memory card battery voltage low.
b4 to b7	Reserved

Setting APO Time

Used to set a time until APO (Auto Power OFF) occurs.

SYNTAX

`int SYS_GetApoTime (int nValue);`

INPUT

nValue = APO time

0	Does not cause APO.
1 to 15	Causes APO in the specified-number of minutes plus 30 seconds.

The actual APO time has an error of +/- 25 seconds.

OUTPUT

    =   0   Normal

    = -1   Parameter error

Note :

Auto Power OFF will work if the power control function is active. For more information about the power control function refer to "Setting/Acquisition of Status of Power Control Function".

Acquisition of APO Time

Acquires the currently set APO time.

SYNTAX

    int SYS_GetApoTime();

INPUT

    None

OUTPUT

    0   Disable the APO.

    1 to 15   Enable the APO in the specified-number of minutes plus 30 seconds.

The actual APO time has an error of +/- 25 seconds.

Note :

Auto Power OFF will work if the power control function is active. For more information about
the power control function refer to "Setting/Acquisition of Status of Power Control Function".

Setting Status of Alarm

This function is used to set the alarm so that Int4Ah will be executed at the specified time. If the set time precedes the currently set RTC (Real Time Clock) time, the alarm will be valid on and after the following day. If the setup time is later than the currently set RTC time, the alarm will be valid from the specified day. To make this possible the user has to set the specified interrupt handling routine to Int4Ah. If this function is not reset using the **SYS_ResetAlarm()** function, the alarm will activate (repeatedly set) for each 24-hour period. Call the **SYS_SetPowerOnAlarm()** function to turn on the system at the alarm time specified by this function.

SYNTAX

        `int SYS_SetAlarm(int hour, int min, int sec);`

INPUT

        hour = hours   (in decimal number)

        min = minutes   (in decimal number)

        sec = seconds   (in decimal number)

OUTPUT

        0   Normal

        < 0   Error (error within INT1Ah)

Note:

- This function simply calls INT1AH (AH = 6) internally. Therefore, if this function or INT1Ah (AH=6) is called and if the alarm has already been set, an error results.
- Note that the validity of parameters as time is not checked.

Acquisition of Alarm Setting

This function is used to acquire the current alarm setting made for the RTC (Real Time Clock).

SYNTAX

        `void SYS_GetAlarm(int *hour, int *min, int *sec);`

INPUT

        hour =   Pointer to the area from which hours is read.

        min =   Pointer to the area from which minutes is read.

        sec =   Pointer to the area from which seconds is read.

OUTPUT

        None

Note :

This function returns the time data set for the RTC. Note that the validity of data as time is not checked.

Resetting Alarm

This function prohibits an INT4Ah interrupt by internally calling INT1Ah (Ah = 7).

Note that neither the time data set for the RTC is erased nor is the power ON alarm setting for the

SYS_SetPowerOnAlarm() function canceled by this function. If this function is called with the

power ON alarm active, the alarm is temporarily reset. However, the RTC will be automatically set

to active after the power is turned off again to enable the power ON alarm.

The power ON alarm can also be canceled using the **SYS_SetPowerOnAlarm()** function.

SYNTAX

int SYS_ResetAlarm();

INPUT

None

OUTPUT

0   Normal

< 0   Error

Setting/Acquisition of Power ON Alarm

This terminal has a function to automatically turn on the power to the main unit at the specified time.

This function requires the RTC (Real Time Clock) function. Normally, an INT4Ah interrupt will

occur when the setting is being made on the RTC. This function makes it possible to add the

function which turns on the main unit at the desired time.

SYNTAX

int SYS_SetPowerOnAlarm(int OnOff);

INPUT

OnOff = Power On setup

0   Does not turn on the power.

Other than 0   Turns on the power.

OUTPUT

0   Normal

SYNTAX

int SYS_GetPowerOnAlarm();

INPUT

None

OUTPUT

0   =   Does not turn on the power.

Other than 0 =   Turns on the power.

Note :

The power ON alarm set with this function will be reset if rebooting occurs because the reset button is pressed or due to the software.

Setting/Acquisition of Status of Power Control Function

This terminal has incorporated unique power control functions: the auto power OFF mode and DOZE mode (CPU low-speed operation mode). Since these functions operate based on monitoring a period free from operator's concern over a given interval, they have the potential of affecting the execution performance of high-speed communication programs, including that of IrDA.

To create such a program call this function from it to disable the power control function.

If the power control function is set to disable, the monitoring of a period free from operator's concern is ceased, resulting in auto-power off not taking place. Since the switch to the DOZE mode does not occur either, the system can always be operable at high-speed. In short, this function is useful if auto-power OFF does not take place during processing, or if enhancing the processing speed.

SYNTAX

```
int SYS_GetPMStatus(void);
```

INPUT

None

OUTPUT

0 =   Disables power control

1 =   Enables power control

SYNTAX

```
void SYS_SetPMStatus(int OnOff);
```

INPUT

OnOff = Power control enable/disable

0   Disables power control

1   Enables power control

OUTPUT

None

Setting Key Click Sound ON

This function is used by application program to turn ON the key click sound. An example of the use is, when an button image on the LCD screen is touched it turns ON the sound. The sound is the same tone as those when ten key and keypad are pressed. The setting of key click sound ON/OFF controls this sound (refer to "Key Click Sound ON/OFF" on page 144.).

SYNTAX

`void SYS_MakeKeyClick();`

INPUT

None

OUTPUT

None

7.6.3 Keypad Library

Overview
This library is used to make an entry through the keypad or acquire the coordinates on the screen being touched.

Note about the Libraries
The key input library is supplied on one of the four models suitable for the application program developed by the user. Any application program that uses the key input library should include **keypad.h** in the corresponding source file. Constants to be passed to library functions and their prototypes are defined in this header file.

PADLIB.H	Header file for keypad
SLIBPADD.LIB	Keypad library for small-memory model
MLIBPADD.LIB	Keypad library for medium-memory model
LLIBPADD.LIB	Keypad library for large-memory model

This library will call the keypad driver internally. If this library is used, permanently install the hardware window manager (**HWWMAN.EXE**) and keypad driver (**KEYPAD.EXE**) in the system. In addition, the EMS driver should be registered in **CONFIG.SYS,** because the keypad driver saves the keypad image data in the EMS memory.

Example of CONFIG.SYS :
```
DEVICE=C:\EMM386.EXE  FRAME=C800 X=C000-C7FF  X=D800-DFFF
I=C800-D7FF
```

Example of AUTOEXEC.BAT :
```
C:\HWWMAN
C:\KEYPAD
```
APPLIC.EXE   ←-- Application program
```
C:\KEYPAD /r
C:\HWWMAN /r
```

Keypad Driver

As internal process, this library calls the keypad driver (KEYPAD.EXE). The relation among this library, the keypad driver and BIOS is as follow.
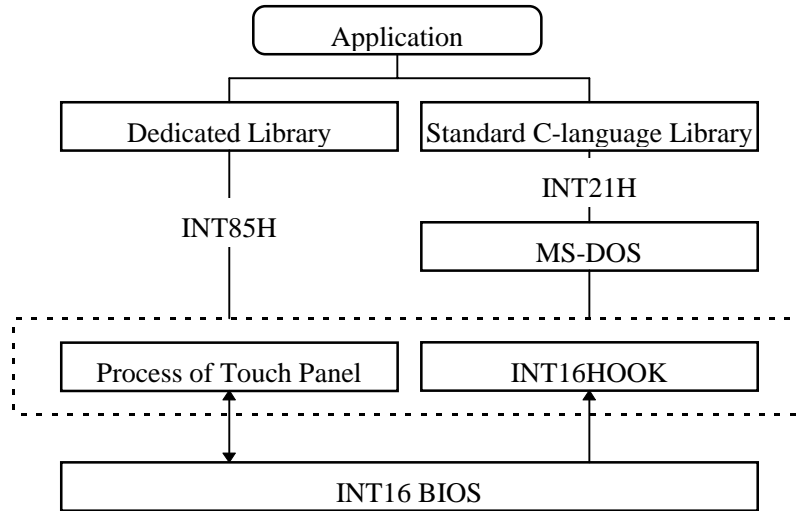
Fig. 7.13

Transition of Keypad State

The following shows the transition of keypad when it is operated.

Input pad for lower-case characters   **Input pad for upper-case characters**

←—Press Shift —→

Press Ctrl   Press Ctrl

Key Input   Key Input

Input pad for controls
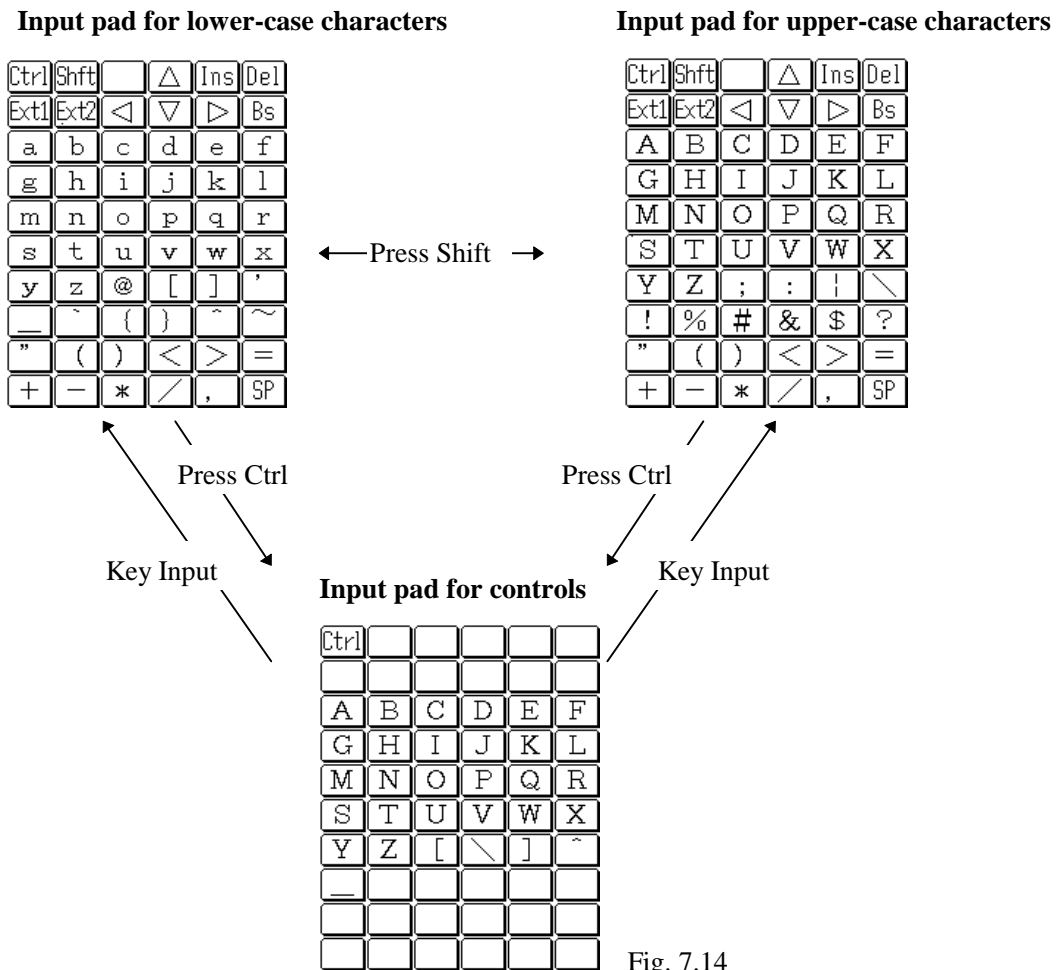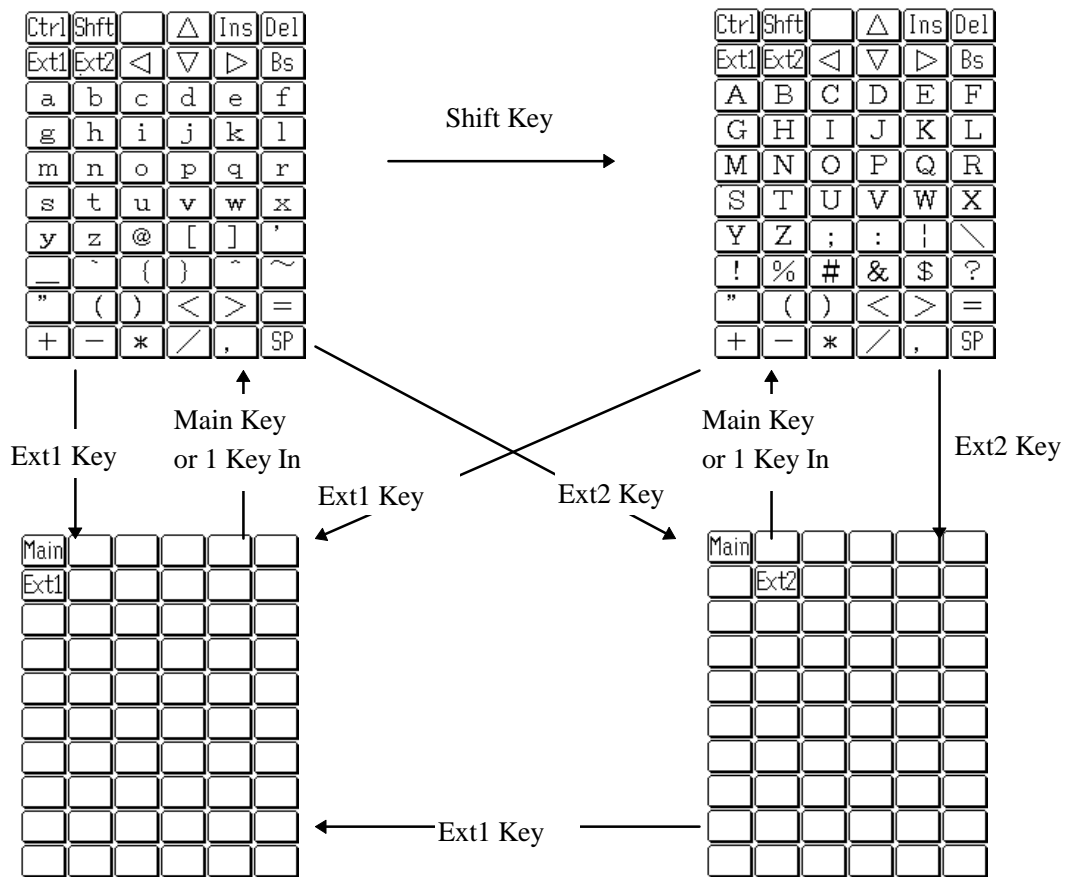
Fig. 7.14

