IB792

VIA CN700

5.25-inch SBC

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

The IB792 5.25-inch SBC incorporates the VIA CN700 chipset. Currently, the board is available in three models, namely:

Model	VIA Eden-C7 CPU	LAN Bypass	Watchdog Timer
IB792F	1.5GHz	Yes	Yes
IB792	1.5GHz	No	Yes

IB792 Features

- Supports 4 Realtek 10/100/1G LAN ports
- Supports 1.5GHz~2GHz VIA Eden-C7 processors
- DDR2 SO-DIMM x 1, up to 1GB
- Mini-PCI slot, Compact Flash socket
- Optional Hardware LAN Bypass function on ETH 0 & 1

Checklist

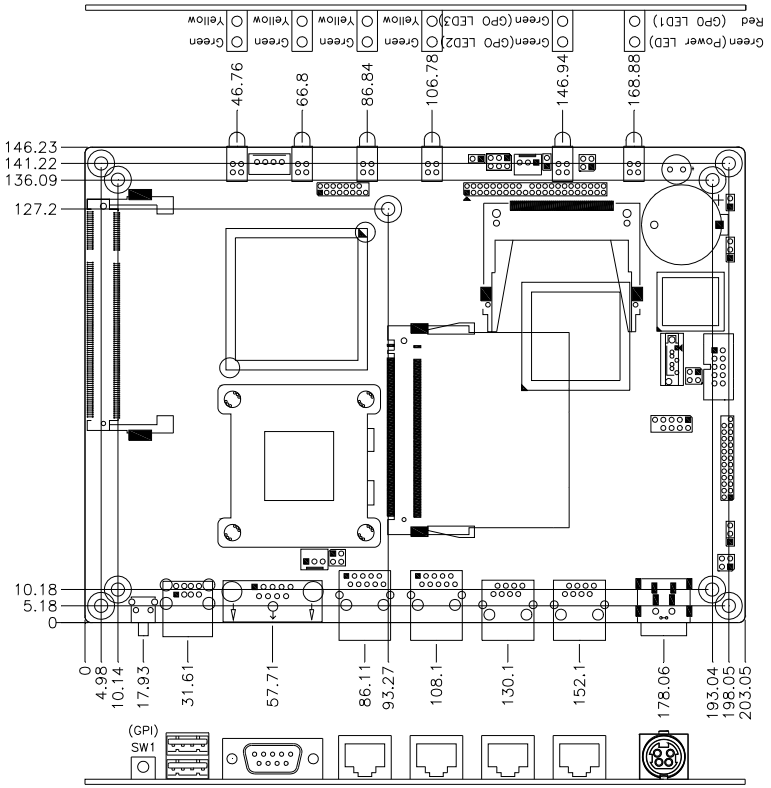
Your IB792 package should include the items listed below.

- The IB792 VIA Eden-C7 motherboard
- This User's Manual
- 1 CD containing chipset drivers and flash memory utility
- Optional cables for VGA, KB/MS, Serial port, Serial ATA

Specifications

CPU Type	Via C7 BGA
CPU Voltage	C7: 1.5~2.0GHz (heatsink w/o fan.) fanless
CPU Frequency / FSB	1.004V ~ 1.148V
CPU FSB	400/533MHz (800MHz to be announced)
Cache	128KB L2 (CPU integrated)
Green / APM	APM1.2
Chipset	VIA CN700/8237R+ chipset North bridge: CN700 567-pin HSBGA South bridge: VT8237R+ 539-pin PBGA
BIOS	Award BIOS supports ACPI function
Memory	One DDRII SO-DIMM socket, supports up to 1GB
VGA	CN700 built-in 128-bit Unichrome Pro 3D/2D controller
LAN Gigabit LAN	Realtek RTL8110SC Gigabit LAN controller x 4 Bypass function on LAN1 & 2
USB	VT8237R+ built-in USB 2.0, 2 ports supported
Serial ATA	VT8237R+ built-in SATA II controller, supports 2 ports w/ RAID 0, 1, use 1 port only
IDE Interface	Two channels; supports Ultra DMA 33/66/100/133 IDE1: 44pin header for HDD IDE2: Compact Flash
Audio	Onboard buzzer only
LPC I/O	Winbond W83697HG: COM1, COM2 (RS-232), Parallel port & hardware monitor (2 thermal inputs, 5 voltage monitor inputs, 1 chassis open detection & 2 fan headers). IrDA & FDD are not used
RTC/CMOS	VT8237R+ built-in RTC with on board Lithium Battery
KB/Mouse	Supports PS/2 Keyboard/Mouse
Expansion	Mini PCI socket x1 Compact Flash socket x1
Edge Connectors	DC-In power connector x1 for +5V DC in RJ45 x4 for LAN 1~4 DB9 connector x 1 for COM1 Dual USB stack connector x 1 USB 1 & 2 Button x1 for GPI
On Board Connector / Header	16 pins pin-header x 1 for VGA Serial ATA connector x 1 44 pins pin-header x 1 for IDE1 Compact Flash socket x1 on IDE2 bus 10 pins pin-header x 1 for COM2 10 pins pin-header for PS/2 KB/Mouse 26Pin pin- header (2.00mm) for printer connection
LED Indicator	4x green/yellow stack LED for LAN status (3 Colors LEDs for 10/100/1000 LAN speed) Orange-Green-Yellow LEDs. 1x green LED for power status 2x green and 1x red led stack for GPO status
Power	+5V DC input
Watchdog Timer	Yes (256 segments: 0, 1, 2,..., 255 sec/min)
Digital I/O	5 programmed In/Out
Board Size	203mm x 146mm (5.25” SBC)

Board Dimensions



Installations

This section provides information on how to use the jumpers and connectors on the IB792 in order to set up a workable system. The topics covered are:

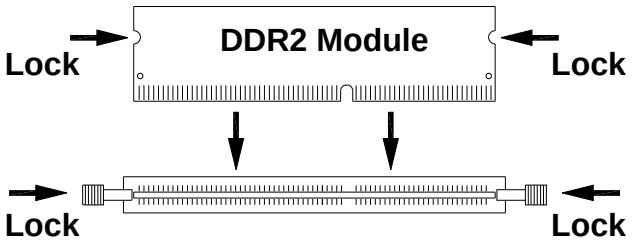
Installing the Memory

The IB792 board supports one DDR2 memory socket for a maximum total memory of 1GB in DDR2 memory type.

Installing and Removing Memory Modules

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

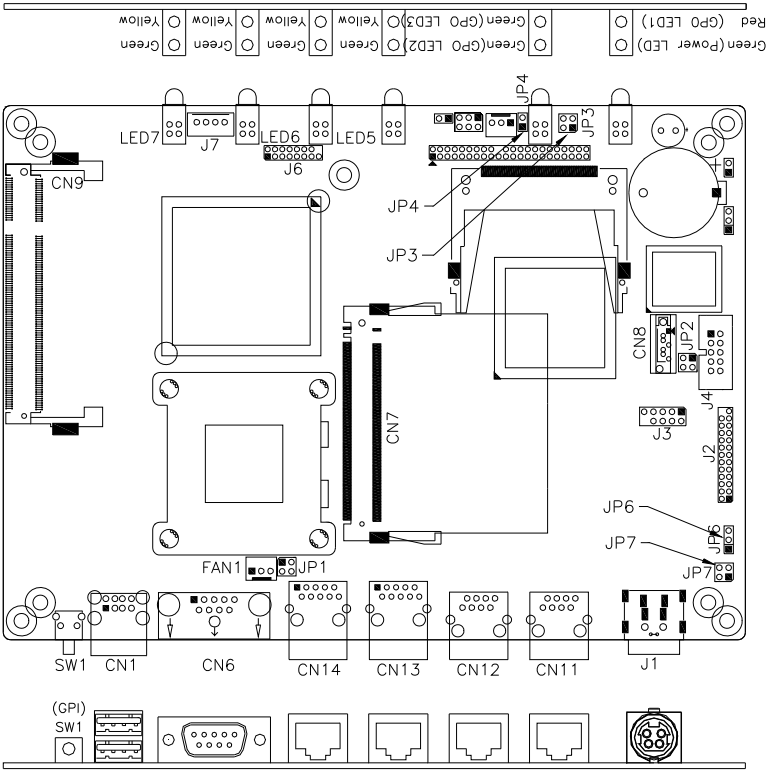
1. Hold the DDR2 module so that the key of the DDR2 module aligns with those on the memory slot.
2. Gently push the DDR2 module in an upright position until the clips of the slot close to hold the DDR2 module in place when the DDR2 module touches the bottom of the slot.
3. To remove the DDR2 module, press the clips with both hands.



