# MB837-1U

## Intel® Atom™ D2550
## 5.25-inch SBC

# USER'S MANUAL

**Version 1.2**

## Acknowledgments

Award is a registered trademark of Award Software International, Inc.
PS/2 is a trademark of International Business Machines Corporation.
**Intel**® and **Atom**™ are registered trademark of Award Software International, Inc. registered trademark and trademark of Intel Corporation.
Microsoft Windows is a registered trademark of Microsoft Corporation.
Winbond is a registered trademark of Winbond Electronics Corporation.
All other product names or trademarks are properties of their respective owners.

# Table of Contents

# Introduction

## Product Description

The MB837-1U is a small footprint single board computer that is configured with the Intel Atom processor D2550 and Intel® NM10 Chipset.

This 5.25-inch SBC provides greater flexibility for developers of embedded computing solutions. It is ideally suited for rugged and compact designs as in Internet devices and applications in automation, industrial control, data acquisition, thin client and other embedded PC applications.

MB837-1U features the Intel's Graphics Media Accelerator, making it compatible with Windows Vista Premium. One DDR3 SO-DIMM on board implements up to 4GB of system memory. Four Gigabit LAN ports are also available.

Advanced connectivity and expansion interfaces are provided by one CF socket, one SATA-II, one Mini PCI, one Mini PCI-e and one USB 2.0.

**MB837-1U Features**

- Supports 4 Intel® Gigabit LAN ports
- Supports D2550 at 1.86GHz processor
- DDR3 SO-DIMM x1, up to 4GB
- Mini-PCI slot x1
- Mini PCI-E slot x1 (USB signal only)
- Compact Flash socket
- LAN Bypass Enable/Disable pre-setting by BIOS (GPIO Control Mode) on Eth 1 & 2

## Checklist

Your MB837-1U package should include the items listed below.

- The MB837-1U embedded board
- Driver DVD containing chipset drivers and flash memory utility
- Cables are optional.

# MB837-1U Specifications

| | |
|---|---|
| Product Name | MB837-1U |
| Form Factor | 5.25" Disk Size SBC |
| CPU Type<br>Operating Frequency | Intel "Cedar view" Processor, 32nm Bulk<br>Atom D2550 = 1.86 GHz [TDP= 10W], Cores = Dual Core |
| Chipset | Intel "Tiger Point" PCH, CG82NM10 [TDP = 2.1W, 130 nm] |
| BIOS | AMI BIOS w/ACPI |
| Ethernet controller | Intel 82583V PCI Express Gigabit ethernet controller x4 |
| Memory | CPU on-die memory controller supporting up to 4GB<br>One DDR3-1066/1333 SO-DIMM socket, Non-ECC, unbuffered |
| LAN | • Console: RS-232 @ RJ45<br>• Eth1, 2, 3 & 4: Intel 82583V @ RJ45 with LED |
| Network Bypass | One segment hardware Bypass (Eth1 & 2)<br>Control by GPIO / Watchdog |
| Watchdog Timer | Yes (256 segments, 0, 1, 2…255 sec/min) |
| Storage | • Onboard CF Socket x1<br>• 22-pin SATA Right Angle Connector Onboard for 2.5" SSD x1 |
| Rear Panel | • Factory Mode Restore Reset Switch (GPIO control)<br>• RJ45 x1 for Console<br>• RJ45 with LED x4 for GLAN |
| USB 2.0 | • [2x4] Pin header Onboard x1<br>• Mini PCI-e Socket x1 (USB Signal Only) |
| Video | VGA pin header onboard x1 |
| LPC I/O | Nuvoton NCT6627UD:<br>• RS-232 [2x5] Pin Header Onboard x1<br>• RJ45 Console x1<br>• KB/Mouse [1x6] Pin header<br>• Hardware monitors<br>• Fan Connector x1 |
| Internal I/O Headers | • 4-pin Smart Fan Connector x1<br>• 2-pin header for DC-in (12V) x1<br>• Keyboard + Mouse ([1x6] Pin Header) x1 |
| Expansion Interface | • Mini PCI Socket x1<br>• Mini PCI-e Socket x1 (USB Signal Only) |
| Power Jack | 2-pin DC +12V |
| Dimensions | 203(W) x 149.5(D) mm |
| Operation Temperature | 0 ~ 60 ˚C (32 ~ 140 ˚F) |
| Storage Temperature | -20 ~ 80 ˚C (-4 ~ 176 ˚F) |

# Installations

This section provides information on how to use the jumpers and connectors on theMB837-1U in order to set up a workable system. The topics covered are:

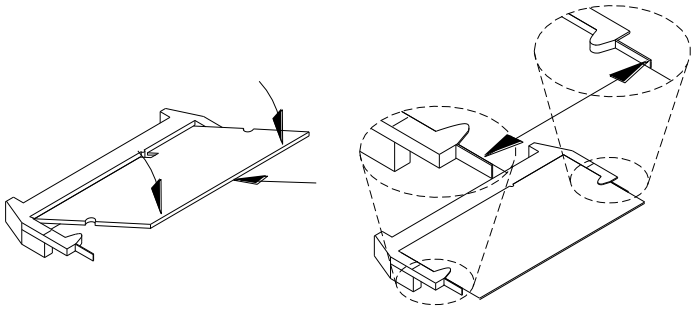
# Installing the Memory

The MB837-1U board supports one DDR3 memory socket that can support up to 4GB memory, DDR3 1066/1333 (w/o ECC function).

**Installing and Removing Memory Modules**

To install the DDR3 module, locate the memory slot on the board and perform the following steps:

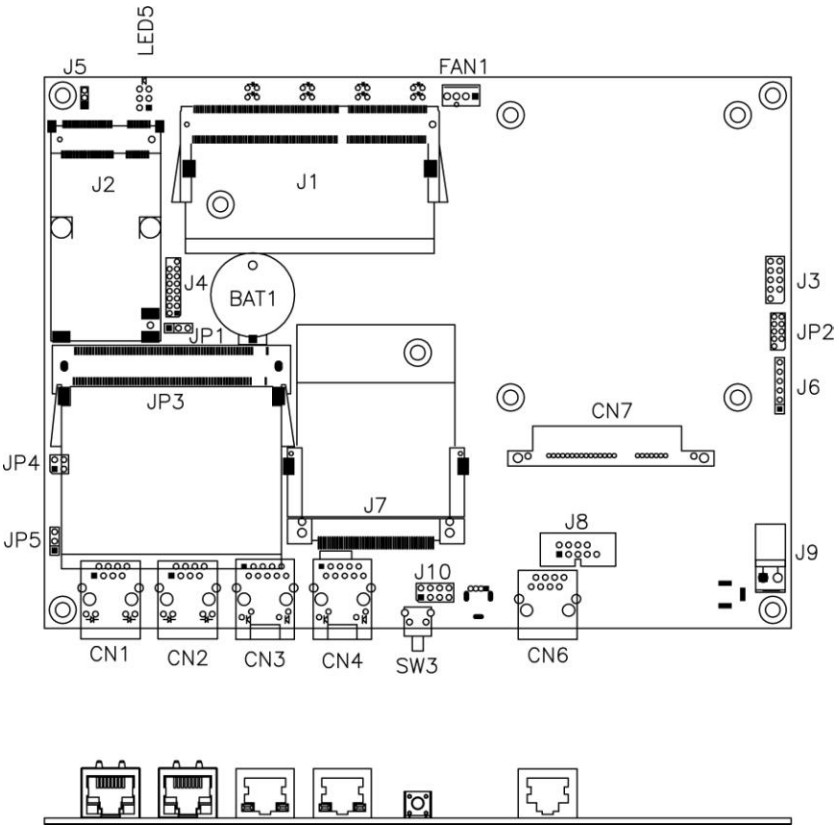
1. Hold the DDR3 module so that the key of the DDR3 module aligns with that on the memory slot. Insert the module into the socket at a slight angle (approximately 30 degrees). Note that the socket and module are both keyed, which means that the module can be installed only in one direction.
2. To seat the memory module into the socket, apply firm and even pressure to each end of the module until you feel it slip down into the socket.
3. With the module properly seated in the socket, rotate the module downward. Continue pressing downward until the clips at each end lock into position.
4. To remove the DDR3 module, press the clips with both hands.



