

Digital
Human Machine Interface —

Pro-face

PL-5900 Series
User Manual

Digital

Digital Electronics Corporation

Introduction

Thank you for purchasing Proface's PL-5900 series Panel Computer, hereafter referred to as "the PL". This unit embodies Proface's latest, cost-effective architecture and is designed for Industrial Automation users.

Prior to using your PL, be sure to read this manual thoroughly to familiarize yourself with the unit's operation procedures and functions.

NOTE:

1. It is forbidden to copy the contents of this manual in whole, or in part, without the permission of the Digital Electronics Corporation.
2. The information in this manual is subject to change without notice.
3. This manual was written with care; however, if you should find any errors or omissions, please contact Digital and inform them of your findings.
4. Please be aware that Digital Electronics Corporation shall not be held liable by the user for any damages, losses, or third party claims arising from the uses of this product.

© 2000 Digital Electronics Corporation

MS-DOS®, Windows® 98, Windows® 2000 and WindowsNT®4.0 are registered trademarks of the Microsoft Corporation.

IBM®, and DOS® are registered trademarks of IBM.

AMD-K6®-III E+ is a trademark of Advanced Micro Devices, Inc.

Product names used in this manual are the trademarks of their respective manufacturers.

Essential Safety Precautions

This manual includes the following cautions concerning procedures that must be followed to operate the PL correctly and safely. Prior to operating the PL, be sure to read this manual and any related materials thoroughly to understand the correct operation and functions of this unit.

Safety Icons

To allow you to use the PL correctly, throughout this manual, the following icons are provided next to items requiring special attention.

These icons indicate the following levels of danger:



Indicates situations where severe bodily injury, death or major equipment damage may occur.



Indicates situations where slight bodily injury or machine damage can occur.

WARNINGS

- **To avoid the possibility of an electric shock, be sure to connect the power cord to the PL before connecting it to the main power supply.**
- **A fire or electrical shock may occur if voltages used with the PL are beyond the specified range. Be sure to use only the specified voltage.**
- **Before opening the PL's protective cover, be sure to turn the unit's power OFF. This is because the PL's internal parts carry high voltages.**
- **To avoid fires or electrical hazards, do not modify the PL in any way.**
- **Do not create touch panel switches that are used to either control or to ensure the safety of equipment and personnel. Mechanical switches, such as an emergency stop switch, a deadman (two-handed) start switch, etc., must be installed and operated via a separate control system.**

 **WARNINGS**

- After the PL's backlight burns out, unlike the PL's "Standby Mode", the touch panel is still active. If the operator fails to notice that the backlight is burned out and touches the panel, a potentially dangerous machine operation mistake can occur.

If your PL's backlight suddenly turns OFF, use the following steps to determine if the backlight is actually burned out.

- 1) If your PL is not set to "Standby Mode" and the screen has gone blank, your backlight is burned out.
 - 2) Or, if your PL is set to Standby Mode, but touching the screen does not cause the display to reappear, your backlight is burned out.
- If metal particles, water or other types of liquids contact any of the PL's internal parts, immediately turn the unit's power OFF, unplug the power cord, and contact either your PL distributor or the Digital Electronics Corporation.
 - Read and understand Chapter 4 "Installation and Wiring" thoroughly in order to select an appropriate installation location for the PL.
 - Before either plugging in or unplugging a board or interface connector, be sure to turn the PL's power OFF.
 - To prevent a possible explosion, do not install the PL in areas containing flammable gases.
 - The PL is not appropriate for use with aircraft control devices, aerospace equipment, central trunk data transmission (communication) devices, nuclear power control devices, or medical life support equipment, due to these devices' inherent requirements of extremely high levels of safety and reliability.
 - When using the PL with transportation vehicles (trains, cars and ships), disaster and crime prevention devices, various types of safety equipment, non-life support related medical devices, etc. redundant and/or failsafe system designs should be used to ensure the proper degree of reliability and safety.



CAUTIONS

- Never strike the touch panel with a hard, heavy or pointed object, or press on the touch panel too strongly, since it may damage the unit.
- Avoid exposing the PL to, or operating the PL in direct sunlight, high temperatures and humidity, and in areas where excessive dust and vibration will occur.
- Avoid using the PL in areas where sudden, extreme changes in temperature can occur. This may cause condensation to form inside the unit, possibly leading to an accident.
- To prevent the PL from overheating, be sure its air circulation vents are clear and clean, and keep the unit's operation area well-ventilated.
- Avoid operating or storing the PL near chemicals, or where chemicals can come into contact with the unit.
- When the Standard display is connected to the PL, after turning the display OFF, be sure to wait at least three (3) seconds before turning it ON again.

When PL Hard Disk (HDD) data is lost:

- The Digital Electronics Corporation cannot be held responsible or provide any compensation for damage(s) caused by the loss of data stored in the PL's hard disk drive (HDD). It is therefore strongly suggested that all important data and software be backed up regularly to an external data backup device.
- Please be aware that the Digital Electronics Corporation bears no responsibility for any damages resulting from the customer's application of this unit's hardware or software.
- Since the PL unit's hard disk drive (HDD) is a consumable item, i.e. it has a limited lifetime, be sure to back up its data regularly and prepare a spare HDD unit.
- To prevent file data damage, be sure to shut down the PL's OS before turning OFF the main power.
- After turning OFF the PL's power, wait until the internal HDD stops spinning before turning on the power again (approx. 5 seconds).

■ About the PL's Display Panel

- The PL's currently displayed data, its voltage and brightness setting each affect the intensity of *Contouring*. (i.e, when some parts of the screen are brighter than others, creating a wavelike pattern)
- There are minute grid-points (dark and light) on the Display Panel's surface. This is part of the PL's design and not a defect.
- Shadows may appear at the top of the LCD. This is normal for an LCD display.
- Sometimes the display area may look as if the display colors have changed. This is a common attribute of LCD's and is not a defect.
- Displaying a single image for long periods can cause an afterimage to remain when the display is changed to another screen. To prevent this, periodically turn the PL OFF and then ON again to remove this afterimage.

Table of Contents

Introduction	1
Essential Safety Precautions	2
Table of Contents	6
Documentation Conventions	10
PL Series Panel Types	10
Package Contents	11
Special Features	12
UL/c-UL(CSA) Application Notes	13
CE Marking Notes	13

Chapter1 PL Basics

1.1 PL System Design	1-1
1.2 PL System Design	1-2
1.3 Optional Items	1-3

Chapter2 Specifications

2.1 General Specifications	2-1
2.1.1 Electrical	2-1
2.1.2 Environmental	2-2
2.1.3 Structural	2-3
2.2 Functional Specifications	2-4
2.2.1 General	2-4
2.2.2 Display	2-4
2.2.3 Expansion Slots	2-4
2.2.4 Clock (RTC) Accuracy	2-4
2.3 Interface Specifications	2-6
2.3.1 Printer Interface (LPT1)	2-6
2.3.2 Keyboard Interface(KEY BOARD)	2-6
2.3.3 Mouse Interface(MOUSE)	2-7
2.3.4 RS-232C Interface (COM1/COM2/COM3)	2-7
2.3.5 RAS Interface(RAS)	2-8
2.4 PL Part Names and Features	2-10
2.5 External Dimensions	2-12
2.5.1 PL-5900T External Dimensions	2-12
2.5.2 PL-5900T with PL-FD500 External Dimensions	2-13
2.5.3 PL-5900T with Mirror Disk Unit External Dimensions	2-14
2.5.4 PL-5900T with PL-RC500 External Dimensions	2-15

2.5.5	PL-5901T External Dimensions	2-16
2.5.6	PL-5901T with PL-FD500 External Dimensions	2-17
2.5.7	PL-5901T with Mirror Disk Unit External Dimensions	2-18
2.5.8	PL-5901T with PL-RC500 External Dimensions	2-19
2.5.9	Panel Cut Dimensions	2-20

Chapter3 Installing Optional Units and Expansion Boards

3.1	Installation	3- 1
3.1.1	Removing the Rear Maintenance Cover	3- 2
3.1.2	Installing the DIM Module (PL-EM500/PL-EM128)	3- 3
3.1.3	Installing the FDD Unit (PL-FD500)	3- 4
3.1.4	Removing/Installing the HDD Unit (PL-HD220)	3- 7
3.1.5	Installing an Expansion Board	3- 8
3.1.6	Connecting the CD-ROM Drive Unit (PL-DK200)	3- 9

Chapter4 Installation and Wiring

4.1	Installation Cautions	4- 1
4.2	Installing the PL	4- 3
4.2.1	Installation Procedures	4- 3
4.3	Wiring the PL	4- 7
4.3.1	Connecting the Power Cord	4- 7
4.3.2	Power Supply Cautions	4- 10
4.3.3	Grounding Cautions	4- 11
4.3.4	Cautions When Connecting I/O Signal Lines	4- 11

Chapter5 System Setup

5.1	Setup Procedures	5- 1
5.2.1	Standard CMOS Features	5- 2
5.2	System Parameters	5- 2
5.2.2	IDE Primary Master/IDE Primary Slave	5- 3
5.2.3	Advanced BIOS Features	5- 4
5.2.4	Advanced Chipset Features	5- 7
5.2.5	Integrated Peripherals	5- 9
5.2.6	Super I/O Device	5- 11
5.2.7	Power Management Setup	5- 13
5.2.8	PM Wake Up Events	5- 15
5.2.9	PnP/PCI Configurations	5- 17
5.2.10	IRQ Resources	5- 19
5.2.11	DMA Resources	5- 20
5.2.12	PC Health Status	5- 21

Preface

5.2.13	Load Fail-Safe Defaults	5- 22
5.2.14	Load Optimized Defaults	5- 22
5.2.15	Set Password	5- 22
5.2.16	Save & Exit Setup	5- 22
5.2.17	Exit Without Saving	5- 22

Chapter6 OS Setup

6.1	CD-ROM Contents	6-1
6.1.1	Diagram	6-1
6.2	Setting Up Your PL OS	6-2
6.3	Installing Drivers	6-4
6.4	Windows NT® 4.0 / Windows ®2000 Cautions	6-9
6.4.1	Automatic System Log-On Setup	6-9
6.4.2	Using an Uninterrupted Power Supply	6-10
6.4.3	When Changing the System Design	6-10
6.4.4	Changing to the NTFS File System	6-11
6.5	Windows® Utility Program	6-12
6.5.1	API-DLL	6-12
6.5.2	Backlight OFF Screen Saver(Backlight control.scr)	6-12
6.5.3	Screen Display ON/OFF Utility(Disp.exe)	6-13
6.5.4	Keyboard Emulator(Keyclick.exe)	6-13
6.5.5	System Monitor/RAS Application (Pl_smon.exe/Pl_wps.exe)	6-13
6.5.6	Function Key Utility(Funckey.exe)	6-14
6.6	MS-DOS® Utility Programs	6-15
6.6.1	Touch Panel Handler(Atph59.exe)	6-15
6.6.2	Serial Port Driver(EXTCOM.SYS)	6-23
6.6.3	Touch Panel Data Calibration(CALIB59.EXE)	6-30
6.6.4	Keyboard Emulator(KEYEM_PL.EXE)	6-32
6.6.5	Backlight Burnout Detection Features Setting Program (BLSET.EXE)	6-39

Chapter7 Maintenance and Inspection

7.1	Regular Cleaning	7- 1
7.1.1	Cleaning the Display	7- 1
7.1.2	Replacing the Installation Gasket	7- 2
7.2	Replacing the Backlight	7- 2
7.3	Periodic Maintenance Points	7- 6




Appendices

A.1	Hardware Configuration	App-1
A.1.1	I/O Map	App-1

	A.1.2	Memory Map	App-2
	A.1.3	Interrupt Map	App-3
A.2		RAS Feature	App-4
	A.2.1	PL's RAS Features	App-4
	A.2.2	RAS Feature Details	App-5
	A.2.3	RAS Feature Overview	App-9
A.3		System Monitor	App-10
	A.3.1	Setup Procedure	App-10
	A.3.2	System Monitor Property Settings (PL_Wps.exe)	App-11
	A.3.3	System Monitor Operation (PL-Smon.exe)	App-12
	A.3.4	Error Messages	App-14
A.4		Serial Communication	App-16
A.5		Touch Panel Handler	App-17
A.6		BIOS List	App-25
A.7		System Monitor/RAS Feature API-DLL	App-39
	A.7.1	Operation Environment	App-39
	A.7.2	Class Contents	App-41
	A.7.3	Visual C Functions	App-43
	A.7.4	Visual C Function Specifications (Details)	App-43
	A.7.5	Visual C++ Functions	App-63
	A.7.6	Visual C++ Function Specifications (Details)	App-65
	A.7.7	Visual Basic Functions	App-93
	A.7.8	Visual Basic Function Specifications (Details)	App-93
A.8		Backlight Control API-DLL	App-116
	A.8.1	Operation Environment	App-116
	A.8.2	Class Contents	App-118
	A.8.3	Visual C Functions	App-119
	A.8.4	Visual C Function Specifications (Details)	App-119
	A.8.5	Visual C++ Functions	App-121
	A.8.6	Visual C++ Function Specifications (Details)	App-121
	A.8.7	Visual Basic Functions	App-124
	A.8.8	Visual Basic Function Specifications (Details)	App-124

Documentation Conventions

The list below describes the documentation conventions used in this manual.

Symbol	Meaning
	Indicates important information or procedures that must be followed for correct and risk-free software/device operation.
	Provides useful or important supplemental information.
*1	Indicates useful or important supplemental information.
	Refers to useful or important supplemental information
1) , 2)	Indicates steps in a procedure. Be sure to perform these steps in the order given.
PL	Abbreviation for the PL-5900 Series Industrial Computers.

PL Series Panel Types

DC24V Series Unit Model Numbers:

PL590* - T* *
 A B C D E

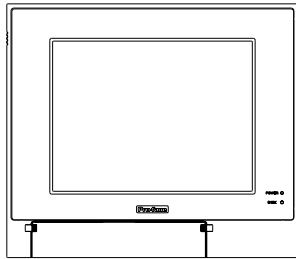
Item	Code	Meaning
A	PL590	PL-5900 Series Unit
B	0	3-slot type
	1	1-slot type
C	T	TFT Color LCD display
D	1	AC100V Model (no certification)
	4	CE Marking, UL/c-UL(CSA) Approval
E	*	Revision No.

Package Contents

The PL's packing box contains the items listed below. Please check to confirm that all items shown below have been included.

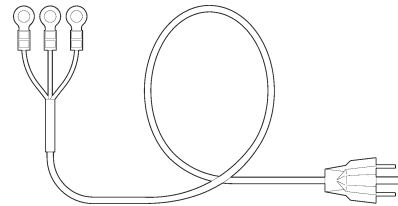
■ PL Unit

PL-5900T/PL-5901T

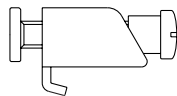


■ Power Cord

(included in the PL5900-T12/PL5901-T12)



■ Installation Fasteners (4 brackets/set)



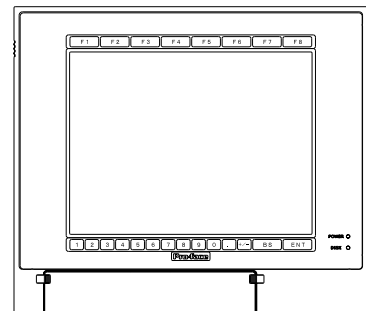
■ Installation Gasket



■ Function Labels



Attach the function labels as shown below.



■ PL-5900 Series User Manual & Driver CD



- Be careful when handling the PL not to damage the built-in HDD
- This cord is designed only for AC100V use. Any other voltage will require a different cord.
- If your PL unit contains a built-in accessory, that accessory's Installation Guide will also be included in the PL's packing box. Please check that all items normally included with that accessory are also included in this box.



- The CD-ROM contains User Manual and PL-5900 Series Utility and Driver. For details, Chapter 6 - Setting Up Your PL OS.

Special Features

The PL-5900 series displays are equipped with the following features:

■ **The Latest, High-Performance Architecture**

Designed around the AMD-K6[®]-III E+ 500 MHz CPU, the PL utilizes the type of high-performance architecture that offers you superior compatibility.

■ **Bright 10.4" LCD with a Wide Viewing Angle**

The PL's large 10.4-inch 640 x 480 dot TFT LCD display offers excellent visibility and brightness.



This top of the line TFT color LCD model allows you to create detailed and powerful visual images, with excellent brightness, a wide viewing angle, and a display capable of 64k colors.

■ **Easy Front Panel Installation**

The PL is designed to be installed easily into the front of any panel or device. It is also rugged enough for use in harsh, industrial environments, such as those found in the factory automation industry and its front panel boasts an IP65f equivalent rating.

■ **High Resolution, Analog Type Resistive Film Touch Panel**

Standard equipment with the PL is a high resolution 1024 x 1024 touch panel. Also, the separately sold mouse emulation utility provides mouse-like functionality and pointer control.

■ **Highly Expandable**

The PL units consist of two types; a 1-slot type (with 1 PCI bus also available), and a 3-slot type (with 2 PCI buses available). These slots can accommodate both Digital's own optional boards as well as other commercially available expansion boards.

Digital also offers a wide variety of optional products, such as FDD units, DIM memory modules and others.

UL/c-UL(CSA) Application Notes

The PL5900-T42-24V/PL5901-T42-24V series units are UL/c-UL (CSA) 1950 recognized products. (UL File No. E171486). Please pay special attention to the following instructions when applying for UL/c-UL approval for machinery which includes any of these PL units.

Equipment with a PL mounted in it requires UL/c-UL evaluation for the combination of the PL and equipment.

The PL conforms as a component to the following standards:

UL 1950, Third Edition, dated March 1,1998 (Standard for Safety of Information Technology Equipment, including Electrical Business Equipment)

CSA-C22.2 No. 950-M95 (Standard for Safety of Information Technology Equipment, including Electrical Business Equipment)

PL5900-T4* (UL Registration Model: 2880065-02)

PL5901-T4* (UL Registration Model: 2880065-01)

- The PL should be used as a built-in component of another product.
- Use the PL indoors only.
- When connecting the PL's power cord, be sure to use a cord that is appropriate for the current and voltage used, and that has conductive wires that are 0.75 mm² or larger.
- When an end-use product will include the PL, be sure to design the PL's power cut-off switch as a separate disconnect device and locate it where the operator can easily reach it.
- Danger of explosion if backup battery is incorrectly replaced. Replaced only with same or equivalent type recommended by the manufacturer. Dispose of used batteries according to the manufacturer's instructions.
- Be sure the unit the PL is built into is a (c)UL1950 approved structure.

CE Marking Notes

The PL5900-T42-24V/PL5901-T42-24V series units are CE marked product that conforms to EMC directive EN55011 (Group 1 Class A) and EN61000-2.

Memo

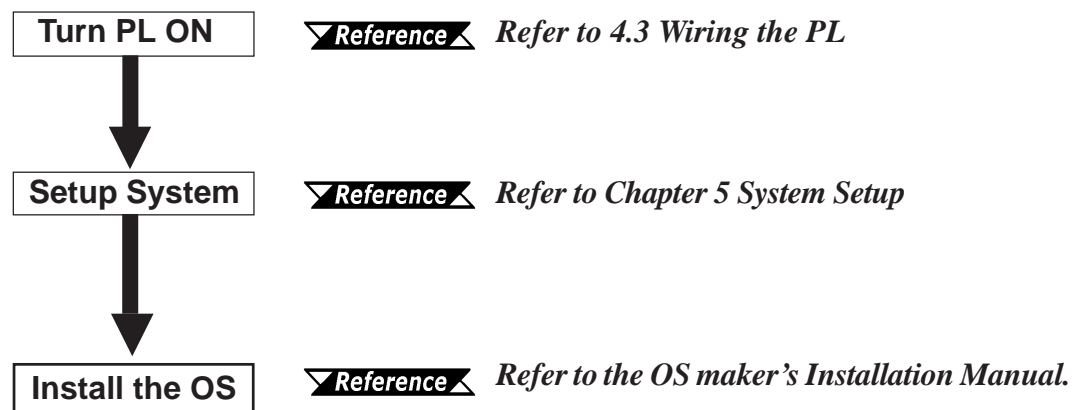
Chapter

1 PL Basics

1. PL System Design
2. Optional Items
3. PL Series Panel Types

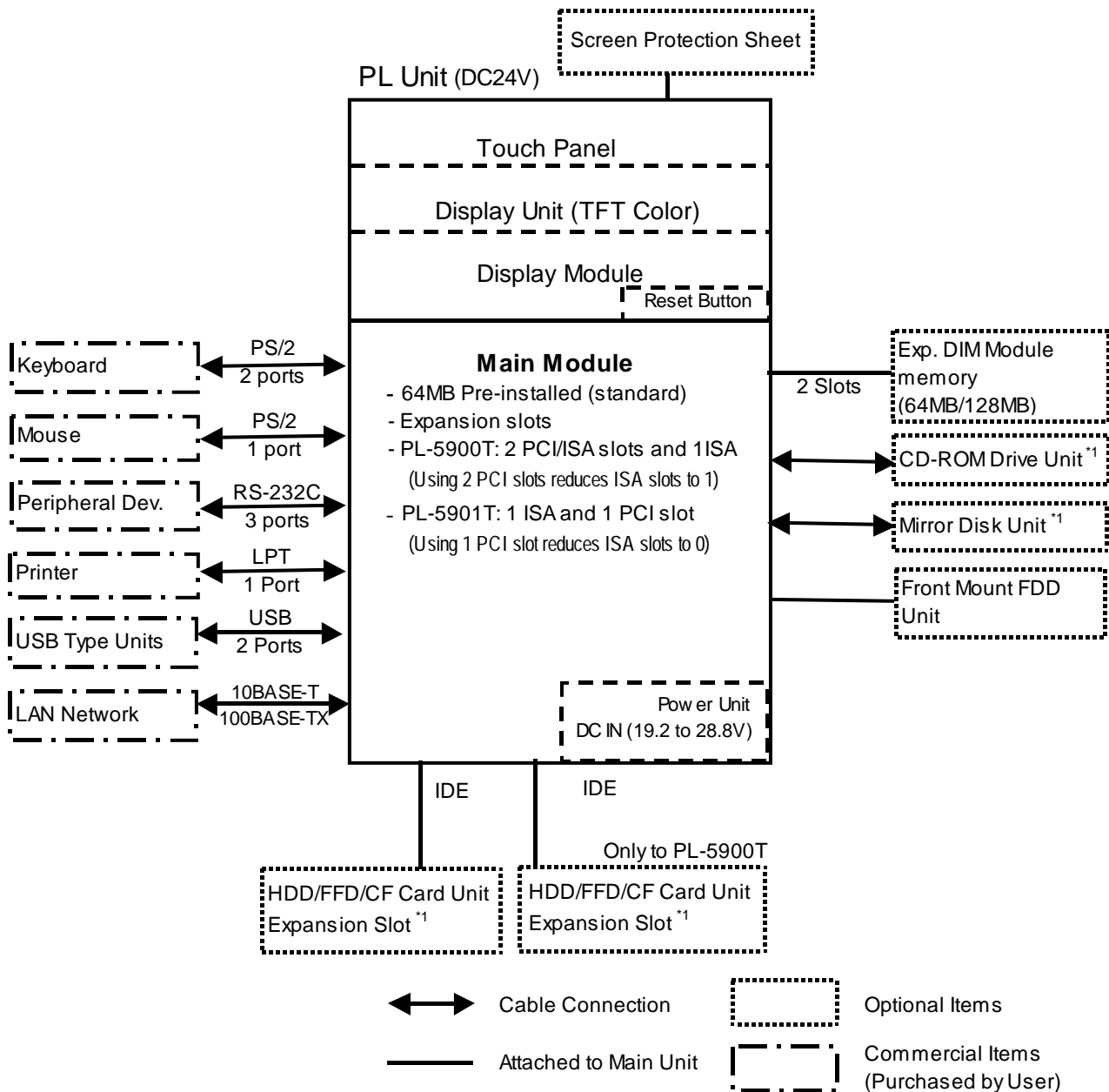
1.1 PL System Design

Prior to actual use, be sure to setup your PL as follows.



- After completing the hardware setup, before any data or applications can be installed on the hard disk drive, the OS (Windows® or MS-DOS®) must be used to initialize the HDD and create partitions. For details concerning these procedures, refer to the OS maker's installation manual.
- After turning the PL OFF, be sure to wait at least 5 seconds before turning ON again. If the unit is started within 5 seconds, it may not start up correctly.
- The PL is designed for use with the MS-DOS®, Windows®98 (SR2), or WindowsNT® 4.0, Windows®2000 operating systems. Other operating systems are not supported by this PL's driver software.

1.2 PL System Design



The above system configuration illustrates the PL's internal design and range of connectable peripherals. The user's actual configuration may differ.

^{*1} Certain limitations exist for the combinations of the HDD Unit, FDD Unit, the CF Card unit, the Mirror Disk Unit and the CD-ROM drive.

Reference 1.3 Optional Items

1.3 Optional Items

■ Options

Name	Model number	Description
DIM Module	PL-EM500	SDRAM (DIMM) Provides 64MB of memory
	PL-EM128	SDRAM (DIMM) Provides 128MB of memory
HDD Unit	PL-HD220	10GB 2.5" HDD Unit (OS not included)
FDD Unit	PL-FD500	IBM PC Compatible 3.5" FDD unit (Attaches to front slot)
FFD Unit (Flash File Disk)	PL-FF210	Flash File Disk Provides 32MB of memory, connected to IDE I/F. Used as HDD.
CD-ROM Unit	PL-DK200	IDE (ATAPI) compatible CD-ROM drive unit – for development and maintenance use. (special connection cable is included with CD-ROM unit)
CF Card Unit	PL-CF200	Designed exclusively for 5V type cards.
CF Card	GP077-CF20	CF card (16M). CF Card Unit PL-CF200 is required.
	GP077-CF30	CF card (32M). CF Card Unit PL-CF200 is required.
Mirror Disk Unit	PL-MD200- HU01	IDE compatible mirror disk unit without OS for data protection in case of HDD malfunction. Contains 2 2.5inch, 2.1GB drives.
Soft Mirror Utility	PL-SM500	Provides RAID Level 1 protection, without Mirror Disk unit.
RS-232C/RS-485 Adaptor	PL-RC500	Converts an RS-232C interface to an RS-485 interface. Connects to COM3.
Screen Protection Sheet	PL-CS001	Disposable, dirt-resistant sheet for screen protection. The Touch Panel can be used through this sheet. (10 sheets/set)
Glare Resistant Sheet	PL-NGS01	Disposable, glare-resistant sheet for screen protection. The Touch Panel can be used through this sheet. (10 sheets/set)
Mouse Emulator V2	PL-TD000	This software adds mouse and keyboard-like functionality to the Touch Panel. (Windows® 98, Windows® 2000 and WindowsNT® 4.0 only)



- **Since the PL's hard disk drive (HDD) is a consumable item, i.e. it has a finite usage lifetime, be sure to back up its data frequently and perform regular maintenance.**
- **The Hard Disk lifetime given here may be reduced due to unforeseen environmental factors, however, generally speaking, at an operating temperature of 20°C the disk should last for 20,000 hours (of operation) or approximately 5 years, whichever comes first.**



- The PL is equipped with three IDE interfaces, two of which can be used by the HDD or FFD units (PL-5901T can use only one), and one which can be used by either the CD-ROM drive or the Mirror Disk unit. Physically, even though up to three IDE drive units can be connected at the same time, IDE interface specifications require that a controller's simultaneous operation be limited to a single master and slave unit, for a total of two devices. The following chart shows the combinations available when using two IDE units (PL-5901T can use only one).

HDD Unit	MS	M	M	M	M	S					S				S			
FFD Unit		S				M	MS	M	M	M		S				S		
Mirror Disk Unit			S					S			M	M	M	M			S	
CD-ROM Drive Unit				S					S				S					S
CF Card Unit					S					S				S	M	M	M	M

MS: Combination of 2 units - Master or Slave, is possible.
 M: Used only for Master.
 S: Used only for Slave.

■ Maintenance Options

Name	Model number	Description
Mirror Disk Unit Replacement HDD	PL-MD200-MD01	Mirror Disk Unit's replacement HDD (1).
Installation Fasteners	GP070-AT01	Used to install the PL into a panel or cabinet. Same as original equipment brackets. (4 brackets/set)
Installation Gasket	PL-WS500	Used to prevent moisture from entering into the PL's case from the front face. Same as original equipment gasket.
Backlight	GP577T-BL00-MS	Spare Backlight for maintenance. (2 bulbs/set)



- *Since the PL's hard disk drive (HDD) is a consumable item, i.e. it has a finite usage lifetime, be sure to back up its data frequently and perform regular maintenance.*
- *The Hard Disk lifetime given here may be reduced due to unforeseen environmental factors, however, generally speaking, at an operating temperature of 20°C the disk should last for 20,000 hours (of operation) or approximately 5 years, whichever comes first.*

■ Commercially Available Items

The PL-5900 Series units can all use commercially available expansion boards (PCI/ISA compatible) as well as a standard keyboard, mouse, printer, etc. However, among the commercially available USB devices, not all will be compatible with the PL unit. For a list of the USB devices that can be used with your PL, please contact your local PL distributor.



- **Be sure to use only DIM modules manufactured by Digital. Installing other DIM modules may result in either damage to or failure of the PL, and will void your warranty.**
- **When using USB type devices, be sure they are USB compatible, and be sure to read that device's installation guide prior to connecting it to the PL.**

Memo

Chapter

2 Specifications

1. General Specifications
2. Functional Specifications
3. Interface Connector Specifications
4. PL Part Names and Features
5. PL Dimensions

2.1 General Specifications

2.1.1 Electrical

■ PL5900-T12, PL5901-T12

	PL5900-T12	PL5901-T12
Rated Voltage	AC100V	
Voltage Range	AC85V to AC132V	
Frequency	50/60Hz	
Allowable Voltage Drop	1 cycle or less (however, pause occurrences must be more than 1 second apart)	
Power Consumption	150VA or less	110VA or less
Voltage Endurance	AC1500V 20mA for 1 minute (between charging and FG terminals)	
Insulation Resistance	10M Ω or higher at DC500V (between charging and FG terminals)	

■ PL5900-T42-24V, PL5901-T42-24V

	PL5900-T42-24V	PL5901-T42-24V
Rated Voltage	DC24V	
Voltage Range	DC19.2V to DC28.8V	
Allowable Voltage Drop	10 ms or less (however, pause occurrences must be more than 1 second apart)	
Power Consumption	100W or less	80W or less
In-rush Current	30A or less	
Voltage Endurance	AC1000V 10mA for 1 minute (between charging and FG terminals)	
Insulation Resistance	10M Ω or higher at DC500V (between charging and FG terminals)	

2.1.2 Environmental

Ambient Operating Temperature	0°C to 45°C (with HDD attached: 5°C to 45°C)
Storage Temperature	-10°C to +60°C
Ambient Humidity	10%RH to 85%RH (A wet bulb temperature of 29°C or less)
Air Purity Level	0.1mg/m ³ or less (free of conductive particles and dust)
Atomosheric Pressure Resistance	800 to 1114hPa (2000 meters or lower)
Vibration Resistance	19.6m/s ² at 10Hz to 25Hz in X, Y, Z directions for 30 minutes With HDD attached: 4.9m/s ² With FD unit attached: 9.8m/s ²
Noise Endurance	Noise Voltage: 1500Vp-p Pulse Width: 50ns, 500ns, 1ms Rise Time: 1ns (via noise simulator)
Electrostatic Discharge Immunity	6kV IEC 61000-4-2 Level 3
Noise Immunity	Power Line: 2kV IEC 61000-4-4 Level 3



- *When using any of the PL's optional devices, be sure to check that device's specifications for any special conditions or cautions that may apply to its use.*
- *Since the PL unit's hard disk drive (HDD) is a consumable item, i.e. it has a limited lifetime, be sure to back up its data regularly and prepare a spare HDD unit.*
- *The Hard Disk lifetime given here may be reduced due to unforeseen environmental factors, however, generally speaking, at an operating temperature of 20°C the disk should last for 20,000 hours (of operation) or approximately 5 years, whichever comes first.*
- *Using the Hard Disk in an environment that is excessively hot and/or humid will shorten the disk's usage lifetime. A wet bulb temperature of 29°C or less is recommended. This is equivalent to the following data.*

Temperature	Humidity
at 35°C	no higher than 64%RH
at 40°C	no higher than 44%RH

2.1.3 Structural

	PL-5900T	PL-5901T
Grounding	Exclusive grounding: Use your country's applicable standard.	
Rating (Front face of installed unit)	Equivalent to IP65f (JEM 1030) ^{*1}	
Weight	6.0 kg (13.2 lb) or less	5.5 kg (12.1 lb) or less
Cooling Method	Natural air ventilation	
External Dimensions	W 311mm[12.24in.] x H 271mm[10.67in.] x D 130mm[5.12in.] (excluding projections)	W 311mm[12.24in.] x H 271mm[10.67in.] x D 93mm[4.57in.] (excluding projections)
Dimensions Including FDD Unit	W 311mm[12.24in.] x H 271mm[10.67in.] x D 130mm[5.12in.] (excluding projections)	W 311mm[12.24in.] x H 271mm[10.67in.] x D 123mm[4.84in.] (excluding projections)
Dimensions Including Mirror Disk Unit	W 311mm[12.24in.] x H 271mm[10.67in.] x D 180mm[7.09in.] (excluding projections)	W 311mm[12.24in.] x H 271mm[10.67in.] x D 143mm[5.63in.] (excluding projections)
Dimensions Including RS- 232C/RS-485 Conversion Unit	W 311mm[12.24in.] x H 271mm[10.67in.] x D 152mm[5.98in.] (excluding projections)	W 311mm[12.24in.] x H 271mm[10.67in.] x D 115mm[4.53in.] (excluding projections)

**1 The front face of the PL unit, installed in a solid panel, has been tested using conditions equivalent to the standard shown in this specification. However even though the PL unit's level of resistance is equivalent to the standard, oils that should have no effect on the PL can possibly harm the unit. This can occur in areas where either vaporized oils are present, or where low viscosity cutting oils are allowed to adhere to the face of the unit for long periods of time. If the PL's front face protection sheet becomes peeled off, these conditions can lead to the ingress of oil into the PL and separate protection measures are suggested. Also, if non-approved oils are present, it may cause deformation or corrosion of the front panel's plastic cover. Therefore, prior to installing the PL be sure to confirm the type of conditions that will be present in the PL's operating environment.*

2.2 Functional Specifications

2.2.1 General

CPU		AMD-K6 [®] -III E+ 500MHz Processor		
DRAM (SDRAM DIMM)		64MB Standard (2 DIMM sockets: max. 256MB)		
BIOS		AWARD PC/AT Compatible		
Secondary Cache Memory		256KB (built-in)		
Graphics		VGA (640 x 480 dots) VESA 16 colors/256 colors/16-bit color		
Video Memory		UMA (Unified memory architecture) type		
Touch Panel	Type	Resistive Film (Analog)		
	Resolution	1024 x 1024		
	Interface	COM4 : uses Mouse Emulator		
Interfaces	Serial	RS-232C	COM1	D-Sub 9 pin male side
		(w/FIFO)	COM2	D-Sub 9 pin male side (RI/+5V Changeover)
			COM3	D-Sub 9 pin male side (RI/+5V Changeover)
	Printer	Centronics Standard (ECP/EPP equivalent) D-sub 25 pin, female		
	Keyboard	PS/2 Interface (mini DIN 6 pin, female)		
	Mouse	PS/2 Interface (mini DIN 6 pin, female)		
	USB^{*1}	USB 1.0 Interface (side/front)		
	Network	IEEE802.3 10BASE-T, 100BASE-TX side		
	RAS	RAS Interface (D-sub 25 pin, male)		
Disk I/F	FDD Unit	Front Access/ 2 modes/ 3.5 inch FD		
	E-IDE	Side-mount 2.5 inch HDD I/F PL-5900T: 2 slots PL-5901T: 1 slot Rear-mount Mirror Disk/CD-ROM (1 slot)		

**1 Since MS-DOS[®] and WindowsNT[®]4.0 do not support this function, this feature cannot be used if those OS types are installed in the PL.*

2.2.2 Display

Display Type	TFT Color LCD
Resolution	640 x 480 pixels
Dot Pitch	0.33 mm x 0.33 mm
Effective Display Area	W211.2 mm x H158.4 mm
Display Colors	16-bit color
Contrast Control	Not available
Backlight	CFL (User replaceable)
Backlight Lifetime	50,000 hours or longer at an ambient temperature of 25°C. (Until the backlight's brightness dims to half of the original level.)



When it is time to change the backlight, please contact your local PL distributor.



7.2 Replacing the Backlight.

2.2.3 Expansion Slots

■ PL-5900T

	Board Size		Slot Pitch	Board Thickness
	PCI (Rev.2.1, 5V/32bit)	ISA		
1 st slot	180 x 122mm	180 x 122mm	—	Less than 13mm
2 nd slot	210 x 122mm	180 x 122mm	25mm	Less than 18mm
3 rd slot	None	210 x 122mm	20mm	Less than 13mm
Power Supply	3.3V : 1A 5V : 3A 12V : 0.6A -5V : 0.1A -12V : 0.1A (total for 3 slots)			



Note: For the 1st and 2nd slots either a PCI or an ISA type expansion board can be used.

■ PL-5901T

	Board Size		Slot Pitch	Board Thickness
	PCI (Rev.2.1, 5V/32bit)	ISA		
1 st slot	180 x 122mm	210 x 122mm	—	Less than 13mm
Power Supply	3.3V : 1A 5V : 1A 12V : 0.5A -5V : 0.1A -12V : 0.1A			

2.2.4 Clock (RTC) Accuracy

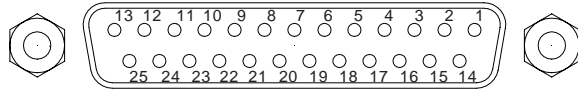
Clock(RTC) accuracy	±180 seconds per month
---------------------	-------------------------------

The PL unit's built-in clock (RTC) has a slight error. At the PL's specified ambient temperature and with the power turned OFF the error is ±180 seconds per month. However, ambient temperature fluctuations and the age of the unit may increase this error to ±300 seconds per month. If the PL unit's RTC clock accuracy is vital to system performance, regular adjustment of this clock is required.

2.3 Interface Specifications

2.3.1 Printer Interface (LPT1)

D-sub 25 Pin (Female)



- O.D.: Open Drain
- T.S.: 3-state Input
- TTLIN: TTL Input

Screw Size: (4-40UNC): Inch Type

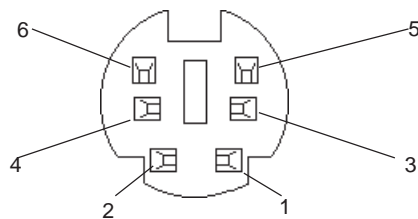
Pin No.	SPP/ECP Mode Signal Name	EPP Mode Signal Name	Direction	Electrical Specif.	Pin No.	SPP/ECP Mode Signal Name	EPP Mode Signal Name	Direction	Electrical Specif.
1*1	STRB	WRITE	In/Output	O.D/T.S	14*1	AUTOFD	DSTRB	In/Output	O.D/T.S
2	DATA0	DATA0	In/Output	T.S	15	ERROR	ERROR	Input	TTL
3	DATA1	DATA1	In/Output	T.S	16*1	INIT	INIT	In/Output	O.D/T.S
4	DATA2	DATA2	In/Output	T.S	17*1	SLCTIN	ADSTRB	In/Output	O.D/T.S
5	DATA3	DATA3	In/Output	T.S	18	GND	GND		
6	DATA4	DATA4	In/Output	T.S	19	GND	GND		
7	DATA5	DATA5	In/Output	T.S	20	GND	GND		
8	DATA6	DATA6	In/Output	T.S	21	GND	GND		
9	DATA7	DATA7	In/Output	T.S	22	GND	GND		
10	ACKNLG	ACKNLG	Input	TTL	23	GND	GND		
11	BUSY	WAIT	Input	TTL	24	GND	GND		
12	PE	PE	Input	TTL	25	GND	GND		
13	SLCT	SLCT	Input	TTL					

*1 When using the printer interface in SPP mode, pins 1, 14, 16 and 17 become O.D.
When using ESC or EPP modes, these pins will change to T.S.

2.3.2 Keyboard Interface(KEY BOARD)

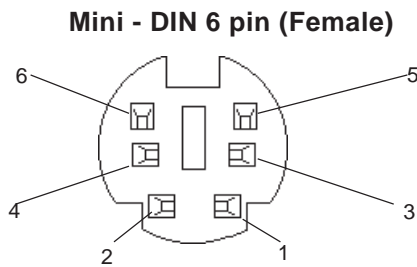
Mini - DIN 6 pin (Female)

(Both front and side)



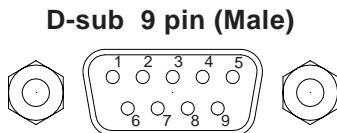
Pin No.	Signal Name
1	KEY DATA
2	NC
3	GND
4	+5V
5	KEY CLK
6	NC
SHIELD	GND

2.3.3 Mouse Interface(MOUSE)



Pin No.	Signal Name
1	Mouse DATA
2	NC
3	GND
4	+5V
5	Mouse CLK
6	NC
SHIELD	GND

2.3.4 RS-232C Interface (COM1/COM2/COM3)



Screw Size: (4-40UNC): Inch Type

Pin No.	Signal Name	Pin No.	Signal Name
1	CD	6	DSR
2	RXD	7	RTS
3	TXD	8	CTS
4	DTR	9	RI/+5V
5	GND		



The GND terminal is the signal ground. Be sure to connect it with the cable's opposite side SG terminal.

No. 9 pin (RI/+5V) is used by COM2 and COM3 only. If COM1 is used, the pin becomes RI. The changeover from RI to +5V is set via the PL side face slide switch.

Reference 2.4 PL Part Names and Features

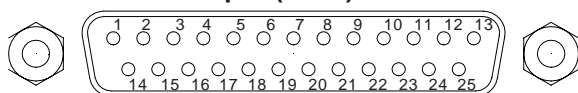


Be sure to confirm what settings will be used by the other device and set the dip switches accordingly. Failure to do so can result in a unit malfunction or damage.

Whenever changing the PL dip switches, be sure to first turn the PL's power supply OFF. Failure to do so can cause a PL malfunction.

2.3.5 RAS Interface(RAS)

D-Sub 25 pin (Male)



Screw Size: (4-40UNC): Inch Type

Pin No.	Signal Name	Pin No.	Signal Name
1	GND	14	GND
2	+5V (max. 100mA)	15	+5V
3	+12V (max.100mA)	16	NC
4	NC	17	NC
5	RESET INPUT (+)	18	NC
6	DIN 0 (+)	19	NC
7	DOUT (-)	20	NC
8	DOUT (+)	21	LAMP OUT (-)
9	ALARM OUT (-)	22	LAMP OUT (+)
10	ALARM OUT (+)	23	NC
11	RESET INPUT (-)	24	DIN1 (-)
12	DIN 0 (-)	25	NC
13	DIN 1 (+)		



Be sure to use only the rated voltage level when using the No.2[+5V] and No.3 [+12V] for external power output. Failure to do so can lead to a unit malfunction or accident.

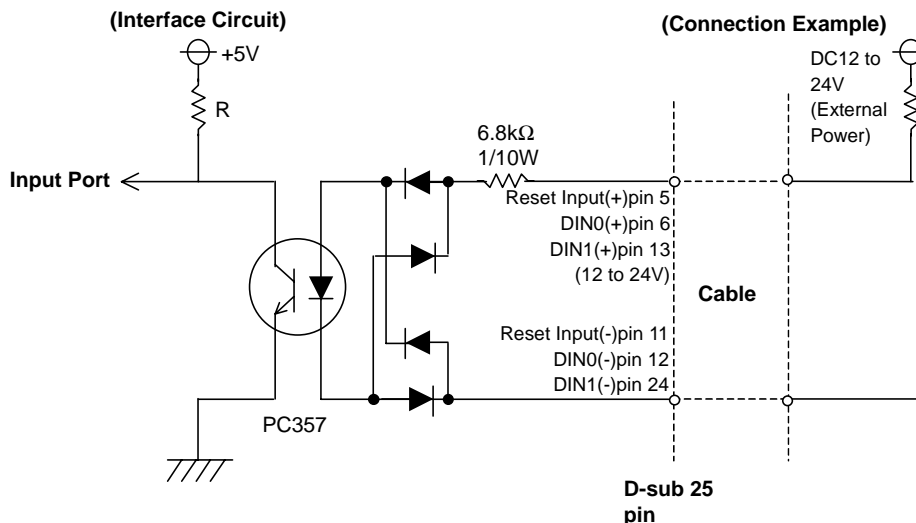


For detailed RAS Feature information,

Reference Appendix 2 RAS Feature.

External Input Signal (Used for both DIN and Remote Set Input)

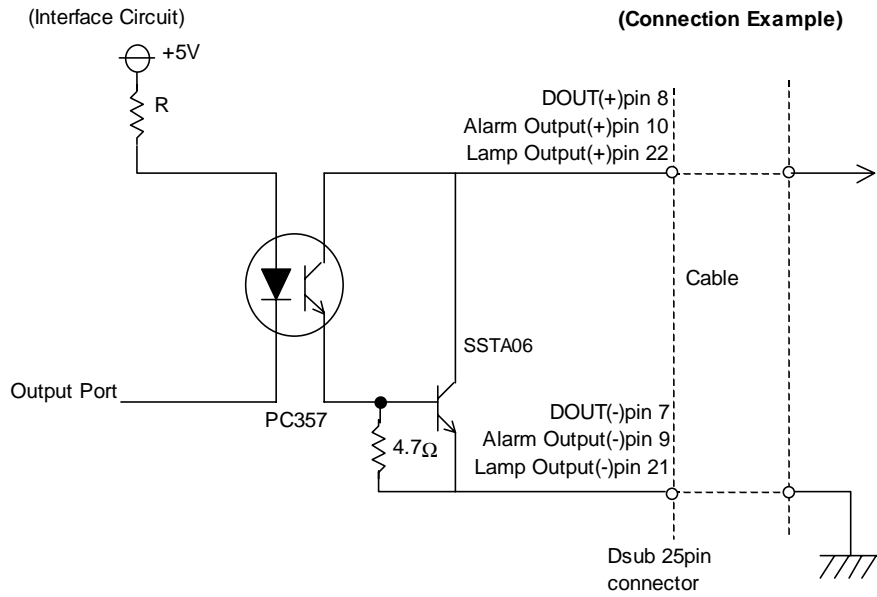
- External Power : DC12V to DC24V
- Input Hold : Hold Diode
- Isolation : Used (Photo isolation)



The power supply used for sink/source type input can use either polar or non-polar connection.

■ External Output Signal (DOUT, Alarm Output, Lamp Output Port)

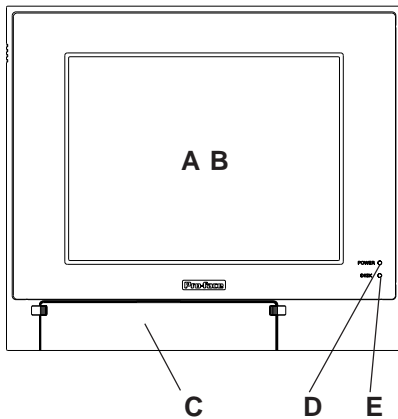
- Output Specification : DC24V 100mA (MAX)
- Isolation : Used (Photo isolation)



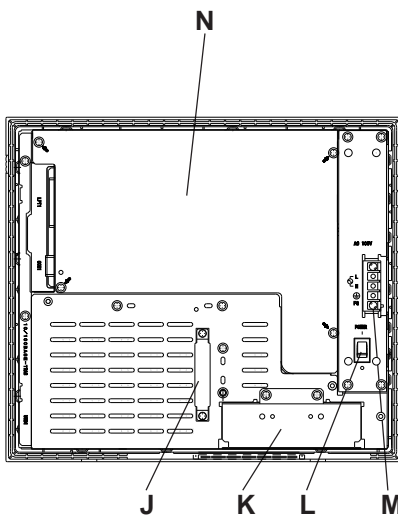
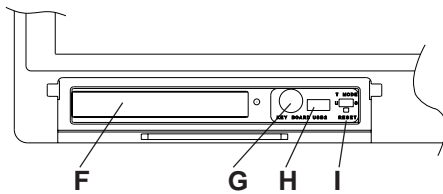
■ External Power Output

- +5V :100mA (MAX)
- +12V :100mA (MAX)

2.4 PL Part Names and Features



<Inside the front maintenance cover>



A: Display

Display output area. The built-in VGA controller supports PC compatible architecture.

B: Touch Panel

This high-resolution analog touch panel allows you to configure a keyboard-less system.

C: Front Maintenance Cover

Open this cover to access the Keyboard I/F, Reset Switch and connect the optional FDD unit.

D: Power Lamp LED (POWER)

The status of the lamp changes according to the alarm type detected by the RAS feature.

Reference 2.3.5 RAS Interface

E: Hard Disk Access LED (DISK)

The LED lights during accesses to the hard disk/flash file disk.

F: FDD Front Face Blank Panel

Remove this cover to install the optional FDD Unit.

G: Keyboard Connector

A PS/2 compatible keyboard is connected here.

H: USB Connector (USB2)

To use the USB connector, you must install Windows 98 (SR2).

I: Hardware Reset Switch (RESET)

J: IDE I/F Cover

To connect the optional CD-ROM drive unit (PL-DK200), the Mirror Disk Unit(PL-MD200-HU01), or RS-232C/RS-485 Adapter remove this cover and use this connector.

K: FDD Rear Face Blank Panel

Remove this cover to install the optional FDD Unit.

L: Power Switch (POWER)

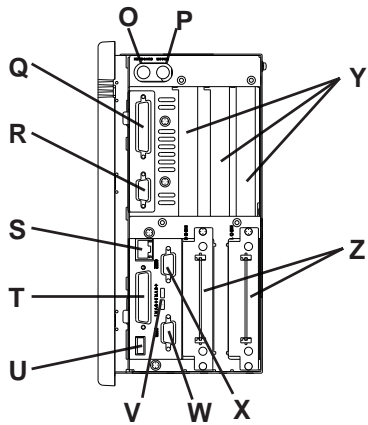
Turns the PL's power ON or OFF.

M: Power Terminals

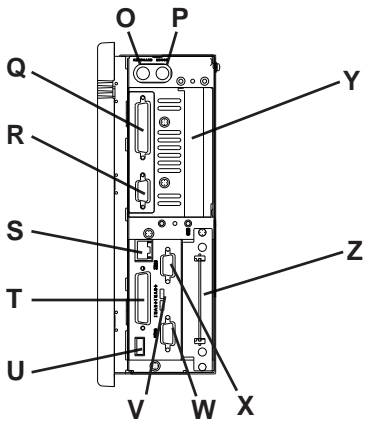
The PL's AC100V/DC24V power cord terminals are connected here.

N: Rear Maintenance Cover

Remove this cover to install the optional DIM module, or an expansion board.



PL-5900T



PL-5901T

O: Keyboard Connector (KEYBOARD)

A PS/2 compatible keyboard can be connected here.

P: Mouse Connector (MOUSE)

A PS/2 compatible mouse can be connected here.

Q: Printer Connector (LPT1)

Centronics standard interface (D-sub 25 pin female connector), which connects a parallel device, such as a printer (supports ECP/EPP).

R: RS-232C Connector (COM1).

S: Ethernet Connector (10/100BASE-TX)

IEEE802.3 standard Ethernet interface. 10BASE-T/100BASE-TX auto changeover.

T: RAS Connector (RAS)

Interface for DIN, DOUT, Watchdog, and Remote Reset. (D-sub 25 pin male connector)

U: USB Connector (USB1)

To use the USB connector, you must install Windows 98 (SR2).

V: Signal Changeover Slide Switch (+5VRI)

This switch changes the COM2/COM 9-pin current from RI to 5V.

W: RS-232C Connector (COM2)

RI/+5V Changeover

X: RS-232C Connector (COM3)

RI/+5V Changeover

Y: Expansion Slot(s)

Z: HDD/FFD/CF Card Unit Expansion Slot

Houses an additional HDD unit, FFD unit or CF Card unit.



