MI889

Intel® Pineview-M/ Pineview-D +ICH8M
Luna-Pier / Luna-Pier Refresh Platform
MiniITX Motherboard

USER'S MANUAL

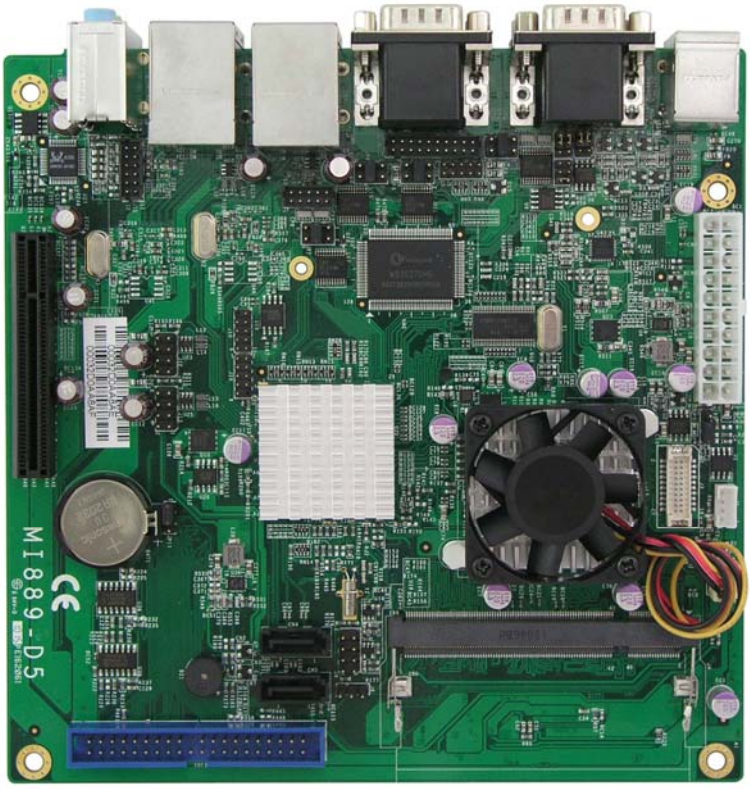
Version 1.0

Acknowledgments

Table of Contents

This page is intentionally left blank.

Introduction



MI889 MINI ITX MOTHERBOARD

MI889 EDGE CONNECTORS

Checklist

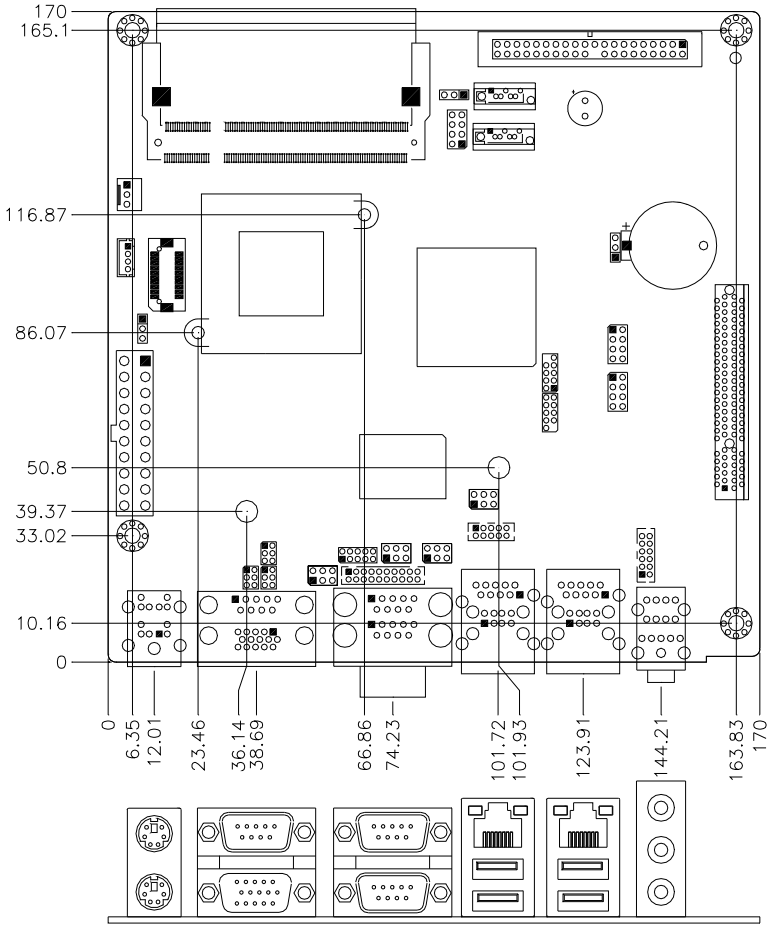
Your MI889 package should include the items listed below.

- The MI889 Intel® Atom Mini-ITX motherboard
- This User's Manual
- 1 CD containing chipset drivers and flash memory utility
- Cables (IDE cable, Serial ATA cable)

MI889 Specifications

Product Name	MI889-D5 (main item) / MI889
Form Factor	MiniITX
CPU Type	Intel® Pineview-M SC / Pineview-D DC Processor
CPU Speed	Intel® Atom™ Single Core N450 / 1.66GHz/ 512 KB L2 cache(5.5W) [MI889] Intel® Atom™ Dual Core D510 / 1.66GHz/ 1MB L2 cache(13W) [MI889-D5]
BIOS	AMI BIOS, support ACPI Function
Chipset	ICH8M: 31mm x 31mm, 676-pin T-PBGA (2.4W)
Memory	DDR2 667MHz SO-DIMM x 1 (w/o ECC), Max. 2GB , Single channel
VGA	Intel® Integrated Graphics Controller (GMA3150) Luna Pier supports DirectX 9 Graphic (200MHz) Luna Pier Refresh supports DirectX 9 Graphic (400MHz) OpenGL 1.4
LVDS	18-bit one channels LVDS interface w/DF13 socket x1
LAN	Realtek 8111DL x 2
USB	ICH8M built-in USB 2.0 host controller, support 8 ports
Serial ATA Ports	ICH8M built-in SATA controller, supports 2 ports
Parallel IDE	ICH8M built-in one channel Ultra DMA 33/66/100, for IDE x 1
Audio	Intel ICH8M built-in audio controller w/ Realtek ALC662 Codec Supports 5.1 CH audio (Line-out, Line-in & MIC)
LPC I/O	Winbond W83627UHG : COM1 (RS232/422/485), COM2~4 (RS232 only) Hardware monitor (2 thermal inputs, 4 voltage monitor inputs, VID0-4 & 1 x Fan Header)
Digital IO	4 in & 4 out
Keyboard/Mouse Connector	Yes
Expansion Slots	PCI-Express (x8) slot x1 [(x4) Link]
Edge Connector	PS/2 KBMS connector x 1 DB15 + DB9 Stack connector x1 for VGA+COM 1 Dual DB9 Stack connector x 1 for COM2,3 RJ45 + Dual USB stack connector x 1 for LAN 1 + USB 1, 2 RJ45 + Dual USB stack connector x 1 for LAN 1 + USB 3, 4 Audio 3-port connector x 1 (Line-out, Line-in, MIC)
Onboard Header/ Connector	8-pin header x2 for 4 USB ports DF13 connector x1 for LVDS DF11 10-pin connector x 1 for COM4 12-pin box header x1 for front audio 40-pin box header x1 for IDE 10-pin header x1 for Digital I/O 10-pin header x1 for LPC [debug purpose]
Watchdog Timer	Yes (256 segments, 0, 1, 2…255 sec/min)
Power Connector	ATX (20-pins)
Board Size	170mm x 170mm

Board Dimensions



Installations

This section provides information on how to use the jumpers and connectors on the MI889 in order to set up a workable system. The topics covered are:

Installing the Memory

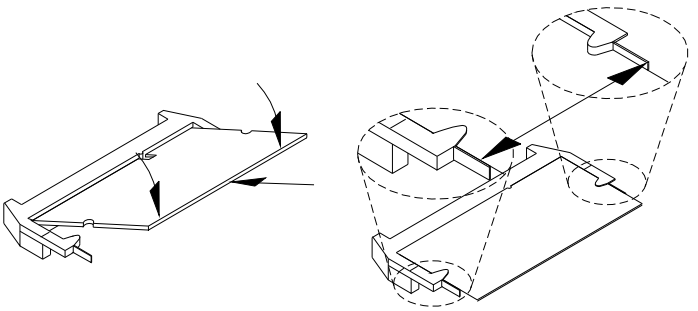
The MI889 board supports one DDR667 DDR2 memory.