# Setting the Jumpers

Jumpers are used on MB837-1U to select various settings and features according to your needs and applications. Contact your supplier if you have doubts about the best configuration for your needs. The following lists the connectors on MB837-1U and their respective functions.

**Jumper & Connector Location on MB837-1U**



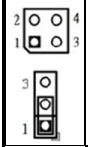
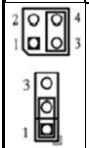
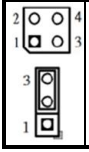
### JP1: Clear CMOS Contents

Use JP1 to clear the CMOS contents.

*Note that the power connector or jack should be disconnected from the board before clearing CMOS.*

| JP1 | Setting | Function |
|---|---|---|
|   | Pin 1-2 Short/Closed | Normal |
|   | Pin 2-3 Short/Closed | Clear CMOS |

### JP4, JP5: LAN Bypass & WDT Reboot Setting

| JP4 JP5 | Setting | Function | Power OFF | | Power ON | | Power ON OS run software | |
|---|---|---|---|---|---|---|---|---|
| | | | Normal | Bypass | Normal | Bypass | Normal | Bypass |
|  | JP4<br>1-2 & 3-4 Open<br>JP5<br>1-2 Closed | LAN bypass upon the time out of WDT. | ✓ | | ✓ | | | ✓ |
|  | JP4<br>3-4 Closed<br>1-2 Open<br>JP5<br>1-2 Closed | LAN bypass & system reboot upon the time out of WDT. | ✓ | | ✓ | | LAN Always Normal | |
| | | | | | | | WDT Reboot System | |
|  | JP4<br>1-2 & 3-4 Open<br>JP5<br>2-3 Closed | LAN bypass controlled by Super IO GP54 or setting in BIOS. | BIOS Setting **<br>GP54 Active:<br>Low: Bypass<br>High: Normal | | | | | |

Default Setting

*** Note that the Bypass setting in BIOS is only working when JP4 & JP5 are set as this configuration.*

# Setting the Connectors

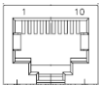
### FAN1: System Fan Power Connector

FAN1 is 4-pin header for System fan power. The fan must be a 12V fan.


| Pin # | Signal Name |
|---|---|
| 1 | Ground |
| 2 | +12V |
| 3 | Rotation detection |
| 4 | Control |

### CN1, CN2, CN3, CN4: 10 / 100 / 1000 LAN Ports

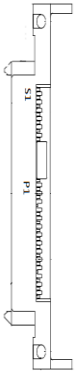
### CN6: COM1 RJ45 Connector




| Pin # | Signal Name (RS-232) |
|---|---|
| 1 | RTS, Request to send |
| 2 | DTR, Data terminal ready |
| 3 | TXD, Transmit data |
| 4 | Ground |
| 5 | Ground |
| 6 | RXD, Receive data |
| 7 | DSR, Data set ready |
| 8 | CTS, Clear to send |

### CN7: SATA HDD Dock

The SATA HDD dock combines a SATA power connector and a SATA interface connector.



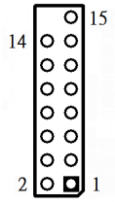

| Signal Name | Pin # | Pin # | Signal Name |
|---|---|---|---|
| GND | S1 | P1 | +3.3V |
| A+ | S2 | P2 | +3.3V |
| A- | S3 | P3 | +3.3V |
| GND | S4 | P4 | GND |
| B+ | S5 | P5 | GND |
| B- | S6 | P6 | GND |
| GND | S7 | P7 | +5V |
| | | P8 | +5V |
| | | P9 | +5V |
| | | P10 | GND |
| | | P11 | GND |
| | | P12 | GND |
| | | P13 | +12V |
| | | P14 | +12V |
| | | P15 | +12V |

### J1: SO-DIMM DDR3 Socket

### J2: Mini PCI-e Connector (USB signal only)

### J3: SPI Debug Port (Factory use only)

### J4: VGA Header



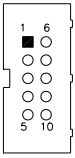

| Signal Name | Pin # | Pin # | Signal Name |
|---|---|---|---|
| DACR | 1 | 2 | +5VCRT |
| DACG | 3 | 4 | GND |
| DACB | 5 | 6 | NC |
| NC | 7 | 8 | CRT_SPD |
| GND | 9 | 10 | HSYNC_C |
| +5VCRT | 11 | 12 | VSYNC_C |
| GND | 13 | 14 | CRT_SPCLK |
| GND | 15 | | |

### J6:PS2 KB/MS Header


| Pin # | Signal Name |
|---|---|
| 1 | KBDATA |
| 2 | KBCLK |
| 3 | MSDATA |
| 4 | MSCLK |
| 5 | GND |
| 6 | +5V |

### J7: Slim Type II Compact Flash Connector

### J8: COM2 Serial Port




| Pin # | Signal Name (RS-232) |
|---|---|
| 1 | DCD, Data carrier detect |
| 2 | RXD, Receive data |
| 3 | TXD, Transmit data |
| 4 | DTR, Data terminal ready |
| 5 | Ground |
| 6 | DSR, Data set ready |
| 7 | RTS, Request to send |
| 8 | CTS, Clear to send |
| 9 | RI, Ring indicator |
| 10 | No Connect. |

### J9: AT_12V Connector

J9 is a DC-in internal connector supporting +12V.


| Pin # | Signal Name |
|---|---|
| 1 | +12V |
| 2 | Ground |

*Note: Do not connect J9 and J11 at the same time.*

### J10: USB Header


| Signal Name | Pin # | Pin # | Signal Name |
|---|---|---|---|
| VCC | 1 | 2 | Ground |
| USB1- | 3 | 4 | USB2+ |
| USB1+ | 5 | 6 | USB2- |
| Ground | 7 | 8 | VCC |

### SW3: Software Reset Button


| Signal Name | Pin # | Pin # | Signal Name |
|---|---|---|---|
| GND | 1 | 2 | Intel NM10 PCH GPIO7 |

IO Base:
Read IO 0x500 and set bit 7 to "1" (Enabled GPIO function)
Read IO 0x504 and set bit 7 to "1" (GPIO act as GPI )
Read IO 0x50C and check the bit 7 (Control Pin)

*Note: SW3 is controlled by GPIO only.*

### JP3: Mini-PCI Connector

# BIOS Setup

This chapter describes the different settings available in the AMI BIOS that comes with the board. The topics covered in this chapter are as follows:

## BIOS Introduction

The BIOS (Basic Input/Output System) installed in your computer system's ROM supports Intel processors. The BIOS provides critical low-level support for a standard device such as disk drives, serial ports and parallel ports. It also password protection as well as special support for detailed fine-tuning of the chipset controlling the entire system.

## BIOS Setup

The BIOS provides a Setup utility program for specifying the system configurations and settings. The BIOS ROM of the system stores the Setup utility. When you turn on the computer, the BIOS is immediately activated. Pressing the <Del> key immediately allows you to enter the Setup utility. If you are a little bit late pressing the <Del> key, POST (Power On Self Test) will continue with its test routines, thus preventing you from invoking the Setup. If you still wish to enter Setup, restart the system by pressing the "Reset" button or simultaneously pressing the <Ctrl>, <Alt> and <Delete> keys. You can also restart by turning the system Off and back On again. The following message will appear on the screen:

```
Press  <DEL>  to  Enter  Setup
```

In general, you press the arrow keys to highlight items, <Enter> to select, the <PgUp> and <PgDn> keys to change entries, <F1> for help and <Esc> to quit.

When you enter the Setup utility, the Main Menu screen will appear on the screen. The Main Menu allows you to select from various setup functions and exit choices.