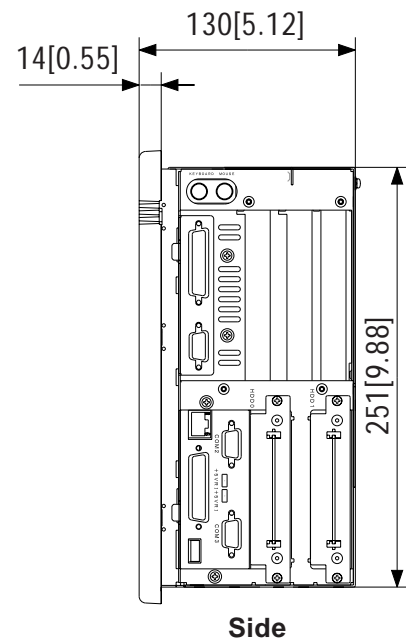
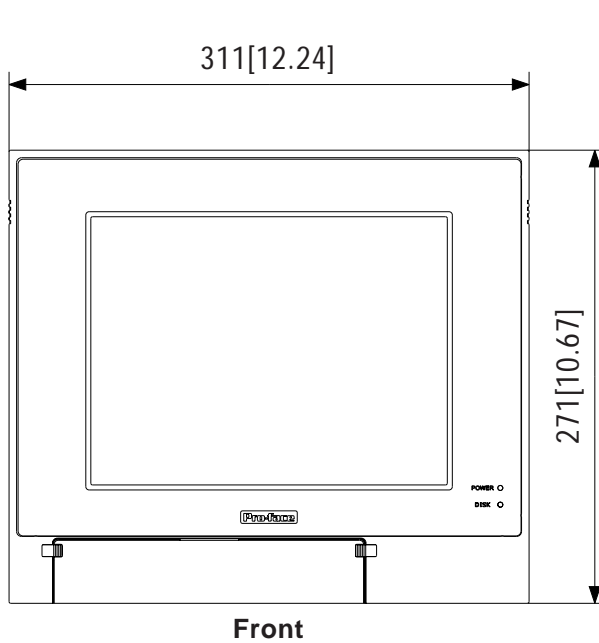
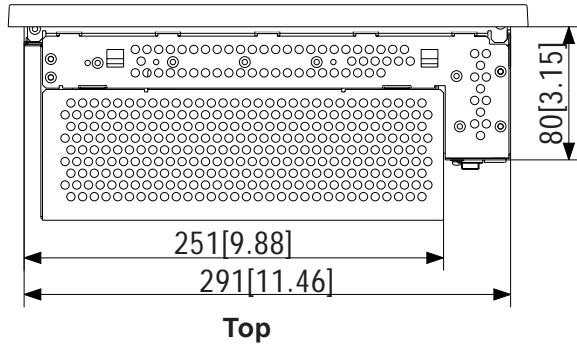
- **When attaching peripheral units to the PL, be sure the PL's power cord is disconnected from the main power supply.**
- **To avoid an electrical shock, be sure to disconnect the PL's power cord from the power supply before connecting the cord's power terminals or any peripheral devices to the PL.**

Reference 4.3.1 Connecting the Power Cord

2.5 External Dimensions

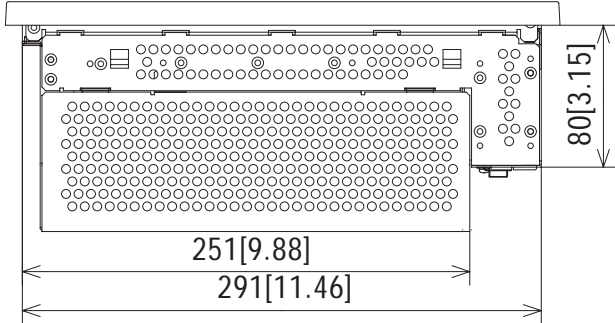
2.5.1 PL-5900T External Dimensions

(Unit: mm [in.] - excluding projections)

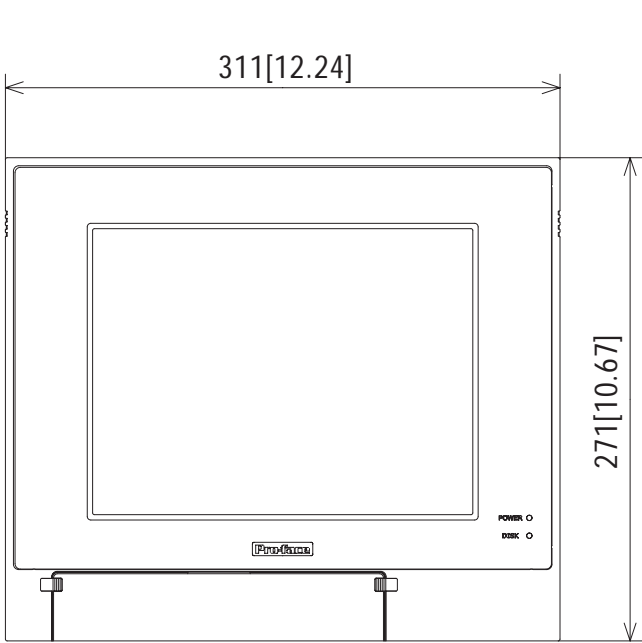


2.5.2 PL-5900T with PL-FD500 External Dimensions

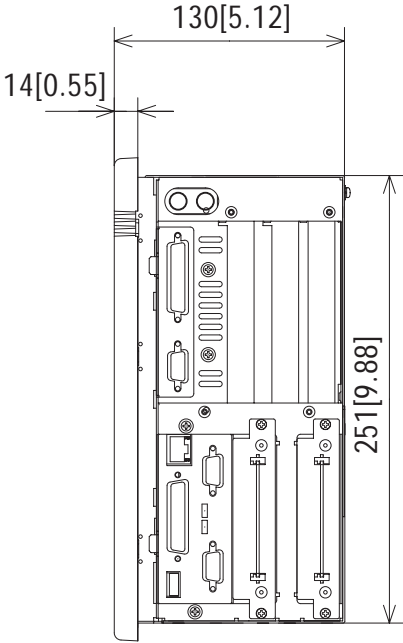
(Unit: mm [in.] - excluding projections)



Top



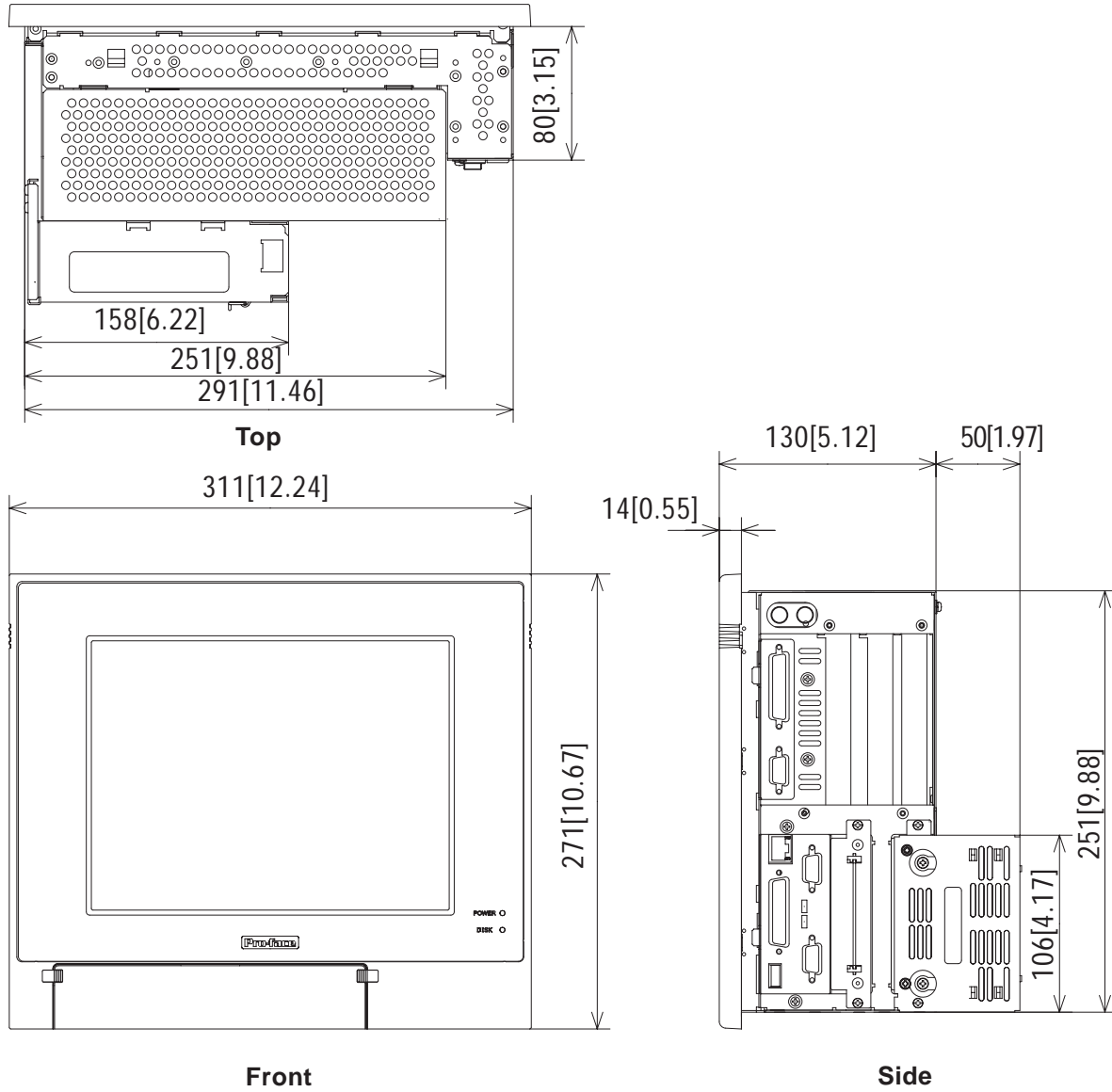
Front



Side

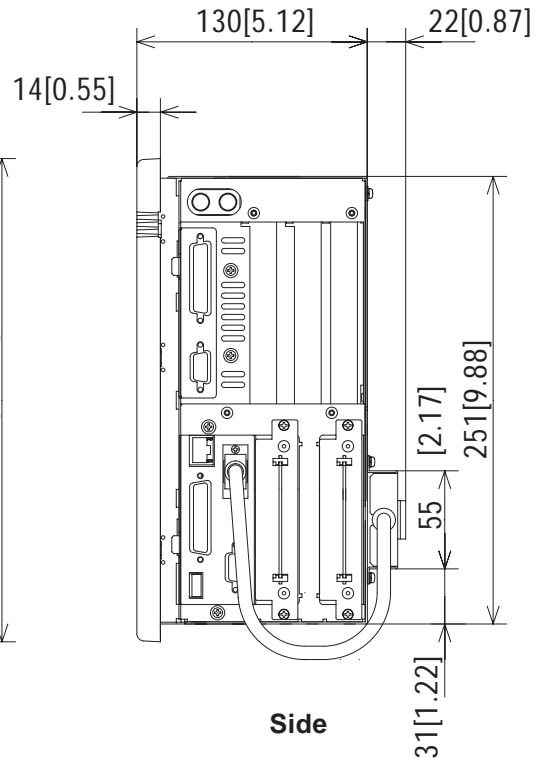
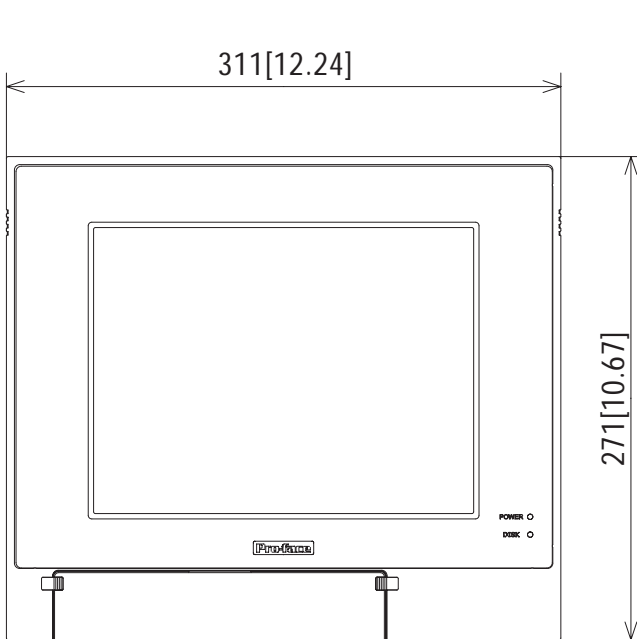
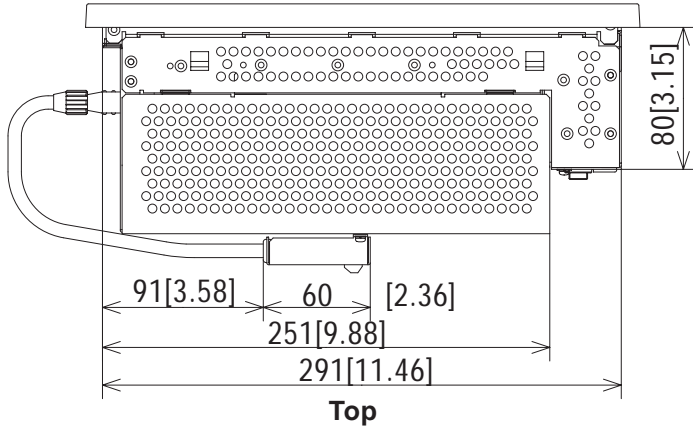
2.5.3 PL-5900T with Mirror Disk Unit External Dimensions

(Unit: mm [in.] - excluding projections)



2.5.4 PL-5900T with PL-RC500 External Dimensions

(Unit: mm [in.] - excluding projections)

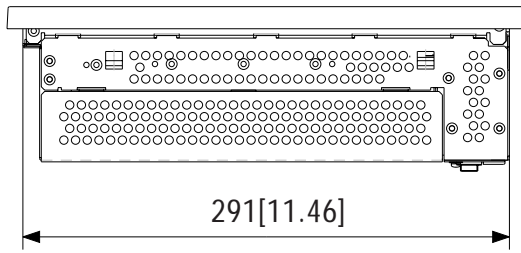


Front

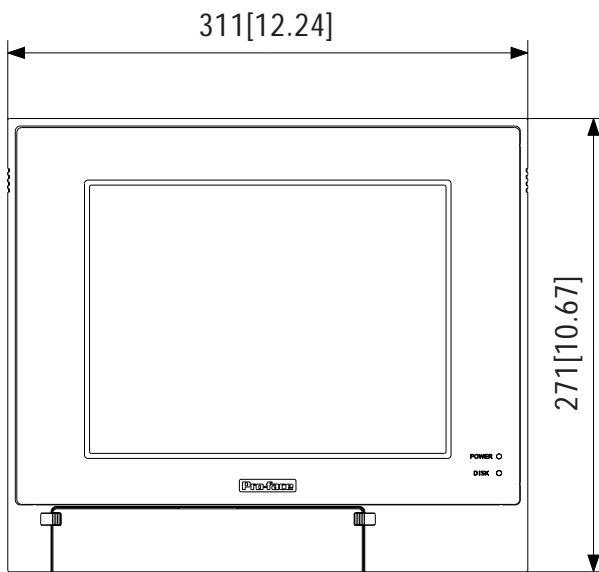
Side

2.5.5 PL-5901T External Dimensions

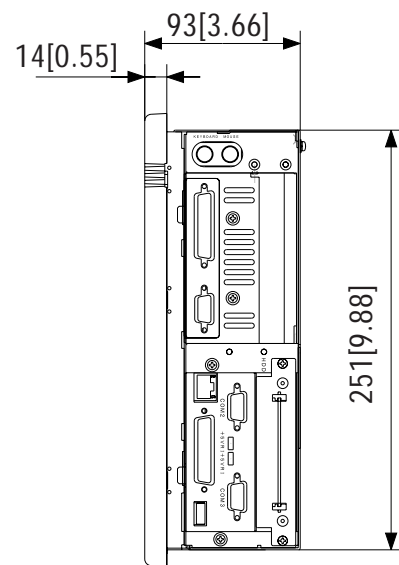
(Unit: mm [in.] - excluding projections)



Top



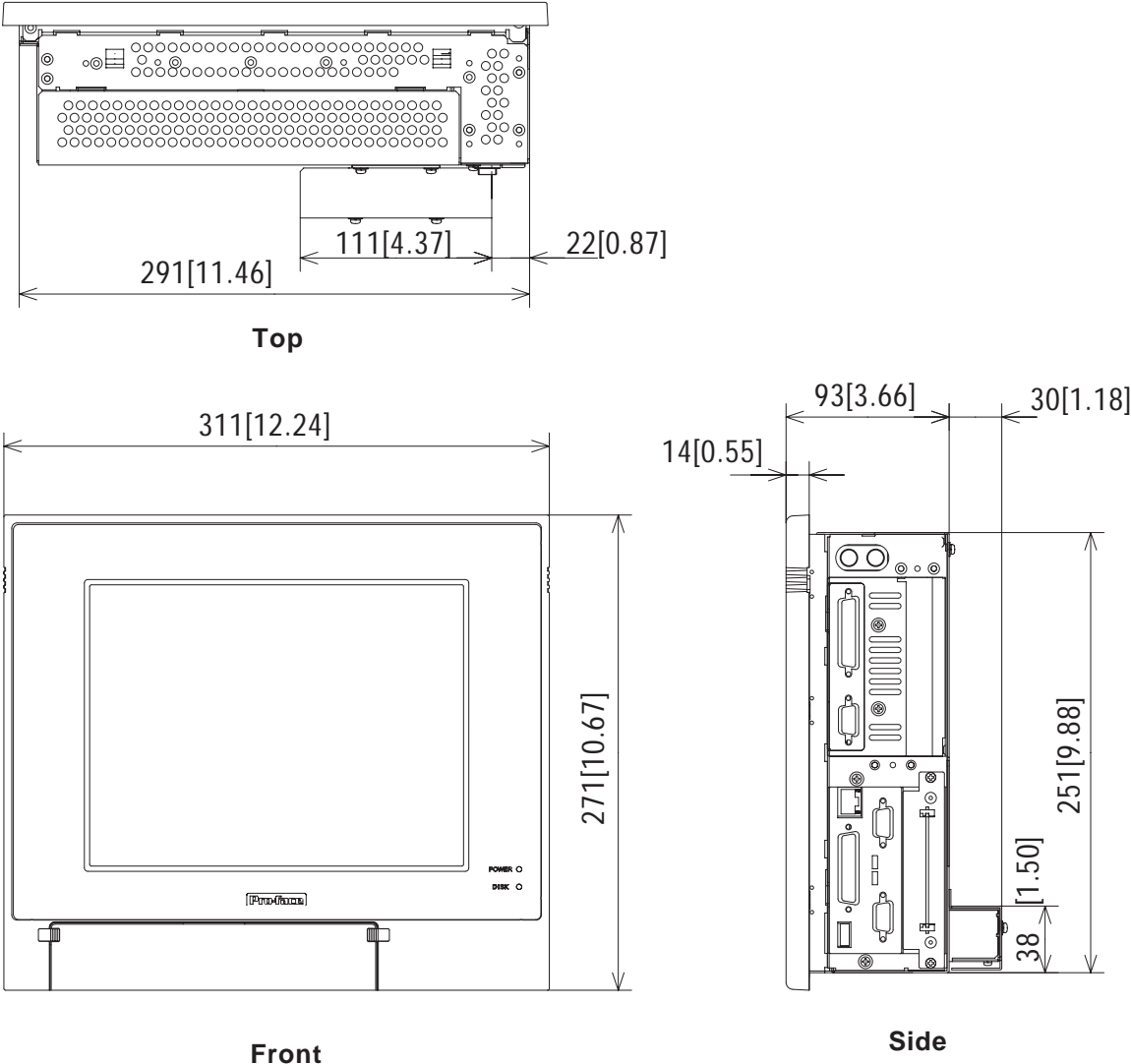
Front



Side

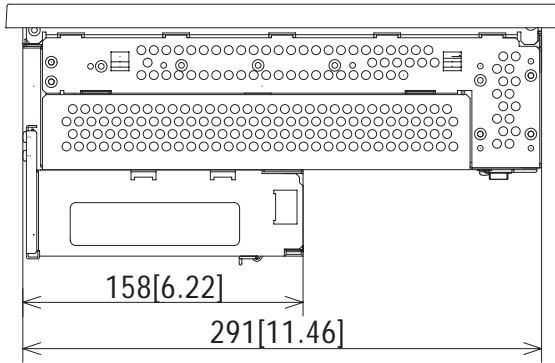
2.5.6 PL-5901T with PL-FD500 External Dimensions

(Unit: mm [in.] - excluding projections)

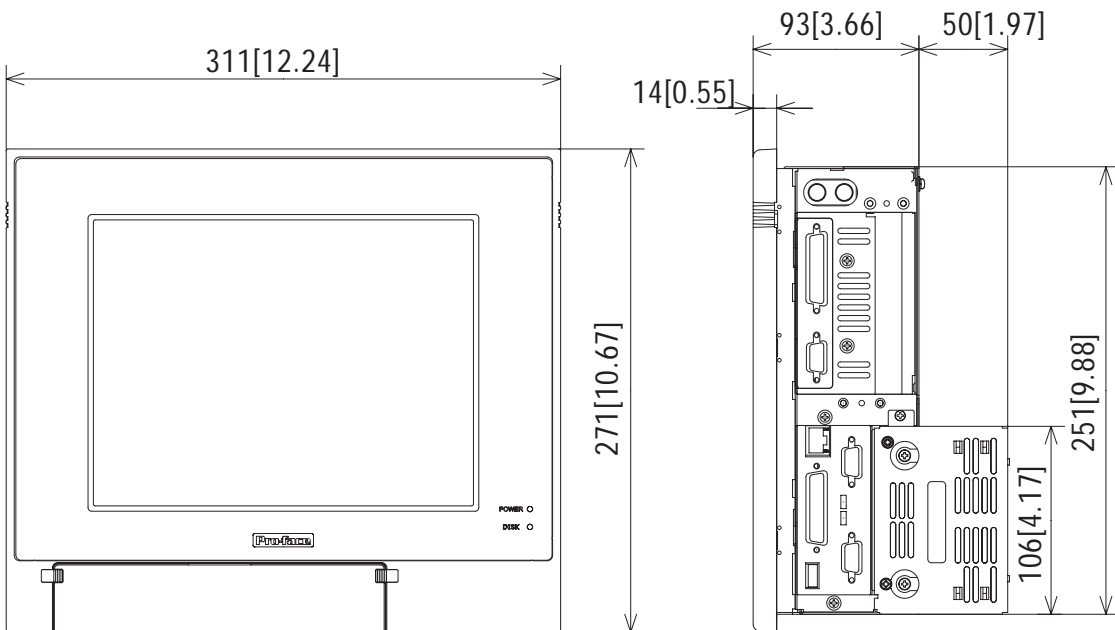


2.5.7 PL-5901T with Mirror Disk Unit External Dimensions

(Unit: mm [in.] - excluding projections)



Top

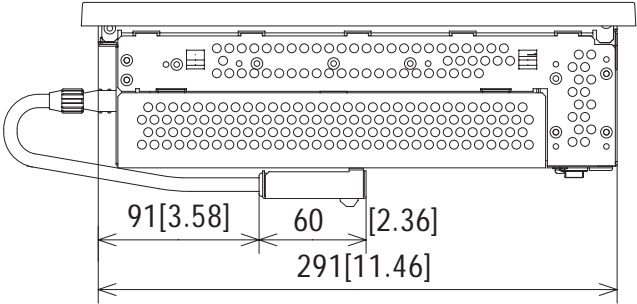


Front

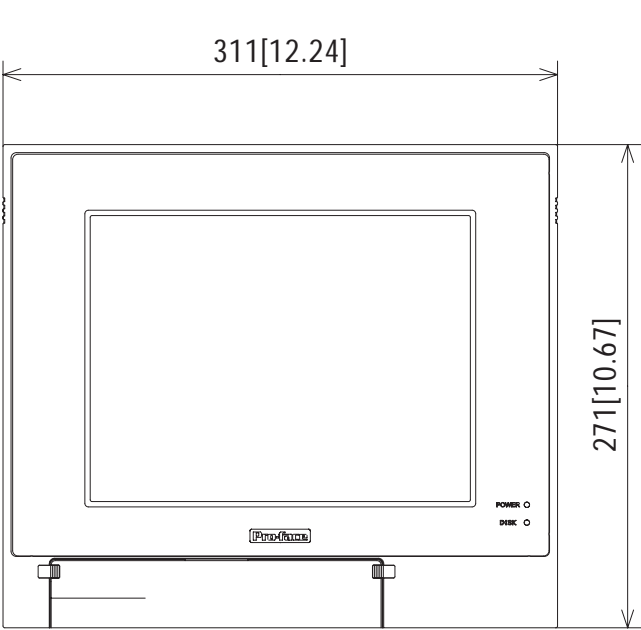
Side

2.5.8 PL-5901T with PL-RC500 External Dimensions

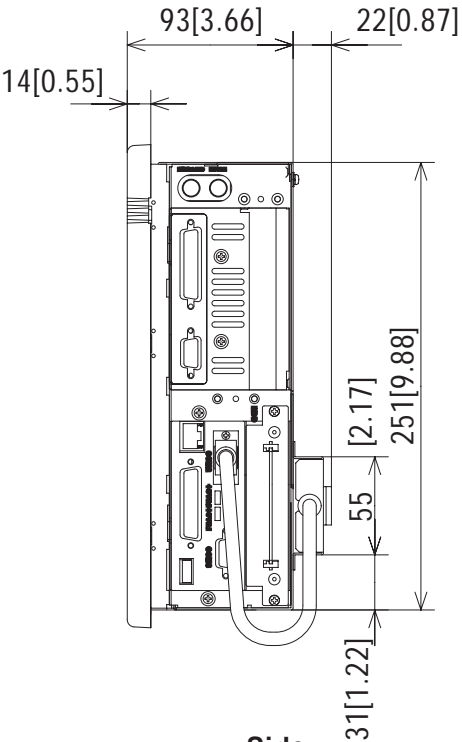
(Unit: mm [in.] - excluding projections)



Top



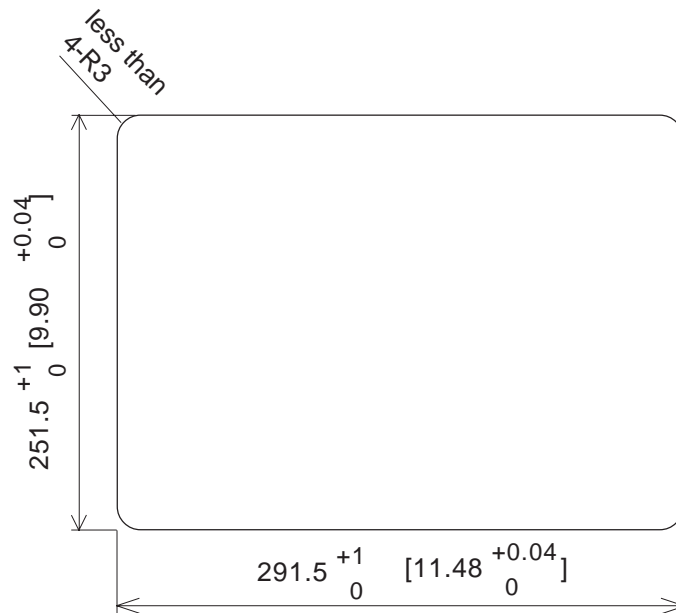
Front



Side

2.5.9 Panel Cut Dimensions

(Unit: mm [in.])



- **Be sure the thickness of the panel is from 1.6 to 10 mm.**
- **All panel surfaces used should be strengthened. Especially, if high levels of vibration are expected and the PL's installation surface (i.e. an operation panel's door, etc.) can move (i.e. open or close) due consideration should be given to the PL's weight.**
- **To insure that the PL's moisture resistance is maintained, be sure to install the PL into a panel that is flat and free of scratches or dents.**
- **Be sure all installation tolerances are maintained to prevent the unit from falling out of its installation panel.**

Chapter

3 Installing Optional Units and Expansion Boards

A wide variety of optional units and expansion boards made by Digital can be installed in the PL, as well as a number of commercially available PCI-bus or ISA-bus compatible boards. This chapter describes how to install these products in the PL.

3.1 Installation

The following explanation pages describe the installation procedures for the PL's DIM module(PL-EM500/PL-EM128), FDD unit(PL-FD500), HDD unit(PL-HD220), expansion boards and CD-ROM drive unit(PL-DK200).

Reference For information about the installation of other option units, please refer to those unit's individual *[Installation Guide]*.

WARNING

To prevent an electric shock or PL damage, confirm that the PL unit's power has been turned OFF before installing any optional units or expansion boards.



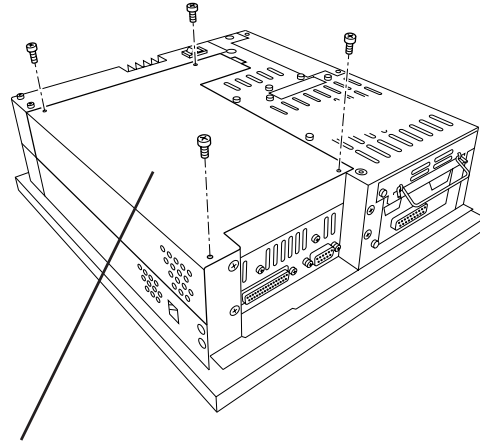
- *Use a screwdriver to loosen or tighten the screws. Be careful not to tighten screws too tightly, since it may damage the equipment.*
- *Be careful when removing or inserting any screws that they do not fall inside the PL.*

3.1.1 Removing the Rear Maintenance Cover



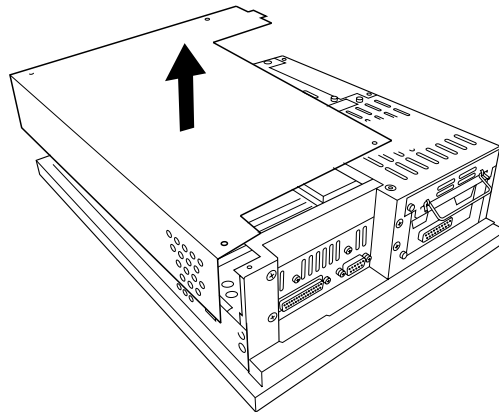
Be sure to handle the rear maintenance cover carefully, since it is made of aluminum and is easily bent.

- 1) Unscrew the four attachment screws used to hold the rear maintenance cover and half cover.



**Rear Maintenance
Cover**

- 2) Remove the rear maintenance cover by lifting the cover in the direction shown.



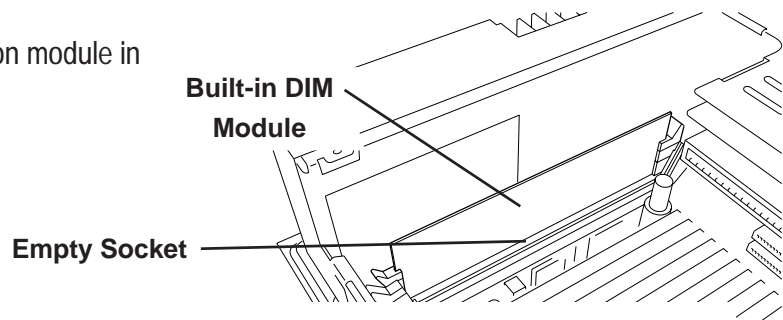
3.1.2 Installing the DIM Module (PL-EM500/PL-EM128)



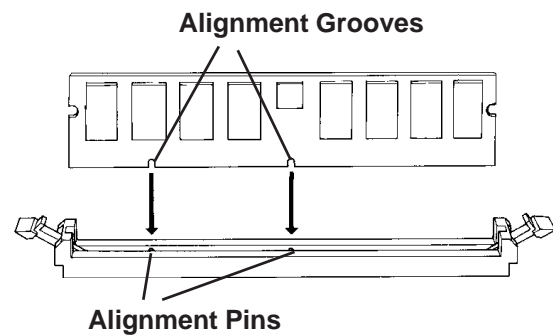
- **Since DIM module sockets are fragile and break easily, be sure to install the DIM module carefully.**
- **Do not change the factory installed DIM module's socket position.**

The PL comes with a single , 64MB DIM module pre-installed. There is one more empty socket that can be used to expand your PL unit's memory. Use the following procedure to install a second DIM module in that socket.

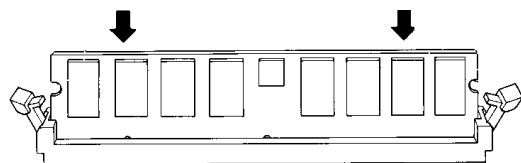
- 1) Install the DIM expansion module in the empty socket.



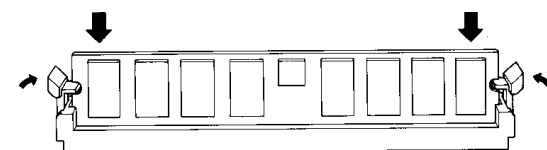
- 2) Position the Alignment Grooves so that they fit the Alignment Pins.



- 3) Insert the DIM module into the DIM module socket.

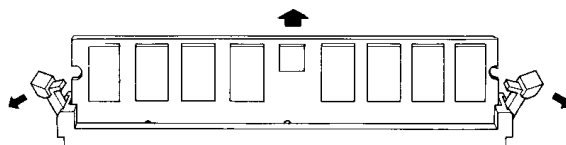


- 4) Push the DIM module down until the ejector tabs lock.
- 5) Replace the rear maintenance cover and the half cover and secure them in place with their attachment screws.



◆ To Remove the DIM Module

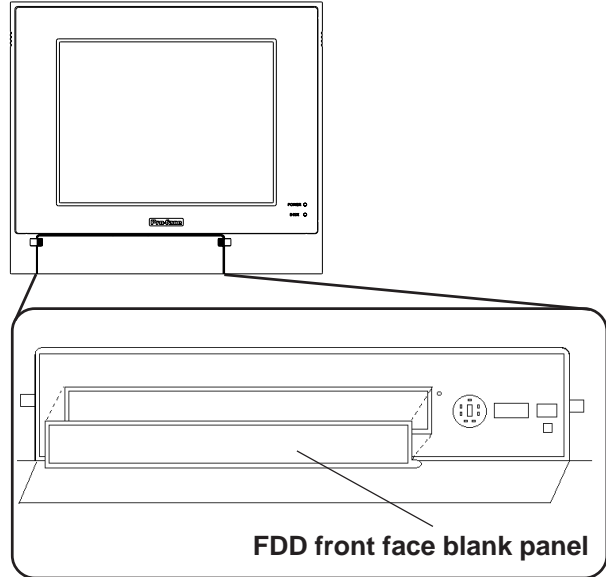
To remove a module, press down on the socket's ejector tabs to release the module.



3.1.3 Installing the FDD Unit (PL-FD500)

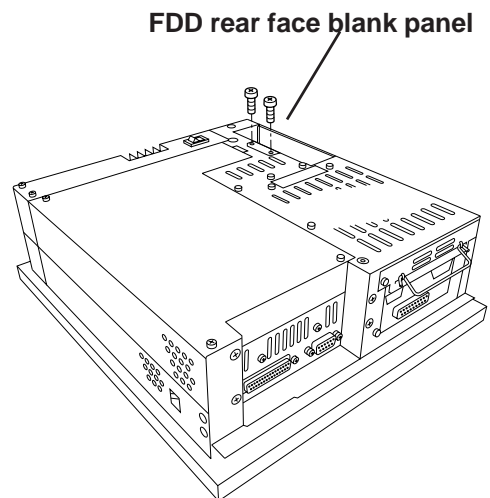
The attachment procedures for the PL-5900T and the PL-5901T are different. The following steps, up to 3), are the same. After that, refer to your unit's specific instructions.

- 1) Open the front maintenance cover and remove the FDD front face blank panel.



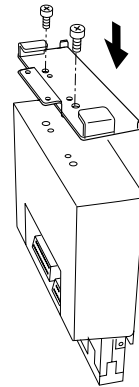
- 2) Close the front maintenance cover.

- 3) Remove the two attachment screws from the FDD rear face blank panel and remove the cover.

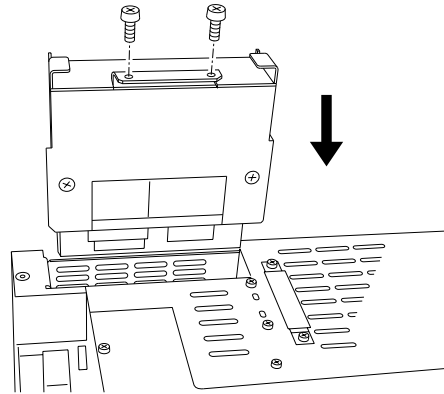


■ PL-5900T (3-Slot model)

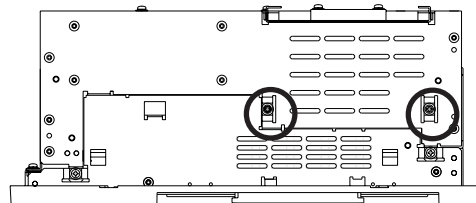
- 4) Attach the FDD rear face blank panel removed in step 3 to the FDD unit and secure it with the two attachment screws.
(Do not use the bracket that comes with the FDD Unit.)



- 5) Insert the FDD unit so that the PL and FDD unit connectors are securely connected.

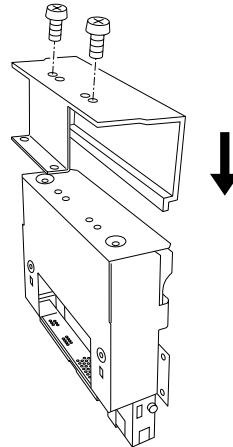


- 6) Secure the FDD unit to the PL using all four attachment screws. (Two FDD unit and two PL bottom face screws)

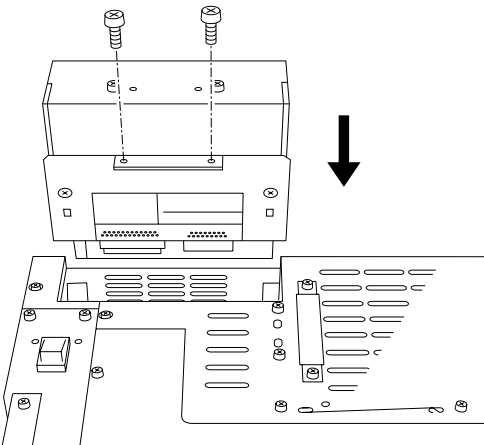


■ PL-5901T (1-Slot model)

- 4) Attach the bracket that comes with the FDD unit and secure it with the two attachment screws. (Do not use the FDD rear blank panel removed in step 3.)

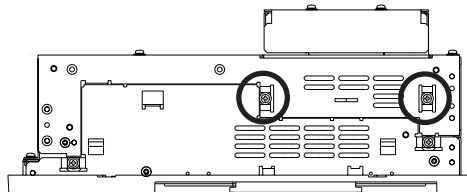


- 5) Insert the FDD unit so that the PL and FDD unit connectors are securely connected.



- 6) Secure the FDD unit to the PL using all four attachment screws. (Two FDD unit and two PL bottom face screws)

Note: As this drawing shows, even when the FDD unit is inserted completely, it will protrude slightly from the back of the PL.



3.1.4 Removing/ Installing the HDD Unit (PL-HD220)



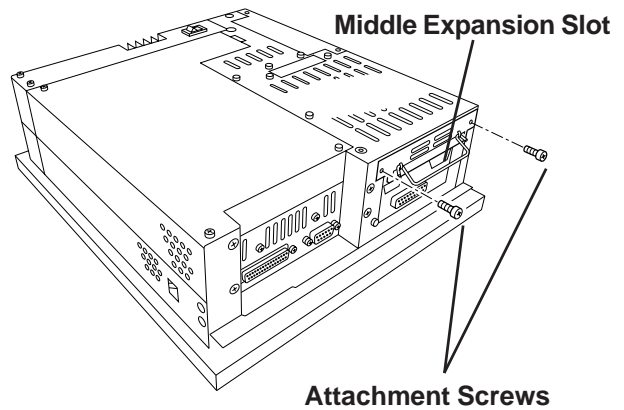
- The following insertion/removal procedure is the same for the FFD unit (PL-FF210) and CF Card unit (PL-CF200).
- Certain usage limitations apply to the HDD, FFD, CF Card Mirror Disk and CD-ROM Drive unit.

Reference *1.3 Optional Items*

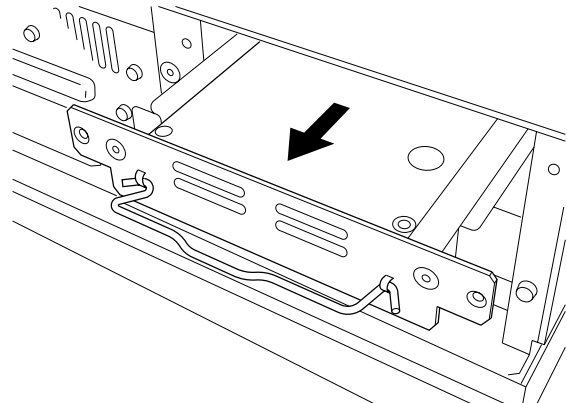


Since the HDD unit is a precision instrument, be sure not to subject it to excessive vibration or sudden shocks.

- 1) Remove the two attachment screws from the Expansion Slot Cover. (Middle cover on the PL-5900)



- 2) Grasp the HDD unit's handle and pull the unit slowly out of the PL. Be sure you do not damage the unit.



- 3) Insert the new HDD unit into the PL's guideways and push it in until its rear connector is securely connected.

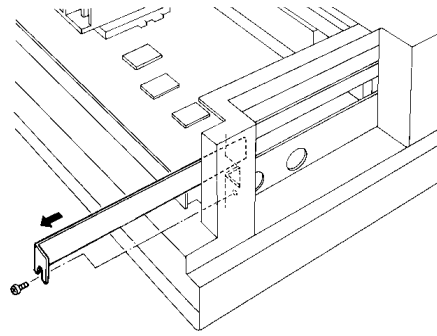
- 4) Secure the unit in place with its two attachment screws.

3.1.5 Installing an Expansion Board

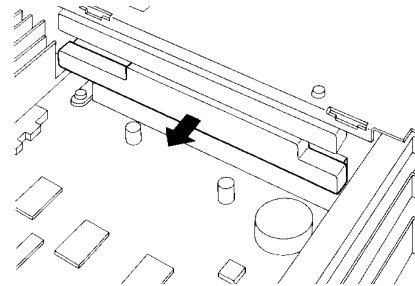
- 1) Unscrew the desired expansion slot's cover attachment screw, and remove the cover.

Reference 3.1.1, *Removing the Rear Maintenance Cover*

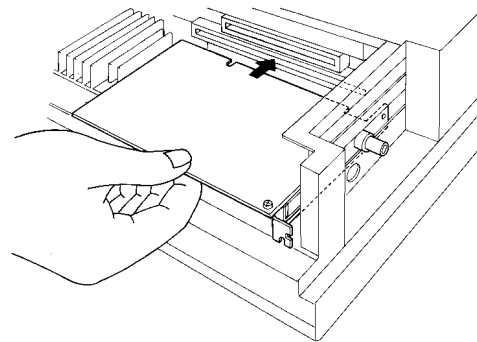
Unscrew the Blank Panel's attachment screw to remove the Blank Panel.



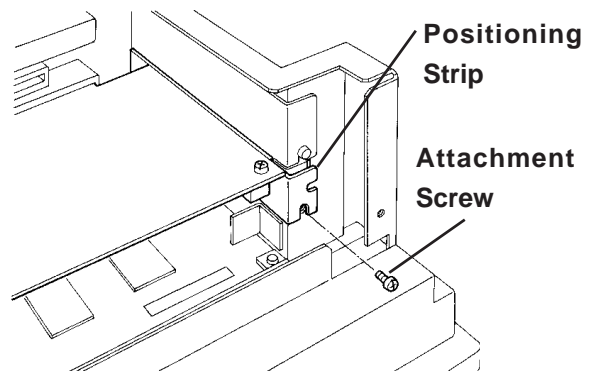
- 2) Remove the expansion slot's duster cover.



- 3) Insert the expansion board into the expansion slot.



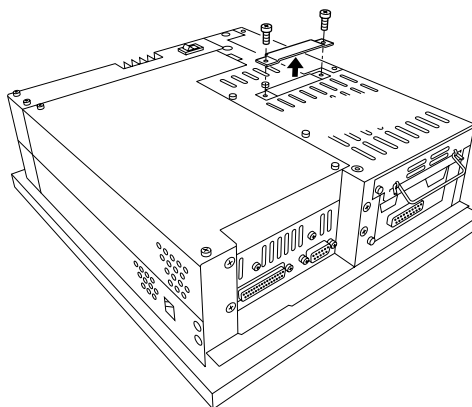
- 4) Secure the expansion board's metal positioning strip in place with its attachment screw.



- 5) Last, replace the rear maintenance cover and half cover and secure them in place with their attachment screws.

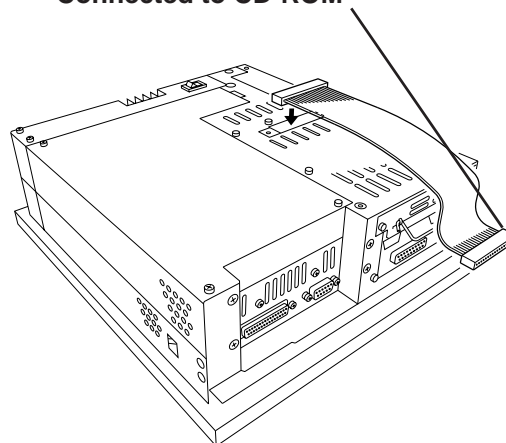
3.1.6 Connecting the CD-ROM Drive Unit (PL-DK200)

- 1) Unscrew the two IDE I/F cover attachment screws, and remove the cover.
- 2) Connect the CD-ROM unit cable to the PL's IDF I/F connector.



Be sure that the cable is securely connected before turning ON the PL's power switch.

Connected to CD-ROM



Memo

Chapter

4 Installation and Wiring

4-1 Installation Cautions

4-2 Installing the PL

4-3 Wiring the PL

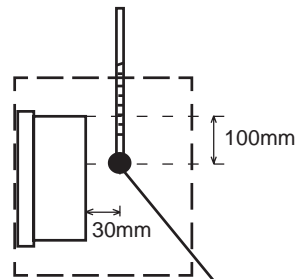
This chapter explains how to install and wire the PL-5900 series units, as well as the cautions required both before and during installation.

4.1 Installation Cautions

1) Temperature Cautions

The PL should be installed in a vertical position, and forced air cooling should be used, instead of natural air circulation.

Also, be sure to confirm that the area near the PL will be within the allowable temperature range by placing a temperature sensor in the location shown in the left-side drawing. If this area's temperature exceeds the allowed limit, a machine breakdown can occur.

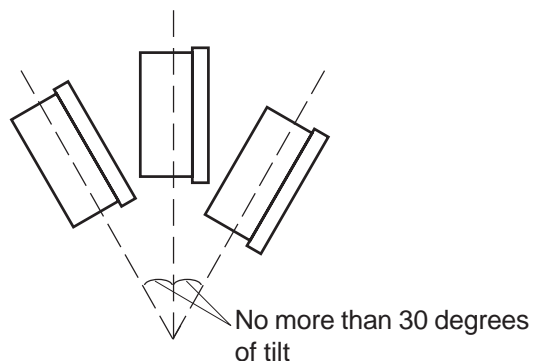


Temperature Sensor
0°C to 45°C (without HDD unit)
5°C to 45°C (with HDD unit)

2) Installation Cautions

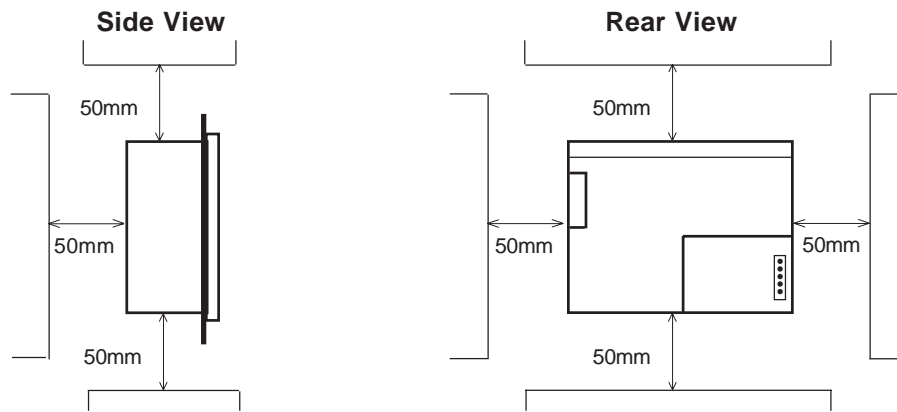
Be sure to install the panel in an upright (vertical) position.

Also, be sure that the panel's viewing angle is tilted no more than 30 degrees from parallel to the operator (i.e. directly in front).



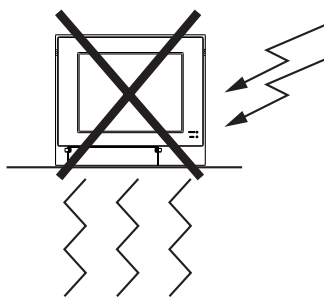
■ **Installation Location**

- Avoid placing the PL next to other devices that might cause overheating.
- Keep the PL away from arc-generating devices such as magnetic switches and non-fuse breakers.
- Avoid using the PL in environments where corrosive gases are present.
- To ensure the reliability, operability and ventilation of the PL, be sure to install it in locations that are more than 50mm away from adjacent structures or equipment. Also, consider the need for installing or removing expansion boards, or connectors when designing and installing your PL.



■ **Vibration and Shocks**

If the PL is moved when its enclosure doors are open, or while it is installed in a rack equipped with caster wheels, the hard disk can receive excessive vibration or jolting. Be especially careful at this time.



PL Configuration	Can Withstand
HDD	4.9m/s ²
FDD	9.8m/s ²
No drives	19.6m/s ²



- **The Hard Disk Drive is precision equipment and should not be moved or jolted . Especially when the PL is turned ON, even changing the PL's direction while it is on a table, or repositioning the unit should not be performed, since it can lead to a hard disk crash or malfunction.**
- **When using a fan to cool the PL unit, be sure that the fan does not point directly at any of the PL's disk drive units, since it can lead to a hard disk crash or malfunction.**

4.2 Installing the PL

4.2.1 Installation Procedures

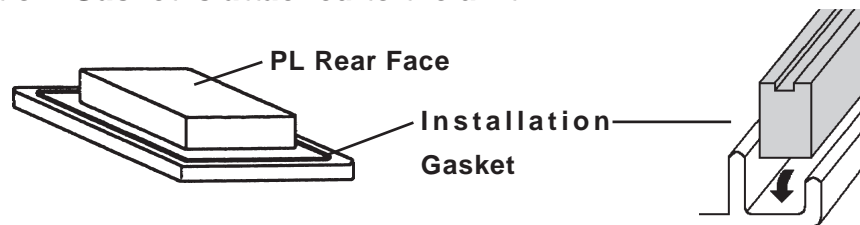
Follow the steps given below when installing the PL.

■ Attaching the Installation Gasket

Even if the your PL's Installation Gasket is not needed to prevent water from entering the unit, the gasket also acts as a vibration absorber and should always be attached. To install the gasket, place the PL face down on a soft surface and attach the gasket to the rear side of the display face, in the plastic bezel's groove (see picture below). Be sure the grooved face of the gasket is vertical.



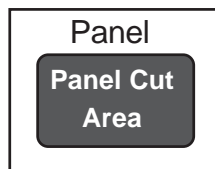
Before mounting the PL into a cabinet or panel, check that the Installation Gasket is attached to the unit.



■ Create a Panel Cut

Create a panel cut for the PL, like that pictured here. Two additional items, the installation gasket and the installation fasteners are also required when installing the PL.

