MI808

Mini-ITX Motherboard

USER'S MANUAL

Version 1.0

Acknowledgments

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

Table of Contents

This page is intentionally left blank.

Introduction

Product Description

MI808 is a Mini ITX board (170mm x 170mm) that comes with the Intel Intel® Pentium ® N3000 series processor. It has two DDR3L SO-DIMM sockets supporting up to 8GB of system memory.
It also features the Intel® Gen8-LP graphics engine with interface for CRT, DVI-D and 24-bit dual channel LVDS displays.

The platform provides four USB 3.0 connector at the board edge and USB 2.0 two ports with pin header. Two SATA III ports are included.

MI808 FEATURES

- Mini ITX form factor, 170mm x 170mm
- Onboard Intel® Pentium ® N3000 series
- Two DDR3L SO-DIMM sockets, DDR3L-1600, Max. 8GB
- Intel® Gen8-LP graphics for CRT, DVI-D interface
- 24-bit dual channel LVDS interface
- Dual Intel I211-AT PCIe Gigabit LAN
- 4x USB 3.0 on edge, 2xUSB 2.0 support
- Two SATA III, 4x COM ports
- Digital I/O 4-in / 4-out, PCIe (1x) slot, 2x Mini-PCIe
- Watchdog Timer, iSMART, RoHS compliance

Checklist

Your MI808 package should include the items listed below.

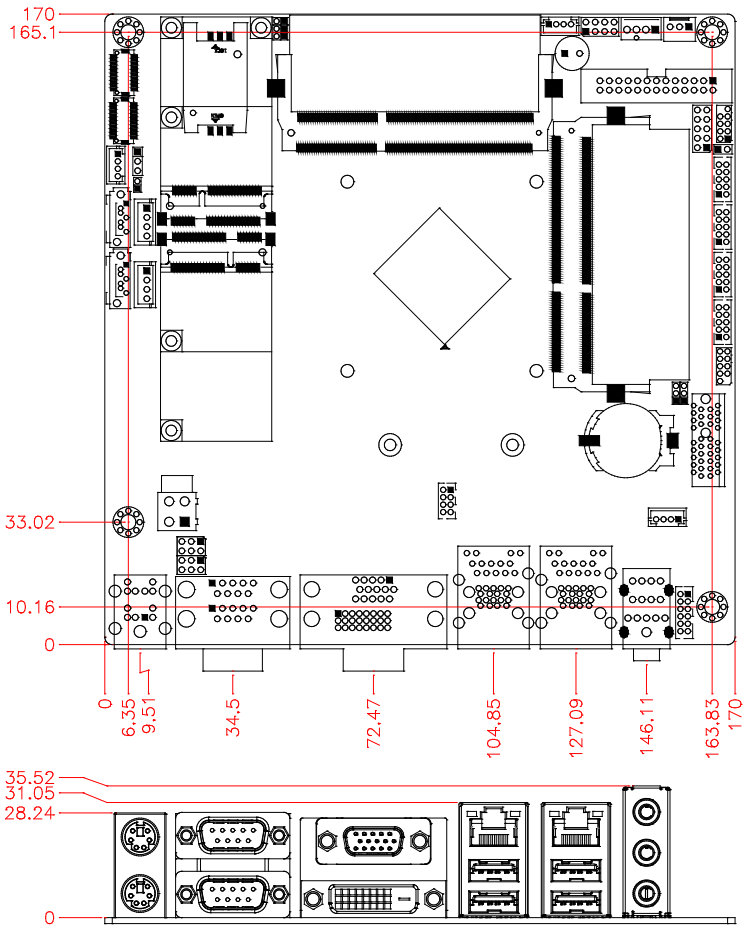
- The MI808 Mini-ITX motherboard
- This User's Manual
- 1 CD containing chipset drivers and flash memory utility
- Serial ATA cable
- I/O shield

MI808 Specifications

Product Name	MI808F-370 (N3700 onboard, DC-in 12V only) MI808FW-370 (N3700 onboard DC-in 18V~24V, wider range) MI808F-300(N3000 onboard DC-in 12V only) MI808FW-300 (N3000 onboard DC-in 18V~24V, wider range) **MI808FW-370A (N3700 onboard DC-in 18V~24V, Non-TPM/CFast)** [Default silk screen model # on PCB is MI808F]
Form Factor	Mini-ITX
SoC Type/Speed	14nm Lithography, FCBGA1170, Package= 25mm x 27mm - Pentium® N3700 QC, up to 2.4Ghz , TDP=6W - Celeron® N3000 DC, up to 2.08GHz, TDP=3W
BIOS	AMI BIOS
Memory	2 xDDR3L SO-DIMM socket [Horizontal type] Maximum to DDR3L-1600@8GB (Non-ECC, Unbuffered, 1.35V)
Display	Intel® Pentium® SoC integrated Gen 8-LP graphics core Supports DX11.1 & OpenGL 4.2, 3 x independent displays - VGA x 1(Thru NXP PTN3392) **Supports up to1920x1200 @ 60Hz - LVDS: 24-bit dual channel via NXP PTN3460 thru eDP **Supports up to1920x1200 @ 60Hz** - DVI-D x1 **Supports up to 3840x2160 @ 60 Hz**
LAN	Intel I211-AT PCIe Gigabit LAN x 2 **Reserved for external EEPROM**
USB	Intel® Braswell SoC built-in USB 3.0 host controller, supports **4** ports - Edge I/O x 4 Intel® Braswell SoC built-in USB 2.0 host controller with USB hub Renesas uPD720115K8-611-BAK-A(C013720115K861000P) For total 4 ports x USB 2.0 - 2 ports thru pin header - 2 ports thru Mini PCIe slot
Serial ATA Ports	Intel® Braswell SoC built-in SATA III controller, supports 2 ports - 2 x SATAIII connector - MSATA (thru Mini PCIe) via switch - CFast (Optional) via switch
Audio	Realtek ALC269QHD-VC**3** with class-D speaker amplifier (2W per channel) Support 2-channel audio out + AMP
LPC I/O	Fintek F81846AD-I [128-pin LQFP, 14x14x1.4mm] COM #1 (RS232/422/485) supports ring-in with power @500mA (selectable for 5V or 12V) - [SP339EER1 232/422/485 transceiver x 1 for jumper-less] - COM #2 (RS232 only), support ring-in with power @500mA (selectable for 5V or 12V), with SP3243EBER - COM #3~COM #4 (RS232 only) with SP3243EBER [Hardware Monitor] 2 x Thermal inputs; 2 x Voltage monitoring 1 x 4-pin for CPU fan (PWM), 1 x 3-pin for SYS fan (DC mode)
Digital IO	4-in / 4-out (User configurable)
Expansion Slots	PCIe (1x) slot x 1 Full-sized Mini-PCIe x **2** [Mounting holes for full-sized (x2) + half-sized (x1)] Mini-PCIe #1 supports PCIe(1x) / USB signal Mini-PCIe #2 supports mSATA / USB signal CFast x 1 @ solder side [Except MI808FW-370A]

Edge Connector (The same as MI802)	PS/2 Keyboard + PS/2 mouse stack connector x 1 Dual DB9 Stack connector x1 for COM#1 / COM#2 VGA+ DVI-D stack connector x 1 for CRT + DVI-D RJ45+Dual USB 3.0 stack connector x 2 Audio 3-port connector x 1 (Line-out, Line-in, MIC)
Onboard Header/Connector	DF11-8 pin connector x 1 for USB (2.0) x 2 ports 2x6 pin header x 1 for front audio (2.54 pitch type) 2x13 pin box header x 1 for LPT DF11-10 pin box header x 4 for COM #3~# 6 DF20 socket connector x 2 for 24-bit dual channel LVDS 4 pins box header x1 for LCD backlight control SATA connector x 2 for SATA III device (Blue color) 4-pin power connector x 2 for SATA device 2x5 pin header x 1 for Digital I/O (2.54 pitch type) 4-pin header for speaker out (from ALC269 internal amplifier) 2x4 pin header x 1 for front I/O panel (2.54 pitch type) 2 x 2 pin connector x 1 for power-in SIM slot x 1 (Location under #1 MiniPCIe)
Watchdog Timer	Yes (256 segments, 0, 1, 2…255 sec/min)
Power Connector	+12V only (±5 % tolerance) or 18V~+24V
Others	1. iSMART 3.2 **2.** TPM 2.0 (Infineon SLB9665) [Except MI808FW-370A] 3. Heatsink for fanless solution
OS supporting	- Windows 8.1 (64-bit) - Window Embedded 8 - Windows 7 (64-bit) - Linux (Fedora, uBuntu)
RoHS	Yes
Certification	CE /FCC/LVD
Environment	Operation Temperature: 0~60 degree C Storage Temperature: -40 ~ 80 degree C Relative humidity: 90%, non-condensing @ 60 degree C
Board Size	170mm x 170mm

Board Dimensions



Installations

This section provides information on how to use the jumpers and connectors on the MI808 in order to set up a workable system. The topics covered are:

Installing the Memory

The MI808 board supports two DDR3L-1600 memory slots.

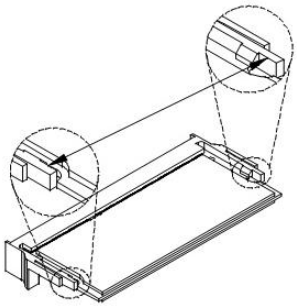
Remarks:
MI808 supports two SO-DIMM (w/o ECC) modules. Total maximum memory supported is 8GB.

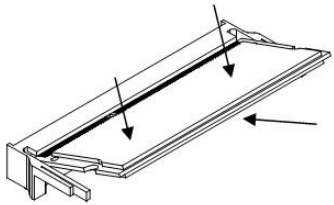
Installing and Removing Memory Modules

To install the DDR3L modules, locate the memory slot on the board and perform the following steps:

1. Hold the DDR3L module so that the key of the DDR3L module aligned with that on the memory slot.
2. Gently push the DDR3L module in an upright position until the clips of the slot close to hold the DDR3L module in place when the DDR3L module touches the bottom of the slot.

To remove the DDR3L module, press the clips with both hands.