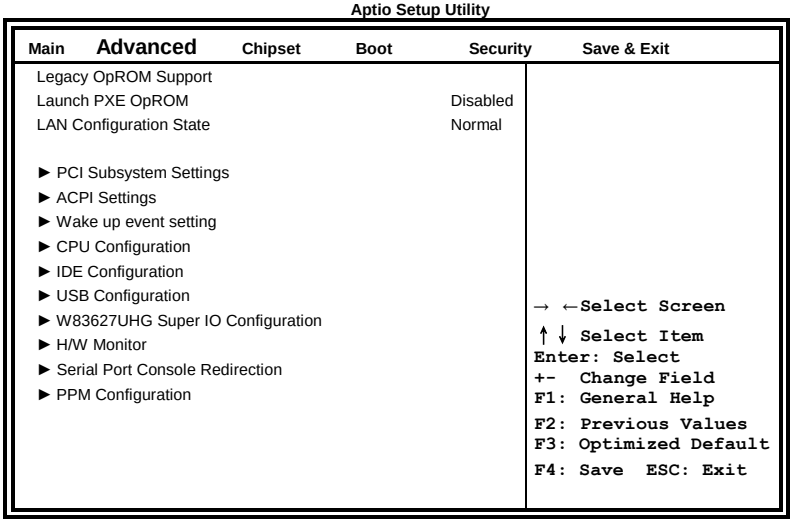
***Warning:*** *It is strongly recommended that you avoid making any changes to the chipset defaults. These defaults have been carefully chosen by both AMI and your system manufacturer to provide the absolute maximum performance and reliability. Changing the defaults could cause the system to become unstable and crash in some cases.*

### System Date
Set the Date. Use Tab to switch between Data elements.

### System Time
Set the Time. Use Tab to switch between Data elements.

## Advanced Settings

This section allows you to configure and improve your system and allows you to set up some system features according to your preference.




### Launch PXE OpROM

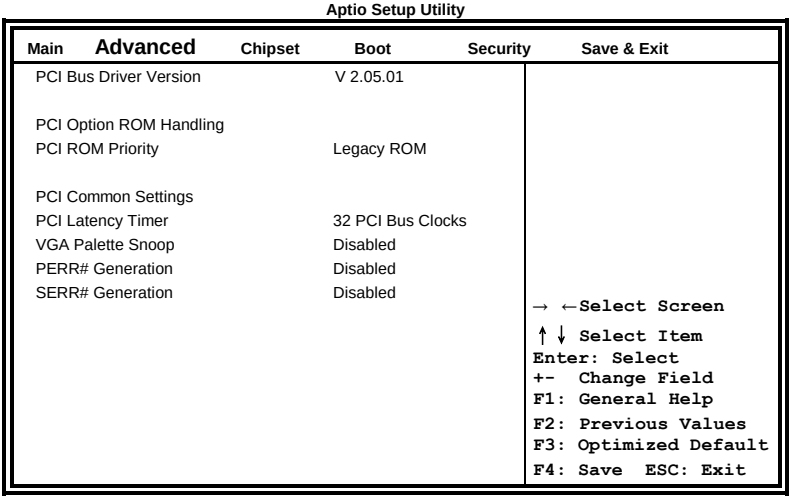
Enable or Disable Boot Option for Legacy Network Devices.

### LAN Configuration State

Enable (Bypass) or Disable (Normal) LAN Bypass. This function is only working on GPIO control mode.

Refer to page 9 to set JP5 pin 2-3 closed.

## PCI Subsystem Settings




### PCI ROM Priority

In case of multiple Option ROMs (Legacy and EFI Compatible), specifies what PCI Option ROM to launch.

### PCI Latency Timer

Value to be programmed into PCI Latency Timer Register.

### VGA Palette Snoop

Enables or Disables VGA Palette Registers Snooping.

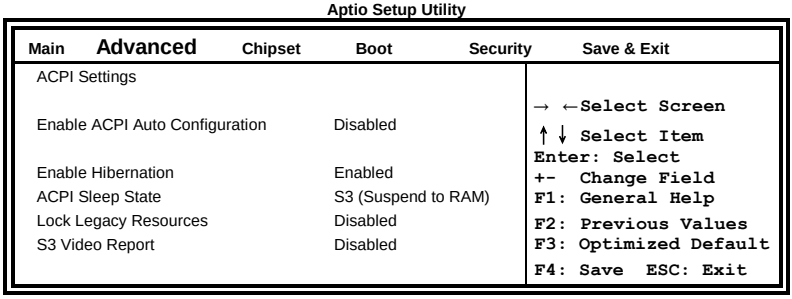
### PERR# Generation

Enables or Disables PCI Device to Generate PERR#.

### SERR# Generation

Enables or Disables PCI Device to Generate SERR#.

## ACPI Settings




### Enabled ACPI Auto Configuration

Enables or Disables BIOS ACPI Auto Configuration.

### Enable Hibernation

Enables or Disables System ability to Hibernate (OS/S4 Sleep State). This option may be not effective with some OS.

### ACPI Sleep State

Select the highest ACPI sleep state the system will enter, when the SUSPEND button is pressed.

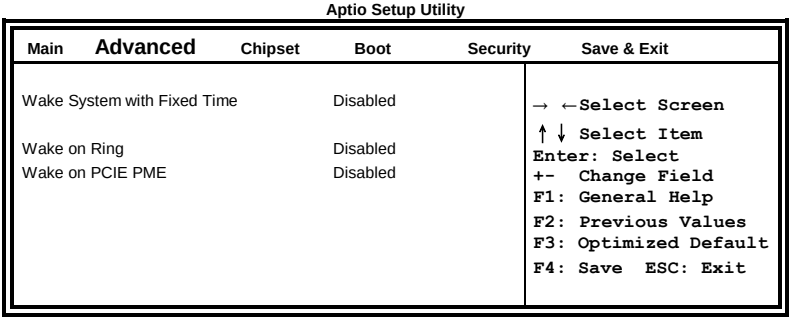
### Lock Legacy Resources

Enables or Disables Lock of Legacy Resources.

### S3 Video Report

Enable or Disable S3 Video Repost.

## Wake up event settings




### Wake System with Fixed Time

Enabled or Disabled System wake on alarm event. When enabled ,System will wake on the hr : min : sec specified.

### Wake on Ring

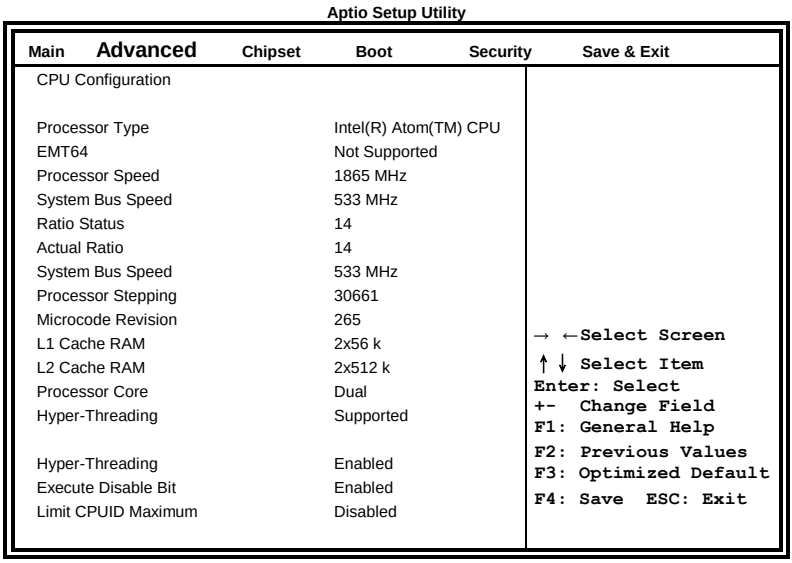
The options are Disabled and Enabled.

### Wake on PCIE PME

The options are Disabled and Enabled.

## CPU Configuration

This section shows the CPU configuration parameters.




## Hyper-threading

Enabled for Windows XP and Linux (OS optimized for Hyper-Threading Technology) and Disabled for other OS (OS not optimized for Hyper-Threading Technology). When Disabled, only one thread per enabled core is enabled.

