ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Find the *Abaco Systems / VMIC VMICPCI-7593-920* at our website: ***Click HERE***

VMICPCI-7593

Single Board Pentium® Processor-Based CompactPCI® CPU

Product Manual

12090 South Memorial Parkway
Huntsville, Alabama 35803-3308, USA
(256) 880-0444 ♦ (800) 322-3616 ♦ Fax: (256) 882-0859

500-657593-000 Rev. B
8/5/99



12090 South Memorial Parkway
Huntsville, Alabama 35803-3308, USA
(256) 880-0444 ♦ (800) 322-3616 ♦ Fax: (256) 882-0859

COPYRIGHT AND TRADEMARKS

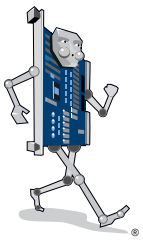
© Copyright July 1998. The information in this document has been carefully checked and is believed to be entirely reliable. While all reasonable efforts to ensure accuracy have been taken in the preparation of this manual, VMIC assumes no responsibility resulting from omissions or errors in this manual, or from the use of information contained herein.

VMIC reserves the right to make any changes, without notice, to this or any of VMIC's products to improve reliability, performance, function, or design.

VMIC does not assume any liability arising out of the application or use of any product or circuit described herein; nor does VMIC convey any license under its patent rights or the rights of others.

For warranty and repair policies, refer to VMIC's Standard Conditions of Sale.

AMXbus, BITMODULE, COSMODULE, DMAbus, Instant OPC wizard logo, IOWorks Access, IOWorks Foundation, IOWorks man figure, IOWorks Manager, IOWorks Server, MAGICWARE, MEGAMODULE, PLC ACCELERATOR (ACCELERATION), Quick Link, RTnet, Soft Logic Link, SRTbus, TESTCAL, "The Next Generation PLC", The PLC Connection, TURBOMODULE, UCLIO, UIOD, UPLC, Visual Soft Logic Control(ler), *VMEaccess*, *VMEmanager*, *VMEmonitor*, VMEnet, VMEnet II, and *VMEprobe* are trademarks of VMIC.



(I/O man figure)



(Instant OPC wizard logo)



(IOWorks man figure)

The I/O man figure, IOWorks, UIOC, Visual IOWorks, and *WinUIOC* are registered trademarks of VMIC.

ActiveX is a trademark and Microsoft, Microsoft Access, MS-DOS, Visual Basic, Visual C++, Win32, Windows, Windows NT, and XENIX are registered trademarks of Microsoft Corporation.

MMX is a trademark and Pentium is a registered trademark of Intel Corporation.

PICMG and CompactPCI are registered trademarks of PCI Industrial Computer Manufacturers' Group.

Other registered trademarks are the property of their respective owners.

VMIC

All Rights Reserved

This document shall not be duplicated, nor its contents used for any purpose, unless granted express written permission from VMIC.



12090 South Memorial Parkway
Huntsville, Alabama 35803-3308, USA
(256) 880-0444 ♦ (800) 322-3616 ♦ Fax: (256) 882-0859

Table of Contents

Chapter 1 - VMICPCI-7593 Features and Options 15
CompactPCI Features 18
VMICPCI-7593 Product Options 19

Chapter 2 - Installation and Setup 21
Unpacking Procedures 22
Hardware Setup 23
Installation 29
BIOS Setup 30
Front Panel Connectors 30
PMC Site Connector 30
LED Definition 32

Chapter 3 - PC/AT Functions 33
CPU Socket 34
Physical Memory 34
I/O Port Map 35
PCI-to-PCI Bridge 37
PC/AT Interrupts 37
PCI Interrupts 40
IIntegrated Peripherals 43
Ethernet Controller 44
10BaseT 44
100BaseTx 44
Remote Ethernet Booting 44
BootWare Features 44

Chapter 4 - IEmbedded PC/RTOS Features 45
Timers 46
General 46
Timer Interrupt Status 46
Clearing the Interrupt 47
Timer Programming 47
Architecture 47
Writing 50
Reading 51
Timer Latch Command 51
Read-Back Command 52
Mode Definitions 54
Flash Disk 55
Configuration 55
Functionality 56
Advanced Configuration 56
Watchdog Timer 59
Time of Day Registers 61
Time of Day Alarm Registers 62
Watchdog Alarm Registers 63
Command Register 63
Battery Backed SRAM 65

Chapter 5 - Maintenance 67
Maintenance Prints 67

Appendix A - Connector Pinouts 69
J1 Connector Pinout 71
J2 Connector Pinout 72
J3 Connector Pinout 73
Ethernet Connector Pinout 74
Keyboard and Mouse Connectors and Pinout 75
Floppy Interface Connector Pinout 76
EIDE Hard Drive Connector Pinout 77
PMC Connector Pinout 78
PMC P11 Connector and Pinout 78
PMC P12 Connector and Pinout 79

Appendix B - System Driver Software 81
Driver Software Installation 82
Windows for Workgroups™ OS (Version 3.11) 83
Windows® 95 OS 85
.Windows NT® OS (Version 4.0) 87

Appendix C - LANWorks BIOS 89
Introduction 89
Remote Ethernet Booting 89
BootWare Features 89
System BIOS Setup Utility 90
BIOS Features Setup 91
LANWorks BIOS Setup 92
Boot Protocol 92
Default Boot 92
Local Boot 92

Appendix D - Award - BIOS 93
System BIOS Setup Utility 94
Standard CMOS Setup 95
Setting The Date 95
Setting The Time 95
Primary Master/Slave 95
Secondary Master/Slave 96
Floppy Disk Drive 96
Floppy Drive A 96
Floppy Drive B 96
.Halt On 96
Memory 96
BIOS Features Setup 97
Anti-Virus Protection 97
CPU Internal Cache 97
External Cache 97
Quick Power On Self Test 97
Boot From LAN First 98
Boot Sequence 98
Swap Floppy Drive 98
Boot Up Floppy Seek 98
Boot Up NumLock Status 98

Boot Up System Speed 98
Typematic Rate Setting 98
Typematic Rate (Chars/Sec) 98
Typematic Delay (Msec) 98
Security Option 99
PCI/VGA Palette Snoop 99
OS Select For DRAM>64MB 99
Report No FDD For WIN 95 99
Chipset Features Setup 100
Advanced Options 100
Auto Configuration 100
DRAM Timing 100
DRAM Leadoff Timing 101
DRAM Read Burst (EDO/FP) 101
DRAM Write Burst Timing 101
Refresh RAS# Assertion 101
Fast RAS to CAS Delay 101
DRAM Page Idle Timer 101
DRAM Enhanced Paging 101
Fast MA to RAS# Delay 101
SDRAM (CAS Lat/RAS-to-CAS) 101
SDRAM Speculative Read 102
System BIOS Cacheable 102
Video BIOS Cacheable 102
8 Bit I/O Recovery Time 102
16 Bit I/O Recovery Time 102
Memory Hole at 15M-16M 102
PCI 2.1 Compliance 102
Power Management 103
ACPI Function 103
Power Management 103
PM Control by APM 104
Video Off Method 104
Video Off After 104
Modem Use IRQ 104
Doze Mode 104
Standby Mode 104
Suspend Mode 104

HDD Power Down 104
Throttle Duty Cycle 105
ZZ Active in Suspend 105
PCI/VGA Active Monitor 105
Soft-Off by PWR-BTTN 105
Power On by Ring 105
IRQ8 Break Suspend 105
Reload Global Timer Events 105
PnP/PCI Configuration 106
PNP OS Installed 106
Resources Controlled By 106
Reset Configuration Data 106
IRQ n Assigned to 106
DMA n Assigned to 107
PCI IDE IRQ Map to 107
Primary/Secondary IDE INT# 107
Used Mem base addr 107
Assign IRQ for USB 107
Integrated Peripherals 108
IDE HDD Block Mode 108
IDE Primary Master PIO 108
IDE Primary Slave PIO 108
IDE Primary Master UDMA 109
IDE Primary Slave UDMA 109
IDE Secondary Master PIO 109
IDE Secondary Slave PIO 109
IDE Secondary Master UDMA 109
IDE Secondary Slave UDMA 109
On-Chip Primary PCI IDE 109
On-Chip Secondary PCI IDE 110
USB Keyboard Support 110
Onboard FDC Controller 110
Onboard Serial Port 1/2 110
UART 2 Mode 110
Onboard Parallel Port 110
Parallel Port Mode 110

Appendix E - Device Configuration: I/O and Interrupt Control 111

BIOS Operations 112

BIOS Control Overview 112
Functional Overview 112
Data Book References 114
Device Address Definition 117
ISA Devices 117
PCI Devices 118
Device Interrupt Definition 119
PC/AT Interrupt Definition 119
ISA Device Interrupt Map 119
PCI Device Interrupt Map 121