Setting the Jumpers

Jumpers are used on IB792 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 IB792 and their respective functions.

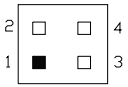
Jumper Locations on IB792



JBAT1: Clear CMOS Setting

JBAT1	Setting
	Normal
	Clear CMOS

JP3: HDD UDMA Selection

JP3	Setting
	1-2 short: BIOS detect
	1-3 short: Upper UDMA 33
	2-4 short: UDMA33 only (default)

JP4: CF Card Master / Slave Selection

JP4	CF Card Setting
	Master
	Slave

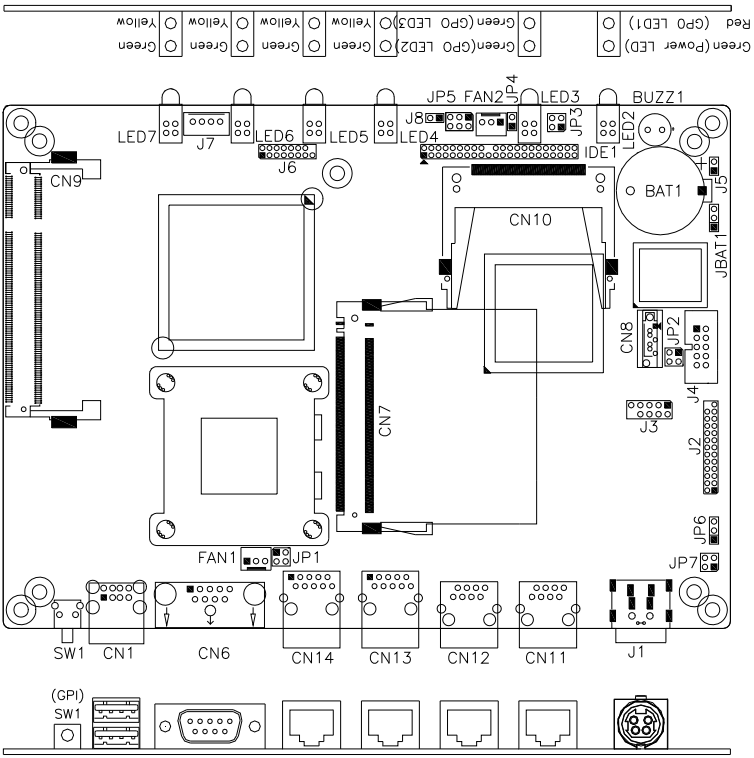
JP6, JP7: Watchdog Timer & LAN1/LAN2 Bypass Settings

JP6, JP7 Setting	Function
JP7 Pin 1-2 & 3-4 Open JP6 Pin 1-2 Closed	System will bypass LANs upon the time out of watchdog timer.
JP7 Pin 1-2 & 3-4 Open JP6 Pin 2-3 Closed	System LANs bypass function controlled by SIO GPIO50.
JP7 Pin 1-2 & 3-4 Closed JP6 Pin 1-2 Closed (Default)	System will reboot upon the time out of watchdog timer.

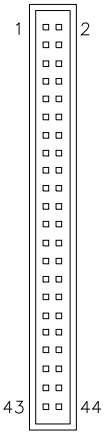
Connectors on IB792

The connectors on IB792 allows you to connect external devices such as keyboard, floppy disk drives, hard disk drives, printers, etc. The following table lists the connectors on IB792 and their respective functions.

Connector Locations on IB792

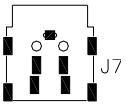


IDE1: Primary IDE Connector

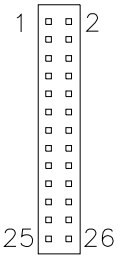


Signal Name	Pin #	Pin #	Signal Name
Reset IDE	1	2	Ground
Host data 7	3	4	Host data 8
Host data 6	5	6	Host data 9
Host data 5	7	8	Host data 10
Host data 4	9	10	Host data 11
Host data 3	11	12	Host data 12
Host data 2	13	14	Host data 13
Host data 1	15	16	Host data 14
Host data 0	17	18	Host data 15
Ground	19	20	Key
DRQ0	21	22	Ground
Host IOW	23	24	Ground
Host IOR	25	26	Ground
IOCHRDY	27	28	Host ALE
DACK0	29	30	Ground
IRQ14	31	32	No connect
Address 1	33	34	No connect
Address 0	35	36	Address 2
Chip select 0	37	38	Chip select 1
Activity	39	40	Ground
Vcc	41	42	Vcc
Ground	43	44	N.C.

J1: Power Connector (5V only)



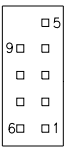
J2: Parallel Port Connector



Signal Name	Pin #	Pin #	Signal Name
Line printer strobe	1	2	AutoFeed
PD0, parallel data 0	3	4	Error
PD1, parallel data 1	5	6	Initialize
PD2, parallel data 2	7	8	Select
PD3, parallel data 3	9	10	Ground
PD4, parallel data 4	11	12	Ground
PD5, parallel data 5	13	14	Ground
PD6, parallel data 6	15	16	Ground
PD7, parallel data 7	17	18	Ground
ACK, acknowledge	19	20	Ground
Busy	21	22	Ground
Paper empty	23	24	Ground
Select	25	N/A	N/A

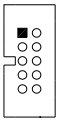
J3: PS/2 Keyboard and Mouse Connector

J3, a 10-pin header, has functions for both keyboard and mouse.



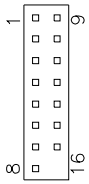
Signal Name	Pin #	Pin #	Signal Name
N.C.	10	5	N.C.
KB clock	9	4	Mouse clock
KB data	8	3	Mouse data
Vcc	7	2	Vcc
Ground	6	1	Ground

J4: COM2 Serial Port



Pin #	Signal Name
1	DCD, Data carrier detect
2	RXD, Receive data
3	TXD, Transmit data
4	DTR, Data terminal ready
5	Ground
6	DSR, Data set ready
7	RTS, Request to send
8	CTS, Clear to send
9	RI, Ring indicator

J6: VGA CRT Connector



Signal Name	Pin	Pin	Signal Name
R	1	9	VCC
G	2	10	GND
B	3	11	NC
NC	4	12	DDCDAT
GND	5	13	HSYNC
GND	6	14	VSYNC
GND	7	15	DDCCLK
GND	8	16	Protect pin

J7: External Power Supply Connector

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

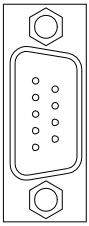
J8: Reset Switch Pin Header

CN1: USB STACK CONNECTOR 1/2 Ports

CN11, CN12, CN13, CN14: RJ45 LAN Connectors

These four LAN (LAN1/2/3/4) connectors are used in conjunction with the four Realtek RTL8110SC Ethernet controllers on the board.