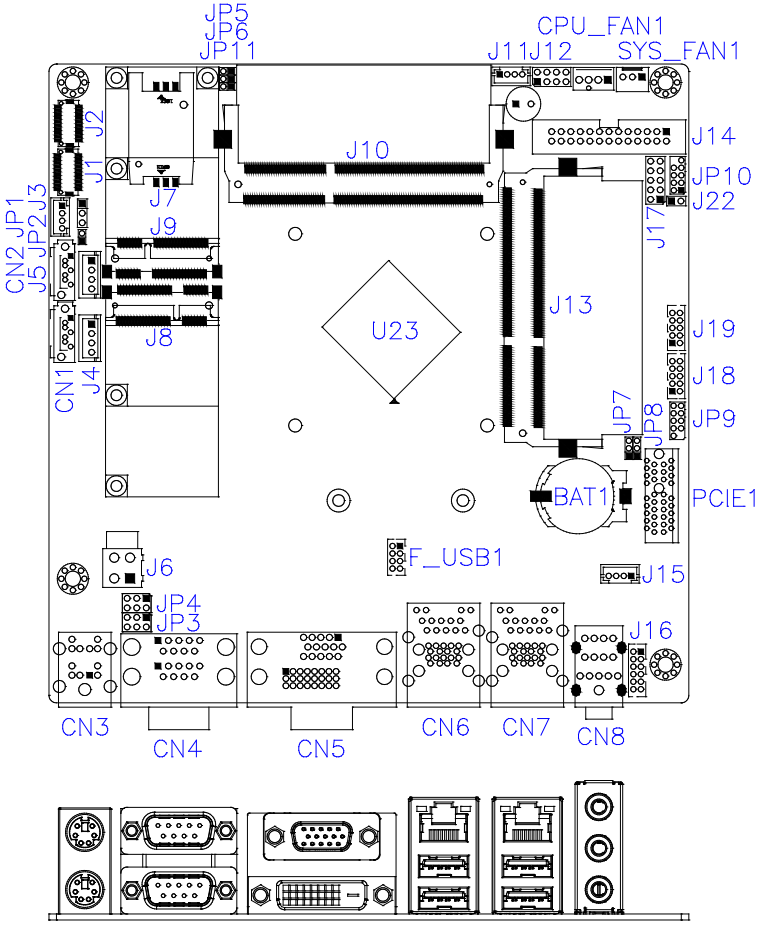




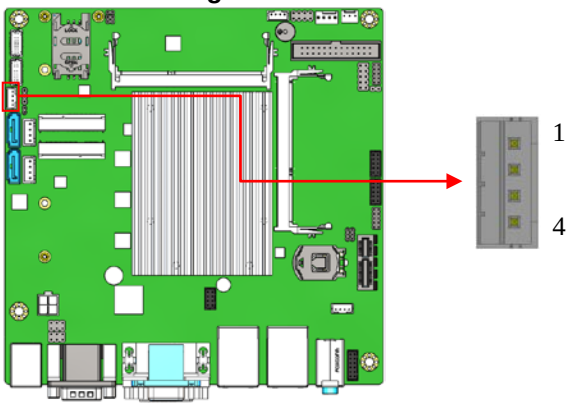
Setting the Jumpers

Jumpers are used on MI808 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 MI808 and their respective functions.

Jumper Locations on MI808

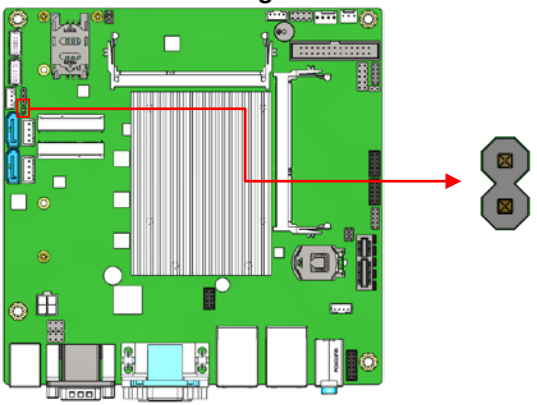


JP1: LCD Backlight Connector



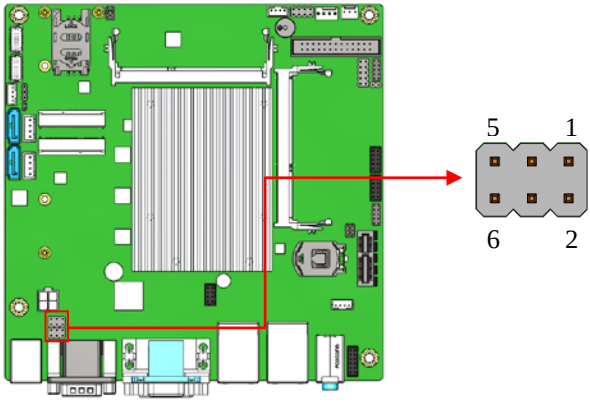
Pin #	Signal Name
1	+12V
2	Backlight Enable
3	Brightness Control
4	Ground

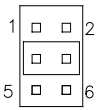
JP2: LVDS Panel Brightness Control Selection



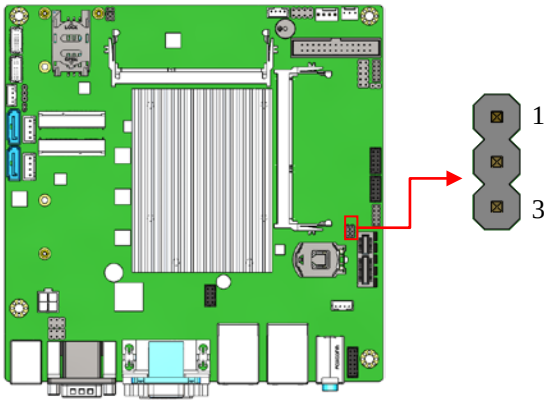
Setting	Brightness Control (PWM mode)
Open	3.3V
Close	5V(Default)

JP3/JP4 COM1/COM2 RS232 RI/+5V/+12V Power Setting



JP3/JP4	Setting	Function
	Pin 1-3 Short/Closed	+12V
	Pin 3-4 Short/Closed	RI
	Pin 3-5 Short/Closed	+5V

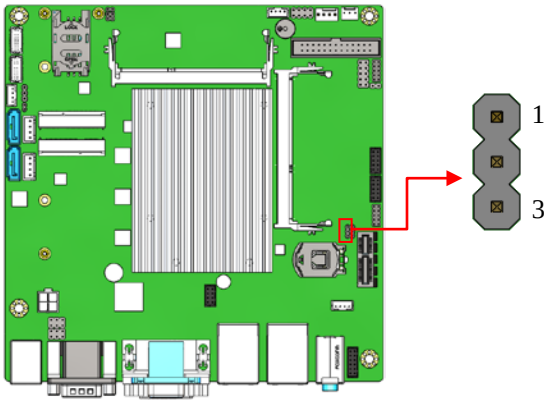
JP8: Clear CMOS Contents



JP8	Setting	Function
	Pin 1-2 Short/Closed	Normal

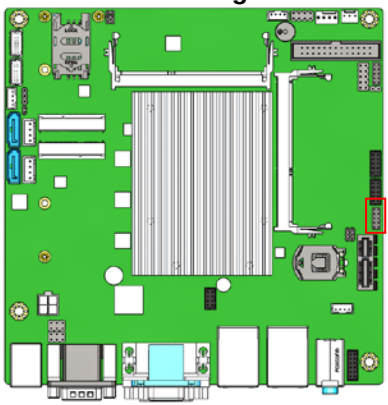
	Pin 2-3 Short/Closed	Clear CMOS

JP7: Clear ME Contents

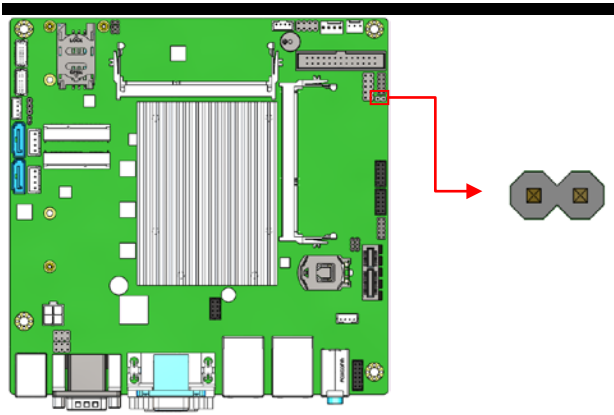


JP7	Setting	Function
	Pin 1-2 Short/Closed	Normal
	Pin 2-3 Short/Closed	Clear ME REGISTER

JP9: For SPI Debug Tools



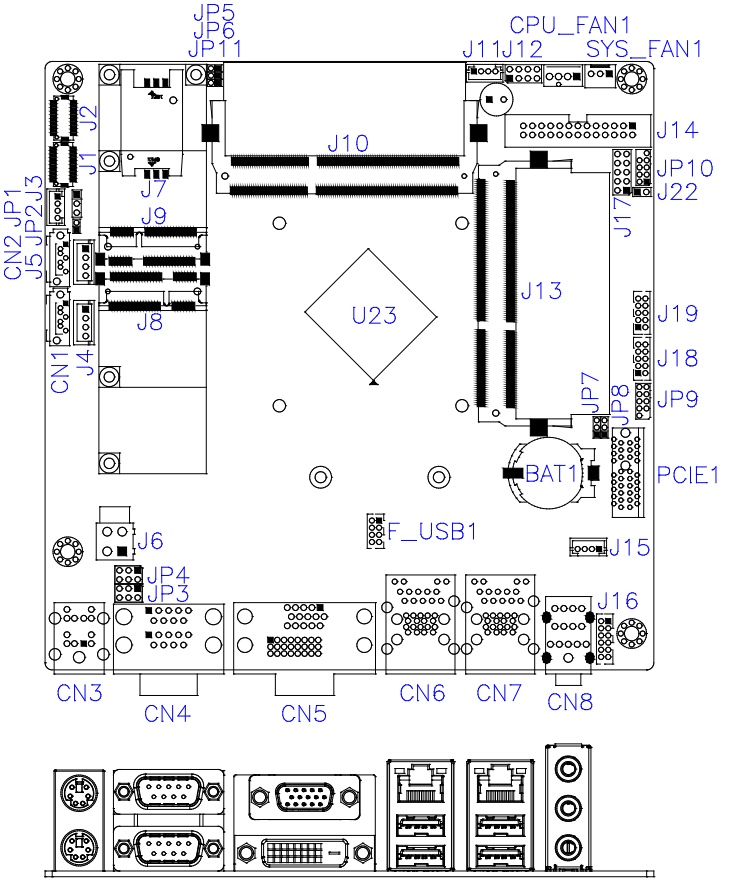
J22: ATX/AT Mode



Setting	Function
Open	ATX (Default)
Close	AT

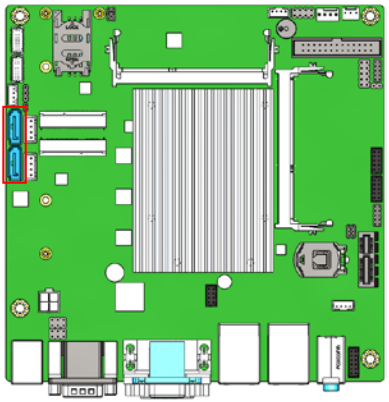
Connectors on MI808

Connector Locations on MI808



Connector Locations on MI808

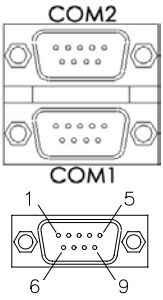
CN1 / CN2: SATA3 Connectors



CN2 Share with CN9 (CFAST)

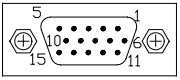
CN3: PS2 Keyboard/Mouse

CN4: COM1 / COM2

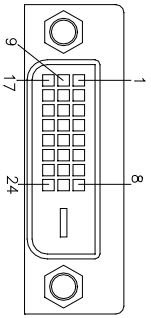


Pin #	Signal Name		
	RS-232	R2-422	RS-485
1	DCD	TX-	DATA-
2	RX	TX+	DATA+
3	TX	RX+	NC
4	DTR	RX-	NC
5	Ground	Ground	Ground
6	DSR	NC	NC
7	RTS	NC	NC
8	CTS	NC	NC
9	RI	NC	NC
10	NC	NC	NC

