IB815

Intel® Atom™ D525 + ICH8M
5.25-inch SBC

USER'S MANUAL

Version 1.0

Acknowledgments

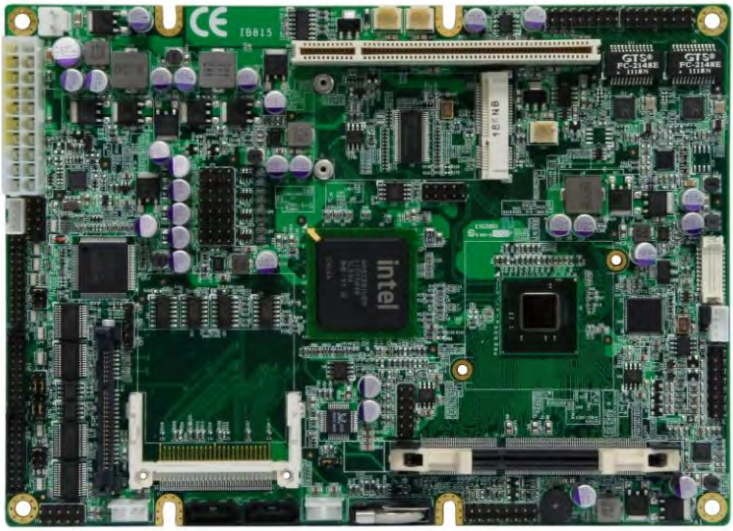
Table of Contents

This page is intentionally left blank.

Introduction

The IB815F 5.25-inch Disk-Size SBC comes with the Intel Atom D525 processor running 1.8GHz. The board is based on the Intel 82810HBM chipset and supports one DDRIII SO-DIMM memory socket for up to 4GB of system memory.

The SBC has useful interfaces for VGA CRT, 24-bit LVDS, dual Realtek 8111E Gigabit Ethernet, printer port, eight USB 2.0 ports, two SATA II ports, CF type II socket and a CFAst 90-degree vertical socket. Additional expansion is available with a PCI and a MiniPCIe slot. The board's dimension is 203mm x 146mm and features DC-in 12V and ATX power input which only one of the two can be used at one time.



IB815 5.25-inch Disk-Size SBC

Checklist

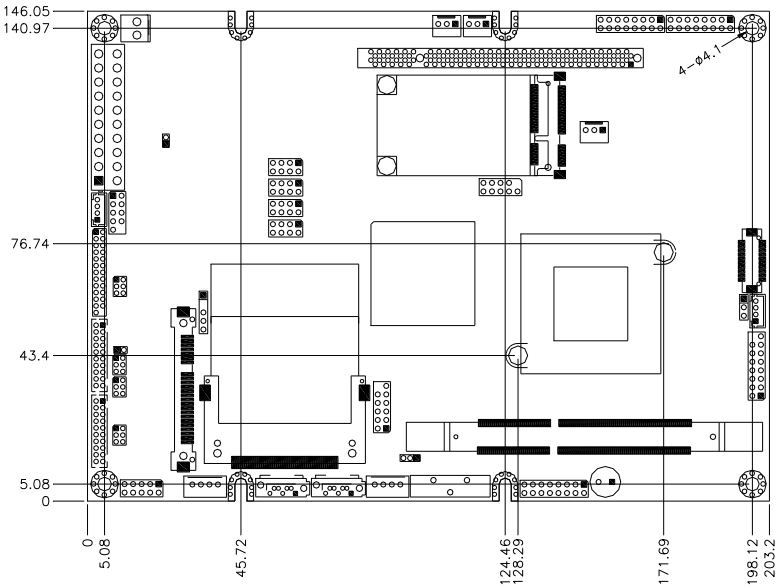
Your IB815 package should include the items listed below.

- The IB815 Intel® Atom Mini-ITX motherboard
- This User's Manual
- 1 CD containing chipset drivers and flash memory utility
- Cables (IDE cable, Serial ATA cable)