CN6: COM1 Serial Port



Pin #	Signal Name
1	DCD, Data carrier detect
2	RXD, Receive data
3	TXD, Transmit data
4	DTR, Data terminal ready
5	Ground
6	DSR, Data set ready
7	RTS, Request to send
8	CTS, Clear to send
9	RI, Ring indicator

CN7: Mini PCI Connector

Pin #	Signal	Pin #	Signal	Pin #	Signal	Pin #	Signal
1	NC	2	NC	63	+3.3V	64	FRAME#
3	NC	4	NC	65	CLKRUN#	66	TRDY#
5	NC	6	NC	67	SERR#	68	STOP#
7	NC	8	NC	69	Gnd	70	+3.3V
9	NC	10	NC	71	PERR#	72	DEVSEL#
11	NC	12	NC	73	C/BE[1]	74	Gnd
13	NC	14	NC	75	AD[14]	76	AD[15]
15	GND	16	EX_INTC#	77	GND	78	AD[13]
17	INTB#	18	+5V	79	AD[12]	80	AD[11]
19	+3.3V	20	INTA#	81	AD[10]	82	GND
21	NC	22	NC	83	GND	84	AD[9]
23	GND	24	+3.3VS	85	AD[8]	86	C/BE[0]
25	CLK	26	RESET#	87	AD[7]	88	+3.3V
27	GND	28	+3.3V	89	+3.3V	90	AD[6]
29	REQ#	30	GNT#	91	AD[5]	92	AD[4]
31	+3.3V	32	GND	93	EX_GNT#	94	AD[2]
33	AD[31]	34	PME#	95	AD[3]	96	AD[0]
35	AD[29]	36	NC	97	+5V	98	NC
37	GND	38	AD[30]	99	AD[1]	100	NC
39	AD[27]	40	+3.3V	101	GND	102	GND
41	AD[25]	42	AD[28]	103	AC_SYNC	104	GND
43	EX_IDSEL#	44	AD[26]	105	AC_SDIN	106	AC_SDOUT
45	C/BE[3]	46	AD[24]	107	AC_BITCLK	108	NC
47	AD[23]	48	IDSEL#	109	NC	110	AC_RST#
49	GND	50	GND	111	NC	112	NC
51	AD[21]	52	AD[22]	113	GNC	114	GND
53	AD[19]	54	AD[20]	115	NC	116	NC
55	GND	56	PAR	117	GND	118	GND
57	AD[17]	58	AD[18]	119	GND	120	GND
59	C/BE[2]	60	AD[16]	121	NC	122	NC
61	IRDY#	62	GND	123	+5V	124	+3.3VS

CN8: Serial ATA Connector

CN9: DDR2 SO-DIMM SOCKET

CN10: Compact Flash Socket

LED2: STATE LED
Upper LED: PWROK LED
Down LED: HDD or GPO_11 LED

LED3: STATE LED
Upper LED: GPO_12 LED
Down LED: GPO_13 LED

LED4, LED5, LED6, LED7: RJ45 LAN Link, Active LED

SW1: GPI Switch Button (GPI_10)

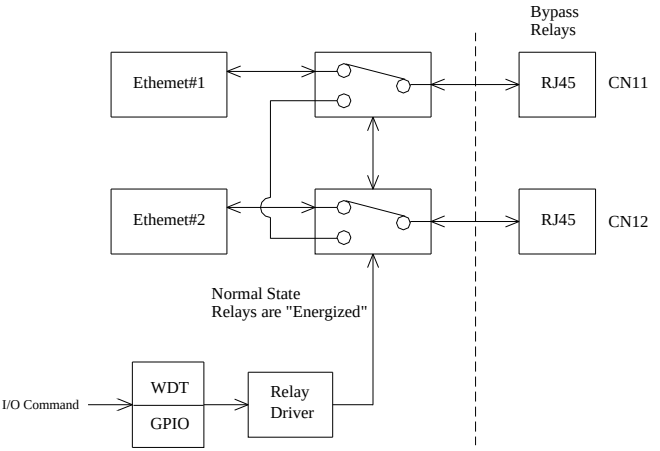
JP5: GPO Control

Signal Name	Pin #	Pin #	Signal Name
GPO_14	6	5	GPO_16
GPO_15	4	3	GPO_17
Vcc	2	1	Ground

Bypass and WDT

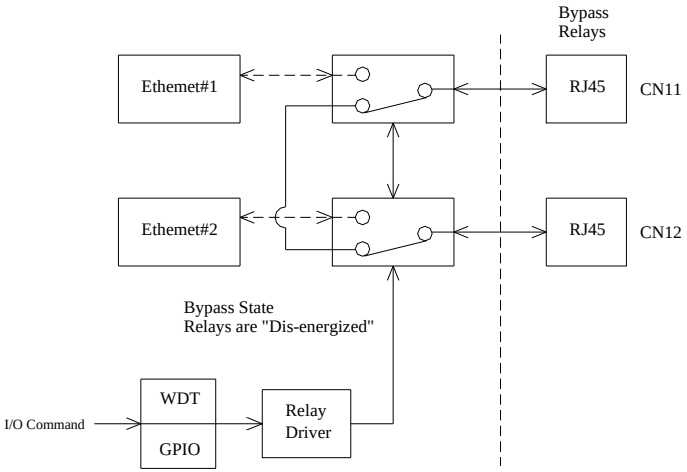
The bypass function is used to link (or short) two independent Ethernet ports when user's application software halt or when power is off.

Block Diagram:



Communication States:
There are two communications states for the bypass function: (1) Normal State, (2) Bypass State. A watchdog timer (WDT) or a GPIO are used to control and switch the communication between the two states.

The block diagram in the section above shows the Normal State, where the two Ethernet ports work independently. The following diagram shows the Bypass State, where the two Ethernet ports are bypassed together.



GPIO Function Definition

The IB792 has defined some of the GPI function to satisfy various requirements. Here are the definitions for the GPIO function:

GPI_10: button for customer defined (SW1)
GPO_11~GPO_13: Status LED (LED2,LED3)
GPO50: Ethernet bypass function
GPO_14~GPO_17: Customer defined (JP5)

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 sort of circumstances, the timer will count to zero and the selected outputs will be driven. Under normal circumstance, the user will restart the WDT at regular intervals before the timer counts to zero.

SAMPLE CODE:
This code and information is provided "as is" without warranty of any kind, either expressed or implied, including but not limited to the implied warranties of merchantability and/or fitness for a particular purpose.

Filename：Main.cpp

```
//============================================================================
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//============================================================================
#include <stdio.h>
#include <stdlib.h>
#include "W697HF.H"
//============================================================================
int main (int argc, char *argv[]);
void copyright(void);
void EnableWDT(int);
void DisableWDT(void);
//============================================================================
int main (int argc, char *argv[])
{
	unsigned char bBuf;
	unsigned char bTime;
	char **endptr;

	copyright();

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

	if (Init_W697HF() == 0)
	{
		printf(" Winbond 83697HF is not detected, program abort.\n");
		return 1;
	}
	bTime = strtol (argv[1], endptr, 10);
	printf("System will reset after %d seconds\n", bTime);

	EnableWDT(bTime);

	return 0;
}
//============================================================================
void copyright(void)
{
	printf("\n======== Winbond 697HF Watch Timer Tester (AUTO DETECT) ========\n"\
			"	Usage : W697WD reset_time\n"\
			"	Ex : W697WD 3  => reset system after 3 second\n"\
			"	     W697WD 0  => disable watch dog timer\n");
}
//============================================================================
void EnableWDT(int interval)
```

```
{
        unsigned char bBuf;

        bBuf = Get_W697HF_Reg(0x29);
        bBuf &= (~0x60);
        bBuf |= 0x20;
        Set_W697HF_Reg(0x29, bBuf);                     //enable WDTO

        Set_W697HF_LD(0x08);                            //switch to logic device 8

        bBuf = Get_W697HF_Reg(0xF3);
        bBuf &= (~0x04);
        Set_W697HF_Reg( 0xF3, bBuf);                    //count mode is second

        Set_W697HF_Reg( 0xF4, interval);                //set timer
        Set_W697HF_Reg( 0x30, 0x01);                    //enable timer
}
//=====================================================================
void DisableWDT(void)
{
        Set_W697HF_LD(0x08);                            //switch to logic device 8
        Set_W697HF_Reg(0x30, 0x00);                     //watchdog disabled
        Set_W697HF_Reg(0xF4, 0x00);                     //clear watchdog timer
}
//=====================================================================
```

Filename：W697hf.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 "W697HF.H"
#include <dos.h>
//=====================================================================================
unsigned int W697HF_BASE;
void Unlock_W697HF (void);
void Lock_W697HF (void);
//=====================================================================================
unsigned int Init_W697HF(void)
{
        unsigned int result;
        unsigned char ucDid;

        W697HF_BASE = 0x2E;
        result = W697HF_BASE;

        ucDid = Get_W697HF_Reg(0x20);
        if ( ucDid == 0x60)
        {       goto Init_Finish;       }

        W697HF_BASE = 0x4E;
        result = W697HF_BASE;

        ucDid = Get_W697HF_Reg(0x20);
        if ( ucDid == 0x60)
        {       goto Init_Finish;       }

        W697HF_BASE = 0x00;
        result = W697HF_BASE;

Init_Finish:
        return (result);
}
//=====================================================================================
void Unlock_W697HF (void)
{
        outportb(W697HF_INDEX_PORT, W697HF_UNLOCK);
        outportb(W697HF_INDEX_PORT, W697HF_UNLOCK);
```