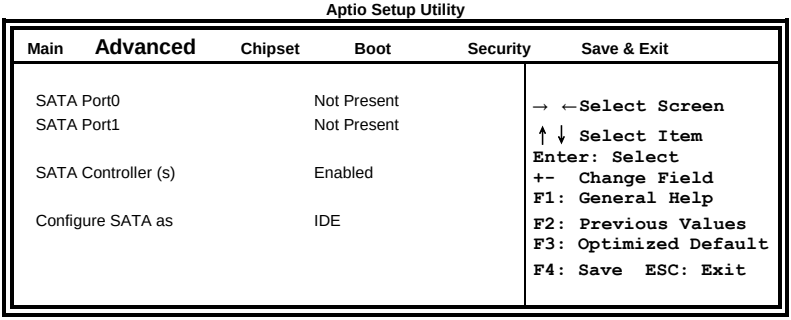
## Execute Disable Bit

XD can prevent certain classes of malicious buffer overflow attacks when combined with a supporting OS (Windows Server 2003 SP1, Windows XP SP2, SuSE Linux 9.2, Re33dHat Enterprise 3 Update 3.)

## Limit CPUID Maximum

Disabled for Windows XP.

## IDE Configuration



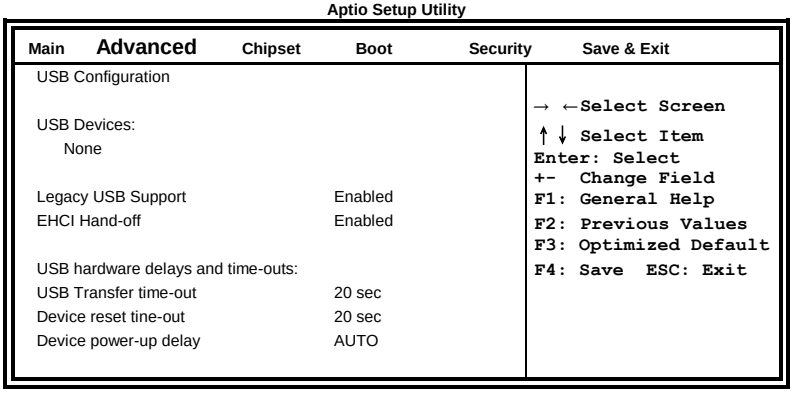

### SATA Controller(s)

Enable / Disable Serial ATA Controller.

### Configure SATA as

(1) IDE Mode.
(2) AHCI Mode.

## USB Configuration




### Legacy USB Support

Enables Legacy USB support.
AUTO option disables legacy support if no USB devices are connected.
DISABLE option will keep USB devices available only for EFI applications.

### EHCI Hand-off

Enabled/Disabled. This is a workaround for OSes without EHCI hand-off support. The EHCI ownership change should be claimed by EHCI driver.

### USB Transfer time-out

The time-out value for Control, Bulk, and Interrupt transfers.

### Device reset tine-out

USB mass Storage device start Unit command time-out.

### Device power-up delay

Maximum time the device will take before it properly reports itself to the Host Controller. 'Auto' uses default value: for a Root port it is 100ms, for a Hub port the delay is taken from Hub descriptor.

## W83627UHG Super IO Configuration

Aptio Setup Utility

| Main | Advanced | Chipset | Boot | Security | Save & Exit |
|---|---|---|---|---|---|
| W83627UHG Super IO Configuration | | | | | |
| Super IO Chip | | | Winbond W83627UHG | | |
| ► W83627UHG Serial Port 0 Configuration | | | | | |
| ► W83627UHG Serial Port 1 Configuration | | | | | |

→ ←Select Screen
↑ ↓ Select Item
Enter: Select
+- Change Field
F1: General Help
F2: Previous Values
F3: Optimized Default
F4: Save ESC: Exit

## Serial Port Configuration

Set Parameters of Serial Ports. User can Enable/Disable the serial port and Select an optimal settings for the Super IO Device.

## H/W Monitor

Aptio Setup Utility

| Main | Advanced | Chipset | Boot | Security | Save & Exit |
|---|---|---|---|---|---|
| System temperature | | | +33 C | | |
| CPU temperature | | | +37 C | | |
| System Fan Speed | | | N/A | | |
| VCORE | | | +1.184 V | | |
| +12V | | | +11.904 V | | |
| +3.3V | | | +3.312 V | | |
| +1.05 | | | +1.040 V | | |
| +5V | | | +4.896 V | | |
| VBAT | | | +3.168 V | | |
| CPU Shutdown Temperature | | | Disabled | | |
| SmartFan Control | | | Disabled | | |

→ ←Select Screen
↑ ↓ Select Item
Enter: Select
+- Change Field
F1: General Help
F2: Previous Values
F3: Optimized Default
F4: Save ESC: Exit

## CPU Smart Fan Control

Disabled (default)
70 ℃
75 ℃
80 ℃
85 ℃
90 ℃
95 ℃

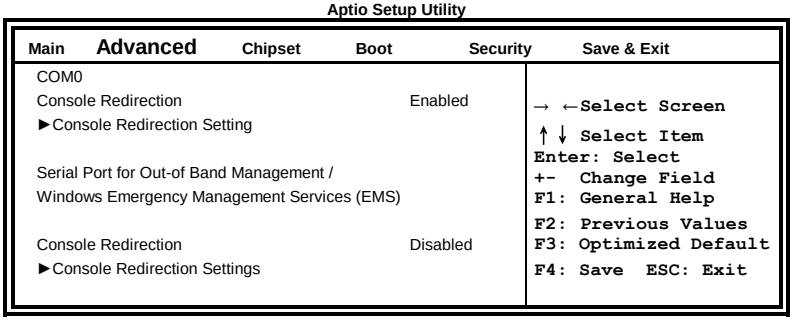
### Temperatures/Voltages

These fields are the parameters of the hardware monitoring function feature of the motherboard. The values are read-only values as monitored by the system and show the PC health status.

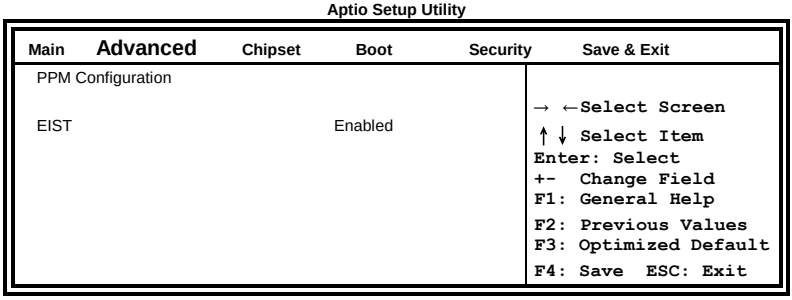
### CPU Shutdown Temperature

The default setting is disabled.

## Serial Port Console Redirection

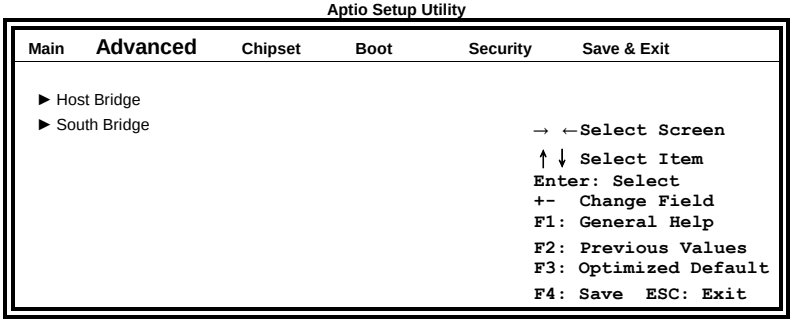



## PPM Configuration



## Chipset Settings

This section allows you to configure and improve your system and allows you to set up some system features according to your preference.




### Host Bridge

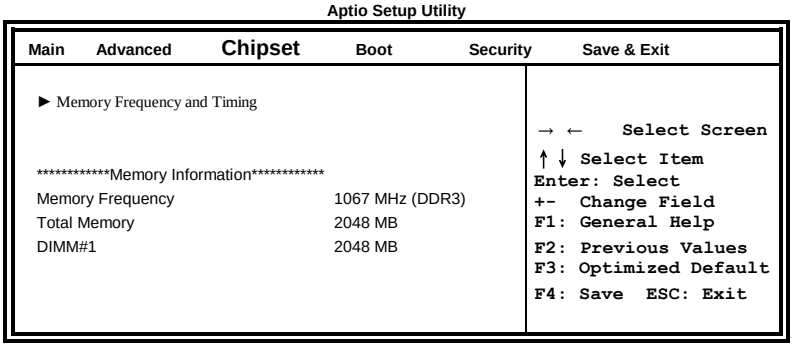
This item shows the Host Bridge Parameters.