CN5: CRT+DVI Connector



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

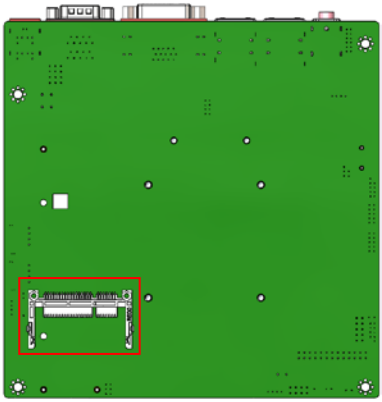


Signal Name	Pin #	Pin #	Signal Name
DATA2-	1	2	DATA2+
GND	3	4	N.C.
N.C.	5	6	DDCCLK
DDCDATA	7	8	N.C.
DATA1-	9	10	DATA1+
GND	11	12	N.C.
N.C.	13	14	VCC
GND	15	16	Hot Plug Detect
DATA0-	17	18	DATA0+
GND	19	20	N.C.
N.C.	21	22	GND
CLK+	23	24	CLK-

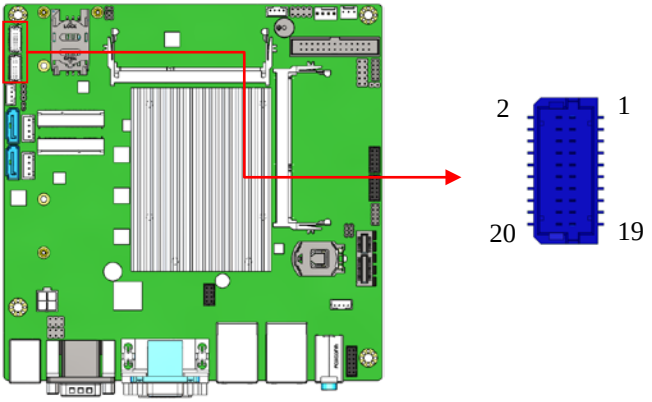
CN6/CN7: Gigabit LAN / USB 3.0 Connector

CN8: Audio Connector

CN9: CFAST Connector



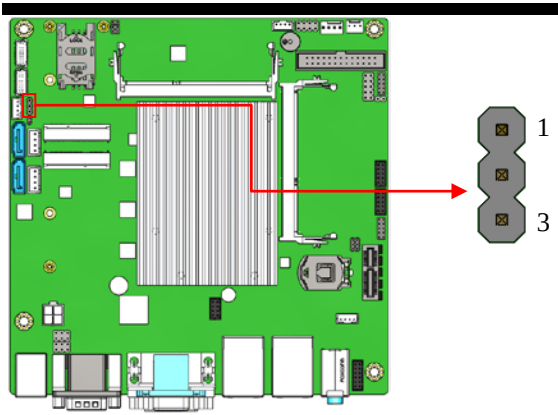
J1, J2: LVDS Connectors, Hirose DF20G-20DP-1V

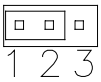
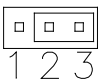


J2: First Channel LVDS
J1: Second Channel LVDS

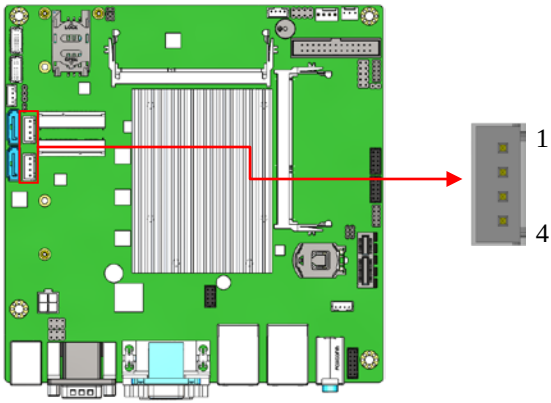
Signal Name	Pin #	Pin #	Signal Name
TX0N	2	1	TX0P
Ground	4	3	Ground
TX1N	6	5	TX1P
Ground	8	7	Ground
TX2N	10	9	TX2P
Ground	12	11	Ground
CLKN	14	13	CLKP
Ground	16	15	Ground
TX3N	18	17	TX3P
Power	20	19	Power

J3: LVDS Panel Power Selection



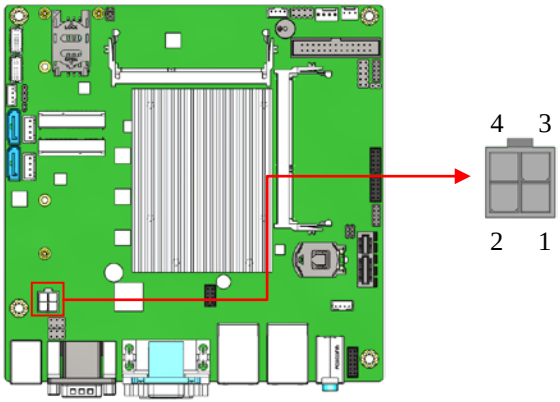
J3	Setting	Panel Voltage
	Pin 1-2 Short/Closed	3.3V (default)
	Pin 2-3 Short/Closed	5V

J4 / J5: SATA HDD Power Connectors



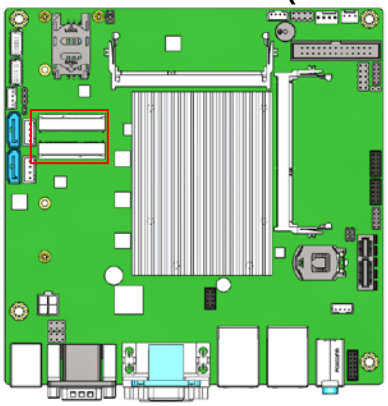
Pin #	Signal Name
1	+5V
2	Ground
3	Ground
4	+12V

J6: DC_IN Connector



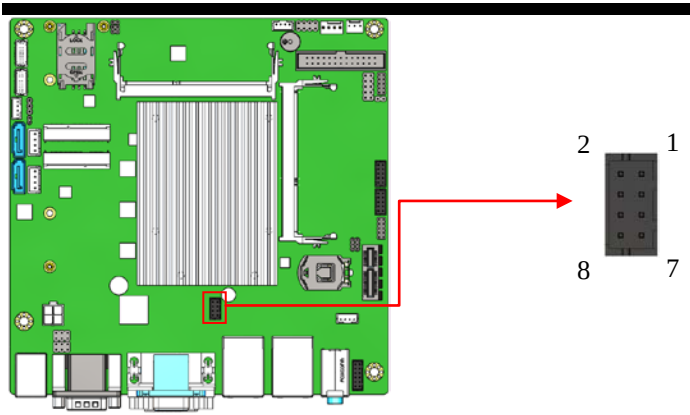
Pin #	Signal Name
1	Ground
2	Ground
3	+12V or +24V
4	+12V or +24V

J8 / J9: Mini PCIE Slot (SIM Card J7)



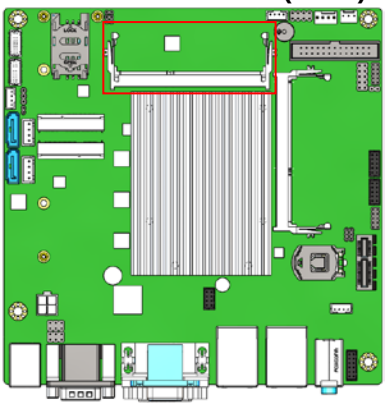
J8 mSATA only (Share with CN1, W/O USB)

F_USB1: USB 2.0

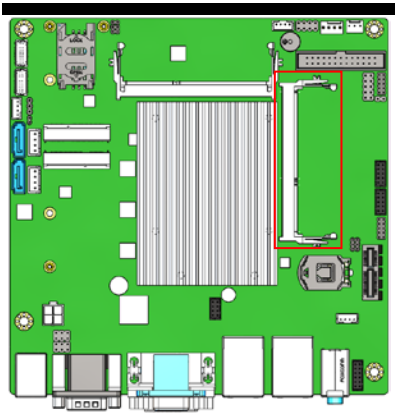


Signal Name	Pin #	Pin #	Signal Name
Vcc	1	2	Ground
D0-	3	4	D1+
D0+	5	6	D1-
Ground	7	8	Vcc

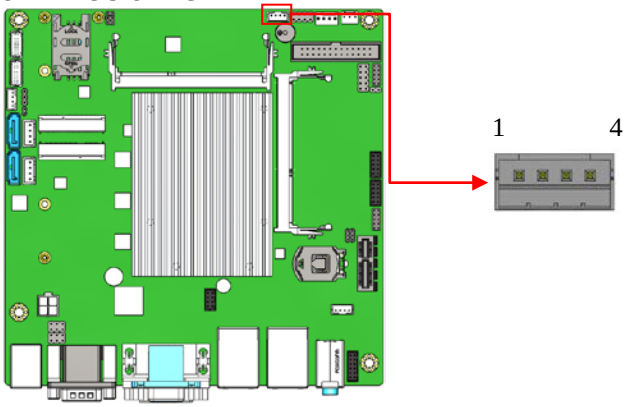
J10: DDR3L SO-DIMM (CH-A) Sockets



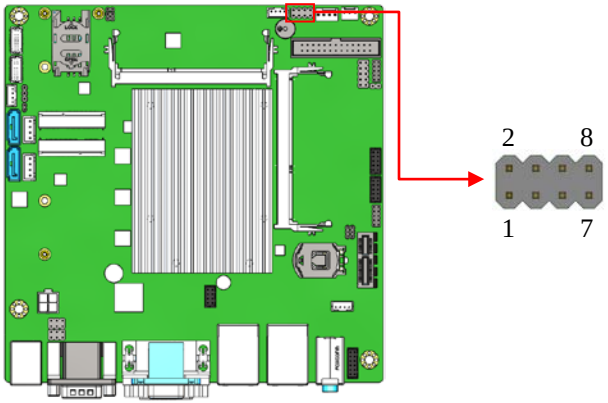
J13: DDR3L SO-DIMM (CH-B) Sockets



J11: MCU JTAG

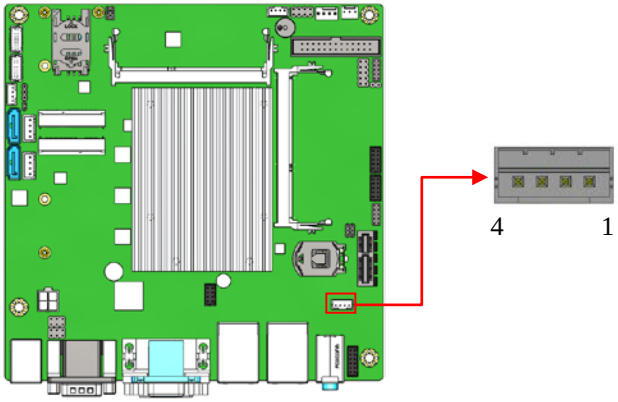


J12: Front Panel



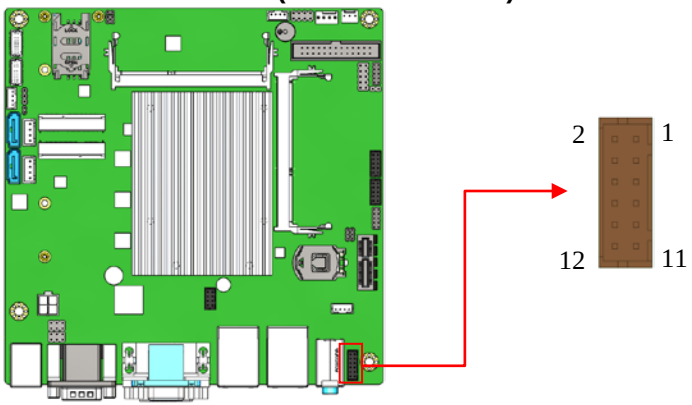
Signal Name	Pin #	Pin #	Signal Name
GND	1	2	PWR_BTN
3.3V	3	4	HDD Active
GND	5	6	Reset
+5V	7	8	GND

