FWA8207 Series

Networking Appliance

User's Manual

Version: 1.3

Table of Contents

Foreword

To prevent damage to the system board, please handle it with care and follow the measures below, which are generally sufficient to protect your equipment from static electricity discharge:

When handling the board, use a grounded wrist strap designed for static discharge elimination grounded to a metal object before removing the board from the antistatic bag. Handle the board by its edges only; do not touch its components, peripheral chips, memory modules or gold contacts.

When handling processor chips or memory modules, avoid touching their pins or gold edge fingers. Return the Network Appliance system board and peripherals back into the antistatic bag when not in use or not installed in the chassis.

Some circuitry on the system board can continue to operate even though the power is switched off. Under no circumstances should the Lithium battery cell used to power the real-time clock be allowed to be shorted. The battery cell may heat up under these conditions and present a burn hazard.

WARNING!

1. "CAUTION: DANGER OF EXPLOSION IF BATTERY IS INCORRECTLY REPLACED.

   REPLACE ONLY WITH SAME OR EQUIVALENT TYPE RECOMMENDED BY THE MANUFACTURER. DISCARD USED BATTERIES ACCORDING TO THE MANUFACTURER'S INSTRUCTIONS"
2. This guide is for technically qualified personnel who have experience installing and configuring system boards. Disconnect the system board power supply from its power source before you connect/disconnect cables or install/remove any system board components. Failure to do this can result in personnel injury or equipment damage.
3. Avoid short-circuiting the lithium battery; this can cause it to superheat and cause burns if touched.
4. Do not operate the processor without a thermal solution. Damage to the processor can occur in seconds.
5. Do not block air vents at least minimum 1/2-inch clearance required.

Chapter 1 Introduction

FWA8207 series was specifically designed for the network security & management market.

Network Security Applications:

- Firewall
- Unified Threat Management (UTM)
- Virtual Private Network (VPN)
- Proxy Server
- Caching Server

Network Management Applications:

- Load balancing
- Quality of Service
- Remote Access Service

The FWA networking appliance product line covers the spectrum from offering platforms designed for:

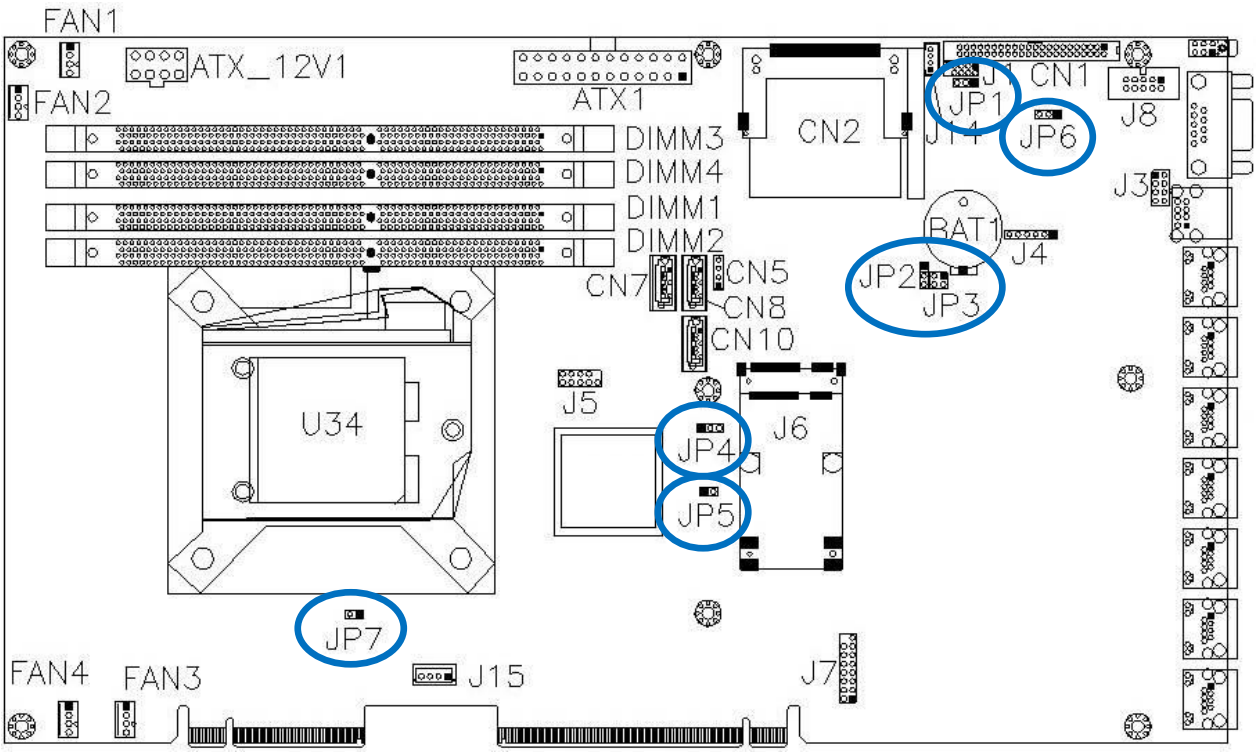
- SOHO
- SMB
- Enterprise

Each product is designed to address the distinctive requirements of its respective market segment from cost effective entry-level solutions to high throughput and performance-bound systems for the Enterprise level.

Chapter 2 System Specification

Product Name	FWA8207
Form Factor	19” 1U Mainstream Networking Product
Motherboard	MB966
CPU	Intel® LGA1156 Series Processors
Chipset	Intel® 3450 PCH
Supported CPUs	● Intel® Xeon X3450, X3430 ● Intel® Core i7-860 ● Intel® Core i5-750 ● Intel® Core i5-660 (FWA8207-G) ● Intel® Core i3-540 (FWA8207-G) ● Intel® Pentium G6950 (FWA8207-G)
Network	● Six onboard GLAN + one Management (ATM 6.0) ● Two segments hardware Bypass
Expansion Slot	● One PCI-e x8 Golden Finger ● One PCI-e x16 (x8 Link) Golden Finger ● CF Card Socket ● Mini PCI-e Socket (m-SATA compatible)
Storage	One internal 2.5” HDD (FWA8207) or One internal 3.5” HDD (FWA8207-2SLOT & FWA8207-G)
Front Panel	● DB-9 Console Port (COM1) ● 2x USB 2.0 type A connector ● 1x RJ-45 for Management port (ATM 6.0) ● 6x RJ-45 with Link/Act, Speed LED for 10/100/1000M Ethernet ● 3x LED (Power, Status)
Rear Panel	● PSU inlet ● 1x or 2x Slot (Depend on product SKU)
USB 2.0	● Two in front ● Two pin header on board
ATM	ATM 6.0
TPM	Winbond WPCT200 TPM1.2 controller for Trust Platform 1.2
VGA	Pin header on board (FWA8207-G)
LCM	2x16 characters LCM
Watchdog Timer	256 segments, 0, 1, 2…255 sec/min
Power Supply	300W Single PSU
Dimensions	44 (H) x 440 (W) x 406.5 (D) mm
Operation Temperature	0 ~ 45 ° C
Storage Temperature	-20 ~ 70 ° C
Operation Humidity	5% ~ 95%
Certifications	CE, FCC
Optional Front Expansion Cards	● IBP161: 4-port RJ-45 10/100/1000 Copper Ethernet Card

MB966 Motherboard Layout



Jumper Setting

JP1: TPM Enable/Disable Setting

JP2, JP3: Watchdog Timer & CN11~CN14 LAN Bypass Settings

JP4: Clear CMOS Contents

JP5: ME (Intel® Management Engine) Enable/Disable

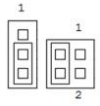
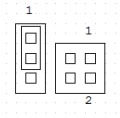
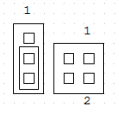
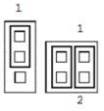
JP6: ATX / AT Power Mode Setting

JP7: PCIE1 & PCIE2 Golden Finger PCIe Configuration

JP1: TPM Enable/Disable Setting

JP1	Setting	Function
	Pin 1-2 Short/Closed	Enable
	Pin 2-3 Short/Closed	Disable

JP2, JP3: Watchdog Timer & LAN Bypass Settings

JP2, JP3	Setting	Function	Power Off	Power On OS Run Software
	JP2 Pin 2-3 Closed JP3 Pin 1-2 Open & 3-4 Closed	System LAN bypass function is controlled by Super I/O GP23	LAN Bypass	GP23 Active: Low: Bypass High: Normal
		System will reboot upon the time out of watchdog timer.		WDT Reboot System
	JP2 Pin 1-2 Closed JP3 Pin 1-2 & 3-4 Open	System will bypass LAN upon the time out of watchdog timer.	LAN Bypass	
 **(Default)**	JP2 Pin 2-3 Closed JP3 Pin 1-2 & 3-4 Open	System LAN bypass function is controlled by Super I/O GP23.		GP23 Active: Low: Bypass High: Normal
	JP2 Pin 1-2 Closed JP3 Pin 1-2 & 3-4 Closed	System LAN is at normal	LAN Bypass	LAN Always Normal
		System will reboot upon the time out of watchdog timer.		WDT Reboot System

JP4: Clear CMOS Contents

Use JP4 to clear the CMOS contents.

Note that the ATX-power connector should be disconnected from the board before clearing CMOS.

JP4	Setting	Function
	Pin 1-2 Short/Closed	Normal
	Pin 2-3 Short/Closed	Clear CMOS

JP5: ME (Intel® Management Engine) Enable/Disable

JP5	Setting	Function
	Open	Enable
	Short/Closed	Disable

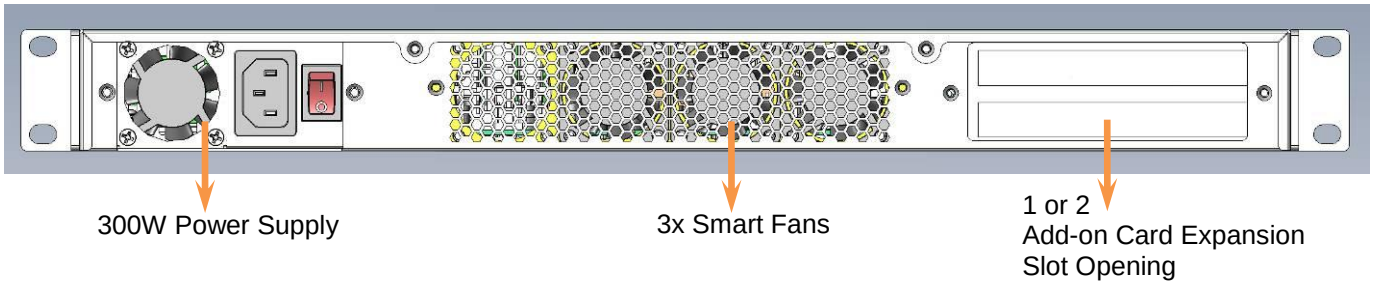
JP6: ATX / AT Power Mode Setting

JP1	Setting	Function
	Pin 1-2 Short/Closed	ATX
	Pin 2-3 Short/Closed	AT

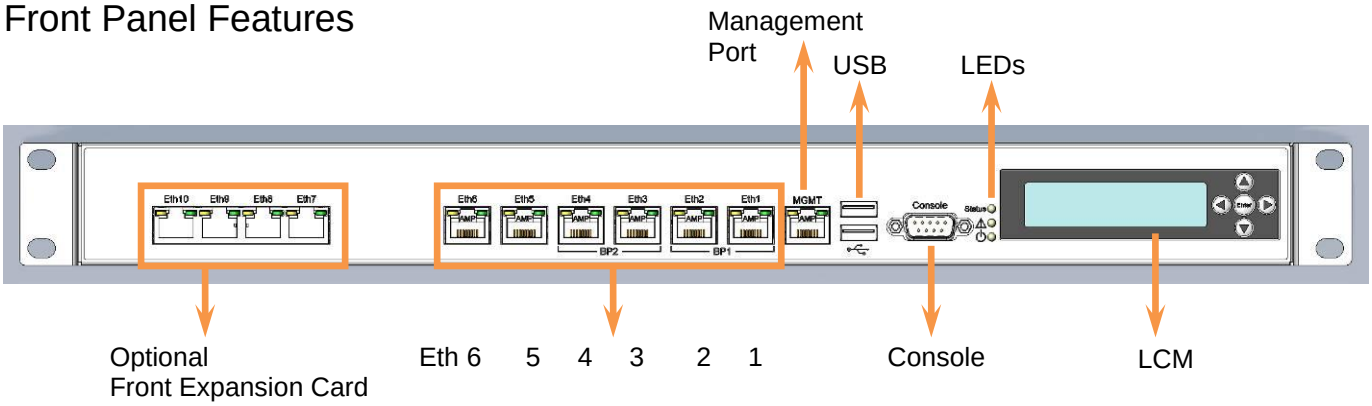
JP7: PCIE1 & PCIE2 Golden Finger PCIe Configuration