Fig. 7.15

Keycode

ASCII keycodes can be acquired with the input function of this library. Scanning codes cannot be acquired.

Acceptance timing of keycode

A key input is performed by touching the appropriate area on the keypad. However, this input will only be accepted as the keycode when the key scanning and key input functions are called. Specifically, the coordinates of the touched position are placed in the buffer by the interrupt. They are retrieved from the buffer and processed when the key scan and key input functions are called. Therefore, no keycode can be accepted, even if a key is touched, unless the application calls the key scanning or key input function. This means that the application should be programmed so that the key scanning or key input function is called at intervals that are as short as possible.

Acquisition of coordinates

The coordinates of a position touched can be acquired. However, while the keypad is being displayed, it is only possible to acquire a set of coordinates if an area outside the keypad is touched. In this case the coordinates will be recorded together with the following coordinate status.

Coordinate status

When a set of coordinate is acquired, the steps performed, from when the pen (or user's finger) touches the panel to when it is removed from the panel, can be tracked. The coordinate data acquired this way also includes the status data, which is used to determine the pen-down position, pen-up position, etc.

A problem may occur with the acquisition of the pen-up coordinates if the power is turned OFF or if the application program clears the buffer while the pen is on the panel. In this case a status code indicating that the pen-up coordinates cannot be acquired will be returned.

The following are the status values that can be obtained at the acquisition of coordinates. They are defined in keypad.h.

KDC_DOWN	If the touch panel is touched.
KDC_RUN	If the pen is moved while it touches the panel (this status will be returned at 2.0 ms intervals if the pen remains still).
KDC_UP	If the pen is removed from the panel.
KDC_CANCEL	If the pen-down status can be returned but pen-down cannot be returned.

The following table contains the list of status values that the application may receive next time after one piece of status information has been received.

		This time			
		Down	Run	Up	Cancel
Previous time	Down	X	O	O	O
	Run	X	O	O	O
	Up	O	X	X	X
	Cancel	O	X	X	X

O : Possibility to receive.   X : No possibility to receive.

Resume-On process

The entire key buffer will be cleared if a resume-on boot is executed by turning the power off and then on. However, if the power is turned off immediately after the key scan function sends a "data present" reply, the data returned by the key scanning function will be returned if key input is performed after the power has been turned on.

Input acceptance mode

There are two modes for accepting key inputs, as follows:

● DOWN acceptance mode

In this mode the area touched in the pen-down action will change to reverse video and the acquired keycode will be stored in the buffer. This reverse-video area is restored to normal when the pen is lifted. However, it will not be canceled, even if the pen goes outside the key area , while the pen continues moving.

● UP acceptance mode

In this mode, the area touched by the pen-down action will change to reverse video. If the pen goes outside the area which was touched by the pen at the start of its run, the reverse-video condition will be canceled and a new area is displayed in reverse video. Key areas displayed in reverse video can be accepted if the pen is up. If an area other than the keypad has been touched before the pen is lifted, the keycode at that point in time will not be accepted.

The current key acceptance mode will continue until one pen-up action occurs, even if the acceptance mode is switched by the application while the key is touching the keypad.

Toggle function

If the CTRL, EXT1, or EXT2 key in the main keypad (in the upper-case alphabet and lower-case alphabet pads) is touched, either of these keypads becomes active. This input mode returns to normal after one key input has been made. However, if the position touched has no keycode, this input mode cannot be exited.

Repeat function

This function is supported only in the DOWN acceptance mode.

Any keys except the pad control keys, such as the Shift and Main keys, can be used. The CTRL, Ext1, and Ext2 keys are also excluded from use, since the Main keypad can be restored to the active state after one key input has been made.

Key click sound

The key click sound will be turned on when the keypad is touched (when the pen is down), irrespective of the current input acceptance mode. However, this will not happen if a position that does not correspond to any keycode is touched. It will not happen, either, if the pen runs from an invalid key, which was touched first, to a valid key in the UP acceptance mode.

Clearing the buffer

If the key buffer clear command is issued while the keypad is being touched, processing for the current operation will also be canceled. For example, a reverse-video display will be restored to normal, and no new key inputs will be permitted until the pen (or user's finger) is removed from the panel.

Input of ten key while the screen is being touched

If a ten key is pressed while the user screen is being touched, the character code corresponding to the ten key will be logged in the interval from when the down coordinates are acquired to when the up coordinates are acquired.

Registering an expanded key

The user can register a keycode and its key face image for the EXT1 and EXT2 keys. Since the registered contents can be stored in a specific area in the driver, the application program does not have to store them. Expanded keys keycodes 01h to FFh can be registered. Scanning codes can also be registered, however, they cannot be acquired through the input function.

Restrictions and notes on use of keycodes

● Do not use the dedicated input functions and standard input functions concurrently.
  For the standard input functions no interface is provided to log coordinate data, since they are called via the DOS system. If the standard input function is called from DOS, the coordinate data obtained will be redundant. Therefore, if the standard input function is called after the dedicated input function acquires the down coordinates, acquisition of the up coordinates will be hampered. If a key scanning reports "data present" as the result of using the dedicated input function, and if the data obtained through the standard input function is coordinate data, it is assumed that there is no corresponding key.
  If this is the case, the program cannot return from the standard input function. This means that an application program must not contain both the input functions. It is even possible to create an application using only the standard input functions if the application does not use any coordinate data. However, use of the dedicated input functions is recommended.

● There may be a case where the up status cannot be acquired and the cancel status data is returned. The up coordinates may not be returned if the power is turned off after the down status has been acquired. In this case the cancel status will be returned. Take this possibility into account when creating applications. It is especially important to exercise care when creating a program which accepts the result of an action when the pen is up.

- **About the buffer clear**

  Consider the appropriate timing to clear the buffer. If the buffer clear command is issued after the down coordinates are acquired, acquisition of the up coordinates is hampered. Or, if the buffer clear command is issued after the "data present" report is received as a result of executing the key scanning function without performing data input, the keycode for which the key scanning function returned the "data present" reply will not be cleared. Also note that on a scanning code, such as a delete code or insert code, that has a first byte of 00h, the second code cannot be cleared, if the first code has already been acquired.

- **About the acceptance timing of keycodes**

  With the keypad of this system the keycode acceptance and video-reversing processes for the touched key are triggered when key scanning and key input are started. Therefore, if a program has an extremely long interval between key inputs, a time lag from the actual operation may occur, resulting in the key area being reversed after the pen is removed. Also, if the touch panel is touched after one ten key has been pressed, the keycode of the touch panel key will be returned first in the subsequent key input process. This occurs because the keypad driver first processes touch panel input and then processes ten key input. The above facts require the program to be created in such a way that the interval between key inputs is not extremely long.

- Input of a ten key while the touch panel is being touched

  If a ten key is pressed while the touch panel is being touched, the corresponding key code will be logged during the interval between when the down coordinates are acquired and when the up coordinates are acquired.

- About the up coordinates

  The coordinates to be returned in the up status are the same as those that were last returned in the run status.

List of Input Functions

Page	Function	Description
160	KEY_CheckExist	Checks if the keypad driver is installed
161	KEY_Read	Acquires a key code or coordinates
162	KEY_Scan	Checks if a key code or coordinates are in the buffer.
163	KEY_Clear	Clears the key code and coordinate buffers.
163	KEY_DisplayPad	Sets the specified pad to Display or Non-display.
164	KEY_GetPadState	Returns whether a keypad is being displayed and the pad number, if displayed.
164	KEY_SetInputMode	Specifies the timing of input acceptance: at pen-up or pen-down.
165	KEY_GetInputMode	Sets whether to read confirmation mode.
165	KEY_SetCoordinateMode	Sets whether to accept coordinate input.
166	KEY_GetCoordinateMode	Acquires the current coordinates input mode.
166	KEY_SetExtKey	Registers the expansion keys.
169	KEY_DelExtKey	Deletes the specified expansion keys registered.

Checking Condition of Installed Keypad Driver

Checks if the keypad driver has been installed or not.

SYNTAX

    int KEY_CheckExist(void);

INPUT

    None

OUTPUT

    = 1   Driver resides .

    = 0   Driver has not been installed.

EXAMPLE

```
#include "padlib.h"
int retcd;
retcd = KEY_CheckExist();
```

Readout from Key Buffer

Acquires a key code or coordinates which have been acquired from the key buffer. If the buffer contains neither a key code nor coordinates, **NO_DATA** will be returned. However, coordinate values can be returned only if the coordinates input mode is set to "enable".

SYNTAX

```
int KEY_Read (unsigned int *Keycode, KDC_POSINFO *Position);
```

INPUT

Keycode = Pointer to the area in which the read key code is stored

Position = Pointer to the **KDC_POSINFO** structure in which the read coordinates are stored

```
typedef struct{
int status;  /*status
                *0     : DOWN_STATUS
                *1     : RUN_STATUS
                *2     : UP_STATUS
                *3     : CANSEL_STATUS
                  This status arises when the power is off after the down
                  status is acquired and before the up status is acquired.
               */
    int xpos; /* X coordinate */
    int ypos; /* Y coordinate */
 }KDC_POSINFO;
```

OUTPUT

= KDC_NONE (0)	No data
= KDC_KEYCODE (1)	A key code is present
= KDC_POSDATA (2)	Coordinates are present

EXAMPLE

```
#include "padlib.h"
unsigned int Keycode;
POSITION  Position;
int retcode;
```

 retcode =   KEY_Read (&Keycode,&Position);

Refer to "KEY_Scan" also.

Checking Key Buffer Content

Checks if a key code or coordinates are in the buffer. If either is present, the key code or coordinates will be returned. The buffer remains uncleared. Use the **KEY_Read** function to load the data. However, the coordinates will not be returned if the coordinates input mode is set to "disable".

SYNTAX

```
int KEY_Scan(unsigned int *Keycode, POSITION Position);
```

INPUT

Keycode = Pointer to the area in which the read key code is stored

Position = Pointer to the **KDC_POSINFO** structure in which the read coordinates are stored

```
typedef struct{

int status;  /*status
              *0  : DOWN_STATUS
              *1  : RUN_STATUS
              *2  : UP_STATUS
              *3  : CANSEL_STATUS
      This status arises when the power is off after the down status is
      acquired and before the up status is acquired.
              */
int xpos;    /* X coordinate */
int ypos;    /* Y coordinate */
}KDC_POSINFO;
```

OUTPUT

= KDC_NONE (0)	No data
= KDC_KEYCODE (1)	A key code is present.
= KDC_POSDATA (2)	Coordinates are present.

EXAMPLE

```
#include "padlib.h"


unsigned int Keycode;
POSITION  Position;
int retcode;


retcode =  KEY_Scan(&Keycode,&Position);
```

Refer to "KEY_Read" also.