Appendix F- Sample C Software 123
Directory SRAM 124
**.File: T_SRAM.C 124
.Directory Timers 127
**File: CPU.H 127
.**File: PCI.H 129
**File: PIC.C 131
**File: T_Timers.C. 134
** File: Timer.C 138
Directory WATCHDOG 143
**File: Watchdog.H 143
**File: W.DT0_RST.C 144

List of Figures

Figure 1-1 VMICPCI-7593 Block Diagram 17
Figure 2-1 VMICPCI-7593 CPU Board, I/O Port, and Jumper Locations 24
Figure 2-2 Installing a PMC Card on the VMICPCI-7593 31
Figure 2-3 Front Panel LED Positions 32
Figure 3-1 Connections for the PC Interrupt Logic Controller 42
Figure 3-2 VMIACC-0559 CompactPCI Rear Transition Utility Board 43
Figure 4-1 Timer Interrupt Status Register Read/Steps 46
Figure 4-2 Timer Interrupt Status Register 47
Figure 4-3 Clearing the Timer Interrupt Status Register 47
Figure 4-4 82C54 Diagram 48
Figure 4-5 Internal Timer Diagram 49
Figure 4-6 Typical System Configuration 55
Figure 4-7 Watchdog Alarm Block 59
IFigure A-1 VMICPCI-7593 Connector and Jumper Locations 70
Figure A-2 J1 Connector 71
Figure A-3 J2 Connector 72
Figure A-4 J3 Connector 73
Figure A-5 Ethernet Connector and Pinout 74
Figure A-6 Keyboard Connector and Pinout 75
Figure A-7 Mouse Connector and Pinout 75
Figure A-8 Floppy Drive Header and Pinout 76
Figure A-9 EIDE Hard Drive Connector and Header Pinout 77
Figure E-1 VMICPCI-7593 Block Diagram 113
Figure E-2 BIOS Default Connections for the PC Interrupt Logic Controller 120

List of Tables

Table 1-1 PC/AT I/O Features 16
Table 2-1 CPU Board Connectors 25
Table 2-2 CPU Bus Frequency Select - Jumpers (E20, E22, and E23) 25
Table 2-3 Programmable Timer Clock Select (User Configurable) - Jumper (E4) 25
Table 2-4 CPU Selection - Jumpers (E10 and E13) 26
Table 2-5 Fan Connector Option - Jumper (E24) 26
Table 2-6 CPU Power Supply Option - Jumpers (E5, E8, E1, E7, and E9) 26
Table 2-7 CPU Clock/Bus Multiple - Jumper Blocks (E11 and E12) 26
Table 2-8 EIDE Routing Option (User Configurable) - Jumper (E18) 27
Table 2-9 Factory Installed - Jumper (E19) 27
Table 2-10 Factory Installed Jumper - (E2) 27
Table 2-11 BIOS Mode Option - Jumper (E3) 27
Table 2-12 430TX Clock Select - Jumper (E12) 28
Table 2-13 Watchdog Timer (User Configurable) - Jumper (E6) 28
Table 2-14 Password Clear (User Configurable) - Jumper (E14) 28
Table 3-1 VMICPCI-7593 I/O Address Map 35
Table 3-2 PC/AT Hardware Interrupt Line Assignments 37
Table 3-3 PC/AT Interrupt Vector Table 38
Table 3-4 NMI Register Bit Descriptions 41
Table 4-1 I/O Address of the Control Word Register and Timers 47
Table 4-2 Control Word Format 50
Table 4-3 ST - Select Timer 50
Table 4-4 RW - Read/Write 50

Table 4-5 M - Mode 51
Table 4-6 BCD 51
Table 4-7 Read-Back Command Format 52
Table 4-8 Read-Back Command Description 52
Table 4-9 Status Byte 53
Table 4-10 Status Byte Description 53
Table 4-11 LOAD Bit Operation 53
Table 4-12 Watchdog Registers 60
Table 4-13 Time of Day Alarm Registers 62
Table A-1 J1 Connector Pinout. 71
Table A-2 J2 Connector Pinout. 72
Table A-3 J3 Connector Pinout. 73
Table E-1 ISA Device Mapping Configuration 117
Table E-2 PCI Device Mapping Configuration 118
Table E-3 PCI Device Interrupt Mapping by the BIOS 121
Table E-4 Default PIRQx to IRQx BIOS Mapping with all Devices Loaded 122

.

CHAPTER
1

VMICPCI-7593 Features and Options

Contents

Introduction 15
CompactPCI® Features. 18
VMICPCI-7593 Product Options 19

Introduction

The VMICPCI-7593 performs all the functions of a standard IBM PC / AT motherboard with the following features:

- Single-slot CPCI bus 6U Eurocard form factor
- Includes a high-performance Intel Pentium® processor
- Up to 128 Mbyte of Synchronous DRAM
- Real-time clock / calendar
- Front panel reset switch and miniature speaker
- Onboard ports for keyboard and mouse,
- UltraEIDE hard drive, floppy drive through J3 or through onboard header
- Two onboard fast Ethernet controllers supporting 10BaseT and 100BaseTX interfaces
- Front panel “vital sign” indicators (power, UltraEIDE hard drive activity, Ready, and Ethernet status for each controller)
- Three general-purpose programmable 16-bit timers
- Software-selectable watchdog timer with reset
- 16 Mbyte of bootable flash on secondary IDE
- 32 Kbyte of battery-backed SRAM
- Two Serial ports, and one Enhanced parallel port through J3

The VMICPCI-7593 supports standard PC/AT I/O features such as those listed in Table 1-1. Figure 1-1 on page 17 shows a block diagram of the VMICPCI-7593 emphasizing the I/O features, including the PCI-to-PCI bridge.

Table 1-1 PC/AT I/O Features

I/O FEATURE	IDENTIFIER	PHYSICAL ACCESS
Two high-performance 16550-compatible serial port	COM1, COM2	Through J3 Connector (See Note)
One Enhanced Parallel Port, Supports ECP/EPP Modes	Parallel	Through J3 connector (See Note)
AT-Style Keyboard Controller	KYB	Front Panel PS/2-Style Connector, Mini-DIN Circular (female)
PS/2 Mouse	MSE	Front Panel PS/2-Style Connector, Mini-DIN Circular (female)
Real-Time Clock/Calendar with miniature speaker	Date, Time, and sound	
Two on-board fast Ethernet controllers supporting 10BaseT and 100BaseTX interfaces	LAN	Front Panel R-J45
Floppy Disk Controller	Drive A	Through J3 or on-board pinheaders (See Note)
Ultra EIDE Hard Drive Controller	Drive C	Through J3 or on-board pinheaders (See Note)
Hardware Reset	RST	Front Panel Push-Button
Power Status, Hard Drive Activity, Ready, and Ethernet Status for each controller	LED Indicators	Front Panel

Note

The VMIACC-0559 CompactPCI Rear Transition Utility Board provides RJ-45 connectors (2), standard (DB25) parallel connector, IDE and Floppy connectors, by way of the CompactPCI backplane Connector (J3).