### South Bridge

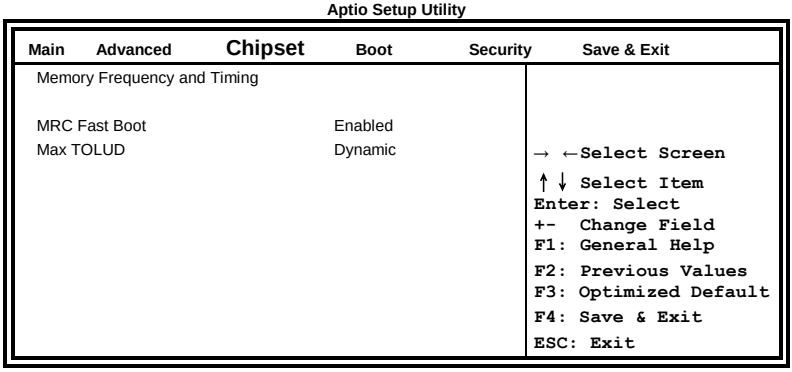
This item shows the South Bridge Parameters.

## Host Bridge

This section allows you to configure the Host Bridge Chipset.




## Memory Frequency and Timing




### MRC Fast Boot

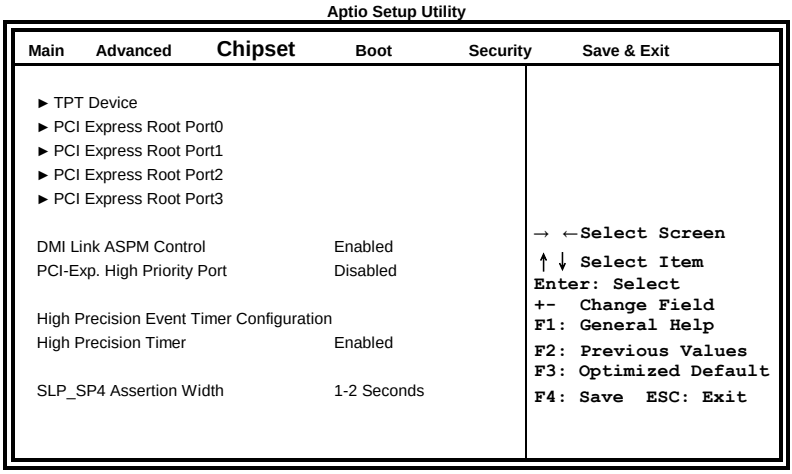
The options are Disabled and Enabled.

### Max TOLUD

Maximum Value of TOLUD. Dynamic assignment would adjust TOLUD automatically based on largest MMIO length of installed graphic controller.

## South Bridge

This section allows you to configure the South Bridge Chipset.




### DMI Clink ASPM Control

The control of Active State Power Management on both NB side and SB side of the DMI Link.

### PCI-Exp. High Priority Port

The options are Disabled, Port1, Port2, Port3, and Port4.

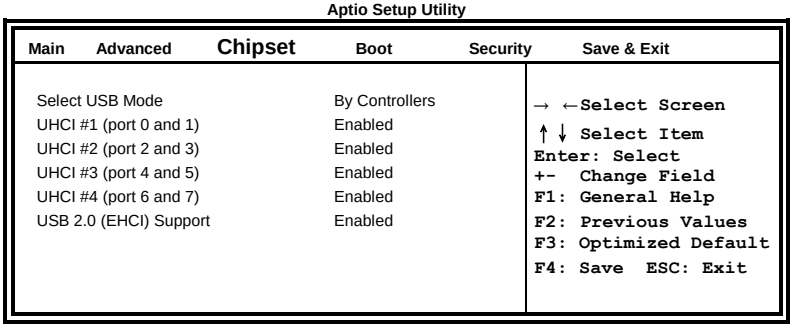
### High Precision Event Timer Configuration

Enable/or Disable the High Precision Event Timer.

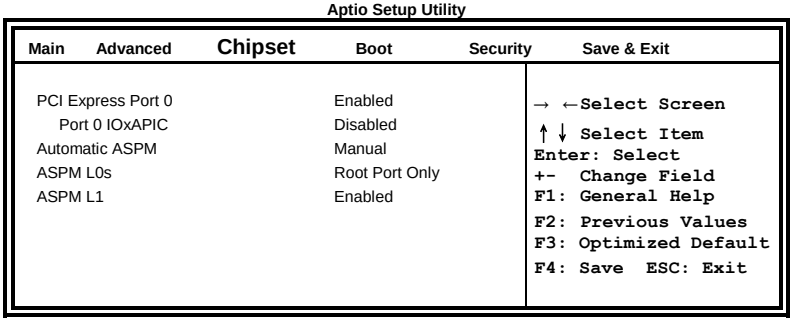
### SLP_S4 Assertion Stretch Enable

Select a minimum assertion width of the SLP_S4# signal.