Remarks:
N450 supports SO-DIMM x 1 (w/o ECC), Max. 1GB , Single channel
D510 supports SO-DIMM x 1 (w/o ECC), Max. 2GB , Single channel

Installing and Removing Memory Modules

To install the DDR2 modules, locate the memory slot on the board and perform the following steps:

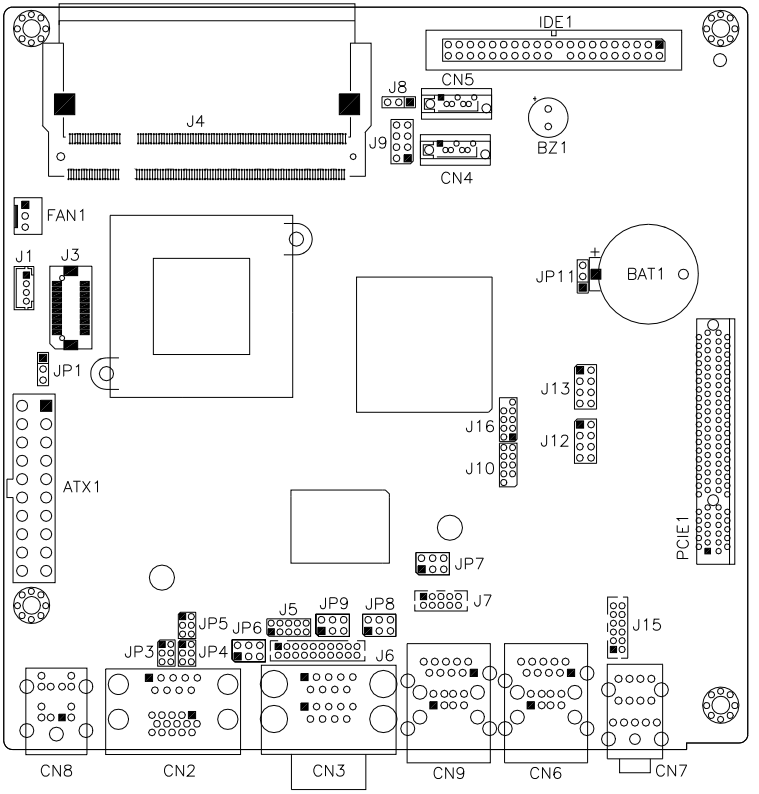
1. Hold the DDR2 module so that the key of the DDR2 module aligns with that on the memory slot. Insert the module into the socket at a slight angle (approximately 30 degrees). Note that the socket and module are both keyed, which means that the module can be installed only in one direction.
2. To seat 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 DDR2 module, press the clips with both hands.



Setting the Jumpers

Jumpers are used on MI889 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 MI889 and their respective functions.

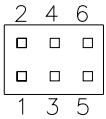
Jumper Locations on MI889



JP1: LCD Panel Power Selection

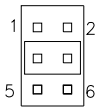
JP2	LCD Panel Power
	3.3V
	5V

JP3, JP4, JP5: RS232/422/485 (COM1) Selection

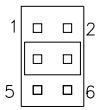


COM1 Function	RS-232	RS-422	RS-485
Jumper Setting (pin closed)	JP5: 1-2 JP3: 3-5 & 4-6 JP4: 3-5 & 4-6	JP5: 3-4 JP3: 1-3 & 2-4 JP4: 1-3 & 2-4	JP5: 5-6 JP3: 1-3 & 2-4 JP4: 1-3 & 2-4

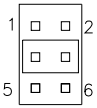
JP6: COM1 RS232 RI/+5V/+12V Power Setting

JP6	Setting	Function
	Pin 1-2 Short/Closed	+12V
	Pin 3-4 Short/Closed	RI
	Pin 5-6 Short/Closed	+5V

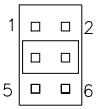
JP8: COM4 RS232 RI/+5V/+12V Power Setting

JP8	Setting	Function
	Pin 1-2 Short/Closed	+12V
	Pin 3-4 Short/Closed	RI
	Pin 5-6 Short/Closed	+5V

JP9: COM3 RS232 RI/+5V/+12V Power Setting

JP9	Setting	Function
	Pin 1-2 Short/Closed	+12V
	Pin 3-4 Short/Closed	RI
	Pin 5-6 Short/Closed	+5V

JP7: COM4 RS232 RI/+5V/+12V Power Setting

JP7	Setting	Function
	Pin 1-2 Short/Closed	+12V
	Pin 3-4 Short/Closed	RI
	Pin 5-6 Short/Closed	+5V

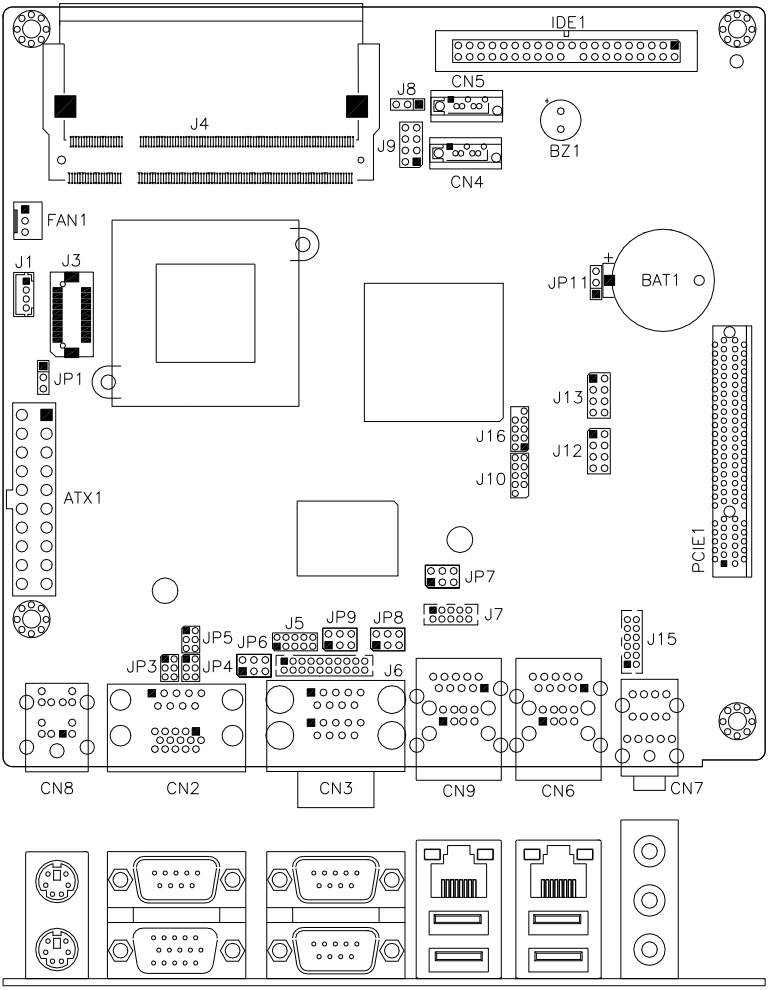
JP11: Clear CMOS Setting

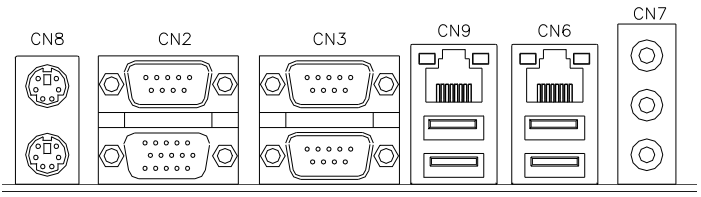
JP11	Setting
	Normal
	Clear CMOS

Connectors on MI889

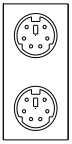
The connectors on MI889 allows you to connect external devices such as keyboard, floppy disk drives, hard disk drives, printers, etc. The following table lists the connectors on MI889 and their respective functions.