IB815 Specifications

Product Name	IB815F
Form Factor	5.25" Disk-sized SBC
CPU Type	Intel® Atom™ D525 processor (45nm Technology) 22mm x 22mm, Micro-FCBGA8 (13W)
CPU Speed	1.8GHz / 1MB L2 cache
Chipset	Intel® 82810HBM (ICH8M): 31mm x 31mm, 676-pin T-PBGA (2.4W)
BIOS	**AMI** BIOS, support ACPI Function
Memory	DDRIII SO-DIMM x 1, Max. 4GB (Non-ECC)
VGA	Intel® Atom™ D525 Integrated Graphics Controller (GMA3150) Supports Direct X 9 Graphic (400MHz) - VGA x 1 thru onboard pin-header
LVDS	Chrontel CH7036A-BF encoder [10mm x 10mm, 88-pin QFN] Converting 18-bit to 24-bit LVDS; Max. resolution up to 1366 x 768
LAN	Realtek 8111E-VL-CQ PCIe GbE x 2 [48-pin QFN]
USB	ICH8M built-in USB 2.0 host controller, support 8 ports
Serial ATA II	ICH8M built-in SATA II controller, supports 2 ports
Audio	ICH8M built-in audio controller w/ Realtek ALC662 Codec Supports 5.1 CH audio (Line-out, Line-in & MIC)
LPC I/O	Nuvoton NCT6627UD (128-pin LQFP, 14mm x 14mm): COM1 (RS232 only) COM2 (RS232/422/485); COM3/COM4(RS232 only) Hardware monitor (2 x thermal inputs, 4 x voltage monitor inputs, VID0-4 & 3 x Fan Header) **IrDA will be shared with COM# 2 **
Parallel Port	Supporting SPP/EPP/ECP mode, thru onboard pin-header
Digital IO	4 in & 4 out
Keyboard/ Mouse	Pin-header type for PS/2 Keyboard/Mouse
Compact Flash Socket	CF Type II x 1 (Thru ICH8M built-in IDE) CFast x1 [Vertical type socket, 90-degree]
Expansion slot	PCI x 1 ; MiniPCIe x 1[Full-sized]
Onboard Header/ Connector	2x8 pin-header x 1 for CRT 2x5 pin-header x 1 for KB/MS 2x4 pin-header x4 for USB 1~8 DF11-10 box header x 4 for COM1~COM4 2x13 pin-header x 1 for LPT 2x5 pin-header x1 for Digital IO 1x5 pin-header x 1 for IrDA [Optional] 1x4 box-header x 1 for Brightness control 20-pin DF13 LVDS connector x **1** CF Socket x 1, CFast x 1, SATA connector x 2 4-pin power connector x 2 for SATA HDD 20-pin ATX connector x 1, 2-pin power connector for DC-in
Watchdog Timer	Yes (256 segments, 0, 1, 2…255 sec/min)
System voltages	1. DC-in 12V [Default] 2. ATX [12V only, 5V; 3.3V; 5VSB with min. loading design] **DC-in & ATX cannot be used at the same time**
RoHS	Yes
Board Size	203mm x 146mm (8" x 5.75")
Others	1. CPU cooler is needed for D525 CPU 2. LAN Wakeup 3. iSMART function [TI-MSP430G2433IRHB32R Micro Controller] 4. Atmel AT24C02C EEPROM [SO8 type]

Board Dimensions



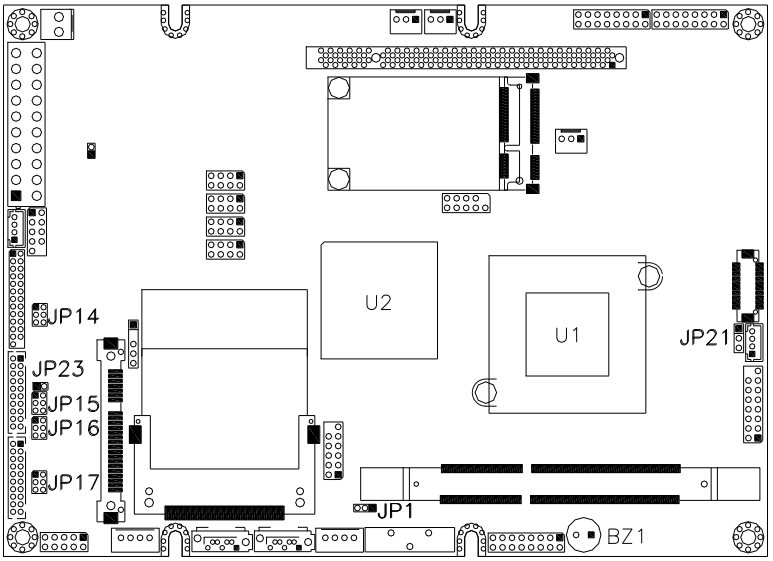
Installations

This section provides information on how to use the jumpers and connectors on the IB815 in order to set up a workable system. The topics covered are:

Setting the Jumpers

Jumpers are used on IB815 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 IB815 and their respective functions.

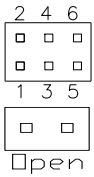
Jumper Locations on IB815



JP1: Clear CMOS Setting

JP1	Setting
	Normal
	Clear CMOS

JP14, JP15, JP16, JP17, JP23: Jumper Select for RS-232, RS-422, RS-485 and IrDA



CONN	Setting			
	RS-232	RS-422	RS-485	IrDA
JP14	1-2	3-4	5-6	Open
JP15	3-5, 4-6	1-3, 2-4	1-3, 2-4	Don't care
JP16	3-5, 4-6	1-3, 2-4	1-3, 2-4	Don't care
JP17	3-4	3-4	3-4	Don't care
JP23	Short	Short	Short	Open

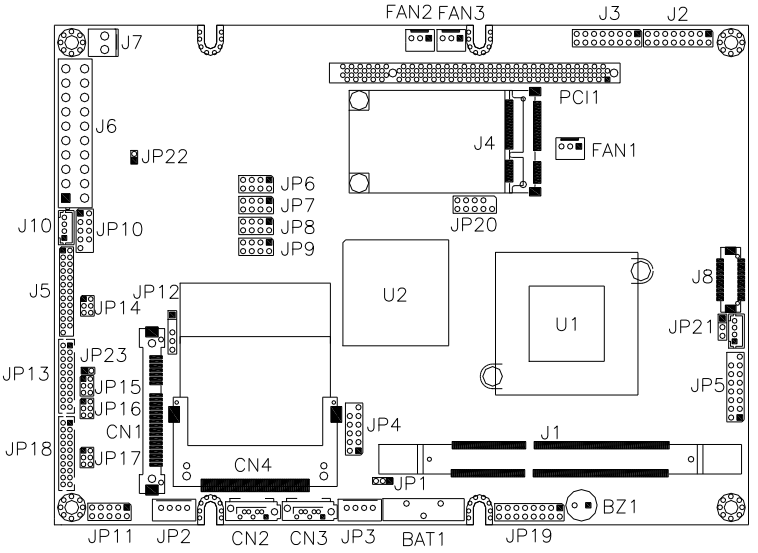
JP21: LVDS Power Setting

JP21	Setting
	+3VS
	+5VS

Connectors on IB815

The connectors on IB815 allows you to connect external devices such as keyboard, floppy disk drives, hard disk drives, printers, etc. The following table lists the connectors on IB815 and their respective functions.