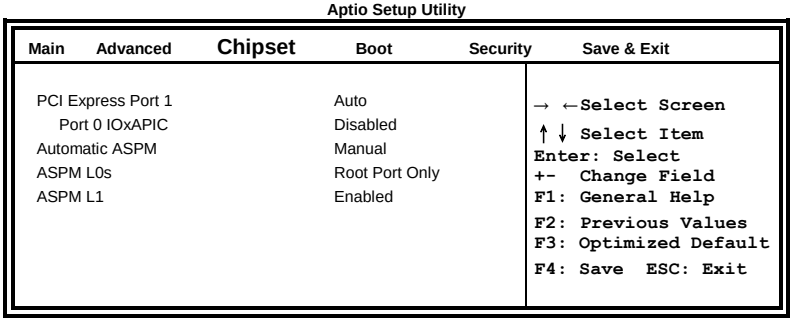
## TPT Device



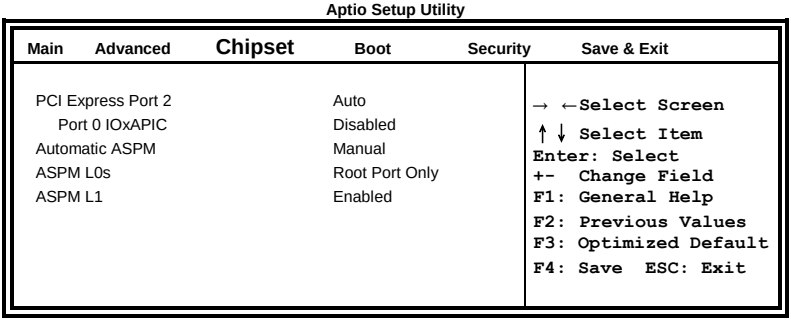

## PCI Express Root Port0



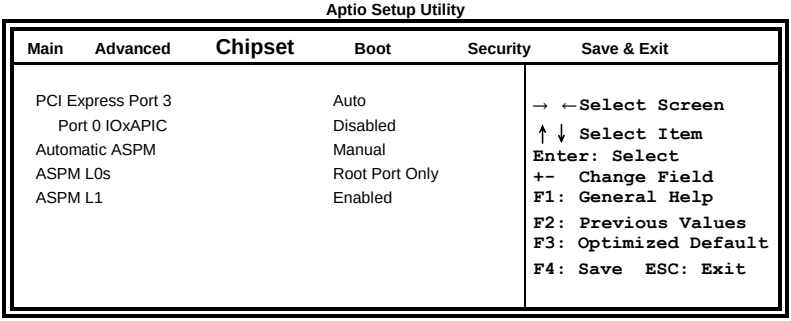

## PCI Express Root Port1



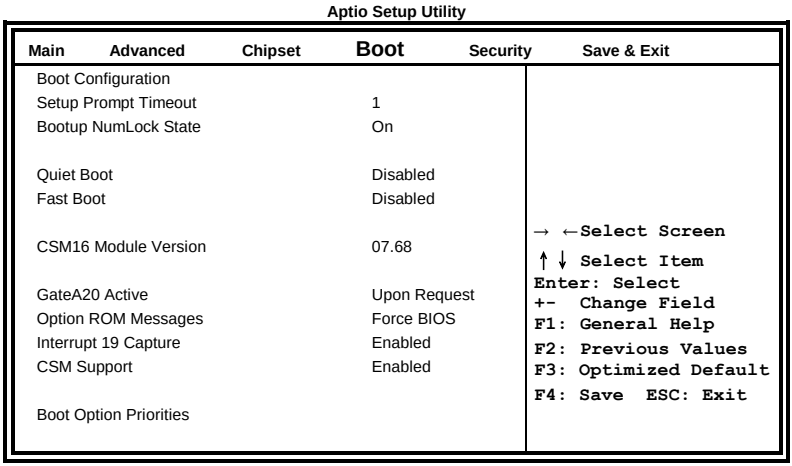
## PCI Express Root Port2




## PCI Express Root Port3



## Boot Settings




### Setup Prompt Timeout
Number of seconds to wait for setup activation key.
65535(0xFFFF) means indefinite waiting.

### Bootup NumLock State
Select the keyboard NumLock state.

### Quiet Boot
Enables/Disables Quiet Boot option.

### Fast Boot
Enables/Disables boot with initialization of a minimal set of devices to launch active boot option. Has no effect for BBS boot options.

### GateA20 Active
UPON REQUEST – GA20 can be disabled using BIOS services.
ALWAYS – do not allow disabling GA20; this option is useful when any RT code is executed above 1MB.

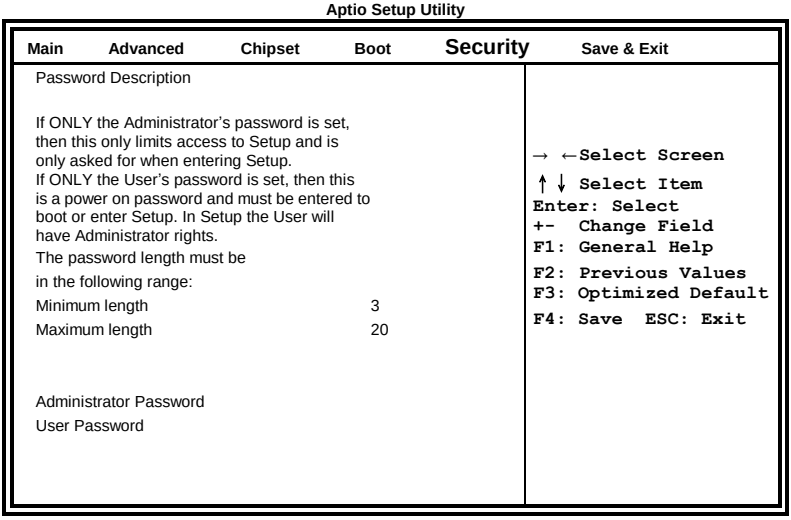
### Option ROM Messages
Set display mode. Options: Force BIOS and Keep Current.

### Interrupt 19 Capture
Enable: Allows Option ROMs to trap Int 19.

### CSM Support
Enables/Disables/Auto CSM Support.

### Boot Option Priorities
Sets the system boot order.

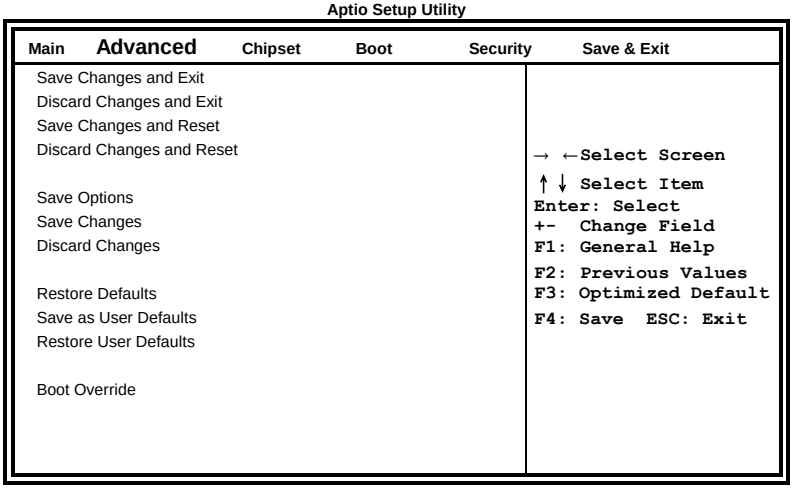
## Security Settings
This section allows you to configure and improve your system and allows you to set up some system features according to your preference.




### Administrator Password
Set Setup Administrator Password.

### User Password
Set User Password.

## Save & Exit Settings




### Save Changes and Exit
Exit system setup after saving the changes.

### Discard Changes and Exit
Exit system setup without saving any changes.

### Save Changes and Reset
Reset the system after saving the changes.

### Discard Changes and Reset
Reset system setup without saving any changes.

### Save Changes
Save Changes done so far to any of the setup options.

### Discard Changes
Discard Changes done so far to any of the setup options.

### Restore Defaults
Restore/Load Defaults values for all the setup options.

### Save as User Defaults
Save the changes done so far as User Defaults.

### Restore User Defaults
Restore the User Defaults to all the setup options.

### Boot Override
Pressing ENTER causes the system to enter the OS.

This page is intentionally left blank.

# Drivers Installation

This section describes the installation procedures for software and drivers. The software and drivers are included with the motherboard. If you find the items missing, please contact the vendor where you made the purchase. The contents of this section include the following:



**IMPORTANT NOTE:**
After installing your Windows operating system, you must install first the Intel Chipset Software Installation Utility before proceeding with the drivers installation.

# Intel Chipset Software Installation Utility

The Intel Chipset Drivers should be installed first before the software drivers to enable Plug & Play INF support for Intel chipset components. Follow the instructions below to complete the installation.

1. Insert the disc that comes with the board. Click ***Intel*** and then ***Intel(R) Cedarview Chipset Drivers***.


2. Click ***Intel(R) Chipset Software Installation Utility.***


3. When the Welcome screen to the Intel® Chipset Device Software appears, click ***Next*** to continue.

4. Click ***Yes*** to accept the software license agreement and proceed with the installation process.

5. On the Readme File Information screen, click ***Next*** to continue the installation.

6. The Setup process is now complete. Click ***Finish*** to restart the computer and for changes to take effect.

# VGA Drivers Installation

1. Insert the disc that comes with the board. Click ***Intel*** and then ***Intel(R) Cedarview Chipset Drivers***.


2. Click ***Intel(R) Cedarview Graphics Driver.***

3. When the Welcome screen appears, click ***Next*** to continue.


4. Click ***Yes*** to to agree with the license agreement and continue the installation.

5. On the Readme File Information screen, click ***Next*** to continue the installation of the Intel® Graphics Media Accelerator Driver.

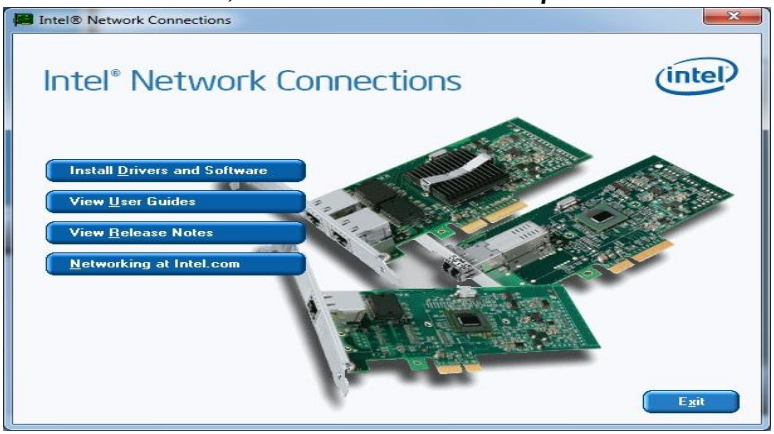
6. On Setup Progress screen, click ***Next*** to continue.