Key Buffer Clear

Clears both the system buffer and key BIOS buffer. If coordinate input by pen touch is continued, coordinate values selected until the pen is lifted will be ignored. Furthermore, any operation in progress on the pad will be canceled at that point. While in the process of USER coordinates acquisition, the coordinates will be cleared after they have been acquired. Also, the **KEY_Scan** and **KEY_Read** function does not collect data until the keypad is touched again.

SYNTAX

```
void KEY_Clear(void);
```

INPUT

None

OUTPUT

None

EXAMPLE

```
#include "padlib.h"
KEY_Clear0;
```

Display / Non-display of Keypad

Sets the specified keypad to either Display or Non-display. If "Display" is specified while the keypad is being displayed, nothing occurs.

SYNTAX

```
int KEY_DisplayPad (int padno);
```

INPUT

padno = Number of the displayed keypad

KDC_PADOFF(0) Non-display

KDC_PADENG(1) Display

OUTPUT

= 0   Normal termination

= -1   Invalid pad number

EXAMPLE

```
#include "padlib.h"
int retcode;
retcode = KEY_DisplayPad (PAD_xxx);
```

Note:

If "Non-display" mode is specified while the keypad is not being displayed, the operation will be terminated normally.

Acquisition of Number of Keypad Being Displayed

Acquires the number of the currently displayed keypad.

SYNTAX

int KEY_GetPadState(void);

INPUT

None

OUTPUT

KDC_PADOFF (0)   Non-display

KDC_PADENG (1)   Being displayed

EXAMPLE

```
#include "padlib.h"
int retcode;
retcode = KEY_GetPadState();
```

Refer to "KEY_DisplayPad" also.

Setting Input Acceptance Mode

Used to specify the acceptance mode for key input.

SYNTAX

int KEY_SetInputMode(int mode);

INPUT

mode = Number of the displayed keypad

KDC_MODEDOWN (0)   DOWN-acceptance (default value)

KDC_MODEUP (1)   UP-acceptance

OUTPUT

= 0   Normal termination

= -1   Acceptance mode specification error

EXAMPLE

```
#include "padlib.h"
int retcode;
retcode = KEY_SetInputMode(MODE_xxx);
```

Acquisition of Input Acceptance Mode

Acquires the acceptance mode currently set.

SYNTAX

int KEY_GetInputMode(void);

INPUT

None

OUTPUT

KDC_MODEDOWN(0)   DOWN-acceptance (default setting)

KDC_MODEUP(1)   UP-acceptance

EXAMPLE

```
#include "padlib.h"
int retcode;
retcode =  KEY_GetInputMode();
```

Refer to "KEY_SetInputMode" aslo.

Setting Coordinates Input Mode

Sets whether coordinate input is accepted.

SYNTAX

int KEY_SetCoordinateMode(int mode);

INPUT

mode =   Number of the displayed keypad

0   Does not input coordinates (default setting).

1   Inputs coordinates.

OUTPUT

= 0   Normal termination

= -1   Input parameter error

EXAMPLE

```
#include "padlib.h"
int retcode;
retcode = KEY_SetCoordinateMode(0);
KEY_GetCoordinateMode
```

Acquisition of Coordinates Input Mode

Acquires the coordinates input mode currently set.

    SYNTAX

```
int KEY_GetCoordinateMode(void);
```

    INPUT

        None

    OUTPUT

        0   Does not input coordinates.

        1   Inputs coordinates.

    EXAMPLE

```
#include "padlib.h"
int retcode;
retcode = KEY_GetCoordinateMode();
```

Refer to "KEY_SetCoordinateMode" also.

Registering Enhanced Keypad

This function is used to assign various eanhanced keys. By specifying the desired display images and key codes to be used as the enhanced keypad the user can create a custom keypad.

    SYNTAX

```
int KEY_SetExtKey(int keyno, KEYLIST *keylist);
```

    INPUT

        keyno =   Enhanced key number

                KDC_EXTKEY1(1)   Enhanced key 1

                KDC_EXTKEY2(2)   Enhanced key 2

        keylist = list of replaceable keys

```
                typedef struct{
                    int data_cnt;  /*Number of replaced keys*/
                    KDC_EXTKEYINFO far *keyinfo;
                            /*Table of replaceable key information */
                }KDC_EXTKEYLIST;
                typedef struct{
                    unsigned int keycode; /* Key code to be set */
                    int change_position;  /* Place of replacement
                                                    (1 to 48)  */
                    char far *image_adr; /* Button display data(32 x 24
                                                    bits) */
                }KDC_EXTKEYINFO;
```

OUTPUT

= 0   Normal termination

= -1   Input parameter error

Places of replacement

Main					
	Ext2				
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

Fig. 7.16

Button Display Image (32 x 24 bits)

Button display image is displayed by bit map data as shown below.

```
char image[] = { 0x1C, 0x03, ...};
```


Fig. 7.17

EXAMPLE

```
#include "padlib.h"

KEYINFO keydata[10];
KEYLIST keylist;
int retcode, i;

for (i = 0; i < countof(keydata); i++) {
    keydata[i].keycode = xxxx;
    keydata[i].change_position = xxxx;
    keydata[i].image_adr = xxxx;
}
keylist.keyinfo = keydata;
keylist.data_cnt = countof(keydata);
retcode = KEY_SetExtKey(KDC_EXTKEY1, &keylist);
```

Refer to "KEY_DelExtKey" also.

Note:

If the key list includes any invalid data, no error results. Only the data pieces that can be registered will be registered

Deleting Enhanced Keys

Deletes the data of the specified enhanced key. Only the frame will be remained.

SYNTAX

```
int KEY_DelExtKey(int keyno, KEYLIST *keylist);
```

INPUT

keyno = Number of the expansion key

KDC_EXTKEY1(1)   Enhanced key 1

KDC_EXTKEY2(2)   Enhanced key 2

keylist =   list of replaceable keys

```
typedef struct{
   int data_cnt;   /*Number of replaced keys*/
   KDC_EXTKEYINFO far *keyinfo;
      /*Table of replaceable key information */
}KDC_EXTKEYLIST;
typedef struct{
   unsigned int keycode;  /*Key code to be set*/
   int change_position;   /*Place of replacement (1 to 48) */
   char far *image_adr;  /*Button display data (32 x 24 bit)*/
}KDC_EXTKEYINFO;
```

OUTPUT

= 0   Normal termination

= -1   Input parameter error
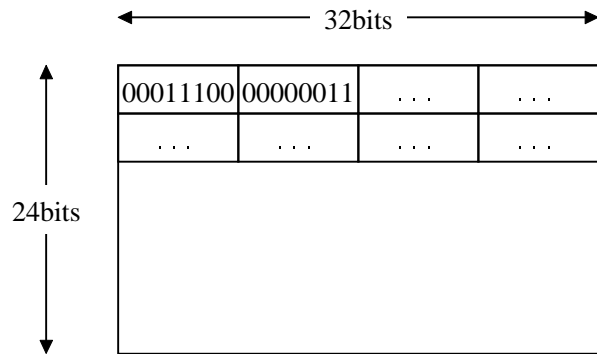
EXAMPLE

```
#include "padlib.h"

KEYLIST keylist:
int retcode, i;

for (i = 0; i < countof(keydata):i++){
   keydata[i].change_position = xxxx;
}
 keylist.data_cnt = countof(keydata);
 retcode = KEY_DelExtKey(KDC_EXTKEY1, &keylist);
```

Refer to " KEY_SetExtKey " also.

Note:

Deletion of a key which is assigned to a position that has not been registered will not result in error.

7.6.4 OBR Library

Overview

The OBR library is used to control the OBR (Barcode Reader) from application programs developed by the user with the C language. It supports the following two OBR types:

DT-9650BCR : Pen-type barcode reader
DT-9656BCR : CCD barcode reader

Note about the Libraries

This library is supplied in one of the three models, small or medium or large-memory model, as suitable for each memory model. Any application program that uses this library should include **obrlib.h** in the corresponding source file. Constants that are passed to the library functions and their prototypes are defined in the following header files.

ORBLIB.H	Header file for the OBR library
SLIBOBRD.LIB	OBR library for small-memory model
MLIBOBRD.LIB	OBR library for medium-memory model
LLIBOBRD.LIB	OBR library for large-memory model

This library supports the following six types of functions. Each function has been macro-defined in **obrlib.h**, however, it can be replaced, as required, with the corresponding function call by specifying either of the compile options: /DDT9650 or /DDT9656.

No.	Function	Description
1	OBR_Open	Initialization of COM port and power on
2	OBR_Close	Release of COM port and power off
3	OBR_Send	Transmission of command to OBR
4	OBR_Stat	Acknowledgment of received data
5	OBR_Read	Read of the received data
6	OBR_Clear	Invalidation of codes in reception buffer

This library must refer to the system library to turn on and off the power supply of the COM port. This means that the system library must also be linked to this library whenever this library is used.

Reception Buffer

This library uses two reception buffers, as shown below, so that during the processing (read) of one of the received barcodes the next barcode can be successfully received.
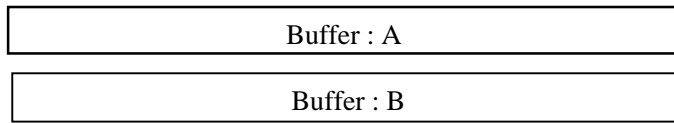
Buffer : A
Buffer : B

Fig. 7.18

The following explains the operation sequence by which codes are put into the reception buffer.

● When the first barcode is received, it will be temporarily stored in Buffer A.

● When the second barcode is received, it will be temporarily stored in Buffer B.

● When the third next barcode is received, it will be temporarily stored in Buffer A.

● When the fourth barcode is received, it will be temporarily stored in Buffer B.

With this library the received barcodes are distributed alternatively to the two buffers as described above. If one of the received barcodes is not read, it will be overwritten by a new barcode. This necessitates any received data to be acknowledged with the **OBR_Stat** function, then read using the **OBR_Read** function after acknowledgment.

Note :

● If programming with this library, first make the **OBR_Open** function call. The **OBR_Open** function will turn on the power supply to the COM port and initialize it. It enables the operation of other functions (**OBR_Send**, **OBR_Read**, etc.) and maintains the power supply to the COM port. Therefore, always call the **OBR_Close** function so that the COM port is turned off and freed before completing the use of the OBR (i.e. application).

● From the above list, only the **OBR_Send** function operates (takes different arguments or return values) specifically according to the type of barcode reader. This is because the **OBR_Send** function is used to set up a selected OBR. Possible setup contents vary according to type of barcode reader. The descriptions in this manual are organized according to type of barcode reader, and the functions other than **OBR_Send** operate without discriminating the type of barcode reader.

● The DT-9650BCR and DT-9656BCR have an EEPROM in which the setup contents can be written and stored. This eliminates the need to perform setup each time the power is turned on.

DT-9650BCR

Reception Buffer

Use the buffer provided in the OBR library to receive OBR codes.

Reception Data Format

The reception data format is defined as follows:
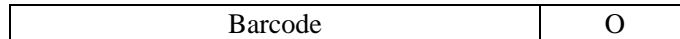
Barcode	O

Fig. 7.19

Read-out Symbols

The following barcode types can be read:

1) WPC
2) WPC add on
3) Industrial 2 of 5
4) Interleaved 2 of 5 (ITF)
5) CODE39
6) NW-7 (CODABAR)
7) CODE11
8) CODE93
9) CODE128

Communication with OBR

Asynchronous

・ 9600 bps
・ Data bits: 8 bits
・ Parity: None
・ Stop bit: 1 bit

Header code

・ None

Termination code

・ CR

List of Available Functions

Page	Function	Decsription
173	OBR_Open	Initialization of COM port and power on
173	OBR_Close	Release of COM port and power off
174	OBR_Send	Transmission of command to OBR
174	OBR_Stat	Acknowledgment of received data
175	OBR_Read	Read of the received data
175	OBR_Clear	Invalidation of codes in reception buffer

Initialization of OBR

Initializes the COM port to establish a connection with the OBR, and turns on the power to the COM port.

SYNTAX

```
#include "obrlib.h"
void OBR_Open();
```

INPUT

None

OUTPUT

None

Note :

If programming with this OBR library, first make this **OBR_Open** function call to initialize the COM port.

Release of COM Port

Releases the COM port and turns off the power to the COM port.

SYNTAX

```
#include "obrlib.h"
void OBR_Close();
```

INPUT

None

OUTPUT

None

Note :

Call this function if completing the use of the OBR (i.e. terminating the application program).

Transmission of Command

Transmits a command represented by a single ASCII code to the OBR. Various options including "Readout mode", "Data transfer format", etc., can be set for this transmission. This setup does not have to be made each time the power is turned on if it is written in the EEPROM.

For information about the setup procedure refer to "Setting Operation Mode".

SYNTAX

```
#include "obrlib.h"
int OBR_Send (char Cmd);
```

INPUT

Cmd = Transmission command   (refer to the Command List.)