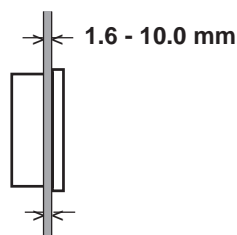
Reference 2.5 PL Dimensions



- To obtain the maximum degree of moisture resistance, be sure to attach the PL to a smooth, flat surface.
- The panel itself can be from 1.6 to 10.0 mm thick.



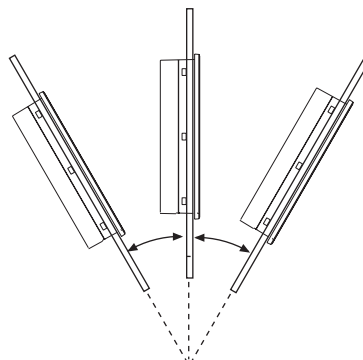
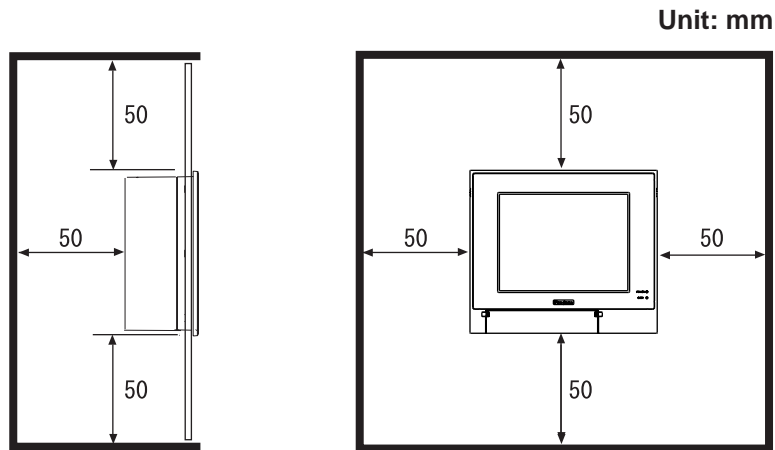
Strengthening may be required for the panel. Be sure to consider the weight of the PL when designing the panel.



Chapter 4 - Installation and Wiring



- To enhance the PL's maintainability, operability and ventilation, allow at least 50 mm clearance between the PL and any other objects. (The clearance must be large enough to allow you to insert or remove expansion boards and to attach connectors.)



less than 30° from vertical



- Avoid using the PL where the ambient temperature will exceed 45°C.
- Avoid placing the PL next to other devices that might cause overheating.
- Be sure that the panel's viewing angle is tilted no more than 30 degrees from parallel to the operator (i.e. operator is directly in front).
- Keep the PL away from arc-generating devices such as magnetic switches and non-fuse breakers.
- Avoid using the PL in environments where corrosive gases are present.

■ Installation

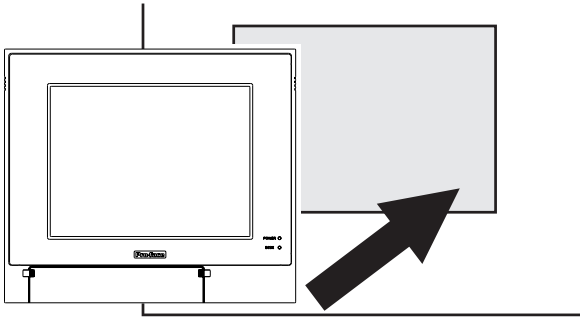
- 1) Insert the PL into the panel.



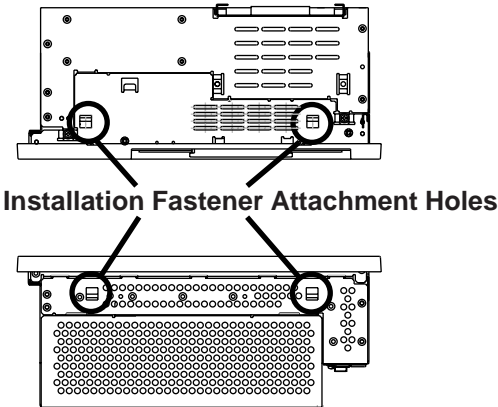
Be sure the panel cut's actual measurements are the same as those given here, otherwise the PL may slip or fall out of the panel.

Reference

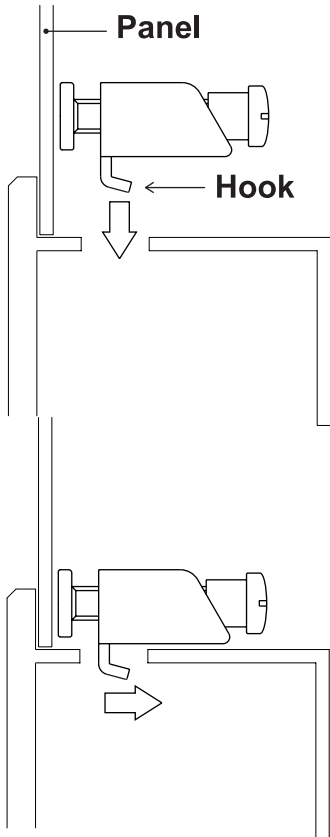
2.5.9 Panel Cut Dimensions



- 2) Insert the installation fastener hooks into the four installation fastener holes on PL's top and bottom sides.



- 3) Slide the installation fasteners to the rear face.

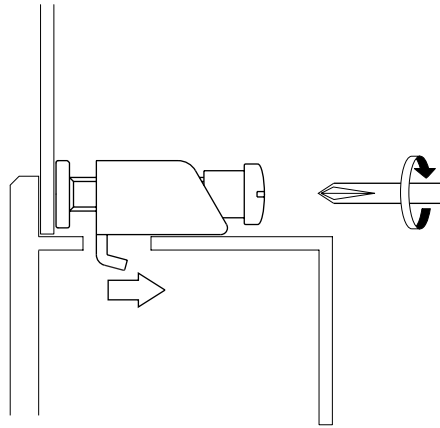


Chapter 4 - Installation and Wiring

- 4) Tighten the screws of the installation fasteners. Be sure to tighten the four screws in an even, crisscross pattern.



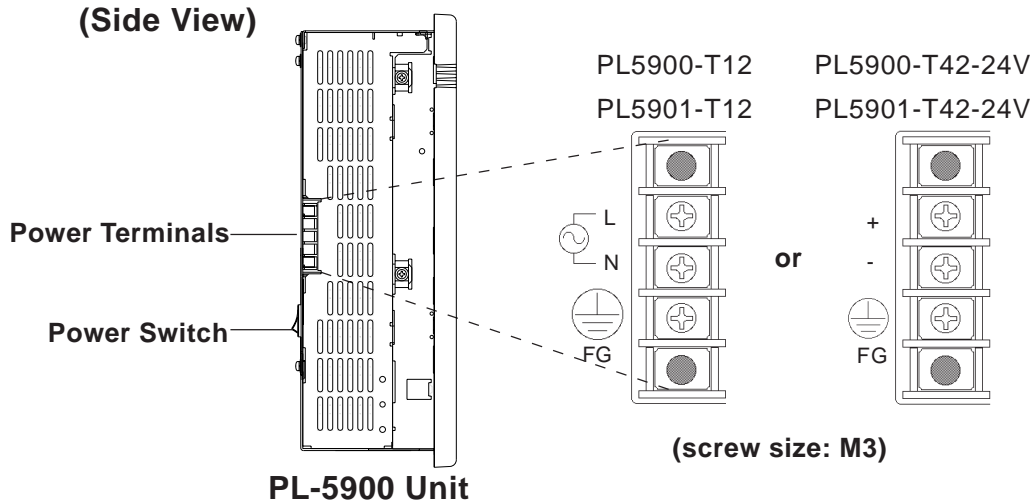
Do not use excessive force when tightening the main unit attachment screws. The torque required to render it waterproof is 0.5 N•m.



4.3 Wiring the PL

4.3.1 Connecting the Power Cord

Connect the PL's power cord to the PL's rear face power terminals.



PL5900-T12, PL5901-T12	
L	AC Input Live Line
N	AC Input Neutral Line
FG	Grounding Terminal connected to the PL chassis.

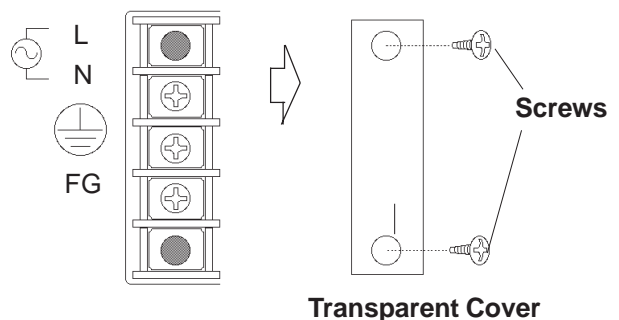
PL5900-T42-24V, PL5900-T42-24V	
+	Positive electrode
-	Negative electrode
FG	Grounding Terminal connected to the PL chassis.

Use the following steps when connecting the power cord to the PL's power terminals.

⚠ WARNINGS

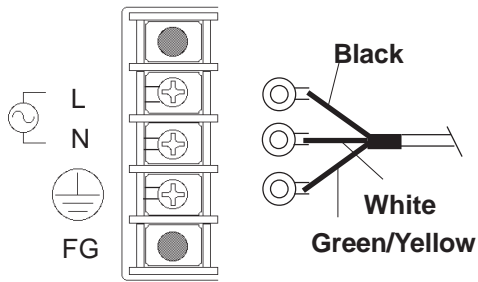
- To prevent an electric shock, be sure to turn the PL's power supply OFF before connecting the power cord terminals to the PL.
- To prevent fires, electrical hazards and equipment damage, be sure to use only the specified power supply voltage when operating the PL.

1) Confirm that the PL's power switch is turned OFF. Then, remove the power terminal's transparent plastic cover.



■ PL5900-T12, PL5901-T12

2) Loosen and remove the middle three screws from the terminal strip. Align the crimp terminals with each screw hole, and tighten the screws.

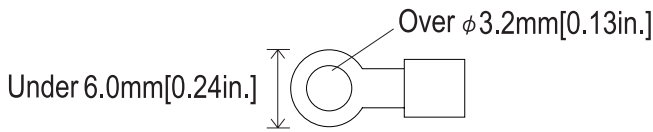


Crimp Terminal Types :



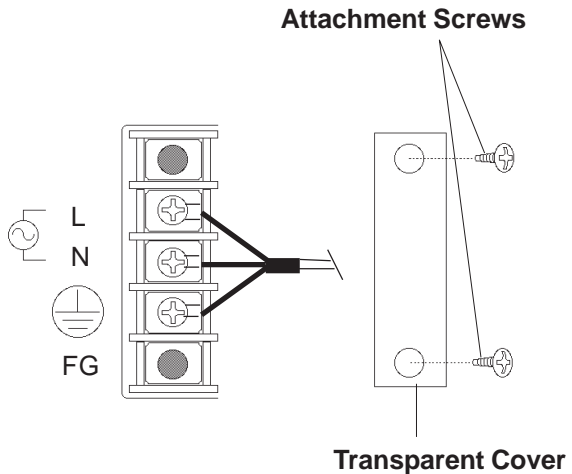
Note: V1.25-3, by J.S.T. or equivalent (JIS standard part number : RAV1.25-3)

- Crimp terminals must be the same as shown below.



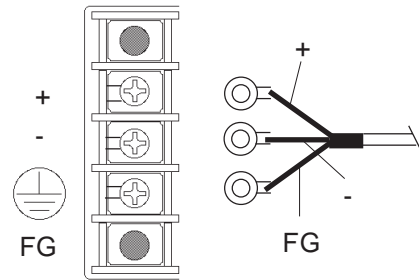
- *The colors used in these figures are for the cable which came with the PL.*
- *This power cable is designed only for AC100V/115V use. Any other power level should use its own specially designed cable.*

3) Reattach the terminal strip's transparent cover and secure it in place with its attachment screws.



■ PL5900-T42-24V, PL5901-T42-24V

- 2) Loosen and remove the middle three screws from the terminal strip. Align the crimp terminals with each screw hole, and tighten the screws.

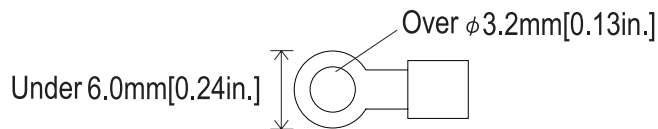


Crimp Terminal Types :

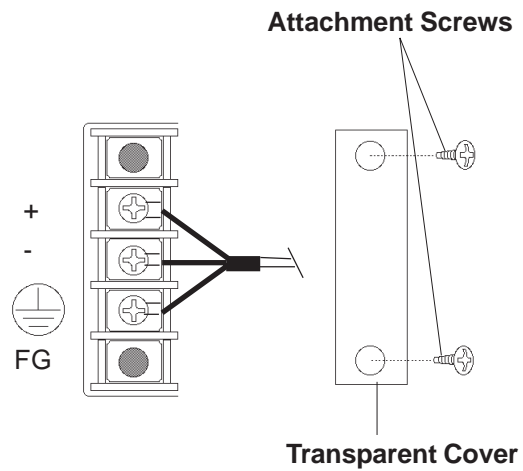


Note: V1.25-3, by J.S.T. or equivalent (JIS standard part number : RAV1.25-3)

- Crimp terminals must be the same as shown below.



- 3) Reattach the terminal strip's transparent cover and secure it in place with its attachment screws.



4.3.2 Power Supply Cautions

When connecting the PL unit's AC power terminals, please be aware of the following:

- If voltage fluctuations are expected to vary beyond the specified range, connect a constant voltage transformer.

Reference 2-1 General Specifications

- Use a low-noise power supply both between the lines and between the PL and its ground. If there is still excess noise, connect an insulating transformer (noise-prevention type).



Note:

Be sure any constant or insulating transformer used has a capacity of 200VA or more.

- Wire the power cords of the PL, I/O devices, and power supply devices separately.

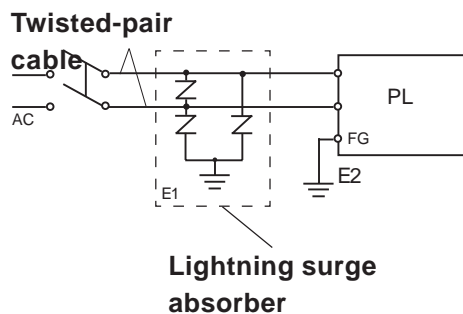
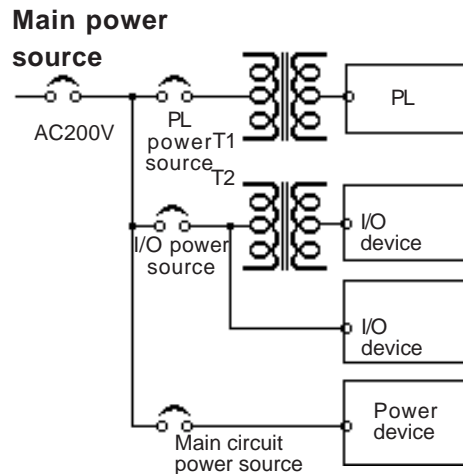
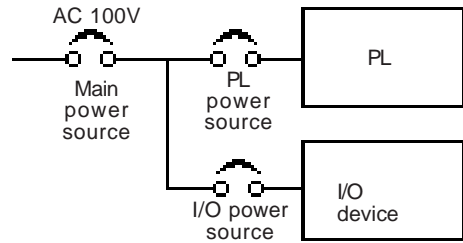
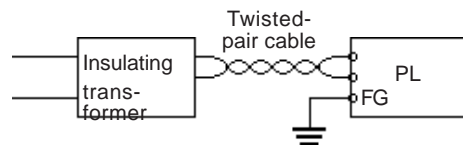
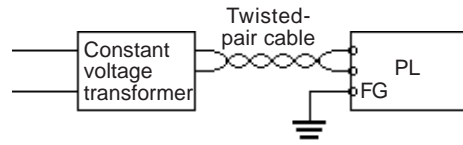
- Attaching a ferrite core to the power cord will improve noise immunity.
- Isolate the main circuit (high voltage, large current) line, I/O signal lines, and power cord, and do not bind or group them together.

- To prevent damage from lightning, connect a lightning surge absorber.



- **Ground the lightning surge absorber (E1) and the PL (E2) separately.**

- **Select a lightning surge absorber which will not exceed the allowable circuit voltage, even when the voltage rises to the maximum.**



4.3.3 Grounding Cautions

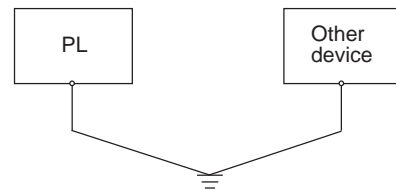
- Set up a dedicated ground when using the rear panel's FG terminal.

(a) Dedicated Ground - best *1

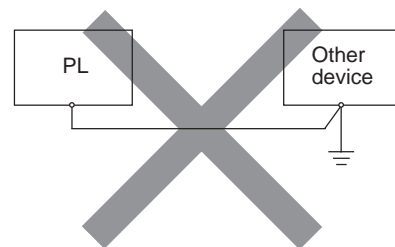


- If a dedicated ground is not possible, use a shared ground, as shown in figure (b).
- The grounding point must be as close to the PL as possible, and the grounding wires must be as short as possible. If the wires must be long, use thick, insulated wires and run them through conduits.

(b) Shared Ground - allowed *1



(c) Shared ground - not allowed



4.3.4 Cautions When Connecting I/O Signal Lines

- I/O signal lines must be wired separately from charged lines. If the power cord needs to be wired together with the (I/O) signal lines for any reason, use shielded lines and ground one end of the shield to the PL's FG terminal.
- To improve noise immunity, attaching a ferrite core to the power cord is recommended.

**1 Use a grounding resistance of less than 100Ω and a 2mm² or thicker wire, or your country's applicable standard. For details, contact your local PL distributor.*

Memo

Chapter

5 System Setup

5-1 Setup Procedures

5-2 System Parameters

5.1 Setup Procedures

This chapter explains how to enter a PL unit's system settings, as well as the cautions required both before and during set up.

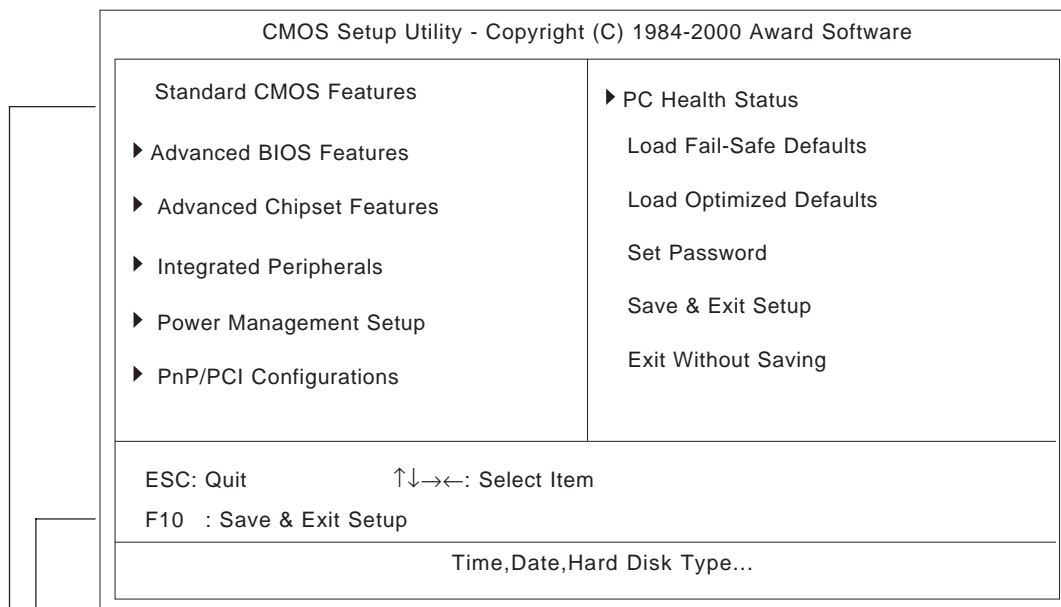


Normally, use only the factory (default) settings.



The following settings are those pre-set at the factory.

- 1) Connect a PS/2 keyboard to the PL.
- 2) Turn the PL unit's power ON.
- 3) After the message "Press to Enter SETUP" appears, press the [DEL] key until the following screen is displayed.



KEYBOARD ACTION KEYS

A summary of the keyboard keys used to move through screens and make selections.

SYSTEM SETTING SELECTION AREA

Each of the titles (areas) listed refers to a system setting area.

- 4) Use the arrow keys to move the cursor to the desired selection and use the [Enter] key to select an item.

5.2 System Parameters

Use the menu screen to select a System Item, and then enter the desired system information. Each item's detailed settings are shown here.



Normally, use only the factory (default) settings.

5.2.1 Standard CMOS Features

Select Standard CMOS Features and the following screen will appear.

CMOS Setup Utility - Copyright (C) 1984-2000 Award Software		
Standard CMOS Features		
Date (mm:dd:yy):	Thu, Aug 24 2000	Item Help
Time (hh:mm:ss):	11 : 15 : 14	
▶ IDE Primary Master	Press Enter10056 MB	Menu Level ▶
▶ IDE Primary Slave	Press Enter None	Change the day, month, year and century
Drive A	1.44M, 3.5 in.	
Drive B	None	
Video	EGA/VGA	
Halt On	All, But Disk/Key	
Base Memory	640K	
Extended Memory	56320K	
Total Memory	57344K	
↑↓→←: Move Enter:Select +/-/PU/PD:Value F10:Save ESC:Exit F1:General Help F5:Previous Values F6:Fail-Safe Defaults F7:Optimized Defaults		

■ Date/Time

This data sets the PL's internal time and date.

Hours :00 - 23

Minutes :00 - 59

Seconds :00 - 59

■ IDE Primary Master/ IDE Primary Slave

This display shows the capacity of the PL's IDE hard disk drive. Press [Enter] to display the parameter setting menu.

Reference "5.2.2 IDE Primary Master/IDE Primary Slave"

■ Drive A/Drive B

This setting determines the format used by the PL's internal floppy disk drive.

The available settings are [360K - 5.25in], [1.2M - 5.25in], [720K - 3.5in], [1.44M - 3.5in.], [2.88M - 3.5in.] and [None].

The A: drive's [1.44M - 3.5in] and the B: drive is [None]. These selections are factory set and recommended for most users.

■ Video

The selections for the screen (video) mode. The [EGA/VGA] and is recommended for most users. The other available settings are [CGA40], [CGA80] and [Mono].

■ **Halt On**

Designates the type of processing that will be performed when an error occurs during the Initial Start-Up Self Test. The [All But Disk /Key] and is recommended for most users.

[All Errors] : Displays all errors and stops the unit.

[No Errors] : Displays all errors and does not stop the unit.

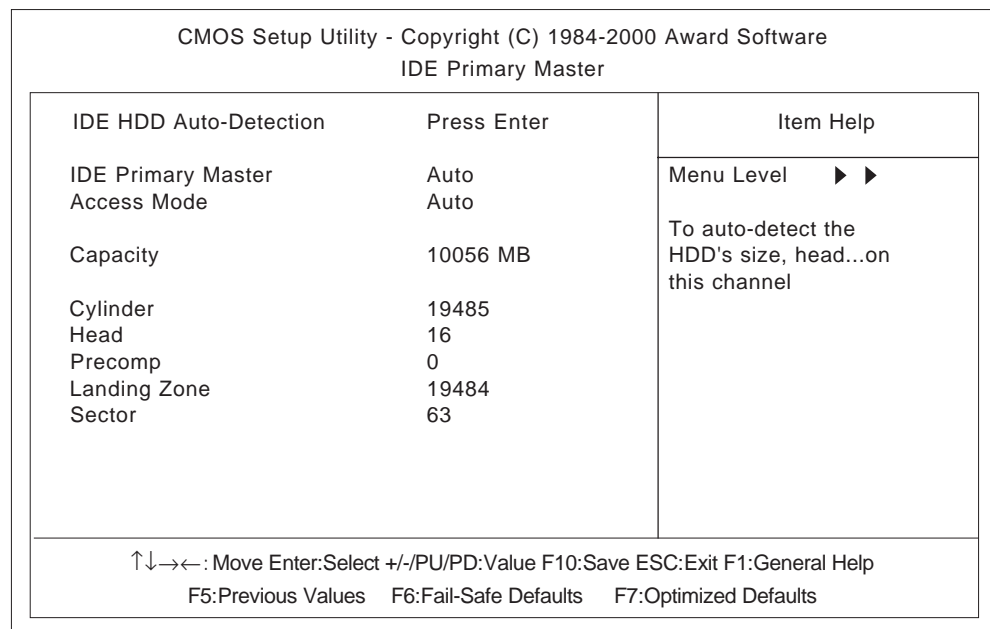
[All, But Keyboard]: Displays all errors, except for those related to the keyboard, and stops the unit. If the User has no keyboard connected, please use this setting.

[All, But Diskette]: Displays all errors, except for those related to the disk drive (FDD), and stops the unit.

[All, But Disk/Key]: Displays all errors, except for those related to the disk drive (FDD) and keyboard, and then stops the unit.

5.2.2 IDE Primary Master/IDE Primary Slave

Selecting IDE Primary Master or IDE Primary Slave from the Standard CMOS Features menu brings up the following screen.



■ **IDE HDD Auto-Detection**

This setting enables auto-detection of the IDE hard disk drive.

■ **Access Mode**

This setting determines the access mode of the PL's IDE hard disk drive. The available settings are [CHS], [LBA], [Large], and [Auto]. The factory default setting is [Auto] and is recommended for most users.

■ **IDE Primary Master (Slave)**

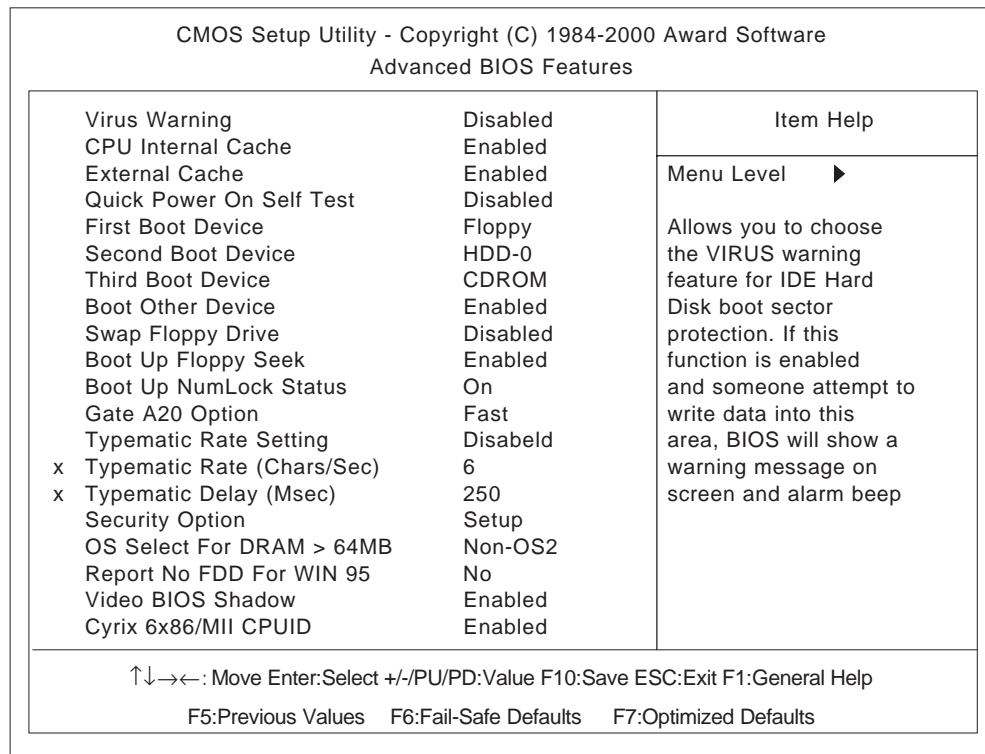
This setting sets the method for determining the parameter settings of the PL's IDE hard disk drive. The available settings are [None], [Auto], and [Manual]. The factory default setting is [Auto] and is recommended for most users.

■ **Capacity/ Cylinder/ Head/ Precomp/ Landing Zone/ Sector**

This setting determines the parameters of the PL's IDE hard disk drive. The setting is required when the [IDE Primary Master (Slave)] is set to [Manual] and the [Access Mode] is set to [CHS]. When the [IDE Primary Master (Slave)] is set to [None] or [Auto], the auto-detected values will be used. Capacity is set automatically.

5.2.3 Advanced BIOS Features

Select Advanced BIOS Features from the Main Menu and the following screen will appear.



■ Virus Warning

This setting determines whether to display a warning when a write to the boot sector is attempted. The available settings are [Enabled] or [Disabled]. The factory default setting is [Disabled] and is recommended for most users.

■ CPU Internal Cache

This setting determines the usage of the CPU's internal cache memory. The available settings are [Disabled] or [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ External Cache

This setting determines the usage of the external cache memory (L2). The available settings are [Disabled] or [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ Quick Power On Self Test

This setting determines whether the quick self test is performed when the power is turned on. The available settings are [Disabled] or [Enabled]. The factory default setting is [Disabled] and is recommended for most users.

■ **First Boot Device/ Second Boot Device/ Third Boot Device**

The selections for the search drive sequence of the operating system. The available settings are [Floppy], [HDD-0], [CDROM], [HDD-1], and [Disabled]. The factory default settings are [Floppy] for the [First Boot Device], [HDD-0] for the [Second Boot Device], and [CDROM] for the [Third Boot Device].

■ **Boot other Device**

This setting determines whether to allow the startup from devices other than those selected as [First Boot Device], [Second Boot Device], and [Third Boot Device]. The available settings are [Disabled] or [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ **Swap Floppy Drive**

This setting swaps Drive A with Drive B for the recognition. The available settings are [Disabled] or [Enabled]. The factory default setting is [Disabled] and is recommended for most users.

■ **Boot Up Floppy Seek**

The setting specifies the function to check whether the floppy disk drive is installed upon the system boot-up process. The available settings are [Disabled] or [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ **Boot Up Numlock Status**

This setting specifies the Numlock key status upon the startup. The available settings are [On] and [Off]. The factory default setting is [On] and is recommended for most users.

■ **Gate A20 Option**

The available settings are [Normal] and [Fast]. When [Normal] is selected, the Keyboard control will be employed to control Gate A20. When [Fast] is selected, the Chipset will be employed. The factory default setting is [Fast] and is recommended for most users.

■ **Typematic Rate Setting**

The setting specifies the keyboard speed for the repeat process of the character. The available settings are [Enabled] and [Disabled]. The factory default setting is [Disabled] and is recommended for most users.

■ **Typematic Rate (Chars/ Sec)**

This setting specifies the actual typematic rate (repeated character input per second) when the [Typematic Rate Setting] option is set to [Enabled]. The available settings are [6], [8], [10], [12], [15], [20], [24], [30]. The factory default setting is [6] and is recommended for most users.

■ **Typematic Delay (Msec)**

When [Typematic Rate Setting] is set to [Enabled], this setting determines the delay period until the initial repetition is started. (msec)=millisecond. The available settings are [250], [500], [750], and [1000]. The [250] selection is factory set.

■ Security Option

This setting designates the area to request a password. Select [Setup] or [System] upon BIOS setup, or [System] upon system startup. This setting is NOT available when the password is set using [Set Password] in the menu items. The factory default setting is [Setup] and is recommended for most users.

▼ **Reference** ▲ "5.2.15 Set Password"

■ OS Select For DRAM > 64MB

The available settings are [Non-OS2] and [OS2]. The factory default setting is [Non-OS2] and is recommended for most users.

■ Report No FDD For WIN 95

This setting determines if the FDD node is reported by BIOS to WIN95. The available settings are [No] and [Yes]. The factory default setting is [No] and is recommended for most users.

■ Video BIOS Shadow

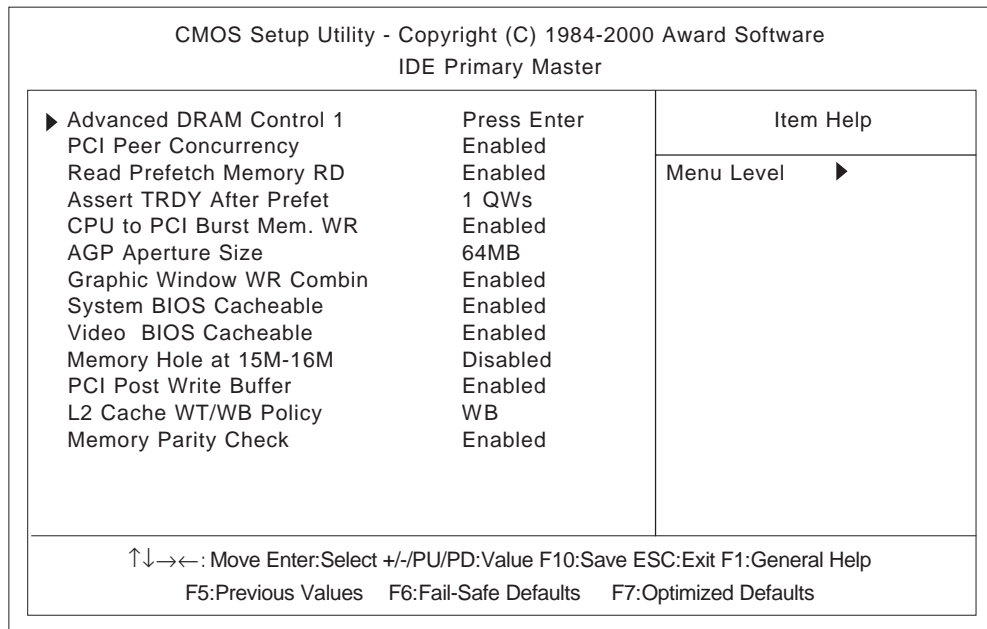
This setting determines whether to expand the Video BIOS ROM in RAM (C0000-C7FFF). The factory default setting is [Enabled] and is recommended for most users.

■ Cyrix 6x86/MII CPUID

This setting determines whether to send CPUID when Netware ver. 5.0 is used. The available settings are [Enabled] and [Disabled]. The factory default setting is [Enabled] and is recommended for most users.

5.2.4 Advanced Chipset Features

Select Advanced Chipset Features from the Main Menu and the following screen will appear.



■ Advanced DRAM Control 1

These selections display the DRAM setting. Press [Enter] to display the setting item menu. The available settings are [100MHz] or [Manual]. The factory default setting is [100MHz] and is recommended for most users.

■ PCI Peer Concurrency

This setting determines if the CPU will use L2/DRAM in parallel with PCI-to-PCI access. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ Read Prefetch Memory RD

This setting determines if the Memory Read command is used by the chipset to prefetch data. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ Assert TRDY After Prefet

This setting determines the TRDY assert data used for memory processing by the chipset. The available settings are [2QWs] and [1QWs]. The factory default setting is [1QWs] and is recommended for most users.

■ CPU to PCI Burst Mem. WR

The setting determines whether the PCI write buffer is used. The write buffer is not used when the [Disabled] option is selected. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ AGP Aperture Size

This setting specifies the memory used by the graphic board. The available settings are [4MB], [8MB], [16MB], [32MB], [64MB], [128MB], and [256MB]. The factory default setting is [64MB] and is recommended for most users.

■ Graphic Window WR Combin

This setting determines if the value designated in the GWBA register is accepted as the Graphic Window Base Address. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ System BIOS Cacheable

This setting determines whether to cache the system BIOS. An OS that uses the system BIOS operate faster. The available settings are [Enabled] and [Disabled]. The factory default setting is [Enabled] and is recommended for most users.

■ Video BIOS Cacheable

This setting determines whether to cache the video BIOS. The available settings are [Enabled] and [Disabled]. The factory default setting is [Enabled]. When this feature is [Enabled], the OS' BIOS ROM range available for caching is from C0000h - C7FFFh, which will improve the video performance. However, if another program tries to write to this area of memory, a system error may occur.

■ Memory Hole At 15M-16M

This setting determines whether to designate the memory space from 15MB to 16MB as the buffer area for the ISA bus card. The available settings are [Disabled] and [Enabled]. The factory default setting is [Disabled] and is recommended for most users.

■ PCI Post Write Buffer

This setting determines if the PCI Post Write Buffer is controlled or not. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ L2 Cache WT/WB Policy

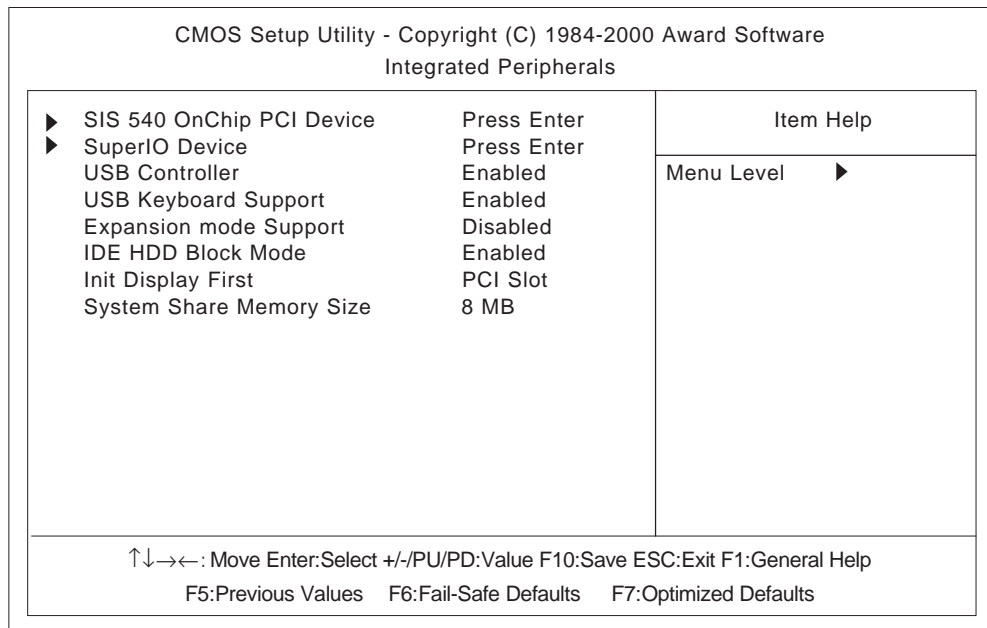
This setting determines degree of similarity between the L2 Cache and the System DRAM. The available settings are [WT] and [WB]. The factory default setting is [WB] and is recommended for most users.

■ Memory Parity Check

This setting determines whether parity checks are performed when parity protected memory is used. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

5.2.5 Integrated Peripherals

Select Integrated Peripherals from the Main Menu and the following screen will appear.



■ SIS 540 Onchip PCI Device

This setting enables the onboard Ethernet interface. Press [Enter] to display the menu items for this setting. The available settings are [Enabled] and [Disabled]. The factory default setting is [Enabled] and is recommended for most users.

■ Super 10 Device

This setting configures the various onboard interfaces. Press [Enter] to display the menu items for the setting.

Reference "5.2.6 Super 10 Device"

■ USB Controller

This setting determines whether to use the USB controller. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ USB Keyboard Support

This setting determines whether to use the USB interface keyboard. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ Expansion mode Support

This setting determines if the VGA Expansion Mode is supported or not. Settings available are [Disabled] and [Enabled]. The factory setting is [Disabled].

■ IDE HDD Block Mode

This setting determines whether to enable the Block Mode on the HDD supporting the Block Mode. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ Init Display First

This setting determines the type of VGA card that is given priority - AGP or PCI. The available settings are [PCI Slot] and [AGP]. The factory default setting is [PCI Slot] and is recommended for most users.

■ System Share Memory Size

This setting determines the amount of system memory used for video. The available settings are [2MB], [4MB], [6MB], [8MB], [16MB], [32MB] and [64MB]. The factory default setting is [8MB] and is recommended for most users.

5.2.6 Super I/O Device

Select Super I/O Device from the Integrated Peripherals menu and the following screen will appear.

CMOS Setup Utility - Copyright (C) 1984-2000 Award Software		
SuperIO Device		
Onboard FDC Controller	Enabled	Item Help
COM Port 1	3F8/IRQ4	
COM Port 4	2E8/IRQ10	Menu Level ▶▶
UART Mode Select	Normal	
x UR2 Duplex Mode	Half	
Onboard Parallel Port	3BC/IRQ7	
Parallel Port Mode	SPP	
x ECP Mode Use DMA	3	
COM Port 2	2F8	
COM Port 2 Use IRQ	IRQ3	
COM Port 3	3E8	
COM Port 3 Use IRQ	IRQ11	

↑↓→←: Move Enter:Select +/-/PU/PD:Value F10:Save ESC:Exit F1:General Help
F5:Previous Values F6:Fail-Safe Defaults F7:Optimized Defaults

■ Onboard FDC Controller

This setting enables or disables the onboard floppy disk controller. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ COM Port 1

This setting determines the port address and the interrupt request number used by the onboard serial port 1. The available settings are [Disabled], [3F8/IRQ4], [2F8/IRQ3], [3E8/IRQ11], [2E8/IRQ10] and [Auto]. The factory default setting is [3F8/IRQ4] and is recommended for most users.

■ COM Port 4

This setting determines the port address and the interrupt request number used by the onboard serial port connected to the touch panel inside the PL. The available settings are [Disabled], [3F8/IRQ4], [2F8/IRQ3], [3E8/IRQ11], [2E8/IRQ10] and [Auto]. The factory default setting is [2E8/IRQ10] and is recommended for most users.

■ UART Mode Select

This setting determines the operation mode of the onboard serial port 2. The available settings are [Normal], [IrDA], [ASKIR] and [SCR]. The factory default setting is [Normal] and is recommended for most users.

■ UR2 Duplex Mode

This setting determines the communication mode when serial port 2 is set to [IrDA] or [ASKIR] mode in [UART Mode Select]. The available settings are [Full] and [Half]. The factory default setting is [Half].

■ Onboard Parallel Port

This setting determines the port address and the interrupt request number used by the onboard parallel port. The available settings are [Disabled], [3BC/IRQ7], [278/IRQ5] and [378/IRQ7]. The factory default setting is [378/IRQ7] and is recommended for most users.

■ Parallel Port Mode

This setting determines the operation mode of the onboard parallel port. The available settings are [SPP], [EPP], [ECP] and [ECP+EPP]. The factory default setting is [SPP]. The available settings are [SPP] and [EPP] for the factory default setting. When the [Onboard Parallel Port] is [378/IRQ7] or [278/IRQ5], the available settings are [SPP], [EPP], [ECP] and [ECP+EPP].

■ ECP Mode Use DMA

This setting determines the DMA channel used in ECP mode. The available settings are [1] and [3]. This setting is user-definable when the Parallel Port Mode is set to [ECP] or [ECP+EPP].

■ COM Port 2

This setting determines the port address used by the onboard serial port 2. The available settings are [Disabled], [3F8], or [2F8], [3E8] and [2E8]. The factory default setting is [2F8] and is recommended for most users.

■ COM Port 2 Use IRQ

This setting determines the interrupt request number used by the onboard serial port 2. The available settings are [IRQ15], [IRQ3], [IRQ4], [IRQ9], [IRQ10] and [IRQ11]. The factory default setting is [IRQ3] and is recommended for most users.

■ COM Port 3

This setting selects the port address used by the onboard serial port 3. The available settings are [Disabled], [3F8], [2F8], [3E8] and [2E8]. The factory default setting is [3E8] and is recommended for most users.

■ COM Port 3 Use IRQ

This setting determines the interrupt request number used by the onboard serial port 3. The available settings are [IRQ15], [IRQ3], [IRQ4], [IRQ9], [IRQ10] and [IRQ11]. The factory default setting is [IRQ11] and is recommended for most users.

5.2.7 Power Management Setup

Select Power Management Setup from the Main Menu and the following screen will appear.

CMOS Setup Utility - Copyright (C) 1984-2000 Award Software		
Power Management Setup		
ACPI function	Disabled	Item Help
Video Off Option	Susp,Stby -> Off	
Video Off Method	V/H SYNC+Blank	Menu Level ▶
Power Button Over Ride	Instant Off	
Watchdog Function	Disabled	
x Watchdog Timer (sec)	5	
▶ PM Wake Up Events	Press Enter	
↑↓→←: Move Enter:Select +/-/PU/PD:Value F10:Save ESC:Exit F1:General Help F5:Previous Values F6:Fail-Safe Defaults F7:Optimized Defaults		

■ ACPI Function

This setting enables the ACPI function when ACPI-compatible peripherals are connected. The available settings are [Enabled] and [Disabled]. The factory default setting is [Disabled] and is recommended for most users.

■ Video Off Option

This setting determines the On/Off condition of the display. The available settings are [Always On], [Suspend → Off], [Susp, Stby → Off], and [All Modes → Off]. The factory default setting is [Susp, Stby → Off] and is recommended for most users.

■ Video Off Method

This setting determines the method to blank the display screen. The available settings are [Blank Screen], [V/H SYNC+Blank], and [DPMS Supported]. The [Blank Screen] selection blanks the display. The [V/H SYNC+Blank] blanks the display and also suspends the Vertical/Horizontal synchronization signal of the display. The [DPMS Supported] selection controls the operation when a CRT that supports DPMS is used. The factory default setting is [V/H SYNC+Blank] and is recommended for most users.

■ Power Button Over Ride

When enabled, this feature forces the system to enter the Soft-Off state if the power button is pushed for more than 4 seconds. The available settings are [Delay 4 sec] and [Instant Off]. The factory default setting is [Instant Off].

Chapter 5 - System Setup

■ Watchdog Function

This setting enables or disables the Watchdog function. The available settings are [Enabled] and [Disabled]. The factory default setting is [Disabled] and is recommended for most users.

■ Watchdog Timer (sec)

This setting determines the length of time of the Watchdog Timer. The available setting range is between [5] and [255]. The factory default setting is [5]. This setting is effective when the [Watchdog Function] is set to [Enabled].

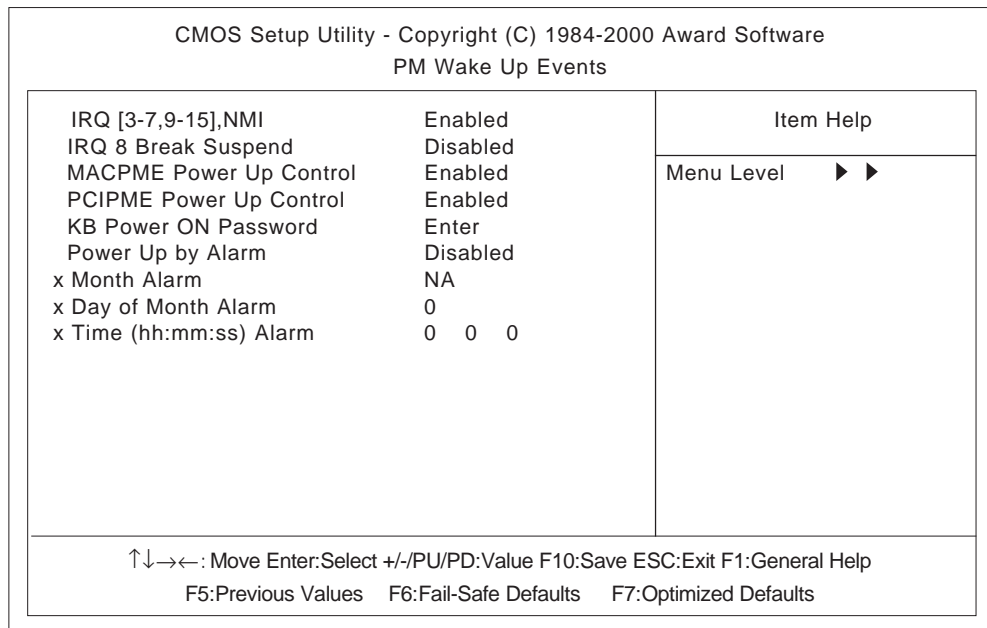
■ PM Wake Up Events

This selection displays the settings for system wake up. Press [Enter] to display the Parameter Setting menu.

▼Reference▲ "5.2.8 PM Wake Up Events"

5.2.8 PM Wake Up Events

Select PM Wake Up Events from the Power Management Setup Menu and the following screen will appear.



■ IRQ [3-7, 9-15], NMI

This setting determines if the system is reset when an allowed interrupt is detected. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ IRQ 8 Break Suspend

This setting determines if the data required to trigger the power-saving suspend mode is allocated to IRQ8 or not. The available settings are [Disabled] and [Enabled]. The factory default setting is [Disabled] and is recommended for most users.

■ MACPME Power Up Control

This setting determines if the PL starts via input from a LAN or not. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ PCIPME Power Up Control

This setting determines if the PL starts via input from a PCI card or not. The available settings are [Disabled] and [Enabled]. The factory default setting is [Enabled] and is recommended for most users.

■ KB Power ON Password

This setting defines the password when the system is powered-on by the keyboard switch. Press [Enter] to display the setup menu.

■ Power Up by Alarm

This setting determines whether to set the timer to start automatic startup. The available settings are [Disabled] and [Enabled]. The factory default setting is [Disabled] and is recommended for most users.

■ Month Alarm

This setting selects the month when the system starts up automatically. The available settings are [NA] and [1] through [12]. This function is available when the [Power Up by Alarm] option is set to [Enabled].

■ Day of Month Alarm

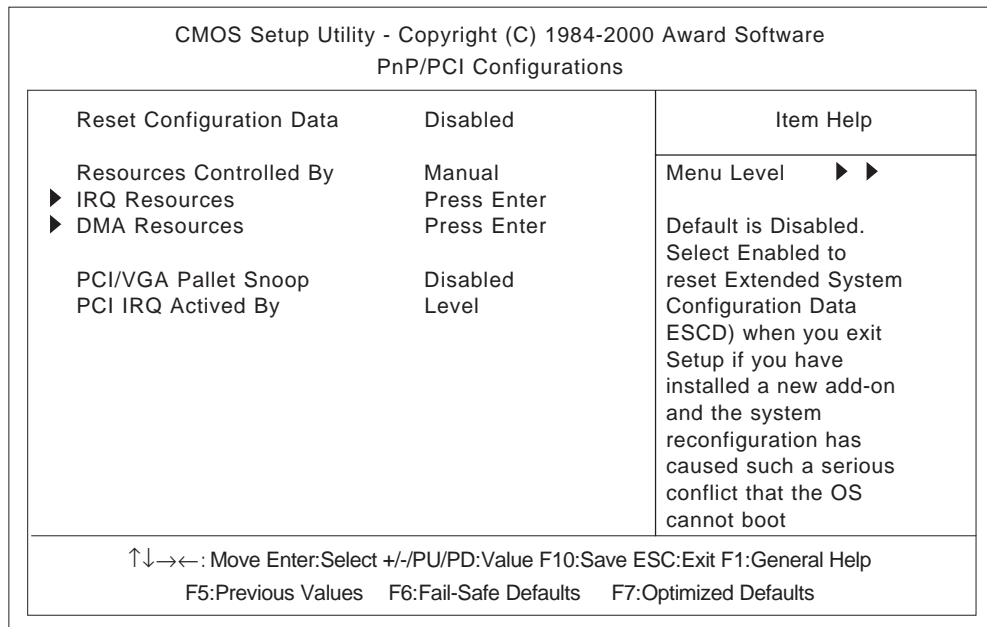
This setting selects the day when the system starts up automatically. The available settings are from [0] to [31]. This function is available when the [Power Up by Alarm] option is set to [Enabled].

■ Time (hh:mm:ss) Alarm

This setting specifies the time when the system starts up automatically. The available settings are [00] to [23] for "hh" (hour), [00] to [59] for "mm" (minute), and [00] to [59] for "ss" (second). This function is available when the [Power Up by Alarm] option is set to [Enabled].

5.2.9 PnP/ PCI Configurations

Select PnP/ PCI Configuration from the Main Menu and the following screen appears.



■ Reset Configuration Data

This setting determines whether to initialize the ESCD (Extended System Configuration Data) used by Plug and Play devices when setup is complete. The available settings are [Disabled] and [Enabled]. The factory default setting is [Disabled] and is recommended for most users.