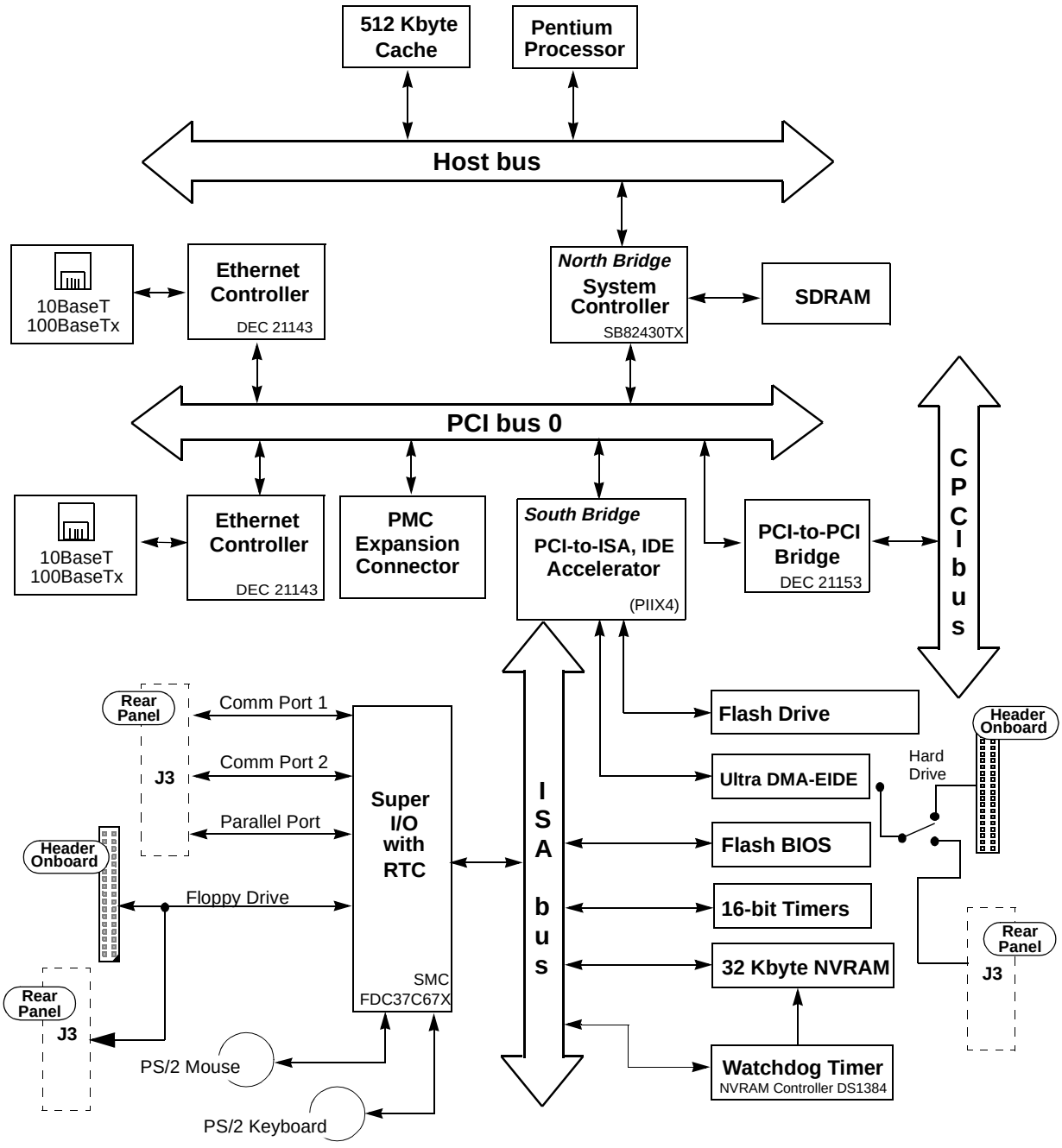


Figure 1-1 VMICPCI-7593 Block Diagram

CompactPCI® Features

In addition to its PC / AT functions, the VMICPCI-7593 has the following CompactPCI features:

- Single-slot, 6U height CPCI board
- Complies fully with Revision 2.1 of the *PCI Local Bus Specification*
- Complies fully with Revision 1.1 of the *PCI-to-PCI Bridge Architecture Specification*
- Complies with Revision 2.1 of the *CompactPCI Specification*
- Implements delayed transactions for all PCI configuration, I / O, and memory read commands-up to three transactions simultaneously in each direction
- Allows 152 bytes of buffering (data and address) for upstream posted memory write commands and 88 bytes of buffering for downstream posted memory write commands- up to nine upstream and five downstream posted write transactions simultaneously
- Allows 152 bytes of read data buffering upstream and 72 bytes of read data buffering downstream
- Provides concurrent primary and secondary bus operation to isolate traffic
- Provides enhanced address decoding
- Includes addressing and VGA palette snooping support
- Supports PCI transaction forwarding for the following commands
  - All I / O and memory commands
  - Type 1 to Type 1 configuration commands
  - Type 1 to Type 0 configuration commands (downstream only)
  - All Type 1 to special cycle configuration commands
- Includes downstream lock support
- Supports both 5 and 3.3 V signaling environments

The VMICPCI-7593 is a versatile single-board solution for CPCI control with familiar PC / AT operation.

VMICPCI-7593 Product Options

VMIC's VMICPCI-7593 is built around two fundamental hardware configurations. These involve processor power, and the RAM memory. *These options are subject to change based on emerging technologies and availability of vendor configurations.*

The options and current details available with the VMICPCI-7593 are defined in the device specification sheet available from your VMIC representative.

The VMICPCI-7593 CompactPCI interface is provided by the PCI-to-PCI bridge (DEC 21153).

CHAPTER
2

Installation and Setup

Contents

Unpacking Procedures 22
Hardware Setup 23
Installation 29

Introduction

This chapter describes the hardware jumper settings, connector definitions, installation, system setup, and operation of the VMICPCI-7593.

Unpacking Procedures

Any precautions found in the shipping container should be observed. All items should be carefully unpacked and thoroughly inspected for damage that might have occurred during shipment. The board(s) should be checked for broken components, damaged printed circuit board(s), heat damage, and other visible contamination. All claims arising from shipping damage should be filed with the carrier and a complete report sent to VMIC Customer Service together with a request for advice concerning the disposition of the damaged item(s).



Some of the components assembled on VMIC's products may be sensitive to electrostatic discharge and damage may occur on boards that are subjected to a high energy electrostatic field. When the board is placed on a bench for configuring, etc., it is suggested that conductive material be inserted under the board to provide a conductive shunt. Unused boards should be stored in the same protective boxes in which they were shipped.

Hardware Setup

The VMICPCI-7593 is factory populated with user-specified options as part of the VMICPCI-7593 ordering information. The CPU speed and RAM size are not user-upgradable. To change CPU speeds or RAM size, contact customer service to receive a Return Material Authorization (RMA).

VMIC Customer Service is available at: 1-800-240-7782.

The VMICPCI-7593 is tested for system operation and shipped with factory-installed header jumpers. The physical location of the jumpers and connectors for the single board CPU are illustrated in Figure 2-1 on page 24. The definitions of the CPU board jumpers and connectors are included in Table 2-1 through Table 2-14.



All jumpers are factory configured and should not be modified by the user. There are four exceptions: the Programmable Timer Clock Select (E4), the IDE Routing Option (E18), the Password Clear (E14), and the Watchdog Timer (E6). Modifying any other jumper will void the Warranty and may damage the unit. The default jumper condition of the VMICPCI-7593 is expressed in Table 2-1 through Table 2-9 with **bold text** in the table cells.