OUTPUT

= 0   : Normal termination

= 1   : Transmission buffer-full error

= 2   : CTS does not turn to ON.

= 3   : Receives NAK from scanner.

= 4   : No response from scanner

Note :

Do not read another barcode during command transmission to the OBR.

Acknowledgment of Received Data

Returns the number of characters in the first barcode stored in the reception buffer.

SYNTAX

```
#include "obrlib.h"
int OBR_Stat();
```

INPUT

None

OUTPUT

The absolute value shows the number of characters in the received barcode (not including a CR). The sign indicates whether the data is a complete barcode or not.

< 0   Incomplete barcode

> 0   Complete barcode

Note :

After acknowledging that the barcode has been received, read it using the **OBR_Read** function.

Readout of Received Data

Acquires the first barcode in the reception buffer and writes it to the specified buffer. The reception data SYNTAX is as follows:
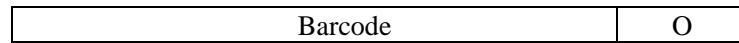
Barcode	O

Fig. 7.20

SYNTAX

```
#include "obrlib.h"
int OBR_Read (unsigned char *pBuf);
```

INPUT

pBuf = Pointer to the buffer that stores the received barcode

OUTPUT

The absolute value shows the number of characters in the received barcode.

The sign indicates the validity of the barcode.

> 0　The read barcode has no data error.

$= 0$　Either the reception acknowledgment is not performed (OBR_Stat function is not used) or there is no received data.

< 0　The read barcode has a data error.

Note :

Before reading a barcode using this function, acknowledge reception with the **OBR_Stat** function. Note that received barcode data will be cleared from the reception buffer after it has been read by the **OBR_Read** function. This means that the following barcode can be read immediately after the preceding one, even if there is an error, has been read.

Invalidating Code in Buffer

Invalidates a barcode in the reception buffer.

SYNTAX

```
#include "obrlib.h"
void OBR_Clear();
```

INPUT

None

OUTPUT

None

Setting Operation Mode / DT-9650BCR

Overview

On this OBR various settings, as listed below, can be made through command transmission.
(For a list of the actual commands refer to the Command List on page 178.)

1. Specifying the number of read digits
2. Specifying the CODE39/NW-7 ICG code
3. Readability of code
4. Data transfer SYNTAX
5. Specifying the buzzer activation and LED ON modes
6. Specifying the output of BEL if decoding is not possible
7. Specifying the scanning mode
8. Specifying the sleep mode/stop mode
9. Write in the EEPROM

Transmission of Command

There are two types of commands: normal commands and expanded commands. They must
be transmitted according to the following procedure.

● **Transmission of normal commands**

In order to transmit a command other than the expanded commands included in the Command
List use the corresponding command symbol without modification.

Example: To set all codes to "Permit read" with the "Readability of code"
    OBR_Send ('X');

● **Transmission of expanded commands**

To transmit an expanded command included in the Command List follow the procedure below.

1. First transmit the "Transmission start" command from the expanded commands.
2. Transmit the objective expanded command.
3. After the objective expanded command has been transmitted, transmit the "Transmission
   complete" command.

Example: To set the CODE39 C/D to "Prevent check (without changing the transfer
    function)" with the corresponding expanded commands
    OBR_Send ('u');
    OBR_Send ('A');
    OBR_Send ('v');

Power-save Mode Control Command

Used to control the power-save mode of the OBR. See the following diagram.

Example
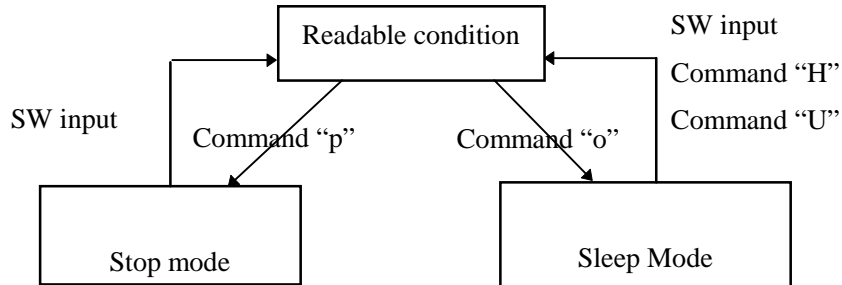
Fig. 7.19

Writing Set Values to EEPROM

The OBR is provided with a function to write the current setting values to EEPROM.

To do this, transmit the 'y' command. If this is not done, other commands that have been transmitted previously to the 'y' command will not be written to EEPROM. As a result, they will be erased when the power is turned off and the settings specified by these commands will not be valid the next time the power is turned on. However, the following commands can not be used to write a setting value to EEPROM.

One period of buzzer activation/LED ON	Command : L
Enable scanning	Command : H
Disable scanning	Command : I
Special mode (disable scanning after one normal reading)	Command : U
Request sleep mode	Command : o
Request stop mode	Command : p
Expanded command control: Transmission start	Command : u
Expanded command control: Transmission complete	Command : v

Command List   (Italic and bold letters indicate default value)

No. of digits	Command	No. of digits	Command	No. of digits	Command
1 to 42	^P	16	(space)	32	0
1	^Q	17	!	33	1
2	^R	18	"	34	2
3	^S	19	#	35	3
4	^T	20	$	36	4
5	^U	21	%	37	5
6	^V	22	&	38	6
7	^W	23	'	39	7
8	^X	24	(	40	8
9	^Y	25	)	41	9
10	^Z	26	*	42	:
11	^[	27	+		
12	^\	28	,(comma)		
13	^]	29	-		
14	^^	30	.(period)		
15	^_	31	/		

1 Specifying the number of read digits (caption above the first table)

Item		Command	Default
2. Specify CODE39/ NW-7 ICG	Less than one ICG character	=	Yes
	Less than eight ICG characters	?	--
3. Readability of code	All codes Enable read	X	--
	Disable read	x	--
	CODE39 Enable read	A	Yes
	Disable read	a	--
	NW-7 Enable read	B	Yes
	Disable read	b	--
	WPC Enable read	C	Yes
	Disable read	c	--
	2 of 5 (Industrial/Standard)		
	Enable read	D	Yes
	Disable read	d	--
	ITF Enable read	E	Yes
	Disable read	e	--
	CODE11 Enable read	F	--
	Disable read	f	Yes
	CODE93 Enable read	G	--
	Disable read	g	Yes
	CODE128 Enable read	W	--
	Disable read	w	Yes
	WPC add on Disable read	l	Yes
	Enable read	m	--
	Forced read	n	--
4. Data transfer format	CODE39		
	Enable full-ASCII conversion	h	--
	Disable full-ASCII conversion	i	Yes
	Transfer start/stop codes	Z	--
	Not transfer start/stop codes	z	Yes

	NW-7 start/stop code		
	Transfer	[	Yes
	Not transfer	{	--
	Change codes to uppercase characters	q	Yes
	Change codes to lowercase characters	r	--
	Enable transfer of ABC code	j	--
	Disable transfer of ABC code	k	Yes
	C/D (CODE39/NW-7/2of5/CODE11)		
	Disable check	R	Yes
	Enable transfer of check	S	--
	Disable transfer of check	T	--
	Readout CODE ID Not transfer	P	Yes
	Transfer	Q	--
5. Specify buzzer activation and LED ON modes	· Enable buzzer/LED ON after normal read	J	Yes
	· Disable buzzer/LED ON after normal read	K	--
	· Enable buzzer/LED ON for one time	L	--
	· LED OFF when command awakes from sleep mode	s	Yes
	· LED ON when command awakes from sleep mode	t	--
6. Specify output of BEL when the code can not be decoded	· Output enable	M	--
	· Output disable	N	Yes
7. Specify scanning mode	· Scanning enable	H	Yes
	· Scanning disable	I	--
	· Special mode Scanning disable after one normal read	U	--
8. Specify sleep mode/stop mode	· Request sleep mode	o	--
	· Request stop mode	p	--
9. Write to EEPROM	· Write defaults	Y	--
	· Write current setting values	y	--
10. Modify settings	· Switch to the setting values currently stored in EEPROM	O	--
11. Expanded commands	· Expanded command control		
	Transmission start	u	--
	Transmission complete	v	--
	· CODE39 C/D		
	Disable check (without changing the transfer function)	A	--
	Enable check/Transfer	B	--
	Enable check/Not transfer	C	--
	Disable check/Not transfer	Y	--
	Disable check/Transfer	Z	Yes

· NW-7 C/D Disable check (without changing the transfer function)	D	--
Enable check/Transfer	E	--
Enable check/Not transfer	F	--
Disable check/Not transfer	[	--
Disable check/Transfer	\	Yes
· 2 of 5 C/D Disable check (without changing the transfer function)	G	--
Enable check/Transfer	H	--
Enable check/Not transfer	I	--
Disable check/Not transfer	]	--
Disable check/Transfer	^	Yes
· CODE11 C/D Enable transfer of check (1)	J	--
Disable transfer of check (1)	K	Yes
Enable transfer of check (2)	L	--
Disable transfer of check (2)	M	--
· CODE93 C/D Enable transfer of no check	N	-
Disable transfer of no check	O	--
Disable transfer of check	P	Yes
Enable transfer of check	Q	--
· CODE128 C/D No check (without changing the transfer function)	S	--
Disable transfer of check	T	Yes
Disable transfer of no check	U	--
Enable transfer of no check	V	--
Disable transfer of check	W	--
Enable transfer of check	X	--

DT-9656BCR

Reception Buffer

Use the buffer provided in the OBR library to receive OBR codes.

Reception Data Format

The reception data format is defined as follows:
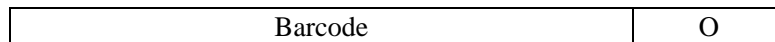
Barcode	O

Fig. 7.22

Read-out Symbols

The following barcode types can be read:

1) UPC/EAN (JAN)

2) UPC/EAN with supplemental

3) DTF

4) ITF

5) CODE39

6) NW-7 (CODABAR)

7) CODE93

8) CODE128

9) MSI/Plessey

Communication with OBR

Asynchronous

・ 9600 bps

・ Data bits: 8 bits

・ Parity: None

・ Stop bit: 1 bit

Header code

・ None

Termination code

・ CR

List of Functions

Page	Function	Description
182	OBR_Open	Initialization of COM port and power on
182	OBR_Close	Release of COM port and power off
183	OBR_Send	Transmission of command to OBR
183	OBR_Stat	Acknowledgment of received data
184	OBR_Read	Read of the received data
184	OBR_Clear	Invalidation of codes in reception buffer

Initialization of OBR

Initializes the COM port to establish the connection with the OBR, and turns on the power to the COM port.

SYNTAX

```
#include "obrlib.h"
void OBR_Open();
```

INPUT

None

OUTPUT

None

Note:

If programming with this OBR library, first place this **OBR_Open** function call to initialize the COM port.

Release of COM Port

Releases the COM port and turns off the power to the COM port.

SYNTAX

```
#include "obrlib.h"
void OBR_Close();
```

INPUT

None

OUTPUT

None

Note:

Call this function whenever completing the use of the OBR (i.e. terminating the application program).

Transmission of Command

Transmits a command to the OBR. Only one command should be specified in each command transmission session. Various options including "Readout mode", "Data transfer format", etc., can be specified for this transmission. These setup specifications, if written to EEPROM, do not have to be set each time the power is on.

For information about the setup procedure refer to "Setting Operation Mode".

SYNTAX

```
#include "obrlib.h"
int OBR_Send (char *pszCmd);
```

INPUT

pszCmd =   Pointer to the transmission command character string
(Refer to the Command List).

OUTPUT

= 0   : Normal termination

= 1   : Transmission buffer-full error

Note:

Do not read another barcode during command transmission to the OBR.

Acknowledgment of Received Data

Returns the number of characters in the first barcode stored in the reception buffer.

SYNTAX

```
#include "obrlib.h"
int OBR_Stat();
```

INPUT

None

OUTPUT

The absolute value indicates the number of characters in the barcode received (not including a CR), and the sign indicates if the data represents a complete barcode.

< 0   An incomplete barcode

> 0   A complete barcode

Note:

After acknowledging the received barcode with this function, read it using the **OBR_Read** function.

Readout of Received Data

Acquires the first barcode in the reception buffer and writes it to the specified buffer. The reception data SYNTAX is as follows:
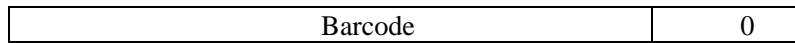
Barcode	0

Fig. 7.23

SYNTAX

```
#include "obrlib.h"
int OBR_Read (unsigned char *pBuf);
```

INPUT

pBuf = Pointer to the buffer that stores the received barcode

OUTPUT

The absolute value indicates the number of characters in the barcode received, and the sign indicates whether the barcode has a data error.