Connector Locations on IB815

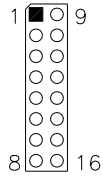


JP2, JP3: HDD Power Connector (Output: Max. 2A)

Pin #	Signal Name
1	+5V
2	Ground
3	Ground
4	+12V

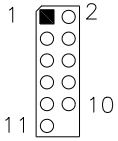
Note: +12V power is provided with 2A maximum load.

J2, J3: LAN Connectors



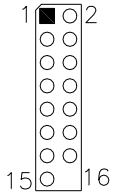
Signal Name	Pin #	Pin #	Signal Name
L1_MDI0P	1	9	L1_MDI0N
L1_VDD10	2	10	CAP to GND
L1_MDI1P	3	11	L1_MDI1N
L1_MDI2P	4	12	L1_MDI2N
NC	5	13	GND
L1_MDI3P	6	14	L1_MDI3N
L1_VDD33	7	15	L1_LED3/EEDO
L1_LED0	8	16	L1_LED1/EESK

JP4: Audio Connectors



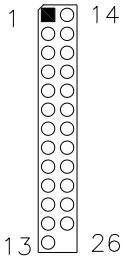
Signal Name	Pin #	Pin #	Signal Name
R_OUT	1	2	L_OUT
GND	3	4	GND
LINE_R	5	6	LINE_L
GND	7	8	GND
MIC_R	9	10	MIC_L
GND	11		

JP5: VGA Connectors



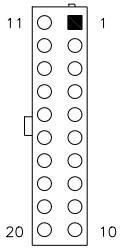
Signal Name	Pin #	Pin #	Signal Name
VGA_R	1	2	+5VS_CRT
VGA_G	3	4	GND
VGA_B	5	6	NC
NC	7	8	VGADDCDATA
GND	9	10	HSYNC
+5VS	11	12	VSYNC
GND	13	14	VGADDCCLK
GND	15		

J5: Parallel Port



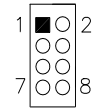
Signal Name	Pin #	Pin #	Signal Name
STB-	1	14	AFD-
PPD0	2	15	ERR#
PPD1	3	16	INIT-
PPD2	4	17	SLIN-
PPD3	5	18	Ground
PPD4	6	19	Ground
PPD5	7	20	Ground
PPD6	8	21	Ground
PPD7	9	22	Ground
ACK#	10	23	Ground
BUSY	11	24	Ground
PE	12	25	Ground
SLCT	13		

J6: AT_12V Connector



Signal Name	Pin #	Pin #	Signal Name
NC	1	11	NC
NC	2	12	NC
GND	3	13	GND
NC	4	14	NC
GND	5	15	GND
NC	6	16	GND
GND	7	17	GND
NC	8	18	NC
NC	9	19	NC
+12VA	10	20	NC

JP6, JP7, JP8, JP9: USB0~USB7 Ports



Signal Name	Pin #	Pin #	Signal Name
+5V	1	2	GND
D-	3	4	D+
D+	5	6	D-
GND	7	8	+5V

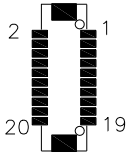
J7: AT_12V Connector

J7 is a DC-in internal connector supporting +12V.

Remarks: J7 and J6 connectors cannot be connected at the same time.

Pin #	Signal Name
1	+12V
2	GND

J8: LVDS Port (24bit)

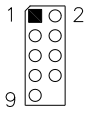


Signal Name	Pin #	Pin #	Signal Name
D0_TX	1	2	D0_TX#
GND	3	4	GND
D1_TX	5	6	D1_TX#
GND	7	8	VDD1
D3_TX	9	10	D3_TX#
D2_TX	11	12	D2_TX#
GND	13	14	GND
CLK_TX	15	16	CLK_TX#
LVDS_BLON-	17	18	VDD1
SPC	19	20	SPD

J9: LVDS Backlight Connector

Pin #	Signal Name
1	+12V
2	Backlight Enable
3	ADJ
4	GND

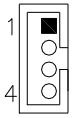
JP10: PS/2 Keyboard and PS/2 Mouse Connectors



Signal Name	Pin #	Pin #	Signal Name
GND	1	2	GND
5V	3	4	5V
MDA	5	6	KBDA
MCL	7	8	KBCL
NC	9		

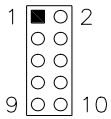
J10: MCU Spy-Bi-Wire Connector

Spy-Bi-Wire test clock/data input during programming and test



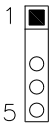
Pin #	Signal Name
1	MCU_3V
2	SBWTCK
3	SBWTDIO
4	GND

JP11: Digital IO 4IN/4OUT Connector



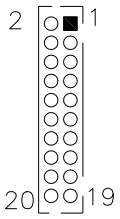
Signal Name	Pin #	Pin #	Signal Name
GND	1	2	+5VS
OUT3	3	4	OUT1
OUT2	5	6	OUT0
IN3	7	8	IN1
IN2	9	10	IN0

JP12: IrDA Connectors



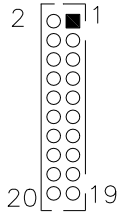
Pin #	Signal Name
1	+5VS
2	
3	IRRX
4	GND
5	IRTX