■ Resources Controlled By

This setting selects the method for allocating the Plug and Play I/O Port, IRQ, and DMA resources. The available settings are [Auto (ESCD)] and [Manual]. If [Auto(ESCD)] is selected, the IRQ Resources and DMA Resources selection will be disabled. The factory default setting is [Manual] and is recommended for most users.

■ IRQ Resources

This selection displays the configuration settings for devices assigned an IRQ. Press [Enter] to display the setup menu. When Resources Controlled By is set to [Manual], device IRQ assigning settings must be performed manually.

Reference 5.2.10 *IRQ Resources*

■ DMA Resources

This selection displays the configuration settings for devices assigned a port address. Press [Enter] to display the setup menu.

Reference 5.2.11 *DMA Resources*

■ **PCI/VGA Pallet Snoop**

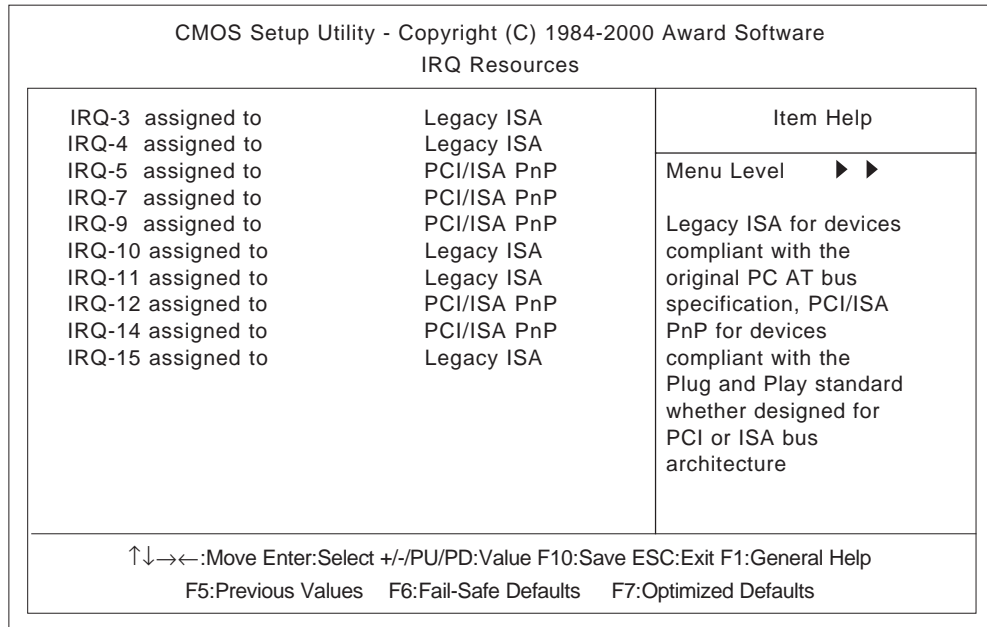
This setting is used when a MPEG card and a VGA card are both installed in the PL. The available settings are [Disabled] and [Enabled]. The factory default setting is [Disabled] and is recommended for most users.

■ **PCI IRQ Activated By**

This setting defines the method of interruption used by the PCI slot. The available settings are [Level] and [Edge]. The factory default setting is [Level] and is recommended for most users.

5.2.10 IRQ Resources

Select IRQ Resources from the PnP/ PCI Configurations menu and the following screen will appear.



■ IRQ-3 assigned to ~ IRQ-15 assigned to

This setting determines the type of device assigned to the IRQ. This function is available when the [Resource Control By] option under the [PnP/ PCI Configurations] menu is set to [Manual].

[PCI/ISA PnP] .. Select to use a PnP-ready PCI or ISA card.

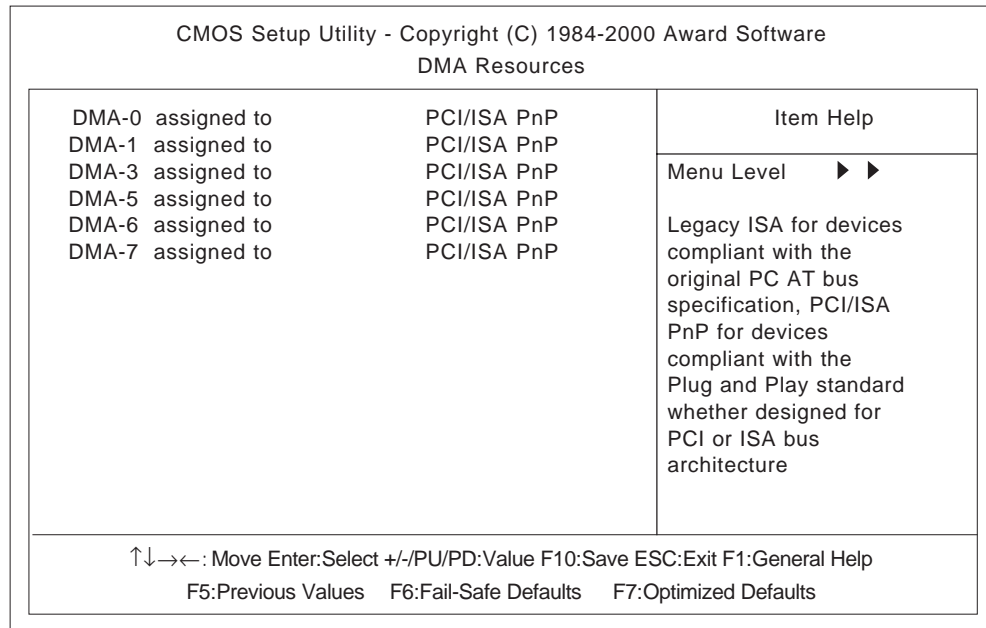
[Legacy ISA] Select to use a non-PnP ISA card.

The initial settings are as shown below.

	Initial Value		Initial Value
IRQ-3 assigned to	Legacy ISA	IRQ-10 assigned to	Legacy ISA
IRQ-4 assigned to	Legacy ISA	IRQ-11 assigned to	Legacy ISA
IRQ-5 assigned to	PCI/ISA PnP	IRQ-12 assigned to	PCI/ISA PnP
IRQ-7 assigned to	PCI/ISA PnP	IRQ-14 assigned to	PCI/ISA PnP
IRQ-9 assigned to	PCI/ISA PnP	IRQ-15 assigned to	Legacy ISA

5.2.11 DMA Resources

Selecting DMA Resources from the PnP/PCI Configuration menu and the following screen will appear.



■ DMA-0 assigned to ~ DMA-7 assigned to

This setting determines the type of device assigned to the port address. This function is available when the [Resource Control By] option under the [PnP/PCI Configurations] menu is set to [Manual].

[PCI/ISA PnP]...Select to use a PnP PCI or ISA card.

[Legacy ISA]...Select to use a non-PnP ISA card.

The initial settings are as shown below.

	Initial Value		Initial Value
DMA-0 assigned to	PCI/ISA PnP	DMA-5 assigned to	PCI/ISA PnP
DMA-1 assigned to	PCI/ISA PnP	DMA-6 assigned to	PCI/ISA PnP
DMA-3 assigned to	PCI/ISA PnP	DMA-7 assigned to	PCI/ISA PnP

5.2.12 PC Health Status

Select PC Health Status from the Main Menu and the following screen will appear.

CMOS Setup Utility - Copyright (C) 1984-2000 Award Software		
PC Health Status		
CPU Warning Temperature	Disabled	Item Help
System Warning Temp	: Disabled	
IN0(Vcore) : Tolerance	: Disabled	Menu Level ▶
IN1(3.3V) : Tolerance	: Disabled	
IN2(5V) : Tolerance	: Disabled	
IN3(12V) : Tolerance	: Disabled	
IN4(-12V) : Tolerance	: Disabled	
IN5(-5V) : Tolerance	: Disabled	
CPU Fan : Tolerance	: Disabled	
Power Fan : Tolerance	: Disabled	
↑↓→←: Move Enter:Select +/-/PU/PD:Value F10:Save ESC:Exit F1:General Help F5:Previous Values F6:Fail-Safe Defaults F7:Optimized Defaults		

■ CPU Warning Temperature

This setting defines the CPU temperature at which a warning will be issued. The available settings are [40°C/104°F], [45°C/113°F], [50°C/122°F], [55°C/131°F], [60°C/140°F], [65°C/149°F], [70°C/158°F], [75°C/167°F], [80°C/176°F] and [Disabled]. The factory default setting is [Disabled].

■ System Warning Temp

This setting defines the motherboard temperature at which a warning will be issued. The available settings are [40°C/104°F], [45°C/113°F], [50°C/122°F], [55°C/131°F], [60°C/140°F], [65°C/149°F], [70°C/158°F], [75°C/167°F], [80°C/176°F] and [Disabled]. The factory default setting is [Disabled].

■ IN0 (Vcore)/ In1 (3.3V)/ IN2 (5V)/ IN3 (12V)/ IN4 (-12V)/ IN5 (-5V)

This setting determines the allowable range of the voltage of IN0 (Vcore)/ In1 (3.3V)/ IN2 (5V)/ IN3 (12V)/ IN4 (-12V)/ IN5 (-5V). The available settings are [+/-4%], [+/-6%] and [Disabled]. The factory default setting is [Disabled].

■ CPU FAN

This setting determines the allowable range of CPU FAN speed. The available settings are [$\pm 30\%$], [$\pm 50\%$] and [Disabled]. The factory default setting is [Disabled] and is recommended for most users.

■ POWER FAN

This setting determines the allowable range of the Power Fan speed. The available settings are [$\pm 30\%$], [$\pm 50\%$] and [Disabled]. The factory default setting is [Disabled] and is recommended for most users.

5.2.13 Load Fail-Safe Defaults

When the Menu screen's [Load Fail-Safe Defaults] is selected, you are able to designate if the minimum number of System Settings is used or not. The selections are [Y] and [N].

5.2.14 Load Optimized Defaults

Selecting [Load Optimized Defaults] designates whether or not you revert to the PL unit's factory settings. The selections are [Y] and [N].

5.2.15 Set Password

This password is used to view system information settings. It is designed to prevent unapproved users from viewing the system information settings. Entering up to 8 characters here will overwrite the current password.

When you wish to have no password, click on the [Enter] key. Next, the words "PASSWORD DISABLE" will appear, providing confirmation that the Password is no longer set.

When password input is required, use the [Advanced BIOS Features] area's [Security Option] feature to enter the password. See 5.2.3 ADVANCED BIOS FEATURES

5.2.16 Save & Exit Setup

This feature saves the settings entered in the Setup Utility and restarts the PL unit.

5.2.17 Exit Without Saving

This feature quits the Setup Utility program without saving any settings entered.

Chapter

6 OS Setup

- 6.1 CD-ROM Contents
- 6.2 Setting Up Your PL OS
- 6.3 Installing Drivers
- 6.4 Application Features
- 6.5 WindowsNT®4.0 / Windows®2000 Cautions
- 6.6 MS-DOS® Utility Programs

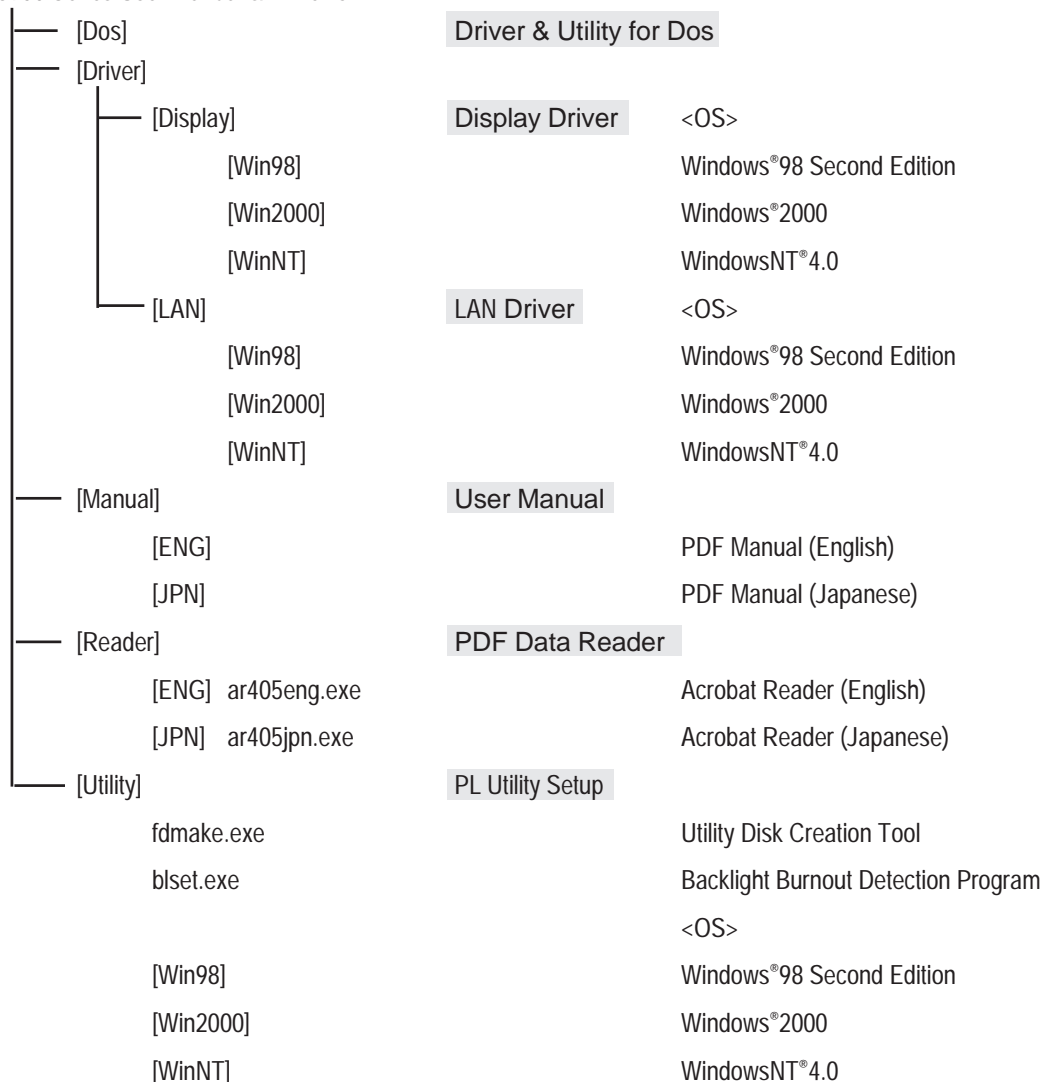
Pro-face has prepared the following additional program files which are not supported by the standard versions of the Windows®98 Second Edition, Windows®2000 and WindowsNT®4.0 operating systems. These files are located on the PL unit's additional "PL-5900 Series User Manual & Driver CD".

6.1 CD-ROM Contents

6.1.1 Diagram

The following tree-structure diagram shows the contents of the CD-ROM disk.

PL-5900 Series User Manual & Driver CD



6.2 Setting Up Your PL OS

Prior to using the PL unit with the MS-DOS®/Windows®98 Second Edition/WindowsNT®4.0/Windows®2000 operating system, certain utility software must be installed.

■ Installing the HDD Unit

A hard disk unit must be installed in the PL unit. Refer to **PL-HD220 Installation Guide**.

■ System Parameters Setup

System Parameters Setup must also be entered into the PL unit. After these setting are entered, check that the HD drive is correctly recognized by the PL.

Refer to **PL-HD220 Installation Guide**.

■ OS Setup

The PL unit is designed to operate using the following OS types.



The PL unit is designed to operate under the following standard Windows OS versions. PL operation with any other maker's OS is not guranteed.

MS-DOS®

Windows® 98 Second Edition

Windows NT® 4.0 (Windows Service Pack 3 or higher)

Windows® 2000

■ PL Utility Setup

Use the “PL-5900 Series User Manual & Driver CD” to install the necessary drivers and utility software.

- To set up the PL, a PS/2 type (Mini DIN) keyboard is required.
- To use the PL unit's touch panel, the touch panel device driver (PL-TD000) is required. When installing this driver, be sure to designate the COM port as COM4.

For installation details, **Reference** the Installation Guide included with the PL-TD000.



◆ **Installing Software from the CD-ROM**

To install the software on the PL, Digital's CD-ROM drive (PL-DK200) is required.

To set up the correct software for your PL unit's OS, be sure to use the "Disk1" folder's "Setup.exe" file.

Ex. When running Windows®98 Second Edition, and the CD-ROM drive is "D":

D:\Utility\Win98*1\Disk1\Setup.exe



When installing utility software included in the PL unit's CD-ROM, certain display problems may occur, such as icons darkening, etc. To solve this problem, after installing any utility software, be sure to also install the Graphics Accelerator driver.

◆ **Installing Software from a FD**



To install the software on the PL, Digital's FDD drive (PL-FD500) is required.

Use the PL's floppy disk drive to create a floppy disk that can be used to install the Driver & Utility programs designed specifically for your PL unit's OS. This floppy disk creation program is designed to run on Windows®.

Step 1 Insert the PL unit's additional CD-ROM disk "PL-5900 Series User Manual & Driver CD" in the PL's CD-ROM drive.

Step 2 Double click on the "Fdmake.exe" program, that is located in the CD-ROM disk's [OS] -> [Disk 1] folder. This will create the FD (Utility Disk). D:\Utility\Fdmake.exe (CD-ROM drive is "D")

Step 3 Insert the FD(Utility Disk) in the PL's FDD unit.

Step 4 Insert the FD into drive A: and double click on the "Setup.exe" file. Follow the instructions given by the Installer program to complete the installation.

◆ **Proface Folder Contents (on PL hard disk)**

When you set up the PL unit's utility software, the folder [Proface] will automatically be created on the C: drive. Inside that folder are the following programs.

(Same for all Windows OS types.)

[Proface]	
├── [Blsaver]	Backlight control screen saver
├── [Disp]	Display ON/OFF utility
├── [Display]	Graphic Accelerator Driver
├── [Keyclick]	On-screen Keyboard Emulator
├── [Lan]	LAN driver
├── [P159api]	API-DLL
└── [Sysmon]	System monitor/RAS application

*1 Windows® 98 Second Edition:	"Win98"
Windows NT® 4.0 (Windows Service Pack 3 or higher):	"Winnt"
Windows® 2000 :	"Win2000"

6.3 Installing Drivers

In order to use the PL unit's special features, 2 types of drivers have been created. ([Graphic Accelerator] and [LAN])

If your PL has no pre-installed OS, or has had its OS recovered, please install the following drivers as required.

The following explanation assumes the utility programs have been previously installed on your PL unit's hard disk in the [Proface] folder.



Drivers set up manually cannot be uninstalled.

■ Installing the Graphic Accelerator Driver

Use the following explanation to install the PL-5900 series Graphic Accelerator driver in your PL unit. Installing this driver will speed up your PL unit's display, using special hardware features.



Once the Graphics Accelerator driver is installed, the screen's resolution (display area) can be changed [640 x 480], [600 x 800] or [1024 x 768]. The resolution supported by PL-5900 Series units is 640 x 480 pixels. Also, the display color can be changed [256], [16bit] or [32bit]. The maximum display color supported by PL-5900 Series units is 16bit color.

◆ With Windows®98 Second Edition

- 1) Double click on [Display] from [Start] -> [Settings] -> [Control Panel (C)].
- 2) Click on [Advanced] from [Display] -> [Settings] tab.
- 3) Click on [Adapter] tab from [ATI Graphic Pro Turbo PCI Property] and then click on [Change].
- 4) Select [Search for a better driver than the one your device is using now] from [Update Device Wizard].
- 5) Enter C:\Proface\Display in [Specify a location] and then click on [Next].
- 6) Confirm [SiS540] exists and then click on [Next].
- 7) Click on [Update Device Wizard]'s [Enabled] button.
- 8) Click on [System Settings Change]'s [Yes] and then restart your PL.

◆ WindowsNT®4.0

- 1) Double click on [Display] from [Start] -> [Settings] -> [Control Panel].
- 2) Click on [Display Details] from [Display] tab in [Display Property].
- 3) Click on [Change] from [Display type].
- 4) Click on [Using Disk] from [Change Display].
- 5) In [Install from Floppy Disk], enter C:\Proface\Display\ in [Copy from] from and then click on [OK].
- 6) Confirm [SiS540] exists and then click on [OK].
- 7) Select [SiS540] from [Change Display] and then click on [OK].
- 8) Click on [System Settings Change]'s [Yes] button and then restart your PL.

◆ Windows®2000

- 1) Double click on [Tool] from [Start] -> [Settings(S)] -> [Control Panel].
- 2) Double click on [Computer].
- 3) Double click on [Device Manager]'s [Video Controller].
- 4) Click on [Video Controller Property]'s [Install driver].
- 5) Select [Search for a better driver than the one your device is using now] from [Hardware Device Driver Install] and then click on [Next].
- 6) Enter C:\Proface\Display in [Copy from] and then click on [OK].
- 7) Select [Specify allocation] from [Specify Driver Files] and then click on [Next].
- 8) Click on [Search for Driver Files]'s [Next].
- 9) Click on [Finished] and then restart your PL.

■ Installing the LAN Driver

Use the following explanation to install the PL-5900 series LAN driver in your PL unit. Installing this driver allows you to access a LAN.



Note:

- Before installing the LAN Driver, be sure the [Integrated Peripherals]'s [SIS 540 Onchip PCI Device] is set to [Enabled]. The factory default setting is [Enabled].
 ▼ **Reference** ▲ 5.2.5 Integrated Peripherals
- Be sure the PL unit's optional CD-ROM drive (PL-DK200) is connected and operating correctly prior to inserting the your OS's CD-ROM into the CD-ROM drive.

◆ With Windows® 98 Second Edition

- 1) Double click on [System] from [Start] -> [Settings] -> [Control Panel].
- 2) Select [PCI Ethernet Controller] from [System Property] -> [Device Manager].
- 3) Click on [Property].
 PCI Ethernet Controller Property Dialog will appear.
- 4) Click on [Change Driver] in [Driver] tab.
 Device Driver wizard will appear.
- 5) Click on [Next] button.
- 6) Select [Search for a better driver than the one your device is using now. (Recommended)] and click on [Next].
- 7) Click on the [Specify a location] check box, enter "C:\Proface\Lan" in the location window, and click on [Next].
- 8) Click on [Next].
- 9) Click on [Finish].
 The system settings dialog box will appear.
- 10) Click on the [Specify a location] check box, enter "C:\Proface\Lan" in the location window, and click on [Next].
 Files will be copied from the Windows95 CD-ROM to the PL unit.
- 11) Click on [Finished].
- 12) Click on [Yes] and restart the PL unit to complete the installation.

◆ With Windows NT® 4.0

- 1) Double click on [Network] icon from [Start] -> [Settings] -> [Control Panel].
The [Network Configuration] dialog box will appear.
- 2) Click on [Yes].
The Network Setup Wizard will appear.
- 3) Select [Wired to the network:], and click on [Next].
- 4) Click on [Select from list].
The Network Adaptor selection dialog box will appear.
- 5) Click on [Have disk].
The "Insert floppy disk" dialog box will appear.
- 6) Enter "C:\Proface\lan" and click [OK].
The "Select OEM Option" dialog box will appear.
- 7) Click on [OK].
The Network Setup wizard will appear.
- 8) Click on [Next].
- 9) Select the desired network protocol and click on [Next].
- 10) Select the desired service to install and click on [Next].
- 11) Click on [Next].
The WindowsNT setup dialog box will appear.
- 12) Enter "D:\I386" and click [Continue].
- 13) Enter "C:\Proface\Lan" and click [Continue].
The "Speed /Duplex mode" dialog box will appear.
- 14) Enter the appropriate settings for your Network.
The "Input Network Address" dialog box will appear.
- 15) Click on [Next].
- 16) Click on [Next].
- 17) Enter the settings to connect with your network.
The Network Setup Wizard will appear.
- 18) Click on [Finished].
- 19) Click on [Yes] to restart your PL.
The Service Control Manager dialog box will appear.
- 20) Restarting the PL will cause an error message to appear, which requires the PL's Service Pack to be reinstalled.
- 21) After the Service Pack is reinstalled, restart the PL.

◆ With Windows® 2000

- 1) Click on [System] icon from [Start] -> [Settings] -> [Control Panel].
The System property window will appear.
- 2) Click on the [Device Manager] from [Hardware] tab.
The Device Manager will appear.
- 3) Click on the [Other Device]'s [Ethernet Controller].
The Ethernet property window will appear.
- 4) Click on [Update Driver].
The Upgrade Device Driver Wizard will appear.
- 5) Click on [Next].
- 6) Select [Search for a suitable driver for my device (recommended)] and click [Next].
- 7) Select [Specify a location] and click [Next].
- 8) Enter "C:\Proface\Lan " and click [OK].
The search for the driver will start.
- 9) Click on [Next].
- 10) Click on [Finish] to restart your PL.

6.4 Windows NT[®] 4.0 / Windows[®] 2000 Cautions

Perform the following settings as required by your OS.

6.4.1 Automatic System Log-On Setup

■ When using Windows NT[®] 4.0

- 1) Click on the 2000 main screen's "Start" button, and select the "Enter Filename" item. Enter the text "C:\WINNT\REGEDIT.EXE" and press [Enter] to start the program.
- 2) When the REGEDIT Registry Tree appears, select the "Winlogon" subkey via the following text:
`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\WindowsNT\Current
Version\Winlogon.`
- 3) In the "DefaultUserName" field, enter the User name to be used for the Automatic Log-on.
- 4) Select the Edit menu's [New/String Value] feature.
- 5) To the Data Items present, add "AutoAdminLogon" to the Name column's data, and then enter "1" in that entry's Data field.
- 6) Add "Default Password" to the Name column's data, and enter the password used previously for the DefaultUserName in the Data field.



A user with no password cannot automatically log on.

- 7) REGEDIT data entry is now finished.



- If a User attempting to automatically log on is not attached to an "Administrators" group, i.e. no Default Password string is specified, Windows NT automatically changes the value of the AutoAdminLogon key from 1(true) to 0(false), thereby disabling the AutoAdminLogon feature. In that case, if the Shift key is held down during Logoff, the "Login Data" dialog box will appear and the User can use the name of another, registered User to log-on successfully.
- If Auto LogOn Setting are not entered, when logging on, a PS/2 type keyboard is required.

■ When using Windows[®] 2000

- 1) Start the Control Panel's [Users and Passwords].
- 2) Select an automatic log-on user and deselect the [Users must enter a user name and password to use this computer (E)] checkbox.
- 3) Click on the [Advanced!] tab and deselect the [Require users to press Ctrl-Alt-Delete before logging on (R)] checkbox.
- 4) Click the [Apply(A)] button and when the automatic login dialog box appears, enter your password.

6.4.2 Using an Uninterrupted Power Supply

Prior to turning OFF the PL's power, be sure to shut down the NT correctly via the NT OS' "Shutdown" feature. It is recommended that an Uninterrupted Power Supply Device is used to prevent the accidental loss of User data, due to an unexpected power outage.

When using an Uninterrupted Power Supply (UPS), the unit can be set to switch to backup power, which will provide enough time to safely shut down your PL, or it can even shut down your PL for you.

For details, please consult your local dealer of UPS units.

6.4.3 When Changing the System Design

When the PL unit is connected to a printer or to a LAN network, the Windows system settings must be changed.

■ When using Windows NT®4.0

• Changing the System Design

When the Windows NT® 4.0 system design is changed, the following messages will appear.

Windows NT Setup

Setup needs to copy some Windows NT files

Setup will look for the files in the location specified below. If you want Setup to look in a different place, type the new location. When the location is correct, click Continue.

Files Needed

Some files on WindowsNT Workstation CD-ROM are needed. Insert WindowsNT Workstation CD-ROM into the drive selected below, and then click OK.

Even when one of these messages appears, designate a new location for the system design change folder (Windows NT® 4.0 CD-ROM's [I386]) and click on [Next].

D:\I386 (CD-ROM drive is "D")

• Reinstalling Service Pack Data

When changing the Windows NT system settings, the system files are written over your existing Service Pack 1 files. Be sure to set up your Service Pack data again.

■ **When using Windows® 2000**

• **Changing the System Design**

When the Windows® 2000 system design is changed, the following messages will appear.

Please insert the floppy disk labeled 'Windows2000 Professional CD-ROM' into drive D and then click OK.

You can also click OK if you want files to be copied from an alternate location. such as a network sever or a compact disc.

Designate the new folder location for the system settings (Windows®2000 CD-ROM's [I386]) and click on [Next].

D:\I386 (CD-ROM drive is "D")

6.4.4 Changing to the NTFS File System

■ **With a hard disk using WindowsNT® 4.0 and Windows® 2000**

If your hard disk was formatted using the Windows DOS compatible FAT32 system, you can use the following command to convert the hard disk to an NTFS system.

convert x:/fs:ntfs, where "x" is the drive name of your hard disk.



After converting data to the NTFS file system, it cannot be converted back to the FAT32 (DOS compatible) file system.

6.5 Windows® Utility Program

The PL unit is equipped with the following special features. The following files have been copied to the PL unit's hard disk and are contained in the [Proface] folder.

File Name	Windows® 98 Second Edition	Windows NT® 4.0/Windows® 2000
PL_BLIOC.DLL	C:\Windows\System	C:\Winnt\System32
PL_DLL.DLL		
PL_IOC.DLL		
Backlight Control.scr		
Disp.exe	C:\Proface\Disp	
Keyclick.exe	C:\Proface\Keyclick	
PL_Smon.exe	C:\Proface\Sysmon	
PL_Wps.exe	C:\Proface\Sysmon	
Funkey.exe	C:\Proface\Funkey	

6.5.1 API-DLL

This is a dynamic library designed to provide access to the System BIOS' RAS feature for User applications. API-DLL consists of three types, which are explained below.

■ Backlight Control API-DLL file (Pl_blioc.dll)

This API-DLL file provides a dynamic library that allows User-created applications to utilize the PL-5900 series' backlight control feature. This file must be installed into the same directory as the User's application. *For details, refer to the Appendix 4 - Backlight Control Feature API-DLL*

■ System Monitor API-DLL (Pl_dll.dll)

This API-DLL file provides a dynamic library that allows User-created applications to utilize the PL-5900 series' System Monitor feature. This file must be installed into the same directory as the User's application.

■ RAS Feature API-DLL (Pl_ioc.dll)

This API-DLL file provides a dynamic library that allows User-created applications to utilize the PL-5900 series' System BIOS' RAS feature.

For details, refer to the Appendix 3 - System Monitor/RAS Feature API-DLL

6.5.2 Backlight OFF Screen Saver(Backlight control.scr)

This software is used to turn OFF the PL's backlight after a specified period of inactivity. The use of this feature will help to extend the life of the PL's backlight.



Certain application programs may not allow the PL's backlight to turn OFF. Please test each program individually to check if the screen saver will operate correctly.

6.5.3 Screen Display ON/OFF Utility(Disp.exe)

This command line utility is used to turn OFF both the PL's backlight and display.

Settings Used	DISP [ON/OFF]
Option Switch	ON: Displayed / OFF: Not Displayed
Return Value	0: Completed Normally / -1: Option Switch Error

6.5.4 Keyboard Emulator(Keyclick.exe)

This program allows the User's mouse operation to perform keyboard-like data input.



- Certain application programs do not support this keyboard emulator. Please test each application individually to check if the keyboard emulator will operate correctly.
- This application cannot be used to enter Windows® startup screen User Name and Password information.
- To change the Keyclick program's font size a keyboard is required.
- For details concerning the Keyclick program's operation, simply click on the HELP button to call up the program's online help data.

6.5.5 System Monitor/RAS Application (PI_smon.exe/PI_wps.exe)

This utility provides monitoring of the PL's temperature, voltage level, and fan's operation, via the system BIOS' RAS and system monitoring functions.

◆ System Monitor Program (PI_Smon.exe)

For details, **Reference** Appendix 3.3 System Monitor Operation

◆ Monitor Parameter Setting Program (PI_Wps.exe)

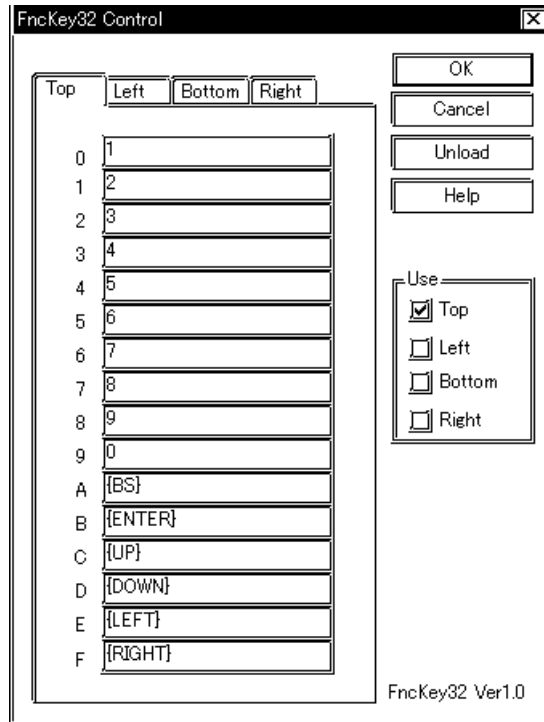
For details, **Reference** Appendix 3.2 System Monitor Property Settings

6.5.6 Function Key Utility(Funckey.exe)

This utility reserves an area of the PL screen for using function keys.

■ **Start-up**

- 1) Start up the Funckey32 Control Dialog Box from [Start] - [Program] - [Funckey] - [Funckey Configuration].
- 2) Select a the desired Function Key tab and click the [USE] area.



◆ **Special Key Settings**

When creating a special key, use the following code;

Key	Setting	Key	Setting
Alt	{ALT}	Tab	{TAB}
Back Space	{BS}	Up	{UP}
Break	{BREAK}	F1	{F1}
Caps	{CAPSLOCK}	F2	{F2}
Ctrl	{CONTROL}	F3	{F3}
Del	{DEL}	F4	{F4}
Down	{DOWN}	F5	{F5}
End	{END}	F6	{F6}
Enter	{ENTER}	F7	{F7}
Esc	{ESC}	F8	{F8}
Help	{HELP}	F9	{F9}
Home	{HOME}	F10	{F10}
Insert	{INSERT}	F11	{F11}
Left	{LEFT}	F12	{F12}
Num Lock	{NUMLOCK}	Shift [DOWN]	{SHIFT+}
Page Down	{PGDN}	Ctrl [DOWN]	{CONTROL+}
Page Up	{PGUP}	Alt [DOWN]	{ALT+}
Print Screen	{PRTSC}	Shift [UP]	{SHIFT-}
Right	{RIGHT}	Ctrl [UP]	{CONTROL-}
Shift	{SHIFT}	Alt [UP]	{ALT-}
Scroll Lock	{SCROLLLOCK}		



When the taskbar has been shifted to the bottom of the PL screen, the lower (Bottom) row of function keys cannot be used.

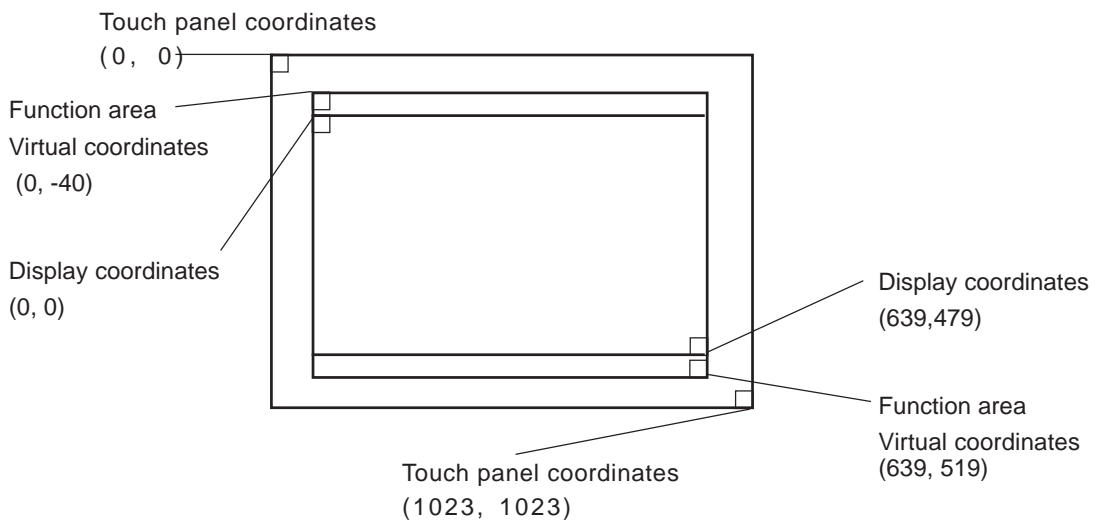
6.6 MS-DOS® Utility Programs

All MS-DOS® utilities can only be used if the PL unit's OS is MS-DOS. Windows OS' MS-DOS® prompt and command prompt cannot be used.

6.6.1 Touch Panel Handler(ATph59.exe)

With an analog touch panel, input is recognized within a 1024 x 1024 pixel area, with the lower left-hand corner as the coordinate origin point. However, most display panels use the upper left-hand corner as the origin point and have a resolution of 640 x 480 pixels. Consequently, depending on the conditions of use, the touch panel position and display position may not be the same. Here, the ATPH59.EXE application solves this problem by converting input from the touch panel into the corresponding display panel coordinates, allowing the use of application programs which use absolute coordinate input or area input from a touch panel.

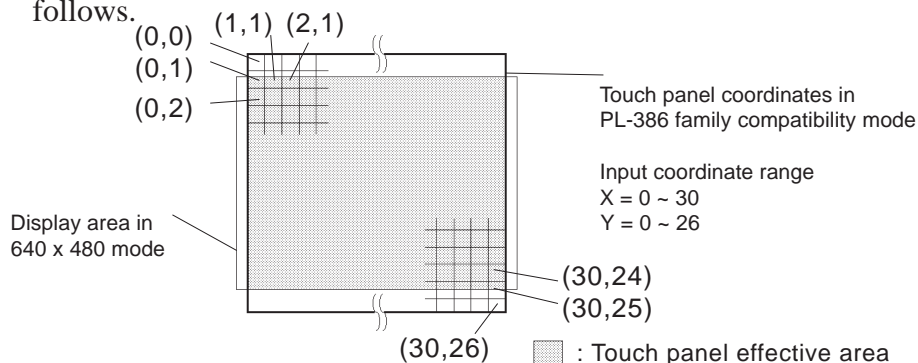
The relationship between touch panel coordinates and display coordinates is as follows.



Relationship between touch panel coordinates and display coordinates when using the PL-386 Series compatibility mode

To ensure compatibility with 16 x 14 (31 x 27 by double precision) touch panels used on the Digital PL-386 family of Panel Computers, the TPH.EXE (PL-386 command) function can be used as is.

The relationship between touch panel coordinates and display coordinates is as follows.



- **A 2-point touch gives the midpoint coordinate between the two coordinate values.**

■ Functions

ATPH59.EXE calls up functions using a software interrupt (default: INT 59h).

For information on functions in the PL-386 compatibility mode, see pages 6-7 to 6-10.

INT 59h Function List

Function code	Description
8000h	Touch panel initialization
8100h	Touch panel input (unrestricted wait)
8101h	Touch panel input (immediate restore)
8102h	Touch panel non-destructive input
8200h	Input buffer clear
8500h	Touch panel status detection

8000h Touch panel initialization

This initializes the touch panel and clears the touch panel's input buffer.

Input AX = 8000h

Output AH = 0: Successfully completed

After the application starts up, this function is issued.



Touch panel cannot be used for 0.5s after the function is issued.

8100h Touch panel input (unrestricted wait)

Returns coordinates of the area/position touched. Waits until data is input.

Input AX = 8100h

Output AH = 0: Successfully completed

BX = Y coordinate in 640 x 480 mode (-40 - 519)

DX = X coordinate in 640 x 480 mode (0 - 639)

CX = Number of valid input buffers on touch panel

8101h Touch panel input (immediate restore)

Returns coordinates of the area/position touched. Immediately restored if nothing is entered.

Input AX = 8101h

Output AH = 0: Input ON

1: Input OFF

BX = Y coordinate in 640 x 480 mode (-40 - 519)

DX = X coordinate in 640 x 480 mode (0 - 639)

CX = Number of valid input buffers on analog touch panel

8102h Touch panel non-destructive input

Returns coordinates of the area/position touched. Does not update the touch panel input buffer.

Input AX = 8102h

Output AH = 0: Input on
 1: Input off
 BX = Y coordinate in 640 x 480 mode (-40 - 519)
 DX = X coordinate in 640 x 480 mode (0 - 639)
 CX = Number of valid input buffers on analog touch panel

8200h Input buffer clear

Clears touch panel input buffers.

Input AX = 8200h

Output AH = 0: Successfully completed

8500h Touch panel status detection

Returns touch panel status

Input AX = 8500h

Output AH = Status in 640 x 480 mode

Bit 1	Bit 0	Description
0	0	Area touched
0	1	Unchanged
1	0	Not available
1	1	Area released

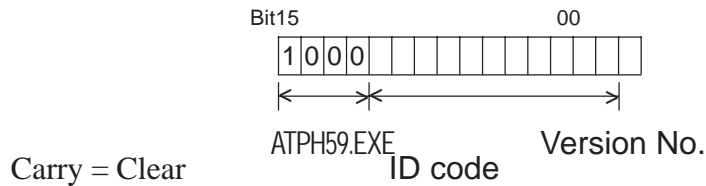
* For an explanation on how to use the function, see “function 500h.”

FE00h Resident check

When the ATPH59.EXE application resides in memory, returns a fixed message and version.

Input AX = FE00h

Output AH = 0: Successfully completed
 BL = ‘Y’
 BH = ‘B’
 CX = Version No.



Functions available in PL-386 Compatibility Mode

The following data details the functions available in the PL-386 compatibility mode (when the user's PL-386 application is used without further conversion, updating or formatting).

<INT 59h Function List>

Function code	Description
0000h	Touch panel initialization
0100h	Touch panel input (unrestricted wait)
0101h	Touch panel input (immediate restore)
0102h	Touch panel non-destructive input
0200h	Input buffer clear
0300h	Coordinate code register
0400h	Coordinate code input (unrestricted wait)
0401h	Coordinate code input (immediate restore)
0402h	Coordinate code non-destructive input

0000h Touch panel initialization

This initializes the touch panel and clears the touch panel's input buffer.

Input AX = 0000h

Output AH = 0: Successfully completed

After the application starts up, this function is issued.



Touch panel cannot be used for 0.5s after the function is issued.

0100h Touch panel input (unrestricted wait)

Returns coordinates of the area/position touched. Waits until input is made.

Input AX = 0100h

Output AH = 0: Successfully completed
 BH = Y coordinate range in PL-386 compatibility mode (0 - 26)
 BL = X coordinate range in PL-386 compatibility mode (0 - 30)
 CX = Number of valid data sets in touch panel input buffers

0101h Touch panel input (immediate restore)

Returns coordinates of the area/position touched. Immediately restored after input is made.

Input AX = 0101h

Output AH = 0: Input on ("1" when input is off)
 BH = Y coordinate range in PL-386 compatibility mode (0 - 26)
 BL = X coordinate range in PL-386 compatibility mode (0 - 30)
 CX = Number of valid data sets in touch panel input buffers

0102h Touch panel non-destructive input

Returns coordinates of the area/position touched. Does not update touch panel input buffer.

Input AX = 0102h

Output AH =0: Input present (No input = 1)
 BH = Y coordinate range in PL-386 compatibility mode (0 - 26)
 BL =X coordinate range in PL-386 compatibility mode (0 - 30)
 CX = Number of valid data sets in touch panel input buffers

0200h Input buffer clear

Clears touch panel input buffers.

Input AX = 0200h

Output AH = 0: Successfully completed

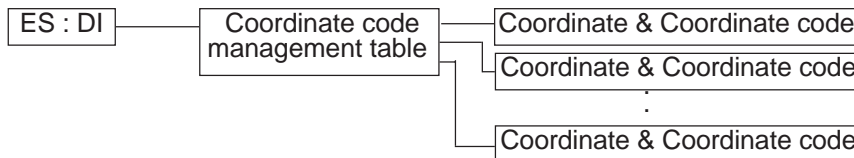
0300h Coordinate code register

Registers code corresponding to the display coordinates.

Input AX = 0300h
 ES = Segment from coordinate code management table
 DI = Offset from coordinate code management table

Output AH =0: Successfully completed

* Relationship between coordinate code management table and coordinate code



*** Coordinate Code Management Table Design**

The coordinate code management table manages positions of multiple coordinate code lists.

Number (n) of registered coordinate code lists
Offset of 1st coordinate code list
Segment of 1st coordinate code list
Offset of 2nd coordinate code list
Segment of 2nd coordinate code list
⋮
Offset of nth coordinate code list
Segment of nth coordinate code list

*** Coordinate Code List Design**

The coordinate code list determines which code is returned when the touch panel is pressed in a specific area. These coordinates specify the display coordinates.

Number (n) of registered coordinate codes
Display coordinate designation mode
Left side X coordinate of 1st area (X1)
Top side X coordinate of 1st area (Y1)
Right side X coordinate of 1st area (X2)
Bottom side X coordinate of 1st area (Y2)
Coordinate code of 1st area (code)
:
Left side X coordinate of nth area (X1)
Top side X coordinate of nth area (Y1)
Right side X coordinate of nth area (X2)
Bottom side X coordinate of nth area (Y2)
Coordinate code of nth area (code)

0400h Coordinate code input (unrestricted wait)

Returns coordinates of the area/position touched and coordinate code. Waits until data is entered.

Input AX = 0400h

Output AH = 0: Successfully completed
 BH = Y coordinate range in PL-386 compatibility mode (0 - 26)
 BL = X coordinate range in PL-386 compatibility mode (0 - 30)
 CX = Number of valid data sets in touch panel input buffers
 DX = Coordinate code



Note: To use function 0400h, it is necessary to first register the coordinate codes.

0401h Coordinate code input (immediate restore)

Returns coordinates of the area/position touched and coordinate code. Immediately restored after data is entered.

Input AX = 0401h

Output AH = 0: Input on ("1" when input is off)
 BH = Y coordinate range in PL-386 compatibility mode (0 - 26)
 BL = X coordinate range in PL-386 compatibility mode (0 - 30)
 CX = Number of valid data sets in touch panel input buffers
 DX = Coordinate code



Note: To use function 0401h, it is necessary to first register the coordinate codes.

0402h Coordinate code non-destructive input

Returns coordinates of the area/position touched. Does not update the touch panel input buffer.

Input AX = 0402h

Output AH = 0: Input on (“1” when input is off)
 BH = Y coordinate range in PL-386 family compatibility mode (0 - 26)
 BL = X coordinate range in PL-386 family compatibility mode (0 - 30)
 CX = Number of valid data sets in input buffers on touch panel
 DX = Coordinate code



Note: To use function 0402h, it is necessary to first register the coordinate codes.

0500h Touch panel status detection

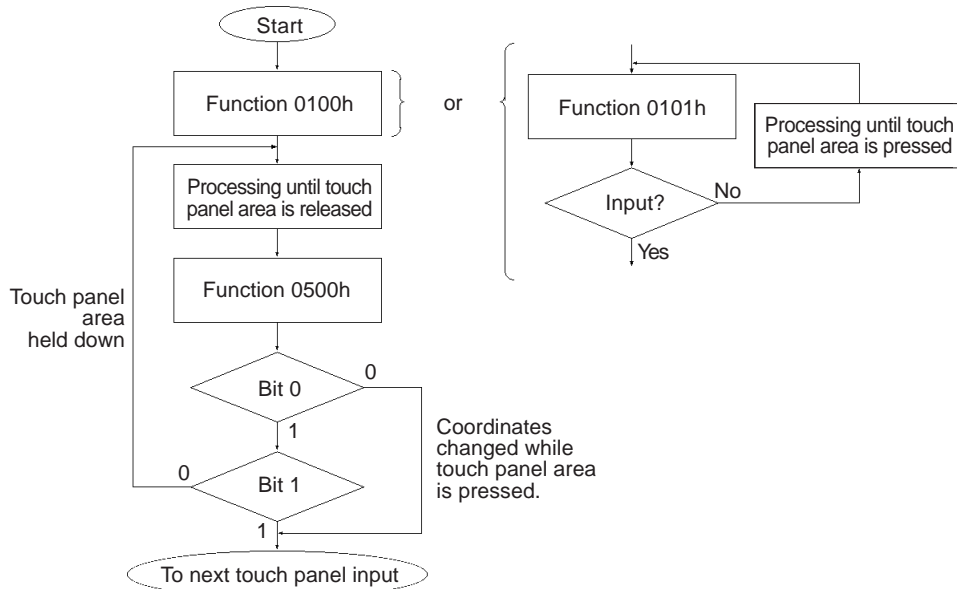
Returns the touch panel’s current status

Input AX = 0500h

Output AH = Status in PL-386 compatibility mode

Bit 1	Bit 0	Description
0	0	Area pressed
0	1	Unchanged
1	0	Not available
1	1	Area released

How to use function 0500h



6.6.2 Serial Port Driver(EXTCOM.SYS)

The Panel Computer (PL)'s RS-232C BIOS (INT 14h) has been enhanced and uses reception interrupt when transmitting data. (Can be used like a PC/AT standard function) As a result, this software does not need to be installed when an application for controlling the serial port directly is used, or when Windows is running.

- A Port Base Address: Reception Buffer Size (unit = KB)

The Port Base Address, or the Reception Buffer Size can be designated here.

Parameters for up to four ports can be entered, using the predefined Port numbers of 0 to 3 (COM1 to COM).

Next, the Port Base Addresses are shown.

Port Number	Port Base Address	Interrupt Device Level	Remarks
0	3F8h	IRQ4	COM1 RS-232C (SIO1)
1	2F8h	IRQ3	COM2 RS-232C (SIO2)
2	3E8h	IRQ11	COM3 RS-485 Multidrop Connection Possible
3	2E8h	IRQ10	COM4 Reserved for Touch Panel (Not available to User)

- N Designates the ports not used by EXTCOM.SYS.

Ex.) When port 1 is not to be used by EXTCOM.SYS;

DEVICE = EXTCOM.SYS -A3F8:1 -N -A3E8:1 -N can be used.



- A and -N recognize Port numbers based on the positions designated for them.
- When DEVICE = EXTCOM.SYS is entered, all the ports are used by EXTCOM.SYS.



With the PL-5900 series units, normally port number 3 (COM4) can not be used. As a result, be sure to designate all the unit's ports in this statement.

Ex.) DEVICE = EXTCOM.SYS -A3F8:1 -N -A3E8:1 -N -M

- M This is used when Port 2 (COM3:RS-485) is used for a Multi-Drop connection. When this port is designated for Multi-Drop, startup is performed with the DTR "OFF" (unable to transfer data).

■ Description of Features

EXTCOM.SYS has the following features and uses a software interrupt (INT 14h) to call the following functions.

INT 14h Function Code Chart

Function Code	Explanation
00h	Initialize Transmission port
01h	Send 1 byte data
02h	Receive 1 byte data
03h	Read Transmission port status
10h	Expansion setting
11h	Forced 1 byte data send
12h	1 byte data reception (immediate reply)
13h	Read Expansion Transmission port Status
15h	Receive 1 byte data safely
16h	Enable RS-485 Sending/Transmission
17h	Disable RS-485 Sending/Transmission
FEh	Resident Check



Note:

- **RS-232C (COM1,COM2) and RS-485 (COM3) can be used even if the EXTCOM.SYS program is not installed. However, the receive interrupt and expansion features cannot be used.**
- **Depending on the BIOS function call used, the RS-232C (COM1, COM2) and RS-485 (COM3) are enabled. Also, the RS-232C (COM1, COM2) can be used with only a device name.**
- **Port 3 (COM4) can be used to control the Touch Panel. Since PL-5900 series units use PLATPH for control, normally EXTCOM.EXE should be set to not use port 3.**

Next, each function will be explained.

The Line Status and the Modem Status bits are used as follows:

(Each of these bits is enabled when it is "1", and disabled when it is "0")

■ **Line Status**

Bit 0	Data Ready
Bit 1	Overrun Error
Bit 2	Parity Error
Bit 3	Frame Error
Bit 4	Break Detect
Bit 5	Transmit's Reserved Register Free
Bit 6	Transmit's Shift Register Free
Bit 7	Time Out Error

■ **Modem Status**

Bit 0	Unused
Bit 1	Unused
Bit 2	Unused
Bit 3	Unused
Bit 4	Clear To Send
Bit 5	Data Set Ready
Bit 6	Ring Indicator
Bit 7	Carrier Detect

Function 00h Initialize Transmission Port

This function performs Transmission Port initialization. Clears the Transmission buffer.