Connector Locations on MI889





CN8: PS/2 Keyboard and PS/2 Mouse Connectors

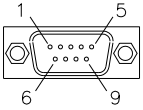


PS/2 Mouse

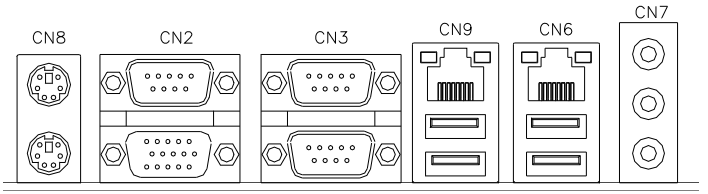
PS/2 Keyboard

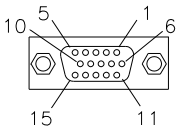
Signal Name	Keyboard	Mouse	Signal Name
Keyboard data	1	1	Mouse data
N.C.	2	2	N.C.
GND	3	3	GND
5V	4	4	5V
Keyboard clock	5	5	Mouse clock
N.C.	6	6	N.C.

CN2: COM1 RS232/RS422/RS485 and VGA Connector



Pin #	Signal Name		
	RS-232	R2-422	RS-485
1	DCD	TX-	DATA-
2	RX	TX+	DATA+
3	TX	RX+	NC
4	DTR	RX-	NC
5	Ground	Ground	Ground
6	DSR	NC	NC
7	RTS	NC	NC
8	CTS	NC	NC
9	RI	NC	NC
10	NC	NC	NC

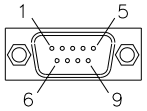




VGA

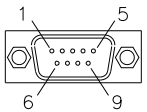
Signal Name	Pin #	Pin #	Signal Name
Red	1	2	Green
Blue	3	4	N.C.
GND	5	6	GND
GND	7	8	GND
N.C.	9	10	GND
N.C.	11	12	N.C.
HSYNC	13	14	VSYNC
NC	15		

CN3: COM2/RS232 and COM3/RS232 Connector



COM2/RS232

Signal Name	Pin #	Pin #	Signal Name
DCD	1	6	DSR
RXD	2	7	RTS
TXD	3	8	CTS
DTR	4	9	RI
GND	5	10	Not Used



COM3/RS232

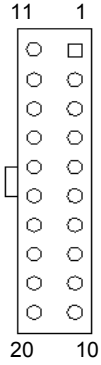
Signal Name	Pin #	Pin #	Signal Name
DCD	1	6	DSR
RXD	2	7	RTS
TXD	3	8	CTS
DTR	4	9	RI
GND	5	10	Not Used

CN6: 10/100/1000 RJ-45 and USB3/4 Ports

CN9:10/100/1000 RJ-45 and USB5/6 Ports

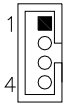
CN7: Line-in, Line-out & Microphone Connector

ATX1: ATX Power Supply Connector



Signal Name	Pin #	Pin #	Signal Name
3.3V	11	1	3.3V
-12V	12	2	3.3V
Ground	13	3	Ground
PS-ON	14	4	+5V
Ground	15	5	Ground
Ground	16	6	+5V
Ground	17	7	Ground
-5V	18	8	Power good
+5V	19	9	5VSB
+5V	20	10	+12V

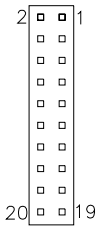
J1: LCD Backlight Connector



Pin #	Signal Name
1	+12V
2	Backlight Enable
3	Brightness Control
4	Ground

J3 : LVDS(18bit) Connectors

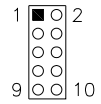
The LVDS connectors on board



Signal Name	Pin #	Pin #	Signal Name
TX0-	2	1	TX0+
Ground	4	3	Ground
TX1-	6	5	TX1+
5V/3.3V	8	7	Ground
NC	10	9	NC
TX2-	12	11	TX2+
Ground	14	13	Ground
TXC-	16	15	TXC+
5V/3.3V	18	17	ENABKL
+12V	20	19	+12V

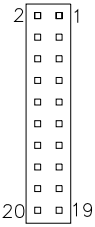
J4: DDR2 SO-DIMM

J5: Digital I/O



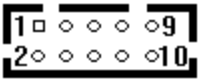
Signal Name	Pin	Pin	Signal Name
GND	1	2	VCC
OUT3	3	4	OUT1
OUT2	5	6	OUT0
IN3	7	8	IN1
IN2	9	10	IN0

J6: COM2/RS232, COM3/RS232 Serial Port



Signal Name	Pin #	Pin #	Signal Name
DSR	2	1	DCD
RTS	4	3	RXD
CTS	6	5	TXD
RI	8	7	DTR
NA	10	9	Ground
DSR	12	11	DCD
RTS	14	13	RXD
CTS	16	15	TXD
RI	18	17	DTR
NA	20	19	Ground

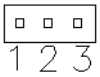
J7: COM4/RS232 Serial Port



Signal Name	Pin #	Pin #	Signal Name
DCD, Data carrier detect	1	6	DSR, Data set ready
RXD, Receive data	2	7	RTS, Request to send
TXD, Transmit data	3	8	CTS, Clear to send
DTR, Data terminal ready	4	9	RI, Ring indicator
GND, ground	5	10	Not Used

J8: Power LED

The power LED indicates the status of the main power switch.



Pin #	Signal Name
1	Power LED
2	No connect
3	Ground

J9: System Function Connector

ATX Power ON Switch: Pins 1 and 2
This 2-pin connector is an "ATX Power Supply On/Off Switch" on the system that connects to the power switch on the case. When pressed, the power switch will force the system to power on. When pressed again, it will force the system to power off.

Hard Disk Drive LED Connector: Pins 3 and 4
This connector connects to the hard drive activity LED on control panel. This LED will flash when the HDD is being accessed.

Pin #	Signal Name
4	HDD Active
3	5V

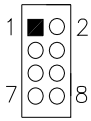
Reset Switch: Pins 5 and 6
The reset switch allows the user to reset the system without turning the main power switch off and then on again. Orientation is not required when making a connection to this header.