JP13, JP18: Serial Port (COM1~COM4)



Signal Name	Pin #	Pin #	Signal Name
DCD#1	1	2	DSR#1
SIN1	3	4	RTS#1
SOUT1	5	6	CTS#1
DTR#1	7	8	RI#1
GND	9	10	NC
DCD2_TX-	11	12	DSR#2
SIN2_TX+	13	14	RTS#2
SOUT2_RX+	15	16	CTS#2
DTR2_RX-	17	18	RI#2
GND	19	20	NC

COM2 (JP13) is jumper selectable for RS-232, RS-422 and RS-485



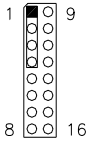
Pin #	Signal Name		
	RS-232	RS-422	RS-485
11	DCD	TX-	DATA-
12	RX	TX+	DATA+
13	TX	RX+	NC
14	DTR	RX-	NC
15	GND	GND	GND
16	DSR	RTS-	NC
17	RTS	RTS+	NC
18	CTS	CTS+	NC
19	RI	CTS-	NC

JP19: System Function Connector

JP19 provides connectors for system indicators that provide light indication of the computer activities and switches to change the computer status. JP19 is a 16-pin header that provides interfaces for the following functions

Speaker: Pins 1 - 4

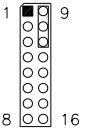
This connector provides an interface to a speaker for audio tone generation. An 8-ohm speaker is recommended.



Pin #	Signal Name
1	SPEAKER
2	NC
3	GND
4	+5VS

Power LED: Pins 9-11

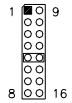
The power LED indicates the status of the main power switch.



Pin #	Signal Name
9	+5VS
10	NC
11	GND

ATX Power ON Switch: Pins 5 and 13

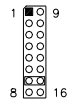
This 2-pin connector is an "ATX Power Supply On/Off Switch" on the system that connects to the power switch on the case. When pressed, the power switch will force the system to power on. When pressed again, it will force the system to power off.



Pin #	Signal Name
5	Power_ON
13	GND

Reset Switch: Pins 7 and 15

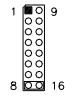
The reset switch allows the user to reset the system without turning the main power switch off and then on again. Orientation is not required when making a connection to this header.



Pin #	Signal Name
7	RESET
15	GND

Hard Disk Drive LED Connector: Pins 8 and 16

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
8	-HDD_LED
16	+5VS

JP20: SPI Debug Port

Signal Name	Pin #	Pin #	Signal Name
		2	NC
SPI_CS0	3	4	3V_DUAL
MISO	5	6	-SPI_HOLD0
-SPI_WP0	7	8	SPICLK
GND	9	10	MOSI

FAN1: CPU Fan Power Connector
FAN1 is a 3-pin header for the CPU fan. The fan must be 12V (Max. 1A).

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

FAN2, FAN3: System Fan Power Connector
FAN2, FAN3 is a 3-pin header for system fans. The fan must be 12V (Max. 1A).

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

CN1: CFAST Port

CN2, CN3: Serial ATA Port

CN4: Compact Flash Connector (top side)

J4: Mini PCI- E(x1) Connector

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:

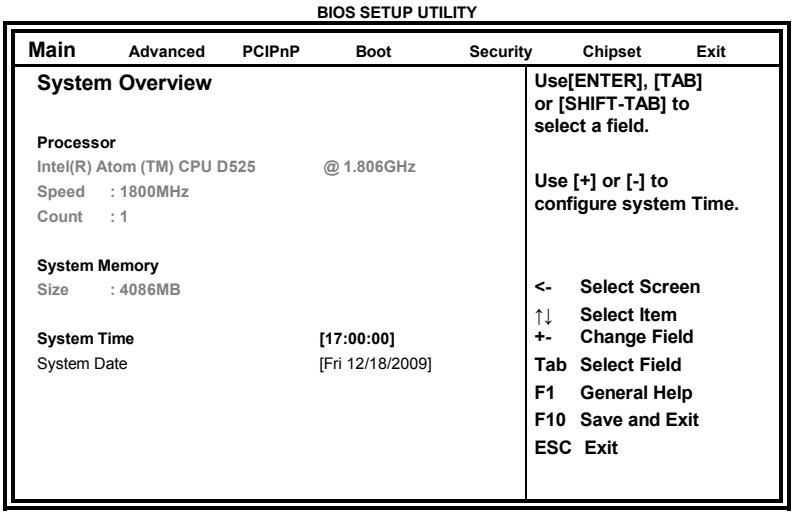
```
Press  <DEL>  to  Enter  Setup
```

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.

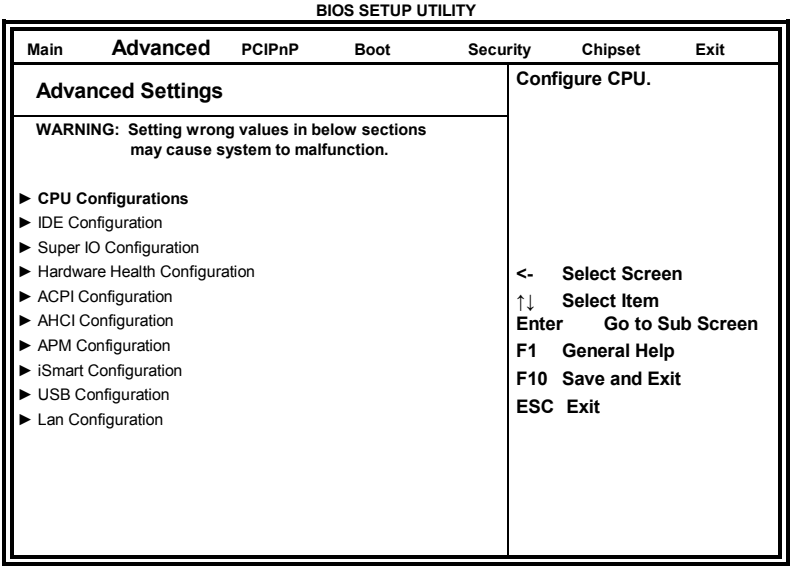


***Note**: If the system cannot boot after making and saving system changes with Setup, the AMI BIOS supports an override to the CMOS settings that resets your system to its default.*