Input: AH = 00h,

AL = Port Parameter

Bit 0,1 Data bit length

00: Unsettable 01: Unsettable 10: 7 bit 11: 8 bit (default)

Bit 2 Stop Bit

0: 1 stop bit 1: 2 stop bits (default)

Bit 3,4 Parity

00: NON (default) 01: ODD 10: NON 11: EVEN

Bit 5,6,7 Baud Rate

000:110, 001:150, 010:300, 011:600, 100:1200,
101:2400, 110:4800, 111:9600 (default)

DX = Port Number (0 to 3)

Output: AH = Line Status,

AL = Modem Status

Function 01h	Transmit 1 Byte Data
---------------------	-----------------------------

Transmits 1 byte data..

Input: AH = 01h

AL = Transmission Data

DX = Port Number (0 to 3)

Output: AH = Line Status (During timeout; bit 7 becomes 1)

AL = Transmission Data

Function 02h	Receive 1 Byte Data
---------------------	----------------------------

Receives 1 byte data. If characters are present in the buffer, while the buffer is being refreshed these characters are returned. If there are no characters in the buffer, the computer waits until a timeout occurs.

Input: AH = 02h

DX = Port Number (0 to 3)

Output: AH = Line Status

AL = Modem Status

Function 03h	Read Transmission Port Status
---------------------	--------------------------------------

Reads the status of the transmission port.

Input: AH = 03h

DX = Port Number (0 to 3)

Output: AH = Line Status

AL = Modem Status

Function 10h Expansion Setting

Designates the type of data transmission method used. XON/XOFF and RTS can be used at the same time.

Input: AH = 10h

DX = Flow Control Method

Bit 0 Flow is controlled according to the XON/XOFF setting at the time of data reception.

0: Not used for control (default) 1: Used for control

Bit 1 Flow is controlled according to the RTS setting at the time of data reception.

0: Not used for control (default) 1: Used for control

Bit 2,3 Reserved (Keep set to "0")

Bit 4 Other party's XON/XOFF setting during data transfer

0: Disabled (default) 1: Enabled

Bit 5 Other party's CTS setting during data transfer

0: Disabled (default) 1: Enabled

Bit 6,7 Reserved (Keep set to "0")

CH = Timing used for enabling XON (default is 25)

Designates what percentage the buffer must empty to before the XON command is enabled.

CL = Timing used for enabling XOFF (default is 75)

Designates what percentage the buffer must fill to before the XOFF command is enabled.

*Please be sure $0 < CH < CL < 100$.

BH = Length of transmit time out (Unit = 500msec) Default - 6 [3 seconds]

BL = Length of receive time out (Unit = 500msec) Default - 6 [3 seconds]

DX = Port Number (0 to 3)

Output: AH = Line Status

0: Normal exit Other than 0: Designated error

Function 11h	Forced 1 Byte Data Send
---------------------	--------------------------------

Regardless of the other party's data flow control, a bit of data is sent.

Input: AH = 11h

AL = Transmission Data

DX = Port Number (0 to 3)

Output: AH = Line Status

AL = Transmission Data

Function 12h	1 Byte Data Reception (Immediate Reply)
---------------------	--

Receives 1 byte data. Responds immediately if no data is present in the reception buffer.

Input: AH = 12h

DX = Port Number (0 to 3)

Output: AH = Line Status (If not data is present, bit 7 changes to "1"(timeout)

AL = Reception Data

Function 13h	Read Expansion Transmission Port Status
---------------------	--

When expansion settings are used, reads the condition of the data transmission port.

Input: AH = 13h

DX = Port Number (0 to 3)

Output: AH = Line Status

AL = Modem Status

BX = Current Data Reception Amount

CL = Condition of Flow Control

Bit 0 Sending (Here) Terminal's XON/XOFF

0: OFF, 1: ON

Bit 1 Sending (Here) Terminal's RTS

0: OFF, 1: ON

Bit 2,3 Unused

Bit 4 Receiving (There) Terminal's XON/XOFF

0: OFF, 1: ON

Bit 5 Receiving (There) Terminal's RTS

0: OFF, 1: ON

Bit 6,7 Unused

Function 15h	Receive 1 Byte Data Safely
---------------------	-----------------------------------

Receives 1 byte data. However, the reception buffer is not refreshed. Also, responds immediately if the buffer contains no data.

Input: AH = 16h

Output: (None)

Function 16h	Enable RS485 Sending/Transmission
---------------------	--

Receives 1 byte data. However, the reception buffer is not refreshed. Also, responds immediately if the buffer contains no data.

Input: AH = 15h

AL = Transmission Data

DX = Port Number (0 to 3)

Output: AH = Line Status

AL = Reception Data

Function 17h	Enable RS485 Sending/Transmission
---------------------	--

Used with Multi-Drop connections, via the RS-485 port (No. 2).

When DTR turns OFF, transmission is not possible.

Input: AH = 17h

Output: (None)

Function FEh	Resident Check
---------------------	-----------------------

When the EXTCOM.SYS program is resident, returns a fixed message and the version (number).

Input: AH = FEh

Output: BL = "Y"

BH = "A"

CX = Version Number

Carry = Clear

6.6.3 Touch Panel Data Calibration(CALIB59.EXE)

By touching the specified position (upper left-hand corner or lower right-hand corner) on the panel, the difference between the screen's logical value and its measured value is corrected. Furthermore, it is possible to create files with data based on calibration results obtained here, to be used with the ATPH59.EXE application (Touch Panel Handler).

■ Start-up

CALIB59 [Parameter] 

* Parameter

- a<n> Specifies the I/O base address of the touch panel's SIO port.
Hexadecimal, Default: 2e8 (COM4)
n= 3f8 (COM1)
2f8 (COM2)
3e8 (COM3)
2e8 (COM4)
- q<n> Specifies the interrupt level (IRQ) of the touch panel's SIO port.
Default: 10
n= 3, 4, 10, 11
- c <path name> Specifies the data file containing the calibrated value obtained from the CALIB59.EXE application (touch panel data calibration).
When defaulted to, ATPH59.CAL of the current directory is specified.



Note:

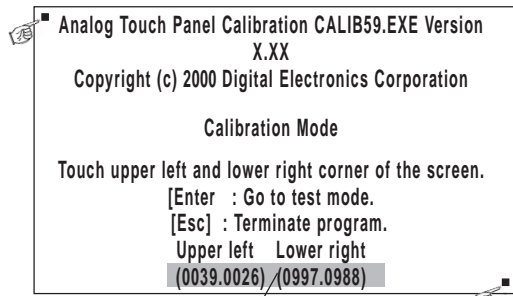
- * Example CALIB59 -a2e8 -q10 -cc:\atph59.cal
- Normally, only "CALIB59" is needed.

■ Operation

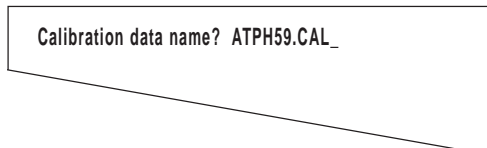
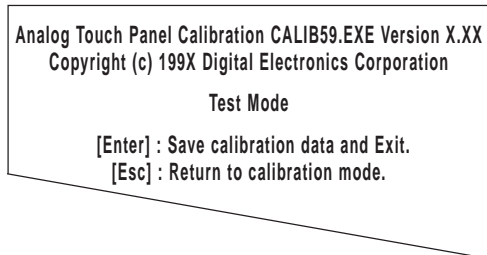
```

■ Analog Touch Panel Calibration CALIB59.EXE Version
X.XX
Copyright(c) 2000 Digital Electronics Corporation
Calibration Mode
Touch upper left and lower right corner of the screen.
[Enter] : Go to test mode.
[Esc] : Terminate program.
Upper left Lower right
(0000,0000) (0000,0000)
    
```

- 1) When the CALIB59.EXE application is opened up, the message shown at the left will appear on the screen, and two points will light up, in both the upper left-hand and lower right-hand corners.



The measured value is displayed on the screen.



2) Touch each point, in the order it appears.



- Do not touch both points simultaneously.
- Touch the panel exactly on the lit up areas.
- The measured value is re-displayed if you touch the panel again.

The difference between the logic value and the measured value is obtained.



To quit programming, press the [Esc] key. Then, when the message shown on the left appears, press the [Y] key. The program will end without saving data. Here, pressing the [N] key will return you to the calibration mode.

3) Start up the “Test Mode” with the [↵] key.

This mode tests the calibrated value to determine if it is correct or not.

The perimeter is OK if it is drawn along the path you traced by finger. Otherwise, return to the “Calibration Mode” and touch the lit up areas again.



You can return to the “Calibration Mode” with the [Esc] key.

4) If test results are OK, press the [Enter] key.

When the message shown at the left appears, input the data file name and press the [↵] key.



When specifying the data file name for the parameter (-C=[path name]) at the CALIB59.EXE start-up, the program ends without displaying the message shown on the left.

6.6.4 Keyboard Emulator(KEYEM_PL.EXE)

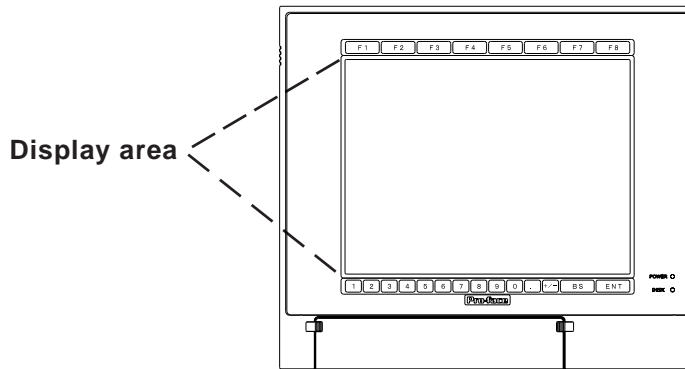
(Note: This program can only be used with the Japanese version of MS-DOS.)

This program graphically displays the keyboard on the screen, allowing keystroke operation with the touch panel using the mouse to perform keyboard-like data input. It also allows the user to define a key to any coordinates (external key definition).



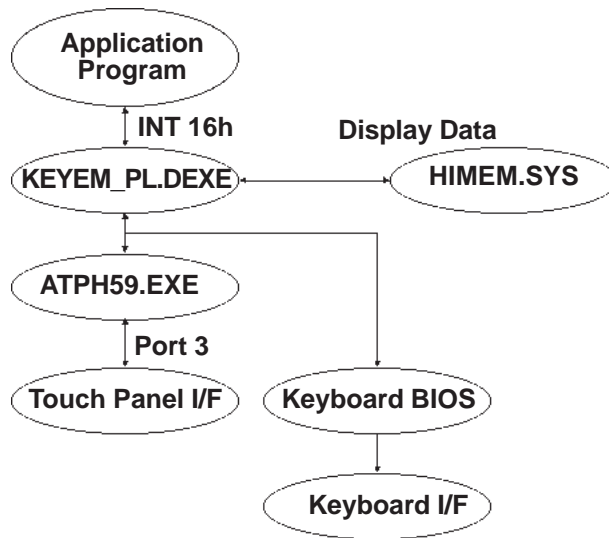
"External Key Definition" uses the touch panel coordinate mode compatible with Digital's family of PL-386 Panel Computers.

Please place the function label included in the package to enhance the usability of the Keyboard Emulator.



The structure and function of KEYEM_PL.EXE is as follows:

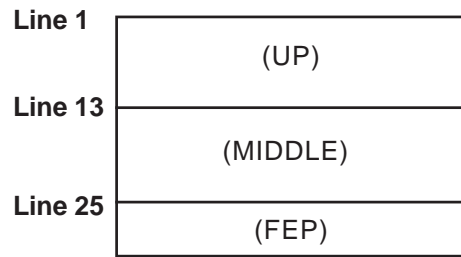
<The structure of KEYEM_PL.EXE>



- To use KEYEM_PL.EXE, HIMEM.SYS and ATPH59.EXE must be installed.
- Install the HIMEM.SYS file for the OS currently in use.

■ Screen Display

The screen is split into two as shown below and the graphical keyboard is displayed in the top or middle section (where the cursor is not residing).



Touch [F8] to turn ON/OFF the graphical keyboard.

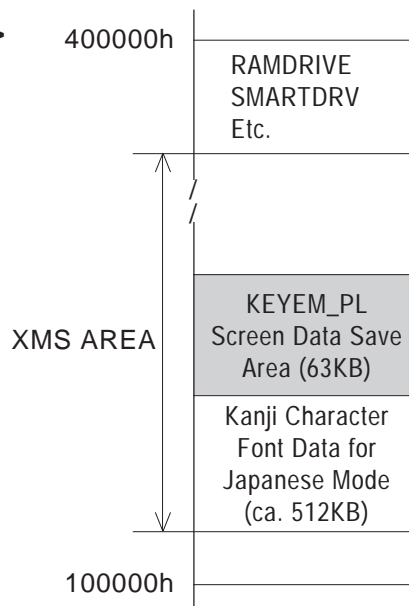


- **The display mode must be the DOS/V-compatible Japanese graphic mode (72h). The graphical keyboard display supports the AX standard Japanese keyboard only. For the actual keyboard display screen, refer to Appendix 5. "Keyboard Emulator Screens".**
- **KEYEM_PL runs on top of ATPH59. To process touch input from ATPH59 with a user application, exclusive control is necessary. Use function A000h and A001h to control the keyboard emulator when key input is necessary. Do not perform touch input processes during key input.**

■ Configurations

When the graphical keyboard is displayed, XMS memory is used to temporarily save the contents of the VRAM (63KB). If HIMEM.SYS is not installed, or if the KEYEM_PL screen data save area is not reserved, this program will not start up. When using this program in conjunction with programs that use expanded memory, be sure to reserve enough memory.

<Sample Memory Map>



■ Start-up

KEYEM_PL [Parameter] [↵] or KEYEM_PL-r [↵]

* Parameters

- 0[=][External Key Definition File]** Specifies the external key definition.
- 2** Displays the graphical keyboard in 2-level mode.
- i<n>** Sets the software interrupt vector number when calling up functions.
Hexadecimal, Default: n=59
- F** Displays the graphical keyboard in 16-level mode.
- T[=]n** Specifies the software interrupt vector number used for the Touch Panel function calling.
(Hexadecimal, Default: n=59)
- r** Cancel the resident command.



Note: • Normally parameter specification is not necessary. (Use the default).

At startup, the following message will appear on the screen.

```
Keyboard Emulator KEYEM_PL.EXE Version X.XX
Copyright (C) 2000 Digital Electronics Corporation
Stay resident.
```

After startup, these commands reside in memory.

■ Functions

KEYEM_PL.EXE includes the following functions and perform the function calls using the software interrupt (INT 16h).

<INT16h Function List>

Function Code	Contents
00h	Key input data read
01h	Key input data check
02h	Shift status read
05h	Keyboard data write
10h	Key input data read (AX keyboard)
11h	Key input data check (AX keyboard)
12h	Shift status read (AX keyboard)
A000h	Keyboard Emulation startup
A001h	Keyboard Emulation interrupt
FE00h	Resident Check

The following data details each function.

Function 00h	Key input data read
---------------------	----------------------------

Reads the input key data.

Input	AH= 00hA
Output	AL= Primary code (character code) AH= Secondary code (scan code)

Function 01h	Key input data check
---------------------	-----------------------------

Detects the data in the keyboard buffer.

Input	AH= 01h
Output	ZF= 0: Readable data in the keyboard buffer. 1: No readable data in the keyboard buffer. AL= Primary code (character code) AH= Secondary code (scan code)

Function 02h	Shift status read
---------------------	--------------------------

Returns the status information of the special keys.

Input	AH= 02h
Output	AL= Shift status Bit 7: Insert key Bit 6: Caps Lock key Bit 5: Num Lock key Bit 4: Scroll Lock key Bit 3: Alt key Bit 2: Ctrl key Bit 1: Shift Left key Bit 0: Shift Right key

Function 05h	The key-in data writing
---------------------	--------------------------------

Writes the data for the CX register settings to the keyboard buffer as input from the keyboard

Input	AH= 05h CL= Primary code (character code) CH= Secondary code (scan code)
Output	AL= Status 0: Successfully completed 1: Aborted (No available space in the keyboard buffer.)

Function 10h	Key input data read (AX keyboard)
---------------------	--

Reads the input key data. (The function is associated with the AX keyboard.)

Input AH= 10h

Output AL= Primary code (character code)
 AH= Secondary code (scan code)

Function 11h	Key input data check. (AX keyboard)
---------------------	--

Detects the data in the keyboard buffer.

(Function is associated with the AX keyboard).

Input AH= 11h

Output ZF= 0: Readable data in the keyboard buffer.
 1: No readable data in the keyboard buffer.
 AL= Primary code (character code)
 AH= Secondary code (scan code)

Function 12h	Shift status read (AX keyboard)
---------------------	--

Returns the status of the special keys.

Input AH= 12h

Output AL= Shift status
 Bit 7: Insert key
 Bit 6: Caps Lock key (lamp status)
 Bit 5: Num Lock key (lamp status)
 Bit 4: Scroll Lock key (lamp status)
 Bit 3: Alt key
 Bit 2: Ctrl key
 Bit 1: Shift Left key
 Bit 0: Shift Right key
 AH= Shift status
 Bit 7: Alt+Sys key
 Bit 6: Caps Lock key (key status)
 Bit 5: Num Lock key (key status)
 Bit 4: Scroll Lock key (key status)
 Bit 3: Alt Right key
 Bit 2: Ctrl Right key
 Bit 1: Shift Left key
 Bit 0: Shift Right key

Function A000h Key emulation startup

Starts up the key emulation.

Input AX= A000h
 BL= 0: Clears the graphical keyboard
 1: Displays the graphical keyboard in the opposite section to the cursor location.
 2: Displays the graphical keyboard in the top section of the screen
 3: Displays the graphical keyboard in the bottom section of the screen

Output None



With the BL=2 and BL=3 status, the graphical keyboard is not cleared automatically by pressing the [Enter] key. When the cursor makes a vertical movement, the change to the key-icon display position and display scroll check are not performed. In this condition, the ON/OFF status of the key-icon display must be controlled by the application.

Function A001h Key emulation interrupt

Interrupts the key emulation.

Input AX= A001h
 BL= 0: Clears the graphical keyboard
 1: Clears the graphical keyboard (All keys are disabled.)

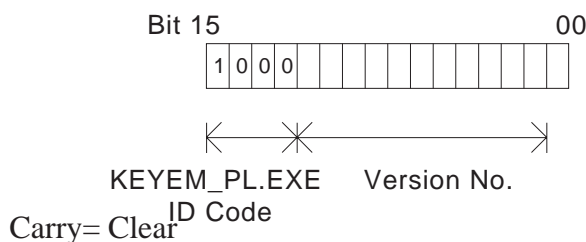
Output None

Function FE00h Resident check

When the KEYEM_PL.EXE application resides in memory, this function returns a fixed message and version information.

Input AX= FE00h

Output AH= 0: Successfully completed
 BL= "Y"
 BH= "C"
 CX= Version number



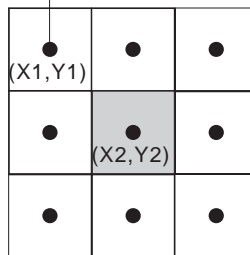
■ **The External Key Definition**

This section explains how to designate a key to arbitrary coordinates with the KEYEM_PL.EXE application.

(Left corner coordinate (X1), Upper corner coordinate (Y1), Right corner coordinate(X2), Lower corner coordinate(Y2))

= the 1st key code [, the 2nd key code [, ...]]; comment line

Touch Panel Coordinates

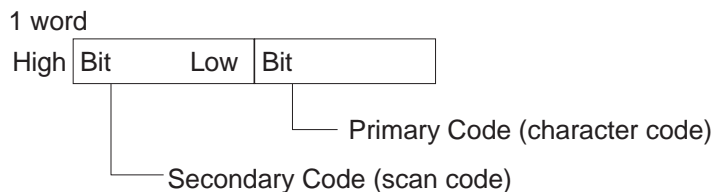


- Designate the values with the touch panel coordinate range in PL-386 compatibility mode (0,1) ~ (30,25).
- If the designated coordinates overlap, the one defined first overrides the other.

If more than two key codes are defined, they operate as if the input occurs in sequence when the coordinated are pressed.

The key code can be defined using the following two methods.

1. Define the key code with 1 word.



* The scan code can be omitted.

2. Define the key code with a character string surrounded by double quotes.



The scan code here is assumed omitted.



- The key codes used in the KEYEM_PL.EXE are as follows.

0x0000: no key input	0x0007: Shift key
0x0001: graphical keyboard ON/OFF	0x0008: Ctrl key
0x0002: reserved	0x0009: Num/Symbol key
0x0003: reserved	0x000A: Japanese/English key
0x0004: Ctrl-Alt-Del	0x000B: ACT key
0x0005: Print key	0x000C: +/- key
0x0006: Pause key	

6.6.5 Backlight Burnout Detection Features Setting Program (BLSET.EXE)

This application detects if the backlight has burned out. This program is stored in [Utility] folder of the PL-5900 Series User Manual & Driver CD. Copy to FD or PL's hard disk and then execute this program with DOS.

■ Start-up

BLSET ON [↵] or BLSET OFF [↵]

ON means the backlight burnout detection is enabled.

OFF means detection is disabled.

For detection setting details, refer to *Appendix 2 RAS Feature*.

Chapter

7 Maintenance and Inspection

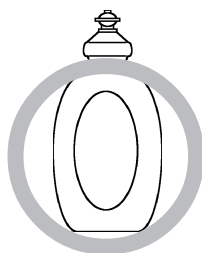
7-1 Regular Cleaning

7-2 Replacing the Backlight

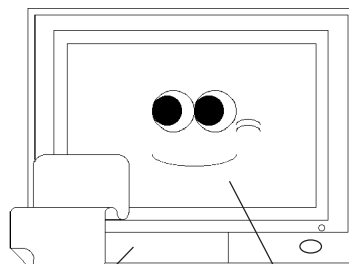
7-3 Periodic Maintenance Points

7.1 Regular Cleaning

7.1.1 Cleaning the Display

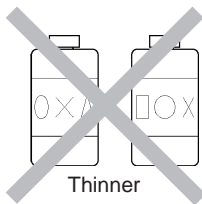


Neutral detergent

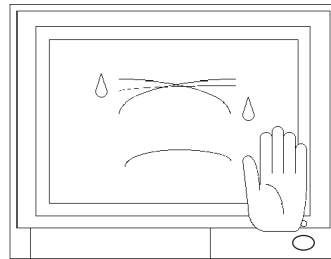


Maintenance panel Display

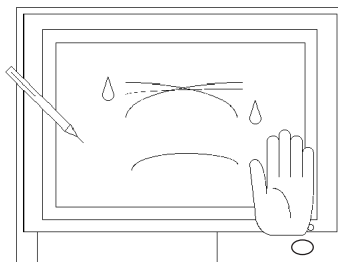
When the display surface or frame become dirty, use a soft cloth moistened with neutral detergent to wipe away any dust or stains.



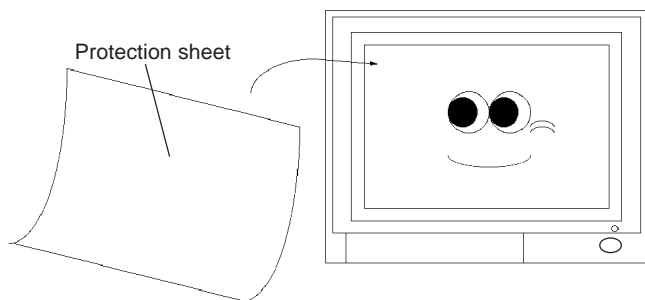
Thinner
Organic solvent
Strong acid



Do not clean the unit with thinner, organic solvents, or strong acids.



Do not use sharp or hard objects, such as a mechanical pencil or screwdriver, to push on the display. This could damage the unit.



Protection sheet

Use the screen protection sheet when using the PL in extremely dirty or dusty areas.

7.1.2 Replacing the Installation Gasket

The installation gasket protects the PL and improves its water resistance. For instructions on installing the PL's gasket.

Reference 4-2 *Installing the PL*



A gasket which has been used for a long period of time may have scratches or dirt on it, and could have lost much of its water resistance. Be sure to change the gasket periodically (or when scratches or dirt become visible).

7.2 Replacing the Backlight

The PL's backlight can be changed after it wears out. Follow the steps explained below.



Please use the following table to identify which backlight model number to use when ordering your backlight.

PL Type	Backlight Type
PL-5900T	GP577RT-BL00-MS
PL-5901T	

WARNINGS

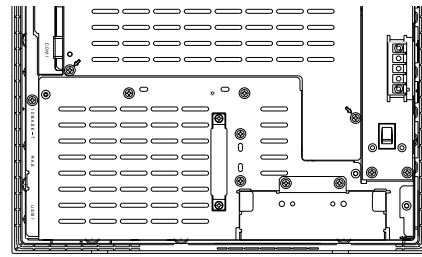
- **Whenever changing the backlight, be sure the PL's power cord has been disconnected and that the unit is cooled down.**
- **When the PL's power cord is connected and the PL is ON, high voltage runs through the wires in the backlight area—*do not touch them!***
- **When the PL's power has just been turned OFF, the backlight area is still very hot! Be sure to wear gloves to prevent being burned.**
- **Do not try to replace the backlight while the PL is installed in a cabinet or panel. Remove the PL first, then begin the backlight replacement procedures.**
- **Be careful when handling the backlight, since it is made of glass and very fragile.**

Follow the steps given below to change the PL's backlight. Be sure to wear cotton gloves when performing this work to prevent burns.

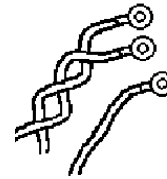
- 1) Unplug the PL's power cord from the main power supply and then disconnect the PL power cord terminals from the PL's power terminal block.



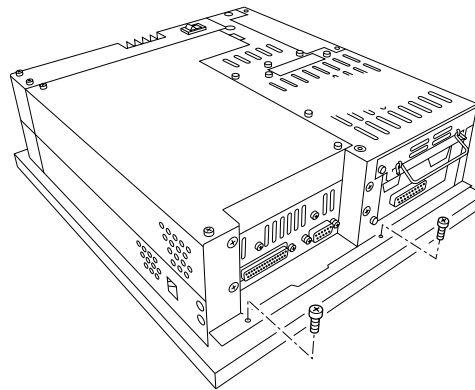
Be sure to perform the backlight changeover on a flat, level surface. This will prevent damage to the PL unit and the accidental cutting of any of its power cord terminals.



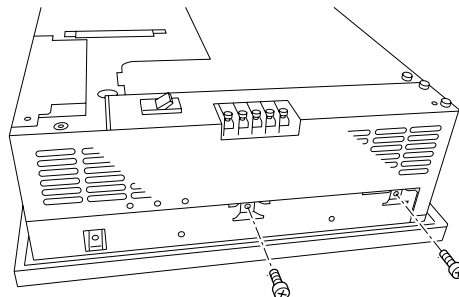
Power Cord



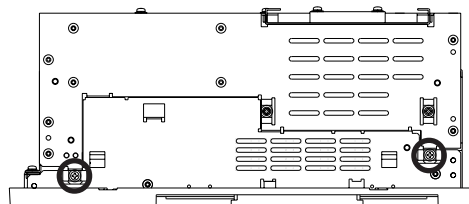
- 2) Remove the power cord.
- 3) Unscrew and remove the two attachment screws from the right side of the PL.



- 4) Unscrew and remove the two attachment screws from the left side of the PL.

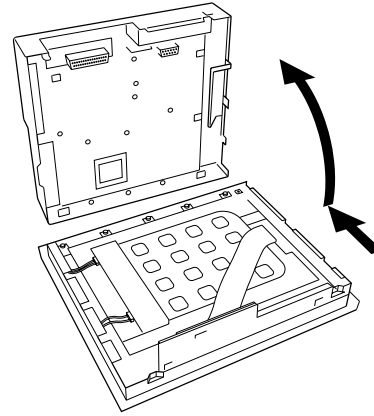


- 5) Unscrew and remove the two attachment screws from the bottom of the PL.



Chapter 7 - Maintenance and Inspection

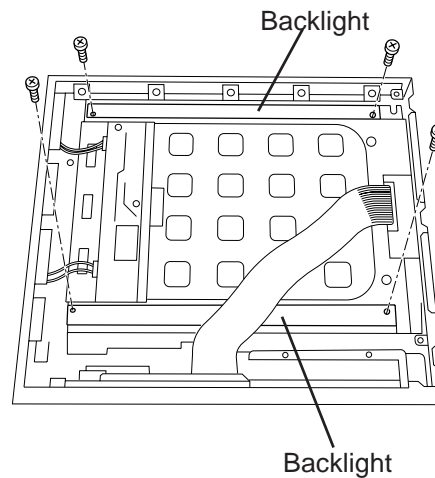
- 6) Slide the rear cover to the top side.
- 7) Lift up the rear face and remove it from the front face.



- 8) Unscrew and remove the four attachment screws securing the backlight.

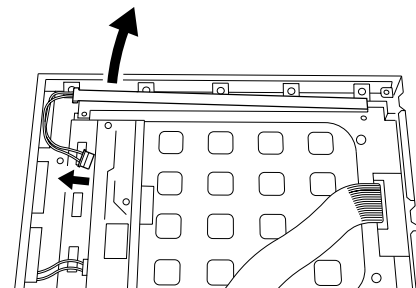


- **Use a “0” sized Phillips screwdriver to remove the backlight attachment screws.**
- **Be careful not to lose the screws.**
- **Be careful when removing the screws so that they do not fall inside the PL, since they may cause the unit to malfunction.**



- 9) Remove the connector from the inverter board.

- 10) Lift up the backlight and slowly pull the unit out of the PL.



- 11) Insert the replacement backlight by reversing the removal procedure. Secure the replacement backlight with the screws and insert the connector to the inverter board.



- ***The PL's backlights are installed at the top and bottom of the LCD panel. Whenever you change a backlight, be sure to change both backlights.***
- ***Be sure to insert the connector to the inverter board and push it until its rear connector is securely connected, or it may cause damage.***

12) Replace the four attachment screws removed in step 8).

13) Reattach the PL rear unit to the PL front unit, using the PL's guide slots, i.e. reverse the procedure use to separate them.

Be careful not to catch or cut any of the PL's internal cables while reattaching the two units.

14) Replace the six attachment screws removed in step 3), 4) and 5).

7.3 Periodic Maintenance Points

Check the PL periodically to ensure it is in good working condition.

Ambient environment check

- Is the ambient temperature within the specified range?
(0°C to 45°C - without HDD unit, 5°C to 45°C - with HDD unit)
- Is the ambient humidity within the specified range (10%RH to 85 %RH) ?
- Is the atmosphere free of corrosive gas ?

Electrical specifications check

- Is the voltage adequate (AC85V to AC132V, 50/60 Hz or DC19.2V to DC28.8V)?

Installation check points

- Is the connection cable firmly connected (not loose) ?
- Are any bolts or screws loose ?
- Are there any flaws or tears in the moisture resistant gasket ?

Display check

- Is the display bright enough ?



When the PL's Standard Display's backlight needs to be replaced, please contact your local PL distributor.



Appendices

- A-1 Hardware Configuration
- A-2 RAS Feature
- A-3 System Monitor
- A-4 Serial Communication
- A-5 Touch Panel Handler Program
- A-6 BIOS List
- A-8 System Monitor/RAS Feature API-DLL

A.1 Hardware Configuration

The following data explains the design of the I/O Map, Memory Map and Interrupt Map, as well as additional hardware design items including the RAS feature.

A.1.1 I/O Map

Address	AT System Device	System Device
0000H - 001FH	DMA controller (8237)	
0020H - 003FH	Interrupt controller (8259A)	
0040H - 005FH	System timer (8254)	
0060H - 006FH	Keyboard (H8/3332)	
0070H - 007FH	Real-time clock, NMI mask	
0080H - 009FH	DMA bank register	
00A0H - 00BFH	Interrupt controller 2 (8259A)	
00C0H - 00DFH	DMA controller 2 (8237)	
00F0H - 00FFH	Numeric data processor	
016CH - 016FH	Used by Main Board	
01F0H - 01FFH	Hard disk (IDE)	
0200H - 0207H	Game I/O *1	
0295H - 0296H	Used by Main Board	
02E8H - 02EFH	Serial Port 4 (COM4): Touch Panel	Touch Panel
02F8H - 02FFH	Serial port 2 (COM2) : General Use	
0378H - 037FH	Parallel port 2 (LPT2)	
03B0H - 03BBH	Video controller (VGA)	
03BCH - 03BFH	Parallel port 1 (LPT1) Printer	
03C0H - 03DFH	Video controller (VGA)	
03E8H - 03EFH	Serial port 3 (COM3): Reserved	
03F0H - 03F7H	Floppy disk controller	
03F8H - 03FFH	Serial port 1 (COM1) :General Use	

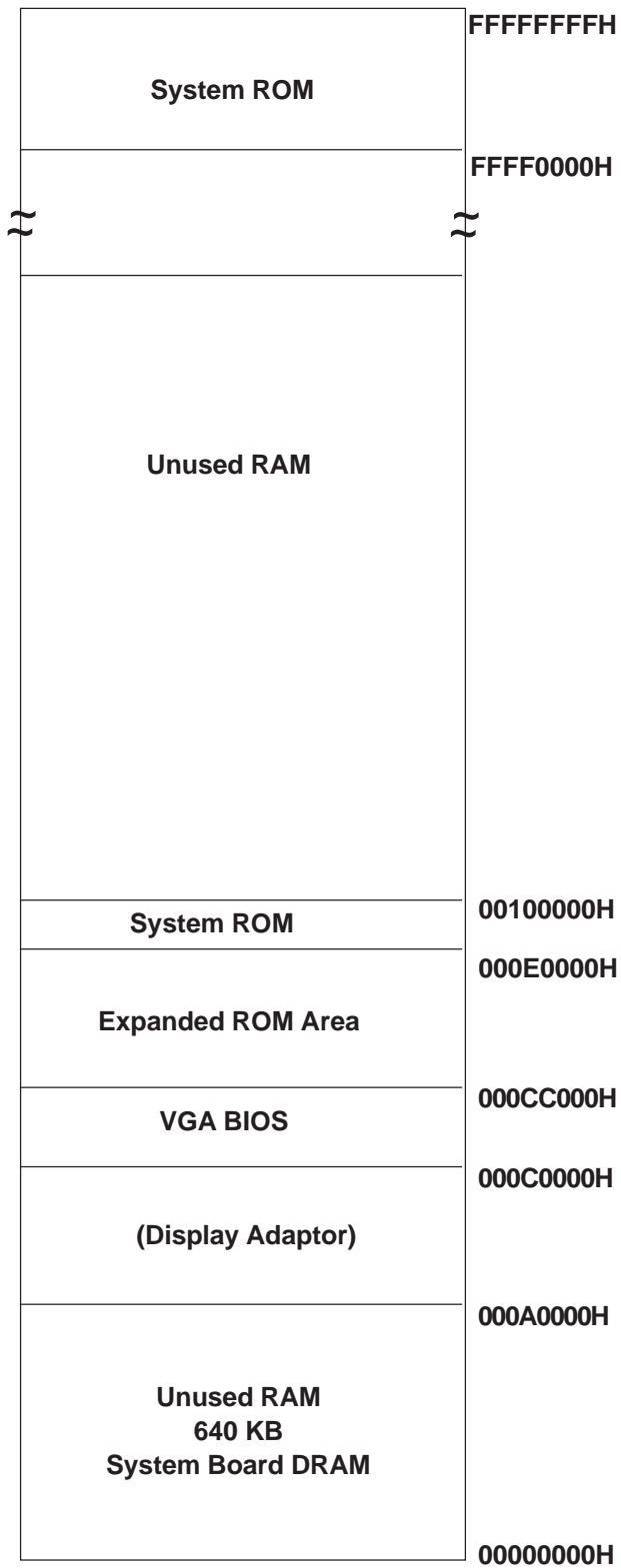
*1 This device is not supported by the system, but is reserved as standard.



Depending on the arrangement of any connected Plug-and-Play devices, these features perform different functions/actions.

Appendices

A.1.2 Memory Map



A.1.3 Interrupt Map

■ Hardware Interrupt List

	Description
NMI 0	Parity Error or I/O Channel Check
IRQ 0	Timer (in the Chipset)
IRQ 1	Keyboard
IRQ 2	Cascade from Controller 2
IRQ 3	Serial Port 2 (COM2): General Use Port
IRQ 4	Serial Port 1 (COM1): General Use Port
IRQ 5	Serial Bus Controller
IRQ 6	Floppy Disk Controller
IRQ 7	Parallel Port 1 (LPT1) : Printer Port
IRQ 8	Real Time Clock
IRQ 9	ACPI Controller
IRQ 10	Serial Port 4 (COM4): Touch Panel (for Standard monitor)
IRQ 11	Serial Port 3 (COM3): General Use Port
IRQ 12	Network Controller
IRQ 13	Numeric Data Processor
IRQ 14	Hard Disk (IDE)
IRQ 15	Available for users



Depending on the arrangement of any connected Plug-and-Play devices, these features perform different functions/actions.

■ DMA Channel List

	Description	
DMA 0		For 8-bit transmission
DMA 1		
DMA 2	Floppy disk controller	
DMA 3		For 16-bit transmission
DMA 4	Cascade to controller 1	
DMA 5		
DMA 6		
DMA 7		

A.2 RAS Feature

A.2.1 PL's RAS Features

RAS, which stands for Reliability, Availability and Serviceability, is a device-level monitoring function that provides a variety of features to improve the reliability of your PL system.

Though the standard set of RAS features used will vary depending on the devices used, the following features are used to provide Alarm Monitoring and External Input Signal support.

Alarm Monitoring	Power Voltage Alarm Cooling Fan Alarm Internal Temperature Alarm Watchdog Timer Time Up Mirror Disk Alarm ^{*1} Backlight Burnout Detection Touch Panel Alarm
External Input Signal	Standard Signal Input (DIN 2 bit) Remote Reset Input ^{*2}

Also, when either the one of the above mentioned alarms occurs, or an external signal input is received, the following types of alarm processing output signals and features are supported.

External Output Signal	Standard Signal Output (DOUT 1 bit) Alarm Output (1 point) Lamp Output (1 point)
Types of Processing (all units)	LED Indicator (3-state display – 1 point) Pop-up Message Output Buzzer Output System Shutdown

Furthermore, using the PL's System Monitor feature ^{*3} (included in the PL's software utility disk), allows the easy setting and control (Enable/Disable) of the aforementioned Alarm Monitor and External Input Signals.

Last, the system monitor feature's use of an Application Link Library (API-DLL) allows it to also be used with other applications.

**1 When a Mirror Disk Alarm occurs and the standard RAS feature settings are used, the alarm output is limited to the Mirror Disk unit's LED indicator. (Alternately flashing orange and green)*

**2 The remote reset feature's input can be either enabled or disabled, the alarm output setting cannot be set to trigger a forced system reset.*

**3 For System Monitor Feature details, refer to the accompanying Driver and Utility disk.*

A.2.2 RAS Feature Details

■ Alarm Monitoring

◆ Power Voltage Alarm

Monitors the condition of the PL's internal and CPU power.

◆ Cooling Fan Alarm

Monitors the condition of the PL's internal power and CPU cooling fans.

◆ Internal Temperature Alarm

Monitors the PL's internal and CPU vicinity temperatures.

The degree of monitoring (3 levels) and the enabling or disabling of the above three items is performed via the System Setup Area's settings.

For detailed information about the monitoring level settings,

Reference *5.2.12 PC Health Status*

This utility can also be used to enable or disable the above mentioned features, as well as designate what type of processing is to be performed.

◆ Watchdog Timer Time Up

This feature alternately writes Time Up Count values from the CPU to the RAS feature's special programmable timer and then periodically clears them, which provides a means of monitoring the CPU's performance. If the clearing of this count value is stopped, the timer will overflow and an alarm will be detected. The System Monitor utility can be used to enable or disable this feature, as well as designate what type of processing is to be performed.

Reference *5.2.7 Power Management Setup*

◆ Mirror Disk Alarm

Whenever a disk crash, or other alarm event occurs to the optional Mirror Disk unit, this unit's LED indicator will flash (either orange or green) to indicate there is a problem.

This unit's error detection occurs independently of and cannot be set by the RAS feature.

◆ Backlight Burnout Detection

This feature allows you to detect when the PL's backlight burns out.

When this feature is enabled and a backlight burnout occurs, touch operation will be disabled and PL's front panel LED indicator will blink. The factory setting is "Enabled" and is recommended to prevent accidental touch panel operation. This setting is controlled via the MS-DOS utility BLSET.EXE.

◆ Touch Panel Alarm

This feature detects a Touch Panel alarm. When this alarm occurs, the PL unit's front panel LED indicator will blink.

Appendices

■ External Input Signal

The PL's RAS interface connector uses the following input signals.

Standard Signal Input (DIN)

This standard digital input is used for alarm detection in external devices. The input signal uses two bits.

The System Monitor utility can be used to enable or disable this feature, as well as designate what type of processing is to be performed once a signal is received.

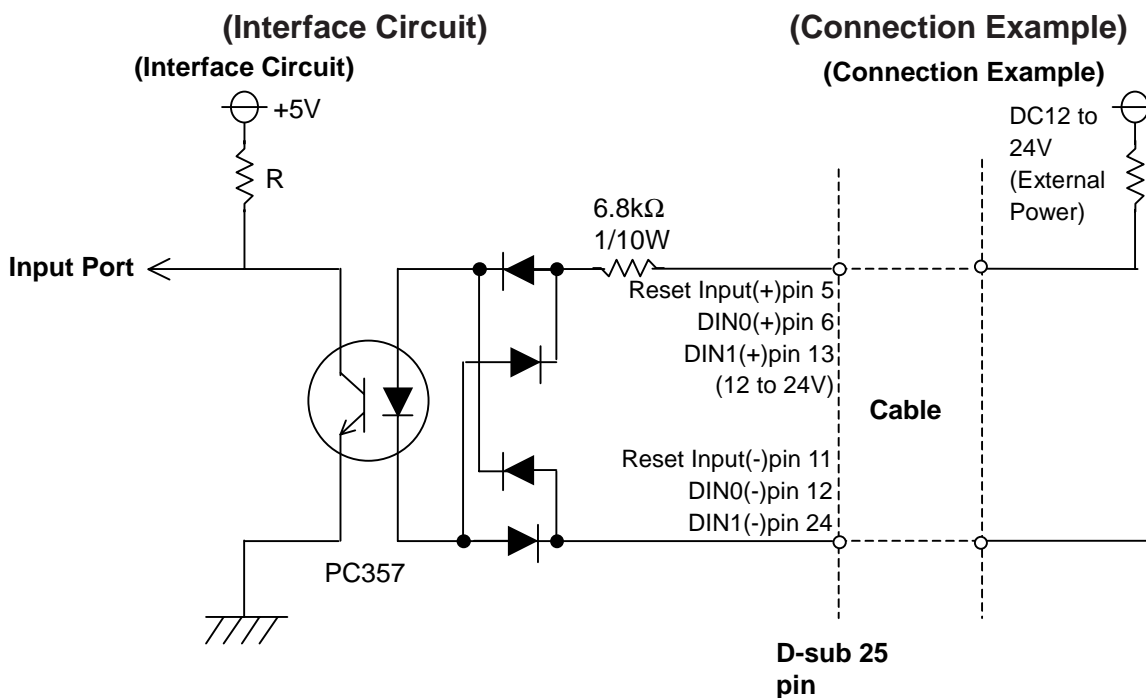
Remote Reset Input

This is the reset signal sent from an external device to the PL. When this signal is enabled, a forced reset of the PL is performed.

The System Monitor utility can be used to enable or disable this feature

External Input Signal (for both DIN and Remote Reset Input)

- External Power DC12V to DC24V connections are possible
- Input Protection Protection Diode
- Isolation Used (photo-isolation)



- The Power supply used for sink/source type input can use either polar or non-polar connection.

- For connection pin location details,

Reference 2.3.5 RAS Interface

External Output Signal

The PL's RAS interface connector uses the following output signals.

General Purpose Signal Output (DOUT)

This general purpose digital output signal provides system condition information to external devices.

The System Monitor's API-DLL are used by applications to control this signal.

Alarm Output (1 point)

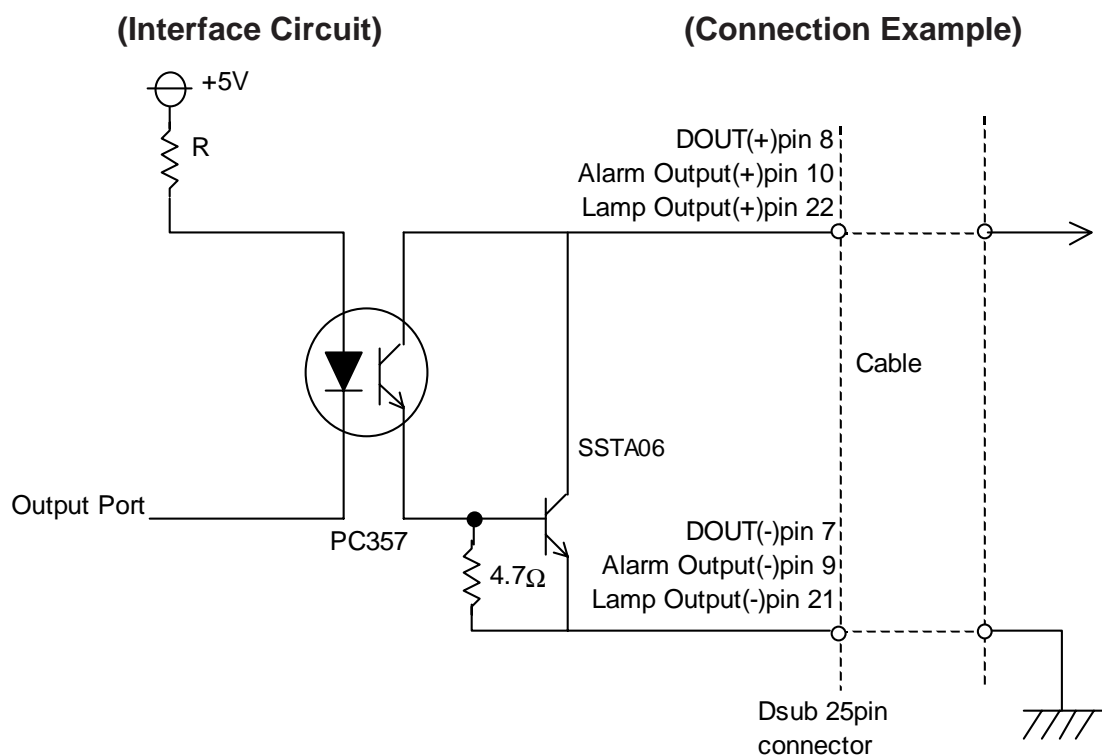
Lamp Output (1 point)

The above mentioned general purpose digital output signals provide system condition information to external devices.

The System Monitor utility can be used to enable or disable any of these output signals.

External Output Signal (used for DOUT, Alarm Output, Lamp Output)

- Output Specification DC24V 100mA (MAX)
- Isolation Used (photo-isolation)



For connection pin location details,

Reference 2.3.5 RAS Interface

Appendices

■ Types of Processing (all units)

The PL provides system condition information via the following methods.

LED Indicator (3-state display – 1 point)

In addition to indicating if the unit's power is ON or OFF, the 3-state LED indicator (power lamp) provides the following system condition information.

Color	System Condition	Output Created
Green	Normal Operation (Power ON)	None
Orange	Touch Panel alarm	None
Flashing Red/Green	Mirror Disk and Backlight alarms have occurred	Any TOOL settings are enabled.
Flashing Orange/ Green	Mirror Disk Alarm has occurred	None
Flashing Orange/Red	Backlight Alarm has occurred	Any TOOL settings are enabled.

Pop-up Message Output

This feature uses the Windows® system's pop-up message feature to indicate that an alarm has occurred.

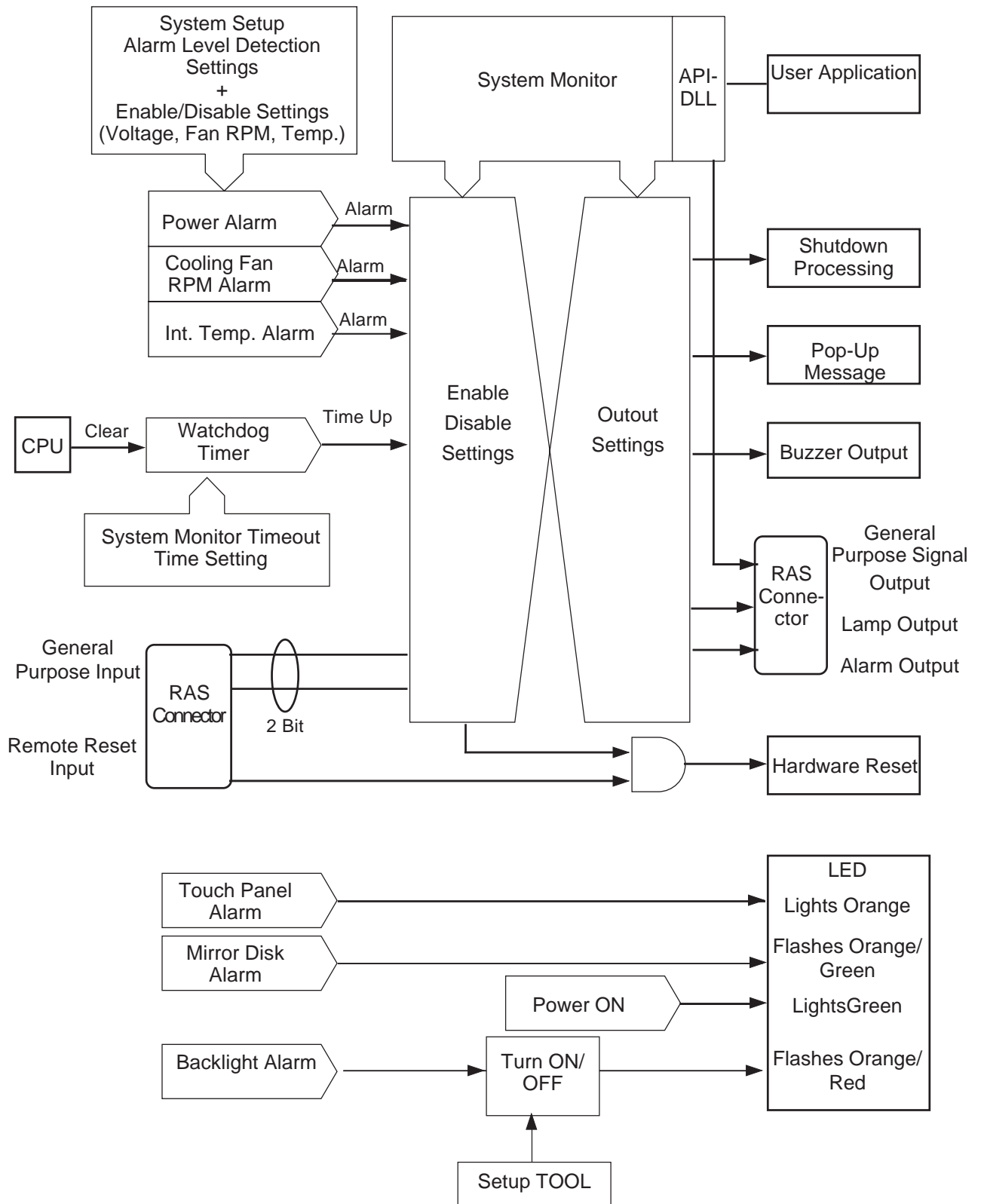
Buzzer Output

This feature uses the PL's internal speaker to indicate the system's condition.

System Shutdown

This feature shuts down the PL's OS (Windows® 95/ Windows® 98 Second Edition). The System Monitor utility can be used to enable or disable this feature.