> 0   The read barcode does not have a data error.

= 0   Either the reception acknowledgment has not been made (**OBR_Stat** function is not used) or there is no received data.

< 0   The read barcode has a data error.

Note:

Before reading a barcode using this function, acknowledge reception with the **OBR_Stat** function. Note that the error status, reception data, and the number of received data pieces will be cleared from the reception buffer after a barcode is read by the **OBR_Read** function.

This means that the following barcode can be read immediately after the preceding one, which may include an error, has been read.

Invalidating Code in Buffer

Invalidates a barcode in the reception buffer.

SYNTAX

```
#include "obrlib.h"
void OBR_Clear();
```

INPUT

None

OUTPUT

None

Setting Operation Mode / DT-9656BCR

Overview

On the OBR various settings, as listed below, can be made through command transmission.
(For a list of actual commands refer to the Command List.)

1. Readability of code
2. Adding a readable code
3. Data transfer SYNTAX
4. Condition for the least significant digits
5. Specifying the buzzer activation mode
6. Specifying the LED ON mode
7. Read mode
8. Read time
9. Mark/base of barcode
10. Redundant read
11. Use of Length CODE
12. Specifying write to EEPROM

Transmission of Command

Commands must be transmitted using the **OBR-Send** function.

Example: To specify "Read all codes"

```
OBR_Send ("A0");
```

Writing Set Values to EEPROM

The OBR is provided with a function to write the current setting values to EEPROM. To do this, transmit the 'Z2' command.

If this is not done, other commands that have been transmitted previously to the 'Z2' command will not be written to EEPROM. As a result, they will be lost when the power is turned off and the settings specified by these commands will not be valid the next time the power is turned on.

Example: To specify "Read all codes" and write to EEPROM

```
OBR_Send ("A0");
OBR_Send ("Z2");
```

Command List

	Item		Command	Default
1. Readability of code	・ Read all codes		A0	--
	・ UPC only		J1	--
	・ UPC + 2 digits of supplemental only		J2	--
	・ UPC + 5 digits of supplemental only		J3	--
	・ EAN only		J4	--
	・ EAN + 2 digits of supplemental only		J5	--
	・ EAN + 5 digits of supplemental only		J6	--
	・ DTF only		J7	--
	・ ITF only		J8	--
	・ CODE39 only		A2	--
	・ NW-7 (CODABAR) only		A3	--
	・ CODE93 only		A5	--
	・ CODE128 only		A6	--
	・ MSI/Plessey only		A7	--
2. Adding readable code	・ UPC	Enable read	R1	Yes
	・ UPC + 2 digits of supplemental	Enable read	R2	--
	・ UPC + 5 digits of supplemental	Enable read	R3	--
	・ EAN	Enable read	R4	Yes
	・ EAN + 2 digits of supplemental	Enable read	R5	--
	・ EAN + 5 digits of supplemental	Enable read	R6	--
	・ DTF	Enable read	R7	Yes
	・ ITF	Enable read	R8	Yes
	・ CODE39	Enable read	B2	Yes
	・ NW-7 (CODABAR)	Enable read	B3	Yes
	・ CODE93	Enable read	B5	--
	・ CODE128	Enable read	B6	--
	・ MSI/Plessey	Enable read	B7	--
3. Data transfer format	・ CODE39			
		Not calculate C/D	C0	Yes
		Calculate C/D	C1	--
		Transfer C/D	C2	Yes
		Not transfer C/D	C3	--
		Not transfer start/stop code	D0	--
		Transfer start/stop code	D1	Yes
	・ NW-7 start/stop code			
		Not transfer	F0	--
		Transfer ABCD/TN*E	F1	--
		Transfer abcd/tn*e	F2	--
		Transfer ABCD/ABCD	F3	--
		Transfer abcd/abcd	F4	Yes
	・ ITF/DTF C/D			
		Not calculate C/D	G0	Yes
		Calculate C/D	G1	--
		Transfer C/D	G2	Yes
		Not transfer C/D	G3	--

	・ UPC-A		
	13 digits: Transfer all	E2	Yes
	12 digits: Not transfer "0" header for adjusting the number of digits	E3	--
	12 digits: Not transfer C/D	E4	--
	11 digits: Not transfer C/D and "0" header for adjusting the number of digits	E5	--
	・ UPC-E		
	8 digits: Transfer all	E6	--
	7digits: Not transfer "0" header for adjusting the number of digits	E7	--
	7 digits: Not transfer C/D	E8	Yes
	6 digits: Not transfer C/D and "0" header for adjusting the number of digits	E9	--
	Acquire only system number "0"	E0	Yes
	Acquire both system numbers "0" and "1"	E1	--
4. Specify the least significant digit	・ CODE39, NW-7: 1 digit, ITF: 2 digits		
	Disable read	H2	Yes
	Enable read	H3	--
5. Specify buzzer activation mode	・ Buzzer of successful read		
	Disable buzzer	W0	--
	Frequency 1 KHz	W1	--
	Frequency 2 KHz	W2	--
	Frequency 4 KHz	W3	Yes
	・ Buzzer-ON period		
	50 msec	W7	--
	100 msec	W4	--
	250 msec	W5	Yes
	500 msec	W6	--
	・ Buzzer volume		
	Small	T3	--
	Medium	T2	--
	Large	T1	--
	Maximum	T0	Yes
6. Specify LED ON mode	・ ON at successful reading		
	Disable	T4	--
	Enable	T8	Yes
	Period of ON : 0.25 sec	T5	--
	Period of ON : 0. 5 sec	T6	--
	Period of ON : 0.75 sec	T7	--
	Synchronize LED and buzzer	T9	Yes
7. Read mode	・ One-shot read	S0	--
	・ Multiple reads	S1	Yes
	・ Continuous read	S7	--

8. Read time	Infinite			Y0	Yes
	2 sec			Y1	--
	4 sec			Y2	--
	6 sec			Y3	--
	8 sec			Y4	--
	10 sec			Y5	--
	15 sec			Y6	--
	20 sec			Y7	--
9. Contrast of normal /reverse	Normal contrast			V2	Yes
	Both normal/reverse contrast			V4	--
10. No. of verifications	No verification			X0	--
	Verification twice			X1	Yes
	Verification three times			X2	--
	Verification four times			X3	--
11. Use of Length CODE	・ UPC-A		Not transfer	2A	Yes
			Transfer	3A	--
	・ UPC-A with supplemental		Not transfer	2B	Yes
			Transfer	3B	--
	・ UPC-E		Not transfer	2C	Yes
			Transfer	3C	--
	・ UPC-E with supplemental		Not transfer	2D	Yes
			Transfer	3D	--
	・ EAN-13		Not transfer	2E	Yes
			Transfer	3E	--
	・ EAN-13 with supplemental		Not transfer	2F	Yes
			Transfer	3F	--
	・ EAN-8		Not transfer	2G	Yes
			Transfer	3G	--
	・ EAN-8 with supplemental		Not transfer	2H	Yes
			Transfer	3H	--
	・ CODE39		Not transfer	2I	Yes
			Transfer	3I	--
	・ NW-7		Not transfer	2J	Yes
			Transfer	3J	--
	・ DTF		Not transfer	2K	Yes
			Transfer	3K	--
	・ ITF		Not transfer	2L	Yes
			Transfer	3L	--

	・ CODE93			
		Not transfer	2M	Yes
		Transfer	3M	--
	・ CODE128			
		Not transfer	2N	Yes
		Transfer	3N	--
	・ MSI/Plessey			
		Not transfer	2O	Yes
		Transfer	3O	--
12. Specify write to EEPROM			Z2	--

8. Utility

8.1 Overview

The development kit contains some utility programs to be used as required.

- **Calculator Utility**

  Calculator program including memory calculation implementing the CASIO standard specifications .

- **Clock Utility**

  Used to refer the date and time of the built-in clock and to set the power ON alarm.

- **Calendar Utility**

  Used to refer to a calendar for a period of the years between January 1980 and December 2079.

- **Remaining Battery Voltage Display Utility**

  Displays on a software meter the amount of battery voltage remaining for main and sub-batteries.

- **FLINK Utility**

  Transfers/receive s file through IrDA interface.

- **XY Utility**

  Transfers/receives file through XMODEM or YMODEM.

- **Reverse Video Utility**

  Changes the color of LCD screen. This utility is used to change the entire screen to reverse video. From the nature of the FSTN semi-transparent type LCD unit of this terminal the density of colors (tones) will be reversed.

- **COM2KEY Utility**

  Using COM cable and PC, it is possible to input through keyboard on the DOS prompt. In other words, a PC keyboard can be used to input characters and numerals to IT-2000 through the DOS prompt.

8.2   Calculator Utility

Overview

Use this calculator utility for decimal calculations. The result of calculation can be acquired from the application program. This utility is provided as an EXE file and should be activated as command line or as a child process of the application program.

File Name

```
CALC.EXE
```
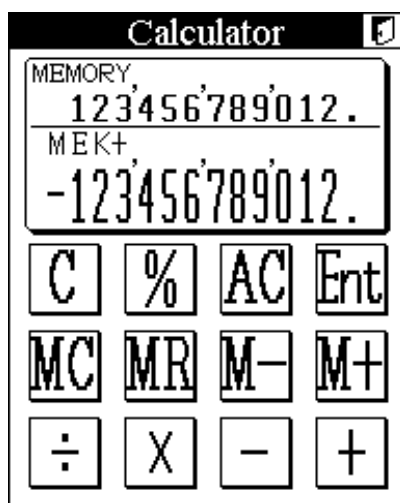

Fig. 8.1

Function

The calculator utility provides the following functions:

- Calculation range: ± 0.00000000001 to ± 999999999999 and 0 (12 digits)
- Apostrophes after the thousandth digit.
- Arithmetical calculation $(+, -, \times, \div)$
- Arithmetical constant calculation $(++, --, \quad \times\times, \div\div)$
- Percentage calculation (%)
- Calculation with memory functions (MC, MR, M+, M–)
- Display of a memorized value
- Value entry function

Condition of Operation

This utility requires two pages (32 Kbytes) of EMS memory and the driver, hardware window manager (**HWWMAN.EXE**) and keypad (**KEYPAD.EXE**) which must be resided always. Refer to Chapter 6.4 "Keypad Driver/Hardware Window Manager" for the detail.

Startup Method

This utility is not stored in the basic drive (C:). It must be copied to RAM disk (A:) or FROM drive (D:) for the utility to be started up. It can be used individually or called as child-process.

Basic Function

Operation of the utility is performed by inputs from Ten key and Touch panel.

Ten Key

Key	Description
0 to 9	Input numeral.
. (decimal)	Input decimal point.
-	Subtraction key
CLR	Cancel key for numeral input and release key for error condition.
ENTER	Confirmation key (same as "=" key) The key is represented as "=".

Touch Panel

Key	Description
C	Cancel key for numeral input and release key for error condition.
%	Percent calculation key.
AC	Clear key for releasing error conditions and numeral inputs except contents of the memory.
ENT	Confirmation key.
MC	Memory clear key.
MR	Memory read key.
M-	Memory subtraction key.
M+	Memory addition key.
+-	Arithmetic calculation keys.(addition, subtraction)
÷X	Arithmetic calculation keys (multiplication, division)

8.3   Clock Utility

Overview

The clock utility is used to reference the current time, set the date and time, or set an alarm.

This utility is provided as an EXE file and should be activated as command line or as a child-process of the application program.


Fig. 8.2

File Name

CLOCK.EXE

Function

The clock utility provides the following functions:

- Displays the current time in digital or analog mode. 12-hour system or 24-hour system can be selected for the digital display mode by the startup option.
- The current date is displayed with the following format: year/month/day/day of the week.
- The current time is displayed with the following format: hour/minute/second.
- 12 hour/24 hour system.
- Date and time can be set from 0 O'clock 0 minutes, January 1 (Tuesday) 1980 to 23 O'clock 59 minutes, December 31 (Sunday) 2079.
- An alarm can be set.
- A logo string can be specified by selecting an appropriate option at start up.

Condition of Operation

This utility requires two pages (32 Kbytes) of EMS memory and the driver, hardware window manager (**HWWMAN.EXE**) and keypad (**KEYPAD.EXE**) which must be resided always. Refer to Chapter 6.4 "Keypad Driver/Hardware Window Manager" for the detail.