***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.*

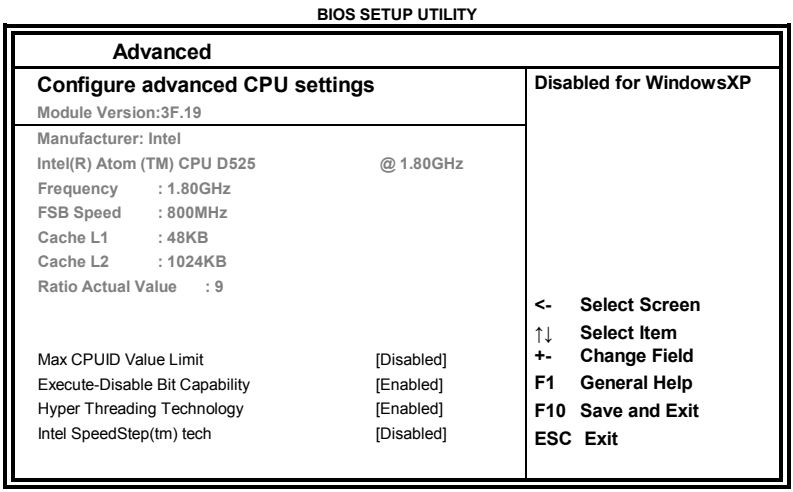
Advanced Settings

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
► iSmart Configuration
► USB Configuration
► LAN 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.



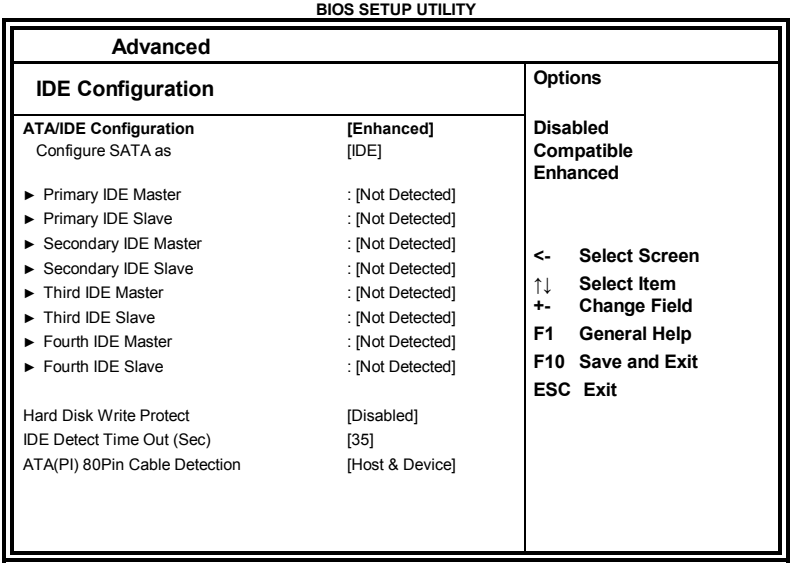
The CPU Configuration menu shows the following CPU details:
Manufacturer: the name of the CPU manufacturer
Brand String: the brand name of the CPU being used
Frequency: the CPU processing speed
FSB Speed: the FSB speed
Cache L1: the CPU L1 cache size
Cache L2: the CPU L2 cache

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.

Intel SpeedStep(tm) tech (Pineview-M)
Disabled: Disable GV3
Enabled: Enable GV3



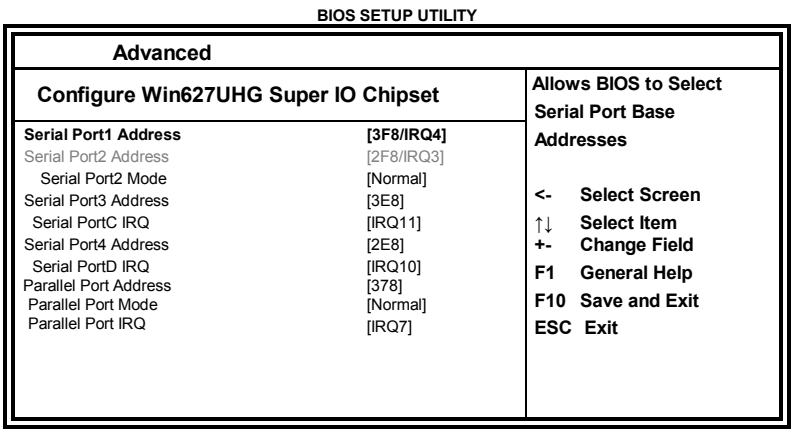
The IDE Configuration menu is used to change and/or set the configuration of the IDE devices installed in the system.