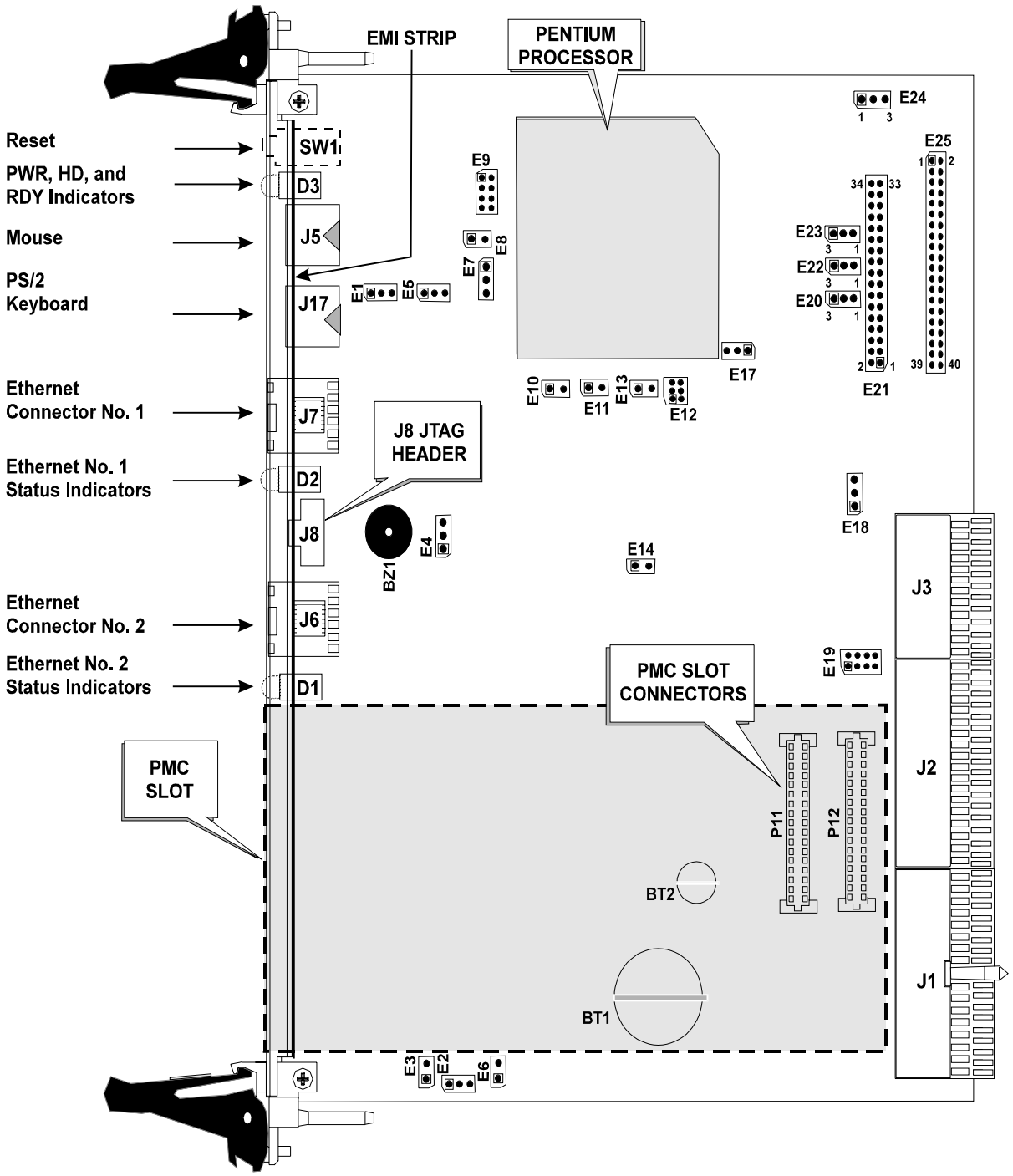


Figure 2-1 VMICPCI-7593 CPU Board, I/O Port, and Jumper Locations

Table 2-1 CPU Board Connectors

Connector	Function
J5	Mouse Connector
J17	PS/2 Keyboard Connector
J3	COM2
J3	COM1
J3	Parallel Port Connector
J7	Ethernet Connector No. 1
J6	Ethernet Connector No. 2
E21, J3	Floppy Interface Connector
J8	JTAG Header
E25, J3	EIDE Connector
E24	Fan Connector
P11, P12	PMC Slot
J1, J2	Compact PCI Connectors

Table 2-2 CPU Bus Frequency Select - Jumpers (E20, E22, and E23)

CPU Bus Frequency (MHz)	PCI Bus Frequency (MHz)	E20 Position	E22 Position	E23 Position
60	30	1-2	1-2	1-2
66.6	***33.3***	***1-2***	***1-2***	***2-3***
50	25	1-2	2-3	1-2
55	27.5	1-2	2-3	2-3
Reserved	Reserved	2-3	1-2	1-2
83.3	33.3	2-3	1-2	2-3

Table 2-3 Programmable Timer Clock Select (**User Configurable**) - Jumper (E4)

Select	Jumper Position
2 MHz	1-2
1 MHz	***2-3***

Table 2-4 CPU Selection - Jumpers (E10 and E13)

Jumper	Default
E10	***In***
E13	***In***

Table 2-5 Fan Connector Option - Jumper (E24)

Select	Jumper Position
+5 V	***1-2***
+12 V	2-3

Table 2-6 CPU Power Supply Option - Jumpers (E5, E8, E1, E7, and E9)

CPU	Select	E5	E8	E1	E7	E9
166	VIO=3.3 V, VCORE=3.3 V	2-3	Out	1-2	1-2	In
200/233	***VIO=3.3 V, VCORE=2.8 V***	***2-3***	***In***	***1-2***	***1-2***	***Out***

Table 2-7 CPU Clock/Bus Multiple - Jumper Blocks (E11 and E12)

CPU	Intel Based		Other (Non-Intel) Based	
	E11	E12	E11	E12
166	Omitted	3-5, 4-6	Out	3-5, 4-6
200	Omitted	1-3, 4-6	Out	1-3, 4-6
233	Omitted	1-3, 2-4	Out	1-3, 2-4

Table 2-8 Hard Drive Routing Option (**User Configurable**) - Jumper (E18)

Select	Jumper Position
Route Hard Drive to Onbound Header (E25)	***1-2***
Route Hard Drive to J3	2-3

Note

The default jumper configuration for jumper E18 is jumper pins 1-2, which enables the utilization of the onboard header E25 for an IDE hard drive interface. If the user is configuring the VMICPCI-7593 for use with a VMIACC-0559 CompactPCI Rear Transition Utility Board, E18 should have a jumper shunt position on pins 2-3.

Table 2-9 Factory Installed - Jumper (E19)

Select	Jumper Position	
INTA	***1 - 2***	***In***
INTB	***3 - 4***	***In***
INTC	***5 - 6***	***In***
INTD	***7 - 8***	***In***

Table 2-10 Factory Installed Jumper - (E2)

E2	Jumper Position	
SRAM	2-3	In

Table 2-11 BIOS Mode Option - Jumper (E3)

Select	Jumper Position
Boot Block Programming	In
No Program	***Out***

Table 2-12 430TX Clock Select - Jumper (E17)

CPU Clock	Jumper Position
66 MHz	1-2

Table 2-13 Watchdog Timer (**User Configurable**) - Jumper (E6)

Select	Jumper Position
Watchdog Timer Reset	In
No Watchdog Timer Reset	***Out***

Table 2-14 Password Clear (**User Configurable**) - Jumper (E14)

	Jumper Position
Normal	***Out***
Clear NVRAM/CMOS/ Password	In

Note

The BIOS has the capability (not currently enabled) of password protecting casual access to the unit's CMOS set-up screens. The Password Clear jumper allows for a means to clear the password feature, this might be necessary to perform in the case of a forgotten password.

To clear the CMOS password:

1. Turn off power to the unit
2. Install a jumper at E14
3. Power up the unit
4. Turn off the power to the unit and remove the jumper from E14

When power is reapplied to the unit, the CMOS password will be cleared.

Installation

The VMICPCI-7593 conforms to the CompactPCI physical specification for a 6U board. The VMICPCI-7593 is a system slot only board. It can be plugged directly into any standard chassis accepting this type of board. The following picture illustrates the symbol use to identify the slots in a standard CPCI chassis.



This symbol identifies the System Controller slot



This symbol identifies the slot in all other boards can be installed



CAUTION: Do not install or remove the board while power is applied.

The following steps describe the VMIC recommended method for installation and powerup of the VMICPCI-7593:

1. Make sure power to the equipment is off.
2. If a PMC module such as VMIC's VMIPMC-7441 is to be used, connect it to the VMICPCI-7593 prior to board installation. Refer to the Product Manual for that particular board for configuration and setup.
3. The VMICPCI-7593 must be installed in the designated system slot of the Compact PCI backplane (See symbols about for selection of the correct slot).



The VMICPCI-7593 requires forced air cooling. It is advisable to install blank panels over any exposed slots. This will allow for better air flow over the VMICPCI-7593 board.

4. Insert the VMICPCI-7593 into the CompactPCI chassis system slot. While ensuring that the board is properly aligned and oriented in the supporting board guides, slide the board smoothly forward against the mating connector until firmly seated.
5. All needed peripherals can be accessed from the front panel and the rear I/O VMIACC-0559 Rear Transition utility board. Each connector is clearly labeled on the front panel, and detailed pinouts are in Appendix A.



The VMICPCI-7593 does not have on-board VGA. A VGA PMC card can be installed in the available PMC site, or in the CPCI chassis along with a keyboard and mouse.

6. Apply power to the system. If a PMC or CPCI video card is used, several messages are displayed on the screen, including names, versions, and copyright

dates for the various BIOS modules on the VMICPCI-7593. Also a Boot Complete LED (Ready) is available on the front panel.

7. The VMICPCI-7593 features a Flash Disk resident on the board. Refer to Chapter 4 for set up details.
8. If an external drive module is installed, the BIOS Setup program must be run to configure the drive types. See Appendix C to properly configure the system.
9. If a drive module is present, install the operating system according to the manufacturer's instructions.
10. A keyboard and a mouse are required if the user has not previously configured the system

See Appendix B for instructions on installing VMICPCI-7593 peripheral driver software during operating system installation.

BIOS Setup

The VMICPCI-7593 has an on-board BIOS Setup program that controls many configuration options. These options are saved in a special nonvolatile, battery-backed memory chip and are collectively referred to as the board's "CMOS configuration." The CMOS configuration controls many details concerning the behavior of the hardware from the moment power is applied.

The VMICPCI-7593 is shipped from the factory with hard drive type configuration set to AUTO in the CMOS.

Details of the VMICPCI-7593 BIOS setup program are included in Appendix D.

Front Panel Connectors

The front panel connections, including connector pinouts and orientation, for the VMICPCI-7593 are defined in detail in Appendix A. Rear panel connections are defined in detail in the Installation Guide for the VMIACC-0559.

PMC Site Connector

The VMICPCI-7593 supports IEEE-P1386 Common Mezzanine Card Specification with a 5 V PMC site. Figure 2-3 on page 26 shows the installation of the PMC card on the VMICPCI-7593.