+5V and 5VSB Signals: Pins 7 and 8

Pin #	Signal Name
7	+5V
8	+5VSB

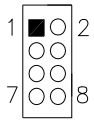
J10: SPI Flash Connector (factory use only)

J12: USB5/USB6 Connector



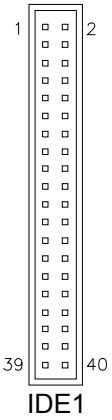
Signal Name	Pin	Pin	Signal Name
Vcc	1	2	Ground
D-	3	4	D+
D+	5	6	D-
Ground	7	8	Vcc

J13: USB7/USB8 Connector



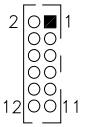
Signal Name	Pin	Pin	Signal Name
Vcc	1	2	Ground
D-	3	4	D+
D+	5	6	D-
Ground	7	8	Vcc

IDE1: 40-pin IDE Connectors



Signal Name	Pin #	Pin #	Signal Name
Reset IDE	1	2	Ground
Host data 7	3	4	Host data 8
Host data 6	5	6	Host data 9
Host data 5	7	8	Host data 10
Host data 4	9	10	Host data 11
Host data 3	11	12	Host data 12
Host data 2	13	14	Host data 13
Host data 1	15	16	Host data 14
Host data 0	17	18	Host data 15
Ground	19	20	Protect pin
DRQ0	21	22	Ground
Host IOW	23	24	Ground
Host IOR	25	26	Ground
IOCHRDY	27	28	Host ALE
DACK0	29	30	Ground
IRQ14	31	32	No connect
Address 1	33	34	No connect
Address 0	35	36	Address 2
Chip select 0	37	38	Chip select 1
Activity	39	40	Ground

J15: Audio Connector (DF11 Connector)



Signal Name	Pin #	Pin #	Signal Name
LINEOUT R	2	1	LINEOUT L
Ground	4	3	JD FRONT
LINEIN R	6	5	LINEIN L
Ground	8	7	JD LINEIN
MIC-In	10	9	MIC L
Ground	12	11	JD MIC1

J16: LPC Debug Purpose (factory use only)

PCIE1: PCIE X8 Slot (supports PCIE X4)

CN4, CN5: SATA Connectors

FAN1: CPU Fan Power Connector

This is a 3-pin header for system fans. The fan must be a 12V (500mA).

Pin #	Signal Name
1	Ground
2	+12V
3	Rotation detection

This page is intentionally left blank.

BIOS SETUP

This chapter describes the different settings available in the AMI (American Megatrends, Inc.) 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 adds virus and 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:

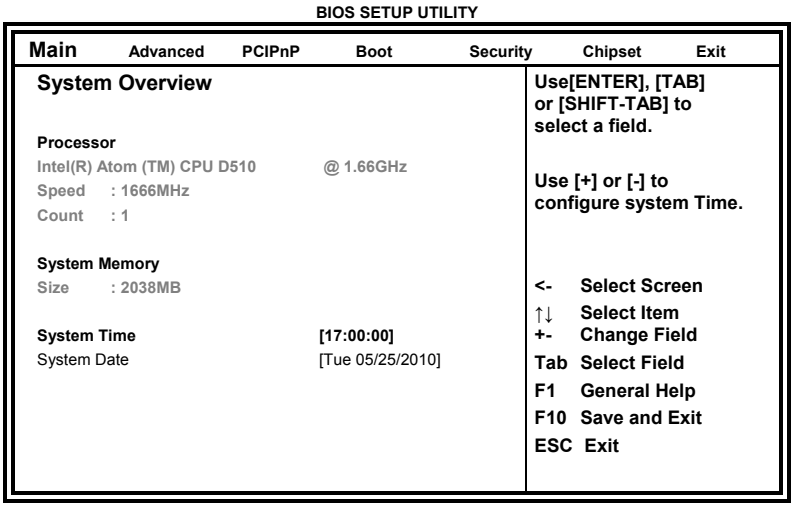
```
Press  <DEL>  to  Enter  Setup
```

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.

Main BIOS Setup

This setup allows you to record some basic hardware configurations in your computer system and set the system clock.

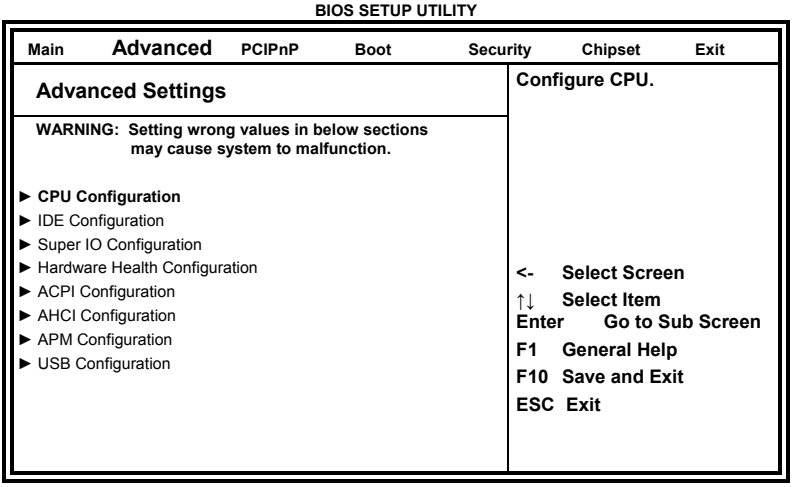


***Note**: If the system cannot boot after making and saving system changes with Setup, the AMI BIOS supports an override to the CMOS settings that resets your system to its default.*

***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

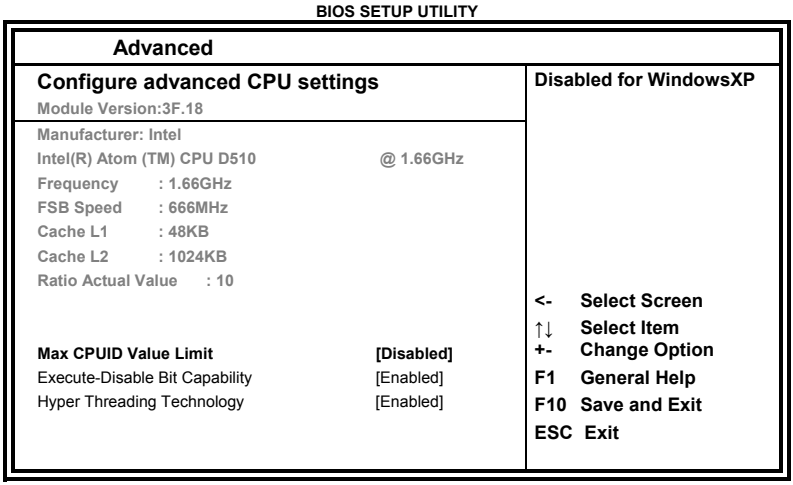
This section allows you to configure and improve your system and allows you to set up some system features according to your preference.



The Advanced BIOS Settings contains the following sections:

► CPU Configurations
► IDE Configuration
► Super IO Configuration
► Hardware Health Configuration
► ACPI Configuration
► AHCI Configuration
► APM Configuration
► USB Configuration

The fields in each section 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. If unsure, please contact technical support of your supplier.



The CPU Configuration menu shows the following CPU details: Manufacturer: the name of the CPU manufacturer
Brand String: the brand name of the CPU being used
Frequency: the CPU processing speed
FSB Speed: the FSB speed
Cache L1: the CPU L1 cache size
Cache L2: the CPU L2 cache

Max CPUID Value Limit

Disabled for WindowsXP.

Execute-Disable Bit Capability

XD can prevent certain classes of malicious buffer overflow attacks when combined with a supporting OS.

Hyper Threading Technology

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.