ATA/IDE Configuration

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

Configure SATA as

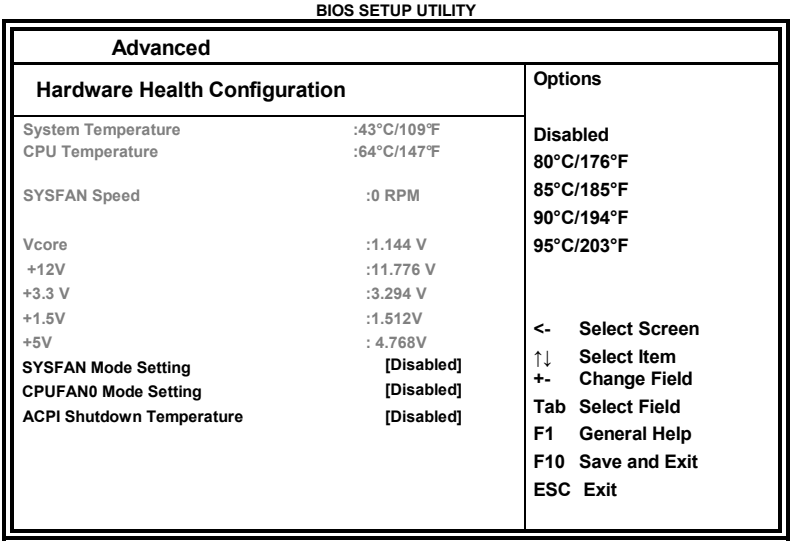
(1) IDE Mode.
(2) AHCI Mode.



Onboard Serial Port

The default values are:

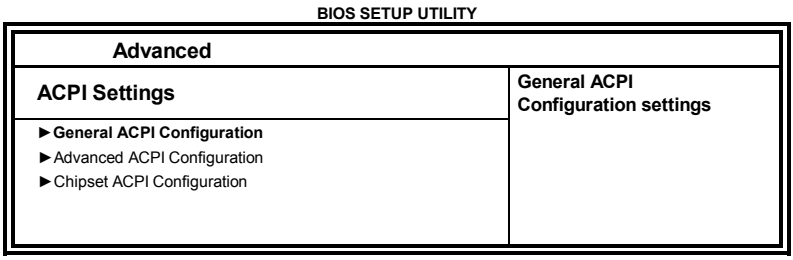
Serial Port 1: 3F8/IRQ4
Serial Port 2: 2F8/IRQ3
Serial Port 3: 3E8/IRQ11
Serial Port 4: 2E8/IRQ10

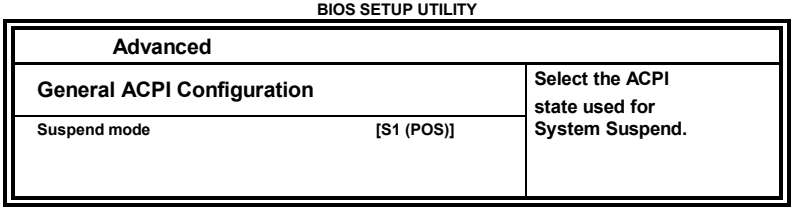


The Hardware Health Configuration menu is used to show the operating temperature, fan speeds and system voltages.

ACPI Shutdown Temperature

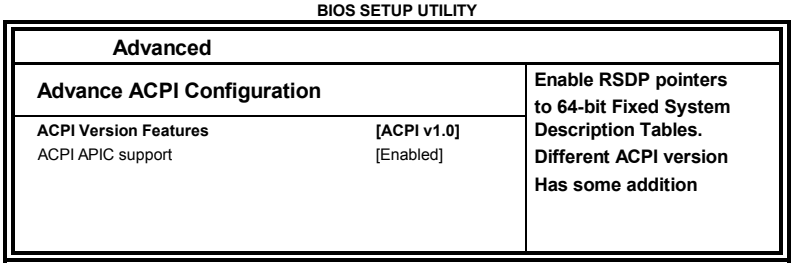
The system will shut down automatically under OS with ACPI mode, when the CPU temperature reaches the configured temperature.

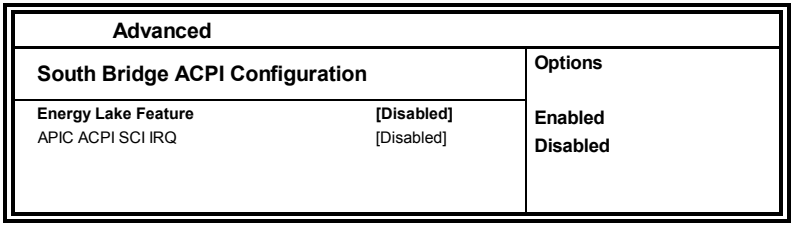




Suspend Mode

By default, the setting for this field is S1(POS) which is the only option supported.





BIOS SETUP UTILITY

Advanced		
AHCI Settings		**While entering setup, BIOS auto detect the presence of IDE device. This displays the status of auto detection of IDE devices.**
AHCI Port0	**[Not Detected]**	
AHCI Port1	[Not Detected]	
AHCI Port2	[Not Detected]	

Advanced		
APM Configuration		**Enable or disable APM.**
Power Management/APM	**[Enabled]**	
Power Button Mode	[On/Off]	
		<- Select Screen
		↑↓ Select Item
Resume On PME#	Disabled	**+- Change Option**
Resume On RTC Alarm	Disabled	**F1 General Help**
		F10 Save and Exit
		ESC Exit

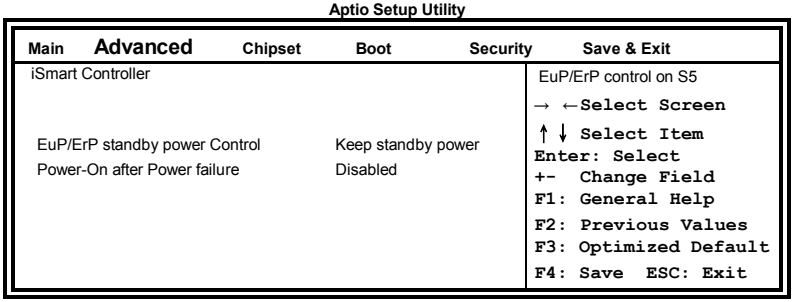
Power Management/APM
By default, this field is set to *Enabled*.