JP7	Setting	Function	Remarks
	Open	Combine to 1x16	For CPU with Integrated Graphics support
	Short/Closed	Separate to2x8	Default Setting for CPU without Integrated Graphics support

Rear Panel Features



Front Panel Features



Chapter 4 Console Mode Information

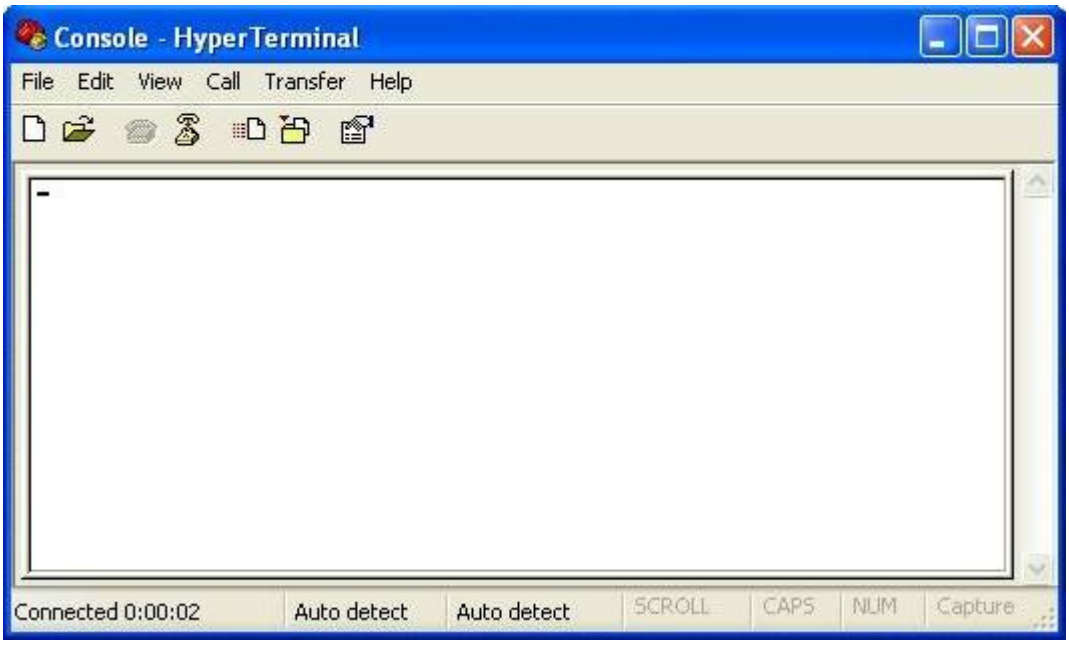
FWA8207 supports output information via Console in BIOS level.

Prepare a computer as client loaded with an existing OS such Windows XP.
Connect client computer and FWA8207 with NULL Modem cable.
Follow the steps below to configure the Windows Hyper Terminal application setting:

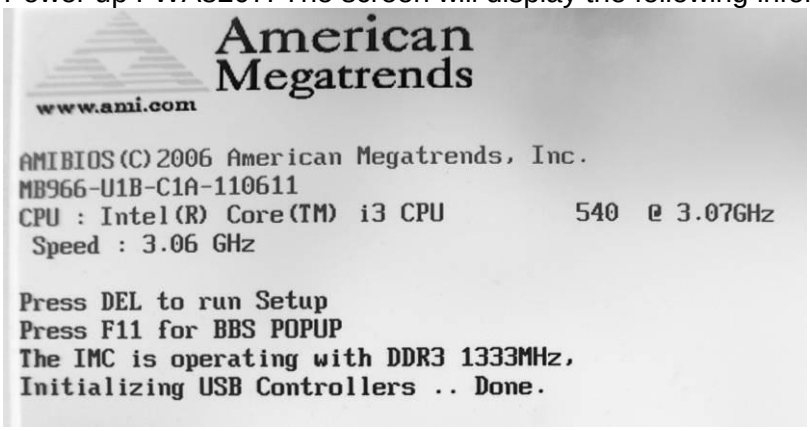
1. Execute Hyper Terminal. Issue command “hypertrm”.
2. Customize your name for the new connection.

3. Choose COM port on the client computer for the connection.

4. Please make the port settings to Baud rate 19200, Parity None, Data bits 8, Stop bits 1



5. Power up FWA8207. The screen will display the following information.



6. Press <**Tab**> key to enter BIOS setup screen in **Console mode**.
   Press <**Del**> key to enter BIOS setup screen in **VGA mode**.

Chapter 5 Open the Chassis

Loosen six screws on sides and rear of chassis, and slide backward to remove the top lead.

Fig. 5-1 *Take off screws*

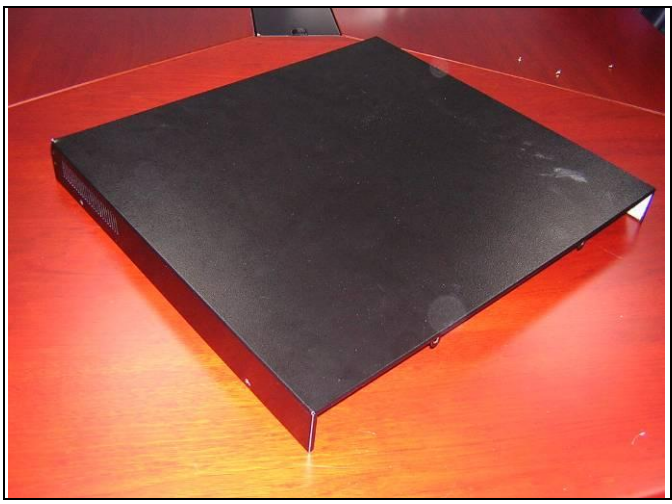


Fig. 5-2 *The top lead*

Fig. 5-3 *The base stand*

Chapter 6 Installing DDR3 Memory

Install system memory by pulling the socket's arm and pressing it into the slot gently.

Fig. 6-1 *Open both arms on DIMM socket*

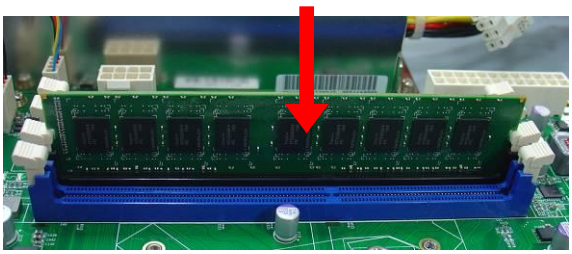


Fig. 6-2 *Install DIMM*

Notice:

1. **MB966 supports two groups of dual channels memory.**
   One group is on the black DIMM sockets, and the other one is blue DIMM sockets.
2. **The recommended height of memory module doesn't exceed 30 mm.**

Chapter 7 Installing CompactFlash Card

Insert CompactFlash card into the socket.

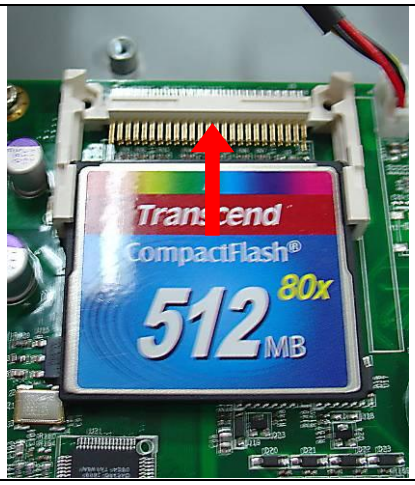


Fig. 7-1 *Insert CompactFlash Card into the CF interface*

Fig. 7-2 *Completion of CompactFlash Card connection*

Chapter 8 Removing and Installing the Battery

1. Press the metal clip back to eject the button battery.
2. Replace it with a new one by pressing the battery with fingertip to restore the battery

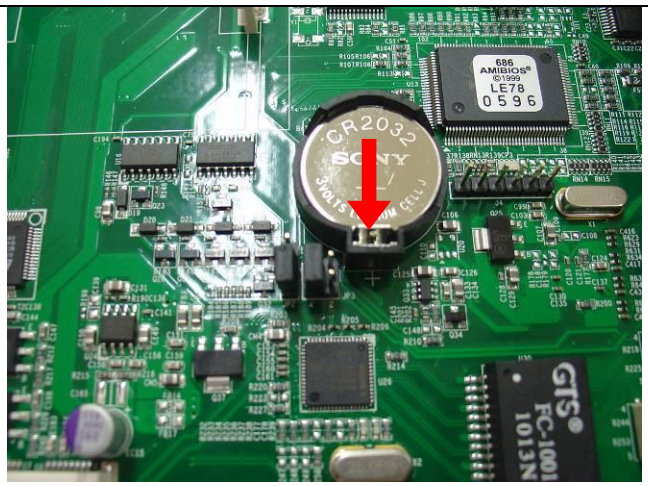


Fig. 8-1 Eject the battery and replace with new one

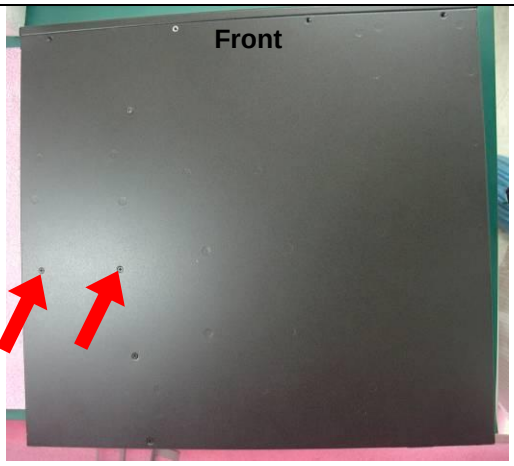


Fig. 9- *Take off two screws on bottom to remove 2.5" HDD bracket.*

Fig. 9-2 *Fasten the four screws to lock HDD and bracket together.*

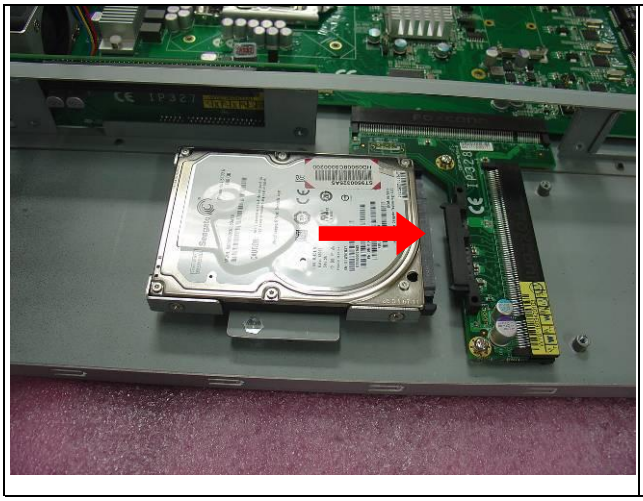


Fig. 9-3 *Push HDD into connector*

Fig. 9-4 *Completion of HDD connection*

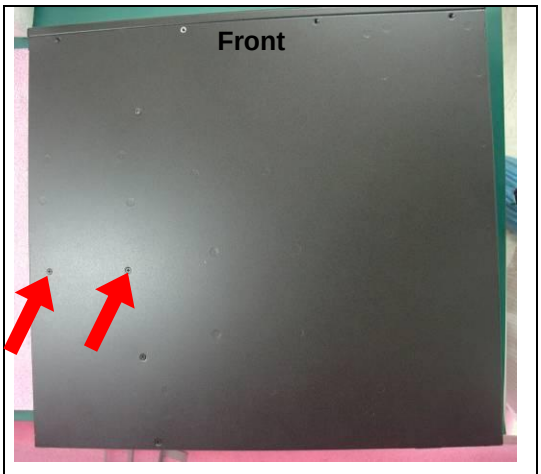


Fig. 9-5 *Fix HDD bracket with two screws*

Chapter 10 Installing Optional Dual 2.5” HDD Kit

The following is for optional Dual 2.5” HDD kit:

Fig. 10-1 Push eight shock-absorbent pads to fasten HDD bracket.