J15 Amplifier Connector



Pin #	Signal Name
1	OUTL+
2	OUTL-
3	OUTR-
4	OUTR+

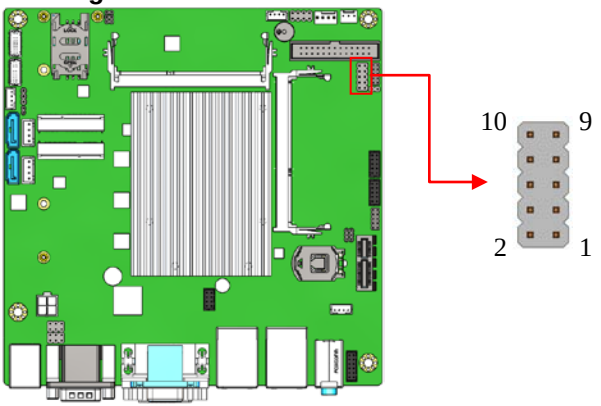
J16 Audio Connector (DF11-12DP-2DSA)



Signal Name	Pin #	Pin #	Signal Name
LINEOUT_R	2	1	LINEOUT_L
Ground	4	3	JD_FRONT
LINEIN_R	6	5	LINEIN_L

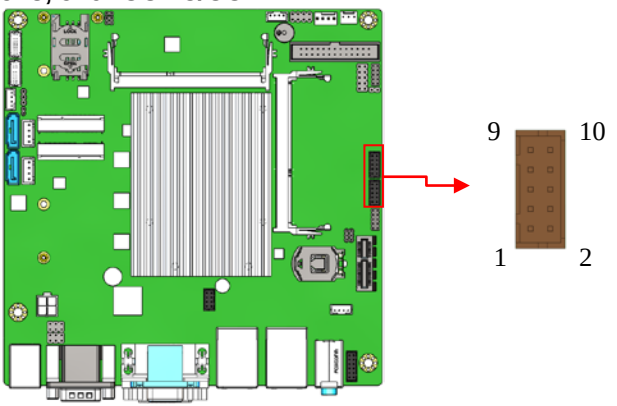
Ground	8	7	JD_LINEIN
MIC-R	10	9	MIC_L
Ground	12	11	JD_MIC1

J17: Digital I/O



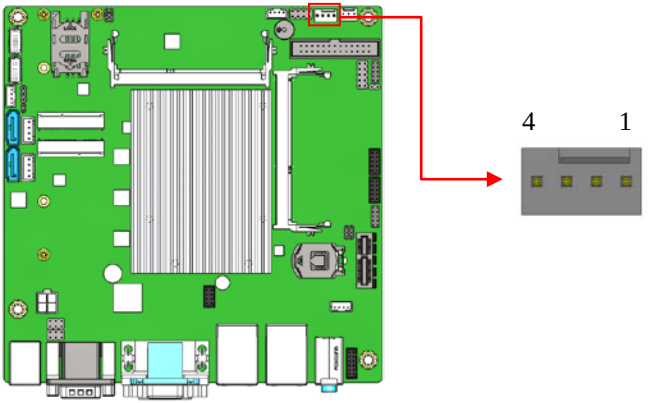
Signal Name	Pin #	Pin #	Signal Name
GND	1	2	VCC
OUT3	3	4	OUT1
OUT2	5	6	OUT0
IN3	7	8	IN1
IN2	9	10	IN0

J18, J19: COM3/COM4



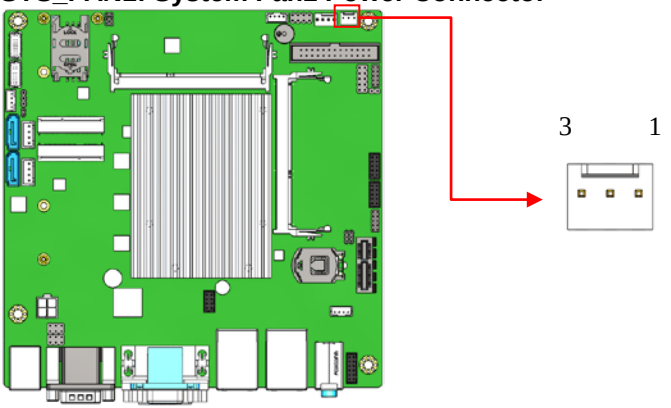
Signal Name	Pin #	Pin #	Signal Name
Data carrier detect	1	2	Receive data
Transmit data	3	4	Data terminal ready
Ground	5	6	Data set ready
Request to send	7	8	Clear to send
Ring indicator	9	10	Not Used

CPU_FAN1: CPU Fan Power Connector

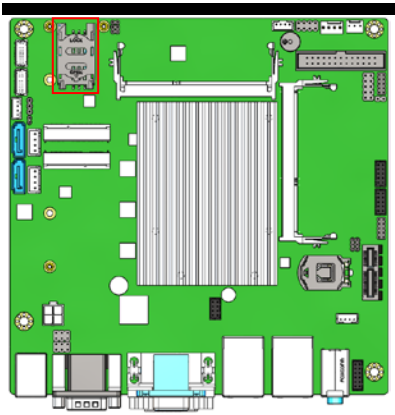


Pin #	Signal Name
1	Ground
2	+12V
3	Rotation detection
4	Control

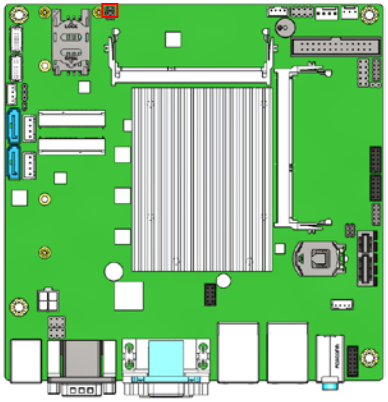
SYS_FAN1: System Fan1 Power Connector



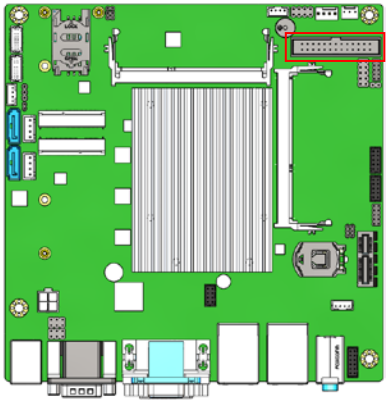
J7: SIM Card Slot



JP5/JP6: Reserved



J14: LPT Port Connector



This page is intentionally left blank.

BIOS Setup

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

BIOS Introduction

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

BIOS Setup

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

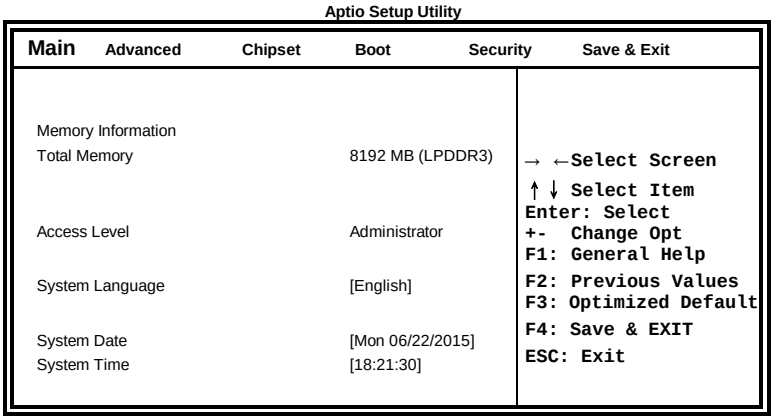
```
Press <DEL> to Enter Setup
```

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

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

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

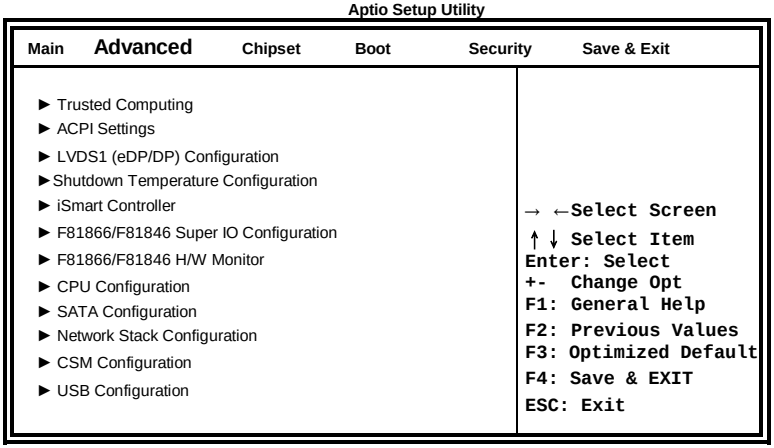
Main Settings



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

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

Advanced Settings



Trusted Computing

Aptio Setup Utility

Main	Advanced	Chipset	Boot	Security	Save & Exit
TPM20 Device Found					→ ←Select Screen
Security Device Support			Enable		↑ ↓ Select Item
Pending operation			None		Enter: Select
Platform Hierarchy			Enabled		+- Change Opt.
Storage Hierarchy			Enabled		F1: General Help
Endorsement Hierarchy			Enabled		F2: Previous Values
HashPolicy			Sha-1		F3: Optimized Defaults
TPM 20 InterfaceType			TIS		F4: Save & Exit
Device Select			Auto		ESC: Exit

Security Device Support

Enables or Disables BIOS support for security device. O.S. will not show Security Device. TCG EFI protocol and INT1A interface will not be available.

Pending operation

Schedule an Operation for the Security Device. NOTE: Your Computer will reboot during restart in order to change State of Security Device.

Platform Hierarchy

Enable or Disable Platform Hierarchy

Storage Hierarchy

Enable or Disable Storage Hierarchy

Endorsement Hierarchy

Enable or Disable Endorsement Hierarchy

HashPolicy

Select the Hash policy to use. SHA-2 is most secure but might not be supported by all Operating Systems

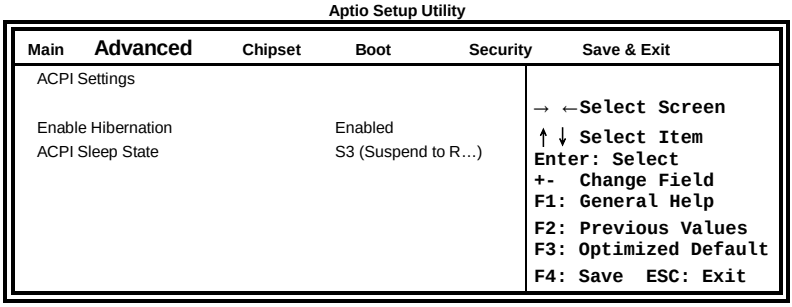
TPM 20 InterfaceType

Select the Communication Interface to TPM 20 Device.

Device Select

TPM 1.2 will restrict support to TPM 1.2 devices, TPM 2.0 will restrict support to TPM 2.0 devices, Auto will support both with the default set to TPM 2.0 devices if not found, TPM 1.2 devices will be enumerated.