```
}
//===========================================================================
void Lock_W697HF (void)
{
        outportb(W697HF_INDEX_PORT, W697HF_LOCK);
}
//===========================================================================
void Set_W697HF_LD( unsigned char LD)
{
        Unlock_W697HF();
        outportb(W697HF_INDEX_PORT, W697HF_REG_LD);
        outportb(W697HF_DATA_PORT, LD);
        Lock_W697HF();
}
//===========================================================================
void Set_W697HF_Reg( unsigned char REG, unsigned char DATA)
{
        Unlock_W697HF();
        outportb(W697HF_INDEX_PORT, REG);
        outportb(W697HF_DATA_PORT, DATA);
        Lock_W697HF();
}
//===========================================================================
unsigned char Get_W697HF_Reg(unsigned char REG)
{
        unsigned char Result;
        Unlock_W697HF();
        outportb(W697HF_INDEX_PORT, REG);
        Result = inportb(W697HF_DATA_PORT);
        Lock_W697HF();
        return Result;
}
//===========================================================================
```

Filename：W697hf.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 __W697HF_H
#define __W697HF_H                      1
//===========================================================================
#define         W697HF_INDEX_PORT               (W697HF_BASE)
#define         W697HF_DATA_PORT                (W697HF_BASE+1)
//===========================================================================
#define         W697HF_REG_LD                   0x07
//===========================================================================
#define W697HF_UNLOCK                   0x87
#define         W697HF_LOCK                             0xAA
//===========================================================================
unsigned int Init_W697HF(void);
void Set_W697HF_LD( unsigned char);
void Set_W697HF_Reg( unsigned char, unsigned char);
unsigned char Get_W697HF_Reg( unsigned char);
//===========================================================================
#endif  //__W697HF_H
```

Digital I/O Sample Configuration

Filename：Main.cpp

```
//---------------------------------------------------------------------------
//
// THIS CODE AND INFORMATION IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY
// KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
// IMPLIED WARRANTIES OF MERCHANTABILITY AND/OR FITNESS FOR A PARTICULAR
// PURPOSE.
//
//---------------------------------------------------------------------------
#include <dos.h>
#include <conio.h>
#include <stdio.h>
#include <stdlib.h>
#include "W697HF.H"
//---------------------------------------------------------------------------
void ClrKbBuf(void);
int main (int argc, char *argv[]);

void SetDio1Output(unsigned char);
unsigned char GetDio1Input(void);
void SetDio1Direction(unsigned char);
unsigned char GetDio1Direction(void);

void SetDio5Output(unsigned char);
unsigned char GetDio5Input(void);
void SetDio5Direction(unsigned char);
unsigned char GetDio5Direction(void);

//---------------------------------------------------------------------------
int main (int argc, char *argv[])
{
                                if (Init_W697HF() == 0)
                                {
                                                printf("Can not detect Winbond 83697HF, program abort.\n");
                                                return(1);
                                }

                                //for GPIO10..17
                                SetDio1Direction(0x0F);                 //GP10..13 = input, GP14..17=output
                                printf("Current DIO direction = 0x%X\n", GetDio1Direction());

                                printf("Current DIO status = 0x%X\n", GetDio1Input());

                                printf("Set DIO output to high\n");
                                SetDio1Output(0x0F);

                                printf("Set DIO output to low\n");
                                SetDio1Output(0x00);

                                //for GPIO50..57
                                SetDio5Direction(0x0F);                 //GP50..53 = input, GP54..57=output
                                printf("Current DIO direction = 0x%X\n", GetDio5Direction());

                                printf("Current DIO status = 0x%X\n", GetDio5Input());

                                printf("Set DIO output to high\n");
                                SetDio5Output(0x0F);

                                printf("Set DIO output to low\n");
                                SetDio5Output(0x00);

                                return 0;
}
//---------------------------------------------------------------------------
void ClrKbBuf(void)
```

```
{
                        while(kbhit())
                        {                                                   getch();
}
//---------------------------------------------------------------------------
void SetDio1Output(unsigned char NewData)
{
                        Set_W697HF_LD( 0x07);
                        Set_W697HF_Reg(0xF1, NewData);
}
//---------------------------------------------------------------------------
unsigned char GetDio1Input(void)
{
                        unsigned char result;

                        Set_W697HF_LD( 0x07);
                        result = Get_W697HF_Reg(0xF1);
                        return (result);
}
//---------------------------------------------------------------------------
void SetDio1Direction(unsigned char NewData)
{
                        unsigned char result;

                        Set_W697HF_LD(0x07);

                        //Enable GPIO_1
                        Set_W697HF_Reg(0x30, 0x01);

                        //NewData : 1 for input, 0 for output
                        Set_W697HF_Reg(0xF0, NewData);
}
//---------------------------------------------------------------------------
unsigned char GetDio1Direction(void)
{
                        unsigned char result;

                        Set_W697HF_LD(0x07);
                        result = Get_W697HF_Reg(0xF0);
                        return (result);
}
//---------------------------------------------------------------------------
void SetDio5Output(unsigned char NewData)
{
                        Set_W697HF_LD(0x08);
                        Set_W697HF_Reg(0xF1, NewData);
}
//---------------------------------------------------------------------------
unsigned char GetDio5Input(void)
{
                        unsigned char result;

                        Set_W697HF_LD(0x08);
                        result = Get_W697HF_Reg(0xF1);
                        return (result);
}
//---------------------------------------------------------------------------
void SetDio5Direction(unsigned char NewData)
{
                        Set_W697HF_LD(0x08);
                        //Enable GPIO_5
                        Set_W697HF_Reg(0x30, 0x01);

                        //NewData : 1 for input, 0 for output
                        Set_W697HF_Reg(0xF0, NewData);
}
//---------------------------------------------------------------------------
unsigned char GetDio5Direction(void)
{
```

```
                    unsigned char result;

                    Set_W697HF_LD(0x08);
                    result = Get_W697HF_Reg(0xF0);
                    return (result);
}
//---------------------------------------------------------------------------
```

Filename：w697hf.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 __W697HF_H
#define __W697HF_H                                      1
//===========================================================================
#define         W697HF_INDEX_PORT                       (W697HF_BASE)
#define         W697HF_DATA_PORT                        (W697HF_BASE+1)
//===========================================================================
#define         W697HF_REG_LD                                           0x07
//===========================================================================
#define W697HF_UNLOCK                                   0x87
#define         W697HF_LOCK                                             0xAA
//===========================================================================
unsigned int Init_W697HF(void);
void Set_W697HF_LD( unsigned char);
void Set_W697HF_Reg( unsigned char, unsigned char);
unsigned char Get_W697HF_Reg( unsigned char);
//===========================================================================
#endif          //__W697HF_H
```

Filename：w697hf.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 "W697HF.H"
#include <dos.h>
//===========================================================================
unsigned int W697HF_BASE;
void Unlock_W697HF (void);
void Lock_W697HF (void);
//===========================================================================
unsigned int Init_W697HF(void)
{
        unsigned int result;

        W697HF_BASE = 0x2E;
        result = W697HF_BASE;
        if (Get_W697HF_Reg(0x20) == 0x60)
        {       goto Init_Finish;   }

        W697HF_BASE = 0x4E;
        result = W697HF_BASE;
        if (Get_W697HF_Reg(0x20) == 0x60)
        {       goto Init_Finish;   }
```

```
                W697HF_BASE = 0x00;
                result = W697HF_BASE;

Init_Finish:
                return (result);
}
//===========================================================================
void Unlock_W697HF (void)
{
                outportb(W697HF_INDEX_PORT, W697HF_UNLOCK);
                outportb(W697HF_INDEX_PORT, W697HF_UNLOCK);
}
//===========================================================================
void Lock_W697HF (void)
{
                outportb(W697HF_INDEX_PORT, W697HF_LOCK);
}
//===========================================================================
void Set_W697HF_LD( unsigned char LD)
{
                Unlock_W697HF();
                outportb(W697HF_INDEX_PORT, W697HF_REG_LD);
                outportb(W697HF_DATA_PORT, LD);
                Lock_W697HF();
}
//===========================================================================
void Set_W697HF_Reg( unsigned char REG, unsigned char DATA)
{
                Unlock_W697HF();
                outportb(W697HF_INDEX_PORT, REG);
                outportb(W697HF_DATA_PORT, DATA);
                Lock_W697HF();
}
//===========================================================================
unsigned char Get_W697HF_Reg(unsigned char REG)
{
                unsigned char Result;
                Unlock_W697HF();
                outportb(W697HF_INDEX_PORT, REG);
                Result = inportb(W697HF_DATA_PORT);
                Lock_W697HF();
                return Result;
}
//===========================================================================
```

BIOS Setup

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

BIOS Introduction

The Award BIOS (Basic Input/Output System) installed in your computer system's ROM supports VIA 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 Award 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 Award 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.