Startup Method

This utility is not stored in the basic drive (C:). It must be copied to RAM disk (A:) or FROM drive (D:) for the utility to be started up. It can be used individually or called as child-process.

```
Format : CLOCK  [options]
```

Options	/D=	Specify the time system, 12-hour or 24-hour. /D=12 or /D=24		
	/F=	Specify the display method of date, month and year. The following display format is used to indicate.		
		YYYY	Year in 4 digits.	
		YY	Year in 2 digits (most least two digits of the year).	
		MMM	Month by abbreviation (three alphabets).	
		MM	Month in 2 digits (by numeral).	
		DD	Day in 2 digits (by numeral)	
		'-', '.', '/'	Characters on the left side are used as delimiter.	
		Ex.　/F=MMM-DD-YYYY　　/F=YY/MM/DD　　/F=YYYY.MM.DD	JAN-28-1998[WED] 98/ 1/28[WED] 1998. 1.28[WED]	
	/T=	Specify the logo of clock by characters. The maximum length of the logo can be 9 characters. Also, it is possible to include characters and numbers combined in the logo.		
		Ex.　/T=CASIO		

8.4 Calendar Utility

Overview

Use this calendar utility for referring to dates. This utility is provided as an EXE file and should be activated as command line or as a child process of the application program.

Fig. 8.3

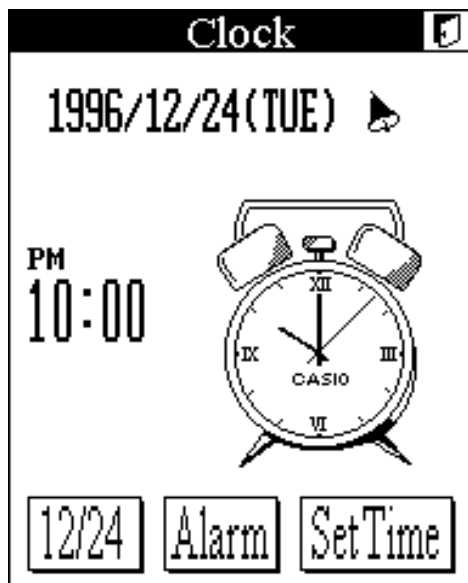**File Name**

```
CAL.EXE
```

Function

The calendar utility provides the following functions:

● Displays a calendar for two months on one screen page.

● Dates between January, 1980 and December, 2079 can be referenced.

Condition of Operation

This utility requires two pages (32 Kbytes) of EMS memory and the driver, hardware window manager (**HWWMAN.EXE**) and keypad (**KEYPAD.EXE**) which must be resided always. Refer to Chapter 6.4 "Keypad Driver/Hardware Window Manager" for the detail.

Startup Method

This utility is not stored in the basic drive (C:). It must be copied to RAM disk (A:) or FROM drive (D:) for the utility to be started up. It can be used individually or called as child-process.

8.5   FLINK Utility

Overview

The FLINK Utility is used to perform communication either between the IT-2000 and PC, or between two IT-2000s by means of the IrDA protocol. This utility is provided as an EXE file and should be activated as command line or as child-process of the application program.

Function

IrDA communication method setup	Sets the IrDA communication method.
File transmission	Transmits files.
File reception	Receives files.
File append	Appends (concatenates) a file on the transmission side to a file on the reception side.
File deletion	Deletes a file on the communication partner side.
File move	Moves a file within the same drive on the communication partner side.
Idle start	Passes the right of communication request to the communication partner and enters the command reception wait state.

File name:

```
FLINK.EXE
```

Startup Method

This utility is supplied on drive (C:). Usually this utility is made available after it is called from the system menu as a child process. However, it can be used either as a single command or as a child process to be called from another application.

Operation Method

With this utility operation priority is placed on only one side and the other side must remain in the command reception wait sate. This is true for both HT-to-HT communication and HT-to-PC communication. Hereinafter the operation side is referred to as the terminal, and the command reception wait side is referred to as the host.

To establish HT-to-HT communication, idle-start (host-start) FLINK on one side and specify the transmission or reception command to execute (terminal-start) FLINK.

To establish HT-to-PC communication, execute the communication host utility called "LMDOS" or "LMWIN" on the PC. For information about this communication host utility refer to the IT-2000 Upload/Download Utility Manual.

In the following pages the method used to specify the start options and information about each function is given.

8.5.1 Communication Parameter Setup Command (/L={,,,}

Sets up command parameters according to the command specified next to "=". If the communication environment command needs to be specified, this command must precede it.

Command Specification Method

FLINK /L={maximum IrDA speed, wait time until the connection is established, data transmission/reception wait time}

Always place the parameters between a pair or braces ("{   }"). Parameters do not need to be specified, however, commas (,,,) must be specified. If a parameter is not specified, the corresponding default values will be used.

IrDA communication speed

Input parameter	Baud rate (bps)	Remark
2400	2400	
9600	9600	
19 K	19200	
38 K	38400	
57 K	57600	
115 K	115.2 K	
576 K	576 K	
1 M	1 M	
4 M	4 M	Default value

Wait time until the connection is established

● Specify between 0 and 3600 seconds.

● If "0" is specified, the application will wait until the connection is established.

● The default value is 1800 seconds.

Data transmission/reception wait time

● Specify between 0 and 600 seconds.

● If "0" is specified, the application will wait until the communication function is normally or abnormally terminated.

● The default value is 300 seconds.

Example of specification

```
FLINK /L={4M, 20, }
```

Meaning:

Communication will be performed with a maximum IrDA speed of 4 Mbps, the wait time until the connection is established is 20 seconds, and the data transmission/reception wait time is default-set to 300 seconds.

8.5.2   File Transmission (/S)

Function

This function transmits a file from the terminal machine to the host machine. If the directory specified by the "storage destination directory name" does not exist on the host side, it will be automatically created. If the identical file name exists on the host side, it will be forcibly overwritten. Even if it is a read-only file, it is possible to overwrite by specifying the "O" option.

Startup Method

`FLINK /S[Option] transmission file pathname [transmission file pathname...] storage`

`destination directory name`

Options

Option	Description
O	If the host side has an identical file name and it is a read only file, it can be forcibly overwritten by specifying this option.
R	If this option is specified and if a wild card is used for the "transmission file pathname," all files under the specified directory including sub- and deeper directories will be transmitted. If the file name specified by the wild card does not exist in the sub-directory, it is not automatically created on the host side. If a wild card is not used, files included in the sub- and deeper directories will not be transmitted.
Q	Designates non-display of the message.
H	If HT-to-HT communication is to be performed, specify this option on the terminal.

Transmission file pathname

● Specify the file on the terminal machine by its full pathname and include the drive name.

● Wild cards (*, ?) can be used for the file name.

● If multiple "transmission file pathnames" are specified, separate each with a space.

Storage destination directory name

● Specify the storage directory name on the communication partner.

● The last parameter input is assumed to be the storage destination directory name.

● The directory name must include the drive name.

● Enter "＼" as the delimiter of the directory name.

Example of specifications of storage destination directory name

Specification of root directory	D:\
Specification of sub-directory	D:\TEST\BIN\
Incorrect specification	D:\TEST

Note:

If the host (reception) side has a file with the identical name, this command will forcibly overwrite that file. However, this overwrite operation is not unconditional. This command first creates a temporary file on the disk of the host, then it overwrites the file after the transmission has been completed. This is a safety measure to protect the original file from, for example, a file transmission failure. Accordingly, if the host side has a file with the identical name, there must be enough space on the disk to store the host-side transmission file. If there may not be sufficient disk space, files on the host side should be deleted in advance or the file delete command (on page 204) on the transmission side should be used to delete files on the host side.

Example of specifications

`FLINK /S A:\TEST\*.DAT D:\TEST2\`	This specification transfers all files that are in "A:\TEST" of the terminal and that have a "DAT" extension to "D:\TEST2\" on the host.
`FLINK /SR A:\TEST\*.DAT D:\TEST2\`	This specification transfers all files that are in "A:\TEST" (including sub-directories) of the terminal and that have a "DAT" extension to "D:\TEST2\" on the host.

8.5.3   File Reception (/R)

Function

This function receives a file from the host. The objective file name is specified by the full pathname (including the drive name) on the host. The received file is saved in the directory specified by the terminal side. If the specified directory does not exist on the terminal, it will be automatically created.

Startup Method

```
FLINK /R[Option] request pathname [request pathname...] reception directory
```

Options

Option	Description
O	If the host side has a file with the identical name and it is a read only file, it can be forcibly overwritten by specifying this option.
R	If this option is specified and if a wild card is used for "request pathname," all files under the specified directory including the sub- and deeper directories will be transmitted. If the file name specified by the wild card does not exist in the sub-directory, it is not automatically created in the host side. If a wild card is not used, files included in the sub- and deeper directories will not be transmitted.
Q	Designates non-display of the message.
H	If HT-to-HT communication is to be performed, specify this option on the terminal.

Request pathname

● Specify the objective file of reception which is on the host machine by its full pathname.

● Wild cards (*, ?) can be used for the file name.

● If multiple "request pathnames" are specified, separate each of them using a space.

Reception directory

● Specify the directory in which the received file is stored.

● The directory name must include the drive name.

● Enter "╲" as the delimiter of the directory name.

Example of specifications of storage destination directory name

Specification of root directory	D:╲
Specification of sub-directory	D:╲CASIO╲BIN╲
Incorrect specification	D:╲CASIO

Note:

If the terminal (reception) side has a file with the identical name, this command will forcibly overwrite that file. However, this overwrite operation is not unconditional. This command first creates a temporary file in the disk of the terminal, then it overwrites the file after transmission has been completed. This is a safety measure to protect the original file from, for example, a file transmission failure.

Accordingly, if the host side has a file with the identical name, there must be enough space on the disk to store the transmission-side transmission file. If there may not be sufficient disk space, files on the terminal side should be deleted in advance.

Example of specifications

`FLINK /R A:\TEST\*.DAT` `D:\TEST2\*.* B:\CHECK\`	This transfers all files that are in "A:\TEST" and that have a "DAT" extension, and all files included in "D:\TEST2\" from the host to "B:\CHECK\" on the terminal.
`FLINK /RR A:\TEST\*.DAT` `D:\TEST2\*.* B:\CHECK\`	This transfers all files that are in "A:\TEST" (including the sub-directory) and that have a "DAT" extension, and all files included in "D:\TEST2\" (including the sub-directory) from the host to "B:\CHECK\" on the terminal.

8.5.4   File Append (/A)

Function

This function appends (concatenates) a file on the terminal to the end of a specified file on the host. The objective file will be appended as a binary file. In other words, the data will be concatenated after the EOF code, if one exists. This function is valid only for transmission. Any files received from the host will not be concatenated to a file that exists on the terminal.

Startup Method

```
FLINK /A[Option] appended file pathname  target file pathname
```

Options

Option	Description
Q	Designates non-display of the message.
H	If HT-to-HT communication is to be performed, specify this option on the terminal.

Appended file pathname

● Specify the file to be appended by its full pathname, including the drive name.

● This file exists on the terminal side.

● Wild cards cannot be used.

Target file pathname

● Specify the target file to be concatenated by its full pathname, including the drive name.

● This file exists on the host side.

● Wild cards cannot be used.

● If the specified file does not exist on the partner side, a new file will be created with the specified pathname.

Example of specifications

FLINK /A A:\MY\CASIO.DAT B:\YOU\MASTER.DAT	This specification concatenates the "CASIO.DAT" file on the execution (transmission) side to the end of the "MASTER.DAT" file on the partner (reception) side.

8.5.5   File Deletion (/D)

Function
This function deletes a file on the host.

Startup Method
```
FLINK /D[Option] deleted pathname [deleted pathname...]
```

Option

Option	Description
H	If HT-to-HT communication is to be performed, specify this option on the terminal.

Deletion by pathname
- Specify the objective file to be deleted by its full pathname, including the drive name.
- If multiple "deleted pathnames" are specified, separate each using a space.

Example of specifications

`FLINK /D A:\TEST\*.DAT` `B:\TEST2\CHECK.DAT`	This specification deletes all files that are in "A:\TEST" and that have a "DAT" extension, and all files included in "B:\TEST2\CHECK.DAT" on the communication partner side.

8.5.6 File Move/Rename (/N)

Function

This function moves a file within the same drive or renames the file on the host. A file cannot be moved into a different drive.

Startup Method

```
FLINK /N[Option] move source pathname  move destination pathname
```

Option

Option	Description
H	If HT-to-HT communication is to be performed, specify this option on the terminal.

Move source pathname

● Specify the objective file to be moved or renamed on the host side by its full pathname, including the drive name.

● Wild cards cannot be used for the file name.

Move destination pathname

● Specify a file name used as the move destination or the resultant file name of rename.
  This file name must be specified by its full pathname, including the drive name.

● If the specified directory does not exist, it will be automatically created.

Example of specifications

FLINK /N A:\TEST\KK.DAT A:\TEST2\	This specification moves "A:\TEST\KK.DAT" to "A:\TEST2\" on the communication partner side.
FLINK /N A:\TEST\KK.DAT A:\TEST2\SJ.DAT	This specification renames "A:\TEST\KK.DAT" as " A:\TEST2\SJ.DAT" on the communication partner side.
FLINK /N A:\TEST\KK.DAT BA:\TEST2\SJ.DAT	A different drive cannot be specified. This specification results in an error.