ACPI Settings



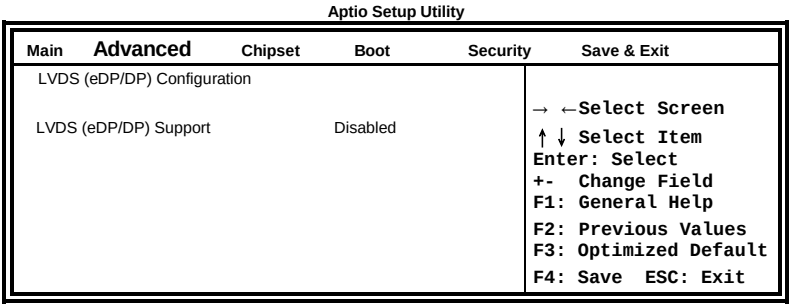
Enable Hibernation

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

ACPI Sleep State

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

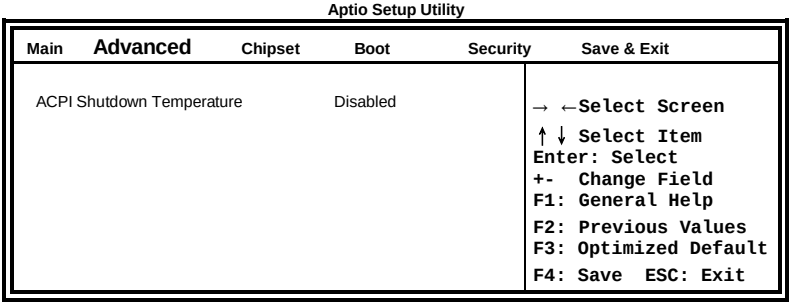
LVDS (eDP/DP) Configuration



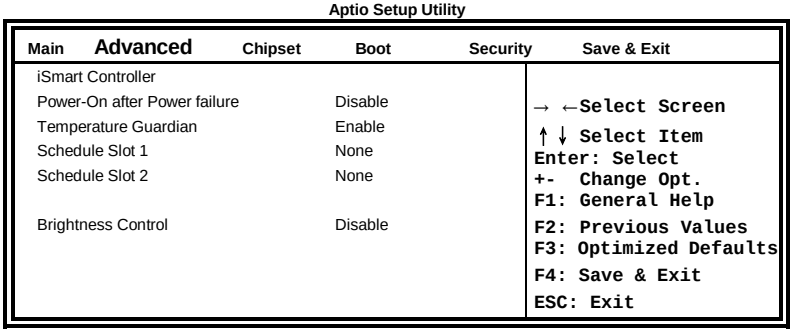
LVDS (eDP/DP) Support

LVDS (eDP/DP) ON/OFF

Shutdown Temperature Configuration



iSmart Controller



Power-On after Power failure

This field sets the system power status whether Disable or Enable when power returns to the system from a power failure situation.

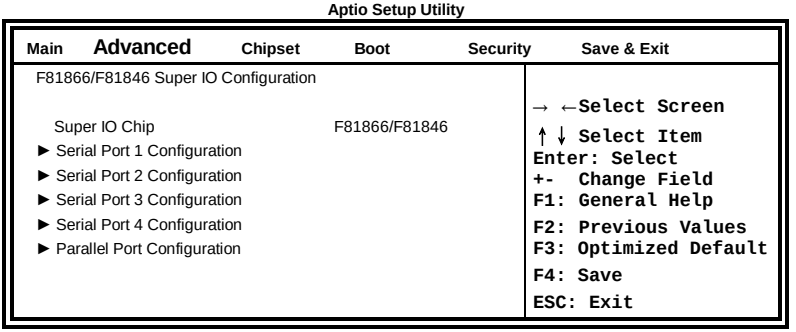
Temperature Guardian

Generate the reset signal when system hangs up on POST.

Schedule Slot 1 / 2

Setup the hour/minute for system power on.

F81866/F81846 Super IO Configuration



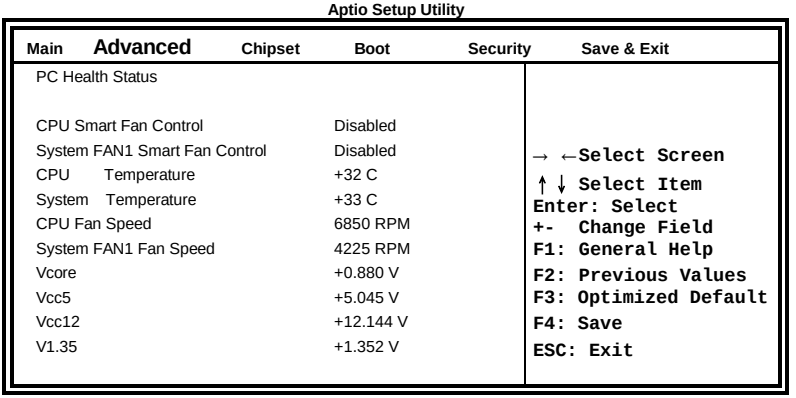
Serial Port Configuration

Set Parameters of Serial Ports (COM)

Parallel Port Configuration

Set Parameters of Parallel Port (LPT/LPTE)

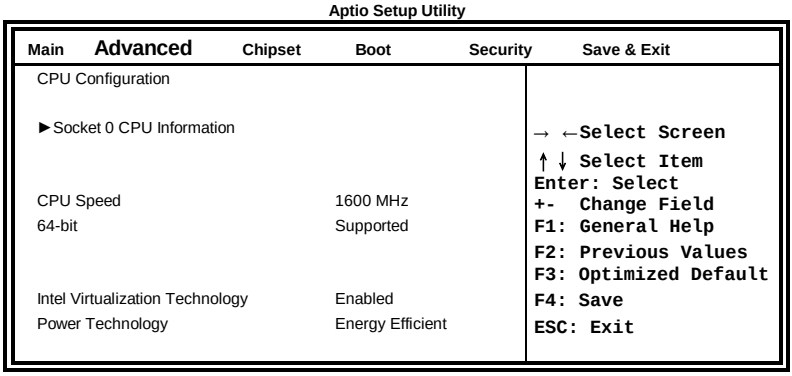
F81866/F81846 H/W Monitor



Temperatures/Voltages

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

CPU Configuration



Socket 0 CPU Information

Socket specific CPU Information

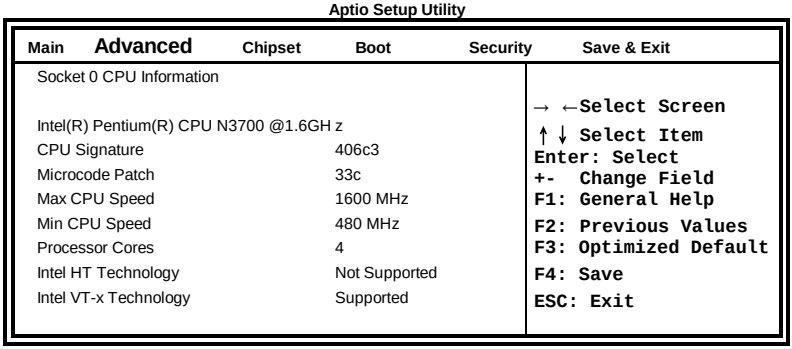
Intel Virtualization Technology

When enabled, a VMM can utilize the additional hardware capabilities provided by Vanderpool Technology.

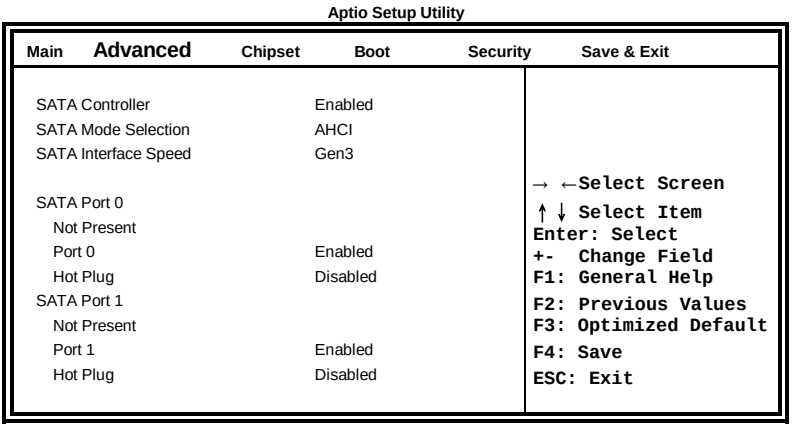
Power Technology

Enable the power management features.

Socket 0 CPU Information



SATA Configuration



SATA Controller
Enable / Disable SATA Device

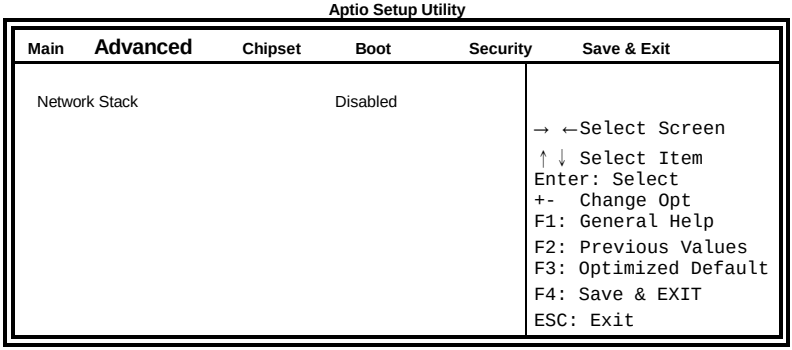
SATA Mode Selection
Determines how SATA controller operate.

SATA Interface Speed
Select SATA Interface Speed CHV A1 always with Gen1 Speed.

Port
Enable or Disable SATA Port

Hot Plug
Designates this port as Hot Pluggable.

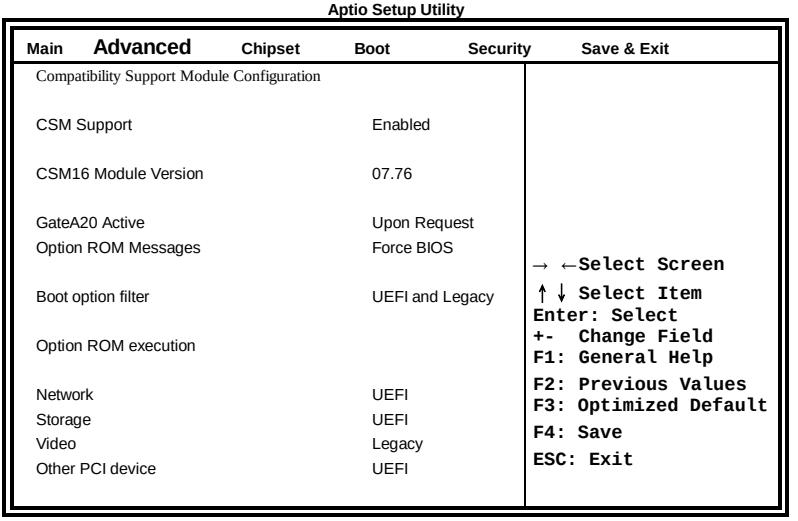
Network Stack Configuration



Network Stack

Enables/Disable UEFI Network Stack

CSM Configuration



CSM Support
Enable/Disable CSM Support.