Phoenix - Award BIOS CMOS Setup Utility

Standard CMOS Features	Frequency/Voltage Control
Advanced BIOS Features	Load Fail-Safe Defaults
Advanced Chipset Features	Load Optimized Defaults
Integrated Peripherals	Set Supervisor Password
Power Management Setup	Set User Password
PnP/PCI Configurations	Save & Exit Setup
PC Health Status	Exit Without Saving
ESC : Quit	↑ ↓ → ← : Select Item
F10 : Save & Exit Setup	
Time, Date, Hard Disk Type…	

The section below the setup items of the Main Menu displays the control keys for this menu. At the bottom of the Main Menu just below the control keys section, there is another section, which displays information on the currently highlighted item in the list.

***Note**: If the system cannot boot after making and saving system changes with Setup, the Award 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 Award 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.*

Standard CMOS Setup

"Standard CMOS Setup" choice allows you to record some basic hardware configurations in your computer system and set the system clock and error handling. If the motherboard is already installed in a working system, you will not need to select this option. You will need to run the Standard CMOS option, however, if you change your system hardware configurations, the onboard battery fails, or the configuration stored in the CMOS memory was lost or damaged.

Phoenix - Award BIOS CMOS Setup Utility
Standard CMOS Features

Date (mm:dd:yy)	Wed, Feb 18 2004	Item Help
Time (hh:mm:ss)	00 : 00 : 00	Menu Level
IDE Channel 0 Master	None	Change the day, month,
IDE Channel 0 Slave	None	Year and century
IDE Channel 1 Master	None	
IDE Channel 1 Slave	None	
IDE Channel 2 Master	None	
IDE Channel 3 Master	None	
Video	EGA/VGA	
Halt On	All, but keyboard	
Base Memory	640K	
Extended Memory	129024K	
Total Memory	130048K	

At the bottom of the menu are the control keys for use on this menu. If you need any help in each item field, you can press the <F1> key. It will display the relevant information to help you. The memory display at the lower right-hand side of the menu is read-only. It will adjust automatically according to the memory changed.

Date

The date format is:

Day : **Sun to Sat**
Month : **1 to 12**
Date : **1 to 31**
Year : **1994 to 2079**

To set the date, highlight the "Date" field and use the PageUp/ PageDown or +/- keys to set the current time.

Time

The time format is: **Hour : 00 to 23**
Minute : 00 to 59
Second : 00 to 59

To set the time, highlight the "Time" field and use the <PgUp>/ <PgDn> or +/- keys to set the current time.

IDE Channel 0/1 HDDs

The onboard PCI IDE connectors provide Primary and Secondary channels for connecting up to four IDE hard disks or other IDE devices. Each channel can support up to two hard disks; the first is the "Master" and the second is the "Slave".

Press <Enter> to configure the hard disk. The selections include Auto, Manual, and None. Select 'Manual' to define the drive information manually. You will be asked to enter the following items.

CYLS :	Number of cylinders
HEAD :	Number of read/write heads
PRECOMP :	Write precompensation
LANDZ :	Landing zone
SECTOR :	Number of sectors

The Access Mode selections are as follows:

Auto	
Normal	(HD < 528MB)
Large	(for MS-DOS only)
LBA	(HD > 528MB and supports Logical Block Addressing)

Video

This field selects the type of video display card installed in your system. You can choose the following video display cards:

EGA/VGA	For EGA, VGA, SEGA, SVGA or PGA monitor adapters. (default)
CGA 40	Power up in 40 column mode.
CGA 80	Power up in 80 column mode.
MONO	For Hercules or MDA adapters.

Halt On

This field determines whether or not the system will halt if an error is detected during power up.

No errors	The system boot will not be halted for any error that may be detected.
All errors	Whenever the BIOS detects a non-fatal error, the system will stop and you will be prompted.
All, But Keyboard	The system boot will not be halted for a keyboard error; it will stop for all other errors
All, But Diskette	The system boot will not be halted for a disk error; it will stop for all other errors.
All, But Disk/Key	The system boot will not be halted for a keyboard or disk error; it will stop for all others.

Advanced BIOS Features

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

Phoenix - Award BIOS CMOS Setup Utility
Advanced BIOS Features

CPU Feature	Press Enter	ITEM HELP
Hard Disk Booty Priority	Press Enter	Menu Level
Virus Warning	Disabled	
CPU L1 & L2 Cache	Enabled	
CPU L2 Cache ECC Checking	Enabled	
Quick Power On Self Test	Enabled	
First Boot Device	Hard Disk	
Second Boot Device	CDROM	
Third Boot Device	LS120	
Boot Other Device	Enabled	
Swap Floppy Drive	Disabled	
Boot Up Floppy Seek	Disabled	
Boot Up Numlock Status	On	
Typematic Rate Setting	Disabled	
Typematic Rate (chars/Sec)	6	
Typematic Delay (Msec)	250	
Security Option	Setup	
MPS Version Control for OS	1.4	
OS Select For DRAM>64MB	Non-OS2	
Console Redirection	Enabled	
Baud Rate	19200	
Agent Connect via	NULL	
Agent wait time(sec)	3	
Agent after boot	Disabled	
Video BIOS Shadow	Enabled	
Small Logo (EPA) Show	Disabled	

CPU Feature

This item allows you to set the “Delay Prior to Thermal” and “Thermal Management” where the default settings are “16 Min” and “Thermal Monitor 1

Hard Disk Booty Priority

This item allows you to arrange the priority of the devices where the system boots from.

Virus Warning

This item protects the boot sector and partition table of your hard disk against accidental modifications. If an attempt is made, the BIOS will halt the system and display a warning message. If this occurs, you can either allow the operation to continue or run an anti-virus program to locate and remove the problem.

CPU L1/L2 Cache

Cache memory is additional memory that is much faster than conventional DRAM (system memory). CPUs from 486-type on up contain internal cache memory, and most, but not all, modern PCs have additional (external) cache memory. When the CPU requests data, the system transfers the requested data from the main DRAM into cache memory, for even faster access by the CPU.

CPU L2 Cache ECC Checking

When enabled, it allows ECC checking of the CPU L2 cache. Enabling this feature is recommended because it will detect and correct single-bit errors in data stored in the L2 cache. It will also detect double-bit errors but not correct them.

Quick Power On Self Test

When enabled, this field speeds up the Power On Self Test (POST) after the system is turned on. If it is set to *Enabled*, BIOS will skip some items.

First/Second/Third Boot Device

These fields determine the drive that the system searches first for an operating system. The options available include *Floppy*, *LS/ZIP*, *HDD-0*, *SCSI*, *CDROM*, *HDD-1*, *HDD-2*, *HDD-3*, *LAN* and *Disable*.

Boot Other Device

These fields allow the system to search for an operating system from other devices other than the ones selected in the First/Second/Third Boot Device.

Swap Floppy Drive

This item allows you to determine whether or not to enable Swap Floppy Drive. When enabled, the BIOS swaps floppy drive assignments so that Drive A becomes Drive B, and Drive B becomes Drive A. By default, this field is set to ***Disabled.***

Boot Up Floppy Seek

This feature controls whether the BIOS checks for a floppy drive while booting up. If it cannot detect one (either due to improper configuration or its absence), it will flash an error message.

Boot Up NumLock Status
This allows you to activate the NumLock function after you power up the system.

Typematic Rate Setting
When disabled, continually holding down a key on your keyboard will generate only one instance. When enabled, you can set the two typematic controls listed next. By default, this field is set to ***Disabled.***

Typematic Rate (Chars/Sec)
When the typematic rate is enabled, the system registers repeated keystrokes speeds. Settings are from 6 to 30 characters per second.

Typematic Delay (Msec)
When the typematic rate is enabled, this item allows you to set the time interval for displaying the first and second characters.