BIOS SETUP UTILITY

Advanced		
IDE Configuration		**Options**
ATA/IDE Configuration	**[Enhanced]**	**Disabled** **Compatible** **Enhanced**
Configure SATA as	[IDE]	
► Primary IDE Master	: [Not Detected]	
► Primary IDE Slave	: [Not Detected]	
► Secondary IDE Master	: [Not Detected]	
► Secondary IDE Slave	: [Not Detected]	**<- Select Screen**
► Third IDE Master	: [Not Detected]	**↑↓ Select Item**
► Third IDE Slave	: [Not Detected]	**+- Change Option**
► Fourth IDE Master	: [Not Detected]	**F1 General Help**
► Fourth IDE Slave	: [Not Detected]	**F10 Save and Exit**
Hard Disk Write Protect	[Disabled]	**ESC Exit**
IDE Detect Time Out (Sec)	[35]	
ATA(PI) 80Pin Cable Detection	[Host & Device]	

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

ATA/IDE Configuration

(1) Disabled.
(2) Compatible.
(3) Enhanced

Configure SATA as

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

BIOS SETUP UTILITY

Advanced		
Configure Win627UHG Super IO Chipset		**Allows BIOS to Select Serial Port Base Addresses**
Serial Port1 Address	**[3F8/IRQ4]**	
Serial Port2 Address	[2F8/IRQ3]	
Serial Port3 Address	[3E8]	
Serial PortC IRQ	[IRQ11]	**<- Select Screen**
Serial Port4 Address	[2E8]	**↑↓ Select Item**
Serial PortD IRQ	[IRQ10]	**+- Change Option**
Restore on AC Power Loss	[Power Off]	**F1 General Help**
		F10 Save and Exit
		ESC Exit

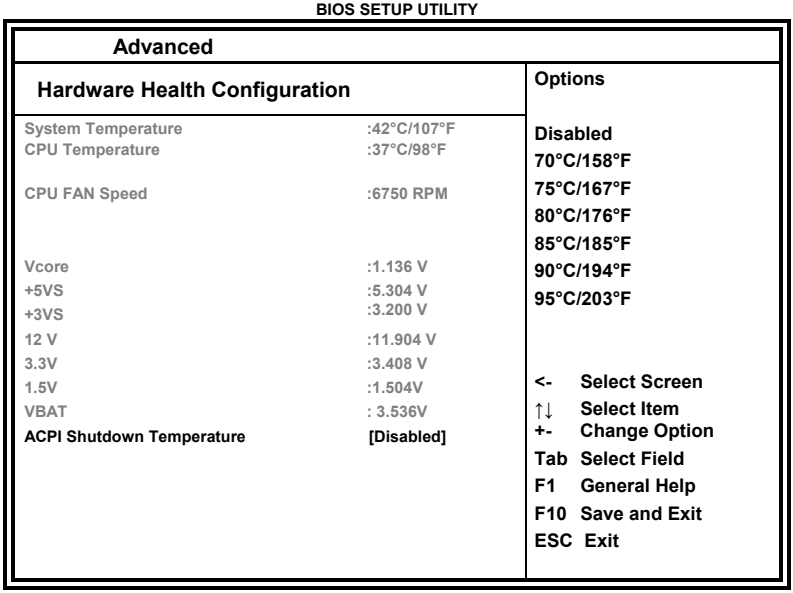
Onboard Serial Port

The default values are:

Serial Port 1: 3F8/IRQ4
Serial Port 2: 2F8/IRQ3
Serial Port 3: 3E8/IRQ11
Serial Port 4: 2E8/IRQ10

Restore on AC Power Loss

This field sets the system power status whether *Power On or Power Off* when power returns to the system from a power failure situation.



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

ACPI Shutdown Temperature

The system will shut down automatically under OS with ACPI mode, when the CPU temperature reaches the configured temperature.

BIOS SETUP UTILITY

Advanced	
ACPI Settings	**General ACPI Configuration settings**
►**General ACPI Configuration** ►Advanced ACPI Configuration ►Chipset ACPI Configuration	**<- Select Screen** **↑↓ Select Item** **Enter Go to Sub Screen** **F1 General Help** **F10 Save and Exit** **ESC Exit**

BIOS SETUP UTILITY

Advanced		
General ACPI Configuration		**Select the ACPI state used for System Suspend.**
Suspend mode	**[S1 (POS)]**	**<- Select Screen** **↑↓ Select Item** **+- Change Option** **F1 General Help** **F10 Save and Exit** **ESC Exit**

Suspend Mode

The options of this field are *S1*, *S3* and *Auto*.

BIOS SETUP UTILITY

Advanced		
Advance ACPI Configuration		**Enable RSDP pointers to 64-bit Fixed System Description Tables. Different ACPI version Has some addition**
ACPI Version Features	**[ACPI v1.0]**	**<- Select Screen** **↑↓ Select Item** **+- Change Option** **F1 General Help** **F10 Save and Exit** **ESC Exit**
ACPI APIC support	[Enabled]	

BIOS SETUP UTILITY

Advanced

South Bridge ACPI Configuration

Energy Lake Feature	**[Disabled]**
APIC ACPI SCI IRQ	[Disabled]

Options

Enabled
Disabled

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

BIOS SETUP UTILITY

Advanced

AHCI Settings

AHCI Port0	**[Not Detected]**
AHCI Port1	[Not Detected]
AHCI Port2	[Not Detected]

While entering setup, BIOS auto detect the presence of IDE device. This displays the status of auto detection of IDE devices.

<- Select Screen
↑↓ Select Item
Enter Go to Sub Screen
F1 General Help
F10 Save and Exit
ESC Exit

BIOS SETUP UTILITY

Advanced

APM Configuration

Power Management/APM	**[Enabled]**
Power Button Mode	[On/Off]
Resume On Ring	Disabled
Resume On PME#	Disabled
Resume On RTC Alarm	Disabled

Enable or disable APM.

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

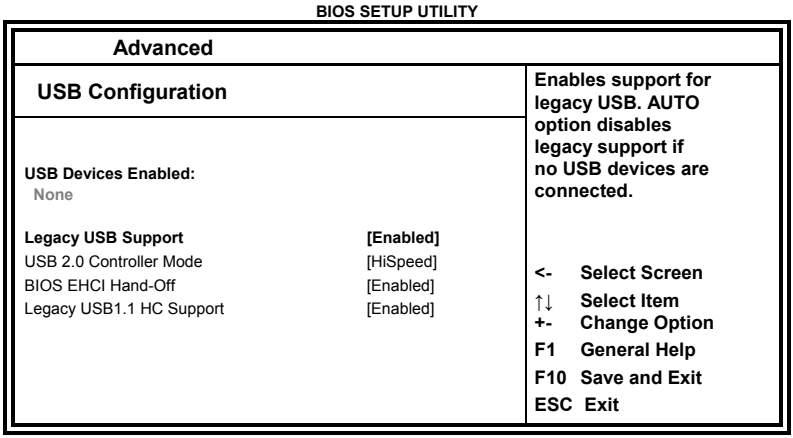
Power Management/APM
By default, this field is set to *Enabled.*

Power Button Mode
Go into On/Off, or Suspend when power button is pressed.

Resume on Ring
This option is used to enable activity on the RI (ring in) modem line to wake up the system from a suspend or standby state. That is, the system will be awakened by an incoming call on a modem.

Resume on PME#
This option is used enable activity on the PCI PME (power management event) controller to wake up the system from a suspend or standby state

Resume On RTC Alarm
This option is used to specify the time the system should be awakened from a suspended state



The USB Configuration menu is used to read USB configuration information and configure the USB settings.

Legacy USB Support

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

USB 2.0 Controller Mode

Configures the USB 2.0 controller in HiSpeed (480Mbps) or FullSpeed (12Mbps).This option is enabled by HiSpeed.

BIOS 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.

Legacy USB1.1 HC Support

Support USB1.1 HC.

PCIPnP Settings

This option configures the PCI/PnP settings.

BIOS SETUP UTILITY

Main	Advanced	PCIPnP	Boot	Security	Chipset	Exit

Advanced PCI/PnP Settings

WARNING: Setting wrong values in below sections may cause system to malfunction.

Plug & Play O/S	**[No]**
Allocate IRQ to PCI VGA	[Yes]
IRQ3	[Available]
IRQ4	[Available]
IRQ5	[Available]
IRQ7	[Available]
IRQ9	[Available]
IRQ10	[Available]
IRQ11	[Available]
IRQ14	[Available]
IRQ15	[Available]
DMA Channel 0	[Available]
DMA Channel 1	[Available]
DMA Channel 3	[Available]
DMA Channel 5	[Available]
DMA Channel 6	[Available]
DMA Channel 7	[Available]
Reserved Memory Size	[Disabled]

NO: lets the BIOS Configure all the Devices in the system. YES: lets the operating system configure Plug and Play (PnP) devices not required for boot if your system has a Plug and Play operating system.