Fig. 10-2 Fasten the screws to lock 2.5” HDD bracket and bracket together.

Fig. 10-3 *Fix HDD bracket on chassis with four screws*

Chapter 11 Installing Add-on Card

Fig. 11-1 Loosen screw on slot bracket.

Fig. 11-2 Slide in PCI-e add-on card.

Fig. 11-3 *Fix the add-on card*

Chapter 12 Installing Mini PCI-e Card

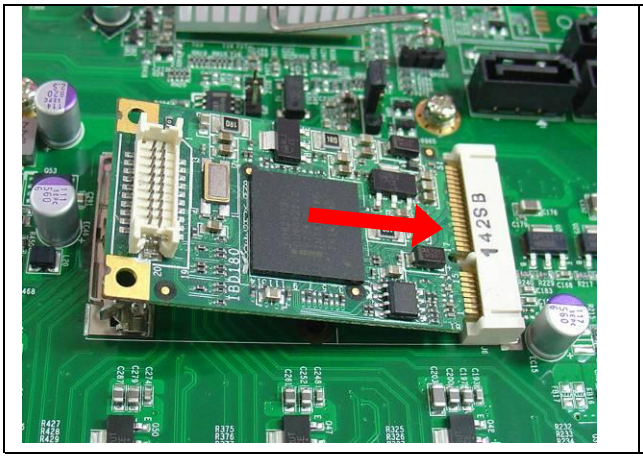


Fig. 12-1 Insert Mini PCI-e card.

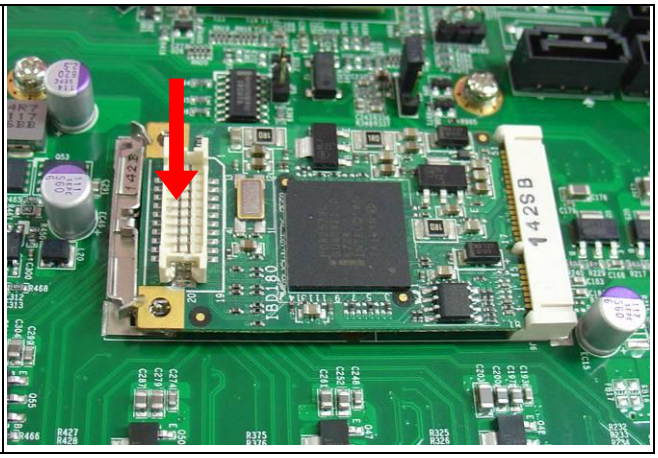
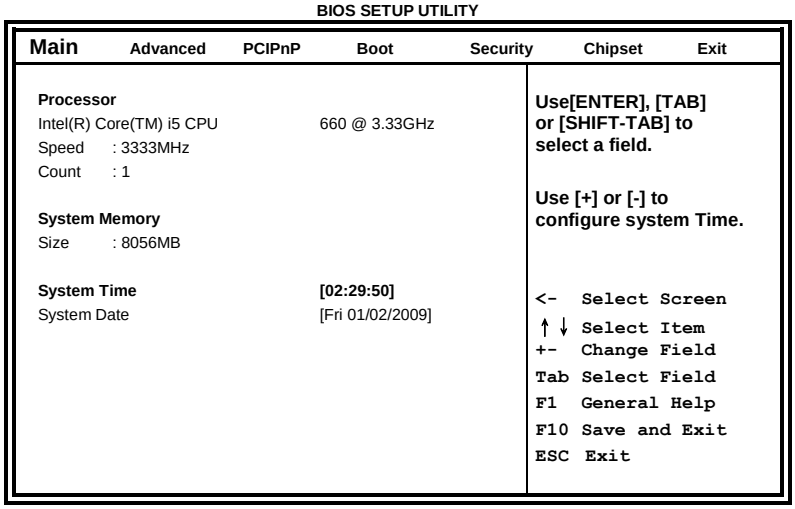


Fig. 12-2 Push down Mini PCI-e card.

Chapter 13 BIOS Information

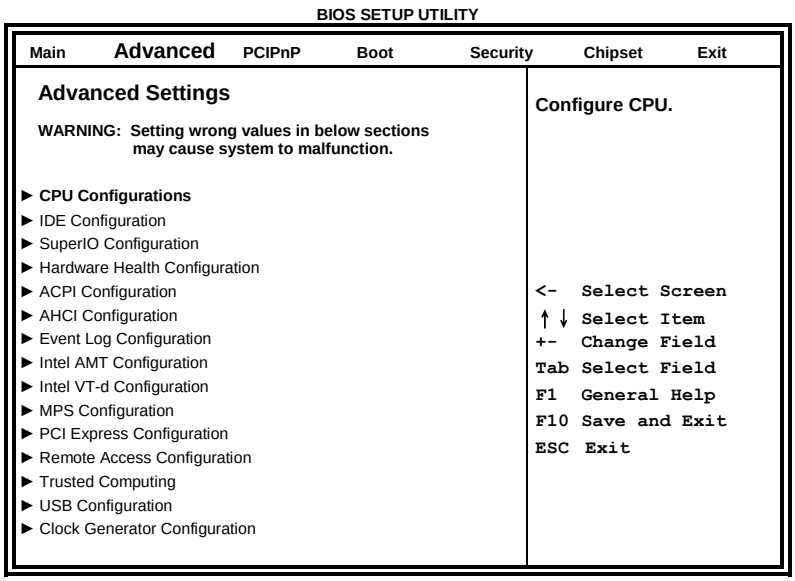
This setup allows you to view processor configuration used in your computer system and set the system time and date.



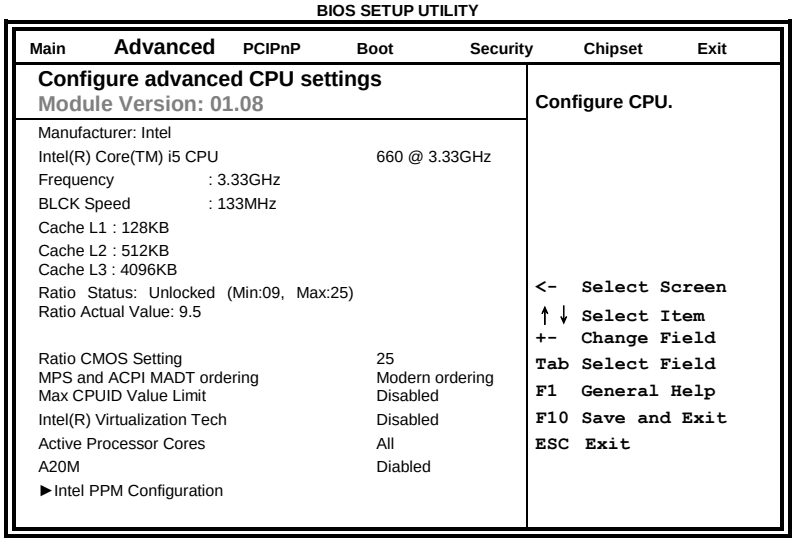
***Notice**: If the system cannot boot after making and saving system changes with Setup, the AMI BIOS supports an override to the CMOS settings that resets your system to its default.*

***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.*

Advanced Settings



The Advanced BIOS Settings configurations are shown in the following pages, as seen in the computer screen. Please note that setting the wrong values may cause the system to malfunction.



The CPU Configuration menu shows the following CPU details including the manufacturer, CPU type, its frequency and cache levels. Other options include:

Ratio CMOS Setting
Sets the ratio between CPU core clock and the FSB frequency.

MPS and ACPI MADT ordering
Modern ordering for Windows XP or later OSes. Legacy ordering for Windows 2000 or earlier OSes.

Max CPU ID Value Limit
Disabled for Windows XP.

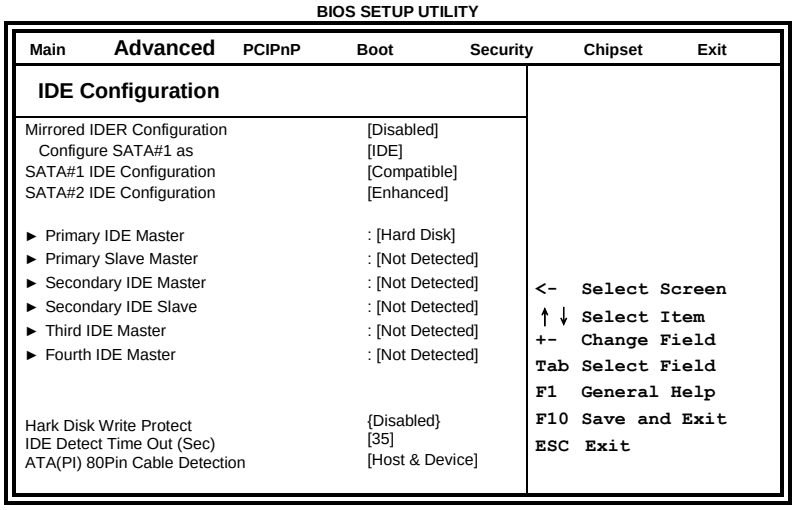
Intel Virtualization Tech
When enabled, a VMM can utilize the additional HW Caps. Provided by Intel® Vitualization Tech. Note: A full reset is required to change the setting.

Active Processor Cores
Number of cores to enable in each processor package.

A20M
Legacy OSes and Aps may need A20 M enabled.

Intel® PPM Configuration
This configuration includes the following options:

Intel SpeedStep tech
Disable: Disable GV3 Enable: Enable GV3
Intel TurboMode tech
Turbo mode allows processor cores to run faster than marked frequency in specific condition.
Intel C-STATE tech
CState: CPU idle is set to C2/C3/C4.
C State package limit setting
Selected option will program into C State package limit register.
C3 State / C6 State
Nehalem C state action select.
C1 Auto Demotion
When enabled, CPU will conditionally demote C3/C6/C7 requests to C1 based on uncore auto-demote information.
C3 Auto Demotion
When enabled, CPU will conditionally demote C6/C7 requests to C3 based on uncore auto-demote information.