Security Option
This field allows you to limit access to the System and Setup. When you select *System,* the system prompts for the User Password every time you boot up. When you select *Setup,* the system always boots up and prompts for the Supervisor Password only when Setup utility is called up.

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

OS Select for DRAM > 64MB
This option allows the system to access greater than 64MB of DRAM memory when used with OS/2 that depends on certain BIOS calls to access memory. The default setting is ***Non-OS/2***.

Video BIOS Shadow
This item allows you to change the Video BIOS location from ROM to RAM. Video Shadow will increase the video speed.

Small Logo (EPA) Show
This field enables the showing of the EPA logo located at the upper right of the screen during boot up.

Advanced Chipset Features

This Setup menu controls the configuration of the chipset.

Phoenix - AwardBIOS CMOS Setup Utility
Advanced Chipset Features

DRAM Clock / Drive Control	Press Enter	ITEM HELP
AGP & P2P Bridge Control	Press Enter	Menu Level
CPU & PCI Bus Control	Press Enter	
Memory Hole	Disabled	
System BIOS Cacheable	Enabled	
Video RAM Cacheable	Disabled	
Init Display First	PCI Slot	

Phoenix - AwardBIOS CMOS Setup Utility
DRAM Clock/Driver Control

Current FSB Frequency	100MHz	ITEM HELP
Current DRAM Frequency	266MHz	Menu Level
DRAM Clock	By SPD	
DRAM Timing	Auto By SPD	
DRAM CAS Latency	2.5/4	
Bank Interleave	Disabled	
Precharge to Active (Trp)	4T	
Active to Precharge (Tras)	7T	
Active to CMD(Trcd)	4T	
REF to ACT/REF to REF(Trfc)	21T	
ACT(0) to ACT(1) TRRD)	3T	
Read to Precharge (Trtp)	2T	
Write to Read CMD (Twtr)	1T/2T	
Write Recovery Time (Twr)	4T	
DRAM Command Rate	2T Command	
RDSAIT Mode	Auto	
RDSAIT Selection	03	

Phoenix - AwardBIOS CMOS Setup Utility
AGP & P2P Bridge Control

AGP Aperture Size	128M	ITEM HELP
AGP 3.0 Mode	8X	
AGP Driving Control	Auto	Menu Level
AGP Driving Value	DA	
AGP Fast Write	Disabled	
AGP Master 1 WS Write	Enabled	
AGP Master 1 WS Read	Enabled	
AGP 3.0 Calibration cycle	Disabled	
VGA Share Memory Size	16M	
Direct Frame Buffer	Enabled	
Outport Port	DIO	
Dithering	Disabled	

Phoenix - AwardBIOS CMOS Setup Utility
CPU & PCI Bus Control

PCI Master 0 WS Write	Enabled	ITEM HELP
PCI Delay Transaction	Enabled	Menu Level
Vlink mode selection	By Auto	Menu Level
Vlink 8x Support	Enabled	
DRDY Timing	Default	

DRAM Clock / Drive Control

This field provides settings related to DRAM. The fields are listed below.

Current FSB Frequency

The default setting of the FSB Frequency is 100MHz.

Current DRAM Frequency

The default setting of the DRAM Frequency is 266MHz.

DRAM Clock

The default setting of the DRAM clock is SPD.

DRAM Timing

This option refers to the method by which the DRAM timing is selected. The default is Auto by SPD.

DRAM CAS Latency

This is the period between when the chipset requests data from memory and when the memory is ready to send the data across the bus.

Bank Interleave

This decides how multiple memory modules communicate. It will only make a difference if you have more than one memory module.

Precharge to Active(Trp)

Theamount of time from a bank precharge request to when it can be activated.

Active to Precharge(Tras)

The Active to Precharge timing controls the length of the delay between the activation and precharge commands – the length of time after activation can the access cycle be started again.

Active to CMD(Trcd)

This is the time between a row access request and a column access request.

REF to ACT/REF to REF(Trfc)

The default setting is 21T.

ACT(0) to ACT(1) (TRRD)

The default time setting is 4T.

DRAM Command Rate

The time to wait after a chip select before activate and read can be started.

Read to Precharge (Trtp)

The default time setting is 2T.

Write to Read CMD (Twtr)
The default time setting is 1T/2T.
Write Recovery Time
The default time setting is 4T.
DRAM Command Rate
The time to wait after a chip select before activate and read can be started.
RDSAIT Mode
The default time setting is Auto.
RDSAIT Selection
The default time setting is 03.

AGP & P2P Bridge Control

The fields related to AGP & P2P Bridge Control are listed below.

AGP Aperture Size
The field sets aperture size of the graphics. The aperture is a portion of the PCI memory address range dedicated for graphics memory address space. Host cycles that hit the aperture range are forwarded to the AGP without any translation. The default setting is 64M.
AGP 3.0 Mode
The default setting is 8X.
AGP Driving Value
This decides how multiple memory modules communicate. It will only make a difference if you have more than one memory module.
AGP Fast Write
This accelerates memory write transactions from the chipset to the AGP device.
AGP Master 1 WS Write
When enabled, this changes the default from a 2ws to a 1ws which will increase AGP Writing.
AGP Master 1 WS Read
By default, the AGP busmastering device waits for at least 2 wait states before it starts a write transaction. When enable, this option sets the delay to 1 wait state.
AGP 3.0 Calibration cycle
By default, this field is disabled.
VGA Share memory Size
By default, this field is set to 16M.
Direct Frame Buffer
By default, this field is set to Enabled.

Outport Port

By default, this field is set to DI0.

Dithering

By default, this field is set to Disabled.

CPU & PCI Bus Control

The fields related to CPU & PCI Bus Control are listed below.

PCI Master 0 WS Write

This determines whether the chipset inserts a delay before any writes from the PCI bus.

PCI Delay Transaction

This is used to meet the latency of PCI cycles to and from the ISA bus.

Vlink mode selection

The default is set to By Auto.

Vlink 8X Support

By default, this field is enabled.

DRDY_Timing

By default, this field is set to Default.

Integrated Peripherals

This section sets configurations for your hard disk and other integrated peripherals.

Phoenix - Award BIOS CMOS Setup Utility
Integrated Peripherals

VIA OnChip IDE Device	Press Enter	ITEM HELP
VIA OnChip PCI Device	Press Enter	Menu Level
SuperIO Device	Press Enter	

VIA OnChip IDE Device

Upon pressing Enter on this field, another window appears. Below are the fields shown with their respective default settings:

OnChip SATA – Enabled
SATA Mode – IDE
IDE DMA transfer access – Enabled
OnChip IDE Channel0 – Enabled
OnChip IDE Channel1 – Enabled
IDE Prefetch Mode – Enabled
Primary Master PIO – Auto
Primary Slave PIO – Auto
Secondary Master PIO – Auto
Secondary Slave PIO – Auto
Primary Master UDMA – Auto
Primary Slave UDMA – Auto
Secondary Master UDMA – Auto
Secondary Slave UDMA – Auto
IDE HDD Block Mode – Enabled

VIA OnChip PCI Device

Upon pressing Enter on this field, another window appears. Below are the fields shown with their respective default settings:

Onboard LAN Boot ROM – Disabled
OnChip USB Controller – Enabled
OnChip EHCI Controller – Enabled
USB Emulation – ON
USB Keyboard Support – Enabled
USB Mouse Support – Enabled

SuperIO Device

Upon pressing Enter on this field, another window appears. Below are the fields shown with their respective default settings:

Onboard Serial Port 1 – 3F8/IRQ4
Onboard Serial Port 2 – 2F8/IRQ3
Onboard Parallel Port – 378/IRQ7
Parallel Port Mode – SPP
EPP Mode Select– EPP1.7
ECP Mode Use DMA – 3

Power Management Setup

The Power Management Setup allows you to save energy of your system effectively.

Phoenix - Award BIOS CMOS Setup Utility
Power Management Setup

		ITEM HELP
ACPI Function	Disabled	
Power Management Option	User Define	
HDD Power Down	Disabled	
Suspend Mode	Disabled	
Video Off Option	Suspend -> Off	
Video Off Method	Blank Screen	
Modem Use IRQ	3	
Run VGA BIOS if S3 Resume	Auto	
IRQ/Event Activity Detect	Press Enter	

ACPI Function

Enable this function to support ACPI (Advance Configuration and Power Interface).