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

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.

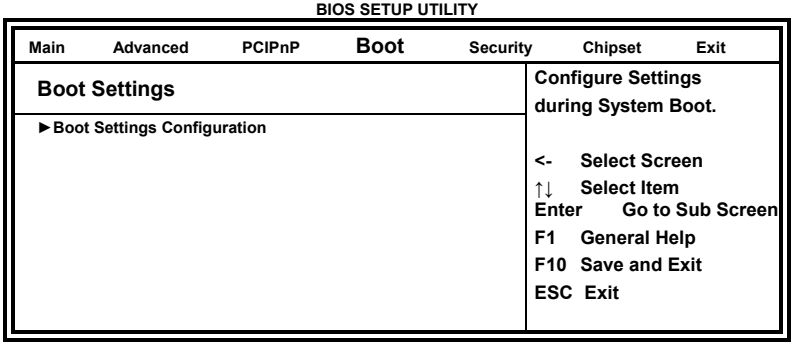
Allocate IRQ to PCI VGA

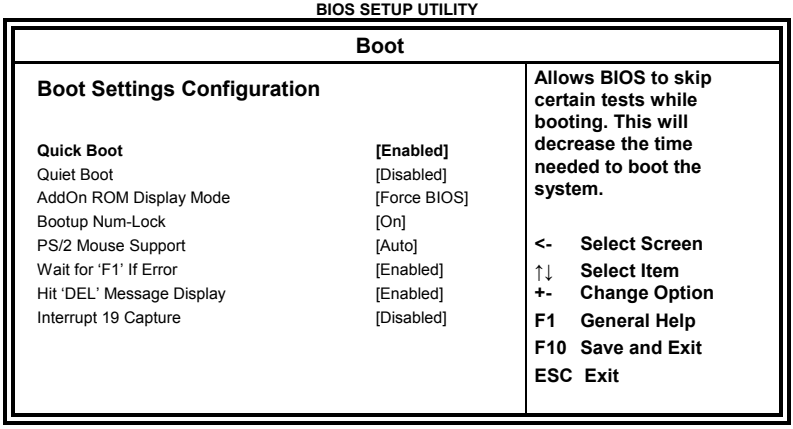
This assigns IRQ to PCI VGA card if card requests IRQ or doesn't assign IRQ to PCI VGA card even if card requests an IRQ.

IRQ#

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

Boot Settings





Quick Boot

This allows BIOS to skip certain tests while booting. This will decrease the time needed to boot the system.

Quite Boot

When disabled, this displays normal POST messages. When enabled, this displays OEM Logo instead of POST messages.

AddOn ROM Display Mode

This allows user to force BIOS/Option ROM of add-on cards to be displayed during quiet boot.

Bootup Num-Lock

This select the power-on state for numlock.

PS/2 Mouse Support
This select support for PS/2 mouse.

Wait for 'F1' If Error
When set to Enabled, the system waits for the F1 key to be pressed when error occurs. This allows option ROM to trap interrupt 19.

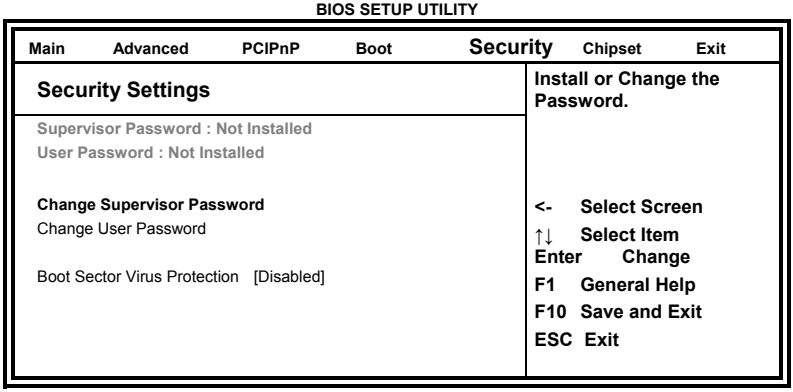
Hit <DEL> Message Display
This displays "Press <DEL> to run Setup" in POST.

Interrupt 19 Capture
This allows option ROMs to trap interrupt 19.

Security Settings

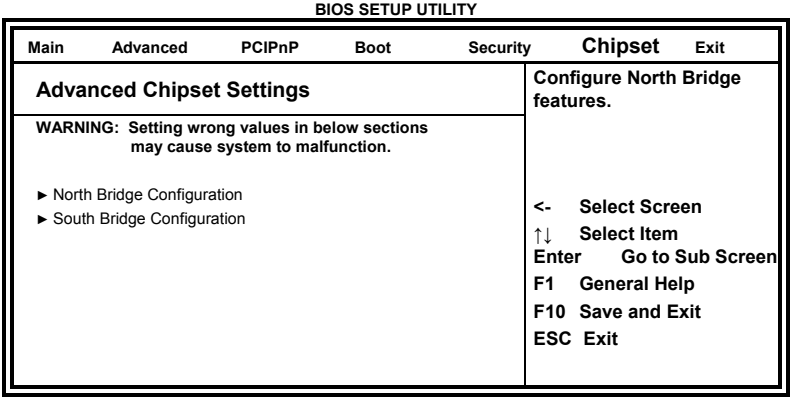
This setting comes with two options set the system password. Supervisor Password sets a password that will be used to protect the system and Setup utility. User Password sets a password that will be used exclusively on the system. To specify a password, highlight the type you want and press <Enter>. 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.

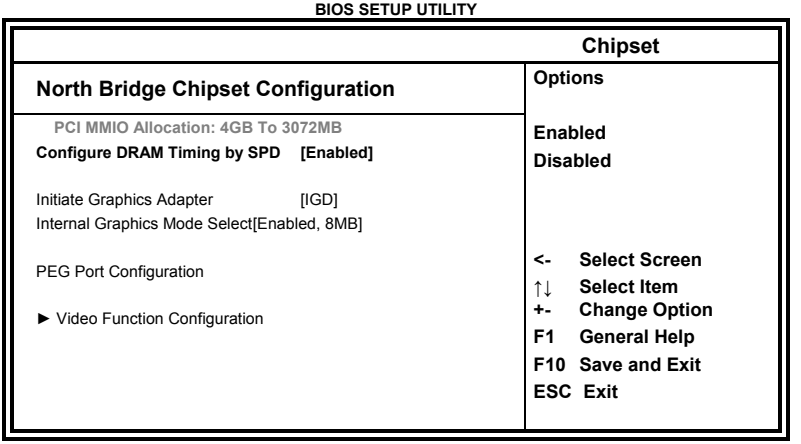
To disable a password, just press the <Enter> key when you are prompted to enter the password. A message will confirm the password to be disabled. Once the password is disabled, the system will boot and you can enter Setup freely.



Advanced Chipset Settings

This setting configures the north bridge and south bridge settings. WARNING! Setting the wrong values may cause the system to malfunction.



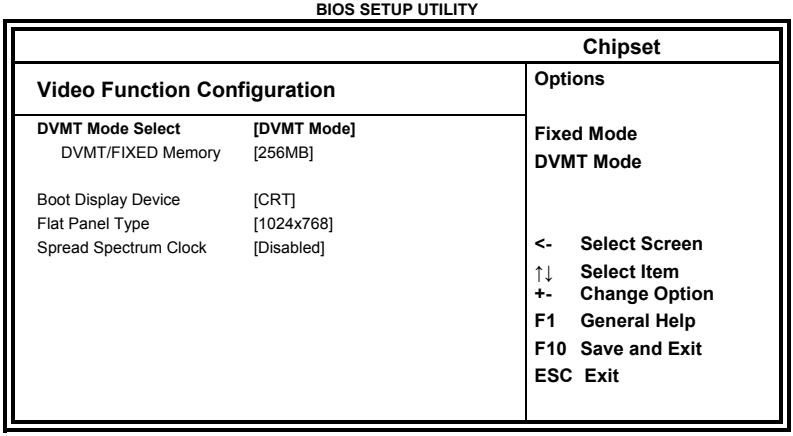


Configure DRAM Timing by SPD

When this item is enabled, the DRAM timing parameters are set according to the DRAM SPD (Serial Presence Detect). When disabled, you can manually set the DRAM timing parameters through the DRAM sub-items.