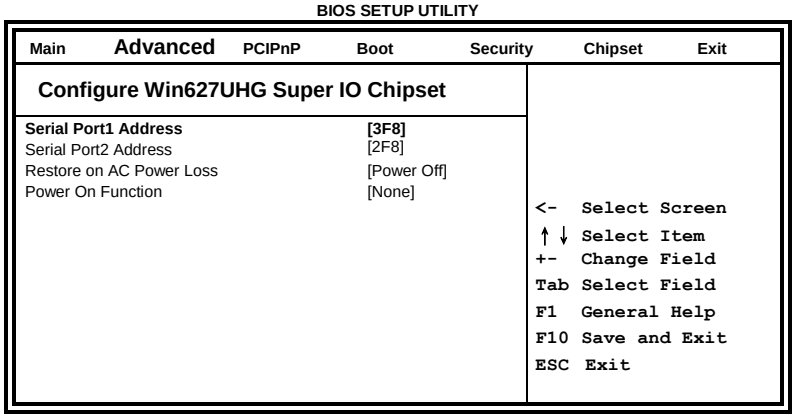


The IDE Configuration menu is used to change and/or set the configuration of the IDE devices installed in the system.

Hard Disk Write Protect
Disable/Enable device write protection. This will be effective only if device is accessed through BIOS.

IDE Detect Time Out (Sec)
Select the time out value for detecting ATA/ATAPI device(s).

ATA(PI) 80pin Cable Detection
Select the mechanism for detecting 80pin ATA(PI) cable.

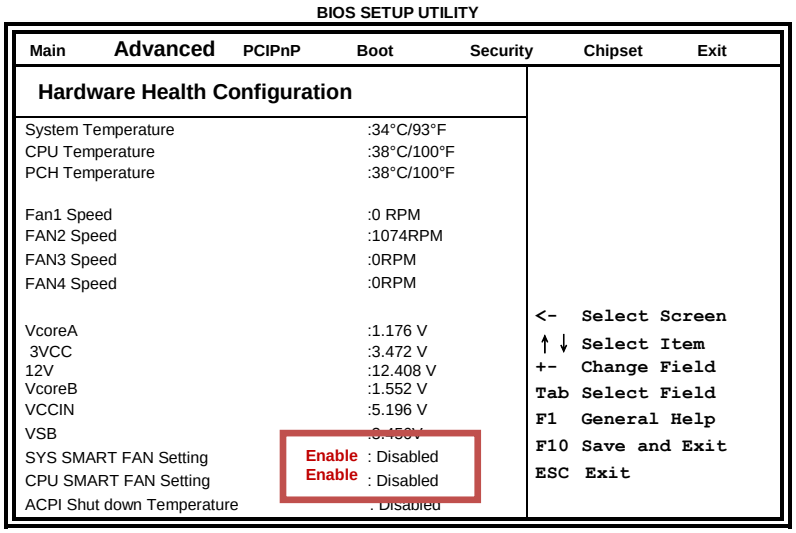


Onboard Serial Port/Parallel Port
These fields allow you to select the onboard serial ports and their addresses. The default values for these ports are:

Serial Port 1 3F8
Serial Port 2 2F8/

Restore on AC Power Loss
This field sets the system power status whether *on or off* when power returns to the system from a power failure situation.

Power On Function
This field is related to how the system is powered on. The options are *None, Mouse Left, Mouse Right,* and *Any Key.*

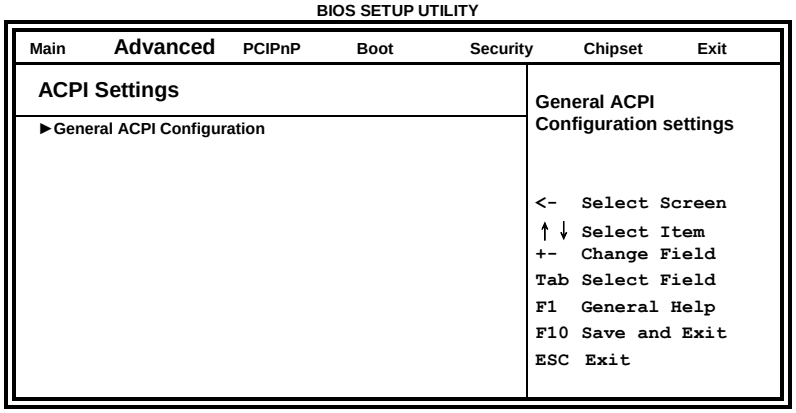


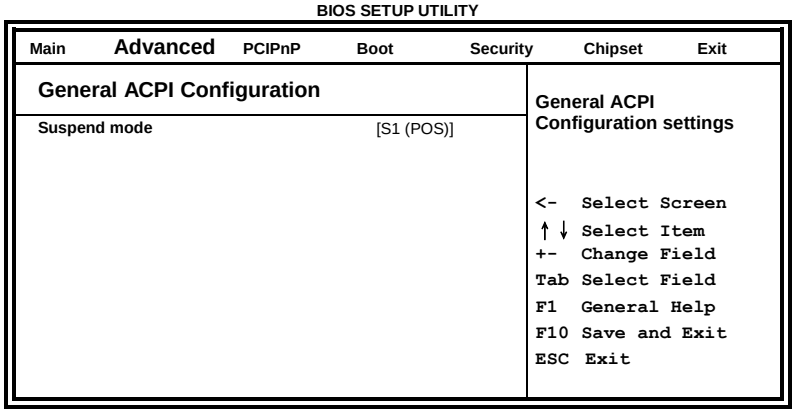
The Hardware Health Configuration menu is used to show the operating temperature, fan speeds and system voltages.

SYS smart fan
The options are *Disabled and Enabled (Default)*

CPU smart fan
The options are *Disabled and Enabled (Default)*

ACPI Shutdown Temperature
The options are *Disabled, 70ºC/158ºF, 75ºC/167ºF, 80ºC/176ºF, 85ºC/185ºF, 90ºC/194ºF,* and *95ºC/203ºF*.





Suspend Mode
The options of this field are *S1*, *S3*.

Repost Video on S3 Resume

Determines whether to invoke VGA BIOS POST on S3/STR resume.

BIOS SETUP UTILITY

Main	Advanced	PCIPnP	Boot	Security	Chipset	Exit

AHCI Settings

AHCI BIOS Support	Enabled
►AHCI Port0 [Not Detected]	
►AHCI Port1 [Not Detected]	
►AHCI Port2 [Not Detected]	
►AHCI Port3 [Not Detected]	
►AHCI Port4 [Not Detected]	
►AHCI Port05[Not Detected]	

<- Select Screen
↑↓ Select Item
+- Change Field
Tab Select Field
F1 General Help
F10 Save and Exit
ESC Exit

AHCI BIOS Support
Enables for supporting AHCI controller operates in AHCI mode during BIOS control otherwise operates in IDE mode

AHCI Port
While entering setup, BIOS auto detects the presence of IDE devices. This displays the status of auto detection of IDE devices.

BIOS SETUP UTILITY

Main	Advanced	PCIPnP	Boot	Security	Chipset	Exit

Event Logging details

View Event Log	
Mark all events as read	
Clear Event Log	
ECC Event Logging	[Disabled]
PCIE Error Log	[Disabled]

view all unread events on the Event Log

<- Select Screen
↑↓ Select Item
+- Change Field
Tab Select Field
F1 General Help
F10 Save and Exit
ESC Exit

BIOS SETUP UTILITY

Main	Advanced	PCIPnP	Boot	Security	Chipset	Exit

Intel AMT Configuration

Intel AMT Support	**[Enabled]**
AMT/ME BIOS Extension (MEBx) Configuration	
ME BIOS Extension (MEBx)	[Enabled]
Unconfigure AMT/ME	[Disabled]
MEBx Ctrl+P Delay (Seconds)	256

Options:
Disabled
Enabled

<- Select Screen
↑↓ Select Item
+- Change Field
Tab Select Field
F1 General Help
F10 Save and Exit
ESC Exit

The Intel® AMT Configuration configures the Intel® Active Management Technology (AMT) options.

BIOS SETUP UTILITY

Main	Advanced	PCIPnP	Boot	Security	Chipset	Exit

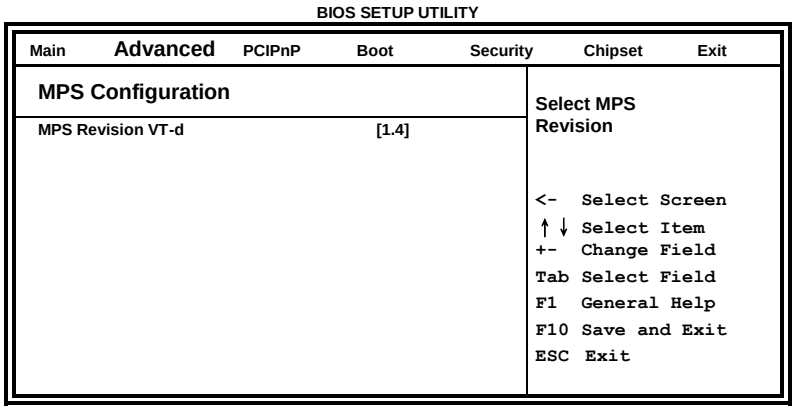
Intel VT-d	**[Disabled]**

Options:
Disabled
Enabled

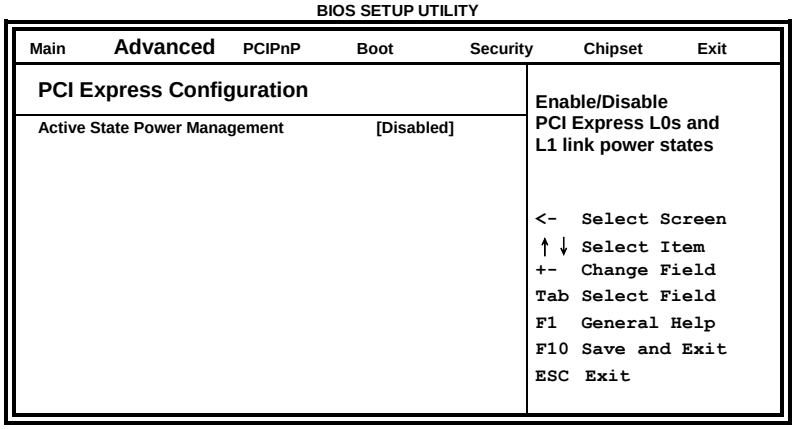
<- Select Screen
↑↓ Select Item
+- Change Field
Tab Select Field
F1 General Help
F10 Save and Exit
ESC Exit

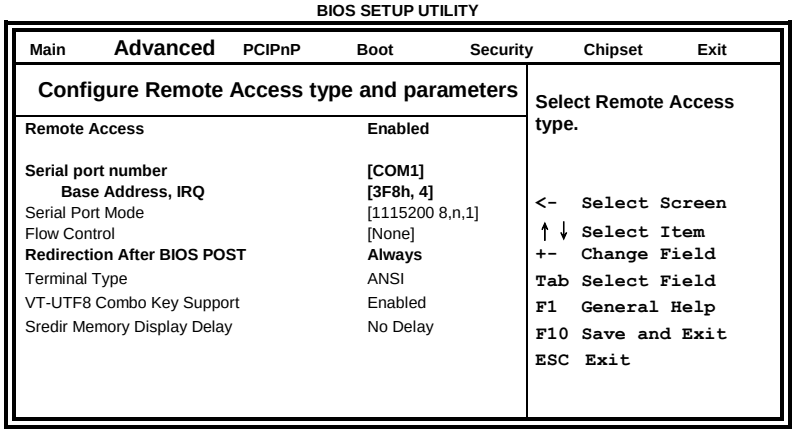
VT-d

Virtualization solutions allow multiple operating systems and applications to run in independent partitions all on a single computer. Using virtualization capabilities, one physical computer system can function as multiple "virtual" systems.