Boot option filter
This option controls Legacy/UEFI ROMs priority

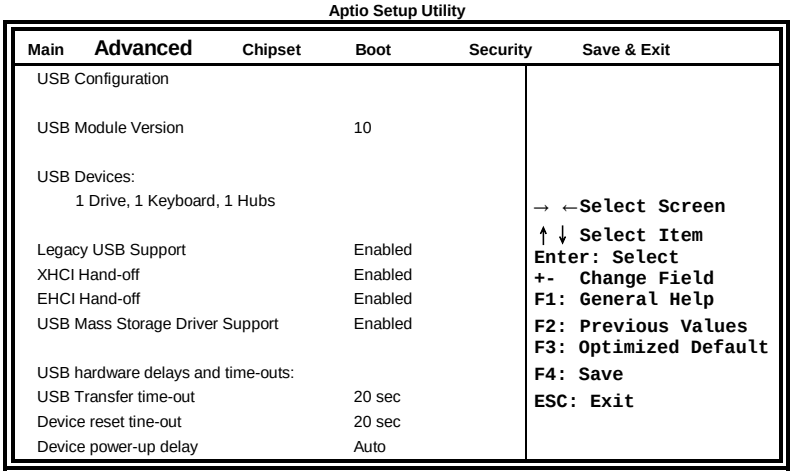
Network
Controls the execution of UEFI and Legacy PXE OpROM

Storage
Controls the execution of UEFI and Legacy Storage OpROM

Video
Controls the execution of UEFI and Legacy Video OpROM

Other PCI device
Determines OpROM execution policy for devices other than Network, Storage, or Video.

USB Configuration



Legacy USB Support

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

XHCI Hand-off

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

EHCI Hand-off

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

USB Mass Storage Driver Support

Enable/Disable USB Mass Storage Driver Support.

USB Transfer time-out

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

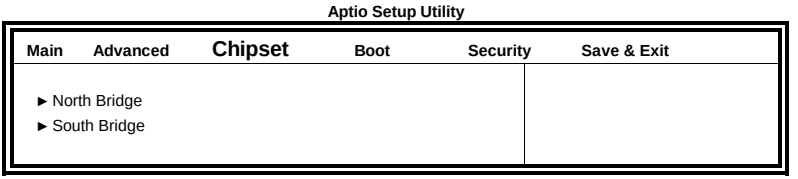
Device reset time-out

USB mass Storage device start Unit command time-out.

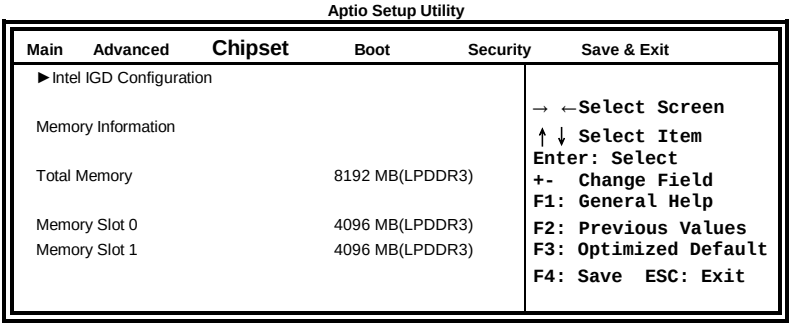
Device power-up delay

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

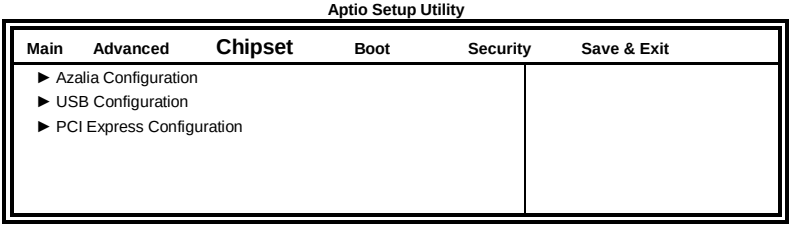
Chipset Settings



North Bridge



South Bridge



Azalia Configuration
Azalia HD Audio Options

USB Configuration
USB Configuration Settings

PCI Express Configuration
PCI Express Configuration Settings

Azalia Configuration

Main	Advanced	Chipset	Boot	Security	Save & Exit
Azalia Configuration					
Audio Controller			Enabled		
Azalia HDMI Codec			Enabled		
Azalia HDMI Codec Port B			Enabled		
Azalia HDMI Codec Port C			Enabled		
Azalia HDMI Codec Port D			Enabled		

Azalia

Control Detection of the Azalia device.
Disabled = Azalia will be unconditionally be disabled.
Enabled = Azalia will be unconditionally be enabled.
Auto = Azalia will be enabled if present, disabled otherwise.

Azalia HDMI Codec

Enable/Disable internal HDMI codec for Azalia

USB Configuration

Main	Advanced	Chipset	Boot	Security	Save & Exit
USB Configuration					
xHCI Mode			Enabled		

xHCI Mode

Mode of operation of xHCI controller.

PCI Express Configuration

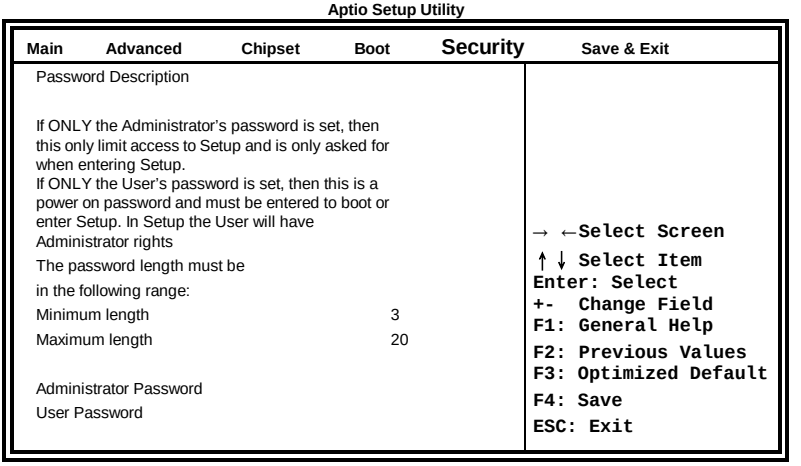
Main	Advanced	Chipset	Boot	Security	Save & Exit
PCI Express Configuration					→ ← Select Screen
					↑ ↓ Select Item
					Enter: Select
► PCI Express Root Port 1					+- Change Field
► PCI Express Root Port 2					F1: General Help
► PCI Express Root Port 3					F2: Previous Values
► PCI Express Root Port 4					F3: Optimized Default
					F4: Save
					ESC: Exit

PCI Express Configuration

PCI Express Root Port Settings.

Security Settings

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



Administrator Password

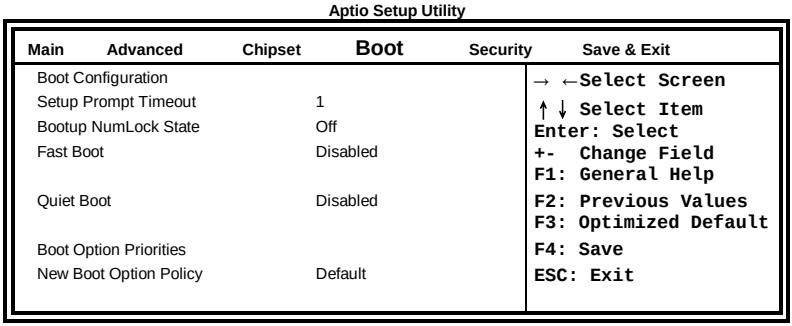
Set Setup Administrator Password.

User Password

Set User Password.

Boot Settings

This section allows you to configure the boot settings.



Setup Prompt Timeout

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

Bootup NumLock State

Select the keyboard NumLock state.

Quiet Boot

Enables/Disables Quiet Boot option.

Fast Boot

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

Save & Exit Settings

Aptio Setup Utility

Main	Advanced	Chipset	Boot	Security	Save & Exit

Save Options
Save Changes and Exit
Discard Changes and Exit

Save Changes and Reset
Discard Changes and Reset

Save Changes
Discard Changes

Defaults Options
Restore Defaults
Save as User Defaults
Restore User Defaults

Boot Override

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

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

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

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

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

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

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

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

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

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

Drivers Installation

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

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

Intel Chipset Software Installation Utility

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

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

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

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

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

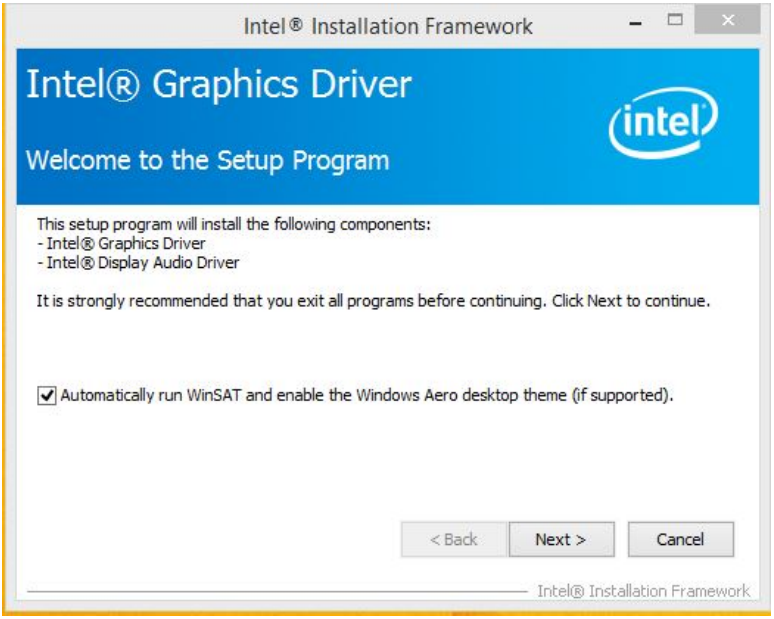
5. On the Readme File Information screen, click ***Install*** to continue the installation.

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

VGA Drivers Installation

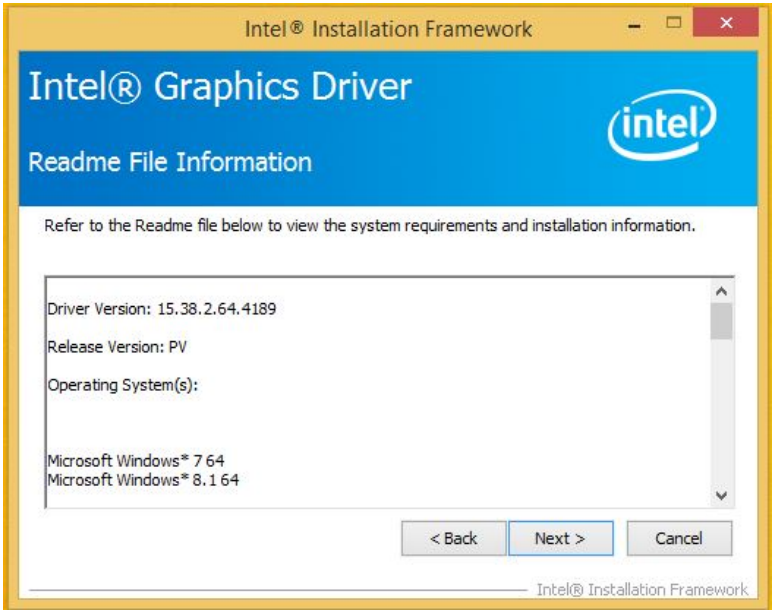
1. Click ***Intel(R) Braswell Graphics Driver***

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



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

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



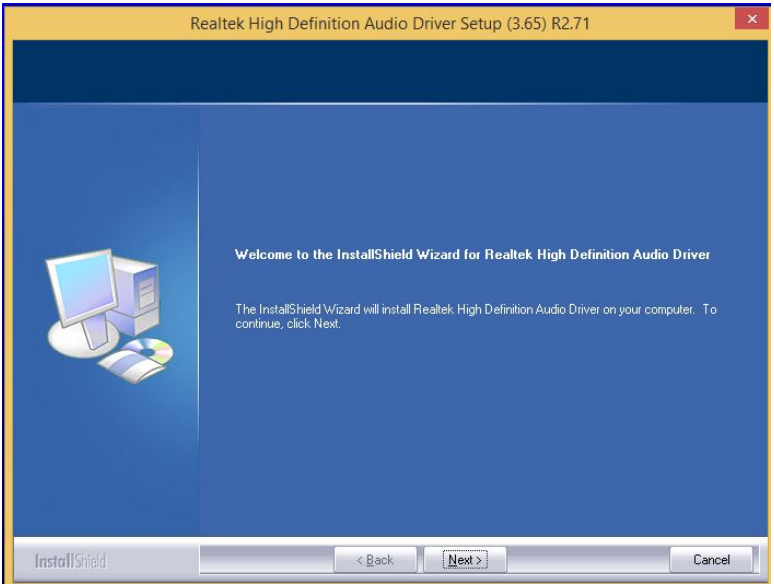
5. On Setup Progress screen, click ***Next*** to continue.