A.2.3 RAS Feature Overview

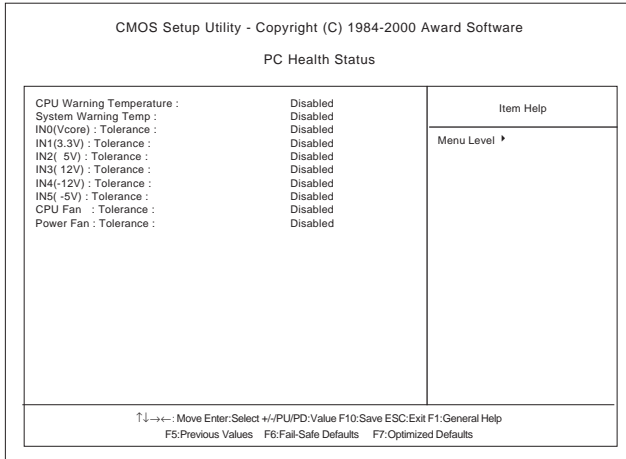


A.3 System Monitor

A.3.1 Setup Procedure

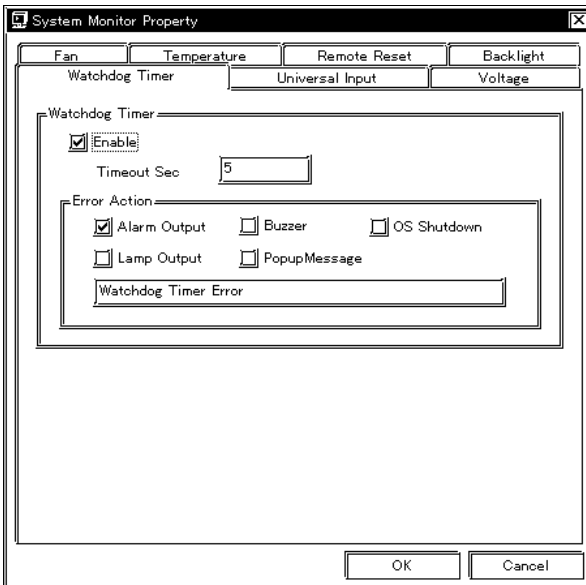
Follow the steps shown here to complete the System Monitor/RAS setup.

■ System Setup Screen Settings

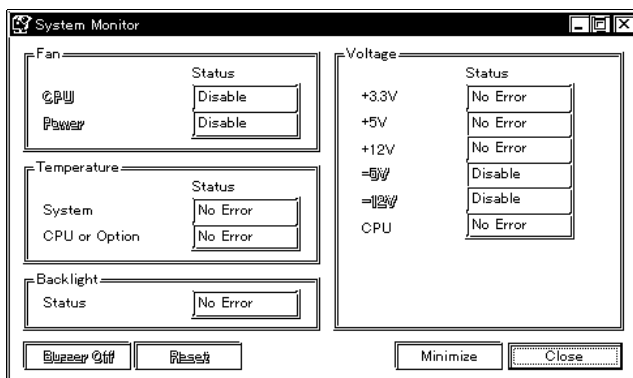


In the System Setup area's PC Health Status menu, enable or disable each feature according to your system needs.

■ System Monitor Property Settings



1) Start the PL unit's OS and click on the [Start] -> [Program] -> [System Monitor]->[System Monitor Property] screen. In this screen enter the System Monitor/RAS Event settings for each feature/tab. Click on the [OK] button and the program will automatically close.



2) Restart the [System Monitor] utility, click on the [System Monitor] button and you can begin to monitor PL activity.

A.3.2 System Monitor Property Settings (PL_Wps.exe)

The following chart shows the features available when any of the monitoring value ranges set in the [PC Health Status] menu is exceeded.

O: Can be set X: Cannot be set

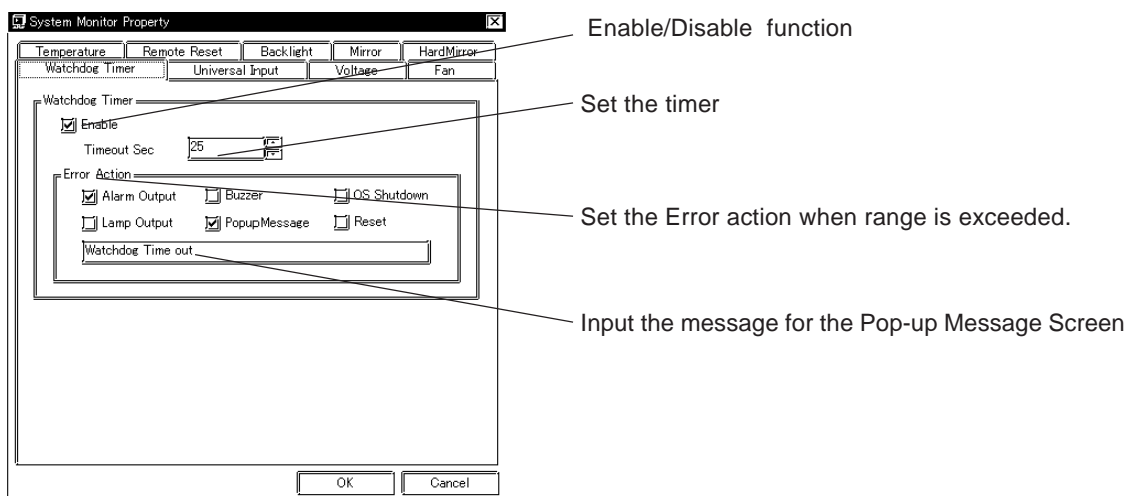
	Alarm Output	Lamp Output	Buzzer	Popup Message	OS Shutdown	Reset
Watchdog Timer	O	O	O	O	O	X
Universal Input	O	O	O	O	O	X
Voltage	O	O	O	O	O	X
Fun	O	O	O	O	O	X
Temperature	O	O	O	O	O	X
Remote Reset *1	X	X	X	X	X	O
Backlight	O	O	O	O	O	X

*1 When setting Enable on Remote Reset, the same action as the Reset occurs. Performing reset without first shutting down the PL unit's OS may destroy the PL unit's file data system.

Each of the above items performs the following operation.

Item	Operation
Alarm Output	RAS Interface Alarm Output (#9 to #10) signal is output.
Lamp Output	RAS Interface Alarm Output (#21 to #22) signal is output.
Buzzer	Buzzer sound is output as an alarm notification. (except for when the OS Shutdown feature is checked)
Popup Message	Error message appears as a Pop-Up Message Screen (on the PL unit's screen)
OS Shutdown	Shuts down the PL unit's OS. This can be set to either display a shutdown confirmation message, or perform a forced OS shutdown. Default is display a shutdown confirmation message.
Reset	Resets the PL unit by performing a forced shutdown.
Enable	Allows setting of monitoring items.

The System Monitor Property Screen is as shown below.

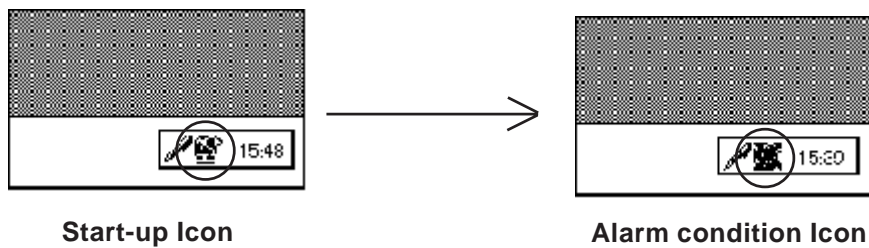


Use the PL unit's BIOS screen to enable/disable only Watchdog Timer features or to set the Timer.

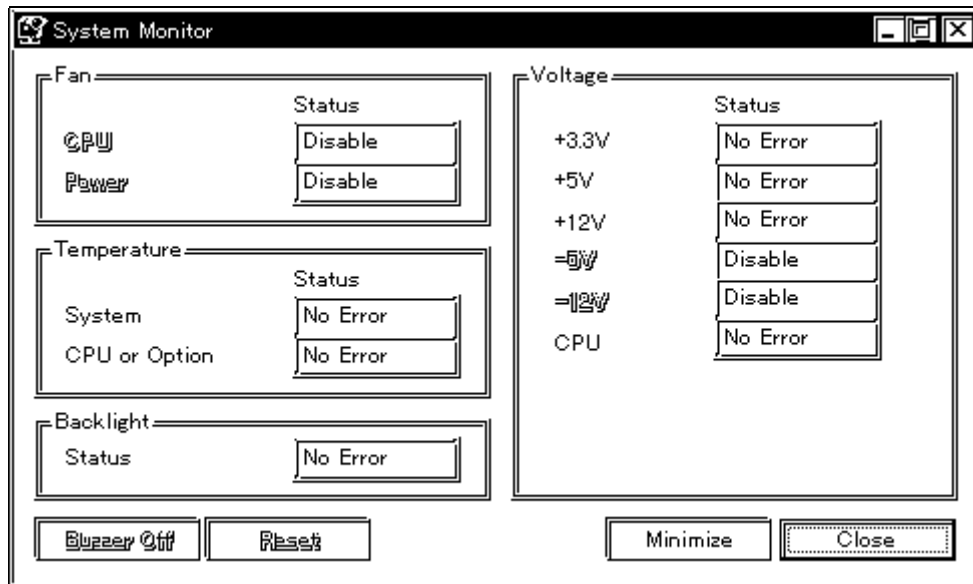
A.3.3 System Monitor Operation (PL-Smon.exe)

As soon as the PL unit's OS starts up, instead of showing the System Monitor Dialog box, the Alarm Monitoring icon appears in the System Tray. (see below) Usually, when a System Monitor dialog box appears, the user checks the current condition and then left-clicks the mouse on the system tray icon to call up the System Monitor screen.

When an alarm is detected, the actions set in the System Monitor Property screen are performed, and an "X" appears over the System Tray icon. When this occurs, double-click on the icon to view the alarm contents.



The System Monitor screen is as shown below.



System Monitor Screen

In addition to the "Buzzer Off", "Reset", "Minimize", and "Close" buttons, the System Monitor screen contains the following features.

Button Name	Meaning
Buzzer Off	Stops the Alarm buzzer.
Reset	Resets alarm operation or System Monitor's internal alarm hold condition.
Minimize	Minimizes the System Monitor icon.
Close	Quits the System Monitor software.

Within the System Monitor screen is are the three words "No Error", "Error", and "Disable". They show the current status of each of the monitoring items, such as the Fan, Voltage, Temperature, Backlight, Software Mirroring Disk, and Hardware Mirroring Disk.

Status Name	Meaning
No Error	Normal operation
Error	Alarm
Disable	Not monitored

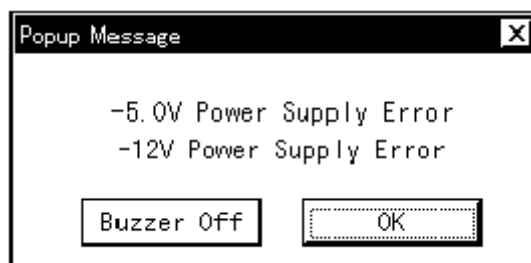
Whenever an alarm occurs for one of the monitored items, or when input (Universal Input) is detected from an external source, The operation designated in the System Monitor Property screen (Error Action) is performed.

Reference 3-2 System Monitor Property Settings

Each error action, once an error or input is detected, is performed only once.

If +3.3V and +5.0V are monitored and a pop-up message is designated for the error action, when the +3.3V alarm is detected, a pop-up message will appear. Click on [OK] and the box is closed. Then, when a +5.0V alarm occurs, the +5.0V pop-up message will appear.

The pop-up message provides information about the type of error and the error contents. When this message box's [Buzzer Off] button is clicked, the buzzer sound will stop. Clicking on [OK] will close the box.



Pop-up Message Box

Once an alarm has occurred, the System Monitor will continue to remain in the "Alarm" state. (i.e. the alarm detected "X" will continue to be displayed on the System Tray icon. To release this condition, click on the System Monitor dialog box's [Rset] button. Or, turn the PL unit's power OFF, find and solve the problem, and turn the PL unit's power ON again.

A.3.4 Error Messages

The following error messages occur via the settings in the System Monitor, and the System Monitor Property dialog boxes.

■ System Monitor

◆ Alarm Pop-up Messages

When the Pop-up messages related to the Error Action are enabled, the following error messages will appear.

Error Type	Message
CPU Voltage	"CPU Power Supply Error"
+3.3V	" +3.3V Power Supply Error"
+5.0V	" +5V Power Supply Error"
+12V	" +12V Power Supply Error"
-12V	" -12V Power Supply Error"
-5V	" -5V Power Supply Error"
Power FAN	"Power FAN Error"
CPU FAN	"CPU or OPTION FAN Error"
System Temperature	"System Temperature Error"
CPU or Option Temperature	"CPU Temperature Error"
Universal Input 0	"Universal Input 0"
Universal Input 1	"Universal Input 1"
Watchdog	"Watch Dog Timer Error"
Software Mirroring	"A Mirror disk error occurred"
Hardware Mirroring	"A Mirror disk error occurred"
Backlight	"Back Light Blowout Error"

Driver Error

"The system monitor driver not found."

"Install the latest driver."

Driver Version Error

"The old system monitor driver version."

"Update the driver."

Overlapping Startup message

"System monitor has started. "

"Terminate the system monitor in starting."

Shutdown Confirmation

"The system monitor is terminated."

"Are you sure?"

■ System Monitor Property Screen

◆ Overlapping startup message

"System Monitor Property has started."

"Terminate the system monitor property in starting."

◆ Shutdown Confirmation message

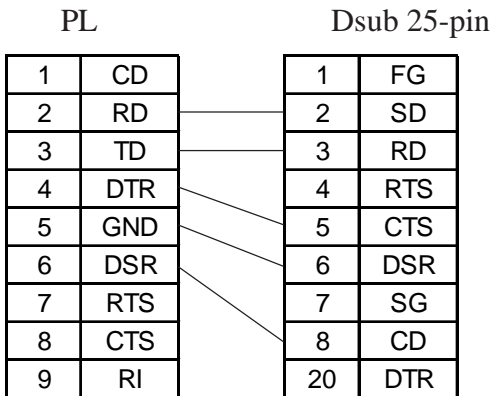
"Save Changes to the registry?"

A.4 Serial Communication

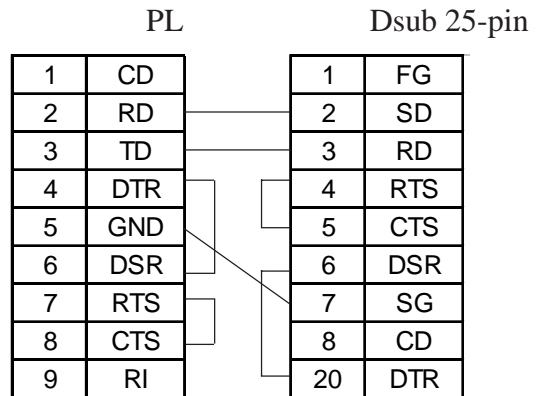
This section explains how to perform serial communications with the PL.

■ RS-232C (COM1/COM2) Cable Connections

<Example 1>



<Example 2>



■ Sample output program

Below is a sample program for sending 1 character from the RS-232C connector.



Note:

Because the PL uses an AT-compatible BIOS, the serial communication BIOS (INT 14 h) does not support communications by interrupt. Therefore, install a reception interrupt function in the application.

```
#include <stdio.h>
#include <dos.h>

union      REGS    ir,or;

main( ){
    ir.h.ah  =  0x00 ;    /* Initialization */
    ir.h.al  =  0xe3 ;    /* 9600bps,8bit,NONE,1stop */
    ir.x.dx  =  0 ;      /* COM1 */
    int86 (0x14,&ir,&or) ;

    ir.h.ah  =  0x01 ;    /* 1 character output */
    ir.h.al  =  0x32 ;    /* '2' */
    ir.x.dx  =  0 ;      /* COM1 */
    int86 (0x14,&ir,&or) ;
}
```



Note:

Changing commands or parameters also changes function (mode setting, data reception, etc.).

A.5 Touch Panel Handler

The following sample program was created with the ATPH59.EXE (Touch Panel Handler) application.

Sample Program made with the Touch Panel Handler

Compiler Method
cl sample.c

If not adding GRAPHICS.LIB to SLIBCE.LIB, add **/link graphics.lib**.

F1	F2	F3	F4	F5	F6	F7	F8
-----------	-----------	-----------	-----------	-----------	-----------	-----------	-----------

Please press one of the touch panel keys.

Pressing either "Z" or on the square shown to the right, will quit this program.

"Quit" key

1	2	3	4	5	6	7	8	9	0	.	+	BS	ENT
---	---	---	---	---	---	---	---	---	---	---	---	----	-----

Function Key area

This sample program is designed to perform processing based on the touch panel input received from either the upper or lower function keys, or the screen's designated Quit key area.

```

/* This sample program is used by the ATPH59.EXE touch panel handler.
 * Input can be performed by pressing on designated areas of the screen.
 * Before starting this program, be sure to start the ATPH59.EXE program
 */

```

```

#include <stdlib.h>
#include <stdio.h>
#include <dos.h>
#include <graph.h>
#include <conio.h>

```

Appendices

```
#define ATPH_SYS_CALL 0x59          /* ATPH System Call */
#define KEYBOARD_BIOS 0x16         /* Key Board BIOS */
#define WRITE_DATA 0x05           /* Key Board Input */
#define TP_IN_NOWAIT 0x8101       /* ATPH Input */
#define TP_CONDITION 0x8500       /*TouchPanelCondition*/

#define ZKEYCODE 0x7a             /* "Z" keycode */
#define DATA8254 0x40            /* Timer Data */
#define CTRL8254 0x43            /* Timer Controller */
#define CTRL8042 0x61            /* Buzzer ON/OFF */
#define BEEP_TIME 20000          /* Beep's ON period */

#define FALSE 0
#define TRUE 1

struct function_coordinate
{
    short sx;                    /* X coordinate (left,upper) */
    short sy;                    /* Y coordinate (left,lower) */
    short ex;                    /* X coordinate (Right Upper) */
    short ey;                    /* Y coordinate (Right,lower) */
    short key;                   /* Key Code */
}

tpcd [] =
{
    480, 40, 559, 80, 0x2c7a,    /* Square's display */

    1, -40, 79, -1, 0x1e61,     /* Function Sheet F1 */
    80, -40, 159, -1, 0x3062,   /* Function Sheet F2 */
    160, -40, 239, -1, 0x2e63,  /* Function Sheet F3 */
    240, -40, 319, -1, 0x2064,   /* Function Sheet F4 */
    320, -40, 399, -1, 0x1265,  /* Function Sheet F5 */
    400, -40, 479, -1, 0x2166,  /* Function Sheet F6 */
    480, -40, 559, -1, 0x2267,  /* Function Sheet F7 */
    560, -40, 639, -1, 0x2368,  /* Function Sheet F8 */
}
```

```

1, 479, 39, 519, 0x0231, /* Function Sheet 1 */
40, 479, 79, 519, 0x0332, /* Function Sheet 2 */
80, 479, 119, 519, 0x0433, /* Function Sheet 3 */
120, 479, 159, 519, 0x0534, /* Function Sheet 4 */
160, 479, 199, 519, 0x0635, /* Function Sheet 5 */
200, 479, 239, 519, 0x0736, /* Function Sheet 6 */
240, 479, 279, 519, 0x0837, /* Function Sheet 7 */
280, 479, 319, 519, 0x0938, /* Function Sheet 8 */
320, 479, 359, 519, 0x0a39, /* Function Sheet 9 */
360, 479, 399, 519, 0x0b30, /* Function Sheet 0 */
400, 479, 439, 519, 0x342e, /* Function Sheet . */
440, 479, 479, 519, 0x272b, /* Function Sheet + */
480, 479, 559, 519, 0x0e08, /* Function Sheet BS */
560, 479, 639, 519, 0x1c0d, /* Function Sheet ENT*/
0, 0, 0, 0, 0
};

```

struct key_code

```

{
    char keyname[3]; /* Touch Key Name */
    short keycode; /* Key Code */
}

kbcd[]=
{
    " Z ", 0x7a, /*Display Square */

    " A ", 0x61, /* Function Sheet F1 */
    " B ", 0x62, /* Function Sheet F2 */
    " C ", 0x63, /* Function Sheet F3 */
    " D ", 0x64, /* Function Sheet F4 */
    " E ", 0x65, /* Function Sheet F5 */
    " F ", 0x66, /* Function Sheet F6 */
    " G ", 0x67, /* Function Sheet F7 */
    " H ", 0x68, /* Function Sheet F8 */

```

Appendices

```
" 1 ", 0x31, /* Function Sheet 1 */
" 2 ", 0x32, /* Function Sheet 2 */
" 3 ", 0x33, /* Function Sheet 3 */
" 4 ", 0x34, /* Function Sheet 4 */
" 5 ", 0x35, /* Function Sheet 5 */
" 6 ", 0x36, /* Function Sheet 6 */
" 7 ", 0x37, /* Function Sheet 7 */
" 8 ", 0x38, /* Function Sheet 8 */
" 9 ", 0x39, /* Function Sheet 9 */
" 0 ", 0x30, /* Function Sheet 0 */
" . ", 0x2e, /* Function Sheet . */
" + ", 0x2b, /* Function Sheet + */
"B S", 0x08, /* Function Sheet BS */
"ENT", 0x0d, /* Function Sheet ENT*/
0, 0
};
```

```
union REGS inregs, outregs;
```

```
/*******/
/* BUZZER ON */
/*******/
void buzzer_on(void)
{
    int timer;

    outp (CTRL8253, 0xb6);
    outp (DATA8253+2, 0x33);
    outp (DATA8253+2, 0x05); /* Length of buzzer sounding */

    outp (CTRL8042, ((inp(CTRL8042) | 0x3) & 0xff));
    /* Buzzer On */

    for (timer = 0 ; timer<BEEP_TIME; timer++);
}
```

```

/*****/
/* BUZZER OFF */
/*****/
void buzzer_off(void)
{
    outp (CTRL8042, (inp(CTRL8042) & 0xfc));    /* Buzzer Off    */
}

/*****/
/* WAIT TOUCHPANEL OFF */
/*****/
void wait_touch_off(void)
{
    while(1)
    {
        inregs.x.ax = TP_CONDITION;
        int86(ATPH_SYS_CALL, &inregs, &outregs);

        if(outregs.h.ah == 3)                /* Touch Panel Off */
        {
            break;
        }

        else if(outregs.h.ah == 0)          /* Touch Panel On  */
        {
            inregs.x.ax = TP_IN_NOWAIT;
            int86(ATPH_SYS_CALL, &inregs, &outregs);
                                                    /* ATPH Function Call*/
        }
    }
}

```

Appendices

```

/*****
/* INPUT TUCHPANEL */
*****/

void touchpanel_on(void)
{
    int i, value;

    inregs.x.ax = TP_IN_NOWAIT;
    int86(ATPH_SYS_CALL, &inregs, &outregs);

    if(outregs.h.ah == 0)                /* Touch Panel On */
    {
        value=FALSE;
        for(i=0; tpcd[i].sx; i++)
        {
            if(                          /* Time in area */
                ((short)outregs.x.dx > tpcd[i].sx)&&
                ((short)outregs.x.bx > tpcd[i].sy)&&
                ((short)outregs.x.dx < tpcd[i].ex)&&
                ((short)outregs.x.bx < tpcd[i].ey))
            {
                inregs.h.ah = WRITE_DATA;
                inregs.x.cx = tpcd[i].key;
                int86(KEYBOARD_BIOS, &inregs, &outregs);

                buzzer_on();
                wait_touch_off();
                buzzer_off();
                value=TRUE;
                break;
            }
        }
    }
}

```



```

        if(value == FALSE)                /* time out (of) area    */
        {
            wait_touch_off();
        }
    }
}

/*****
/* KEYBOARD INPUT*/
*****/
int keyboard_on(void)
{
    int i, ky;

    if(kbhit())
    {
        ky=getch();
        for(i=0; kbcd[i].keycode; i++)
        {
            if(ky == kbcd[i].keycode)
            {
                printf("push key is [%s]. \r",kbcd[i].keyname);
                if(ky == ZKEYCODE)
                {
                    printf(" push [%s]key. finishprogram. \n",kbcd[i].keyname);

                    return(TRUE);        /* Program End    */
                }
                break;
            }
        }
    }
    return(FALSE);
}

```

Appendices

```

/*****/
/* MAIN VARIABLES */
/*****/
void main(void)
{
    if(!(_setvideomode(_VRES16EXCOLOR)))
    {
        printf("error: can't set graphics mode ");

        exit(1);                /* ERROR:Program End */
    }
    _rectangle(_GBORDER,tpcd[0].sx,tpcd[0].sy,tpcd[0].ex,tpcd[0].ey);
                                /* MAKE A SCREEN FRAME */
    printf("please push touchpanel or key \n");
    printf("End program by pressing [ Z ]key or screen. \n");

    while(1)
    {
        touchpanel_on();
        if(keyboard_on())
        {
            _setvideomode(_DEFAULTMODE);    /* RETURN TO VIDEOMODE */
            exit(0);
        }
    }
}

```

A.6 BIOS List

■ INT 5h Display Hard Copy

Operation	Input	Output
Screen hard copy		0050:0000h Print-screen flag 0: Hard copy function unused or completed successfully 1: Now printing -1: Error

■ INT 10h Video BIOS

Operation	Input	Output
Setting video mode	AH = 00h AL = Value of selected mode (Bits 0 to 6), Video RAM clear (Bit 7)	
Setting cursor shape	AH = 01h CG = Cursor start position and display CL = Cursor end position	
Setting cursor position	AH = 02h BH = Page No. DH = Value of specified line DL = Value of specified column	
Reading cursor position	AH = 03h BH = Page No.	CH = Cursor start position and display CL = Cursor end position DH = Current cursor line position DL = Current cursor column position
Reading light pen position	AH = 04h	AH = 0: Light pen switch is off. = 1: Light pen switch is on. BX = X coordinate (0 - 319, 639) CH = Y coordinate (0 - 199) CX = Y coordinate in new graphic mode (0, XXX) DH = Light pen line position in character units DL = Light pen column position in character units
Switching active page	AH = 05h AL = Page No.	
Scrolling up	AH = 06h AL = Number of lines to scroll up BH = Attribute of line to clear CH = Highest line in scroll up range CL = Farthest left column in scroll up range DH = Lowest line in scroll up range DL = Farthest right column in scroll up range	
Scrolling down	AH = 07h AL = Number of lines to scroll down BH = Attribute of line to clear CH = Highest line in scroll down range CL = Farthest left column in scroll down range DH = Lowest line in scroll down range DL = Farthest right column in scroll down range	

Appendices

Operation	Input	Output
Reading character/attribute at cursor position	AH = 08h BH = Page No.	AL = Character code AH = Attribute (In text mode)
Writing character/attribute at cursor position	AH = 09h AL = Character code BH = Page No. BL = Attribute CX = Number of characters to output	
Writing character at cursor position	AH = 0Ah AL = Character code BH = Page No. BL = Attribute (Effective only in graphic mode) CX = Number of characters to output	
Color Pallet Settings		
Setting overscan, background color, and display color brightness	AH = 0Bh BH = 00h BL = Color code	
Setting color group	AH = 0Bh BH = 01h BL = Color group	
Writing point in graphic mode	AH = 0Ch AL = Attribute BH = Page No. CX = X coordinate DX = Y coordinate	
Reading point in graphic mode	AH = 0Dh BH = Page No. CX = X coordinate DX = Y coordinate	AL = Attribute
Writing character/attribute at cursor position and move cursor	AH = 0Eh AL = Character code BL = Attribute (Effective only in graphic mode)	
Reading video data	AH = 0Fh	AH = Number of single lines AL = Video mode BH = Active page No.
Setting pallet register		
Setting pallet register	AH = 10h AL = 00h BH = Pallet code BL = Pallet register No.	
Setting overscan register	AH = 10h AL = 01h BH = Pallet code	
Setting pallet register and overscan register	AH = 10h AL = 02h ES:DX= 17-byte data address	
Setting attribute code intensity/blink	AH = 10h AL = 03h BH = 00h: Set attribute bit 7 to intensity function. = 01: Set attribute bit 7 to blink function.	
Reading basic pallet register	AH = 10h AL = 07h BL = Basic pallet register to read (0 - 15)	BH = Value read

Operation	Input	Output
Setting pallet register		
Reading overscan register	AH = 10h AL = 08h	BH = Value read
Reading basic pallet register and overscan register	AH = 10h AL = 09h ES:DX = 17-byte buffer in which return value is located	
Setting expansion pallet register	AH = 10h AL = 10h BX = Selected expansion pallet register DH = Red value CH = Green value CL = Blue value	
Setting block in expansion pallet register	AH = 10h AL = 10h ES:DX = Color value table BX = First expansion pallet register to set CX = Number of blocks set in expansion pallet registers	
Selecting pallet page mode	AH = 10h AL = 12h BL = 00h BH = Pallet page mode = 00h: 4-pallet page mode = 01h: 16-pallet page mode	
Selecting pallet page	AH = 10h AL = 13h BL = 01h BH = Expansion pallet page No.	
Reading expansion pallet register	AH = 10h AL = 15h BX = Expansion pallet register to read	
Reading block in expansion pallet register	AH = 10h AL = 17h ES:DX = Buffer address of value to read BX = Expansion pallet address No. where to start reading CX = Number of registers to read	Buffer selected in ES:DX
Reading pallet page	AH = 10h AL = 17h	BL = Current pallet page mode = 0: 4-pallet page mode = 1: 16-pallet page mode BH = Current expansion pallet pagemode
Gray scale conversion	AH = 10h AL = 1Bh BX = First expansion pallet register CX = Number of expansion pallet registers to convert	
Font registration		
Font registration of user-defined character	AH = 11h AL = 00h BH = Number of vertical bits in character BL = Character generator bank No. CX = Number of characters to register DX = First character code to register ES:BP = Top address in user-defined table	

Appendices

Operation	Input	Output
Font registration		
8 x 14 dot font registration (PC character set, inside video BIOS ROM)	AH = 11h AL = 01h BL = Character generator bank No.	
8 x 8 dot font registration (PC character set, inside video BIOS ROM)	AH = 11h AL = 021h BL = Character generator bank No.	
Overwriting character map register	AH = 11h AL = 03h BL = Character map register value	
8 x 16 dot font registration (PC character set, inside video BIOS ROM)	AH = 11h AL = 04h BL = Character generator bank No.	
Font registration of user- defined character (Video controller auto set)	AH = 11h AL = 10h BH = Number of vertical bits in character (Horizontal fixed to 8 bits) BL = Character generator bank No. CX = Number of characters to register DX = First character code to register ES:BP = Top address in user-defined table	
8 x 14 dot font registration (PC character set, inside video BIOS ROM) [Video controller auto set]	AH = 11h AL = 11h BL = Character generator bank No.	
8 x 8 dot font registration (PC character set, inside video BIOS ROM) [Video controller auto set]	AH = 11h AL = 12h BL = Character generator bank No.	
8 x 16 dot font registration (PC character set, inside video BIOS ROM) [Video controller auto set]	AH = 11h AL = 14h BL = Character generator bank No.	
Font registration (CGA) in 0:7Ch (INT 1Fh)	AH = 11h AL = 20h ES:BP = Top address in font table	
Font registration (CGA) in 0:10Ch (INT 43h)	AH = 11h AL = 21h BL = 0: (Number of lines per DL register value) 1: 14 lines 2: 25 lines 3: 43 lines CX = Number of vertical bits per character DL = Number of screen lines (When BL = 0) ES:BP = Top address in font table	
8 x 14 dot font registration (PC character set, inside video BIOS ROM) [Graphics]	AH = 11h AL = 22h BL = 0: (Number of lines per DL register value) 1: 14 lines 2: 25 lines 3: 43 lines DL = Number of screen lines (When BL = 00h)	

Operation	Input	Output
Font registration		
8 x 8 dot font registration (PC character set, inside video BIOS ROM) [Graphics]	AH = 11h AL = 23h BL = 0: (Number of lines per DL register value) 1: 14 lines 2: 25 lines 3: 43 lines DL = Number of screen lines (When BL = 00h)	
8 x 16 dot font registration (PC character set, inside video BIOS ROM) [Graphics]	AH = 11h AL = 24h BL = 0: (Number of lines per DL register value) 1: 14 lines 2: 25 lines 3: 43 lines DL = Number of screen lines (When BL = 00h)	
Reading font table data	AH = 11h AL = 30h BL = 0: Return INT 1Fh (CGA font) entry address with ES:BP register. 1: Return INT 43h entry address with ES:BP register. 2: Return 8 x 14 font address with ES:BP register. 3: Return 8 x 8 font address with ES:BP register. 4: Return 8 x 8 font (80h) address with ES:BP register. 5: Reserved	CX = Number of vertical bits per DL = character ES:BP Number of lines per screen - 1 = Address of data specified in BH
Performance selection		
Reading video mode data (Color/B/W mode)	AH = 12h BL = 10h	BH = 0: Color mode 1: B/W mode (MDA mode) BL = Video RAM memory size CH = Reserved CL = DIP switch setting
Selecting video BIOS print-screen	AH = 12h BL = 10h	
Selecting vertical resolution	AH = 12h BL = 30h AL = Selected vertical resolution = 00h: 200 display lines = 01h: 350 display lines = 02h: 400 display lines	AL = 12h Successfully executed
Default pallet load enable/disable	AH = 12h BL = 31h AL = Load selection 0: Default pallet loading enabled 1: Default pallet loading disabled	AL = 12h Successfully executed
Video enable/disable	AH = 12h BL = 32h AL = 0: Video enabled 1: Video disabled	AL = 12h Successfully executed
Gray scale enable/disable	AH = 12h BL = 33h AL = 0: Gray scale enabled 1: Gray scale disabled	AL = 12h Successfully executed

Appendices

Operation	Input	Output
Performance selection		
Cursor emulator enable/disable	AH = 12h BL = 34h AL = Selection 0: Cursor emulator enabled 1: Cursor emulator disabled	AL = 12h : Successfully executed
Screen ON/OFF	AH = 12h BL = 6h AL = 0: Screen ON 1: Screen OFF	AL = 12h : Successfully executed
Character string output to screen		
Character output without cursor movement	AH = 13h AL = 00h BH = Page No. BL = Attribute CX = Length DH = Starting line position of first character in character string DL = Starting column position of first character in character string ES:BP = Top address in character string	
Character output with cursor movement	AH = 13h AL = 01h BH = Page No. BL = Attribute CX = Number of characters DH = Starting line position of first character in character string DL = Starting column position of first character in character string ES:BP = Top address in character string	
Character output and attribute selection without cursor movement	AH = 13h AL = 02h BH = Page No. CX = Number of characters DH = Starting line position of first character in character string DL = Starting column position of first character in character string ES:BP = Top address in character string	
Character output and attribute selection with cursor movement	AH = 13h AL = 03h BH = Page No. CX = Number of characters DH = Starting line position of first character in character string DL = Starting column position of first character in character string ES:BP = Top address in character string	
Display combination code		
Reading display combination code	AH = 1Ah AL = 00h	AL = 1Ah : Successfully executed BL = Active-display code BH = Nonactive-display code
Writing display combination code	AH = 1Ah AL = 01h BL = Active-display code BH = Nonactive-display code	AL = 1Ah : Successfully executed
Status information	AH = 1Bh BX = Implementation type ES:DI = Return information buffer	AL = 1Bh : Successfully executed

Operation	Input	Output
Video status save & restore		
Reading buffer size	AH = 1Ch AL = 00H CX = Request status Bit 0: Video hardware status Bit 1: Video BIOS data area Bit 2: Expansion pallet register save / restore	AL = 1Ch : Successfully executed BX = Buffer size in 64-byte units
Status save	AH = 1Ch AL = 01h CX = Request status Bit 0: Video hardware status Bit 1: Video BIOS data area Bit 2: Expansion pallet register save/ restore ES:BX = Address of data save area	AL = 1Ch : Successfully executed
Status save as	AH = 1Ch AL = 02h CX = Request status Bit 0: Video hardware status Bit 1: Video BIOS data area ES:BX = Bit 2: Expansion pallet register ssave / restore	

■ INT 11h Reading System Data

Operation	Input	Output
Reading system data		AX = System configuration data AX bit Bits 15 & 14 : Number of printer ports Bits 11, 10 & 9 : Number of RS-232C ports Bits 7 & 6 : Number of internal FDDs 0, 0 : 1 0, 1 : 2 Bits 5 & 4 : Screen mode 0, 1 : 40 x 25 line mode 1, 0 : 80 x 25 line mode Bit 1 : 80287 CPU yes/no 0 : No 1 : Yes Bits 13, 12, 8, 3, 2 & 0 : Reserved

■ INT 12h Reading Memory Size

Operation	Input	Output
Reading memory size		AX =Memory size in 1 K units

Appendices

■ INT 13 Disk BIOS

Operation	Input	Output																																																
Disk reset	AH = 00h DL = Drive No. (FDD:00h-01h;HDD:80h-81h)	CY = 0 : Successfully completed = 1 : Error AH = Status Fbppy disk status table <table border="1"> <thead> <tr> <th>Status</th> <th>Description</th> </tr> </thead> <tbody> <tr><td>00h:</td><td>Successfully completed</td></tr> <tr><td>01h:</td><td>Wrong command sent</td></tr> <tr><td>02h:</td><td>Cannot find address mark</td></tr> <tr><td>03h:</td><td>Attempted writing on protected disk</td></tr> <tr><td>04h:</td><td>Cannot find requested sector.</td></tr> <tr><td>06h:</td><td>Disk was changed.</td></tr> <tr><td>08h:</td><td>DMA overrun</td></tr> <tr><td>10h:</td><td>CRC error during diskette reading</td></tr> <tr><td>20h:</td><td>Crashed FDC</td></tr> <tr><td>40h:</td><td>Crashed during seek operation</td></tr> <tr><td>80h:</td><td>Timer overflowed</td></tr> </tbody> </table> Hard disk status table <table border="1"> <thead> <tr> <th>Status</th> <th>Description</th> </tr> </thead> <tbody> <tr><td>00h:</td><td>Successfully completed</td></tr> <tr><td>01h:</td><td>Wrong command sent</td></tr> <tr><td>02h:</td><td>Cannot find address mark</td></tr> <tr><td>04h:</td><td>Cannot find requested sector.</td></tr> <tr><td>07h:</td><td>Drive parameter error</td></tr> <tr><td>08h:</td><td>DMA overrun</td></tr> <tr><td>10h:</td><td>Error during reading</td></tr> <tr><td>20h:</td><td>Crashed HDC</td></tr> <tr><td>40h:</td><td>Crashed during seek operation</td></tr> <tr><td>80h:</td><td>Timer overflowed</td></tr> <tr><td>BBh:</td><td>Undefined error</td></tr> </tbody> </table>	Status	Description	00h:	Successfully completed	01h:	Wrong command sent	02h:	Cannot find address mark	03h:	Attempted writing on protected disk	04h:	Cannot find requested sector.	06h:	Disk was changed.	08h:	DMA overrun	10h:	CRC error during diskette reading	20h:	Crashed FDC	40h:	Crashed during seek operation	80h:	Timer overflowed	Status	Description	00h:	Successfully completed	01h:	Wrong command sent	02h:	Cannot find address mark	04h:	Cannot find requested sector.	07h:	Drive parameter error	08h:	DMA overrun	10h:	Error during reading	20h:	Crashed HDC	40h:	Crashed during seek operation	80h:	Timer overflowed	BBh:	Undefined error
Status	Description																																																	
00h:	Successfully completed																																																	
01h:	Wrong command sent																																																	
02h:	Cannot find address mark																																																	
03h:	Attempted writing on protected disk																																																	
04h:	Cannot find requested sector.																																																	
06h:	Disk was changed.																																																	
08h:	DMA overrun																																																	
10h:	CRC error during diskette reading																																																	
20h:	Crashed FDC																																																	
40h:	Crashed during seek operation																																																	
80h:	Timer overflowed																																																	
Status	Description																																																	
00h:	Successfully completed																																																	
01h:	Wrong command sent																																																	
02h:	Cannot find address mark																																																	
04h:	Cannot find requested sector.																																																	
07h:	Drive parameter error																																																	
08h:	DMA overrun																																																	
10h:	Error during reading																																																	
20h:	Crashed HDC																																																	
40h:	Crashed during seek operation																																																	
80h:	Timer overflowed																																																	
BBh:	Undefined error																																																	
Reading disk drive data	AH = 01h DL = Drive No. (FDD:00h-01h;HDD:80h-81h)	AH = Disk Drive Status																																																
Reading sector	AH = 02h AL = Number of sectors WithFDD CH = Track No. CL = Sector No. WithHDD CH = Cylinder No. CL = Insignificant 8 bits Sector No.(Bits 0 - 6) = Cylinder No. Significant 2 bits (Bits 7 - 8) DH = Head No. DL = Drive No. (FDD : 00h - 01h, HDD : 80h - 81h) ES BX = Buffer address	CY = 0 : Successfully completed = 1 : Error AH = Status																																																

Operation	Input	Output
Writing sector	AH = 03h AL = Number of sectors With FDD CH = Track No. CL = Sector No. With HDD CH = Cylinder No. Insignificant 8 bits CL = Sector No. (Bits 0 - 5) = Cylinder No. Significant 2 bits (Bits 6-7) DH = Head No. DL = Drive No. (FDD: 00h - 01h, HDD: 80h - 81h) ES:BX = Buffer address	CY = 0: Successfully completed = 1: Error AH = Status
Sector content check	AH = 04h AL = Number of sectors With FDD Track No. CH = Sector No. CL = Cylinder No. Insignificant 8 bits With HDD Sector No. (Bits 0 - 5) CH = Cylinder No. Significant 2 bits (Bits 6-7) CL = Head No. = Drive No. (FDD: 00h - 01h, HDD: 80h - 81h) DL =	CY = 0: Successfully completed = 1: Error AH = Status
Track/Cylinder format	AH = 05h AL = Track/Cylinder No. Insignificant 8 bits CL = Track/Cylinder No. Significant 2 bits DH = Head No. DL = Drive No. ES:BX = Top address in format data table	CY = 0: Successfully completed = 1: Error AH = Status
Reading drive parameters	AH = 08h DL = Drive No. (FDD: 00h - 01h, HDD: 80h - 81h)	CY = 0: Successfully completed = 1: Error AH = Status When FDD drive No. specified in DL ES:DI = Drive parameter starting address CH = Insignificant 8 bits in 10 bit value for max. number of tracks CL = Bits 7 & 6: Insignificant 2 bits in 10 bit value for max. number of tracks Bits 5 - 0: Max. number of sectors per track DH = Max. number of heads DL = Number of built-in floppy disk drives BL = Bits 3 - 0: CMOS valid drive type value Bits 7 - 4: Fixed to 0 When HDD drive No. specified in DL CH = Max. number of cylinders CL = Max. number of usable sectors and Significant bit in max. number of cylinders DH = Max. number of usable heads DL = Number of built-in hard disk drives

Appendices

Operation	Input	Output
Hard disk drive parameter initialization	AH = 09h DL = Drive No.	CY = 0: Successfully completed = 1: Error AH = Status
Hard disk seek	AH = 0Ch CH = Insignificant 8 bits in cylinder No. CL = Significant 2 bits in cylinder No. DL = Drive No. DH = Head No.	CY = 0: Successfully completed = 1: Error AH = Status
Hard disk drive reset	AH = 0Dh DL = Drive No.	CY = 0: Successfully completed = 1: Error AH = Status
Hard disk drive ready check	AH = 10h DL = Drive No.	CY = 0: Successfully completed = 1: Error AH = Status
Hard disk drive recalibration	AH = 11h DL = Drive No.	CY = 0: Successfully completed = 1: Error AH = Status
Disk type check	AH = 15h DL = Drive No.	CY = 0: Successfully completed = 1: Error CX:DX = Total number of sectors (HDD only) AH = 00h: No disk = 01h: Floppy disk unchanged = 02h: Disk was changed = 03h: Hard disk
Media change check	AH = 16h DL = Drive No.	AH = 00h: Disk unchanged = 01h: Inconsistent parameter = 06h: Disk was changed (Carry bit ON) = 80h: Drive not ready
Setting drive type for formatting	AH = 17h DL = Drive No. AL = 0: No disk 1: Use 2-sided disk on 2-sided drive. 2: Use 2-sided disk on high-density drive. 3: Use 2HD disk on high-density drive. 4: Use 720K disk on 720K-byte drive.	CY = 0: Successfully completed = 1: Error AH = Status
Setting media type for formatting	AH = 18h CH = Insignificant 8 bits in 10 bit value for max. number of tracks CL = Bits 7 & 6: 10 bit value for max. number of tracks Bits 5 - 0: Max. number of sectors per track DL = Drive No.	ES:DI = Address of drive parameter table for floppy disk type AH = 00h and CY =0: Supports tracks and number of tracks per sector. AH = 01h and CY =1: Cannot use function. AH = 00h and CY =1: Does not support tracks and number of tracks per sector.

■ INT 14h RS-232C

Operation	Input	Output
Setting RS-232C line mode	AH = 00h AL = Line mode/Parameter Bits 7, 6, 5: Baud rate 0, 0, 0: 110 Baud 0, 0, 1: 150 Baud 0, 1, 0: 300 Baud 0, 1, 1: 600 Baud 1, 0, 0: 1200 Baud 1, 0, 1: 2400 Baud 1, 1, 0: 4800 Baud 1, 1, 1: 9600 Baud Bits 4&3: Parity X, 0: No parity 0, 1: Odd parity 1, 2: Even parity Bit 2: Stop bits 0: 1 bit length 1: 2 bit length Bits 1&0: Word length 1, 0: 7 bits 1, 1: 8 bits DX = Port No.	
Sending 1-byte data	AH = 01h AL = Data to send DX = Port No.	AH = Line control status Bit 7: Time over error Bit 6: Transmission shift register empty Bit 5: Transmission hold register empty Bit 4: Break detected Bit 3: Framing error Bit 2: Parity error Bit 1: Overrun Bit 0: Data ready
Receiving 1-byte data	AH = 02h DX = Port No.	AL = Received data AH = Line control status
Reading communication port status	AH = 03h DX = Port No.	AL = Line control status AL = Modem status Bit 7: Carry detected Bit 6: Call signal received Bit 5: Data-set ready Bit 4: Transmission enabled (CTS) Bit 3 - 0: Not in use

■ INT 15h Other System Services

Operation	Input	Output
Wait time check	AH = 83h AL = 00h: Interval timer start 01h: Interval timer stop ES:BX = End flag address CX:DX = Wait time in 1 µsec units	CY = 0: Successfully completed = 1: Error

Appendices

Operation	Input	Output								
Joystick										
Reading button status	AH = 84h DX = 00h	AL = Button status 00h: Pressed 01h: Not pressed Bit 7: 2nd button status on 2nd joystick Bit 6: 1st button status on 2nd joystick Bit 5: 2nd button status on 1st joystick Bit 4: 1st button status on 1st joystick								
Reading resistance value	AH = 84h DX = 01h	AX = Resistance value of horizontal coordinate of 1st joystick BX = Resistance value of vertical coordinate of 1st joystick CX = Resistance value of horizontal coordinate of 2nd joystick DX = Resistance value of vertical coordinate of 2nd joystick								
Wait timer overwait	AH = 86h CXDX = Wait time in μ sec units	CY = 0: Successfully completed = 1: Already triggered or not supported								
Transmitting memory block in protect mode	AH = 87h ES:SI = Top address in descriptor ES:SI = <table border="1" style="margin-left: 20px;"> <tr><td>Dummy (00h)</td></tr> <tr><td>GDT(00h)</td></tr> <tr><td>Source segment descriptor</td></tr> <tr><td>Destination segment descriptor</td></tr> <tr><td>BIOS C5 (00h)</td></tr> <tr><td>SS (00h)</td></tr> </table> CX = Number of words to transfer	Dummy (00h)	GDT(00h)	Source segment descriptor	Destination segment descriptor	BIOS C5 (00h)	SS (00h)	ZF = 1: Successfully completed CY = 1: Error AH = 00h: Successfully completed = 01h: RAM parity error = 02h: Not in protect mode		
Dummy (00h)										
GDT(00h)										
Source segment descriptor										
Destination segment descriptor										
BIOS C5 (00h)										
SS (00h)										
Starting protect mode	AH = 89h BH = Offset of insignificant (mask) interrupt vector address BL = Offset of significant (slave) interrupt vector address ES:SI = Top address in following table <table border="1" style="margin-left: 20px;"> <tr><td>CS dummy (00h)</td></tr> <tr><td>GDT</td></tr> <tr><td>IDT</td></tr> <tr><td>DS</td></tr> <tr><td>ES</td></tr> <tr><td>SS</td></tr> <tr><td>CS</td></tr> <tr><td>TEMP BIOS (00h)</td></tr> </table>	CS dummy (00h)	GDT	IDT	DS	ES	SS	CS	TEMP BIOS (00h)	CY = 0: Successfully completed AH = 00h CS = CS value specified in ES:SI table DS = DS value specified in ES:SI table ES = ES value specified in ES:SI table SS = DS value specified in ES:SI table CY = 1: Error AH = FFh
CS dummy (00h)										
GDT										
IDT										
DS										
ES										
SS										
CS										
TEMP BIOS (00h)										

■ INT 16h Keyboard BIOS

Operation	Input	Output
Reading data by key input	AH = 00h	AH = Secondary code AL = Primary code (Character code)
Data check by key input	AH = 01h	ZF = 0: Input data exists. = 1: Input data does not exist. AH = Secondary code AL = Primary code (Character code)
Reading shift status	AH = 02h	AL = Shift status
Setting repeat delay and repeat rate	AH = 03h AL = 05h BH = Delay time (Bits 0 & 1) BL = Primary code (Bits 0 - 4)	
Writing key data	AH = 05h CH = Secondary code CL = Primary code	AL = 00h: Successfully completed = 01h: No available space in buffer (CY =1)
Wiring data by key input (101/AXkeyboard compatible)	AH = 10h	AH = Secondary code AL = Primary code (Character code)
Data check by key input (101/AXkeyboard compatible)	AH = 11h	ZF = 0: Input data exists. = 1: Input data does not exist. AH = Secondary code AL = Primary code (Character code)
Reading shift status (101/AXkeyboard compatible)	AH = 12h	AL = Shift status 1 AH = Shift status 2

■ INT 17h Printer BIOS

Operation	Input	Output
1-character output	AH = 00h AL = Character code of character to output DX = output Device No.	AH = Printer status
Printer initialization	AH = 01h DX = Device No.	AH = Printer status
Status check	AH = 02h DX = Device No.	AH = Printer status

Appendices

■ INT 1Ah Setting/Reading Time and Date

Operation	Input	Output
Reading clock	AH = 01h	CX = Significant 16 bits of current clock data DX = Insignificant 16 bits of current clock data AL = Overflow flag on 24-hour system
Setting clock	AH = 01h CX = Significant 16 bits of clock data DX = Insignificant 16 bits of clock data	
Reading time	AH = 02h	CH = Hours (BCD) CL = Minutes (BCD) DH = Seconds (BCD) DL = Summertime option (0 or 1) CY = End status 0: Successfully completed 1: Error
Setting time	AH = 03h CH = Hours (BCD) CL = Minutes (BCD) DH = Seconds (BCD) DL = Summertime option (0 or 1)	
Reading date	AH = 04h	CH = Western calendar (Year given as 2-digit BCD, significant 2 bits: 19 or 20) CL = Year (BCD) DH = Month (BCD) DL = Day (BCD) CY = End status 0: Successfully completed 1: Error
Setting date	AH = 05h CH = Western calendar (Year given as 2-digit BCD, significant 2 bits: 19 or 20) CL = Year (BCD) DH = Month (BCD) DL = Day (BCD)	
Setting alarm	AH = 06h CH = Hours (BCD) CL = Minutes (BCD) DH = Seconds (BCD)	CY = End status 0: Successfully completed 1: Error
Clearing alarm	AH = 07h	

A.7 System Monitor/RAS Feature API-DLL

A.7.1 Operation Environment

The following information explains the Dynamic Link Libraries used by the System Monitor/RAS feature on a PL-5900 Series unit.