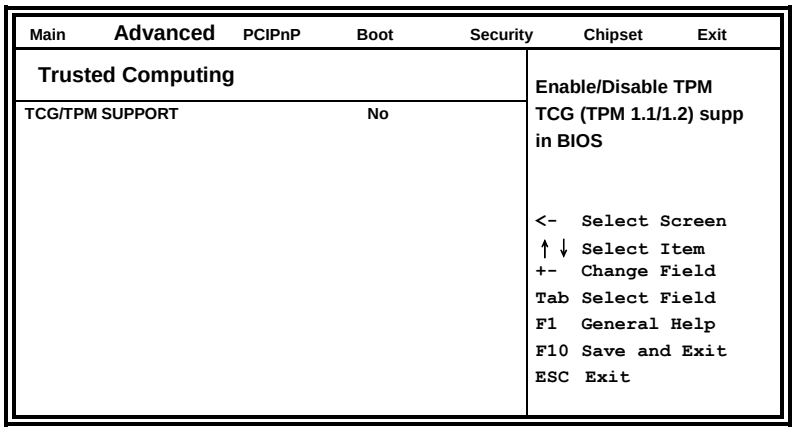
MPS Version Control for OS
This option is specifies the MPS (Multiprocessor Specification) version for your operating system. MPS version 1.4 added extended configuration tables to improve support for multiple PCI bus configurations and improve future expandability.
The default setting is ***1.4***.





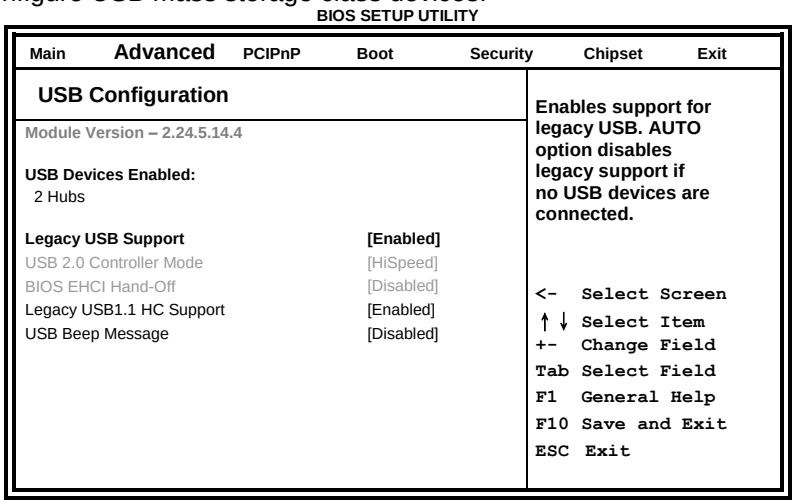
When enabled, the Remote Access type and parameters are shown:
Serial port number - Select Serial Port for console redirection.
Serial port mode - Select Serial Port settings.
Flow Control - Select Flow Control for console redirection.
Redirection After BIOS POST
Disable: Turns off the redirection after POST.
Boot Loader: Redirection is active during POST and during Boot Loader.
Always: Redirection is always active. (Some OSs may not work if set to Always.)
Terminal Type - Select the target terminal type.
VT-UTF8 Combo Key Support – Enable VT-UTF8 Combination Key Support for ANSI/VT100 terminals.

Sredir Memory Display Delay – Gives the delay in seconds to display memory information.



USB Configuration

This option is used to configure USB mass storage class devices.



Legacy USB Support

Enables support for legacy USB. AUTO option disables legacy support if no USB devices are connected.

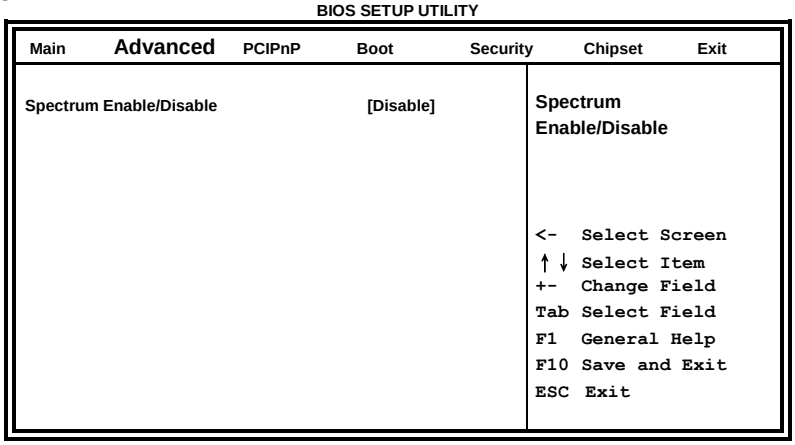
Legacy USB1.1 HC Support

Support USB 1.1 HC.

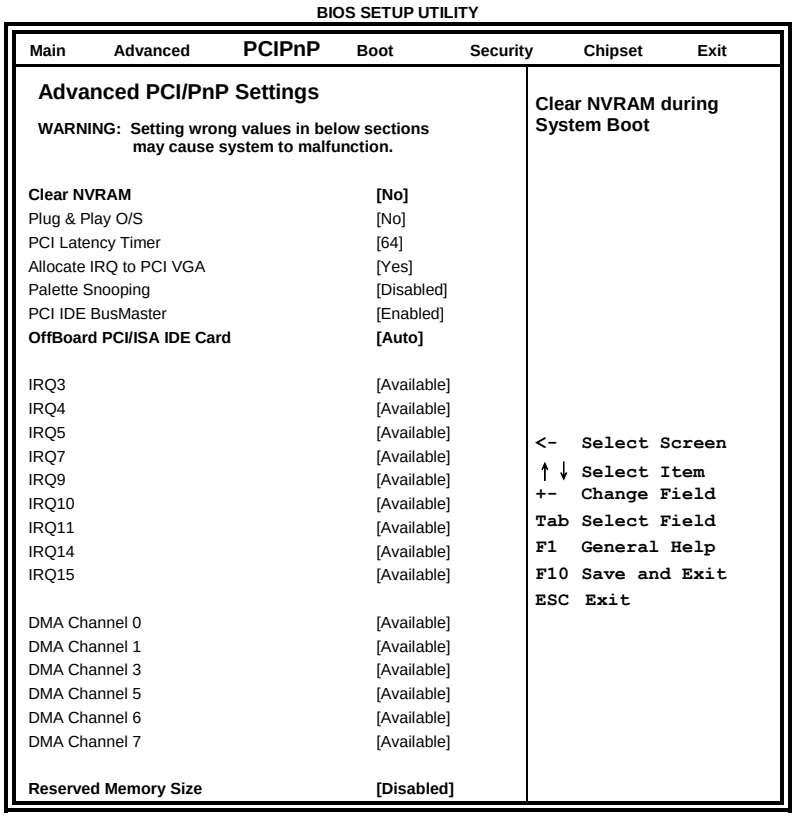
USB Beep Message

Enable the beep during USB device enumeration.

Clock Generator Configuration



PCIPnP Settings



Clear NVRAM
This item is used for clearing NVRAM during system boot.

Plug & Play O/S
This lets BIOS configure all devices in the system or lets the OS configure PnP devices not required for boot if your system has a Plug and Play OS.

PCI Latency Timer
This item sets value in units of PCI clocks for PCI device latency timer register. Options are: 32, 64, 96, 128, 160, 192, 224, 248.

Allocate IRQ to PCI VGA
This assigns IRQ to PCI VGA card if card requests IRQ or doesn't assign IRQ to PCI VGA card even if card requests an IRQ.

Palette Snooping
This informs the PCI devices that an ISA graphics device is installed in the system so the card will function correctly.

PCI IDE BusMaster
This uses PCI busmastering for BIOS reading / writing to IDE devices.

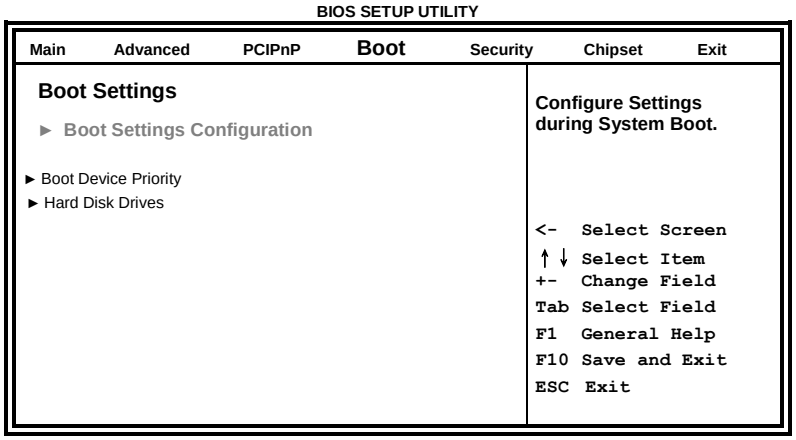
OffBoard PCI/ISA IDE Card
Some PCI IDE cards may require this to be set to the PCI slot number that is holiding the card. ***AUTO***: Works for most PCI IDE cards.

IRQ#
Use the IRQ# address to specify what IRQs can be assigned to a particular peripheral device.

Reserved Memory Size
Size of memory block to reserve legacy ISA devices.

Boot Settings

This option configures the settings during system boot including boot device priority and HDD/CD/DVD drives.



Boot Settings Configuration

This configuration includes the following items:
Quick Boot - Allows BIOS to skip certain tests while booting. This will decrease the time needed to boot the system.
Quite Boot – ***Disabled***: Displays normal POST messages. ***Enabled***: Displays OEM Logo instead of POST messages.
Bootup Num-Lock – Select Power-on state for Numlock.
PS/2 Mouse Support – Select support for PS/2 Mouse.
Wait for 'F1' If Error – Wait for F1 key to be pressed if error occurs.
Hit 'DEL' Message Display – Displays "Press DEL to run Setup" in POST.
Interrupt 19 Capture – This allows option ROMS to trap interrupt 19.

Boot Device Priority

This specifies the boot sequence from the available devices. A device enclosed in parenthesis has been disabled in the corresponding type menu.

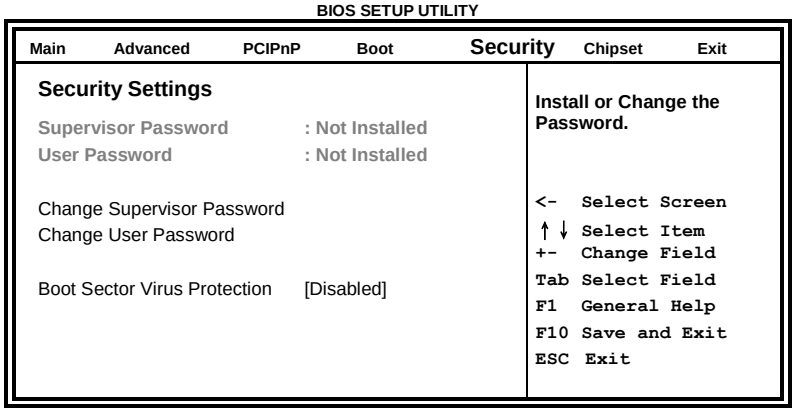
Hard Disk Drives

This specifies the Boot Device Priority sequence from available Hard Drives.