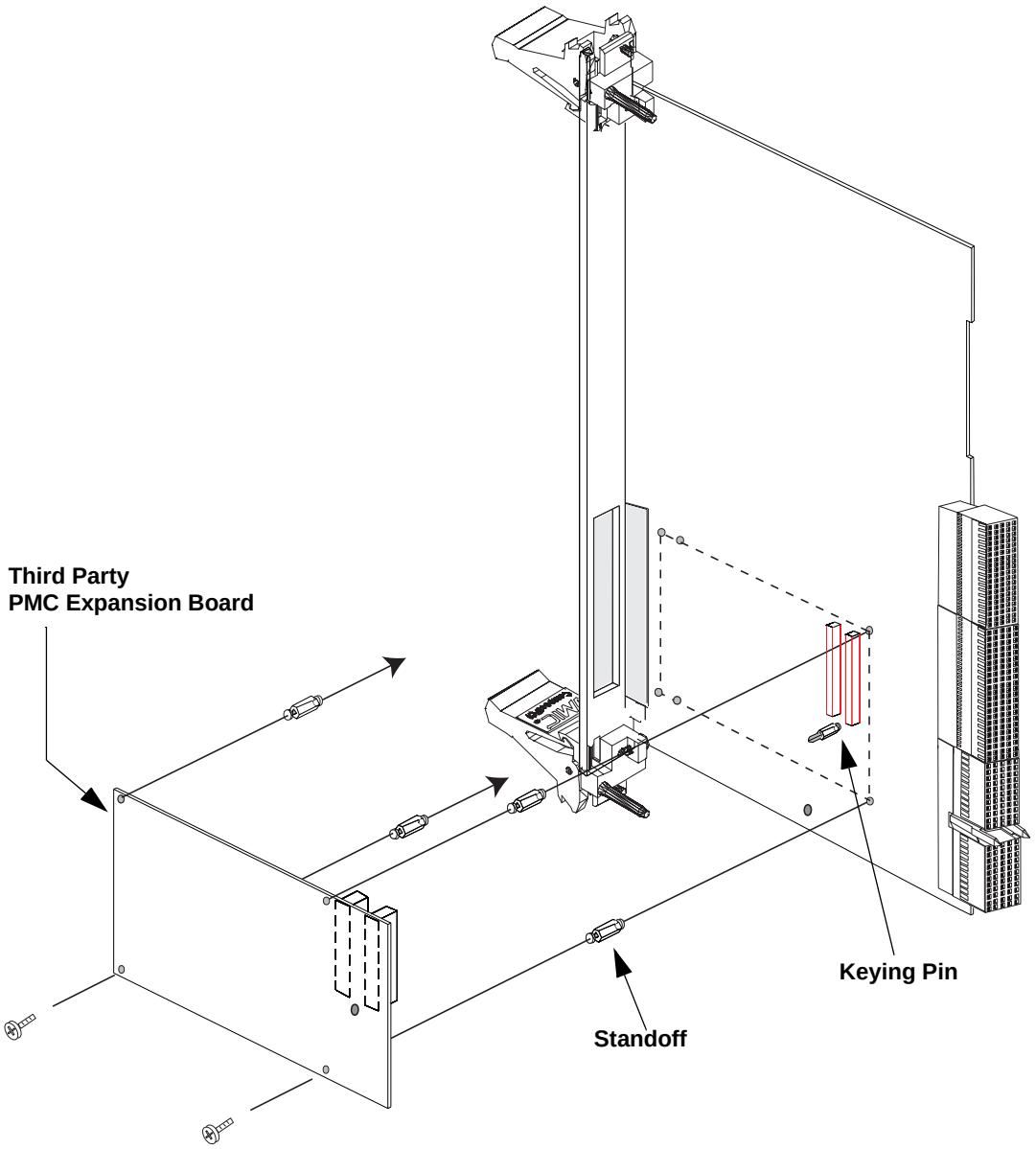
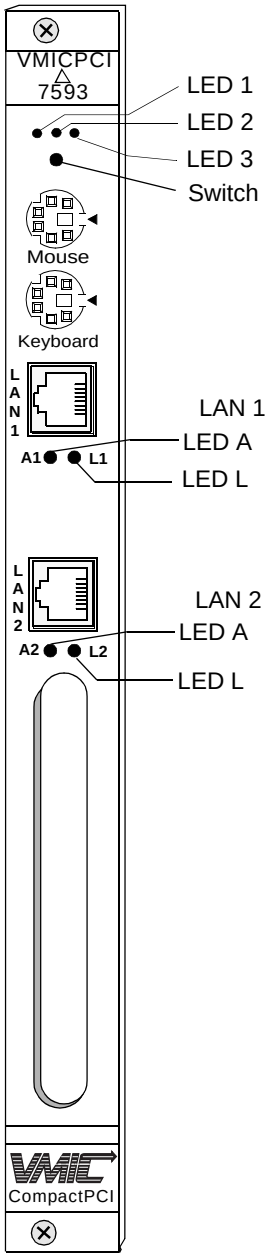


Figure 2-2 Installing a PMC Card on the VMICPCI-7593

LED Definition



LED 1	*Power* - Indicates when power is applied to the board.
LED 2	*Hard Drive Indicator* - Indicates when hard drive activity is occurring.
LED 3	*Ready* - Indicates BIOS Boot is complete, and module is ready
Switch	*Reset* - Allows the system to be reset from the front panel.
LED A1	LAN 1: *Ethernet Active* - Indicates when the Ethernet is transmitting or receiving data.
LED L1	LAN 1: *Ethernet Link* - *Yellow* indicates when the Ethernet is linked in 10BaseT mode. *Green* indicates when the Ethernet is linked in 100BaseTx mode.
LED A2	LAN 2: *Ethernet Active* - Indicates when the Ethernet is transmitting or receiving data.
LED L2	LAN 2: *Ethernet Link* - *Yellow* indicates when the Ethernet is linked in 10BaseT mode. *Green* indicates when the Ethernet is linked in 100BaseTx mode.

Figure 2-3 Front Panel LED Positions

CHAPTER
3

PC/AT Functions

Contents

CPU Socket 34
Physical Memory 34
PC / AT Interrupts 37
PCI Interrupts 40
Integrated Peripherals 43
Ethernet Controller 44

Introduction

The VMICPCI-7593 provides a complete IBM PC / AT-compatible Pentium® processor-based computer. The design includes a high-speed microprocessor with current technology memory. Reference the VMIC product specifications for available component options.

Because the design is PC / AT compatible, it retains standard PC memory and I / O maps along with standard interrupt architecture. Furthermore, the VMICPCI-7593 includes two PCI-compatible Ethernet controllers.

The following sections describe in detail the PC / AT functions of the VMICPCI-7593.

CPU Socket

The VMICPCI-7593 CPU socket is factory populated with a , high-speed Pentium processor. The CPU speed and RAM size are user-specified as part of the VMICPCI-7593 ordering information. The CPU speed is not user-upgradable.

To change CPU speeds or RAM size, contact customer service to receive a Return Material Authorization (RMA).

VMIC Customer Service is available at: 1-800-240-7782.

Physical Memory

The VMICPCI-7593 provides Synchronous DRAM (SODIMM) as on-board system memory. Memory can be accessed as bytes, words, or longwords.

The VMICPCI-7593 accepts one 144-pin SODIMM DRAM module for a maximum memory capacity of 128 Mbyte. The on-board DRAM is dual-ported to the CPCI bus through the PCI-to-PCI bridge. The memory is addressable by the local processor.



When using the Configure utility of VMIC's IOWorks Access with Windows NT 4.0 to configure RAM, do not request more than 25 percent of the physical RAM. Exceeding the 25 percent limit may result in a known Windows NT bug causing unpredictable behavior during the Windows NT boot sequence and require the use of an emergency repair disk to restore the computer. The bug is present in Windows NT 4.0 service pack level 3. It is recommended that an emergency repair disk be kept up-to-date and easily accessible.

The VMICPCI-7593 provides 16 Mbyte of flash memory (See Note) accessible through the Secondary IDE port. The VMICPCI-7593 BIOS includes an option to allow the board to boot from the flash memory or from Ethernet port 1.

The VMICPCI-7593 memory includes 32 Kbyte of battery-backed SRAM addressed at $D8000 to $DFFFF. All but the first 14 bytes are accessable as SRAM. Bytes $D8000 - $D800D are reserved for the Watchdog Timer registers. The battery-backed SRAM can be accessed by the CPU at anytime, and is used to store system data that must not be lost during power-off conditions. Battery support for the 32 Kbyte of SRAM is BT2 and uses Jumper E15.



Memory capacity may be extended as parts become available.