Initiate Graphic Adapter
Select which graphics controller to use as the primary boot device. This option, by default, is set to IGD.

Internal Graphics Mode Select
Use the feature to set the amount of system memory to be used by the Internal. graphics device. expansion cards that require a specified area of memory to work properly.



DVMT Mode Select
Select the control mode of memory built-in graphics capabilities. This option, by default, is set to DVMT Mode.

DVMT/FIXED Memory
Sets the maximum memory size assigned to the integrated graphics capabilities. This option, by default, is set to 256MB.

Boot Display Device
This option is used to select the display device used by the system when it boots.

Flat Panel Type
This option is used to select the type of flat panel connected to the system. Options include: 640x480 / 800x600 / 1024x768 / 800x480 / 1280x768 / 1280x800 / 1280x600.

Spread Spectrum Clock
By default, this field is set to *Disabled*.

BIOS SETUP UTILITY

Main	Advanced	PCIPnP	Boot	Security	Chipset	Exit

South Bridge Chipset Configuration

USB Function	**[8 USB Ports]**
USB 2.0 Controller	[Enabled]
HAD Controller	[Enabled]
SMBUS Controller	[Enabled]
Clock-Gen Spectrum	[Disabled]
PCIE Ports Configuration	
PCIE Port 0	[Auto]
PCIE Port 1	[Auto]
PCIE Port 2	[Auto]
PCIE Port 3	[Auto]
PCIE Port 4	[Auto]
PCIE Port 5	[Auto]
PCIE High Priority Port	[Disabled]
PCIE Port 0 IOxAPIC Enable	[Disabled]
PCIE Port 1 IOxAPIC Enable	[Disabled]
PCIE Port 2 IOxAPIC Enable	[Disabled]
PCIE Port 3 IOxAPIC Enable	[Disabled]
PCIE Port 4 IOxAPIC Enable	[Disabled]
PCIE Port 5 IOxAPIC Enable	[Disabled]
Enable Onboard PCI option ROM	[Disabled]

Options

Disabled
2 USB Ports
4 USB Ports
6 USB Ports
8 USB Ports
10 USB Ports

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

USB Function
This option enables the number of USB ports desired or disables the USB function.

USB 2.0 Controller
This option is disabled by default.

HDA Controller
This option is used to enable the Southbridge high definition audio controller.

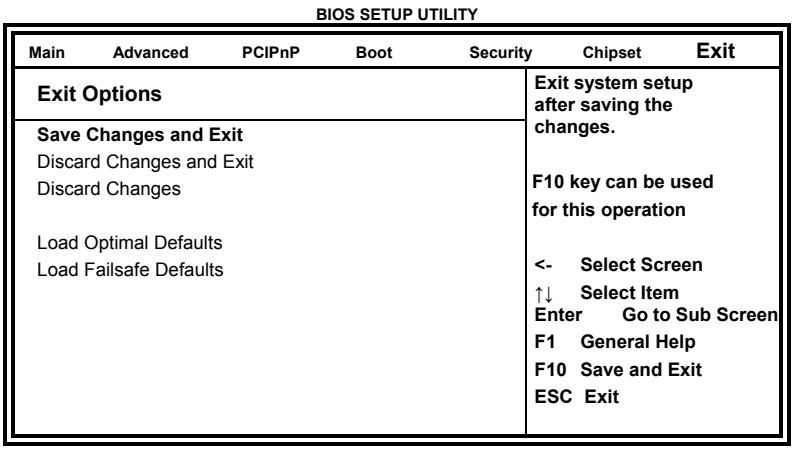
SMBUS Controller
This option is enabled by default.

Clock-Gen Spectrum
This option is disabled by default.

Enable Onboard PCI option ROM
This option is disabled by default.

Exit Setup

The exit setup has the following settings which are:



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.

This page is intentionally left blank.

Drivers Installation

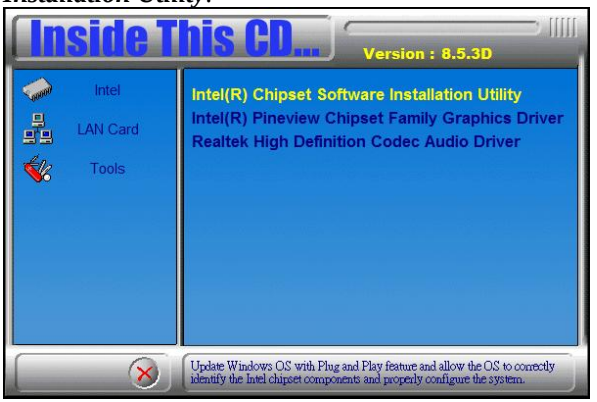
This section describes the installation procedures for software and drivers under the Windows XP, Windows Vista and Windows 7. The software and drivers are included with the motherboard. If you find the items missing, please contact the vendor where you made the purchase. The contents of this section include the following:

IMPORTANT NOTE:
After installing your Windows operating system (Windows XP/ Vista/ 7), 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 under Windows XP/Vista/7.

1. Insert the drivers DVD into the DVD drive. Click ***Intel*** and then ***Intel(R) Pineview Chipset Drivers***. Click ***Intel(R) Chipset Software Installation Utility***.



2. When the welcome screen to the Intel(R) Chipset Software Installation Utility appears, click ***Next*** to continue.

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

4. On the Readme Information screen, click ***Next*** to continue.When the Setup Progress screen appears, click ***Next*** to continue.

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

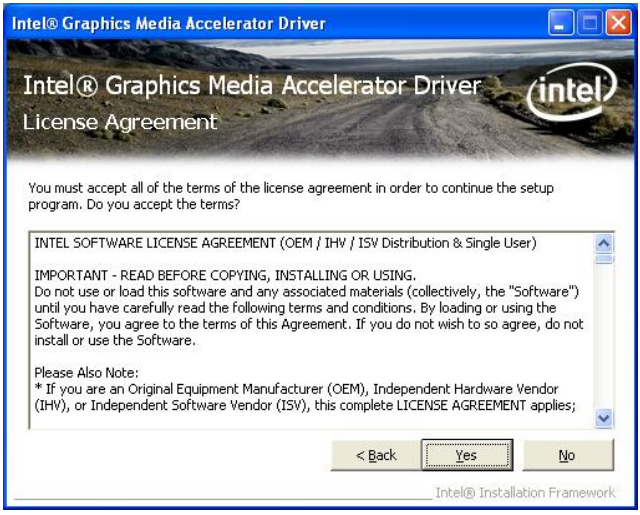
Intel Pineview Chipset Family Graphics Driver Installation

To install the VGA drivers, follow the steps below to proceed with the installation.

1. Insert the drivers DVD into the DVD drive. Click ***Intel*** and then ***Intel(R) Pineview Chipset Drivers***. Click ***Intel(R) Pineview Chipset Family Graphics Driver***.