Security Settings

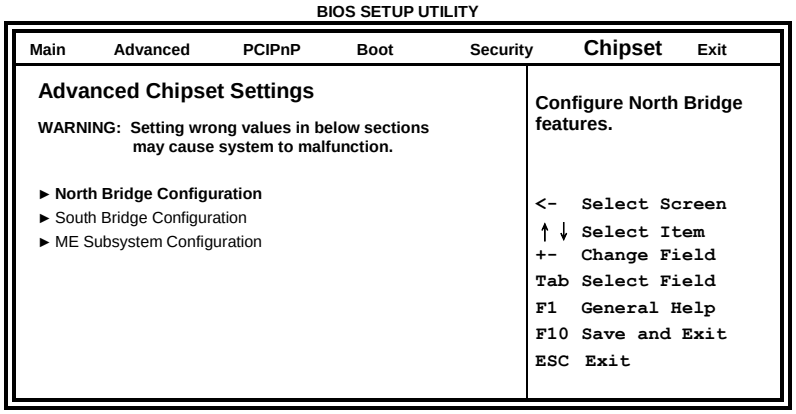
This setting comes with two options set the system password. Supervisor Password sets a password that will be used to protect the system and Setup utility. User Password sets a password that will be used exclusively on the system. To specify a password, highlight the type you want and press <Enter>. The Enter Password: message prompts on the screen. Type the password and press <Enter>. The system confirms your password by asking you to type it again. After setting a password, the screen automatically returns to the main screen.

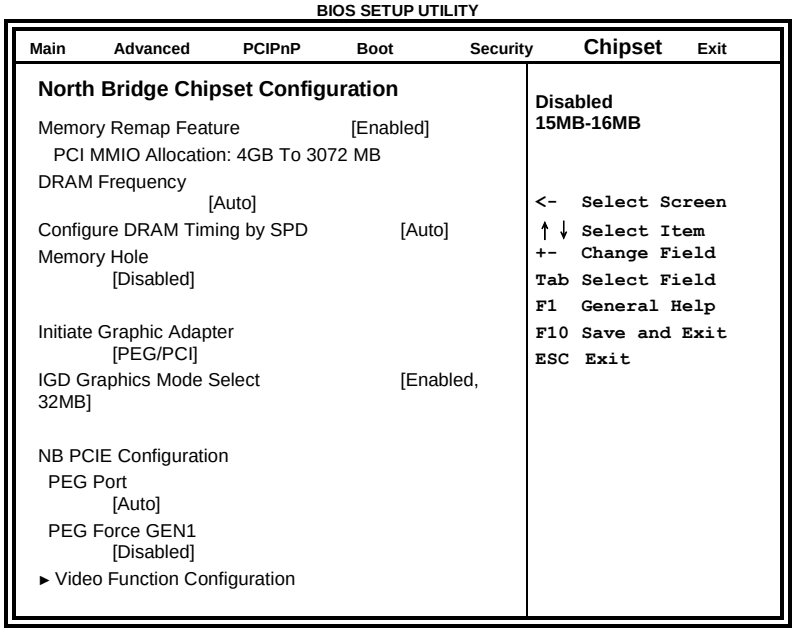
To disable a password, just press the <Enter> key when you are prompted to enter the password. A message will confirm the password to be disabled. Once the password is disabled, the system will boot and you can enter Setup freely.



Advanced Chipset Settings

This setting configures the north bridge, south bridge and the ME subsystem. WARNING! Setting the wrong values may cause the system to malfunction. -





Memory Remap Feature
This allows remapping of overlapped PCI memory above the total physical memory.

DRAM Frequency
The options are ***Auto***, ***1067 MHz*** and ***1333 MHz***.

Configure DRAM Timing by SPD
The options are ***Auto*** and ***Manual***.

Memory Hole
This option is used to reserve memory space between 15MB and 16MB for ISA expansion cards that require a specified area of memory to work properly.

Initiate Graphic Adapter
This option selects which graphics controller to use as the primary boot device.

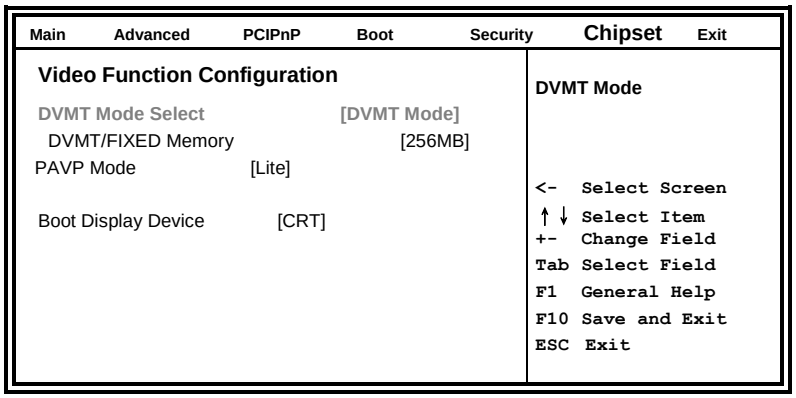
IGD Graphics Mode Select
This option selects the amount of system memory used by the internal graphics device.

PEG Port
The options are ***Auto*** and ***Disabled***.

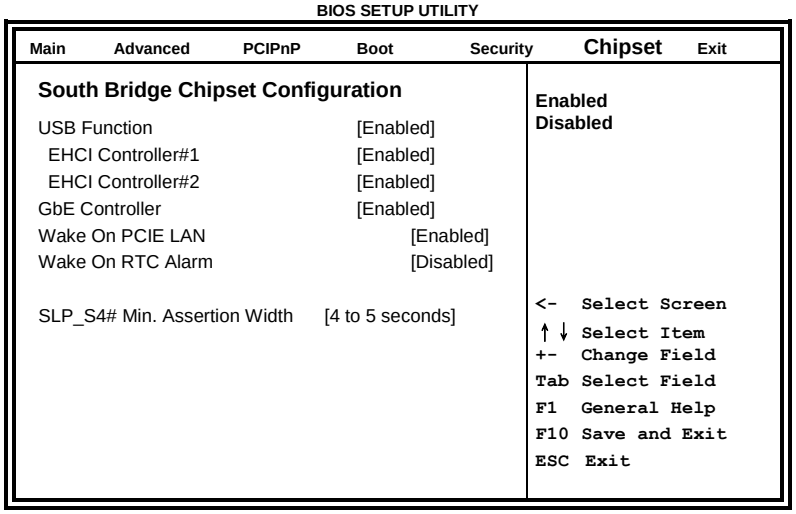
PEG Force GEN1
Some non-graphics PCI-E devices may not follow PCI-E specifications and may incorrectly report their GEN capability or link width.

Video Function Configuration

The configuration allows setting to DVMT/FIXED memory.



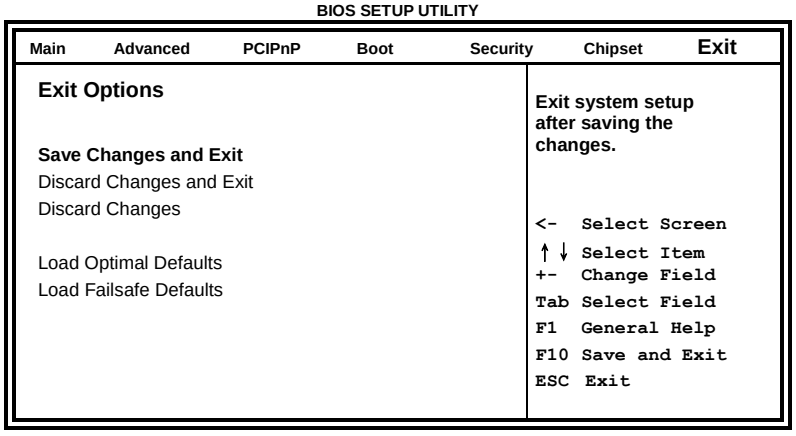
BIOS SETUP UTILITY



Exit Setup

The exit setup has the following settings that are:

BIOS SETUP UTILITY



Save Changes and Exit

This option allows you to determine whether or not to accept the modifications and save all changes into the CMOS memory before exit.

Discard Changes and Exit

This option allows you to exit the Setup utility without saving the changes you have made in this session.

Discard Changes

This option allows you to discard all the changes that you have made in this session.

Load Optimal Defaults

This option allows you to load the default values to your system configuration. These default settings are optimal and enable all high performance features.

Load Failsafe Defaults

This option allows you to load the troubleshooting default values permanently stored in the BIOS ROM. These default settings are non-optimal and disable all high-performance features.

Chapter 14 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:
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.

Filename：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 "W627DHG.H"
//---------------------------------------------------------------------------
int main (void);

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

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

	SIO = Init_W627DHG();
	if (SIO == 0)
	{
		printf("Can not detect Winbond 83627DHG, program abort.\n");
		return(1);
	}

	WDTInitial();

	WDTEnable(10);

	WDTDisable();

	return 0;
}
//---------------------------------------------------------------------------
void WDTInitial(void)
{
	unsigned char bBuf;

	bBuf = Get_W627DHG_Reg(0x2D);
	bBuf &= (~0x01);
	Set_W627DHG_Reg(0x2D, bBuf);			//Enable WDTO
}
//---------------------------------------------------------------------------
void WDTEnable(unsigned char NewInterval)
{
	unsigned char bBuf;

	Set_W627DHG_LD(0x08);				//switch to logic device 8
	Set_W627DHG_Reg(0x30, 0x01);			//enable timer

	bBuf = Get_W627DHG_Reg(0xF5);
	bBuf &= (~0x08);
	Set_W627DHG_Reg(0xF5, bBuf);			//count mode is second

	Set_W627DHG_Reg(0xF6, NewInterval);		//set timer
}
//---------------------------------------------------------------------------
void WDTDisable(void)
{
```

```
	Set_W627DHG_LD(0x08);                                   //switch to logic device 8
	Set_W627DHG_Reg(0xF6, 0x00);                            //clear watchdog timer
	Set_W627DHG_Reg(0x30, 0x00);                            //watchdog disabled
}
//----------------------------------------------------------------------------
```

Filename：W627DHG.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 "W627DHG.H"
#include <dos.h>
//----------------------------------------------------------------------------
unsigned int W627DHG_BASE;
void Unlock_W627DHG (void);
void Lock_W627DHG (void);
//----------------------------------------------------------------------------
unsigned int Init_W627DHG(void)
{
	unsigned int result;
	unsigned char ucDid;

	W627DHG_BASE = 0x4E;
	result = W627DHG_BASE;

	ucDid = Get_W627DHG_Reg(0x20);
	if (ucDid == 0xA0)                                      //W83627DHG
	{	goto Init_Finish;	}
	else if (ucDid == 0xB0)                                 //W83627DHG-P
	{	goto Init_Finish;	}

	W627DHG_BASE = 0x2E;
	result = W627DHG_BASE;

	ucDid = Get_W627DHG_Reg(0x20);
	if (ucDid == 0xA0)                                      //W83627DHG
	{	goto Init_Finish;	}
	else if (ucDid == 0xB0)                                 //W83627DHG-P
	{	goto Init_Finish;	}

	W627DHG_BASE = 0x00;
	result = W627DHG_BASE;

Init_Finish:
	return (result);
}
//----------------------------------------------------------------------------
void Unlock_W627DHG (void)
{
	outportb(W627DHG_INDEX_PORT, W627DHG_UNLOCK);
	outportb(W627DHG_INDEX_PORT, W627DHG_UNLOCK);
}
//----------------------------------------------------------------------------
void Lock_W627DHG (void)
{
	outportb(W627DHG_INDEX_PORT, W627DHG_LOCK);
}
//----------------------------------------------------------------------------
void Set_W627DHG_LD( unsigned char LD)
{
	Unlock_W627DHG();
	outportb(W627DHG_INDEX_PORT, W627DHG_REG_LD);
	outportb(W627DHG_DATA_PORT, LD);
	Lock_W627DHG();
}
//----------------------------------------------------------------------------
void Set_W627DHG_Reg( unsigned char REG, unsigned char DATA)
{
	Unlock_W627DHG();
	outportb(W627DHG_INDEX_PORT, REG);
	outportb(W627DHG_DATA_PORT, DATA);
	Lock_W627DHG();
}
//----------------------------------------------------------------------------
unsigned char Get_W627DHG_Reg(unsigned char REG)
{
```