I/O Port Map

The Pentium processor-based CPU includes special input/output instructions that access I/O peripherals residing in I/O addressing space (separate and distinct from memory addressing space). Locations in I/O address space are referred to as *ports*. When the CPU decodes and executes an I/O instruction, it produces a 16-bit I/O address on lines A00 to A15 and identifies the I/O cycle with the M/I/O control line. Thus, the CPU includes an independent 64 Kbyte I/O address space which is accessible as bytes, words, or longwords.

Standard PC/AT hardware circuitry reserves only 1,024 byte of I/O addressing space from I/O $000 to $3FF for peripherals. All standard PC I/O peripherals such as serial and parallel ports, hard and floppy drive controllers, video system, real-time clock, system timers, and interrupt controllers are addressed in this region of I/O space. The BIOS initializes and configures all these registers properly; adjusting these I/O ports directly is not normally necessary.

The assigned and user-available I/O addresses are summarized in the I/O Address Map, Table 3-1.

Table 3-1 VMICPCI-7593 I/O Address Map

I/O ADDRESS RANGE	SIZE IN BYTES	HW DEVICE	PC/AT FUNCTION
$000 - $00F	16		DMA Controller 1 (Intel 8237A Compatible)
$010 - $01F	16		Reserved
$020 - $021	2		Master Interrupt Controller (Intel 8259A Compatible)
$022 - $03F	30		Reserved
$040 - $043	4		Programmable Timer (Intel 8254 Compatible)
$044 - $05F	30		Reserved
$060 - $064	5		Keyboard, Speaker, Eqpt. Configuration (Intel 8042 Compatible)
$065 - $06F	11		Reserved
$070 - $071	2		Real-Time Clock,
$072 - $07F	14		Reserved
$080 - $08F	16		DMA Page Registers
$090 - $091	2		Reserved
$092	1		Alt. Gate A20/Fast Reset Register
$093	1		Reserved

Table 3-1 VMICPCI-7593 I/O Address Map (Continued)

I/O ADDRESS RANGE	SIZE IN BYTES	HW DEVICE	PC/AT FUNCTION
$094	1		
$095 - $09F	11		Reserved
$0A0 - $0A1	2		Slave Interrupt Controller (Intel 8259A Compatible)
$0A2 - $0BF	30		Reserved
$0C0 - $0DF	32		DMA Controller 2 (Intel 8237A Compatible)
$0E0 - $16F	142		Reserved
$170 - $177	8	PIIX4	Secondary Hard Disk Controller
$178 - $1EF	120		User I/O
$1F0 - $1F7	8	PIIX4	Primary Hard Disk Controller
$1F8 - $277	128		User I/O
$278 - $27F	8	I/O Chip*	LPT2 Parallel I/O*
$280 - $2E7	104		Reserved
$2E8 - $2EE	7	UART*	COM4 Serial I/O*
$2EF - $2F7	9		User I/O
$2F8 - $2FE	7	Super-I/O Chip	COM2 Serial I/O (16550 Compatible)
$2FF - $36F	113		Reserved
$370 - $377	8	Super-I/O Chip	Secondary Floppy Disk Controller
$378 - $37F	8	Super-I/O Chip	LPT1 Parallel I/O
$380 - $3E7	108		Reserved
$3E8 - $3EE	7	UART*	COM3 Serial I/O*
$3F0 - $3F7	8	Super-I/O Chip	Primary Floppy Disk Controller
$3F8 - $3FE	7	Super-I/O Chip	COM1 Serial I/O (16550 Compatible)
$3FF - $4FF			Reserved
$500 - $503	4	82C54 Timer	Programmable Internal Timer
$504 - CFF			Reserved

* While these I/O ports are reserved for the listed functions, they are not implemented on the VMICPCI-7593. They are listed here to make the user aware of the standard PC/AT usage of these ports.

PCI-to-PCI Bridge

The VMICPCI-7593 uses the DEC 21153 PCI-to-PCI bridge to interface between the primary PCI bus of unit and the CompactPCI (CPCI) bus. The CompactPCI bus appears as a secondary PCI bus, and all devices in the seven peripheral slots of the CPCI chassis are auto detected by the BIOS and respond to normal PCI accesses.

PC/AT Interrupts

In addition to an I/O port address, an I/O device has a separate hardware interrupt line assignment. Assigned to each interrupt line is a corresponding interrupt vector in the 256-vector interrupt table at $00000 to $003FF in memory. The 16 maskable interrupts and the single Non-Maskable Interrupt (NMI) are listed in Table 3-2 along with their functions. Table 3-3 on page 38 details the vectors in the interrupt vector table. The interrupt number in HEX and decimal are also defined for real and protected mode in Table 3-3.

The interrupt hardware implementation on the VMICPCI-7593 is standard for computers built around the PC/AT architecture, which evolved from the IBM PC/XT. In the IBM PC/XT computers, only eight interrupt request lines exist, numbered from IRQ0 to IRQ7 at the PIC. The IBM PC/AT computer added eight more IRQx lines, numbered IRQ8 to IRQ15, by cascading a second slave PIC into the original master PIC. IRQ2 at the master PIC was committed as the cascade input from the slave PIC. This architecture is represented in Figure 3-1 on page 42.

To maintain backward compatibility with PC/XT systems, IBM chose to use the new IRQ9 input on the slave PIC to operate as the old IRQ2 interrupt line on the PC/XT Expansion Bus. Thus, in AT systems, the IRQ9 interrupt line connects to the old IRQ2 pin (pin B4) on the AT Expansion Bus (or ISA bus).

Table 3-2 PC/AT Hardware Interrupt Line Assignments

IRQ	AT FUNCTION	COMMENTS
NMI	Parity Errors (Must be enabled in BIOS Setup)	Used by VMICPCI-7593 PCIbus Interface
0	System Timer	Set by BIOS Setup
1	Keyboard	Set by BIOS Setup
2	Duplexed to IRQ9	
3	COM2/COM4	
4	COM1/COM3	
5	Timer	Assigned to On-Board Timer
6	Floppy Controller	
7	LPT1	
8	Real-Time Clock	

Table 3-2 PC/AT Hardware Interrupt Line Assignments

IRQ	AT FUNCTION	COMMENTS
9	Old IRQ2	SVGA or Network I/O
10	Not Assigned	Determined by BIOS
11	Not Assigned	Determined by BIOS
12	Mouse	
13	Math Coprocessor	
14	AT Hard Drive	
15	Flash Drive	

Table 3-3 PC/AT Interrupt Vector Table

INTERRUPT NO.		IRQ LINE	REAL MODE	PROTECTED MODE
HEX	DEC			
00	0		Divide Error	Same as Real Mode
01	1		Debug Single Step	Same as Real Mode
02	2	NMI	Memory Parity Error, CompactPCI Interrupts	Same as Real Mode (Must be enabled in BIOS Setup)
03	3		Debug Breakpoint	Same as Real Mode
04	4		ALU Overflow	Same as Real Mode
05	5		Print Screen	Array Bounds Check
06	6			Invalid OpCode
07	7			Device Not Available
08	8	IRQ0	Timer Tick	Double Exception Detected
09	9	IRQ1	Keyboard Input	Coprocessor Segment Overrun
0A	10	IRQ2	BIOS Reserved	Invalid Task State Segment
0B	11	IRQ3	COM2 Serial I/O	Segment Not Present
0C	12	IRQ4	COM1 Serial I/O	Stack Segment Overrun
0D	13	IRQ5	Timer	Same as Real Mode
0E	14	IRQ6	Floppy Disk Controller	Page Fault
0F	15	IRQ7	LPT1 Parallel I/O	Unassigned
10	16		BIOS Video I/O	Coprocessor Error

Table 3-3 PC/AT Interrupt Vector Table (Continued)

INTERRUPT NO.		IRQ LINE	REAL MODE	PROTECTED MODE
HEX	DEC			
11	17		Eqpt Configuration Check	Same as Real Mode
12	18		Memory Size Check	Same as Real Mode
13	19		XT Floppy/Hard Drive	Same as Real Mode
14	20		BIOS Comm I/O	Same as Real Mode
15	21		BIOS Cassette Tape I/O	Same as Real Mode
16	22		BIOS Keyboard I/O	Same as Real Mode
17	23		BIOS Printer I/O	Same as Real Mode
18	24		ROM BASIC Entry Point	Same as Real Mode
19	25		Bootstrap Loader	Same as Real Mode
1A	26	IRQ8	Real-Time Clock	Same as Real Mode
1B	27		Control/Break Handler	Same as Real Mode
1C	28		Timer Control	Same as Real Mode
1D	29		Video Parameter Table Pntr	Same as Real Mode
1E	30		Floppy Parm Table Pntr	Same as Real Mode
1F	31		Video Graphics Table Pntr	Same as Real Mode
20	32		DOS Terminate Program	Same as Real Mode
21	33		DOS Function Entry Point	Same as Real Mode
22	34		DOS Terminate Handler	Same as Real Mode
23	35		DOS Control/Break Handler	Same as Real Mode
24	36		DOS Critical Error Handler	Same as Real Mode
25	37		DOS Absolute Disk Read	Same as Real Mode
26	38		DOS Absolute Disk Write	Same as Real Mode
27	39		DOS Program Terminate, Stay Resident	Same as Real Mode
28	40		DOS Keyboard Idle Loop	Same as Real Mode
29	41		DOS CON Dev. Raw Output	Same as Real Mode
2A	42		DOS 3.x+ Network Comm	Same as Real Mode