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

Resume on 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

iSmart Controller



EuP/ErP standby power Control

Saving the power consumption on power off.

Power-On after Power failure

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

BIOS SETUP UTILITY

Advanced		
USB Configuration		Enables support for legacy USB. AUTO option disables legacy support if no USB devices are connected.
Module Version - 2.24.5-13.4		
USB Devices Enabled: 1 Keyboard, 1 Mouse		
Legacy USB Support	**[Enabled]**	
USB 2.0 Controller Mode	[HiSpeed]	
BIOS EHCI Hand-Off	[Enabled]	
		<- Select Screen ↑↓ Select Item +- Change Field F1 General Help F10 Save and Exit ESC Exit

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.

BIOS SETUP UTILITY

Advanced		
Lan Configuration		Options
Onboard LAN Option ROM	**[Disabled]**	Enabled Disabled

Onboard LAN Option ROM
Enable or Disable Boot Option for Legacy Network Devices.

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.

Clear NVRAM	**[No]**
Plug & Play O/S	[No]
PCI Latency Timer	[64]
Allocate IRQ to PCI VGA	[Yes]
Palette Snooping	[Disabled]
PCI IDE BusMaster	[Disabled]
OffBoard PCI/ISA IDE Card	**[Auto]**
IRQ3	[Reserved]
IRQ4	[Reserved]
IRQ5	[Available]
IRQ7	[Reserved]
IRQ9	[Available]
IRQ10	[Reserved]
IRQ11	[Reserved]
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 Field
F1 General Help
F10 Save and Exit
ESC Exit

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.

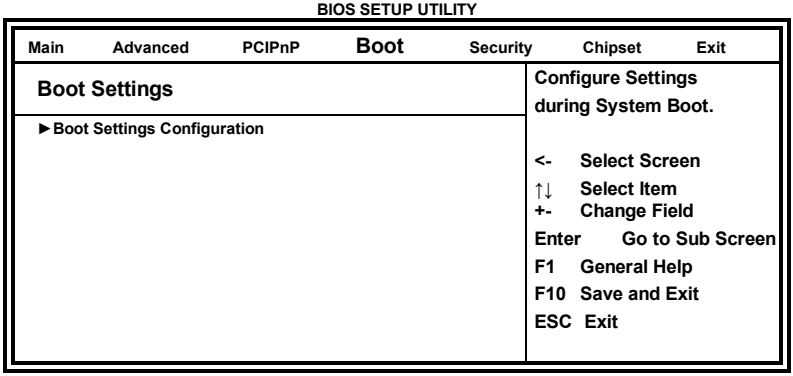
PCI IDE BusMaster
This uses PCI busmastering for BIOS reading / writing to IDE devices.

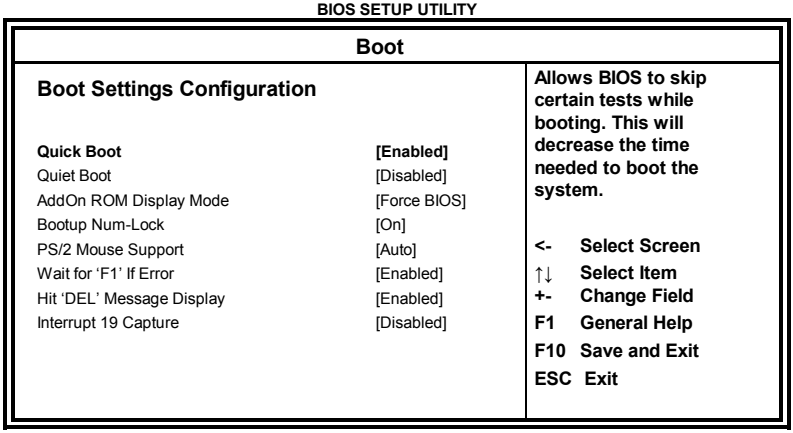
OffBoard PCI/ISA IDE Card
Some PCI IDE cards may require this to be set to the PCI slot number that is holiding the card. ***AUTO***: Works for most PCI IDE cards.

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 for legacy ISA devices.

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.