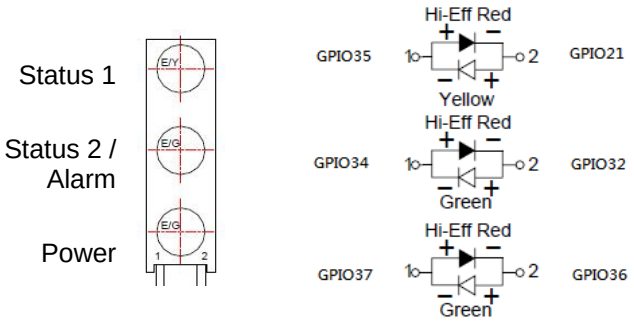
```
	unsigned char Result;
	Unlock_W627DHG();
	outportb(W627DHG_INDEX_PORT, REG);
	Result = inportb(W627DHG_DATA_PORT);
	Lock_W627DHG();
	return Result;
}
//----------------------------------------------------------------------------
```

Filename：W627DHG.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 __W627DHG_H
#define __W627DHG_H                     1
//----------------------------------------------------------------------------
#define    W627DHG_INDEX_PORT           (W627DHG_BASE)
#define    W627DHG_DATA_PORT            (W627DHG_BASE+1)
//----------------------------------------------------------------------------
#define    W627DHG_REG_LD               0x07
//----------------------------------------------------------------------------
#define W627DHG_UNLOCK                  0x87
#define    W627DHG_LOCK                 0xAA
//----------------------------------------------------------------------------
unsigned int Init_W627DHG(void);
void Set_W627DHG_LD( unsigned char);
void Set_W627DHG_Reg( unsigned char, unsigned char);
unsigned char Get_W627DHG_Reg( unsigned char);
//----------------------------------------------------------------------------
#endif      //__W627DHG_H
```

Chapter 15 LED GPIO Definition

This chapter describes GPIO definition of three LEDs on front panel.



Status1 LED	GPIO35	GPIO21
RED	H	L
YELLOW	L	H

Status2 / Alarm LED	GPIO34	GPIO32
RED	H	L
GREEN	L	H

Power LED	GPIO37	GPIO36
RED	H	L
GREEN	L	H

Digital I/O Sample Configuration

Filename：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 "W627DHG.H"
//---------------------------------------------------------------------------
int main (void);

void Dio2Initial(void);
void Dio2SetOutput(unsigned char);
unsigned char Dio2GetInput(void);
void Dio2SetDirection(unsigned char);
unsigned char Dio2GetDirection(void);

void Dio3Initial(void);
void Dio3SetOutput(unsigned char);
unsigned char Dio3GetInput(void);
void Dio3SetDirection(unsigned char);
unsigned char Dio3GetDirection(void);
//---------------------------------------------------------------------------
int main (void)
{
        char SIO;

        SIO = Init_W627DHG();
        if (SIO == 0)
        {
                printf("Can not detect Winbond 83627DHG, program abort.\n");
                return(1);
        }

        Dio2Initial();
        Dio3Initial();

        //for GPIO30..37
        Dio3SetDirection(0x0F);         //GP30..33 = input, GP34..37=output
        printf("Current DIO direction = 0x%X\n", Dio3GetDirection());

        printf("Current DIO status = 0x%X\n", Dio3GetInput());

        printf("Set DIO output to high\n");
        Dio3SetOutput(0x0F);

        printf("Set DIO output to low\n");
        Dio3SetOutput(0x00);

        return 0;
}
//---------------------------------------------------------------------------
void Dio2Initial(void)
{
        unsigned char ucBuf;

        //switch GPIO multi-function pin
        ucBuf = Get_W627DHG_Reg(0x24);
        ucBuf &= 0xFE;
        Set_W627DHG_Reg(0x24, ucBuf);

        Set_W627DHG_LD(0x09);                                   //switch to logic device 9

        //enable the GP2 group
        ucBuf = Get_W627DHG_Reg(0x30);
        ucBuf |= 0x01;
        Set_W627DHG_Reg(0x30, ucBuf);
}
//---------------------------------------------------------------------------
void Dio2SetOutput(unsigned char)
{
        Set_W627DHG_LD(0x09);                                   //switch to logic device 9
        Set_W627DHG_Reg(0xE4, NewData);
```

```
}
//-----------------------------------------------------------------------
unsigned char Dio2GetInput(void)
{
	unsigned char result;

	Set_W627DHG_LD(0x09);                                   //switch to logic device 9
	result = Get_W627DHG_Reg(0xE4);
	return (result);
}
//-----------------------------------------------------------------------
void Dio2SetDirection(unsigned char)
{
	//NewData : 1 for input, 0 for output
	Set_W627DHG_LD(0x09);                                   //switch to logic device 9
	Set_W627DHG_Reg(0xE3, NewData);
}
//-----------------------------------------------------------------------
unsigned char Dio2GetDirection(void)
{
	unsigned char result;

	Set_W627DHG_LD(0x09);                                   //switch to logic device 9
	result = Get_W627DHG_Reg(0xE0);
	return (result);
}
//-----------------------------------------------------------------------
void Dio3Initial(void)
{
	unsigned char ucBuf;

	//switch GPIO multi-function pin
	ucBuf = Get_W627DHG_Reg(0x2C);
	ucBuf &= 0x1F;
	Set_W627DHG_Reg(0x2C, ucBuf);                           //clear

	Set_W627DHG_LD(0x09);                                   //switch to logic device 9

	//enable the GP3 group
	ucBuf = Get_W627DHG_Reg(0x30);
	ucBuf |= 0x02;
	Set_W627DHG_Reg(0x30, ucBuf);

	//input detect type
	Set_W627DHG_Reg(0xFE, 0xFF);
}
//-----------------------------------------------------------------------
void Dio3SetOutput(unsigned char NewData)
{
	Set_W627DHG_LD(0x09);                                   //switch to logic device 9
	Set_W627DHG_Reg(0xF1, NewData);
}
//-----------------------------------------------------------------------
unsigned char Dio3GetInput(void)
{
	unsigned char result;

	Set_W627DHG_LD(0x09);                                   //switch to logic device 9
	result = Get_W627DHG_Reg(0xF1);
	return (result);
}
//-----------------------------------------------------------------------
void Dio3SetDirection(unsigned char NewData)
{
	//NewData : 1 for input, 0 for output
	Set_W627DHG_LD(0x09);                                   //switch to logic device 9
	Set_W627DHG_Reg(0xF0, NewData);
}
//-----------------------------------------------------------------------
unsigned char Dio3GetDirection(void)
{
	unsigned char result;

	Set_W627DHG_LD(0x09);                                   //switch to logic device 9
	result = Get_W627DHG_Reg(0xF0);
	return (result);
}
//-----------------------------------------------------------------------
```

Filename：W627DHG.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 "W627DHG.H"
#include <dos.h>
//---------------------------------------------------------------------------
unsigned int W627DHG_BASE;
void Unlock_W627DHG (void);
void Lock_W627DHG (void);
//---------------------------------------------------------------------------
unsigned int Init_W627DHG(void)
{
        unsigned int result;
        unsigned char ucDid;

        W627DHG_BASE = 0x4E;
        result = W627DHG_BASE;

        ucDid = Get_W627DHG_Reg(0x20);
        if (ucDid == 0xA0)                                      //W83627DHG
        {       goto Init_Finish;       }
        else if (ucDid == 0xB0)                                 //W83627DHG-P
        {       goto Init_Finish;       }

        W627DHG_BASE = 0x2E;
        result = W627DHG_BASE;

        ucDid = Get_W627DHG_Reg(0x20);
        if (ucDid == 0xA0)                                      //W83627DHG
        {       goto Init_Finish;       }
        else if (ucDid == 0xB0)                                 //W83627DHG-P
        {       goto Init_Finish;       }

        W627DHG_BASE = 0x00;
        result = W627DHG_BASE;

Init_Finish:
        return (result);
}
//---------------------------------------------------------------------------
void Unlock_W627DHG (void)
{
        outportb(W627DHG_INDEX_PORT, W627DHG_UNLOCK);
        outportb(W627DHG_INDEX_PORT, W627DHG_UNLOCK);
}
//---------------------------------------------------------------------------
void Lock_W627DHG (void)
{
        outportb(W627DHG_INDEX_PORT, W627DHG_LOCK);
}
//---------------------------------------------------------------------------
void Set_W627DHG_LD( unsigned char LD)
{
        Unlock_W627DHG();
        outportb(W627DHG_INDEX_PORT, W627DHG_REG_LD);
        outportb(W627DHG_DATA_PORT, LD);
        Lock_W627DHG();
}
//---------------------------------------------------------------------------
void Set_W627DHG_Reg( unsigned char REG, unsigned char DATA)
{
        Unlock_W627DHG();
        outportb(W627DHG_INDEX_PORT, REG);
        outportb(W627DHG_DATA_PORT, DATA);
        Lock_W627DHG();
}
//---------------------------------------------------------------------------
unsigned char Get_W627DHG_Reg(unsigned char REG)
{
        unsigned char Result;
        Unlock_W627DHG();
        outportb(W627DHG_INDEX_PORT, REG);
        Result = inportb(W627DHG_DATA_PORT);
        Lock_W627DHG();
        return Result;
}
//---------------------------------------------------------------------------
```

Filename：W627DHG.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 __W627DHG_H
#define __W627DHG_H                     1
//---------------------------------------------------------------------------
#define    W627DHG_INDEX_PORT           (W627DHG_BASE)
#define    W627DHG_DATA_PORT            (W627DHG_BASE+1)
//---------------------------------------------------------------------------
#define    W627DHG_REG_LD               0x07
//---------------------------------------------------------------------------
#define W627DHG_UNLOCK                  0x87
#define    W627DHG_LOCK                 0xAA
//---------------------------------------------------------------------------
unsigned int Init_W627DHG(void);
void Set_W627DHG_LD( unsigned char);
void Set_W627DHG_Reg( unsigned char, unsigned char);
unsigned char Get_W627DHG_Reg( unsigned char);
//---------------------------------------------------------------------------
#endif      //__W627DHG_H
```

Chapter 16 Drivers Installation

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

Intel® Chipset Software Installation Utility
Intel® Graphics Driver Installation
LAN Drivers Installation
Intel® Management Engine Interface

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. (Before installed Intel® Chipset Software Installation Utility.

1. Insert the DVD that comes with the board. Click ***Intel*** and then ***Intel(R) Chipset Software Installation Utility.***

3. When the Welcome screen appears, click ***Next*** to continue.
4. Click ***Yes*** to accept the software license agreement and proceed with the installation process.
5. On the Readme Information screen, click ***Next*** to continue the installation.
6. When the Setup Progress screen appears, click ***Next*** to continue.

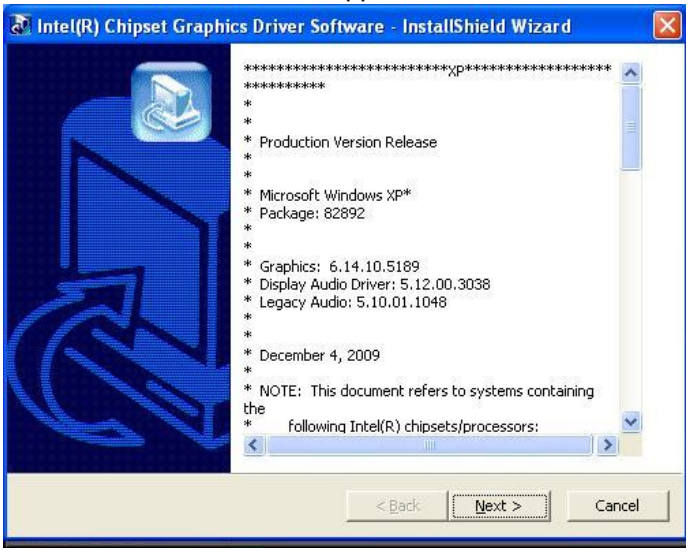
7. The Setup process is now complete. Click ***Finish*** then restart the computer and for changes to take effect.