8.5.7 Idle Start

Function

This function passes the right of communication request to the terminal and enters the command reception wait state. This function will be terminated if it is abnormally terminated, if it transmits a designation of termination, or if reception has been completed.

Startup Method

FLINK

  (No specific command exists.)

Example of specifications

FLINK (No command specification)	Waits for a request from the terminal.

Termination Codes and Messages

In the following table, termination codes and their error messages returned by FLINK.EXE are described.

Error Code		Error Message	Description
Category (High)	Detail (Low)		
Normal End			
0x00	0x00	NORMAL ENDING	Normal end.
0xDC	0x00	A～ZDRIVE FORMAT NOTICE	Format notification of drives A to Z.
0xF6	0x00	POWER OFF ENDING NOTICE	Notification of the end of the power.
0xF7	0x00	RESET ENDING NOTICE	Notification of the end of reset.
0xF8	0x00	BREAK KEY INTERRUPT ENDING	Notification of abortion by user.
Protocol Error			
0x01	0x00	COMMAND ERROR	Command error (undefined function code)
0x01	0x01	COMMAND ERROR	Command error (undefined sub-function code)
0x01	0x02	COMMAND ERROR	Command cannot be executed.
0x01	0x03	CHECK SUM ERROR	Check-sum error
0x01	0x04	COMMAND SEQUENCE ERROR	Command sequence error.
0x01	0x05	SEQUENCE NUMBER ERROR	Sequence number error.
0x01	0x06	OTHER PROTOCOL ERROR	Protocol is illegal.
0x01	0x07	PARAMETER ERROR	Parameter error.
0x01	0x08	TIMEOUT ERROR	Timeout error.
File Error (INT21h)			
0x02	0x02	FILE NOT FOUND	File cannot be found.
0x02	0x03	PATH NOT FOUND	Path cannot be found.
0x02	0x0B	INVALID FORMAT	Invalid formatting.
0x02	0x0F	INVALID DISK DRIVE	Invalid disk.
0x02	0x10	CANNOT DELETE DIRECTORY	Delete request is specified to current directory.
0x02	0x11	NOT SAME DISK	Disk is not the same.
0x02	0x12	FILE NOTHING	File cannot be found.
Note: Besides the detail codes which are defined in File Error (INT21h) above, other error codes may be returned as extension error code of DOS.			
File Error (INT24h)			
0x03	0x13	WRITE PROTECT ERROR	Write protect error.
0x03	0x14	UNKNOWN UNIT	Undefined unit.
0x03	0x15	DRIVE NOT READY	Drive is not ready.
0x03	0x17	DATA ERROR (CRC)	Data error.
0x03	0x19	SEEK ERROR	Seek error.
0x03	0x1A	UNKNOWN DISK FORMAT	Disk is not formatted.
0x03	0x1B	SECTOR NOT FOUND	Sector cannot be found.
0x03	0x1D	WRITE ERROR	Write error.
0x03	0x1E	READ ERROR	Read error.
0x03	0x1F	UNKNOWN ERROR	Error cannot be defined.
0x03	0x20	FILE SHARE ERROR	Specified file is already opened.
0x03	0x21	FILE LOCK ERROR	File lock error.
0x03	0x22	INVALID DISK CHANGED	Invalid disk exchange.
0x03	0x23	FCB FULL	FCB is full.
0x03	0x53	FATAL ERROR	Fatal error (Unsuccess INT24h).

Note: Besides the detail codes which are defined in File Error (INT21h) above, other error codes may be returned as fatal error code of DOS.			
Protocol Error (File)			
0x04	0x00	CANNOT OVERWRITE	File is read-only.
IrDA Protocol Error (For detail refer to the table on the next page.)			
0x80	0x01	IrDA PROTOCOL ERROR	Open error.
0x80	0x02		Data send error.
0x80	0x03		Data receive error.
0x80	0x04		Close error.
0x80	0x05		Error in setting of self-station ability.
0x80	0x06		Error in setting of communication status.
Internal Error			
0x0F	0x01	INTERNAL ERROR	Parameter error.
0x0F	0x02		Command buffer overflow.
0x0F	0x03		Analysis on received data.
-	-	RECEIVED ERROR REQUEST	When error notification is received from the communication partner.

The following error codes are output when an error occurs in the IrDA library.

IrDA Library Error		
Termination Code	Message	Description
0X000000001		Resources are not enough.
0X000000002		No device to connect.
0X000000004		No service available at the destination device.
0X000000008		Connecting is failed. Timeout to abort or to wait for the connection.
0X000000010		Opened file is accessed to open.
0X000000020		IR_OPEN is not executed.
0X000000040		Specifying WIRE is illegal.
0X000000080		Parameter error.
0X000000100		Tmeout to wait for send/receive.
0X000000200		Over-run error.
0X000000400		Parity error.
0X000000800		Flaming error.
0X000001000		CS signal timeout.
0X000002000		DR signal timeout.
0X000004000		CI signal timeout.
0X000008000		CD signal timeout.

8.6 XY Utility

Overview

The XY utility is used to perform communication either between an IT-2000 and PC, or between two IT-2000 terminals by means of XMODEM or YMODEM BATCH protocol.

This utility is provided as an EXE file and should be activated as command line or as child-process of the application program.

File name:

```
XY.EXE
```

Function

Transmission of a file	Transmits a file.
Reception of a file	Receives a file.
Selection of a protocol	Select either XMODEM protocol or YMODEM-BATCH protocol.
Specification of the error check method	Select the error check method as the checksum or CRC method.
Specification of a packet length	Select the packet length as 128 or 1024 bytes.
Specification of a baud rate	Select a baud rate between 1200 and 115200 bps.
Transmission of multiple files (only for YMODEM)	By using a wild card it is possible to transmit multiple files at one time. In addition, files included in the sub- and deeper directories can be transmitted.

Startup Method

This utility is supplied on drive (C:). Usually this utility is made available after it is called from the system menu as a child process. However, it can be used either as a single command or as a child process to be called from another application.

Note:

When the cable comes off while the communication takes place:

If the connection cable is accidentally unplugged while communication between the IT-2000 and PC is taking place, a communication error results and communication is interrupted. In this case the communication software on the PC will display an error message and interrupt transmission/reception, however, some data may remain in the transmission buffer. If an attempt is made to restart communication in this condition, the XY utility will receive illegal packets, hampering normal communication. If this occurs, terminate the communication software on the PC side then restart it to restore normal communication.

About time stamping of files:

This utility supports the function to exchange time stamp information between the transmitted file and received file. The time stamp information to be exchanged will be processed assuming that it is Greenwich standard time. In contrast, the time used by the IT-2000 is the local time, and the time stamp of IT-2000 files are accordingly controlled based on the local time.

The XY utility, for file transmission/reception by means of the YMODEM protocol, will convert a time stamp in Greenwich standard time to a time stamp in local time, or vice versa. This time conversion is achieved according to the environment variable, TZ. In communication between two IT-2000 terminals, if, for example, TZ of the transmission side is "JST+5", the time stamp of a file to be transmitted will have five hours added. In this case the reception side will create a file by subtracting five hours from the time stamp of the received file.

If the environment variable TZ is not set, this time conversion is not performed.

The time stamp made at XMODEM communication uses the system time of the reception side.

Transmission side   Reception side

IT-2000(TZ=none)	12:00 →	±0	→ 12:00 →	±0	→ 12:00	IT-2000(TZ=none)
IT-2000(TZ=GMT)	12:00 →	±0	→ 12:00 →	±0	→ 12:00	IT-2000(TZ=GMT)
IT-2000(TZ=JST+5)	12:00 →	+5	→ 17:00 →	-5	→ 12:00	IT-2000(TZ=JST+5)
IT-2000(TZ=JST+5)	12:00 →	+5	→ 17:00 →	?	→ ??:??	PC
PC	12:00 →	?	→ ??:?? →	-5	→ (??-5):??	IT-2000(TZ=JST+5)

About key input during communication:

Do not press any key during communication, otherwise file transmission/reception may be hampered.

Using this utility where COM2KEY.EXE is resident:

To use this utility where a debugging tool called COM2KEY.EXE is resident, the /N option must be specified. Since COM2KEY.EXE will transfer the displayed characters to the COM port, the characters displayed by this utility will also be transferred to the COM port, hampering normal transmission.

Function and operation method

Always specify necessary start parameters. These parameters include the essential command and its option, other parameters, and the transmitted/received file name. Each parameter must be separated by a space or TAB code.

```
XY /command+option  /parameter [/parameter...] file name [file name...]
```

Command

Always specify /S or /R. This command must be specified as the first parameter.

> `/R` | `/S`   Transmission or reception specification
>
>> `/R:`   File reception
>>
>> `/S:`   File transmission
>>
>> (Both /R and /S cannot be specified at the same time.)

Option

After the command, specify the appropriate options. The options must be specified in the following order:

> **X/Y**   Communication protocol specification. This must directly follow either /R or /S.
>
>> X: XMODEM protocol communication.
>>
>> Y: YODEM protocol communication.
>>
>> (Both X and Y cannot be specified at the same time.)
>
> **M | C**   Error check method. This can be specified only if either /R or /S is specified.
>
>> M: Checksum (only for XMODEM)
>>
>> C: CRC
>>
>> (Both M and C cannot be specified at the same time.)
>>
>> If this specification is not made, M is automatically used if XMODEM communication is specified, and C is automatically used if YMODEM communication is specified. The M specification will be invalid if the Y option is specified.
>
> **N | L**   Packet length.
>
>> N: Normal (128 bytes)
>>
>> L: Long (1024 bytes)
>>
>> (Both N and L cannot be specified at the same time.)
>>
>> If this specification is not made, N is automatically used if XMODEM communication is specified, and L is automatically used if YMODEM communication is specified.

Other parameters

Specify the options immediately after (without inserting a space) the command. Options must be specified in the following order:

> **/N**   Suppression of message display
>
>> Specify this option if a copyright message or error message is suppressed from being outputted.

/BN   Specification of a baud rate   (If omitted, 2 (9600 bps) is employed.)

N =   0:   2,400 bps
1:     4,800 bps
2:     9,600 bps
3:    19,200 bps
4:    23,040 bps
5:    28,800 bps
6:    38,400 bps
7:    57,600 bps
8:   115,200 bps

/P   For file transmission via YMODEM protocol this option sets a pathname on the destination side from the pathname of the object file that exists on the transmission source. This file name must be specified by its full pathname.

/PXXX   Modifies the pathname of a file to be transmitted via YMODEM protocol.
XXX= path (maximum 250 characters)

/U   With this option if a wild card is used for a file name to be transmitted via YMODEM protocol, files included in the sub-directory can be the objectives of file transmission. This option is also used to mirror-copy a drive.

File name

XMODEM: Transmission (/SX) : Specify only one file.

Reception (/RX)   : Specify one file name.

* Multiple files cannot be used.

* Wild cards cannot be specified.

YMODEM: Transmission (/SY) : Specify file names. Multiple files can be specified as a lump. If specifying multiple files, separate each of them using a space. Wild cards (*, ?) can be used.

Reception (/RY)   : Wild cards cannot be used.

Example of specifications

`XY /SY A:\WORK\TEST.DAT`	Transfers "A:\WORK\TEST.DAT" at transmission side. "TEST.DAT" can be copied in the current directory at reception side.
`XY /SY /P A:\WORK\TEST.DAT`	Transfers "A:\WORK\TEST.DAT" at transmission side. "A:\WORK\TEST.DAT" can be copied at reception side. If "A:\WORK" does not exist, it is created newly.
`XY /SY /P B:\TEST A:\WORK\TEST.DAT`	Transfers "A:\WORK\TEST.DAT" at transmission side. "B:\TEST\TEST.DAT" can be copied at reception side. If "B:\TEST" does not exist, it is created newly.

Termination Codes and Messages

Termination Code	Message	Description
00	NORMAL END	End normally.
01	ABNORMAL END	Abort by CLR key. Or, the communication partner aborts.
02	(Reserved)	
03	FILE NOT FOUND	Input file cannot be found.
04	FILE NOT CREATE	File cannot be created.
05	TIME OUT	Timeout has occurred.
06	(Reserved)	
07	WRITE FAILURE	Error in writing has occurred.
08	COMMUNICATION ERROR	Error during communication has occurred.
09	(Reserved)	
10	FILE SIZE ZERO	Size of specified file is 0 byte. (when XMODEM is used.)