Table 3-3 PC/AT Interrupt Vector Table (Continued)

INTERRUPT NO.		IRQ LINE	REAL MODE	PROTECTED MODE
HEX	DEC			
2B	43		DOS Internal Use	Same as Real Mode
2C	44		DOS Internal Use	Same as Real Mode
2D	45		DOS Internal Use	Same as Real Mode
2E	46		DOS Internal Use	Same as Real Mode
2F	47		DOS Print Spooler Driver	Same as Real Mode
30-60	48-96		Reserved by DOS	Same as Real Mode
61-66	97-102		User Available	Same as Real Mode
67-71	103-113		Reserved by DOS	Same as Real Mode
72	114	IRQ10	Not Assigned	
73	115	IRQ11	Not Assigned	
74	116	IRQ12	Mouse	
75	117	IRQ13	Math Coprocessor	
76	118	IRQ14	AT Hard Drive	
77	119	IRQ15	Flash Drive	
78-7F	120-127		Reserved by DOS	Same as Real Mode
80-F0	128-240		Reserved for BASIC	Same as Real Mode
F1-FF	241-255		Reserved by DOS	Same as Real Mode

PCI Interrupts

Interrupts on Peripheral Component Interconnect (PCI) Local Bus are optional and defined as “level sensitive,” asserted low (negative true), using open drain output drivers. The assertion and de-assertion of an interrupt line, INTx#, is asynchronous to CLK. A device asserts its INTx# line when requesting attention from its device driver. Once the INTx# signal is asserted, it remains asserted until the device driver clears the pending request. When the request is cleared, the device de-asserts its INTx# signal.

PCI defines one interrupt line for a single function device and up to four interrupt lines for a multifunction device or connector. For a single function device, only INTA# may be used while the other three interrupt lines have no meaning. Figure 3-1 on page 42 depicts the VMICPCI-7593 interrupt logic pertaining to CompactPCI operations and the PMC site.

Any function on a multifunction device can be connected to any of the INTx# lines. The Interrupt Pin register defines which INTx# line the function uses to request an interrupt. If a device implements a single INTx# line, it is called INTA#; if it implements two lines, they are called INTA# and INTB#; and so forth. For a multifunction device, all functions may use the same INTx# line or each may have its own (up to a maximum of four functions) or any combination thereof. A single function can never generate an interrupt request on more than one INTx# line.

The slave PIC accepts the CompactPCI interrupts through lines that are defined by the BIOS. The BIOS defines which interrupt line to utilize depending on which system requires the use of the line.

The PCI-to-PCI Bridge has the capability of generating a Non-Maskable Interrupt (NMI) via the PCI SERR# line. Table 3-4 describes the register bits that are used by the NMI. The SERR interrupt is routed through logic back to the NMI input line on the CPU. The CPU reads the NMI Status Control register to determine the NMI source (bits set to 1). After the NMI interrupt routine processes the interrupt, software clears the NMI status bits by setting the corresponding enable / disable bit to 1. The NMI Enable and Real-Time Clock register can mask the NMI signal and disable / enable all NMI sources.

Table 3-4 NMI Register Bit Descriptions

Status Control Register (I/O Address $061, Read/Write, Read Only)	
Bit 7	SERR# NMI Source Status (Read Only) - This bit is set to 1 if a system board agent detects a system board error. It then asserts the PCI SERR# line. To reset the interrupt, set bit 2 to 0 and then set it to 1. When writing to port $061, bit 7 must be 0.
Bit 2	PCI SERR# Enable (Read/Write) - 1 = Clear and Disable, 0 = Enable
Enable and Real-Time Clock Address Register (I/O Address $070, Write Only)	
Bit 7	NMI Enable - 1 = Disable, 0 = Enable

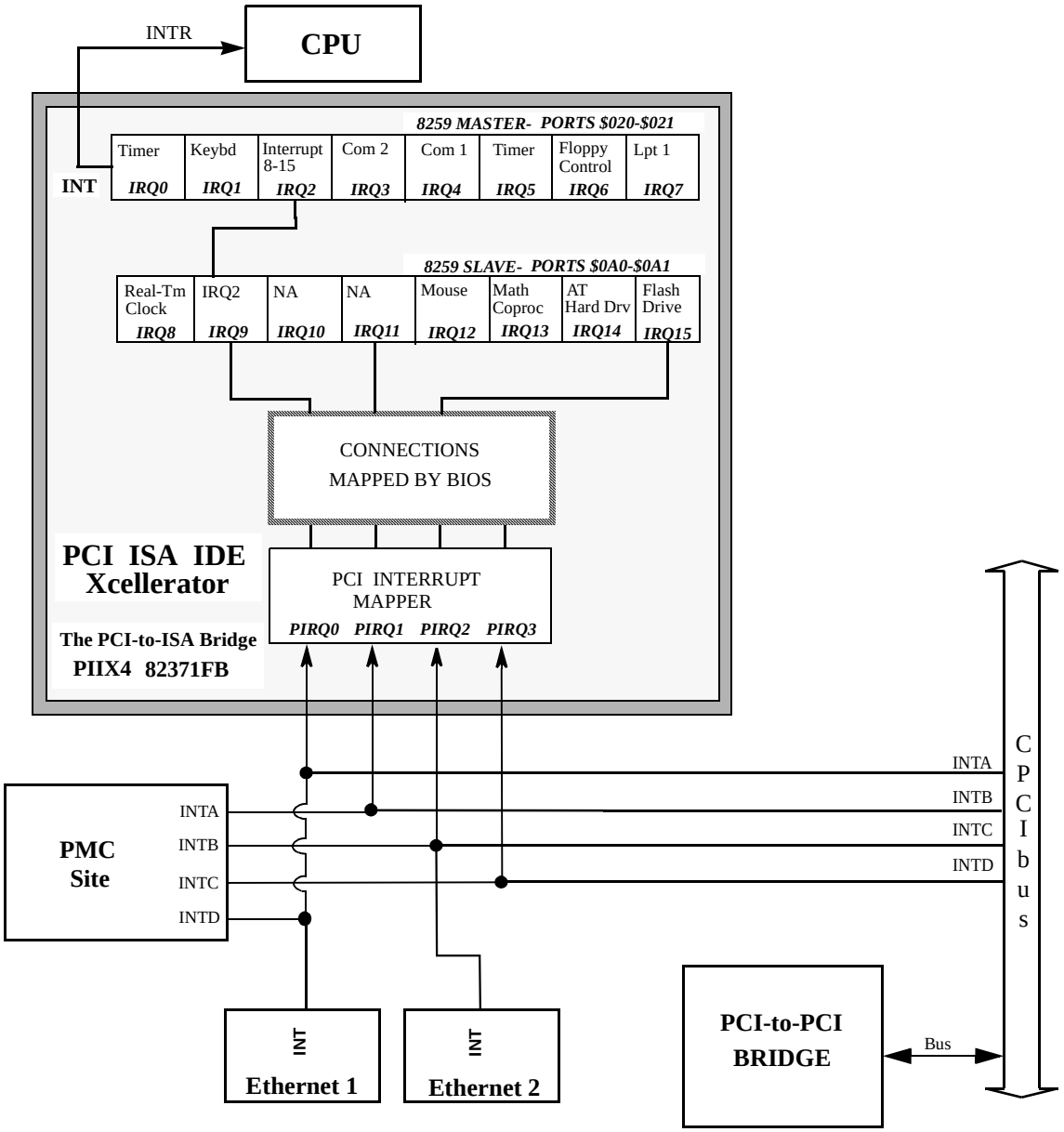


Figure 3-1 Connections for the PC Interrupt Logic Controller

Integrated Peripherals

The VMICPCI-7593 incorporates the SMC Super-I/O chip. The SMC chip provides the VMICPCI-7593 with a standard floppy drive controller, 16550 UART-compatible serial port, and one Enhance parallel port with ECP/EPP modes. The Ultra-EIDE hard drive interface is provided by the Intel 82371SB (PIIX4) PCI ISA IDE Xcelerator chip. All ports are located on the J3 connector, for use with VMIC's VMIACC-0559 CompactPCI Rear Transition Utility Board (see Figure 3-2 below). The Ultra-EIDE hard drive and floppy interface can be implemented trough on-board headers if preferred.