Power Management Option

This field allows you to select the type of power saving management modes. There are four selections for Power Management.

Min. Power Saving	Minimum power management
Max. Power Saving	Maximum power management.
User Define	Each of the ranges is from 1 min. to 1hr. Except for HDD Power Down which ranges from 1 min. to 15 min.

HDD Power Down

When enabled, and after the set time of system inactivity, the hard disk drive will be powered down while all other devices remain active.

Suspend Mode

BIOS will turn the HDD's motor off when system is in SUSPEND mode. By default, this field is disabled.

Video Off Option

This field sets the video off option. By default, video goes into suspend state and then Off.

Video Off Method

This field defines the Video Off features. There are three options.

V/H SYNC + Blank	Default setting, blank the screen and turn off vertical and horizontal scanning.
DPMS	Allows BIOS to control the video display.
Blank Screen	Writes blanks to the video buffer.

Modem Use IRQ

The default setting of this field is 3.

Run VGABIOS if S3 Resume

The default setting is Auto.
AC power loss.

IRQ/Event Activity Detect

The items under this field are I/O events that can prevent the system from entering a power saving mode or can awaken the system from such a mode. When an I/O device wants to gain the attention of the operating system, it signals this by causing an IRQ to occur. When the operating system is ready to respond to the request, it interrupts itself and performs the service.

Phoenix - AwardBIOS CMOS Setup Utility
IRQ/Event Activity Detect

USB Resume from S3	Disabled	ITEM HELP
VGA	OFF	Menu Level
LPT & COM	LPT / COM	
HDD & FDD	ON	
PCI Master	OFF	
RTC Alarm Resume	Disabled	
Date (of Month)	0	
Resume Time (hh : mm : ss)	0 : 0 : 0	
IRQs Activity Monitoring	Press Enter	

IRQ Activity Monitoring

When you press Enter on this field, the following window appears.

Phoenix - AwardBIOS CMOS Setup Utility
IRQs Activity Monitoring

Primary INTR	ON	ITEM HELP
IRQ3 (COM2)	Disabled	Menu Level
IRQ4 (COM1)	Enabled	
IRQ5 (LPT 2)	Enabled	
IRQ6 (Floppy Disk)	Enabled	
IRQ7 (LPT 1)	Enabled	
IRQ8 (RTC Alarm)	Disabled	
IRQ9 (IRQ2 Redir)	Disabled	
IRQ10 (Rserved)	Disabled	
IRQ11 (Reserved)	Disabled	
IRQ12 (PS/2 Mouse)	Enabled	
IRQ13 (Coprocessor)	Enabled	
IRQ14 (Hard Disk)	Enabled	
IRQ15 (Reserved)	Disabled	

PNP/PCI Configurations

This option configures the PCI bus system. All PCI bus systems on the system use INT#, thus all installed PCI cards must be set to this value.

Phoenix - Award BIOS CMOS Setup Utility
PnP/PCI Configurations

PNP OS Installed	No	ITEM HELP
Reset Configuration Data	Disabled	Menu Level
Resources Controlled By	Auto (ESCD)	Default is Disabled. Select Enabled to reset Extended System Configuration Data (ESCD) when you exit Setup if you have installed a new add-on and the system reconfiguration has caused such a serious conflict that the OS cannot boot
IRQ Resources	Press Enter	
PCI/VGA Palette Snoop	Disabled	
Assign IRQ for VGA	Enabled	
Assign IRQ for USB	Enabled	

PNP OS Installed

Enable the PNP OS Install option if it is supported by the operating system installed. The default value is ***No***.

Reset Configuration Data

This field allows you to determine whether to reset the configuration data or not. The default value is ***Disabled.***

Resources Controlled by

This PnP BIOS can configure all of the boot and compatible devices automatically with the use of a use a PnP operating system such as Windows 95.

PCI/VGA Palette Snoop

Some non-standard VGA display cards may not show colors properly. This field allows you to set whether or not MPEG ISA/VESA VGA cards can work with PCI/VGA. When this field is enabled, a PCI/VGA can work with an MPEG ISA/VESA VGA card. When this field is disabled, a PCI/VGA cannot work with an MPEG ISA/VESA card.

Assign IRQ for VGA

This field enables the assigning of an IRQ for VGA.

Assign IRQ for USB

This field enables the assigning of an IRQ for USB.

PC Health Status

Phoenix - Award BIOS CMOS Setup Utility
PC Health Status

		ITEM HELP
Thermal Duty Cycle	Disabled	
CPU Warning Temperature	Disabled	
Current System Temp.	39°C/102°F	
Current CPU Temp.	32°C/89°F	
Current FAN Speed	0 RPM	
Current FAN2 Speed	0 RPM	
Vcore	1.63V	
3.3V	3.37V	
+5V	5.05V	
+12V	12.09V	
VBAT(V)	3.21V	
5VSB(V)	5.05V	
Shutdown Temperature	Disabled	

Thermal Duty Cycle

By default, this field is disabled.

CPU Warning Temperature

This field allows the user to set the temperature so that when the temperature is reached, the system sounds a warning. This function can help prevent damage to the system that is caused by overheating.

Temperatures/Fan Speeds/Voltages

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

Shutdown Temperature

This field allows the user to set the temperature by which the system automatically shuts down once the threshold temperature is reached. This function can help prevent damage to the system that is caused by overheating.

Frequency/Voltage Control

This section shows the user how to configure the processor frequency.

Phoenix - Award BIOS CMOS Setup Utility
Frequency/Voltage Control

CPU Clock Ratio	15 X	ITEM HELP
Auto Detect PCI/DIMM Clk	Disabled	
Spread Spectrum	Disabled	Menu Level
CPU Host/AGP/PCI Clock	Default	

Auto Detect PCI/DIMM Clk

This field enables or disables the auto detection of the PCI/DIMM clock.

Spread Spectrum

This field sets the value of the spread spectrum. The default setting is ***Disabled***. This field is for CE testing use only.

CPU Host/AGP/PCI Clock

This field is set to Default.

Load Fail-Safe 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.

Load Optimized 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.

Set Supervisor/User Password

These 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, up to eight characters in length, 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.

Save & Exit Setup

This option allows you to determine whether or not to accept the modifications. If you type “Y”, you will quit the setup utility and save all changes into the CMOS memory. If you type “N”, you will return to Setup utility.

Exit Without Saving

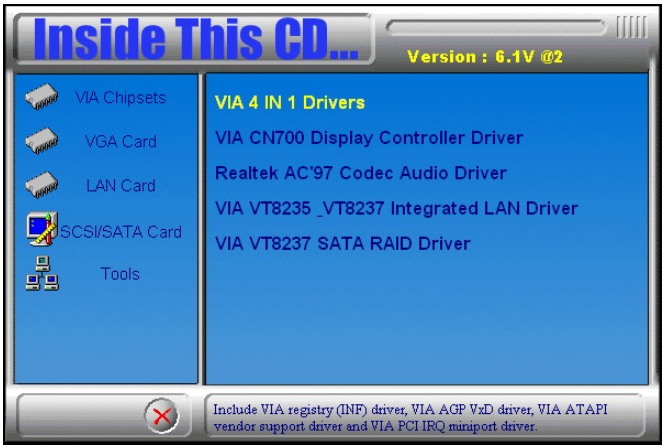
Select this option to exit the Setup utility without saving the changes you have made in this session. Typing “Y” will quit the Setup utility without saving the modifications. Typing “N” will return you to Setup utility.