API-DLLs provide the interface for applications to access the System Monitor/RAS feature (System Monitor/RAS Device Driver). Applications can use DLLs to access the following types of features.

1. Driver Version information
2. System Monitor feature status
3. Read out (Get) various monitoring parameters (voltage, fan, temperature)
4. System Monitor current data (voltage, fan, temperature)
5. Watchdog parameters
6. Alarm processing
7. General input processing
8. Reset (of PL unit)
9. Monitoring the status of mirror disk driver
10. Event handling

■ Compatible Operating Systems

The API-DLLs contained on the PL unit's CD-ROM are compatible with the following OS types.

- Microsoft® Windows®98 Second Edition
- Microsoft® WindowsNT®4.0 (Windows Service Pack 3 or higher)
- Microsoft® Windows®2000

Each OS must use its corresponding System Monitor/RAS Device Driver.

■ Compatible Languages

- Microsoft® Visual C
- Microsoft® Visual C++
- Microsoft® Visual Basic

Appendices

◆ Required Files

The following files are required when using DLLs. Each language requires its own set of files.

• Visual C

File Name	Description
PL_Iocif.h	Driver interface definition "include" file
PL_Ioc.LIB	Library definition file
PL_Ioc.dll	Dynamic link library file

• Visual C++

File Name	Description
PL_Iocif.h	Driver interface definition "include" file
PL_Iocall.h	CPL_Iocall class definition "include" file
PL_Ioctl.h	CPL_Ioctl class definition "include" file
PL_Ioc.LIB	Library definition file
PL_Ioc.dll	Dynamic Link library file
PL_SmiIoctl.h	CPL_SmiIoctl class definition "include" file (used only with software mirroring feature)
Sm.h	Software mirroring definition file (used only with software mirroring feature)

* "#include header files should be "included" in the following order.

```
#include PL_Iocif.h
```

```
#include PL_Ioctl.h
```

```
#include Sm.h
```

```
#include PL_SmiIoctl.h
```

PL_Iocall.h is automatically included, and does not need to be directly designated.

• Visual Basic

File Name	Description
PL_Ioc.bas	Driver interface definition file
PL_Ioc.LIB	Library definition file
PL_Ioc.dll	Dynamic link library file

■ Dynamic Link Library (DLL)

In order for an application to use PL_Ioc.dll, it should be copied to the following folder.

OS	Location
Windows [®] 98	C:\Windows\System
WindowsNT [®] 4.0/Windows [®] 2000	C:\Winnt\System32

A.7.2 Class Contents

■ CPL_Ioctl Class

This class is used to set the parameters for device driver access using CPL_Ioctl class.

Key Word	Type	Variable Name	Description
public	HANDLE	m_Drvhandle	Device driver handle

■ CPL_Iocal Class

This uses the parameters set in CPL_Ioctl, and calls up DeviceIoControl (Driver Access function).

However, since this class succeeds CPL_Ioctl, it cannot be used directly.

Key Word	Type	Variable Name	Description
public	HANDLE	m_h	Device driver handle
public	LONG	m_long	Control code for action to perform
public	void *	m_ibp	Input data buffer address
public	ULONG	m_ibsize	Input data buffer size
public	void *	m_obp	Output data buffer address
public	ULONG	m_obsz	Output data buffer size
public	DWORD	m_retsz	Address for actual no. of output bytes
public	LPOVERLAPPED	m_ovlp	Address of overlap design

■ CPL_SmiIoctl Class

This class is used to set the parameters for device driver access using CPL_SmiIoctl class.

This class is only used when using the Software Mirroring driver.

Key Word	Type	Variable Name	Description
public	HANDLE	m_Drvhandle	Device driver handle

A.7.3 Visual C Functions

Function Name	Description
InitIoctl	Creates the CPL_ioctl object
EndIoctl	Destroys the CPL_ioctl object
GetDrvHandle	Gets the driver handle
CloseDrvHandle	Destroys the driver handle
GetDrvVersion	Gets the driver version
GetMonitorSetup	Gets the monitoring enabled/disabled setting
GetVoltParam	Gets the voltage monitoring parameter
GetCurrentVolt	Gets the current voltage value
GetFanParam	Gets the fan monitoring parameter
GetCurrentFan	Gets the current fan value
GetTempParam	Gets the temperature monitoring parameter
GetCurrentTemp	Gets the current temperature value
GetWdtCounter	Gets the watchdog timer counter
SetWdtMask	Sets warning masking in case of watchdog timer timeout
GetWdtMask	Gets warning masking in case of watchdog timer timeout
StartWdt	Starts the watchdog timer
StopWdt	Stops the watchdog timer
RestartWdt	Restarts the watchdog timer
RunningWdt	Gets the watchdog timer operation status
SetWarningOut	Sets warning output
GetWarningOut	Gets warning output
GetUniversalln	Gets universal input
ClearUniversalln	Clears the universal input latched status
SetUniversallnMask	Sets universal input masking
GetUniversallnMask	Gets universal input masking
SetResetMask	Sets reset-masking
GetResetMask	Gets reset-masking
SetIdeErr	Sets data mirroring (software) error
GetIdeErrHard	Gets data mirroring (hardware) error
GetLightblowErr	Gets backlight burnout status
GetEvent	Gets the error event
ClearEvent	Clears the error event
StartInsideBuzzer	Starts PL internal buzzer
StopInsideBuzzer	Stops PL internal buzzer
ChkInsideBuzzer	Checks PL internal buzzer status
GetWdtTimeout	Gets the timeout status of the watchdog timer
ClearWdtTimeout	Clears the timeout status of the watchdog timer
SetWarningDOUT	Sets the warning output DOUT
GetWarningDOUT	Gets the warning output DOUT
GetSmiDrvHandle	Gets Software Mirroring driver handle
CloseSmiDrvHandle	Destroys Software Mirroring driver handle
GetSmiAryStatus	Gets status of Software Mirroring Array
GetSmiDevStatus	Gets status of Software Mirroring Device

A.7.4 Visual C Function Specifications (Details)

InitIoctl

Call Format	void WINAPI InitIoctl(void)
Return Value	None
Arguments	None
Processing	Creates a CPL_Ioctl object. The object is not destroyed until the EndIoctl function is called.
Example	InitIoctl();

EndIoctl

Call Format	void WINAPI EndIoctl(void)
Return Value	None
Arguments	None
Processing	Destroys the object created using the InitIoctl function.
Example	EndIoctl();

GetDrvHandle

Call Format	int WINAPI GetDrvHandle(HANDLE * pHndl)
Return Value	0: Normal 1: Error
Arguments	(I/O) HANDLE *pHndl Pointer to the device driver handle
Processing	Gets the device driver handle to communicate with the device driver.
Example	int ret; HANDLE hndl; ret = GetDrvHandle(&hndl);



Note: An error occurs if the System Monitor/RAS Device Driver is not running.

CloseDrvHandle

Call Format	BOOL WINAPI CloseDrvHandle(void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None
Processing	Destroys the device driver handle created using the GetDrvHandle function.
Example	BOOL ret; //Destroys the handle ret = CloseDrvHandle();

Appendices

GetDrvVersion

Call Format	BOOL WINAPI GetDrvVersion(int *pMajor, int *pMinor)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I/O) int *pMajor Pointer to version information (Major, 0 to 99). (I/O) int *pMinor Pointer to version information (Minor, 0 to 99).
Processing	Gets the driver's version information.
Example	BOOL ret; int Major, Minor; ret = GetDrvVersion(&Major, &Minor);



Note: If the version is 1.10, then you will get

Major: 1 (decimal)
Minor: 10 (decimal).

GetMonitorSetup

Call Format	BOOL WINAPI GetMonitorSetup(int Selector, int *pSetup)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I)intSelector Parameters MONITOR_VOLT_CPU CPU core voltage MONITOR_VOLT_P33 +3.3 V MONITOR_VOLT_P50 5.0 V MONITOR_VOLT_P12 +12 V MONITOR_VOLT_M12 -12 V MONITOR_VOLT_M50 -5.0 V MONITOR_TEMP_SYSTEM System temperature MONITOR_TEMP_CPU CPU temperature MONITOR_TEMP_OPT Option temperature MONITOR_FAN_CPU CPU fan MONITOR_FAN_POWER Power fan MONITOR_FAN_OPT Option fan (I/O) int *pSetup Pointer to Data 0: Disabled 1: Enabled
Processing	Gets the current monitoring status (enabled/disabled).
Example	BOOL ret; int Setup; // Gets the CPU core voltage setup status. ret = GetMonitorSetup(MONITOR_VOLT_CPU, &Setup);

GetVoltParam

Call Format	BOOL WINAPI GetVoltParam (int Selector, int *pULimit, int *pLLimit)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Parameters MONITOR_VOLT_CPU CPU core voltage MONITOR_VOLT_P33 +3.3 V MONITOR_VOLT_P50 +5.0 V MONITOR_VOLT_P12 +12 V MONITOR_VOLT_M12 -12 V MONITOR_VOLT_M50 -5.0 V (I/O) int *pULimit Pointer to upper-limit voltage value (Unit: mV) (I/O) int *pLLimit Pointer to lower-limit voltage value (Unit: mV)
Processing	Gets the voltage monitoring parameter.
Example	<pre> BOOL ret; int ULimit, LLimit; // Get the upper and lower-limit values of the CPU core // voltage. ret = GetVoltParam(MONITOR_VOLT_CPU, &ULimit, &LLimit); </pre>



Since the data taken from this function is shown in mV units, the following conversion is needed for use in (Volt) units:

$$\text{Data in Volt unit} = \text{Data in mV unit} / 1000$$

Appendices

GetCurrentVolt

Call Format	BOOL WINAPI GetCurrentVolt(int Selector, int *pData)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Parameters MONITOR_VOLT_CPU CPU core voltage MONITOR_VOLT_P33 +3.3 V MONITOR_VOLT_P50 +5.0 V MONITOR_VOLT_P12 +12 V MONITOR_VOLT_M12 -12 V MONITOR_VOLT_M50 -5.0 V (I/O) int *pData Pointer to the voltage value (Unit: mV)
Processing	Gets the current voltage value.
Example	BOOL ret; int Data; // Gets the CPU core voltage value. ret = GetCurrentVolt(MONITOR_VOLT_CPU, &Data);



Note: Since the data taken from this function is in mV units, the following conversion is needed for use in (Volt) units:
Data in Volt unit = Data in mV unit/1000

GetFanParam

Call Format	BOOL WINAPI GetFanParam (int Selector, int *pLLimit)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Parameters MONITOR_FAN_CPU CPU fan MONITOR_FAN_POWER Power fan MONITOR_FAN_OPT Option fan (I/O)int *pLLimit Pointer to the lower-limit fan rotation speed (Unit: RPM) (RPM: Revolutions Per Minute)
Processing	Gets the fan monitoring parameter.
Example	BOOL ret; int LLimit; // Gets the lower-limit CPU fan rotation speed. ret = GetFanParam(MONITOR_FAN_CPU, &LLimit);

GetCurrentFan

Call Format	BOOL WINAPI GetCurrentFan(int Selector, int *pData)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Parameters MONITOR_FAN_CPU CPU fan MONITOR_FAN_POWER Power fan MONITOR_FAN_OPT Option fan (I/O) int *pData Pointer to the fan rotation speed (Unit: RPM) (RPM: Revolutions Per Minute)
Processing	Gets the current fan rotational speed.
Example	<pre> BOOL ret; int Data; // Gets the CPU fan rotational speed. ret = GetCurrentFan(MONITOR_FAN_CPU, &Data); </pre>

GetTempParam

Call Format	BOOL WINAPI GetTempParam(int Selector, int *pULimit)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Parameters MONITOR_TEMP_SYSTEM System temperature MONITOR_TEMP_CPU CPU temperature MONITOR_TEMP_OPT Option temperature (I/O) int *pULimit Pointer to the upper-limit temperature (Unit: Degrees Celsius)
Processing	Gets the temperature monitoring parameter.
Example	<pre> BOOL ret; int ULimit; // Gets the system temperature upper-limit value. ret = GetTempParam(MONITOR_TEMP_SYSTEM, &ULimit); </pre>

Appendices

GetCurrentTemp

Call Format BOOL WINAPI GetCurrentTemp(int Selector, int *pData)

Return Value TRUE: Normal

FALSE: Error

Arguments (I)int Selector Parameters

MONITOR_TEMP_SYSTEM System temperature

MONITOR_TEMP_CPU CPU temperature

MONITOR_TEMP_OPT Option temperature

(I/O) int *pData Pointer to the temperature

(Unit: Degrees Celsius)

Processing Gets the current temperature value.

Example BOOL ret;

int Data;

// Gets the system temperature value.

ret = GetCurrentTemp(MONITOR_TEMP_SYSTEM, &Data);

GetWdtCounter

Call Format BOOL WINAPI GetWdtCounter (int *pCounter)

Return Value TRUE: Normal

FALSE: Error

Arguments (I/O) int *pCounter Gets to the watchdog timer's initial counter value (5 to 255) (Unit: Seconds)

Processing Gets the current watchdog timer's initial counter value.

Example BOOL ret;

int Counter;

ret = GetWdtCounter(&Counter);

SetWdtMask

Call Format	BOOL WINAPI SetWdtMask(int Selector, int Mask)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Setting Item WARNING_LAMP LAMP WARNING_ALARM ALARM (I) int Mask Masking Information MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Sets masking for the warning output used when watchdog timer time-out occurs.
Example	<pre> BOOL ret; // Enables masking for the lamp output. ret = SetWdtMask(WARNING_LAMP, MASK_ON); // Disables masking for the alarm output. ret = SetWdtMask(WARNING_ALARM, MASK_OFF); </pre>

GetWdtMask

Call Format	BOOL WINAPI GetWdtMask(int Selector, int *pMask)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Setting Item WARNING_LAMP LAMP WARNING_ALARM ALARM (I/O) int *pMask Pointer to Masking Information MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Gets the masking information used for warning output when watchdog timer time-out occurs.
Example	<pre> BOOL ret; int Mask; // Gets the masking information for the LAMP. ret = GetWdtMask(WARNING_LAMP, &Mask); // Gets the masking information for the alarm. ret = GetWdtMask(WARNING_ALARM, &Mask); </pre>

Appendices

StartWdt

Call Format	BOOL WINAPI StartWdt(void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None
Processing	Starts watchdog timer countdown.
Example	BOOL ret; ret = StartWdt();

StopWdt

Call Format	BOOL WINAPI StopWdt(void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None
Processing	Stops watchdog timer countdown.
Example	BOOL ret; ret = StopWdt();

RestartWdt

Call Format	BOOL WINAPI RestartWdt(void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None
Processing	Restarts watchdog timer countdown after resetting to the initialvalue.
Example	BOOL ret; ret = RestartWdt();



Note: RestartWdt can only be used after the StartWdt countdown has started.If RestartWdt is used after the timeout,it should be after ClearWdtTimeout has used to clear the timeout condition and after StartWdt has started the countdown.

RunningWdt

Call Format	BOOL WINAPI RunningWdt(int *pRunFlag)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I/O) int *pRunFlag Pointer to Watchdog Timer Operation Status WATCHDOG_STOP Stopped WATCHDOG_COUNTDOWN Count down in progress
Processing	Gets the watchdog timer's operation status.
Example	BOOL ret; int RunFlag; ret = RunningWdt(&RunFlag);

SetWarningOut

Call Format	BOOL WINAPI SetWarningOut(int Selector, int WarnOut)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Setting Item WARNING_LAMP LAMP WARNING_ALARM ALARM (I) int WarnOut Output Status OUTPUT_OFF Output OFF OUTPUT_ON Output ON
Processing	Sets setting item warning information (LAMP or ALARM).
Example	<pre> BOOL ret; // Sets the LAMP output status to ON. ret = SetWarningOut(WARNING_LAMP, OUTPUT_ON); // Sets the ALARM output status to OFF. ret = SetWarningOut(WARNING_ALARM, OUTPUT_OFF); </pre>

GetWarningOut

Call Format	BOOL WINAPI GetWarningOut(int Selector, int *pWarnOut)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Setting Item WARNING_LAMP LAMP WARNING_ALARM ALARM (I/O) int *pWarnOut Pointer to Output Status OUTPUT_OFF Output OFF OUTPUT_ON Output ON
Processing	Gets currently set item's warning status (LAMP or ALARM).
Example	<pre> BOOL ret; int WarnOut; // Gets the LAMP output status. ret = GetWarningOut(WARNING_LAMP, &WarnOut); // Gets the ALARM output status. ret = GetWarningOut(WARNING_ALARM, &WarnOut); </pre>

Appendices

GetUniversalIn

Call Format	BOOL WINAPI GetUniversalIn(int Selector, int *pUniIn)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Designated Port PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1 (I/O) int *pUniIn Pointer to Input Status INPUT_OFF Input OFF INPUT_ON Input ON
Processing	Gets the input status of the designated port (Universal Input 0, Universal Input 1).
Example	<pre>BOOL ret; int UniIn; // Get the input status of Universal Input 0. ret = GetUniversalIn(PORT_UNI0, &UniIn); // Get the input status of Universal Input 1. ret = GetUniversalIn(PORT_UNI1, &UniIn);</pre>

ClearUniversalIn

Call Format	BOOL WINAPI ClearUniversalIn(int Selector)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Designated Port PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1
Processing	Cancels the input status of the designated port (Universal Input 0, Universal Input 1).
Example	<pre>BOOL ret; // Cancels the input status of Universal Input 0. ret = ClearUniversalIn(PORT_UNI0); // Cancels the input status of Universal Input 1. ret = ClearUniversalIn(PORT_UNI1);</pre>

SetUniversalInMask

Call Format	BOOL WINAPI SetUniversalInMask(int Selector, int Mask)	
Return Value	TRUE: Normal FALSE: Error	
Arguments	(I) int Selector	Designated Port PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1 (I) int Mask Masking Information MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Sets the masking information for the designated port (Universal Input 0, Universal Input 1).	
Example	<pre> BOOL ret; // Disable masking for Universal Input 0. ret = SetUniversalInMask(PORT_UNI0, MASK_OFF); // Enable masking for Universal Input 1. ret = SetUniversalInMask(PORT_UNI1, MASK_ON); </pre>	

GetUniversalInMask

Call Format	BOOL WINAPI GetUniversalInMask(int Selector, int *pMask)	
Return Value	TRUE: Normal FALSE: Error	
Arguments	(I) int Selector	Designated Port PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1 (I/O) int *pMask Pointer to Masking Information MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Gets the masking information for the designated port (Universal Input 0, Universal Input 1).	
Example	<pre> BOOL ret; int Mask; // Gets the masking information for Universal input 0. ret = GetUniversalInMask(PORT_UNI0, &Mask); // Gets the masking information for Universal input 1. ret = GetUniversalInMask(PORT_UNI1, &Mask); </pre>	

Appendices

SetResetMask

Call Format	BOOL WINAPI SetResetMask(int Mask)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Mask MaskingInformation MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Sets reset-masking.
Example	BOOL ret; //Disable reset-masking. ret = SetResetMask(MASK_OFF);

GetResetMask

Call Format	BOOL WINAPI GetResetMask(int *pMask)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I/O) int *pMask Pointer to Masking Information MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Gets the current reset-masking information.
Example	BOOL ret; int Mask; ret = GetResetMask(&Mask);

SetIdeErr

Call Format	BOOL WINAPI SetIdeErr(int IdeErr)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int IdeErr Error Output Status IDE_ERROR_OFF Error Output OFF IDE_ERROR_ON Error Output ON
Processing	Sets the software control used to create IDE error output.
Example	BOOL ret; // Sets IDE error output to OFF. ret = SetIdeErr(IDE_ERROR_OFF);

GetIdeErrHard

Call Format BOOL WINAPI GetIdeErrHard(int Selector, int *pIdeErr)

Return Value TRUE: Normal
 FALSE: Error

Arguments (I) int Selector Parameters
 IDE_ERROR_1 IDE_ERR1
 IDE_ERROR_2 IDE_ERR2
 (I/O) int *pIdeErr Pointer to error signal
 IDE_ERROR_OFF Normal
 IDE_ERROR_ON Error

Processing Gets the current IDE error signal output by the hardware.

Example BOOL ret;
 int IdeErr;
 // Gets the IDE_ERR1 signal
 ret = GetIdeErrHard(IDE_ERROR_1, &IdeErr);

GetLightblowErr

Call Format BOOL WINAPI GetLightblowErr(int *pLightErr)

TRUE: Normal
 FALSE: Error

Arguments (I/O) int *pLightErr Error Information
 BACKLIGHT_OK OK
 BACKLIGH_ERR NG

Processing Gets Backlight's current burnout error output.

Example BOOL ret;
 int LightErr;
 // Gets backlight's burnout condition.
 ret = ::GetLightblowErr(&LightErr);

Appendices

GetEvent

Call Format	BOOL WINAPI GetEvent(int Selector, int *pEvent)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Parameters EVENT_VOLT_CPU CPU core voltage EVENT_VOLT_P33 +3.3 V EVENT_VOLT_P50 +5.0 V EVENT_VOLT_P12 +12 V EVENT_VOLT_M12 -12 V EVENT_VOLT_M50 -5.0 V EVENT_FAN_CPU CPU FAN EVENT_FAN_POWER POWER FAN EVENT_FAN_OPT OPTION FAN EVENT_TEMP_SYSTEM SYSTEM temperature EVENT_TEMP_CPU_OPT CPU or option temperature EVENT_UNI_IN0 Universal Input 0 EVENT_UNI_IN1 Universal Input 1 EVENT_WDT_TIMEOUT Watchdog Timeout (I/O) int *pEvent Pointer to Error Event Information ERROR_EVENT_OFF Without error event ERROR_EVENT_ON With error event
Processing	Checks the machine for voltage, fan, and temperature errors, and the Universal Input information (event) and Watchdog Timeout information.
Example	<pre>BOOL ret; int Event; // Gets the error event information for the CPU core voltage. ret = GetEvent(EVENT_VOLT_CPU, &Event);</pre>

ClearEvent

Call Format	BOOL WINAPI ClearEvent(int Selector)																												
Return Value	TRUE: Normal FALSE: Error																												
Arguments	(I) int Selector Parameters used for cancelling error events <table> <tr> <td>EVENT_VOLT_CPU</td> <td>CPU core voltage</td> </tr> <tr> <td>EVENT_VOLT_P33</td> <td>+3.3 V</td> </tr> <tr> <td>EVENT_VOLT_P50</td> <td>+5.0 V</td> </tr> <tr> <td>EVENT_VOLT_P12</td> <td>+12 V</td> </tr> <tr> <td>EVENT_VOLT_M12</td> <td>-12 V</td> </tr> <tr> <td>EVENT_VOLT_M50</td> <td>-5.0 V</td> </tr> <tr> <td>EVENT_FAN_CPU</td> <td>CPU FAN</td> </tr> <tr> <td>EVENT_FAN_POWER</td> <td>POWER FAN</td> </tr> <tr> <td>EVENT_FAN_OPT</td> <td>OPTION FAN</td> </tr> <tr> <td>EVENT_TEMP_SYSTEM</td> <td>SYSTEM temperature</td> </tr> <tr> <td>EVENT_TEMP_CPU_OPT</td> <td>CPU or option temperature</td> </tr> <tr> <td>EVENT_UNI_IN0</td> <td>Universal input 0</td> </tr> <tr> <td>EVENT_UNI_IN1</td> <td>Universal input 1</td> </tr> <tr> <td>EVENT_WDT_TIMEOUT</td> <td>Watchdog Timeout</td> </tr> </table>	EVENT_VOLT_CPU	CPU core voltage	EVENT_VOLT_P33	+3.3 V	EVENT_VOLT_P50	+5.0 V	EVENT_VOLT_P12	+12 V	EVENT_VOLT_M12	-12 V	EVENT_VOLT_M50	-5.0 V	EVENT_FAN_CPU	CPU FAN	EVENT_FAN_POWER	POWER FAN	EVENT_FAN_OPT	OPTION FAN	EVENT_TEMP_SYSTEM	SYSTEM temperature	EVENT_TEMP_CPU_OPT	CPU or option temperature	EVENT_UNI_IN0	Universal input 0	EVENT_UNI_IN1	Universal input 1	EVENT_WDT_TIMEOUT	Watchdog Timeout
EVENT_VOLT_CPU	CPU core voltage																												
EVENT_VOLT_P33	+3.3 V																												
EVENT_VOLT_P50	+5.0 V																												
EVENT_VOLT_P12	+12 V																												
EVENT_VOLT_M12	-12 V																												
EVENT_VOLT_M50	-5.0 V																												
EVENT_FAN_CPU	CPU FAN																												
EVENT_FAN_POWER	POWER FAN																												
EVENT_FAN_OPT	OPTION FAN																												
EVENT_TEMP_SYSTEM	SYSTEM temperature																												
EVENT_TEMP_CPU_OPT	CPU or option temperature																												
EVENT_UNI_IN0	Universal input 0																												
EVENT_UNI_IN1	Universal input 1																												
EVENT_WDT_TIMEOUT	Watchdog Timeout																												
Processing	Cancels the error event.																												
Example	<pre> BOOL ret; // Cancels the CPU core voltage error event. ret = ClearEvent(EVENT_VOLT_CPU); </pre>																												

Appendices

StartInsideBuzzer

Call Format	BOOL WINAPI StartInsideBuzzer (int hz, int ms)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int hz Buzzer frequency (Hz) (I) int ms Buzzer length (ms)
Processing	Starts the PL unit's internal buzzer, based on the designated frequency and length.
Example	BOOL ret; int hz = 600; int ms = 1000; // PL internal buzzer will sound at 600MHz for 1 second. ret = StartInsideBuzzer (hz, ms);



This feature cannot be used with a PL running WindowsNT 4.0 or Windows 2000, due to the use of Windows 98 functions.

StopInsideBuzzer

Call Format	BOOL WINAPI StopInsideBuzzer (void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None.
Processing	Stops the PL unit's internal buzzer.
Example	BOOL ret; // Stops PL internal buzzer. ret = StopInsideBuzzer ();



This feature cannot be used with a PL running Windows NT or Windows 2000, due to the use of and Windows 98 functions.

ChkInsideBuzzer

Call Format	BOOL WINAPI ChkInsideBuzzer (int *BuzzerParam)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I/O) int *BuzzerParam Pointer to Buzzer Condition BUZZER_ON Buzzer is ON BUZZER_OFF Buzzer is OFF
Processing	Checks the buzzer's operation status.
Example	<pre> BOOL ret; int BuzzerParam; // Checks buzzer status. ret = ChkInsideBuzzer (&BuzzerParam); </pre>



Note: This feature cannot be used with a PL running WindowsNT 4.0 or Windows 2000, due to the use of and Windows 98 functions.

GetWdtTimeout

Call Format	BOOL WINAPI GetWdtTimeout(int *pTimebuf)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I/O) int *pTimebuf Pointer to Watchdog Timeout Status TIMEOUT_OK Not timeout TIMEOUT_ERROR Timeout
Processing	Gets watchdog timeout status.
Example	<pre> BOOL ret; int Timebuf; // Gets watchdog timeout status. ret = GetWdtTimeout(&Timebuf); </pre>

Appendices

ClearWdtTimeout

Call Format BOOL WINAPI ClearWdtTimeout(void)

Return Value TRUE: Normal
 FALSE: Error

Arguments None

Processing Clears the watchdog timeout status.

Example BOOL ret;
 // Clears the watchdog timeout status.
 ret = GetWdtTimeout();

SetWarningDOUT

Call Format BOOL WINAPI SetWarningDOUT(int WarningOut)

Return Value TRUE: Normal
 FALSE: Error

Arguments (I) int WarningOut Output status
 OUTPUT_OFF Output OFF
 OUTPUT_ON Output ON

Processing Sets DOUT warning status of current setting item.

Example BOOL ret;
 // Sets warning DOUT output status to OFF.
 ret = SetWarningDOUT(OUTPUT_OFF);

GetWarningDOUT

Call Format BOOL WINAPI GetWarningDOUT(int *pWarningOut)

Return Value TRUE: Normal
 FALSE: Error

Arguments (I/O) int *pWarningOut Pointer to Output Status
 OUTPUT_OFF Output OFF
 OUTPUT_ON Output ON

Processing Gets DOUT warning status of current setting item.

Example BOOL ret;
 int WarningOut;
 // Gets DOUT Output status.
 ret = GetWarningDOUT(&WarningOut);
 // Sets the ALARM output status to OFF.
 ret = ::SetWarningOut(WARNING_ALARM, OUTPUT_OFF);

GetSmiDrvHandle

Call Format	int WINAPI GetSmiDrvHandle(void)
Return Value	0: Normal 1: Error
Arguments	None
Processing	Gets Software Mirroring Device Driver Handle.
Example	int ret; ret = GetSmiDrvHandle();



When the Software Mirroring Device Driver is not operating, an error occurs.

CloseSmiDrvHandle

Call Format	BOOL WINAPI CloseSmiDrvHandle(void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None
Processing	Destroys the device driver handle created using the GetSmiDrvHandlefunction.
Example	BOOL ret; // Destroys the device driver handle created using the GetSmiDrvHandlefunction. ret = ClosetSmiDrvHandle();

Appendices

GetSmiAryStatus

Call Format	BOOL WINAPI GetSmiAryStatus(int *pStatus)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I/O) int *pStatus Pointor to Software Mirroring Disk Status ARYSTAT_GOOD Good ARYSTAT_UNCOMFIG Unonfigured ARYSTAT_REBUILD Rebuilding ARYSTAT_REDUCE Reduced ARYSTAT_DEAD Dead
Processing	Gets SoftMirror Status
Example	<pre>BOOL ret; int Status; // Gets Software Mirroring Status. ret = GetSmiAryStatus(&Status);</pre>

GetSmiDevStatus

Call Format	BOOL WINAPI GetSmiDevStatus(int Id,int *pType,int *pStatus)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Id Device ID 0 : Master HDD 1 : Slave HDD (I/O) int *pType Device Type ATADEVICE ATA DEVICE ATAPIDEVICE CD-ROM NODEVICE No DEVICE (I/O) int *pStatus Device Status DEVSTAT_GOOD Good DEVSTAT_NOTEXIST No DEVICE DEVSTAT_BROKEN BROKEN
Processing	Gets Software Mirroring Device Status
Example	<pre>BOOL ret; int Id, Type, Status; // Gets the device status Id = 0; ret = GetSmiDevStatus(Id ,&Type ,&Status);</pre>

A.7.5 Visual C++ Functions

Function Name	Description
GetDrvHandle	Gets the driver handle
CloseDrvHandle	Destroys the driver handle
GetDrvVersion	Gets the driver version
GetMonitorSetup	Gets the monitoring enabled/disabled setting
GetVoltParam	Gets the voltage monitoring parameter
GetCurrentVolt	Gets the current voltage value
GetFanParam	Gets the fan monitoring parameter
GetCurrentFan	Gets the current fan value
GetTempParam	Gets the temperature monitoring parameter
GetCurrentTemp	Gets the current temperature value
GetWdtCounter	Gets the watchdog timer counter
SetWdtMask	Sets warning masking in case of watchdog timer time-out
GetWdtMask	Gets warning masking in case of watchdog timer time-out
StartWdt	Starts the watchdog timer
StopWdt	Stops the watchdog timer
RestartWdt	Restarts the watchdog timer
RunningWdt	Gets the watchdog timer operation status
SetWarningOut	Sets warning output
GetWarningOut	Gets warning output
GetUniversalln	Gets universal input
ClearUniversalln	Clears the universal input latched status
SetUniversallnMask	Sets universal input masking
GetUniversallnMask	Gets universal input masking
SetResetMask	Sets reset-masking
GetResetMask	Gets reset-masking
SetIdeErr	Sets software mirroring error
GetIdeErrHard	Gets hardware mirroring error
GetLightblowErr	Get BackLight Err status
GetEvent	Gets the error event
ClearEvent	Clears the error event
StartInsideBuzzer	Starts PL unit's internal buzzer
StopInsideBuzzer	Stops PL unit's internal buzzer
ChkInsideBuzzer	Checks PL unit's internal buzzer
GetWdtTimeout	Gets watchdog timeout status
ClearWdtTimeout	Clears the watchdog timeout status
SetWarningDOUT	Sets warning DOUT
GetWarningDOUT	Gets warning DOUT
GetSmiDrvHandle	Gets Software Mirroring driver handle
CloseSmiDrvHandle	Gets Software Mirroring Status
GetSmiAryStatus	Gets Software Mirroring feature Status
GetSmiDevStatus	Gets Software Mirroring Device Status

A.7.6 Visual C++ Function Specifications (Details)

GetDrvHandle

Call Format	int GetDrvHandle(void)
Return Value	0: Normal 1: Error
Arguments	None
Processing	Gets the device driver handle to communicate with the device driver. The handle Getsed is stored into the member variable m_handle.
Example 1	<pre>CPL_Iocctl m_Ioc; m_Ioc.GetDrvHandle();</pre>
Example 2	<pre>int ret; HANDLE hndl; ret = ::GetDrvHandle(&hndl);</pre>



Note: An error occurs if the System Monitor/RAS Device Driver is not running.

CloseDrvHandle

Call Format	BOOL CloseDrvHandle(void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None
Processing	Destroys the device driver handle created using the GetDrvHandle function.
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; // Destroys the device driver handle. ret = m_Ioc.CloseDrvHandle();</pre>
Example 2	<pre>BOOL ret; // Destroys the device driver handle. ret = ::CloseDrvHandle();</pre>

GetDrvVersion

Call Format	BOOL GetDrvVersion(int *pMajor, int *pMinor)	
Return Value	TRUE: Normal	
	FALSE: Error	
Arguments	(I/O) int *pMajor	Pointer to version information (Major, 0 to 99).
	(I/O) int *pMinor	Pointer to version information (Minor, 0 to 99).
Processing	Gets the driver's version information.	
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int Major, Minor; ret = m_Ioc.GetDrvVersion(&Major, &Minor);</pre>	
Example 2	<pre>BOOL ret; int Major, Minor; ret = ::GetDrvVersion(&Major, &Minor);</pre>	



Note:

If the version is 1.10, then you will get

Major: 1 (decimal)

Minor: 10 (decimal).

Appendices

GetMonitorSetup

Call Format	BOOL GetMonitorSetup(int Selector, int *pSetup)	
Return Value	TRUE: Normal FALSE: Error	
Arguments	(I)intSelector	Parameters
		MONITOR_VOLT_CPU CPU core voltage
		MONITOR_VOLT_P33 +3.3 V
		MONITOR_VOLT_P50 +5.0 V
		MONITOR_VOLT_P12 +12 V
		MONITOR_VOLT_M12 -12 V
		MONITOR_VOLT_M50 -5.0 V
		MONITOR_TEMP_SYSTEM System temperature
		MONITOR_TEMP_CPU CPU temperature
		MONITOR_TEMP_OPT Option temperature
		MONITOR_FAN_CPU CPU fan
		MONITOR_FAN_POWER Power fan
		MONITOR_FAN_OPT Option fan
	(I/O) int *pSetup	Pointer to Getsed Data
		0: Disabled
		1: Enabled
Processing	Gets the current monitoring enabled/disabled status.	
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int Setup; // Gets the CPU core voltage setup status. ret = m_Ioc.GetMonitorSetup(MONITOR_VOLT_CPU, &Setup);</pre>	
Example 2	<pre>BOOL ret; int Setup; // Get the CPU core voltage setup status. ret = ::GetMonitorSetup(MONITOR_VOLT_CPU, &Setup);</pre>	

GetVoltParam

Call Format `BOOL GetVoltParam(int Selector, int *pULimit, int *pLLimit)`

Return Value TRUE: Normal

FALSE: Error

Arguments

(I) int Selector Parameters

MONITOR_VOLT_CPU CPU core voltage

MONITOR_VOLT_P33 +3.3 V

MONITOR_VOLT_P50 +5.0 V

MONITOR_VOLT_P12 +12 V

MONITOR_VOLT_M12 -12 V

MONITOR_VOLT_M50 -5.0 V

(I/O) int *pULimit Pointer to upper-limit voltage value (Unit: mV)

(I/O) int *pLLimit Pointer to lower-limit voltage value (Unit: mV)

Processing

Gets the voltage monitoring parameter.

Example 1

```
CPL_Iocctl m_Ioc;
```

```
BOOL ret;
```

```
int ULimit, LLimit;
```

```
//Get the upper and lower-limit values of the CPU core voltage.
```

```
ret = m_Ioc.GetVoltParam( MONITOR_VOLT_CPU,
```

```
&ULimit, &LLimit );
```

Example 2

```
BOOL ret;
```

```
int ULimit, LLimit;
```

```
//Get the upper and lower-limit values of the CPU core voltage.
```

```
ret = ::GetVoltParam( MONITOR_VOLT_CPU, &ULimit, &LLimit );
```



Since the data taken from this function is shown in mV units, the following conversion is needed for use in (Volt) units:

Data in Volt unit = Data in mV unit/1000

Appendices

GetCurrentVolt

Call Format	BOOL GetCurrentVolt(int Selector, int *pData)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Parameters MONITOR_VOLT_CPU CPU core voltage MONITOR_VOLT_P33 +3.3 V MONITOR_VOLT_P50 +5.0 V MONITOR_VOLT_P12 +12 V MONITOR_VOLT_M12 -12 V MONITOR_VOLT_M50 -5.0 V (I/O) int *pData Pointer to the voltage value (Unit: mV)
Processing	Gets the current voltage value.
Example 1	<pre>CPL_Ioc1 m_Ioc; BOOL ret; int Data; // Get the CPU core voltage value. ret = m_Ioc.GetCurrentVolt(MONITOR_VOLT_CPU, &Data);</pre>
Example 2	<pre>BOOL ret; int Data; // Get the CPU core voltage value. ret = ::GetCurrentVolt(MONITOR_VOLT_CPU, &Data);</pre>



Since the data taken from this function is shown in mV units, the following conversion is needed for use in (Volt) units:

Data in Volt unit = Data in mV unit/1000

GetFanParam

Call Format	BOOL GetFanParam (int Selector, int *pLLimit)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Parameters MONITOR_FAN_CPU CPU fan MONITOR_FAN_POWER Power fan MONITOR_FAN_OPT Option fan (I/O) int *pLLimit Pointer to the lower-limit fan rotation speed (Unit: RPM) (RPM: Revolutions Per Minute)
Processing	Gets the fan monitoring parameter.
Example 1	<pre>CPL_Ioctl m_Ioc; BOOL ret; int LLimit; // Get the lower-limit CPU fan rotational speed. ret=m_Ioc.GetFanParam(MONITOR_FAN_CPU, &LLimit);</pre>
Example 2	<pre>BOOL ret; int LLimit; // Get the lower-limit CPU fan rotation speed. ret=::GetFanParam(MONITOR_FAN_CPU, &LLimit);</pre>

Appendices

GetCurrentFan

Call Format	BOOL GetCurrentFan(int Selector, int *pData)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Parameters MONITOR_FAN_CPU CPU fan MONITOR_FAN_POWER Power fan MONITOR_FAN_OPT Option fan (I/O) int *pData Pointer to the fan rotation speed (Unit: RPM) (RPM: Revolutions Per Minute)
Processing	Gets the current fan rotation speed.
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int Data; // Get the CPU fan rotational speed. ret = m_Ioc.GetCurrentFan(MONITOR_FAN_CPU, &Data);</pre>
Example 2	<pre>BOOL ret; int Data; // Get the CPU fan rotational speed. ret = ::GetCurrentFan(MONITOR_FAN_CPU, &Data);</pre>

GetTempParam

Call Format `BOOL GetTempParam(int Selector, int *pULimit)`
 Return Value `TRUE: Normal`
 `FALSE: Error`
 Arguments `(I)intSelector` Parameters
 `MONITOR_TEMP_SYSTEM` System temperature
 `MONITOR_TEMP_CPU` CPU temperature
 `MONITOR_TEMP_OPT` Option temperature
 `(I/O)int*pULimit` Pointer to the upper-limit temperature
 (Unit: Degrees Celsius)

Processing Gets the temperature monitoring parameter.

Example 1 `CPL_Iocctl m_Ioc;`
 `BOOL ret;`
 `int ULimit;`
 `// Get the system temperature upper-limit value.`
 `ret = m_Ioc.GetTempParam(MONITOR_TEMP_SYSTEM,`
 `&ULimit);`

Example 2 `BOOL ret;`
 `int ULimit;`
 `ret = ::GetTempParam(MONITOR_TEMP_SYSTEM, &ULimit);`

GetCurrentTemp

Call Format `BOOL GetCurrentTemp(int Selector, int *pData)`
 Return Value `TRUE: Normal`
 `FALSE: Error`
 Arguments `(I)intSelector` Parameters
 `MONITOR_TEMP_SYSTEM` System temperature
 `MONITOR_TEMP_CPU` CPU temperature
 `MONITOR_TEMP_OPT` Option temperature
 `(I/O)int*pData` Pointer to the temperature
 (Unit: Degrees Celsius)

Processing Gets the current temperature value.

Example 1 `CPL_Iocctl m_Ioc;`
 `BOOL ret;`
 `int Data;`
 `// Gets the system temperature value.`
 `ret = m_Ioc.GetCurrentTemp(MONITOR_TEMP_SYSTEM, &Data);`

Example 2 `BOOL ret;`
 `int Data;`
 `// Gets the system temperature value.`
 `ret = ::GetCurrentTemp(MONITOR_TEMP_SYSTEM, &Data);`

Appendices

GetWdtCounter

Call Format	BOOL GetWdtCounter(int *pCounter)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I/O) int *pCounter Pointer to the watchdog timer's initial counter value (Unit: Seconds)
Processing	Gets the current watchdog timer's initial counter value.
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int Counter; ret = m_Ioc.GetWdtCounter(&Counter);</pre>
Example 2	<pre>BOOL ret; int Counter; ret = ::GetWdtCounter(&Counter);</pre>

SetWdtMask

Call Format	BOOL SetWdtMask(int Selector, int Mask)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Setting Item WARNING_LAMP LAMP WARNING_ALARM ALARM (I) int Mask Masking Information MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Sets masking for the warning that is output when watchdog timer time-out occurs.
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; // Enable masking for LAMP output. ret = m_Ioc.SetWdtMask(WARNING_LAMP, MASK_ON); // Disable masking for ALARM output. ret = m_Ioc.SetWdtMask(WARNING_ALARM, MASK_OFF);</pre>
Example 2	<pre>BOOL ret; // Enable masking for LAMP output. ret = ::SetWdtMask(WARNING_LAMP, MASK_ON); // Disable masking for ALARM output. ret = ::SetWdtMask(WARNING_ALARM, MASK_OFF);</pre>

GetWdtMask

Call Format	BOOL GetWdtMask(int Selector, int *pMask)	
Return Value	TRUE: Normal FALSE: Error	
Arguments	(I) int Selector	Setting Item WARNING_LAMP LAMP WARNING_ALARM ALARM
	(I/O) int *pMask	Pointer to Masking Information MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Gets the masking information for warning output that is created when a watchdog timer time-out occurs.	
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int Mask; // Gets the LAMP masking information. ret = m_Ioc.GetWdtMask(WARNING_LAMP, &Mask); // Get the ALARM masking information. ret = m_Ioc.GetWdtMask(WARNING_ALARM, &Mask);</pre>	
Example 2	<pre>BOOL ret; int Mask; // Gets the LAMP0 masking information. ret = ::GetWdtMask(WARNING_LAMP, &Mask); // Get the ALARM masking information. ret = ::GetWdtMask(WARNING_ALARM, &Mask);</pre>	

Appendices

StartWdt

Call Format	BOOL StartWdt(void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None
Processing	Starts watchdog timer countdown.
Example 1	CPL_Iocctl m_Ioc; BOOL ret; ret = m_Ioc.StartWdt();
Example 2	BOOL ret; ret = ::StartWdt();

StopWdt

Call Format	BOOL StopWdt(void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None
Processing	Stops watchdog timer countdown.
Example 1	CPL_Iocctl m_Ioc; BOOL ret; ret = m_Ioc.StopWdt();
Example 2	BOOL ret; ret = ::StopWdt();

RestartWdt

Call Format	BOOL RestartWdt(void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None
Processing	Restarts watchdog timer countdown after resetting to the initialvalue.
Example 1	CPL_Iocctl m_Ioc; BOOL ret; m_Ioc.RestartWdt();
Example 2	BOOL ret; ret = ::RestartWdt();



RestartWdt can only be used after the StartWdt countdown has started.If RestartWdt is used after the timeout,it should be after ClearWdtTimeout has used to clear the timeout condition and after StartWdt has started the countdown.

RunningWdt

Call Format	BOOL RunningWdt(int *pRunFlag)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I/O) int *pRunFlag Pointer to Watchdog Timer Operation Status WATCHDOG_STOP Stopped WATCHDOG_COUNTDOWN Countdown in progress
Processing	Gets the watchdog timer's operation status.
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int RunFlag; ret = m_Ioc.RunningWdt(&RunFlag);</pre>
Example 2	<pre>BOOL ret; int RunFlag; ret = ::RunningWdt(&RunFlag);</pre>

SetWarningOut

Call Format	BOOL SetWarningOut(int Selector, int WarnOut)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Setting Item WARNING_LAMP LAMP WARNING_ALARM ALARM (I) int WarnOut Output Status OUTPUT_OFF Output OFF OUTPUT_ON Output ON
Processing	Sets the warning information for the set item (lamp or alarm).
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; // Sets the LAMP output status to ON. ret = m_Ioc.SetWarningOut(WARNING_LAMP, OUTPUT_ON); // Sets the ALARM output status to OFF. ret = m_Ioc.SetWarningOut(WARNING_ALARM, OUTPUT_OFF);</pre>
Example 2	<pre>BOOL ret; // Sets the LAMP output status to ON. ret = ::SetWarningOut(WARNING_LAMP, OUTPUT_ON); // Sets the ALARM output status to OFF. ret = ::SetWarningOut(WARNING_ALARM, OUTPUT_OFF);</pre>

Appendices

GetWarningOut

Call Format	BOOL GetWarningOut(int Selector, int *pWarnOut)	
Return Value	TRUE: Normal FALSE: Error	
Arguments	(I) int Selector	Setting Item WARNING_LAMP LAMP WARNING_ALARM ALARM
	(I/O) int *pWarnOut	Pointer to Output Status OUTPUT_OFF Output OFF OUTPUT_ON Output ON
Processing	Gets the warning status of the current set item (LAMP or ALARM).	
Example 1	<pre>CPL_IocI m_Ioc; BOOL ret; int WarnOut; // Gets the LAMP output status. ret = m_Ioc.GetWarningOut(WARNING_LAMP, &WarnOut); // Gets the ALARM output status. ret = m_Ioc.GetWarningOut(WARNING_ALARM, &WarnOut);</pre>	
Example 2	<pre>BOOL ret; int WarnOut; // Gets the LAMP output status. ret = ::GetWarningOut(WARNING_LAMP, &WarnOut); // Gets the ALARM output status. ret = ::GetWarningOut(WARNING_ALARM, &WarnOut);</pre>	

GetUniversalIn

Call Format	BOOL GetUniversalIn(int Selector, int *pUniIn)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Designated Port PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1 (I/O) int *pUniIn Pointer to Input Status INPUT_OFF Input OFF INPUT_ON Input ON
Processing	Gets the input status of the designated port (Universal Input 0, Universal Input 1).
Example 1	<pre> CPL_Iocctl m_Ioc; BOOL ret; int UniIn; // Gets the input status of Universal Input 0. ret = m_Ioc.GetUniversalIn(PORT_UNI0, &UniIn); // Gets the input status of Universal Input 1. ret = m_Ioc.GetUniversalIn(PORT_UNI1, &UniIn); </pre>
Example 2	<pre> BOOL ret; int UniIn; // Gets the input status of Universal Input 0. ret = ::GetUniversalIn(PORT_UNI0, &UniIn); // Gets the input status of Universal Input 1. ret = ::GetUniversalIn(PORT_UNI1, &UniIn); </pre>

Appendices

ClearUniversalIn

Call Format	BOOL ClearUniversalIn(int Selector)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Designated Port PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1
Processing	Clears the input status of the designated port (Universal Input 0, Universal Input 1).
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; // Cancels the output of Universal Input 0. ret = m_Ioc.ClearUniversalIn(PORT_UNI0); // Cancels the output of Universal Input 1. ret = m_Ioc.ClearUniversalIn(PORT_UNI1);</pre>
Example 2	<pre>BOOL ret; // Cancels the output of Universal Input 0. ret = ::ClearUniversalIn(PORT_UNI0); // Cancels the output of Universal Input 1. ret = ::ClearUniversalIn(PORT_UNI1);</pre>

SetUniversalInMask

Call Format	BOOL SetUniversalInMask(int Selector, int Mask)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Designated Port PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1 (I/O) int Mask Masking Information MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Sets the masking information for the designated port (Universal Input 0, Universal Input 1).
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; // Disable masking for Universal Input 0. ret = m_Ioc.SetUniversalInMask(PORT_UNI0, MASK_OFF); // Enable masking for Universal Input 1. ret = m_Ioc.SetUniversalInMask(PORT_UNI1, MASK_ON);</pre>
Example 2	<pre>BOOL ret; // Disable masking for Universal Input 0. ret = ::SetUniversalInMask(PORT_UNI0, MASK_OFF); // Enable masking for Universal Input 1. ret = ::SetUniversalInMask(PORT_UNI1, MASK_ON);</pre>

Appendices

GetUniversalInMask

Call Format	BOOL GetUniversalInMask(int Selector, int *pMask)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Designated Port PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1 (I/O) int *pMask Pointer to Masking Information MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Gets the masking information for the designated port (Universal Input 0, Universal Input 1).
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int Mask; // Gets the masking information for Universal input 0. ret = m_Ioc.GetUniversalInMask(PORT_UNI0, &Mask); // Gets the masking information for Universal input 1. ret = m_Ioc.GetUniversalInMask(PORT_UNI1, &Mask);</pre>
Example 2	<pre>BOOL ret; int Mask; // Gets the masking information for Universal input 0. ret = ::GetUniversalInMask(PORT_UNI0, &Mask); // Gets the masking information for Universal input 1. ret = ::GetUniversalInMask(PORT_UNI1, &Mask);</pre>

SetResetMask

Call Format	BOOL SetResetMask(int Mask)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Mask Masking Information MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Sets reset-masking.
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; //Disable reset-masking. ret = m_Ioc.SetResetMask(MASK_OFF);</pre>
Example 2	<pre>BOOL ret; //Disable reset-masking. ret = ::SetResetMask(MASK_OFF);</pre>

GetResetMask

Call Format	BOOL GetResetMask(int *pMask)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I/O) int *pMask Pointer to Masking Information MASK_OFF Masking disabled MASK_ON Masking enabled
Processing	Gets the current reset-masking information.
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int Mask; ret = m_Ioc.GetResetMask(&Mask);</pre>
Example 2	<pre>BOOL ret; int Mask; ret = ::GetResetMask(&Mask);</pre>

Appendices

SetIdeErr

Call Format	BOOL SetIdeErr(int IdeErr)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int IdeErr Error Output Information IDE_ERROR_OFF Error Output OFF IDE_ERROR_ON Error Output ON
Processing	Uses software control to set IDE error output.
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; ret = m_Ioc.SetIdeErr(IDE_ERROR_OFF);</pre>
Example 2	<pre>BOOL ret; ret = ::SetIdeErr(IDE_ERROR_OFF);</pre>

GetIdeErrHard

Call Format	BOOL GetIdeErrHard(int Selector, int *pIdeErr)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int Selector Parameters IDE_ERROR_1 IDE_ERR1 IDE_ERROR_2 IDE_ERR2 (I/O) int *pIdeErr Pointer to Output Status IDE_ERROR_OFF Normal IDE_ERROR_ON Error
Processing	Gets hardware's current IDE error signal.
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int IdeErr; // Gets the IDE_ERR1 signal status ret = m_Ioc.GetIdeErrHard(IDE_ERROR_1, &IdeErr);</pre>
Example 2	<pre>BOOL ret; int IdeErr; // Gets the IDE_ERR1 signal status ret = ::GetIdeErrHard(IDE_ERROR_1, &IdeErr);</pre>

GetLightblowErr

Call Format	<pre> BOOL GetLightblowErr(int *pLightErr) TRUE: Normal FALSE: Error </pre>
Arguments	<pre> (I/O) int *pLightErr Error Information BACKLIGHT_OK OK BACKLIGH_ERR NG </pre>
Processing	Gets Backlight's current burnout error output.
Example 1	<pre> CPL_Iocctl m_Ioc; BOOL ret; int LightErr; // Gets backlight's burnout condition. ret = m_Ioc.GetLightblowErr(&LightErr); </pre>
Example 2	<pre> BOOL ret; int LightErr; // Gets backlight's burnout condition. ret = ::GetLightblowErr(&LightErr); </pre>

Appendices

GetEvent

Call Format `BOOL GetEvent(int Selector, int *pEvent)`

Return Value `TRUE`: Normal

`FALSE`: Error

Arguments

(I) int Selector

Parameters

<code>EVENT_VOLT_CPU</code>	CPU core voltage
<code>EVENT_VOLT_P33</code>	+3.3 V
<code>EVENT_VOLT_P50</code>	+5.0 V
<code>EVENT_VOLT_P12</code>	+12 V
<code>EVENT_VOLT_M12</code>	-12 V
<code>EVENT_VOLT_M50</code>	-5.0 V
<code>EVENT_FAN_CPU</code>	CPU fan
<code>EVENT_FAN_POWER</code>	Power fan
<code>EVENT_FAN_OPT</code>	Option fan
<code>EVENT_TEMP_SYSTEM</code>	System temperature
<code>EVENT_TEMP_CPU_OPT</code>	CPU or option temperature
<code>EVENT_UNI_IN0</code>	Universal input 0
<code>EVENT_UNI_IN1</code>	Universal input 1
<code>EVENT_WDT_TIMEOUT</code>	Watchdog Timeout

(I/O) int *pEvent Pointer to Error Event Information

`ERROR_EVENT_OFF` Without error event

`ERROR_EVENT_ON` With error event

Processing

Checks the machine for voltage, fan, and temperature errors, and the Universal Input information (event) and Watchdog Timeout error.