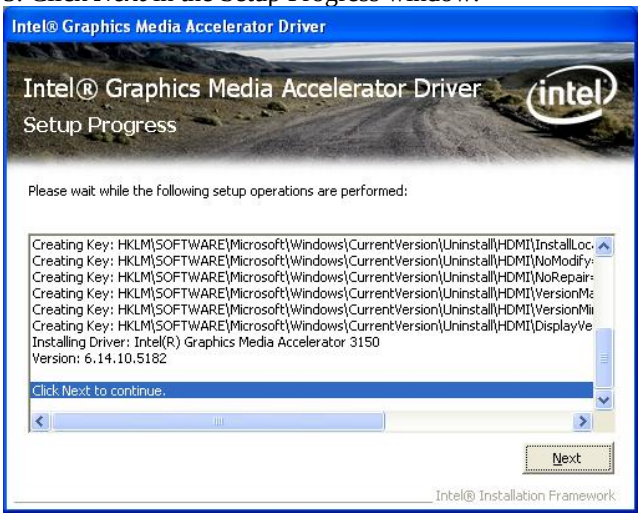
2. When the welcome screen of the Intel(R) Graphics Media Accelerator Driver appears, click ***Next*** to continue.

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



4. Click ***Next*** in the Readme File Information window.

5. Click ***Next*** in the Setup Progress window.



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

Realtek High Definition Codec Audio Driver Installation

Follow the steps below to install the Realtek HD Codec Audio Drivers.

1. Insert the drivers DVD into the DVD drive. Click ***Intel*** and then ***Intel(R) Pineview Chipset Drivers***. Click ***Realtek High Definition Codec Audio Driver.***

2.When the welcome screen to InstallShield Wizard for ***Realtek High Definition Audio Driver*** appears, click ***Next*** to start the installation.

3.When the InstallShieldWizard has finished performing maintenance operations on Realtek High Definition Codec Audio Audio Driver, click ***Finish*** to restart the computer.

Realtek RTL8111DL LAN Drivers Installation

Follow the steps below to install Realtek RTL8111DL LAN Drivers.

1. Insert the drivers DVD into the DVD drive. Click ***LAN Card*** and then ***Realtek LAN Controller Drivers***. Click ***Realtek RTL8111DL LAN Drivers***.

2. In the welcome screen of the InstallShield Wizard for REALTEK GbE & FE Ethernet PCI-E NIC Driver, click ***Next***.

3. In the InstallShield Wizard screen, click ***Install*** to begin the installation.

4. InstallShield Wizard completed. Click ***Finish*** to exit the Wizard.

This page is intentionally left blank.

Appendix

A. I/O Port Address Map

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

Address	Device Description
0000h-0CF7h	PCI bus
0000h-0CF7h	Direct memory access controller
0010h-001Fh	Motherboard resources
0020h-0021h	Programmable interrupt controller
0022h-003Fh	Motherboard resources
0040h-0043h	System timer
0044h-005Fh	Motherboard resources
0060h-0060h	Motherboard resources
0061h-0061h	System speaker
0062h-0063h	Motherboard resources
0064h-0064h	Motherboard resources
0065h-006Fh	Motherboard resources
0070h-0071h	System CMOS/real time clock
0072h-007Fh	Motherboard resources
0080h-0080h	Motherboard resources
0081h-0083h	Direct memory access controller
0084h-0086h	Motherboard resources
0087h-0087h	Direct memory access controller
0088h-0088h	Motherboard resources
0089h-008Bh	Direct memory access controller
008Ch-008Eh	Motherboard resources
008Fh-008Fh	Direct memory access controller
0090h-009Fh	Motherboard resources
00A0h-00A1h	Programmable interrupt controller
00A2h-00BFh	Motherboard resources
00C0h-00DFh	Direct memory access controller
00E0h-00EFh	Motherboard resources
00F0h-00FFh	Numeric data processor
01F0h-01F7h	Primary IDE Channel
0274h-0277h	ISAPNP Read Data Port
0279h-0279h	ISAPNP Read Data Port

Address	Device Description
02E8h-02EFh	Communications Port (COM4)
02F8h-02FFh	Communications Port (COM2)
03B0h-03BBh	Intel(R) Graphics Media Accelerator 3150
03C0h-03DFh	Intel(R) Graphics Media Accelerator 3150
03E8h-03EFh	Communications Port (COM3)
03F6h-03F6h	Primary IDE Channel
03F8h-03FFh	Communications Port (COM1)
0400h-041Fh	Intel(R) ICH8 Family SMBus Controller - 283E
04D0h-04D1h	Motherboard resources
0500h-053Fh	Motherboard resources
0800h-087Fh	Motherboard resources
0A00h-0A0Fh	Motherboard resources
0A10h-0A1Fh	Motherboard resources
0A79h-0A79h	ISAPNP Read Data Port
0D00h-FFFFh	PCI bus
B480h-B48Fh	Intel(R) ICH8M 3 port Serial ATA Storage Controller - 2828
B800h-B80Fh	Intel(R) ICH8M 3 port Serial ATA Storage Controller - 2828
B880h-B883h	Intel(R) ICH8M 3 port Serial ATA Storage Controller - 2828
BC00h-BC07h	Intel(R) ICH8M 3 port Serial ATA Storage Controller - 2828
C000h-C003h	Intel(R) ICH8M 3 port Serial ATA Storage Controller - 2828
C080h-C087h	Intel(R) ICH8M 3 port Serial ATA Storage Controller - 2828
C400h-C41Fh	Standard Universal PCI to USB Host Controller
C480h-C49Fh	Intel(R) ICH8 Family USB Universal Host Controller - 2832
C800h-C81Fh	Intel(R) ICH8 Family USB Universal Host Controller - 2831
C880h-C89Fh	Intel(R) ICH8 Family USB Universal Host Controller - 2830

Address	Device Description
CC00h-CC07h	Intel(R) Graphics Media Accelerator 3150
D000h-DFFFh	Intel(R) ICH8 Family PCI Express Root Port 5 - 2847
D800h-D8FFh	Realtek PCIe GBE Family Controller
E000h-EFFFh	Intel(R) ICH8 Family PCI Express Root Port 6 - 2849
E800h-E8FFh	Realtek PCIe GBE Family Controller #2
FFA0h-FFAFh	Intel(R) ICH8M Ultra ATA Storage Controllers - 2850

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
IRQ0	System timer
IRQ1	PS/2 Keyboard
IRQ3	Communications Port (COM2)
IRQ4	Communications Port (COM1)
IRQ5	Intel(R) ICH8 Family SMBus Controller - 283E
IRQ8	System CMOS/real time clock
IRQ9	Microsoft ACPI-Compliant System
IRQ10	Communications Port (COM4)
IRQ11	Communications Port (COM3)
IRQ12	PS/2 Mouse
IRQ13	Numeric data processor
IRQ14	Primary IDE Channel
IRQ16	Intel(R) Graphics Media Accelerator 3150
IRQ16	Realtek PCIe GBE Family Controller
IRQ16	Standard Universal PCI to USB Host Controller
IRQ17	Realtek PCIe GBE Family Controller #2
IRQ18	Intel(R) ICH8 Family USB Universal Host Controller - 2832
IRQ18	Intel(R) ICH8M 3 port Serial ATA Storage Controller - 2828
IRQ19	Intel(R) ICH8 Family USB Universal Host Controller - 2831
IRQ21	Microsoft UAA Bus Driver for High Definition Audio
IRQ22	Intel(R) ICH8 Family PCI Express Root Port 1 - 283F
IRQ22	Intel(R) ICH8 Family PCI Express Root Port 5 - 2847
IRQ23	Intel(R) ICH8 Family PCI Express Root Port 6 - 2849
IRQ23	Intel(R) ICH8 Family USB Universal Host Controller - 2830
IRQ23	Intel(R) ICH8 Family USB2 Enhanced Host Controller - 2836

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);.....
        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); .....
        Set_W627UHG_Reg(0xF6, 0x00);.....//clear watchdog timer
        Set_W627UHG_Reg(0x30, 0x00);.....
}
//-----------------------------------------------------------------------
```

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);
}
//---------------------------------------------------------------------------
```