8.7 Remaining Battery Voltage Display Utility

Overview

The remaining battery voltage display utility is used to monitor the remaining voltage of the battery. This utility is provided as an EXE file and should be activated as command line or as child-process of the application program.
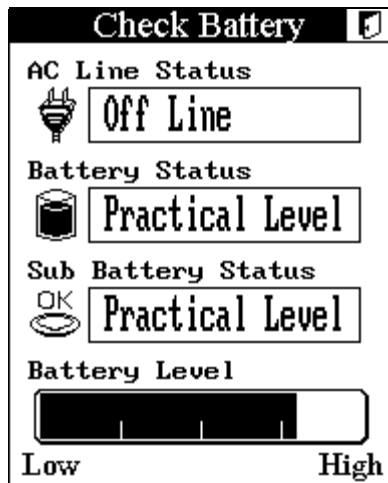

Fig. 8.4

File Name

```
CHKBATT.EXE
```

Function

Display for remaining battery voltage of main battery	The remaining battery voltage can be displayed as a percentage and as a bar chart. It can also display if the output voltage from the battery is low.
Display for power supply connection states	The connection status of AC adaptor and I/O Box can be displayed.
Display for remaining battery voltage of sub-battery	The remaining battery voltage of sub-battery can be displayed.

Note:

Display of remaining battery voltage is determined by checking on the voltage output by the main battery. The maximum indication of remaining battery voltage may not be displayed if the worn-out battery is used even if it is fully recharged.

Condition of Operation

This utility requires two pages (32 Kbytes) of EMS memory and the driver, hardware window manager (**HWWMAN.EXE**) and keypad (**KEYPAD.EXE**) which must be resided always. Refer to Chapter 6.4 "Keypad Driver/Hardware Window Manager" for the detail.

Startup Method

This utility is not stored in the basic drive (C:). It must be copied to RAM disk (A:) or FROM drive (D:) for the utility to be started up. It can be used individually or called as child-process.

8.8   Reverse Video Utility

Overview

This utility is used to change the entire screen to reverse video.

From the nature of the FSTN semi-transmittive type LCD unit of this terminal the density of colors (tones) will be reversed. So, for example, a light color appears dark and a dark color appears light. To avoid this problem use this supplied utility to represent colors as closely as possible.

File name

```
LCDREV.EXE
```

Startup Method

This utility is not supplied on the basic drive (C:). Copy it in the F-ROM drive (D:) or RAM disk (A:) before use. This program can be used either as a single command or as a child process.

Operation Method

```
Format: LCDREV Option
```

Option	Function
0	Normal (Returns to default value at a time of boot up)
1	Only text is reversed
2	Only graphics are reversed.
3	Both text and graphics are reversed.

8.9   COM2KEY Utility

Overview

This utility is a debug tool that allows key input at the DOS prompt.

If this utility is resident in memory, the data entered in COM1 will be passed to the key buffer, and the characters displayed on the DOS prompt screen will be outputted for COM1. Therefore, if this terminal is connected to a PC via the COM cable and if the terminal emulator is used on the PC, characters can be entered in the DOS prompt screen of this terminal through the PC's keyboard.

File name

```
COM2KEY.EXE
```

Operation Method

- Connect the COM1 (8-pin) port of this terminal to the COM port of the PC with a cable.
- Initiate the terminal emulator software on the PC and make the following setups.

    Baud rate   9600 bps
    Data bits   8 bits
    Parity bit   None
    Stop bit   1 bit

- Permanently install COM2KEY on the IT-2000 side with the following procedure.
- If a key input is made on the PC side, the entered character will be displayed in the DOS prompt screen of this terminal.

Startup Method

This utility is supplied and is stored in the basic drive (C:). This utility is an EXE file-type device driver. It can be used as a single command or specified by CONFIG.SYS.

If executed from DOS prompt line :

```
Format: COM2KEY [Option]
```

If specified by CONFIG.SYS :

```
Format: DEVICE=C:＼COM2KEY.EXE
```

Option	Function
None	Permanently install COM2KEY.
/R	Cancels residence of COM2KEY.

APPENDIX A   TFORMAT.EXE

In this chapter, TFORMAT.EXE, the formatter for F-ROM drive (D:) of IT-2000, is explained.
The TFORMAT.EXE is necessary to format the drive (D:). It is resided in the basic drive (C:).

The syntax of the TFORMAT command is;

```
TFORMAT [drive-letter]
[/LABEL:label]
[/SPARE:n]
[/Y]
Example of Syntax :  TFORMAT 2 /SPARE:64
```

Note:

Even if the TFORMAT.EXE is excuted with option attached, the usage of program can be observed
The following options are supported only by IT-2000.

Drive-letter	DOS drive letter of the F-ROM drive. The drive number of F-ROM in IT-2000 is set to 2. Always specify "2" for the drive.
/LABEL:label	A string to be used as the DOS label of the formatted medium.
/SPARE:n	Leave n Flash erase units as spare units for garbage collection. The default is 1. At least one unit should be specified for the Flash medium to operate as a true read-write device. More than one spare unit may be specified to format media that have bad Flash units. In such a case the number of spare units should exceed the number of bad units by at least 1. It is also possible to specify more than one spare unit in anticipation of Flash units becoming in the future. A value of 0 spare units may be specified to create a WORM (Write-Once-Read-Many) disk. When formatting with this option, the Flash medium can be written once only, after which it will become a read-only medium. File System will report that the medium is write-protected when space for writing is exhausted. This option provides very limited functionality, and should not be used except in special cases. The option has the advantage of lowering the formatting overhead of File System, since a spare Flash erase zone is not needed for spare reclamation.
/Y	Do not pause for confirmation before beginning to format.

APPENDIX B   PC Card Driver

In this chapter, each PC card driver which is called by CONFIG.SYS or by AUTOEXEC.BAT is explained. These PC card drivers and INI file are stored in the directory, C:\CARDSOFT, on the basic driver (C:).

SystemSoft's *CardWizard* PC card solution provides OEMs with a complete software solution for integrating PCMCIA controllers and slots into their computers. The *CardWizard* software suite provides a complete "plug and play" system software solution for both DOS and Windows 3.1. This solution consists of the following drivers and utilities. Please be aware that your particular configuration may not include all drivers and utilities.

Socket Services (SS365SL.EXE)

Socket Services provides a standard software interface to host controller chips and isolates the socket hardware from higher level software. Socket Services includes functions such as configuring a socket for an I/O or memory interface and controlling socket power voltages. The Socket Services driver included depends upon the host controller chip that the system supports.

> **Option**
>> `/SKT:x`   Number of supported slots
>>> Range: 1 to 4 (Default: 4)

Specifies the number of slots that driver supports. On machines that have a PCMCIA adaptor that can support more slots than are present in the machines, this value should be set to the exact number of slots present.

Card Services (CS.EXE)

The Card Services driver manages competition for system resources and manages adapter and card resources and configuration

> **Option**
>> `/POLL`   Poll for status change
>>> Range : 0 to 1 (Default : 0)

When set to 1, Card Services will not use a card-status-change interrupt to determine status changes on the system. It will instead poll for status changes (inserted card has been removed, empty slot is now occupied, etc.). This parameter should be set to 1 if the system does not have an available IRQ to use as a card-status-change interrupt, or if it does not support a card-status-change interrupt.

Card Identification (CARDID.EXE)

This client device driver detects the insertion and removal of PC cards, automatically determines the card type upon insertion, and then configures the card and slot/adaptor (if it is an I/O Card).

SRAM Card Driver (MTSRAM.EXE)

This SystemSoft device driver recognizes and supports SRAM cards.

IDE/ATA Support (ATADRV.EXE)

ATADRV.EXE is a block device driver that supports ATA Type II Flash Disk or ATA Type III hard disk PC cards.

Option

/S:x   Safe mode

Range: 0 to 8 (Default: 2)

Specifies if ATADRV is to be run in slave mode. The MTD Driver (MTDDRV) is the only master control driver currently available. Installs the ATADRV device driver as a slave(/S:x) to MTDDRV. It also specifies the number of devices (1 to 8) it can support. A value of 0 can also be used with /D or /S. When a value of 0 is used, only the mode that was specified (/D or /S) is implemented, not the number of devices assigned during installation or specified using the CONFIG utility. When this /S switch is used, ATADRV must be installed in CONFIG.SYS before MTDDRV and both ATADRV and MTDDRV must be installed before CARDID. Refer to ATA Driver Modes section which follows.

Option

/D:x   Number of Drive units

Range: 0 to 8 (Default: 2)

Specifies the number of drives that the system supports when installed either as a block device driver or as a slave device driver. Installs the ATADRV device driver as a block device driver (/D:x). It also specifies the number of drives (1 to 8) it can support. A value of 0 can also be used with /D or /S. When a value of 0 is used, only the mode that was specified (/D or /S) is implemented, not the number of drives assigned during installation or specified using the CONFIG utility. When the /D switch is used, ATADRV must be installed in CONFIG.SYS before utility. When the /D switch is used, ATADRV must be installed in CONFIG.SYS before CARDID.

Refer to ATA Driver Modes section which follows.

Card Service Power Management Enabler (CS_APM.EXE)

CS_APM.EXE is a DOS-based background task that enables Card Services to process system power management Suspend/Resume requests. When a Suspend request is initiated by system power management software, CS_APM notifies Card Services, which then verifies that the system PCMCIA slots are idle, and can be powered down. Card Services then passes this information back to CS_APM, which then notifies the power management software that the sockets can be powered off. When a Resume request is received by CS_APM, it informs Card Services, which then powers the sockets on again.

Memory Technology Driver (MTDDRV.EXE)

This component must be installed in order to support all Memory cards. It works in conjunction with card-specific MTDs to support a wide variety of current Flash Memory cards. It also supports SRAM cards (providing MTSRAM.EXE is also installed), and allows sharing of drive letters between the different types of memory cards (Flash, SRAM, and ATA).

SSVCD.386(SSVCD311.386 for Windows for Workgroups), SSVRDD.386, PCCARD.386 (for IT-2000W only)

These drivers permit hot insertion/removal of communications I/O, memory, and removable drive cards within Windows. These files are stored in the directory, `E:\Windows`.

APPENDIX C Acquisition of Suspend/Resume Event and Power Status

Overview

Suspend/Resume event is notified by multiplex interrupt (INT2Fh). If any event such as power ON/OFF occurs, consequently the interrupt (INT2Fh) will occur. An application can acquire the event by catching the interrupt. Since the interrupt INT2Fh is multiplex interrupt, application must reset values in all the registers to the previous values after catching the interrupt and then return the control to the old-vector.

Broadcast for Power Event	

INT2Fh	Input:
	    AH = 53h
	    AL = 0Bh
	    BH = (Reserved)
	    BL = 1 System wait request
	       = 2 System abortion request
	       = 3 Normal resume notification (if the method of the previous OFF
	                is by normal suspend.)
	       = 4 Critical resume notification (if the method of the previous OFF is by
	                critical suspend.)
	       = 5 Battery state notification
	
	Output:
	    BH = 80h Application refuses request.
	       = 00h Others
	
	 The power event is notified by POWER.EXE. In order to use the notification function, POWER.EXE must be pre-installed. An application must check first if the POWER.EXE has been installed or not by using the functions detailed below.

Function to Check POWER.EXE	

INT2Fh	Input:
	    AH = 54h
	    AL = 00h
	
	Output:
	    AX = 5400h Not installed.
	       = others Version numbers
	    BH = 50h "P"
	    BL = 4Dh "M"

Acquisition of Power Status

Application can acquire current power status by calling APM BIOS through the interrupt INT15h.

The following power statuses can be acquired by using the method.

- AC line status
- Battery status
- Battery flag
- Remaining battery life - percentage of charge
- Remaining battery life - time unites

The functions detailed below will acquire the power statuses stated above.

Acquisition of Power Status	

INT15h	Input:
	    AH = 53h
	    AL = 0Ah
	    BX = 0001h
	
	Output:
	If function successful:
	    Carry = 0
	    BH = AC line status
	            00h Off-line
	            01h On-line
	            02h On backup power
	            FFh Unknown
	            All other values are reserved.
	
	    BL = Battery status
	            00h High
	            01h Low
	            02h Critical
	            03h Charging
	            FFh Unknown
	             All other values are reserved.
	    CH = Battery flag
	            bit 0 = 1 High
	            bit 1 = 1 Low
	            bit 2 = 1 Critical
	            bit 3 = 1 Charging
	            bit 7 = 1 No system battery
	            All other bits are reserved.
	            FFh Unknown
	            All other values are reserved.
	    CL = Remaining battery life-percentage of charge
	            0 to 100 : Percentage of the battery charging, 100 represents full charge in battery.
	            FFh : Unknown
	            All other values are reserved.

DX =   Remaining battery life - time unit
              bit 15 =   0 : Time unit is in second.
                         1 : Time unit is in minute
              bits 14 to 0 =   value for second or minutes
                         0 to 7FFFh : Valid value for second or minute
                         FFh   : Unknown
If function unsuccessful:
Carry =   1
AH   =   Error code
         09h : Unrecognized device ID