Example 1

```
CPL_IocI m_Ioc;
BOOL ret;
int Event;
// Gets the error event information for the CPU core voltage.
ret = m_Ioc.GetEvent( EVENT_VOLT_CPU, &Event );
```

Example 2

```
BOOL ret;
int Event;
// Gets the error event information for the CPU core voltage.
ret = ::GetEvent( EVENT_VOLT_CPU, &Event );
```

ClearEvent

Call Format	BOOL ClearEvent(int Selector)																												
Return Value	TRUE: Normal FALSE: Error																												
Arguments	(I) int Selector Designated Parameters for ClearEvent <table> <tr> <td>EVENT_VOLT_CPU</td> <td>CPU core voltage</td> </tr> <tr> <td>EVENT_VOLT_P33</td> <td>+3.3 V</td> </tr> <tr> <td>EVENT_VOLT_P50</td> <td>+5.0 V</td> </tr> <tr> <td>EVENT_VOLT_P12</td> <td>+12 V</td> </tr> <tr> <td>EVENT_VOLT_M12</td> <td>-12 V</td> </tr> <tr> <td>EVENT_VOLT_M50</td> <td>-5.0 V</td> </tr> <tr> <td>EVENT_FAN_CPU</td> <td>CPU fan</td> </tr> <tr> <td>EVENT_FAN_POWER</td> <td>Power fan</td> </tr> <tr> <td>EVENT_FAN_OPT</td> <td>Option fan</td> </tr> <tr> <td>EVENT_TEMP_SYSTEM</td> <td>System temperature</td> </tr> <tr> <td>EVENT_TEMP_CPU_OPT</td> <td>CPU or option temperature</td> </tr> <tr> <td>EVENT_UNI_IN0</td> <td>Universal input 0</td> </tr> <tr> <td>EVENT_UNI_IN1</td> <td>Universal input 1</td> </tr> <tr> <td>EVENT_WDT_TIMEOUT</td> <td>Watchdog Timeout</td> </tr> </table>	EVENT_VOLT_CPU	CPU core voltage	EVENT_VOLT_P33	+3.3 V	EVENT_VOLT_P50	+5.0 V	EVENT_VOLT_P12	+12 V	EVENT_VOLT_M12	-12 V	EVENT_VOLT_M50	-5.0 V	EVENT_FAN_CPU	CPU fan	EVENT_FAN_POWER	Power fan	EVENT_FAN_OPT	Option fan	EVENT_TEMP_SYSTEM	System temperature	EVENT_TEMP_CPU_OPT	CPU or option temperature	EVENT_UNI_IN0	Universal input 0	EVENT_UNI_IN1	Universal input 1	EVENT_WDT_TIMEOUT	Watchdog Timeout
EVENT_VOLT_CPU	CPU core voltage																												
EVENT_VOLT_P33	+3.3 V																												
EVENT_VOLT_P50	+5.0 V																												
EVENT_VOLT_P12	+12 V																												
EVENT_VOLT_M12	-12 V																												
EVENT_VOLT_M50	-5.0 V																												
EVENT_FAN_CPU	CPU fan																												
EVENT_FAN_POWER	Power fan																												
EVENT_FAN_OPT	Option fan																												
EVENT_TEMP_SYSTEM	System temperature																												
EVENT_TEMP_CPU_OPT	CPU or option temperature																												
EVENT_UNI_IN0	Universal input 0																												
EVENT_UNI_IN1	Universal input 1																												
EVENT_WDT_TIMEOUT	Watchdog Timeout																												
Processing	Cancels the error event.																												
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; // Cancels the error event for the CPU core voltage. ret = m_Ioc.ClearEvent(EVENT_VOLT_CPU);</pre>																												
Example 2	<pre>BOOL ret; // Cancels the error event for the CPU core voltage. ret = ::ClearEvent(EVENT_VOLT_CPU);</pre>																												

Appendices

StartInsideBuzzer

Call Format	BOOL StartInsideBuzzer (int hz, int ms)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int hz Buzzer frequency (Hz) (I) int ms Buzzer length (ms)
Processing	Starts the PL unit's internal buzzer, based on the designated frequency and length.
Example 1	BOOL ret; int hz = 600; int ms = 1000; // PL internal buzzer will sound at 600MHz for 1 second. ret = m_Ioc.StartInsideBuzzer (hz, ms);
Example 2	BOOL ret; int hz = 600; int ms = 1000; // PL internal buzzer will sound at 600MHz for 1 second. ret = ::StartInsideBuzzer (hz, ms);



Note: This feature cannot be used with a PL running WindowsNT® 4.0 or Windows® 2000 due to the use of Windows® 98 functions.

StopInsideBuzzer

Call Format	BOOL StopInsideBuzzer (void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None.
Processing	Stops the PL unit's internal buzzer.
Example 1	CPL_Ioc1 m_loc; BOOL ret; // Stops PL internal buzzer. ret = m_Ioc.StopInsideBuzzer ();
Example 2	BOOL ret; // Stops PL internal buzzer. ret = ::StopInsideBuzzer ();



Note: This feature cannot be used with a PL running WindowsNT® 4.0 or Windows® 2000 due to the use of Windows® 98 functions.

ChkInsideBuzzer

Call Format	BOOL ChkInsideBuzzer (int *BuzzerParam)		
Return Value	TRUE: Normal FALSE: Error		
Arguments	(I/O) int *BuzzerParam	Buzzer Status Pointer	
		BUZZER_ON	Buzzer is ON
		BUZZER_OFF	Buzzer is OFF
Processing	Checks the buzzer's operation status.		
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int BuzzerParam; // Checks buzzer status. ret = m_Ioc.ChkInsideBuzzer (&BuzzerParam);</pre>		
Example 2	<pre>BOOL ret; // Checks buzzer status. ret = ::ChkInsideBuzzer (&BuzzerParam);</pre>		



This feature cannot be used with a PL unit running WindowsNT® 4.0 or Windows® 2000, due to the use of Windows® 98 functions.

GetWdtTimeout

Call Format	BOOL GetWdtTimeout(int *pTimebuf)		
Return Value	TRUE: Normal FALSE: Error		
Arguments	(I/O) int *pTimebuf	Pointer to Watchdog Status	
		TIMEOUT_OK	Not timeout
		TIMEOUT_ERROR	Now timeout
Processing	Gets watchdog timeout status.		
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int Timebuf; // Gets watchdog timeout status. ret = Gm_Ioc.GetWdtTimeout(&Timebuf);</pre>		
Example 2	<pre>BOOL ret; int Timebuf; // Gets watchdog timeout status. ret = ::GetWdtTimeout(&Timebuf);</pre>		

Appendices

ClearWdtTimeout

Call Format	BOOL ClearWdtTimeout(void)
Return Value	TRUE: Normal FALSE: Error
Arguments	None
Processing	Clears the watchdog timeout status.
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; // Clears the watchdog timeout status. ret = m_Ioc.GetWdtTimeout();</pre>
Example 2	<pre>BOOL ret; // Clears the watchdog timeout status. ret = ::GetWdtTimeout();</pre>

SetWarningDOUT

Call Format	BOOL SetWarningDOUT(int WarningOut)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int WarningOut Output status OUTPUT_OFF Output OFF OUTPUT_ON Output ON
Processing	Sets alarm status of DOUT.
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; // Sets DOUT output status to OFF. ret = m_Ioc.SetWarningDOUT(OUTPUT_OFF);</pre>
Example 2	<pre>BOOL ret; // Sets DOUT output status to OFF. ret = ::SetWarningDOUT(OUTPUT_OFF);</pre>

GetWarningDOUT

Call Format	BOOL GetWarningDOUT(int *pWarningOut)	
Return Value	TRUE: Normal FALSE: Error	
Arguments	(I/O) int *pWarningOut	Pointer to Output Status OUTPUT_OFF Output OFF OUTPUT_ON Output ON
Processing	Gets alarm status of DOUT.	
Example 1	<pre>CPL_Iocctl m_Ioc; BOOL ret; int WarningOut; // Gets alarm status of DOUT. ret = m_Ioc.GetWarningDOUT(&WarningOut);</pre>	
Example 2	<pre>BOOL ret; int WarningOut; // Gets alarm status of DOUT. ret = ::GetWarningDOUT(&WarningOut);</pre>	

GetSmiDrvHandle

Call Format	intGetSmiDrvHandle(void)	
Return Value	0: Normal 1 : Error	
Arguments	NONE	
Processing	Gets device driver handle for communication with Software Mirroring device driver.	
Example 1	<pre>CPL_SmiIoctl m_SmiIoc; BOOL ret; // Gets Software Mirroring driver handle. ret = m_SmiIoc.GetSmiDrvHandle();</pre>	
Example 2	<pre>BOOL ret; // Gets Software Mirroring driver handle. ret = ::GetSmiDrvHandle();</pre>	



Note:

When the Software Mirroring Driver is not loaded, an error is returned.

Appendices

CloseSmiDrvHandle

Call Format	BOOL CloseSmiDrvHandle(void)
Return Value	True: Normal False: Error
Arguments	NONE
Processing	Destroys handle created in GetSmiDrvHandle.
Example 1	<pre>CPL_SmiIoctl m_SmiIoc; BOOL ret; //Destroys Software Mirroring driver handle. ret=m_SmiIoc.CloseSmiDrvHandle();</pre>
Example 2	<pre>BOOL ret; //Destroys Software Mirroring driver handle. ret=::CloseSmiDrvHandle();</pre>

GetSmiAryStatus

Call Format	BOOL GetSmiAryStatus(int *pStatus)										
Return Value	TRUE: Normal FALSE: Error										
Arguments	(I/O) int *pStatus Pointer to Mirroring Status <table><tr><td>ARYSTAT_GOOD</td><td>Good</td></tr><tr><td>ARYSTAT_UNCONFIG</td><td>Unconfigured</td></tr><tr><td>ARYSTAT_REBUILD</td><td>Rebuilding</td></tr><tr><td>ARYSTAT_REDUCE</td><td>Reduced</td></tr><tr><td>ARYSTAT_DEAD</td><td>Dead</td></tr></table>	ARYSTAT_GOOD	Good	ARYSTAT_UNCONFIG	Unconfigured	ARYSTAT_REBUILD	Rebuilding	ARYSTAT_REDUCE	Reduced	ARYSTAT_DEAD	Dead
ARYSTAT_GOOD	Good										
ARYSTAT_UNCONFIG	Unconfigured										
ARYSTAT_REBUILD	Rebuilding										
ARYSTAT_REDUCE	Reduced										
ARYSTAT_DEAD	Dead										
Processing	Gets Software Mirroring status.										
Example 1	<pre>CPL_SmiIoctl m_Smiloc; BOOL ret; int Status; // Gets Software Mirroring status. ret=m_Smiloc.GetSmiAryStatus(&Status);</pre>										
Example 2	<pre>BOOL ret; int Status; // Gets Software Mirroring status. ret=::GetSmiAryStatus(&Status);</pre>										

GetSmiDevStatus

Call Format	BOOL GetSmiDevStatus(int Id ,int *pType ,int *pStatus)		
Return Value	TRUE: Normal FALSE: Error		
Arguments	(I) int Id	Device ID 0 : Master HDD 1 : Slave HDD	
	(I/O int* pType	Device Type	
		ATADEVICE	ATA DEVICE
		ATAPIDEVICE	CD-ROM
		NODEVICE	No DEVICE
	(I/O) int* pStatus	Device Status	
		DEVSTAT_GOOD	Good
		DEVSTAT_NOTEXIST	No DEVICE
		DEVSTAT_BROKEN	BROKEN
Processing	Gets Device Status of software mirroring.		
Example 1	<pre>CPL_SmiIoctl m_SmiIoc; BOOL ret; int Id, Type, Status; // Gets device status. Id = 0; ret = m_SmiIoc.GetSmiDevStatus(ID ,&Type ,&Status);</pre>		
Example 2	<pre>BOOL ret; int Id, Type, Status; // Gets device status. Id = 0; ret = ::GetSmiDevStatus(ID ,&Type ,&Status);</pre>		

A.7.7 Visual Basic Functions

Function Name	Description
InitIoctl	Creates a CPL_Ioctl object
EndIoctl	Destroys a CPL_Ioctl object
GetDrvHandle	Gets the driver handle
CloseDrvHandle	Destroys the driver handle
GetDrvVersion	Gets the driver version
GetMonitorSetup	Gets the enabled/disabled monitor settings
GetVoltParam	Gets the voltage monitoring parameters
GetCurrentVolt	Gets the current value of the voltage
GetFanParam	Gets the parameters for monitoring the FAN
GetCurrentFan	Gets the current value of the FAN
GetTempParam	Gets the parameters for monitoring the temperature
GetCurrentTemp	Gets the current value of the temperature
GetWdtCounter	Gets the watchdog timer counter
SetWdtMask	Sets the watchdog timer counter time-out status warning mask
GetWdtMask	Gets the watchdog timer counter time-out status warning mask
StartWdt	Starts the watchdog timer
StopWdt	Stops the watchdog timer
RestartWdt	Restarts the watchdog timer
RunningWdt	Gets the watchdog status
SetWarningOut	Sets the warning output
GetWarningOut	Gets the warning output
GetUniversalln	Gets the universal input
ClearUniversalln	Clears the universal input latch
SetUniversallnMask	Sets the universal input mask
GetUniversallnMask	Gets the universal input mask
SetResetMask	Sets the reset mask
GetResetMask	Gets the reset mask
SetIdeErr	Sets the mirroring error (software error)
GetIdeErrHard	Gets the mirroring error (hardware error)
GetEvent	Gets an error event
ClearEvent	Clears an error event
StartInsideBuzzer	Starts PL internal buzzer
StopInsideBuzzer	Stops PL internal buzzer
ChkInsideBuzzer	Checks PL internal buzzer
GetWdtTimeout	Gets the time-out status of the watchdog timer
ClearWdtTimeout	Clear the time-out status of the watchdog timer
SetWarningDOUT	Sets the warning output DOUT
GetWarningDOUT	Gets the warning output DOUT
GetSmiDrvHandle	Gets Software Mirroring driver handle
CloseSmiDrvhandle	Destroys Software Mirroring driver handle
GetSmiAryStatus	Gets status of Software Mirroring Array
GetSmiDevStatus	Gets status of Software Mirroring Device

A.7.8 Visual Basic Function Specifications (Details)

InitIoctl

Call format	Declare Sub InitIoctl Lib "PL_Ioc.dll" ()
Return value	None
Argument	None
Processing	Creates a CPL_Ioctl object. The created object will not be released until the "EndIoctl" function is called.
Example	InitIoctl()

EndIoctl

Call format	Declare Sub EndIoctl Lib "PL_Ioc.dll" ()
Return value	None
Argument	None
Processing	Destroys the object created with the "InitIoctl" function.
Example	EndIoctl()

GetDrvHandle

Call format	Declare Function GetDrvHandle Lib "PL_Ioc.dll" (ByRef hndl As Long) As Long
Return value	0: Normal 1: Error
Argument	hndl As Long Device driver handle (pass by reference)
Processing	Gets the device driver handle to exchange information with the device driver.
Example	Dim ret As Long Dim hndl As Long ret = GetDrvHandle(hndl)



Note: An error will result if the system monitor/RAS device driver is not operating.

CloseDrvHandle

Call format	Declare Function CloseDrvHandle Lib "PL_Ioc.dll"() As Long
Return value	Other than 0: Normal 0: Error
Argument	None
Processing	Destroys the handle acquired with the "GetDrvHandle" function.
Example	Dim ret As Long // Destroy handle ret = CloseDrvHandle()

Appendices

GetDrvVersion

Call format	Declare Function GetDrvVersion Lib "PL_Ioc.dll" (ByRef Major As Long, ByRef Minor As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Major As Long Version data (Major, 0 to 99) (pass by reference) Minor As Long Version data (Major, 0 to 99) (pass by reference)
Processing	Gets the driver version.
Example	Dim ret As Long Dim Major As Long Dim Minor As Long ret = GetDrvVersion(Major, Minor)



When the version is 1.10,

Major:1 (Decimal)
Minor:10 (Decimal)

GetMonitorSetup

Call format	Declare Function GetMonitorSetup Lib "PL_Ioc.dll" (ByVal Selector As Long, ByVal Setup As Long) As Long																																																
Return value	Other than 0: Normal 0: Error																																																
Argument	<table border="0" style="width: 100%;"> <tr> <td style="width: 35%;">Selector As Long</td> <td style="width: 30%;">Parameters (pass by value)</td> <td style="width: 35%;"></td> </tr> <tr> <td></td> <td>MONITOR_VOLT_CPU</td> <td>CPU Core voltage</td> </tr> <tr> <td></td> <td>MONITOR_VOLT_P33</td> <td>+3.3V</td> </tr> <tr> <td></td> <td>MONITOR_VOLT_P50</td> <td>+5.0V</td> </tr> <tr> <td></td> <td>MONITOR_VOLT_P12</td> <td>+12V</td> </tr> <tr> <td></td> <td>MONITOR_VOLT_M12</td> <td>-12V</td> </tr> <tr> <td></td> <td>MONITOR_VOLT_M50</td> <td>-5.0V</td> </tr> <tr> <td></td> <td>MONITOR_TEMP_SYSTEM</td> <td>SYSTEM Temp.</td> </tr> <tr> <td></td> <td>MONITOR_TEMP_CPU</td> <td>CPU Temp.</td> </tr> <tr> <td></td> <td>MONITOR_TEMP_OPT</td> <td>OPTION Temp.</td> </tr> <tr> <td></td> <td>MONITOR_FAN_CPU</td> <td>CPU FAN</td> </tr> <tr> <td></td> <td>MONITOR_FAN_POWER</td> <td>POWER FAN</td> </tr> <tr> <td></td> <td>MONITOR_FAN_OPT</td> <td>OPTION FAN</td> </tr> <tr> <td></td> <td>Setup As Long</td> <td>Get data (pass by reference)</td> </tr> <tr> <td></td> <td></td> <td>0:Disable</td> </tr> <tr> <td></td> <td></td> <td>1:Enable</td> </tr> </table>	Selector As Long	Parameters (pass by value)			MONITOR_VOLT_CPU	CPU Core voltage		MONITOR_VOLT_P33	+3.3V		MONITOR_VOLT_P50	+5.0V		MONITOR_VOLT_P12	+12V		MONITOR_VOLT_M12	-12V		MONITOR_VOLT_M50	-5.0V		MONITOR_TEMP_SYSTEM	SYSTEM Temp.		MONITOR_TEMP_CPU	CPU Temp.		MONITOR_TEMP_OPT	OPTION Temp.		MONITOR_FAN_CPU	CPU FAN		MONITOR_FAN_POWER	POWER FAN		MONITOR_FAN_OPT	OPTION FAN		Setup As Long	Get data (pass by reference)			0:Disable			1:Enable
Selector As Long	Parameters (pass by value)																																																
	MONITOR_VOLT_CPU	CPU Core voltage																																															
	MONITOR_VOLT_P33	+3.3V																																															
	MONITOR_VOLT_P50	+5.0V																																															
	MONITOR_VOLT_P12	+12V																																															
	MONITOR_VOLT_M12	-12V																																															
	MONITOR_VOLT_M50	-5.0V																																															
	MONITOR_TEMP_SYSTEM	SYSTEM Temp.																																															
	MONITOR_TEMP_CPU	CPU Temp.																																															
	MONITOR_TEMP_OPT	OPTION Temp.																																															
	MONITOR_FAN_CPU	CPU FAN																																															
	MONITOR_FAN_POWER	POWER FAN																																															
	MONITOR_FAN_OPT	OPTION FAN																																															
	Setup As Long	Get data (pass by reference)																																															
		0:Disable																																															
		1:Enable																																															
Processing	Gets the current enabled/disabled monitor status.																																																
Example	<pre>Dim ret As Long Dim Setup As Long // Get the setup status of the CPU core voltage ret = GetMonitorSetup(MONITOR_VOLT_CPU, Setup)</pre>																																																

Appendices

GetVoltParam

Call format	Declare Function GetVoltParam Lib "PL_Ioc.dll" (ByVal Selector As Long, ByRef ULimit As Long, ByRef LLimit As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Parameters (pass by value) MONITOR_VOLT_CPU CPU Core voltage MONITOR_VOLT_P33 +3.3V MONITOR_VOLT_P50 +5.0V MONITOR_VOLT_P12 +12V MONITOR_VOLT_M12 -12V MONITOR_VOLT_M50 -5.0V ULimit As Long Voltage value upper limit (unit: mV) (pass by reference) LLimit As Long Voltage value lower limit (unit: mV) (pass by reference)
Processing	Gets the voltage monitoring parameter.
Example	Dim ret As Long Dim ULimit As Long Dim LLimit As Long // Get the upper/lower limit of the CPU core voltage value ret = GetVoltParam(MONITOR_VOLT_CPU, ULimit, LLimit)



Since the data received from this function is in mV units, the following conversion is needed for use in (Volt) units:

$$\text{Data in Volt unit} = \text{Data in mV unit} / 1000$$

GetCurrentVolt

Call format	Declare Function GetCurrentVolt Lib "PL_Ioc.dll" (ByVal Selector As Long, ByVal Data As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Parameters (pass by value) MONITOR_VOLT_CPU CPU Core voltage MONITOR_VOLT_P33 +3.3V MONITOR_VOLT_P50 +5.0V MONITOR_VOLT_P12 +12V MONITOR_VOLT_M12 -12V MONITOR_VOLT_M50 -5.0V Data As Long Voltage value(unit: mV)(pass by reference)
Processing	Gets the current voltage value.
Example	Dim ret As Long Dim Data As Long // Get the CPU core voltage value. ret = GetCurrentVolt(MONITOR_VOLT_CPU, Data)



Since the data received from this function is in mV units, the following conversion is needed for use in (Volt) units:

$$\text{Data in Volt unit} = \text{Data in mV unit}/1000$$

GetFanParam

Call format	Declare Function GetFanParam Lib "PL_Ioc.dll" (ByVal Selector As Long, ByVal LLimit As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Parameters (pass by value) MONITOR_FAN_CPU CPU FAN MONITOR_FAN_POWER POWER FAN MONITOR_FAN_OPT OPTION FAN LLimit As Long CPU FAN revolution lower limit value (unit: RPM) (pass by value) (RPM: revolutions per minute)
Processing	Gets the parameter for monitoring the FAN.
Example	Dim ret As Long Dim LLimit As Long // Get the CPU FAN lower limit rpm value ret = GetFanParam(MONITOR_FAN_CPU, LLimit)

Appendices

GetCurrentFan

Call format	Declare Function GetCurrentFan Lib "PL_Ioc.dll" (ByVal Selector As Long, ByRef Data As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Parameters (pass by value) MONITOR_FAN_CPU CPU FAN MONITOR_FAN_POWER POWER FAN MONITOR_FAN_OPT OPTION FAN Data As Long CPU FAN revolution lower limit value (unit: RPM) (pass by reference) (RPM: revolutions per minute)
Processing	Gets the current FAN rpm.
Example	Dim ret As Long Dim Data As Long // Get the number of revolutions of the CPU FAN ret = GetCurrentFan(MONITOR_FAN_CPU, Data)

GetTempParam

Call format	Declare Function GetTempParam Lib "PL_Ioc.dll" (ByVal Selector As Long, ByRef ULimit As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Get parameter (pass by value) MONITOR_TEMP_SYSTEM SYSTEM temp. MONITOR_TEMP_CPU CPU temp. MONITOR_TEMP_OPT OPTION temp. ULimit As Long Temperature upper limit (unit: °C) (pass by reference)
Processing	Gets the parameter for monitoring the temperature.
Example	Dim ret As Long Dim ULimit As Long // Gets the upper limit of SYSTEM temperature ret = GetTempParam(MONITOR_TEMP_SYSTEM, ULimit)

Appendices

SetWdtMask

Call format	Declare Function SetWdtMask Lib "PL_Ioc.dll" (ByVal Selector As Long, ByVal Mask As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Setup items (pass by value) WARNING_LAMP LAMP WARNING_ALARM ALARM Mask As Long Mask data (pass by value) MASK_OFF Release mask MASK_ON Mask
Processing	Sets the warning mask to be output when a watchdog timer time-out occurs.
Example	Dim ret As Long // Mask the LAMP output ret = SetWdtMask(WARNING_LAMP, MASK_ON) // Release the mask for the ALARM output ret = SetWdtMask(WARNING_ALARM, MASK_OFF)

GetWdtMask

Call format	Declare Function GetWdtMask Lib "PL_Ioc.dll" (ByVal Selector As Long, ByRef Mask As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Setup items (pass by reference) WARNING_LAMP LAMP WARNING_ALARM ALARM Mask As Long (pass by reference) MASK_OFF Release the mask MASK_ON Mask
Processing	Gets the WDT timeout warning output mask data.
Example	Dim ret As Long Dim Mask As Long // Gets LAMP mask data ret = GetWdtMask(WARNING_LAMP, Mask) // Gets ALARM mask data ret = GetWdtMask(WARNING_ALARM, Mask)

StartWdt

Call format	Declare Function StartWdt Lib "PL_Ioc.dll" () As Long
Return value	Other than 0: Normal 0: Error
Argument	None
Processing	Stops the WDT countdown.
Example	Dim ret As Long ret = StartWdt()

StopWdt

Call format	Declare Function StopWdt Lib "PL_Ioc.dll" () As Long
Return value	Other than 0: Normal 0: Error
Argument	None
Processing	Stops the WDT countdown.
Example	Dim ret As Long ret = StopWdt()

RestartWdt

Call format	Declare Function RestartWdt Lib "PL_Ioc.dll" () As Long
Return value	Other than 0: Normal 0: Error
Argument	None
Processing	Resets the initial value of the watchdog timer to the default value, and restarts the countdown.
Example	Dim ret As Long ret = RestartWdt()



Note: RestartWdt can only be used after the StartWdt countdown has started. If RestartWdt is used after the timeout, it should be after ClearWdtTimeout has been used to clear the timeout condition and after StartWdt has started the countdown.

Appendices

RunningWdt

Call format	Declare Function RunningWdt Lib "PL_Ioc.dll" (ByRef RunFlag As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	RunFlag As Long Operating status of the watchdog timer (pass by reference) WATCHDOG_STOP Stopped WATCHDOG_COUNTDOWN Counting down
Processing	Gets the operating status of the watchdog timer.
Example	Dim ret As Long Dim RunFlag As Long ret = RunningWdt(RunFlag)

SetWarningOut

Call format	Declare Function SetWarningOut Lib "PL_Ioc.dll" (ByVal Selector As Long, ByVal WarnOut As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Setting items (pass by value) WARNING_LAMP LAMP WARNING_ALARM ALARM WarnOut As Long Output condition (pass by value) OUTPUT_OFF Output OFF OUTPUT_ON Output ON
Processing	Sets warning data for the setup items (LAMP and ALARM).
Example	Dim ret As Long // Set the output status of the LAMP to ON ret = SetWarningOut(WARNING_LAMP, OUTPUT_ON) // Set the output status of the ALARM to OFF ret = SetWarningOut(WARNING_ALARM, OUTPUT_OFF)

GetWarningOut

Call format	Declare Function GetWarningOut Lib "PL_Ioc.dll" (ByVal Selector As Long, ByVal WarnOut As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Setting items (pass by value) WARNING_LAMP LAMP WARNING_ALARM ALARM WarnOut As Long Output condition (pass by reference) OUTPUT_OFF Output OFF OUTPUT_ON Output ON
Processing	Gets the current warning status of the setup items (LAMP and ALARM).
Example	Dim ret As Long Dim WarnOut As Long // Gets the output status of the LAMP ret = GetWarningOut(WARNING_LAMP, WarnOut) // Get the output status of the ALARM ret = GetWarningOut(WARNING_ALARM, WarnOut)

GetUniversalIn

Call format	Declare Function GetUniversalIn Lib "PL_Ioc.dll" (ByVal Selector As Long, ByVal UniIn As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Designated port (pass by value) PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1 UniIn As Long Input status (pass by reference) INPUT_OFF No input INPUT_ON Input
Processing	Gets the input status of the designated port (Universal Input 0 and Universal Input 1).
Example	Dim ret As Long Dim UniIn As Long // Get the input status of the Universal Input 0 ret = GetUniversalIn(PORT_UNI0, UniIn) // Get the input status of the Universal Input 1 ret = GetUniversalIn(PORT_UNI1, UniIn)

Appendices

ClearUniversalIn

Call format	Declare Function ClearUniversalIn Lib "PL_Ioc.dll" (ByVal Selector As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Designated port (pass by value) PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1
Processing	Clears the input status of the designated port (Universal Input 0 and Universal Input 1).
Example	Dim ret As Long // Clear the input status of Universal Input 0 ret = ClearUniversalIn(PORT_UNI0) // Clear the input status of Universal Input 1 ret = ClearUniversalIn(PORT_UNI1)

SetUniversalInMask

Call format	Declare Function SetUniversalInMask Lib "PL_Ioc.dll" (ByVal Selector As Long, ByVal Mask As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Designated port (pass by value) PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1 Mask As Long Mask data (pass by value) MASK_OFF Clear mask MASK_ON Mask
Processing	Sets the masking information of the designated ports (Universal Input 0 and Universal Input 1).
Example	Dim ret As Long // Release the masking for Universal Input 0 ret = SetUniversalInMask(PORT_UNI0, MASK_OFF) // Mask Universal Input 1 ret = SetUniversalInMask(PORT_UNI1, MASK_ON)

GetUniversalInMask

Call format	Declare Function GetUniversalInMask Lib "PL_Ioc.dll" (ByVal Selector As Long, ByVal Mask As Long) As Long	
Return value	Other than 0: Normal 0: Error	
Argument	Selector As Long	Designated port (pass by value) PORT_UNI0 Universal Input 0 PORT_UNI1 Universal Input 1
	Mask As Long	Mask data (pass by reference) MASK_OFF Release mask MASK_ON Mask
Processing	Gets the masking information of the subject ports (Universal Input 0 and Universal Input 1).	
Example	Dim ret As Long Dim Mask As Long // Get the masking information for Universal Input 0 ret = GetUniversalInMask(PORT_UNI0, Mask) // Get the masking information for Universal Input 1 ret = GetUniversalInMask(PORT_UNI1, Mask)	

SetResetMask

Call format	Declare Function SetResetMask Lib "PL_Ioc.dll" (ByVal Mask As Long) As Long	
Return value	Other than 0: Normal 0: Error	
Argument	Mask As Long	Mask data (pass by value) MASK_OFF Release mask MASK_ON Mask
Processing	Sets the reset mask.	
Example	Dim ret As Long // Releases the reset mask ret = SetResetMask(MASK_OFF)	

Appendices

GetResetMask

Call format	Declare Function GetResetMask Lib "PL_Ioc.dll" (ByRef Mask As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Mask As Long Mask data (pass by reference) MASK_OFF Release mask MASK_ON Mask
Processing	Gets the current reset mask information.
Example	Dim ret As Long Dim Mask As Long ret = GetResetMask(Mask)

SetIdeErr

Call format	Declare Function SetIdeErr Lib "PL_Ioc.dll" (ByVal IdeErr As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	IdeErr As Long Error output data (pass by reference) IDE_ERROR_OFF Disables error output IDE_ERROR_ON Enables error output
Processing	Uses software control to set the IDE error output.
Example	Dim ret As Long // Set the system to disable the IDE error output ret = SetIdeErr(IDE_ERROR_OFF)

GetIdeErrHard

Call format	Declare Function GetIdeErrHard Lib "PL_Ioc.dll" (ByVal Selector As Long, ByVal IdeErr As Long) As Long		
Return value	Other than 0: Normal 0: Error		
Argument	Selector As Long	Get parameter (pass by value)	
		IDE_ERROR_1	IDE_ERR1
		IDE_ERROR_2	IDE_ERR2
	IdeErr As Long	Error signal (pass by reference)	
		IDE_ERROR_OFF	Normal
		IDE_ERROR_ON	Error
Processing	Gets the current IDE error signal output by the hardware.		
Example	<pre>Dim ret As Long Dim IdeErr As Long // Gets the IDE ERR1 signal ret = GetIdeErrHard(IDE_ERROR_1, IdeErr)</pre>		

GetLightblowErr

Call format	Declare Function GetLightblowErr Lib "PL_Ioc.dll" (ByRef LightblowErr As Long) As Long		
Return value	Other than 0: Normal 0: Error		
Argument	LightblowErr As Long	Error data (pass by reference)	
		BACKLIGHT_OK	Normal
		BACKLIGHT_ERR	Error
Processing	Gets the current backlight error information.		
Example	<pre>Dim ret As Long Dim LightblowErr As Long // Gets the backlight error information. ret = GetLightblowErr(LightblowErr)</pre>		

Appendices

GetEvent

Call format	Declare Function GetEvent Lib "PL_Ioc.dll" (ByVal Selector As Long, ByRef Event As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Selector As Long Parameters (pass by value) EVENT_VOLT_CPU CPU core voltage EVENT_VOLT_P33 +3.3V EVENT_VOLT_P50 +5.0V EVENT_VOLT_P12 +12V EVENT_VOLT_M12 -12V EVENT_VOLT_M50 -5.0V EVENT_FAN_CPU CPU FAN EVENT_FAN_POWER POWER FAN EVENT_FAN_OPT OPTION FAN EVENT_TEMP_SYSTEM SYSTEM temp. EVENT_TEMP_CPU_OPT CPU or OPTION temp. EVENT_UNI_IN0 Universal Input 0 EVENT_UNI_IN1 Universal Input 1 EVENT_WDT_TIMEOUT Watchdog Timeout Event As Long Error event data (pass by reference) ERROR_EVENT_OFF No error event ERROR_EVENT_ON Error event
Processing	Checks for the irregularities in the machine voltage, FAN, and temperature, Universal Input function (event) data, and WatchDog Timeout data.
Example	Dim ret As Long Dim Event As Long // Gets the error event data of the CPU core voltage ret = GetEvent(EVENT_VOLT_CPU, Event)

Appendices

StartInsideBuzzer

Call format	Declare Function StartInsideBuzzer Lib "PL_Ioc.dll" (ByVal hz As Long, ByVal ms As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	hz As Long Buzzer frequency (pass by value) ms As Long Buzzer sound period (pass by value)
Processing	Triggers the internal buzzer to sound at the specified frequency and for the specified period.
Example1	Dim ret As Long Dim hz As Long Dim ms As Long // Sound the buzzer for 1 second at 600 Hz hz = 600 ms = 1000 ret = StartInsideBuzzer(hz, ms)



Note: This feature cannot be used with a PL running WindowsNT® 4.0 or Windows® 2000 due to the use of Windows® 98 functions.

StopInsideBuzzer

Call format	Declare Function StopInsideBuzzer Lib "PL_Ioc.dll" () As Long
Return value	Other than 0: Normal 0: Error
Argument	None
Processing	Stops the internal buzzer.
Example	Dim ret As Long // Stops the internal Buzzer. ret = StopInsideBuzzer()



Note: This feature cannot be used with a PL running WindowsNT® 4.0 or Windows® 2000 due to the use of Windows® 98 functions.

ChkInsideBuzzer

Call format	Declare Function ChkInsideBuzzer Lib "PL_Ioc.dll" (ByRef buff As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	BuzzerParam As Long Buzzer status (pass by reference) BUZZER_ON Buzzer is ON BUZZER_OFF Buzzer is OFF
Processing	Checks for the ON/OFF status of the internal buzzer.
Example	Dim ret As Long Dim BuzzerParam As Long // Checks the buzzer status ret = ChkInsideBuzzer(BuzzerParam)



Note: This feature cannot be used with a PL running WindowsNT® 4.0 or Windows® 2000 due to the use of Windows® 98 functions.

GetWdtTimeout

Call format	Declare Function GetWdtTimeout Lib "L_Ioc.dll" (ByRef Timebuf As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Timebuf As Long WDT status (pass by reference)
Processing	Gets the watchdog timeout status.
Example	Dim ret As Long Dim Timebuf As Long // Gets the timeout status of the watchdog. ret = GetWdtTimeout(Timebuf)

Appendices

ClearWdtTimeout

Call format	Declare Function ClearWdtTimeout Lib "PL_Ioc.dll" () As Long
Return value	Other than 0: Normal 0: Error
Argument	None
Processing	Clears the timeout status of the watchdog.
Example	Dim ret As Long // Clear the timeout status of the watchdog. ret = ClearWdtTimeout()

SetWarningDOUT

Call format	Declare Function SetWarningDOUT Lib "PL_Ioc.dll" (ByVal WarningOut As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	WarningOut As Long Output status (pass by value) OUTPUT_OFF Output OFF OUTPUT_ON Output ON
Processing	Sets the warning status of the current setup item (DOUT).
Example	Dim ret As Long // Set the output status of DOUT to OFF. ret = SetWarningDOUT(OUTPUT_OFF)

GetWarningDOUT

Call format	Declare Function GetWarningDOUT Lib "PL_Ioc.dll" (ByRef WarningOut As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	WarningOut As Long Output status (pass by reference) OUTPUT_OFF Output OFF OUTPUT_ON Output ON
Processing	Gets the warning status of the current setup item (DOUT).
Example1	Dim ret As Long Dim WarningOut As Long ret = GetWarningDOUT(WarningOut)

GetSmiDrvHandle

Call format	DeclareFunctionGetSmiDrvHandle Lib "PL_Ioc.dll" () As Long
Return value	0: Normal 1: Error
Argument	None
Processing	Gets the device driver handle to exchange information with the software mirroring device driver.
Example1	Dim ret As Long ret = GetSmiDrvHandle()



An error will occur if the software mirroring device driver is not running.

CloseSmiDrvHandle

Call format	DeclareFunctionCloseSmiDrvHandle Lib "PL_Ioc.dll" () As Long
Return value	Other than 0: Normal 0: Error
Argument	None
Processing	Destroys the handle acquired with the "GetSmiDrvHandle" function.
Example	Dim ret As Long // Destroys the handle. ret = CloseSmiDrvHandle()

GetSmiAryStatus

Call format	DeclareFunctionGetSmiAryStatus Lib "PL_Ioc.dll" (ByRef Status As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Status As Long Software mirroring status (pass by reference) ARYSTAT_GOOD Normal ARYSTAT_UNCONFIG Not configured ARYSTAT_REBUILD Being rebuilt ARYSTAT_REDUCE Being reduced ARYSTAT_DEAD Mirror status destroyed
Processing	Gets the status of the software mirroring feature.
Example	Dim ret As Long Dim Status As Long // Get the status of the software mirroring feature. ret = GetSmiAryStatus(Status)

Appendices

GetSmiDevStatus

Call format	Declare Function GetSmiDevStatus Lib "PL_Ioc.dll" (ByVal Id As Long, ByRef Type As Long, ByRef Status As Long) As Long
Return value	Other than 0: Normal 0: Error
Argument	Id As Long Device ID (pass by value) 0 : Master HDD 1 : Slave HDD Type As Long Device type (pass by reference) ATADEVICE ATA type device ATAPIDEVICE CD-ROM NODEVICE No device Status As Long Device status (pass by reference) DEVSTAT_GOOD Normal DEVSTAT_NOTEXIST Not connected DEVSTAT_BROKEN Device failure
Processing	Gets the device status of the software mirroring feature.
Example	Dim ret As Long Dim Id As Long Dim Type As Long Dim Status As Long // Gets the device status of the software mirroring feature. Id = 0 ret = GetSmiDevStatus(Id, Type, Status)

Memo

A.8 Backlight Control API-DLL

A.8.1 Operation Environment

The following information explains the Dynamic Link Libraries used by the backlight control feature on a PL-5900 Series unit.

API-DLLs provide the interface for applications to access the System Monitor/RAS feature (System Monitor/RAS Device Driver). Applications can use DLLs to access the following feature.

1. Backlight Control ON/OFF

■ Compatible Operating Systems

The API-DLLs contained on the PL unit's CD-ROM are compatible with the following OS types.

- Microsoft® Windows®98 Second Edition
- Microsoft® WindowsNT®4.0 (Windows Service Pack 3 or higher)
- Microsoft® Windows®2000

Each OS must use its corresponding Backlight Control Device.

■ Compatible Languages

- Microsoft® Visual C
- Microsoft® Visual C++
- Microsoft® Visual Basic

◆ Required Files

The following files are required when using DLLs. Each language requires its own set of files.

• Visual C

File Name	Description
PL_BLIocif.h	Driver interface definition "include" file
PL_BLIoc.LIB	Library definition file
PL_BLIoc.dll	Dynamic link library file

• Visual C++

File Name	Description
PL_BLIocif.h	Driver interface definition "include" file
PL_BLIocall.h	CPL_BLIocall class definition "include" file
PL_BLIocctl.h	CPL_BLIocctl class definition "include" file
PL_BLIoc.LIB	Library definition file
PL_BLIoc.dll	Dynamic Link library file

* "#include header files should be "included" in the following order.

```
#include PL_BLIocif.h
```

```
#include PL_BLIocctl.h
```

PL_BLIocall.h is automatically included, and does not need to be directly designated.

• Visual Basic

File Name	Description
PL_BLIoc.bas	Driver interface definition file
PL_BLIoc.LIB	Library definition file
PL_BLIoc.dll	Dynamic link library file

■ Dynamic Link Library (DLL)

In order for an application to use PL_BLIoc.dll, it should be copied to the following folder.

OS	Location
Windows [®] 98	C:\Windows\System
WindowsNT [®] 4.0/Windows [®] 2000	C:\Winnt\System32

A.8.2 Class Contents

■ CPL_BLIoctl Class

This class is used to set the parameters for device driver access using CPL_BLIoctl class.

Key Word	Type	Variable Name	Description
public	HANDLE	m_Drvhandle	Device driver handle

■ CPL_BLIocal Class

This uses the parameters set in CPL_BLIoctl, and calls up DeviceIoControl (Driver Access function).

However, since this class succeeds CPL_BLIoctl, it cannot be used directly.

Key Word	Type	Variable Name	Description
public	HANDLE	m_h	Device driver handle
public	LONG	m_long	Control code for action to perform
public	void *	m_ibp	Input data buffer address
public	ULONG	m_ibsize	Input data buffer size
public	void *	m_obp	Output data buffer address
public	ULONG	m_obsize	Output data buffer size
public	DWORD	m_retsize	Address for actual no. of output bytes
public	LPOVERLAPPED	m_ovlp	Address of overlap design

A.8.3 Visual C Functions

Function Name	Description
InitBLIoctl	Creates the CPL_BLIoctl object
EndBLIoctl	Destroys the CPL_BLIoctl object
GetBLDrvHandle	Gets the driver handle
GetBLDrvVersion	Gets the driver version
SetBLControl	Sets the backlight control values
GetBLControl	Gets the backlight control settings

A.8.4 Visual C Function Specifications (Details)

InitBLIoctl

Call Format	void WINAPI InitBLIoctl(void)
Return Value	None
Arguments	None
Processing	Creates a CPL_BLIoctl object. The object once created is not destroyed until the EndBLIoctl function is called.
Example	InitBLIoctl();

EndBLIoctl

Call Format	void WINAPI EndBLIoctl(void)
Return Value	None
Arguments	None
Processing	Destroys the object created using the InitBLIoctl function.
Example	EndBLIoctl();

GetBLDrvHandle

Call Format	int WINAPI GetBLDrvHandle(HANDLE * pHndl)
Return Value	0: Normal 1: Error
Arguments	(I/O) HANDLE *pHndl Pointer to the device driver handle
Processing	Gets the device driver handle to communicate with the device driver.
Example	int ret; HANDLE hndl; ret = GetBLDrvHandle(&hndl);



Note: An error will occur if the Backlight Control Device Driver is not running.

Appendices

GetBLDrvVersion

Call Format BOOL WINAPI GetBLDrvVersion
 (int *pMajor, int *pMinor)

Return Value TRUE: Normal
 FALSE: Error

Arguments (I/O) int *pMajor Pointer to version information (Major, 0 to 99).
 (I/O)int *pMinor Pointer to version information (Minor, 0 to 99).

Processing Gets the driver's version information.

Example BOOL ret;
 int Major, Minor;
 ret = GetBLDrvVersion(&Major, &Minor);



If the version is 1.10, then you will get

Major: 1 (decimal)
Minor: 10 (decimal).

SetBLControl

Call Format BOOL WINAPI SetBLControl (int BLFlag)

Return Value TRUE: Normal
 FALSE: Error

Arguments (I) int BLF flag Setting Parameters
 BACKLIGHT_OFF Backlight OFF
 BACKLIGHT_ON Backlight ON

Processing Sets the backlight ON/OFF.

Example BOOL ret;
 // Turns the backlight control ON.
 ret = SetBLControl(BACKLIGHT_ON);

GetBLControl

Call Format BOOL WINAPI GetBLControl (int pBLFlag)

Return Value TRUE: Normal
 FALSE: Error

Arguments (I/O) int *pBLFlag Pointer to backlight condition
 BACKLIGHT_OFF Backlight OFF
 BACKLIGHT_ON Backlight ON

Processing Gets the backlight control (settings) condition.

Example BOOL ret;
 int BLFlag;
 // Gets the backlight control (settings) condition.
 ret = GetBLControl(&BLFlag);

A.8.5 Visual C++ Functions

Function Name	Description
GetBLDrvHandle	Gets the driver handle
GetBLDrvVersion	Gets the driver version
SetBLControl	Sets the backlight control values
GetBLControl	Gets the backlight control settings

A.8.6 Visual C++ Function Specifications (Details)

GetBLDrvHandle

Call Format `int GetBLDrvHandle(void)`

Return Value 0: Normal
 1: Error

Arguments None

Processing Gets the device driver handle to communicate with the device driver. The handle obtained is stored in the member variable `m_handle`.

Example 1 `CPL_BLIocctl m_BLIoc;`
 `m_BLIoc.GetBLDrvHandle();`

Example 2 `int ret;`
 `HANDLE hndl;`
 `ret = ::GetBLDrvHandle(&hndl);`



An error will occur if the Backlight Control Device Driver is not running.

Appendices

GetBLDrvVersion

Call Format	BOOL GetBLDrvVersion(int *pMajor, int *pMinor)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I/O) int *pMajor Pointer to version information (Major, 0 to 99). (I/O)int *pMinor Pointer to version information (Minor, 0 to 99).
Processing	Gets the driver's version information.
Example 1	<pre>CPL_BLIocctl m_BLIoc; BOOL ret; int Major, Minor; ret = m_BLIoc.GetBLDrvVersion(&Major, &Minor);</pre>
Example 2	<pre>BOOL ret; int Major, Minor; ret = ::GetBLDrvVersion(&Major, &Minor);</pre>



If the version is 1.10, then you will get

Major: 1 (decimal)

Minor: 10 (decimal).

SetBLControl

Call Format	BOOL SetBLControl (int BLFlag)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I) int BLFlag Setting Parameters BACKLIGHT_OFF Backlight OFF BACKLIGHT_ON Backlight ON
Processing	Sets the backlight ON/OFF.
Example 1	<pre>CPL_BLIoc m_BLIoc; BOOL ret; // Turns the backlight control ON. ret = m_BLIoc.SetBLControl(BACKLIGHT_ON)</pre>
Example 2	<pre>BOOL ret; // Turns the backlight control ON. ret = ::SetBLControl(BACKLIGHT_ON);</pre>

GetBLControl

Call Format	BOOL GetBLControl (int *pBLFlag)
Return Value	TRUE: Normal FALSE: Error
Arguments	(I/O) int *pBLFlag Pointer to backlight condition BACKLIGHT_OFF Backlight OFF BACKLIGHT_ON Backlight ON
Processing	Gets the backlight control (settings) status.
Example 1	<pre>CPL_BLIoc m_BLIoc; BOOL ret; int BLFlag; // Gets the backlight control condition. ret = m_BLIoc.GetBLControl(&BLFlag);</pre>
Example 2	<pre>BOOL ret; int BLFlag; // Turns the backlight control ON. ret = ::GetBLControl(&BLFlag);</pre>

A.8.7 Visual Basic Functions

Function Name	Description
InitBLIoctl	Creates the CPL_ioctl object
EndBLIoctl	Destroys the CPL_ioctl object
GetBLDrvHandle	Gets the driver handle
GetBLDrvVersion	Gets the driver version
SetBLControl	Sets the backlight control values
GetBLControl	Gets the backlight control settings

A.8.8 Visual Basic Function Specifications (Details)

InitBLIoctl

Call Format	Declare Sub InitBLIoctl Lib "PL_BLIoc.dll" ()
Return Value	None
Arguments	None
Processing	Creates a CPL_BLIoctl object. The object once created is not destroyed until the EndBLIoctl function is called.
Example	Call InitBLIoctl

EndBLIoctl

Call Format	Declare Sub EndBLIoctl Lib "PL_BLIoc.dll" ()
Return Value	None
Arguments	None
Processing	Destroys the object created using the InitBLIoctl function.
Example	Call EndBLIoctl

GetBLDrvHandle

Call Format	Declare Function GetBLDrvHandle Lib "PL_BLIoc.dll" (ByRef hndl As Long) As Long
Return Value	0: Normal 1: Error
Arguments	hndl As Long Pointer to device driver handle (pass by reference)
Processing	Gets the device driver handle to communicate with the device driver.
Example	Dim ret As Long Dim hndl As Long ret = GetBLDrvHandle(hndl)



Note: An error will occur if the Backlight Control Device Driver is not running.

GetBLDrvVersion

Call Format	Declare Function GetBLDrvVersion Lib "PL_BLIoc.dll" (ByRef Major As Long, ByRef Minor As Long) As Long
Return Value	Other than 0: Normal 0: Error
Arguments	Major As Long Pointer to version information (Major, 0 to 99) (pass by reference) Minor As Long Pointer to version information (Minor, 0 to 99) (pass by reference)
Processing	Gets the driver's version information.
Example	Dim ret As Long Dim Major As Long Dim Minor As Long ret = GetBLDrvVersion(Major, Minor)



If the version is 1.10, then you will get

Major: 1 (decimal)

Minor: 10 (decimal).

SetBLControl

Call Format	Declare Function SetBLControl Lib "PL_BLIoc.dll" (ByVal BLFlag As Long) As Long
Return Value	Other than 0: Normal 0: Error
Arguments	BLFlag As Long Setting Parameters (pass by value) BACKLIGHT_OFF Backlight OFF BACKLIGHT_ON Backlight ON
Processing	Sets the backlight ON/OFF.
Example	Dim ret As Long; // Turns the backlight control ON. ret = SetBLControl(BACKLIGHT_ON)