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

Realtek HD Audio Driver Installation

1. Click ***Realtek High Definition Audio Driver.***

2. On the Welcome to the InstallShield Wizard screen, click ***Next*** to proceed with and complete the installation process.



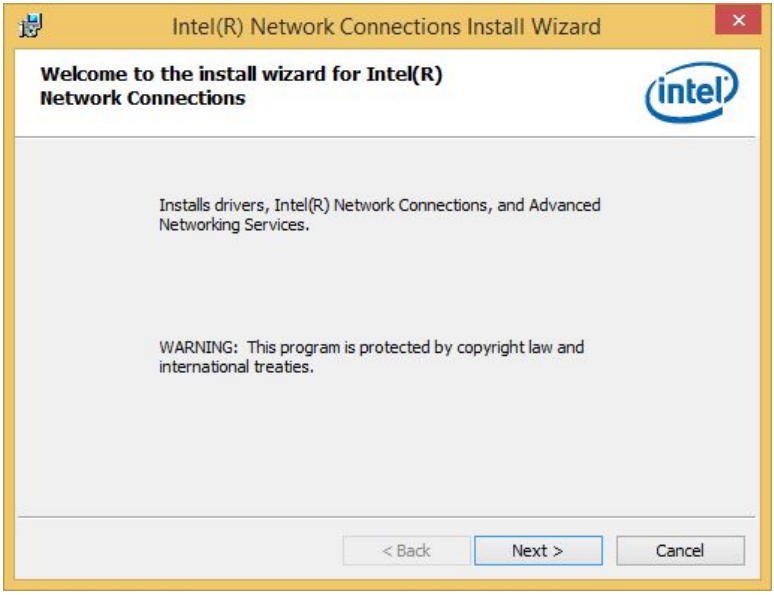
3. Restart the computer when prompted.

LAN Drivers Installation

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

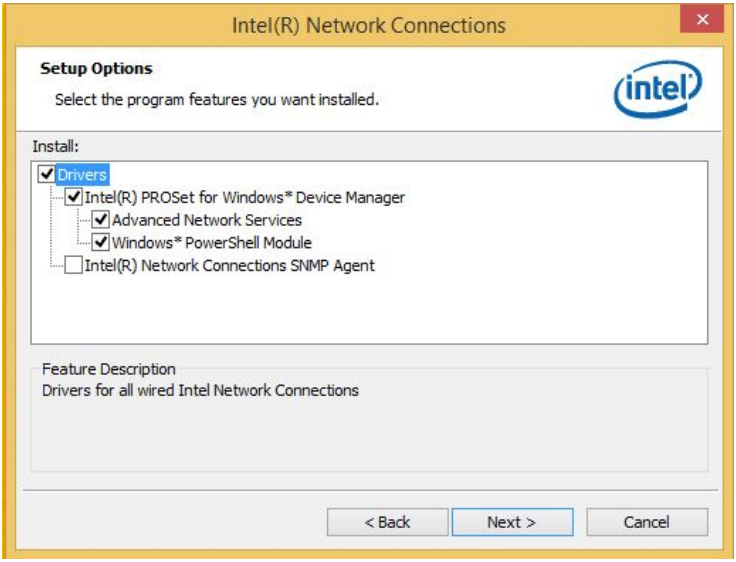
2. Click ***Intel(R) I21x Gigabit Network Drivers***

3. In the Welcome screen, click ***Next***.

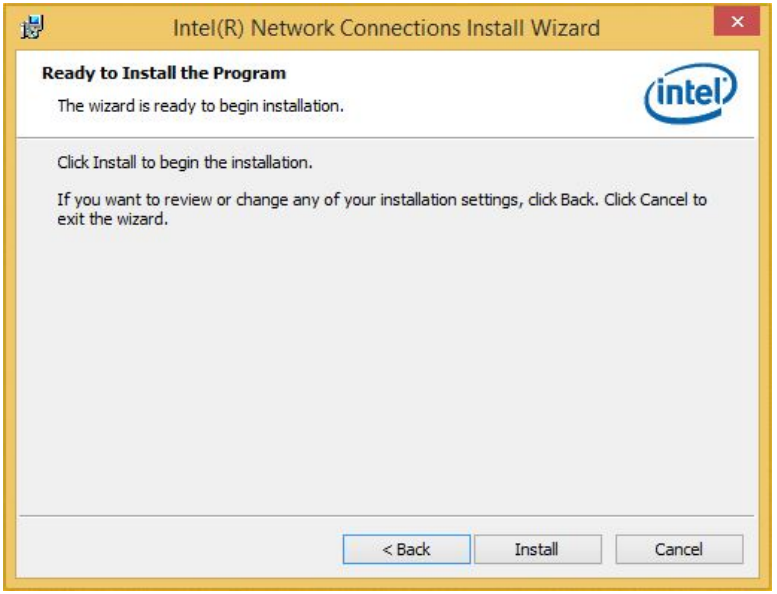


4. In the License Agreement screen, click ***I accept the terms in license agreement*** and ***Next*** to accept the software license agreement and proceed with the installation process.

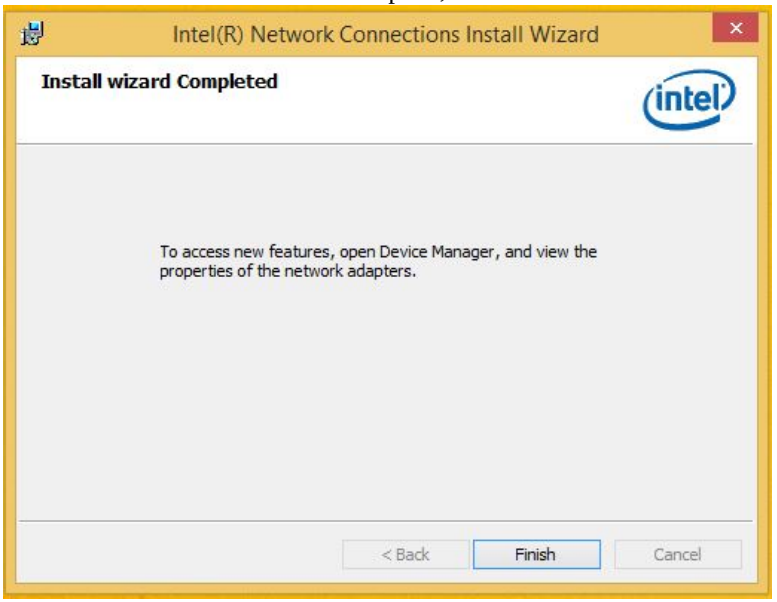
5. Click the checkbox for **Drivers** in the Setup Options screen to select it and click **Next** to continue.



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



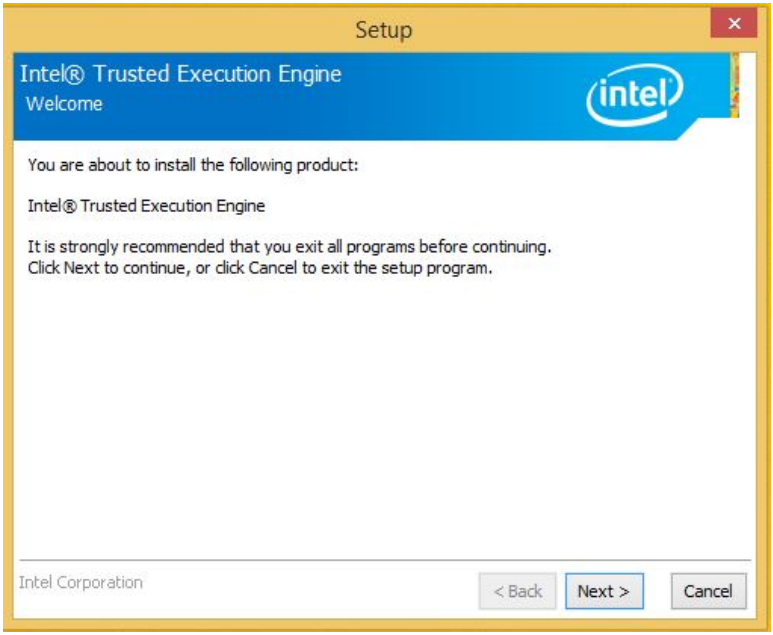
7. When InstallShield Wizard is complete, click ***Finish***.



TXE Drivers Installation

1. Click ***Intel(R) TXE Driver***

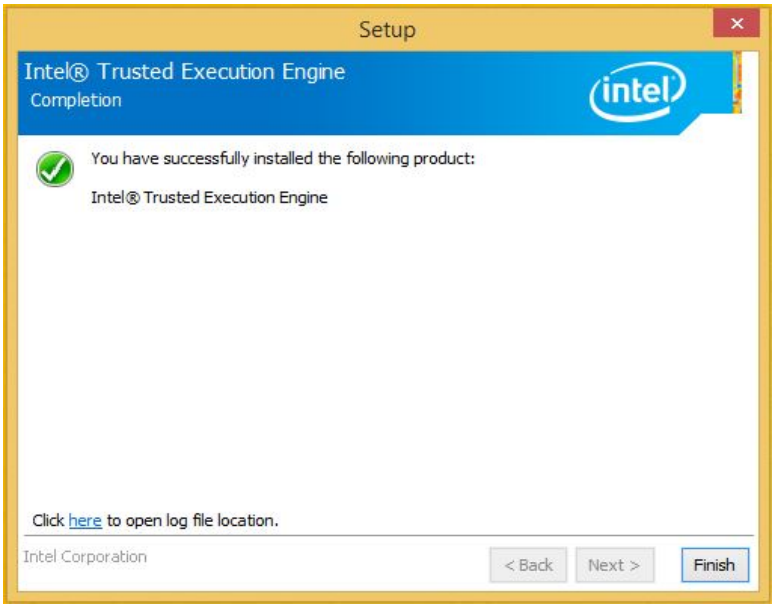
2. When the Welcome screen appears, click ***Next*** to continue.