Intel® Graphics Driver Installation

1. Insert the DVD that comes with the board. Click ***Intel -> Intel® Q57 Chipset Family Graphics Driver***.

2. When the InstallShield Wizard screen appears, click ***Next***.

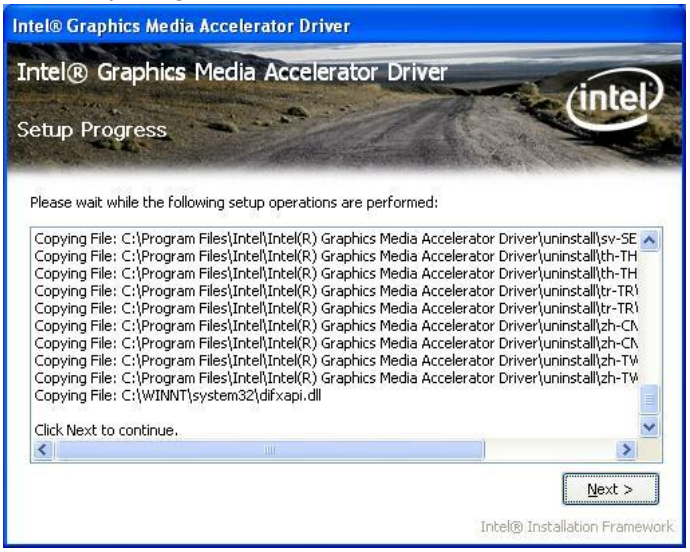
3. When the Welcome screen appears, click ***Next*** to continue.



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

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

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



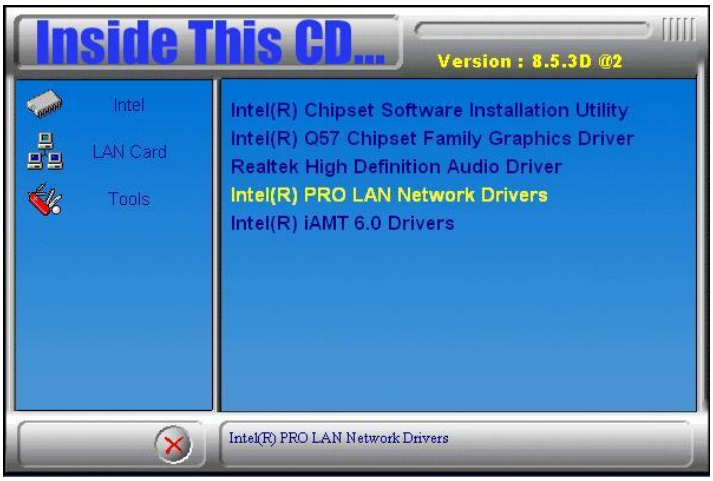
7. The Setup process is now complete. Click ***Finish*** to restart the computer and for changes to take effect.

LAN Drivers Installation

Follow the steps below to start installing the Intel® LAN drivers.

1. Insert the DVD that comes with the board. Click ***Intel*** and then ***Intel(R) PRO LAN Network Drivers***.

2. Click ***Intel(R) PRO LAN Network Drivers.***

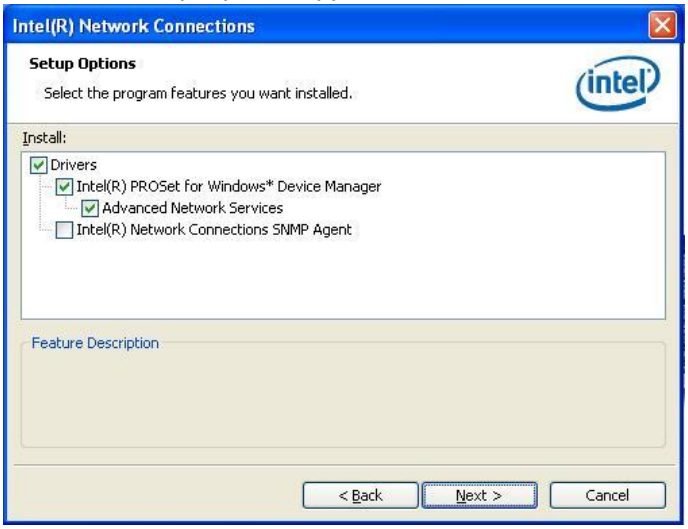


3. On the next screen, click ***Install Drivers*** to start the drivers installation.

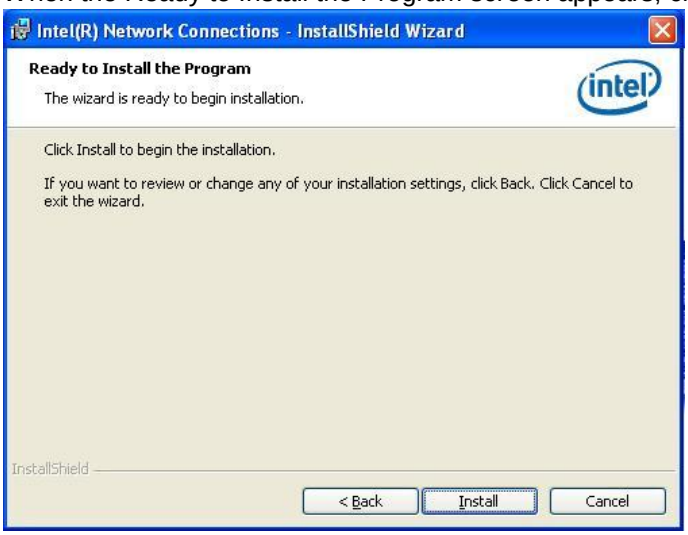
4. When the Welcome screen appears, click ***Next*** to continue.

5. In the License Agreement screen, click ***I accept the terms in license agreement*** and ***Next*** to accept the software license agreement and proceed with the installation process.

6. When the Setup Options appears, click ***Drivers*** as shown below and ***Next*** to continue.



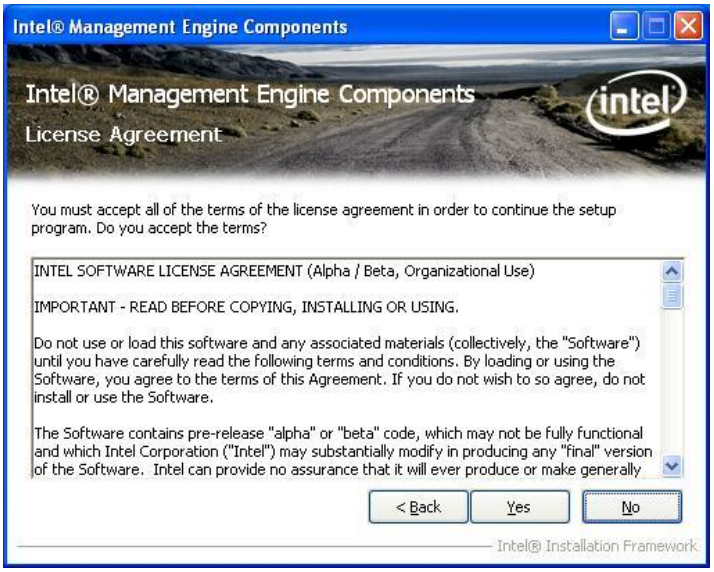
7. When the Ready to Install the Program screen appears, click ***Install*** to continue.



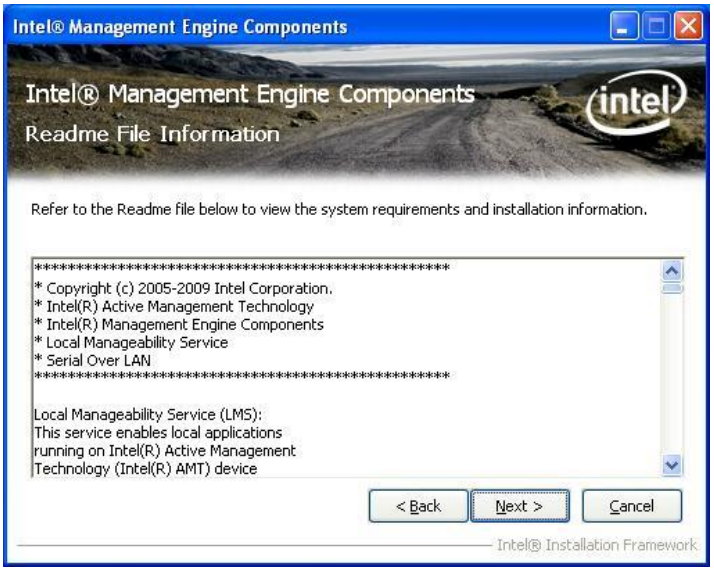
8. The Setup process is now complete (InstallShield Wizard Completed). Click ***Finish*** to restart the computer and for changes to take effect.

Intel® Management Engine Interface

1. Insert the drivers disc that comes with the motherboard. Click ***Intel*** and then ***Intel(R) AMT 6.0 Drivers***. When the welcome screen of the Intel® Management Engine Components appears, click ***Next*** to continue. On the next screen, click ***Next*** to agree to the license agreement.



2. On the next screen, the Readme File Information shows the system requirements and installation information, click ***Next***.



3. When the Setup Progress screen appears, click ***Next*** to continue. Then, click ***Finish*** when the setup progress has been successfully installed to restart the computer.

Appendix-A FWA8207 Series Configurations

The following lists the available SKUs of FWA8207 for different system requirement.

FWA8207

2.5” HDD x1, PCI-e add-on card rear expansion x1, front panel expansion card x1, 300W PSU

- MB966 x1
- IP327 x1: 1-to-1 Riser Card
- IP328 x1: PCI-e Adapter
- Single 2.5” HDD Bracket x1
- 4-pin Smart Fan x3
- 300W Single PSU

FWA8207 Optional Items

- IBP161: 4-port GLAN Card
- PS2G PS/2 Keyboard /Mouse Cable
- Dual 2.5” HDD Kit

FWA8207-2SLOT

3.5” HDD x1, PCI-e add-on card rear expansion x2, 300W PSU

- MB966 x1
- IP329 x1: 2-to-2 Riser Card
- Single 3.5” HDD Bracket x1
- SATA Cable x1
- 4-pin Smart Fan x3
- 300W Single PSU

FWA8207-2SLOT Optional Items

- PS2G PS/2 Keyboard /Mouse Cable
- Dual 2.5” HDD Kit

FWA8207-G

Supports Integrated Graphics CPUs, 3.5” HDD x1, PCI-e add-on card rear expansion x1, 300W PSU

- MB966 x1
- IP327 x1: 1-to-1 Riser Card
- Single 3.5” HDD Bracket x1
- SATA Cable x1
- 4-pin Smart Fan x3
- 300W Single PSU

FWA8207-G Optional Items

- PS2G PS/2 Keyboard /Mouse Cable
- VGA4 Cable
- Dual 2.5” HDD Kit

FWA8207-G-NB

No Bypass function MB966, Supports Integrated Graphics CPUs, 3.5” HDD x1, PCI-e add-on card rear expansion x1, 300W PSU

- **MB966-NB x1**
- **IP327 x1: 1-to-1 Riser Card**
- **Single 3.5” HDD Bracket x1**
- **SATA Cable x1**
- **4-pin Smart Fan x3**
- **300W Single PSU**

FWA8207-G-NB Optional Items

- **PS2G PS/2 Keyboard /Mouse Cable**
- **VGA4 Cable**
- **Dual 2.5” HDD Kit**