7. Setup complete. Click ***Finish*** to restart the computer and for changes to take effect.

# LAN Drivers Installation

1. Insert the disc that comes with the board. Click ***LAN Card*** and then ***Intel LAN Cedarview Drivers***.

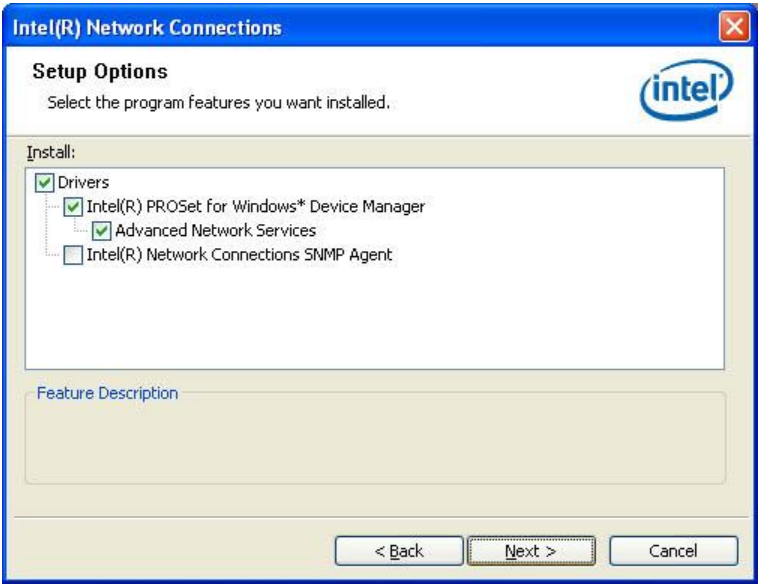

2. On the next screen, click ***Install Drivers and Software***.




3. In the Welcome screen, click ***Next***. On the next screen, click ***Yes*** to to agree with the license agreement.

4. Agree with the license agreement and click ***Next***.

5. Click the checkbox for **Drivers** in the Setup Options screen to select it and click **Next** to continue.




6. The wizard is ready to begin installation. Click ***Install*** to begin the installation.

7. When InstallShield Wizard is complete, click ***Finish***.

# Appendix

## A. I/O Port Address Map

Each peripheral device in the system is assigned a set of I/O port addresses which also becomes the identity of the device. The following table lists the I/O port addresses used.

| Address | Device Description |
|---|---|
| 000h - 01Fh | DMA Controller #1 |
| 020h - 03Fh | Interrupt Controller #1 |
| 040h - 05Fh | Timer |
| 060h - 06Fh | Keyboard Controller |
| 070h - 07Fh | Real Time Clock, NMI |
| 080h - 09Fh | DMA Page Register |
| 0A0h - 0BFh | Interrupt Controller #2 |
| 0C0h - 0DFh | DMA Controller #2 |
| 0F0h | Clear Math Coprocessor Busy Signal |
| 0F1h | Reset Math Coprocessor |
| 1F0h - 1F7h | IDE Interface |
| 2F8h - 2FFh | Serial Port #2(COM2) |
| 2B0h- 2DFh | Graphics adapter Controller |
| 360h - 36Fh | Network Ports |
| 3F8h - 3FFh | Serial Port #1(COM1) |

# B. Interrupt Request Lines (IRQ)

Peripheral devices use interrupt request lines to notify CPU for the service required. The following table shows the IRQ used by the devices on board.

| Level | Function |
|---|---|
| IRQ 0 | System timer |
| IRQ 1 | Standard PS/2 Keyboard |
| IRQ 3 | Communications Port (COM2) |
| IRQ 4 | Communications Port (COM1) |
| IRQ 5 | Communications Port (COM3) |
| IRQ 7 | Communications Port (COM4) |
| IRQ 8 | System CMOS/real time clock |
| IRQ 11 | Intel(R) N10/ICH7 Family SMBus Controller - 27DA |
| IRQ 12 | Microsoft PS/2 Mouse |
| IRQ 13 | Numeric data processor |
| IRQ 16 | Intel(R) N10/ICH7 Family USB Universal Host Controller - 27CB |
| IRQ 18 | Intel(R) N10/ICH7 Family USB Universal Host Controller - 27CA |
| IRQ 19 | Intel(R) N10/ICH7 Family Serial ATA Storage Controller - 27C0 |
| IRQ 19 | Intel(R) N10/ICH7 Family USB Universal Host Controller - 27C9 |
| IRQ 22 | High Definition Audio Controller |
| IRQ 23 | Intel(R) N10/ICH7 Family USB2 Enhanced Host Controller - 27CC |
| IRQ 23 | Intel(R) N10/ICH7 Family USB Universal Host Controller - 27C8 |
| IRQ 4294967290 | Intel(R) 82583V Gigabit Network Connection |
| IRQ 4294967291 | Intel(R) 82583V Gigabit Network Connection #2 |
| IRQ 4294967292 | Intel(R) Graphics Media Accelerator 3600 Series |
| IRQ 4294967293 | Intel(R) N10/ICH7 Family PCI Express Root Port - 27D2 |
| IRQ 4294967294 | Intel(R) N10/ICH7 Family PCI Express Root Port - 27D0 |

# C. Watchdog Timer Configuration

The WDT is used to generate a variety of output signals after a user programmable count. The WDT is suitable for use in the prevention of system lock-up, such as when software becomes trapped in a deadlock. Under these sorts of circumstances, the timer will count to zero and the selected outputs will be driven. Under normal circumstance, the user will restart the WDT at regular intervals before the timer counts to zero.

SAMPLE CODE:

```
File of the W627UHG.CPP
//-----------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//-----------------------------------------------------------------------------
#include "W627UHG.H"
#include <dos.h>
//-----------------------------------------------------------------------------
unsigned int W627UHG_BASE;
void Unlock_W627UHG (void);
void Lock_W627UHG (void);
//-----------------------------------------------------------------------------
unsigned int Init_W627UHG(void)
{
        unsigned int result;
        unsigned char ucDid;

        W627UHG_BASE = 0x4E;
        result = W627UHG_BASE;

        ucDid = Get_W627UHG_Reg(0x20);
        if (ucDid == 0xA2)                              //W83627UHG??
        {       goto Init_Finish;    }

        W627UHG_BASE = 0x2E;
        result = W627UHG_BASE;

        ucDid = Get_W627UHG_Reg(0x20);
        if (ucDid == 0xA2)                              //W83627UHG??
        {       goto Init_Finish;    }

        W627UHG_BASE = 0x00;
        result = W627UHG_BASE;

Init_Finish:
        return (result);
}
//-----------------------------------------------------------------------------
void Unlock_W627UHG (void)
```

```
{
        outportb(W627UHG_INDEX_PORT, W627UHG_UNLOCK);
        outportb(W627UHG_INDEX_PORT, W627UHG_UNLOCK);
}
//-----------------------------------------------------------------------------
void Lock_W627UHG (void)
{
        outportb(W627UHG_INDEX_PORT, W627UHG_LOCK);
}
//-----------------------------------------------------------------------------
void Set_W627UHG_LD( unsigned char LD)
{
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, W627UHG_REG_LD);
        outportb(W627UHG_DATA_PORT, LD);
        Lock_W627UHG();
}
//-----------------------------------------------------------------------------
void Set_W627UHG_Reg( unsigned char REG, unsigned char DATA)
{
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, REG);
        outportb(W627UHG_DATA_PORT, DATA);
        Lock_W627UHG();
}
//-----------------------------------------------------------------------------
unsigned char Get_W627UHG_Reg(unsigned char REG)
{
        unsigned char Result;
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, REG);
        Result = inportb(W627UHG_DATA_PORT);
        Lock_W627UHG();
        return Result;
}
//-----------------------------------------------------------------------------
```