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

4. On the Confirmation screen, click ***Next*** to continue the installation of the Intel(R) TXE Driver.

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



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
000h - 06Fh	PCI Express Root Complex
040h - 043h	System timer
070h - 077h	System CMOS/real time clock
2E8h – 2EFh	Communications Port (COM4)
2F8h – 2FFh	Communications Port (COM2)
378h – 37Fh	Printer Port (LPT1)
3B0h – 3BBh	Intel(R) HD Graphics
3E8h – 3EFh	Communications Port (COM3)
3F8h – 3FFh	Communications Port (COM1)
040h – 05Fh	Intel(R) Celeron(R)/Pentium(R) SM Bus Controller
060h – 07Fh	Standard SATA AHCI Controller

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	Standard PS/2 Keyboard
IRQ3	Communications Port (COM2)
IRQ4	Communications Port (COM1)
IRQ6	Communications Port (COM3)
IRQ10	Communications Port (COM4)
IRQ11	PS/2 Compatible Mouse
IRQ12	Intel(R) Celeron(R)/Pentium(R) SM Bus Controller
IRQ19	Standard SATA AHCI Controller

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:

```
//---------------------------------------------------------------------------
#include <dos.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include "F81866.H"
//---------------------------------------------------------------------------
int main (int argc, char *argv[]);
void EnableWDT(int);
void DisableWDT(void);
//---------------------------------------------------------------------------
int main (int argc, char *argv[])
{
        unsigned char bBuf;
        unsigned char bTime;
        char **endptr;

        char SIO;

        printf("Fintek 81866 watch dog program\n");

        SIO = Init_F81866();
        if (SIO == 0)
        {
                printf("Can not detect Fintek 81866, program abort.\n");
                return(1);
        }//if (SIO == 0)

        if (argc != 2)
        {
                printf(" Parameter incorrect!!\n");
                return (1);
        }

        bTime = strtol (argv[1], endptr, 10);
        printf("System will reset after %d seconds\n", bTime);

        if (bTime)
        {       EnableWDT(bTime); }
        else
        {       DisableWDT();       }

        return 0;
}
//---------------------------------------------------------------------------
void EnableWDT(int interval)
{
        unsigned char bBuf;
```

```
        bBuf = Get_F81866_Reg(0x2B);
        bBuf &= (~0x20);
        Set_F81866_Reg(0x2B, bBuf);                     //Enable WDTO

        Set_F81866_LD(0x07);                            //switch to logic device 7
        Set_F81866_Reg(0x30, 0x01);                     //enable timer

        bBuf = Get_F81866_Reg(0xF5);
        bBuf &= (~0x0F);
        bBuf |= 0x52;
        Set_F81866_Reg(0xF5, bBuf);                     //count mode is second

        Set_F81866_Reg(0xF6, interval);                 //set timer

        bBuf = Get_F81866_Reg(0xFA);
        bBuf |= 0x01;
        Set_F81866_Reg(0xFA, bBuf);                     //enable WDTO output

        bBuf = Get_F81866_Reg(0xF5);
        bBuf |= 0x20;
        Set_F81866_Reg(0xF5, bBuf);                     //start counting
}
//-----------------------------------------------------------------------------
void DisableWDT(void)
{
        unsigned char bBuf;

        Set_F81866_LD(0x07);                            //switch to logic device 7

        bBuf = Get_F81866_Reg(0xFA);
        bBuf &= ~0x01;
        Set_F81866_Reg(0xFA, bBuf);                     //disable WDTO output

        bBuf = Get_F81866_Reg(0xF5);
        bBuf &= ~0x20;
        bBuf |= 0x40;
        Set_F81866_Reg(0xF5, bBuf);                     //disable WDT
}
//-----------------------------------------------------------------------------
```

```
//---------------------------------------------------------------------------
#include "F81866.H"
#include <dos.h>
//---------------------------------------------------------------------------
unsigned int F81866_BASE;
void Unlock_F81866 (void);
void Lock_F81866 (void);
//---------------------------------------------------------------------------
unsigned int Init_F81866(void)
{
	unsigned int result;
	unsigned char ucDid;

	F81866_BASE = 0x4E;
	result = F81866_BASE;

	ucDid = Get_F81866_Reg(0x20);
	if (ucDid == 0x10)                                  //Fintek 81866
	{	goto Init_Finish;	}

	F81866_BASE = 0x2E;
	result = F81866_BASE;

	ucDid = Get_F81866_Reg(0x20);
	if (ucDid == 0x10)                                  //Fintek 81866
	{	goto Init_Finish;	}

	F81866_BASE = 0x00;
	result = F81866_BASE;

Init_Finish:
	return (result);
}
//---------------------------------------------------------------------------
void Unlock_F81866 (void)
{
	outportb(F81866_INDEX_PORT, F81866_UNLOCK);
	outportb(F81866_INDEX_PORT, F81866_UNLOCK);
}
//---------------------------------------------------------------------------
void Lock_F81866 (void)
{
	outportb(F81866_INDEX_PORT, F81866_LOCK);
}
//---------------------------------------------------------------------------
void Set_F81866_LD( unsigned char LD)
{
	Unlock_F81866();
	outportb(F81866_INDEX_PORT, F81866_REG_LD);
	outportb(F81866_DATA_PORT, LD);
	Lock_F81866();
}
//---------------------------------------------------------------------------
void Set_F81866_Reg( unsigned char REG, unsigned char DATA)
{
	Unlock_F81866();
	outportb(F81866_INDEX_PORT, REG);
	outportb(F81866_DATA_PORT, DATA);
	Lock_F81866();
}
//---------------------------------------------------------------------------
unsigned char Get_F81866_Reg(unsigned char REG)
{
	unsigned char Result;
	Unlock_F81866();
	outportb(F81866_INDEX_PORT, REG);
	Result = inportb(F81866_DATA_PORT);
	Lock_F81866();
	return Result;
```

```
}
//---------------------------------------------------------------------------


//---------------------------------------------------------------------------
#ifndef __F81866_H
#define __F81866_H                      1
//---------------------------------------------------------------------------
#define         F81866_INDEX_PORT               (F81866_BASE)
#define         F81866_DATA_PORT                (F81866_BASE+1)
//---------------------------------------------------------------------------
#define         F81866_REG_LD                   0x07
//---------------------------------------------------------------------------
#define F81866_UNLOCK                   0x87
#define         F81866_LOCK                             0xAA
//---------------------------------------------------------------------------
unsigned int Init_F81866(void);
void Set_F81866_LD( unsigned char);
void Set_F81866_Reg( unsigned char, unsigned char);
unsigned char Get_F81866_Reg( unsigned char);
//---------------------------------------------------------------------------
#endif  //__F81866_H
```

D. Digital I/O Sample Code

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
//rev date    name            description
//---------------------------------------------------------------------------
//s01 20091028 Sunny     set gpio multi-function pin
//---------------------------------------------------------------------------
#include <dos.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include "F81865.H"

#define BIT0 0x01
#define BIT1 0x02
#define BIT2 0x04
#define BIT3 0x08
#define BIT4 0x10
#define BIT5 0x20
#define BIT6 0x40
#define BIT7 0x80

//---------------------------------------------------------------------------
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;
      unsigned char DIO;

      printf("Fintek 81865/81866 digital I/O test program\n");

      SIO = Init_F81865();
      if (SIO == 0)
      {
         printf("Can not detect Fintek 81865/81866, program abort.\n");
         return(1);
      }//if (SIO == 0)

      Dio5Initial();
/*
      //for GPIO50..57
      Dio5SetDirection(0xF0);   //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);
*/

	//for GPIO50..57
	Dio5SetDirection(0xF0);  //GP50..53 = input, GP54..57=output

	Dio5SetOutput(0x00);			//clear
//	DIO = Dio5GetInput() & 0x0F;

	Dio5SetOutput(0x00);			//clear
	DIO = Dio5GetInput() & 0x0F;
	if (DIO != 0x0A)
	{
	   printf("The Fintek 81865 digital IO abnormal, abort.\n");
	   return(1);
	}//if (DIO != 0x0A)

	Dio5SetOutput(0xA0);			//clr# is high
	Dio5SetOutput(0xF0);			//clk and clr# is high
	Dio5SetOutput(0xA0);			//clr# is high

	DIO = Dio5GetInput() & 0x0F;
	if (DIO != 0x05)
	{
	   printf("The Fintek 81865 digital IO abnormal, abort.\n");
	   return(1);
	}
	printf("!!! Pass !!!\n");
	return 0;
}
//---------------------------------------------------------------------------
void Dio5Initial(void)
{
	unsigned char ucBuf;

//iBase-Sunny20091028[s01] start >>
	//switch GPIO multi-function pin for    gpio 50~57

//gpio53~57 UR5_FULL_EN(bit1), clear UR6_FULL_EN(bit3)
	//set UR5_FULL_EN,should set UR_GP_PROG_EN = 1 (reg26,bit0) first

	ucBuf = Get_F81865_Reg(0x26);
	ucBuf |= BIT0;
	Set_F81865_Reg(0x26, ucBuf);
	//set UR5_FULL_EN(bit1), clear UR6_FULL_EN(bit3)
	ucBuf = Get_F81865_Reg(0x2A);
	ucBuf &= ~BIT3;//clear bit 3,
	ucBuf |= BIT1;//set bit 1,
	Set_F81865_Reg(0x2a, ucBuf);

//GPIO51 ~ GPIO52
	//clear UR6_ALT_EN(bit5),	IR_ALT_EN(bit4),set FDC_GP_EN(bit3)

//GPIO50
//set FDC_GP_EN(bit3), clear  RTS6_ALT_EN(RTS6_2_ALT_EN)(bit6)

	ucBuf = Get_F81865_Reg(0x2A);
	ucBuf &= ~(BIT4+BIT5+BIT6);		//clear UR6_ALT_EN(bit5),	IR_ALT_EN(bit4),
RTS6_ALT_EN(RTS6_2_ALT_EN)(bit6)
	Set_F81865_Reg(0x2a, ucBuf);
```

```
        //set FDC_GP_EN(bit3), should clear UR_GP_PROG_EN (reg26,bit0) first
        ucBuf = Get_F81865_Reg(0x26);
        ucBuf &= ~BIT0;
        Set_F81865_Reg(0x26, ucBuf);//clear UR_GP_PROG_EN = 0 (reg26,bit0)

        ucBuf = Get_F81865_Reg(0x2A);
        ucBuf |= BIT3;          //set FDC_GP_EN(bit3),
        Set_F81865_Reg(0x2a, ucBuf);

//<<iBase-Sunny20091028[s01] end
        Set_F81865_LD(0x06);                                    //switch to logic device 6

        //enable the GP5 group
        ucBuf = Get_F81865_Reg(0x30);
        ucBuf |= 0x01;
        Set_F81865_Reg(0x30, ucBuf);

        Set_F81865_Reg(0xA0, 0x00);                             //define as input mode
        Set_F81865_Reg(0xA3, 0xFF);                             //push pull mode
}
//-----------------------------------------------------------------------------
void Dio5SetOutput(unsigned char NewData)
{
        Set_F81865_LD(0x06);                                    //switch to logic device 6
        Set_F81865_Reg(0xA1, NewData);
}
//-----------------------------------------------------------------------------
unsigned char Dio5GetInput(void)
{
        unsigned char result;

        Set_F81865_LD(0x06);                                    //switch to logic device 6
        result = Get_F81865_Reg(0xA2);
        return (result);
}
//-----------------------------------------------------------------------------
void Dio5SetDirection(unsigned char NewData)
{
        //NewData : 1 for input, 0 for output
        Set_F81865_LD(0x06);                                    //switch to logic device 6
        Set_F81865_Reg(0xA0, NewData);
}
//-----------------------------------------------------------------------------
unsigned char Dio5GetDirection(void)
{
        unsigned char result;

        Set_F81865_LD(0x06);                                    //switch to logic device 6
        result = Get_F81865_Reg(0xA0);
        return (result);
}
//-----------------------------------------------------------------------------
```