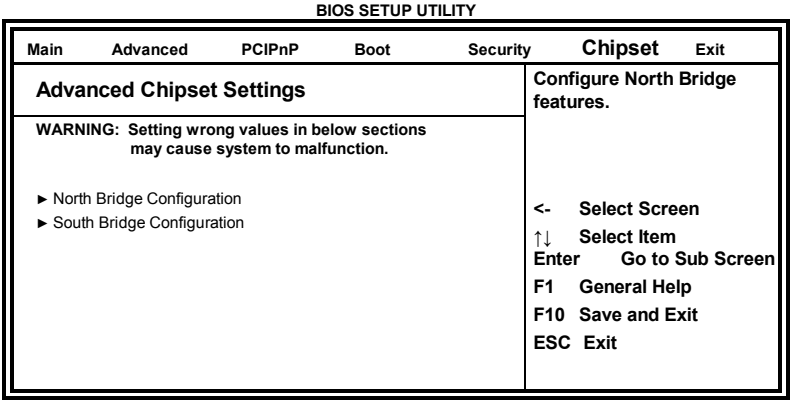
BIOS SETUP UTILITY

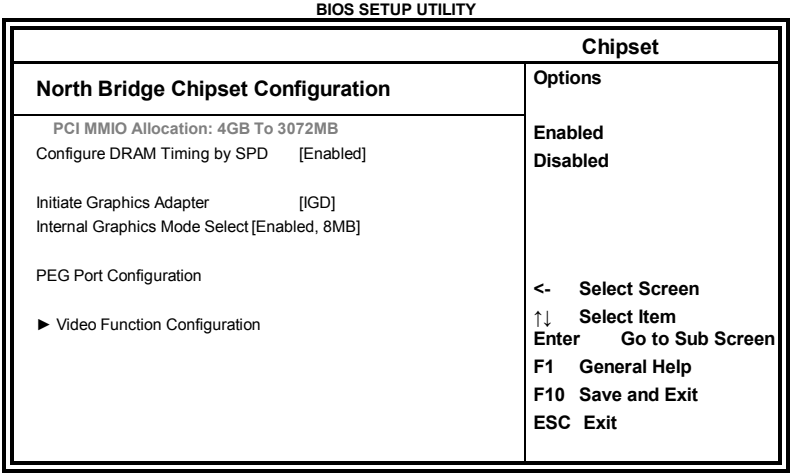
Main	Advanced	PCIPnP	Boot	Security	Chipset	Exit

Security Settings	Install or Change the Password.
Supervisor Password : Not Installed User Password : Not Installed	
Change Supervisor Password Change User Password	<- Select Screen ↑↓ Select Item Enter Change
Boot Sector Virus Protection [Disabled]	F1 General Help F10 Save and Exit ESC Exit

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.





DRAM Frequency

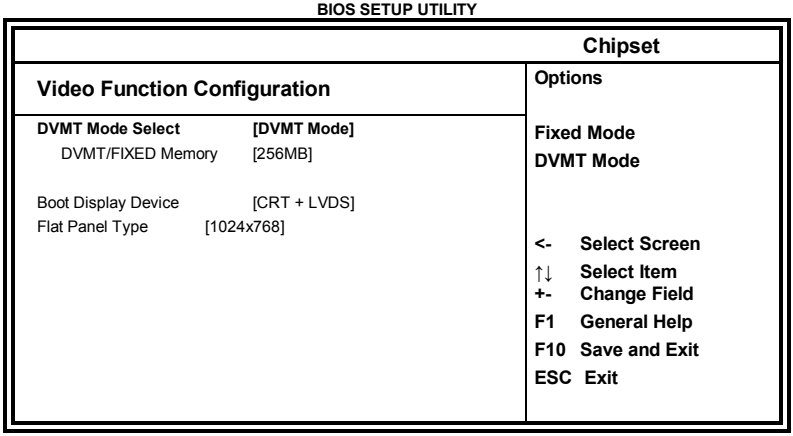
This option supports only 667 MHz – which is the default setting.

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 / 1280x768 / 1440x1050 / 8000x400.

BIOS SETUP UTILITY

Main	Advanced	PCIPnP	Boot	Security	Chipset	Exit

South Bridge Chipset Configuration

USB Function	**[10 USB Ports]**
USB 2.0 Controller	[Enabled]
HDA Controller	[Enabled]
SMBUS Controller	[Enabled]
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	[Disabled]
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]

Options

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

<- Select Screen
↑↓ Select Item
+- Change Field
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.

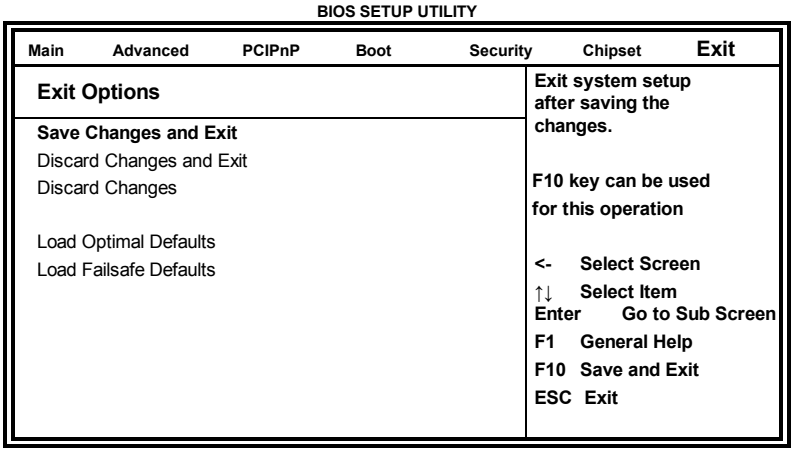
USB 2.0 Controller
This option is enabled by default.

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

SMBUS Controller
This option is enabled 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.

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 LAN Drivers Installation

Follow the steps below to install Realtek 8111E LAN Drivers.

1. Click ***Realtek 8111E LAN Controller Driver..***

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

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

4. 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.

5. When the Setup Options appears, click ***Install Drivers only*** and ***Next*** to continue.

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

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

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
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

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)
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
IRQ15	Intel(R) ICH8 Family SMBus Controller - 283E
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);.....                                    //watcho
}
//-------------------------------------------------------------------------
```

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