File of the W627UHG.H

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#ifndef __W627UHG_H
#define __W627UHG_H                1
//---------------------------------------------------------------------------
#define       W627UHG_INDEX_PORT            (W627UHG_BASE)
#define       W627UHG_DATA_PORT             (W627UHG_BASE+1)
//---------------------------------------------------------------------------
#define       W627UHG_REG_LD                0x07
//---------------------------------------------------------------------------
#define W627UHG_UNLOCK                0x87
#define       W627UHG_LOCK                  0xAA
//---------------------------------------------------------------------------
unsigned int Init_W627UHG(void);
void Set_W627UHG_LD( unsigned char);
void Set_W627UHG_Reg( unsigned char, unsigned char);
unsigned char Get_W627UHG_Reg( unsigned char);
//---------------------------------------------------------------------------
#endif//__W627UHG_H
```

File of the MAIN.CPP

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#include <dos.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include "W627UHG.H"
//---------------------------------------------------------------------------
int main (void);

void WDTInitial(void);
void WDTEnable(unsigned char);
void WDTDisable(void);

//---------------------------------------------------------------------------
int main (void)
{
        char SIO;

        SIO = Init_W627UHG();
        if (SIO == 0)
        {
        ..... printf("Can not detect Winbond 83627UHG, program abort.\n");
        ..... return(1);
        }

        WDTInitial();

        WDTEnable(10);

        WDTDisable();

        return 0;
}
//---------------------------------------------------------------------------
void WDTInitial(void)
{
        unsigned char bBuf;
        Set_W627UHG_LD(0x08); .....//switch to logic device 8
        bBuf = Get_W627UHG_Reg(0x30);
        bBuf &= (~0x01);
        Set_W627UHG_Reg(0x30, bBuf); ..... //Enable WDTO
}
//---------------------------------------------------------------------------
void WDTEnable(unsigned char NewInterval)
{
        unsigned char bBuf;

        Set_W627UHG_LD(0x08); .....//switch to logic device 8
        Set_W627UHG_Reg(0x30, 0x01); ..... //enable timer
```

```
        bBuf = Get_W627UHG_Reg(0xF5);
        bBuf &= (~0x08);
        Set_W627UHG_Reg(0xF5, bBuf); ..... //count mode is second

        Set_W627UHG_Reg(0xF6, NewInterval);..... //set timer
}
//-------------------------------------------------------------------------
void WDTDisable(void)
{
        Set_W627UHG_LD(0x08); .....//switch to logic device 8
        Set_W627UHG_Reg(0xF6, 0x00); ..... //clear watchdog timer
        Set_W627UHG_Reg(0x30, 0x00); .....//watchdog disabled
}
//-------------------------------------------------------------------------
```

# D. Digital I/O Sample Code

File of the W627UHG.H

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#ifndef __W627UHG_H
#define __W627UHG_H                1
//---------------------------------------------------------------------------
#define     W627UHG_INDEX_PORT          (W627UHG_BASE)
#define     W627UHG_DATA_PORT           (W627UHG_BASE+1)
//---------------------------------------------------------------------------
#define     W627UHG_REG_LD              0x07
//---------------------------------------------------------------------------
#define W627UHG_UNLOCK                  0x87
#define     W627UHG_LOCK                0xAA
//---------------------------------------------------------------------------
unsigned int Init_W627UHG(void);
void Set_W627UHG_LD( unsigned char);
void Set_W627UHG_Reg( unsigned char, unsigned char);
unsigned char Get_W627UHG_Reg( unsigned char);
//---------------------------------------------------------------------------
#endif//__W627UHG_H
```

File of the W627UHG.CPP

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#include "W627UHG.H"
#include <dos.h>
//---------------------------------------------------------------------------
unsigned int W627UHG_BASE;
void Unlock_W627UHG (void);
void Lock_W627UHG (void);
//---------------------------------------------------------------------------
unsigned int Init_W627UHG(void)
{
        unsigned int result;
        unsigned char ucDid;

        W627UHG_BASE = 0x4E;
        result = W627UHG_BASE;

        ucDid = Get_W627UHG_Reg(0x20);
        if (ucDid == 0xA2)                                      //W83627UHG??
        {       goto Init_Finish;    }

        W627UHG_BASE = 0x2E;
        result = W627UHG_BASE;

        ucDid = Get_W627UHG_Reg(0x20);
        if (ucDid == 0xA2)                                      //W83627UHG??
        {       goto Init_Finish;    }

        W627UHG_BASE = 0x00;
        result = W627UHG_BASE;

Init_Finish:
        return (result);
}
//---------------------------------------------------------------------------
void Unlock_W627UHG (void)
{
        outportb(W627UHG_INDEX_PORT, W627UHG_UNLOCK);
        outportb(W627UHG_INDEX_PORT, W627UHG_UNLOCK);
}
//---------------------------------------------------------------------------
void Lock_W627UHG (void)
{
        outportb(W627UHG_INDEX_PORT, W627UHG_LOCK);
}
//---------------------------------------------------------------------------
void Set_W627UHG_LD( unsigned char LD)
```

```
{
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, W627UHG_REG_LD);
        outportb(W627UHG_DATA_PORT, LD);
        Lock_W627UHG();
}
//-----------------------------------------------------------------------------
void Set_W627UHG_Reg( unsigned char REG, unsigned char DATA)
{
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, REG);
        outportb(W627UHG_DATA_PORT, DATA);
        Lock_W627UHG();
}
//-----------------------------------------------------------------------------
unsigned char Get_W627UHG_Reg(unsigned char REG)
{
        unsigned char Result;
        Unlock_W627UHG();
        outportb(W627UHG_INDEX_PORT, REG);
        Result = inportb(W627UHG_DATA_PORT);
        Lock_W627UHG();
        return Result;
}
//-----------------------------------------------------------------------------
```

File of the MAIN.CPP

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#include <dos.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include "W627UHG.H"
//---------------------------------------------------------------------------
int main (void);

void Dio5Initial(void);
void Dio5SetOutput(unsigned char);
unsigned char Dio5GetInput(void);
void Dio5SetDirection(unsigned char);
unsigned char Dio5GetDirection(void);
//---------------------------------------------------------------------------
int main (void)
{
	char SIO;

	SIO = Init_W627UHG();
	if (SIO == 0)
	{
		printf("Can not detect Winbond 83627UHG, program abort.\n");
		return(1);
	}

	Dio5Initial();

	//for GPIO50..57
	Dio5SetDirection(0x0F); //GP50..53 = input, GP54..57=output
	printf("Current DIO direction = 0x%X\n", Dio5GetDirection());

	printf("Current DIO status = 0x%X\n", Dio5GetInput());

	printf("Set DIO output to high\n");
	Dio5SetOutput(0x0F);

	printf("Set DIO output to low\n");
	Dio5SetOutput(0x00);

	return 0;
}
//---------------------------------------------------------------------------
void Dio5Initial(void)
{
	unsigned char ucBuf;

	Set_W627UHG_LD(0x08);                    //switch to logic device 8
	//enable the GP5 group
	ucBuf = Get_W627UHG_Reg(0x30);
	ucBuf |= 0x02;
```

```
	Set_W627UHG_Reg(0x30, ucBuf);
}
//---------------------------------------------------------------------------
void Dio5SetOutput(unsigned char NewData)
{
	Set_W627UHG_LD(0x08);				//switch to logic device 8
	Set_W627UHG_Reg(0xE1, NewData);
}
//---------------------------------------------------------------------------
unsigned char Dio5GetInput(void)
{
	unsigned char result;

	Set_W627UHG_LD(0x08);				//switch to logic device 8
	result = Get_W627UHG_Reg(0xE1);
	return (result);
}
//---------------------------------------------------------------------------
void Dio5SetDirection(unsigned char NewData)
{
	//NewData : 1 for input, 0 for output
	Set_W627UHG_LD(0x08);				//switch to logic device 8
	Set_W627UHG_Reg(0xE0, NewData);
}
//---------------------------------------------------------------------------
unsigned char Dio5GetDirection(void)
{
	unsigned char result;

	Set_W627UHG_LD(0x08);				//switch to logic device 8
	result = Get_W627UHG_Reg(0xE0);
	return (result);
}
//---------------------------------------------------------------------------
```