Drivers Installation

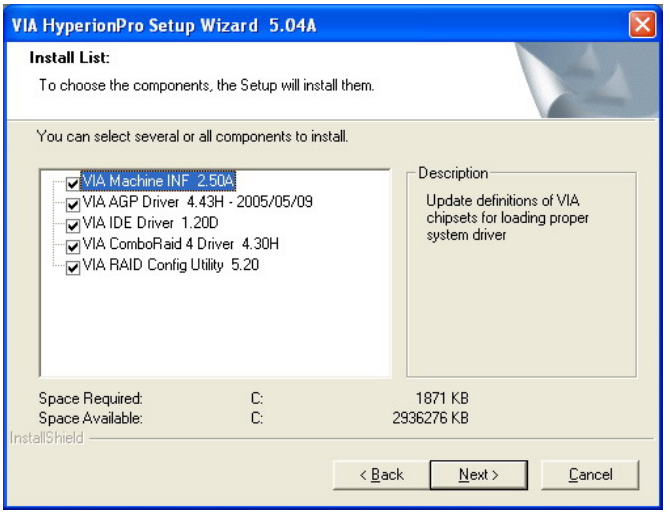
This section describes the installation procedures for software and drivers under the Windows OS. The software and drivers are included with the motherboard. If you find the items missing, please contact the vendor where you made the purchase. The contents of this section include the following:

VIA CN700 4 in 1 Driver Installation

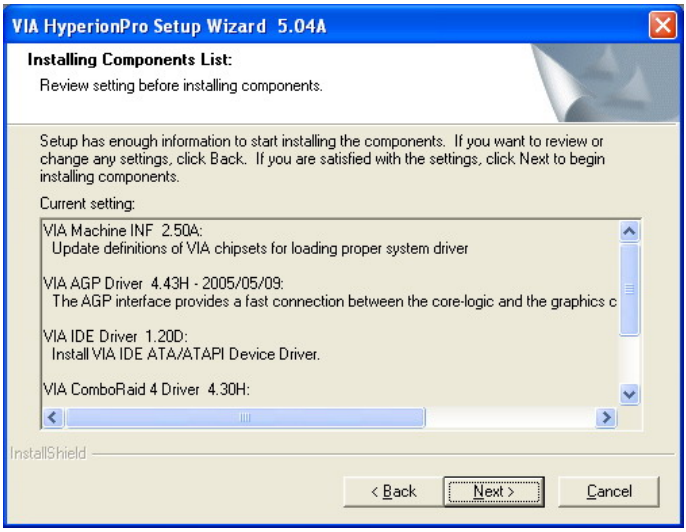
1. Insert the CD that comes with the board and the screen below would appear. Click **4 IN 1 Drivers**. The VIA HyperionPro Setup Wizard welcome screen will appear. Click **Next** to continue. When the license agreement window appears, click "**I Agree**", then click **Next** to continue.



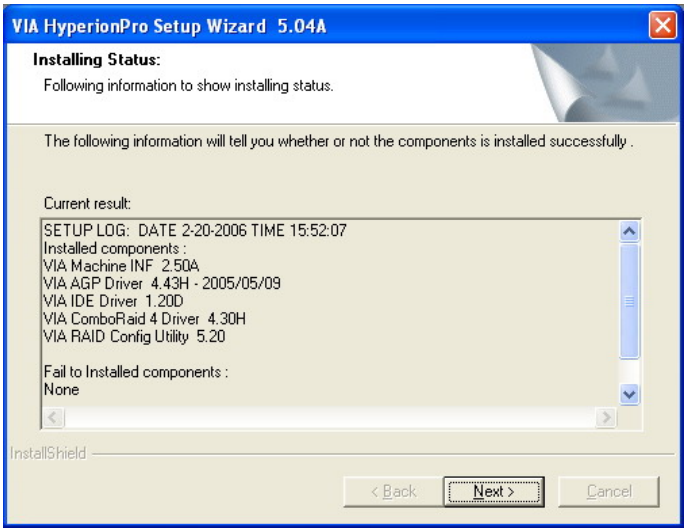
2. In the following window, click all the components to be installed, and click **Next** to continue.



3. You are now asked to review setting before installing the components, click **Next** to continue.

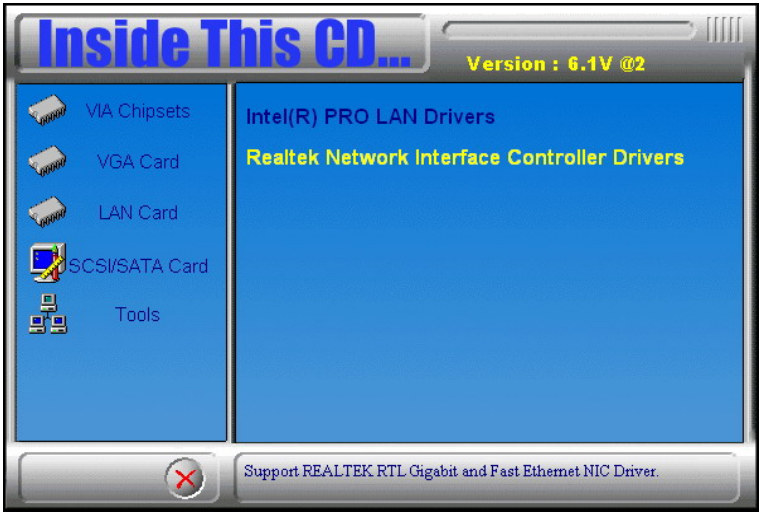


4. The following window shows you the installing status with the result. Click **Next**. The installation process is now complete. Restart the computer as prompted for changes to take effect



Realtek LAN Driver Installation

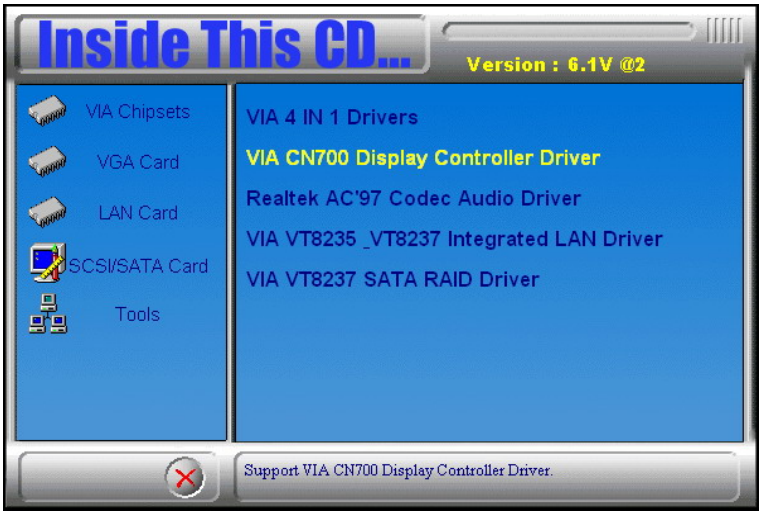
1. Insert the CD that comes with the motherboard and the screen below would appear. Click on **LAN Card**. On the next screen, click on **Realtek Network Interface Controller Drivers**. When the welcome screen appears, click **Next** to continue. After the installation, you will be prompted to restart the computer. Click **Finish**.



VIA CN700 VGA Driver Installation

Follow the steps below to install the VIA CN700 VGA Driver under Windows 2000/XP.

1. Insert the CD that comes with the motherboard and the screen below would appear. Click on **VIA CN700 Display Controller Driver** to start the installation.



2. Click **Finish** to complete the installation.

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
000h - 01Fh	DMA Controller #1
020h - 03Fh	Interrupt Controller #1
040h - 05Fh	Timer
060h - 06Fh	Keyboard Controller
070h - 07Fh	Real Time Clock, NMI
080h - 09Fh	DMA Page Register
0A0h - 0BFh	Interrupt Controller #2
0C0h - 0DFh	DMA Controller #2
0F0h	Clear Math Coprocessor Busy Signal
0F1h	Reset Math Coprocessor
1F0h - 1F7h	IDE Interface
278 - 27F	Parallel Port #2(LPT2)
2F8h - 2FFh	Serial Port #2(COM2)
2B0 - 2DF	Graphics adapter Controller
378h - 3FFh	Parallel Port #1(LPT1)
360 - 36F	Network Ports
3B0 - 3BF	Monochrome & Printer adapter
3C0 - 3CF	EGA adapter
3D0 - 3DF	CGA adapter
3F0h - 3F7h	Floppy Disk Controller
3F8h - 3FFh	Serial Port #1(COM1)

B. Interrupt Request Lines (IRQ)

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

Level	Function
IRQ0	System Timer Output
IRQ1	Keyboard
IRQ2	Interrupt Cascade
IRQ3	Serial Port #2
IRQ4	Serial Port #1
IRQ5	Reserved
IRQ6	Floppy Disk Controller
IRQ7	Parallel Port #1
IRQ8	Real Time Clock
IRQ9	Reserved
IRQ10	Reserved
IRQ11	Reserved
IRQ12	PS/2 Mouse
IRQ13	80287
IRQ14	Primary IDE
IRQ15	Secondary IDE