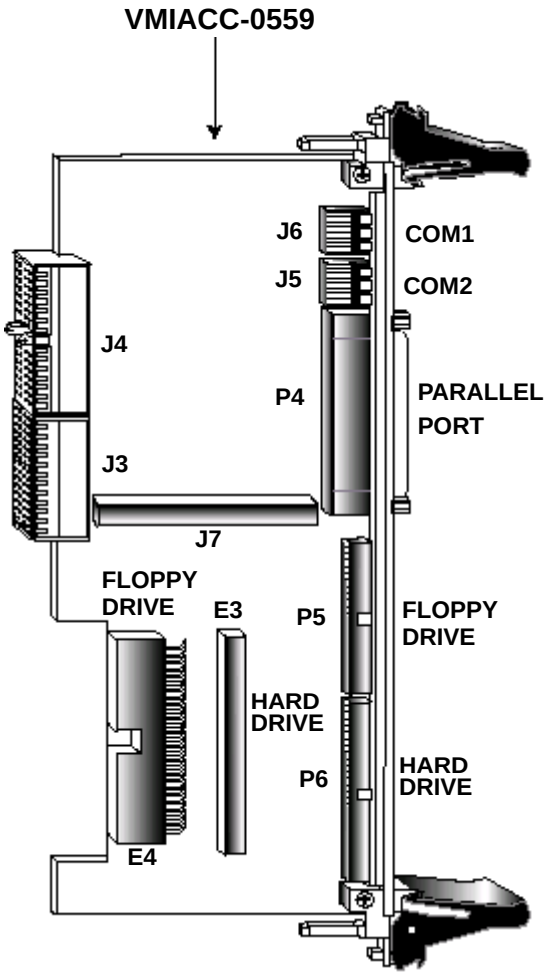


Figure 3-2 VMIACC-0559 CompactPCI Rear Transition Utility Board

Ethernet Controller

The network capability is provided by the Digital Semiconductor's 21143. This Ethernet controller is PCI bus based and is software configurable. The VMICPCI-7593 supports 10BaseT and 100BaseTx Ethernet.

10BaseT

A network based on the 10BaseT standard uses unshielded twisted-pair cables, providing an economical solution to networking by allowing the use of existing telephone wiring and connectors. The RJ-45 connector is used with the 10BaseT standard. 10BaseT has a maximum length of 100 meters from the wiring hub to the terminal node.

100BaseTx

The VMICPCI-7593 also supports the 100BaseTx Ethernet. A network based on a 100BaseTx standard uses unshielded twisted-pair cables and a RJ-45 connector. The 100BaseTx has a maximum deployment length of 250 meters.

Remote Ethernet Booting

The VMICPCI-7593 is capable of booting from a server over the Ethernet. Utilizing Lanworks BootWare®. BootWare gives you the ability to remotely boot the VMICPCI-7593 using Netware, TCP/IP, or RPL network protocols. The Ethernet must be connected through the LAN 1, RJ-45 front panel connector to boot remotely. This feature allows users to create systems without the worry of disk drive reliability, or the extra cost of adding Flash drives.

BootWare Features:

- Netware, TCP/IP, RPL Compatible
- Unparalleled boot sector virus protection
- Detailed boot configuration screens
- Comprehensive diagnostics
- Optional disabling of local boots
- Dual-boot option lets users select network or local booting

CHAPTER

4

Embedded PC/RTOS Features

Contents

Timers 46
Flash Disk 55
Watchdog Timer 59
Battery Backed SRAM 65

Introduction

VMIC's VMICPCI-7593 features additional capabilities beyond those of a typical IBM PC / AT-compatible CPU. The unit provides three software-controlled, general-purpose timers along with a programmable Watchdog timer for synchronizing and controlling multiple events in embedded applications. The VMICPCI-7593 also provides a bootable Flash Disk system, and 32 kbyte of nonvolatile, battery-backed SRAM. These features make the unit ideal for embedded applications, particularly applications where standard hard drives and floppy disk drives cannot be used.

Timers

General

The VMICPCI-7593 provides a user-programmable 82C54 internal timer/counter. The 82C54 provides three independent, 16-bit timers each operating at 1 or 2 MHz clock speed determined by the configuration of jumper E4; reference Table 2-3 on page 24 of Chapter 2. These timers are completely available to the user, and are not dedicated to any PC/AT function. These timers may be used to generate system interrupts.

Events can be timed by either polling the timers or generating a system interrupt via circuity external to the 82C54. The external circuity consists of logic which generates the interrupt and a Timer Interrupt Status register which indicates which of the three Timers generated an interrupt.

The 82C54 timers are mapped at I/O address $500. The interrupt used by the Timers is IRQ5. The Timer Interrupt Status register is available via the Power Management I/O address space. The access to this space is explained in the Timer Interrupt Status section.

Timer Interrupt Status

A single interrupt, IRQ5, is used by all three Timers. A Timer Interrupt Status register is provided in order to determine which Timer(s) initiated an interrupt. The interrupt status register is a general-purpose input register located (refer to Figure 4-1), external to the 82C54, at offset 31H from the Power Management Base I/O address. The interrupt status register address can be found by first determining the PCI Configuration Base address for Device ID 7113H and Vendor ID 8086H. The Power Management Base I/O address can be found by reading offset 40H from this PCI Configuration address. The Timer Interrupt Status register bits are located at offset 31h from the Power Management Base I/O address, bits 5, 6, and 7.

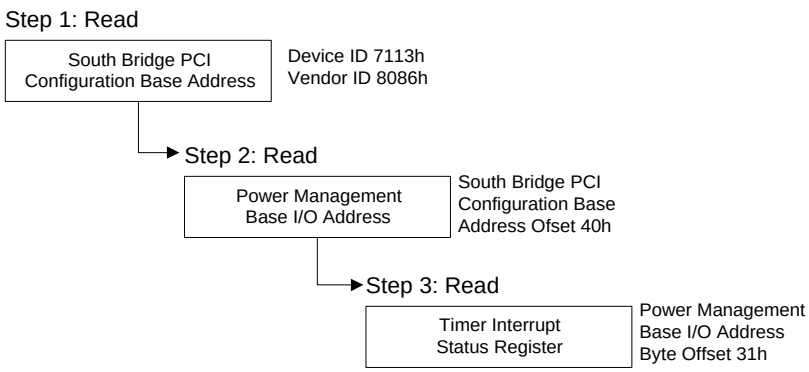


Figure 4-1 Timer Interrupt Status Register Read/Steps

A byte read of Offset 31h from the Power Management Base I/O address is used to obtain these bits. Bits 5, 6, and 7 correspond to Timers 2, 1, and 0, respectively

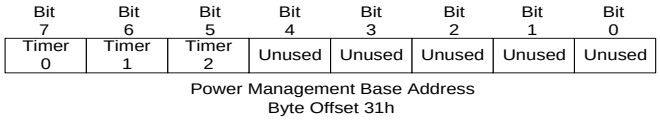


Figure 4-2 Timer Interrupt Status Register

In order to clear the Timer Interrupt Status register, first write zeros (0's) to the general-purpose output register located at offset 37h of the Power Management Base I/O address bits 3, 4, and 6 (Not bits 3, 4 and 5). Then write ones (1's) to these same bits to re-enable the Timer Interrupt Status register. Bits 3, 4, and 6 correspond to Timers 2, 1, and 0, respectively.

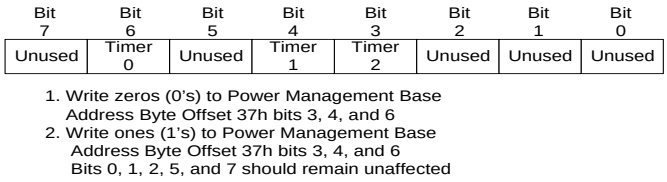


Figure 4-3 Clearing the Timer Interrupt Status Register

Clearing the Interrupt

The Timer Interrupts are cleared using the standard procedure for clearing PC/AT IRQ5. Refer to Appendix D for an example of using the 82C54 timers.

Timer Programming

Architecture

The VMICPCI-7593 Timers are mapped in I/O address space starting at $500. See Table 4-1. The Timers, consisting of three 16-bit timers and a Control Word Register (see Figure 4-4) are read from/written to via an 8-bit data bus.

Table 4-1 I/O Address of the Control Word Register and Timers

I/O Address	Select
$500	Timer 0
$501	Timer 1
$502	Timer 2
$503	Control Word Register

Table 4-1 shows the I/O address's of the Control Word Register and Timers.

The Control Word Register is write only. The Timer status information can be obtained from the Read-Back command (see the Section titled Reading on page 51).

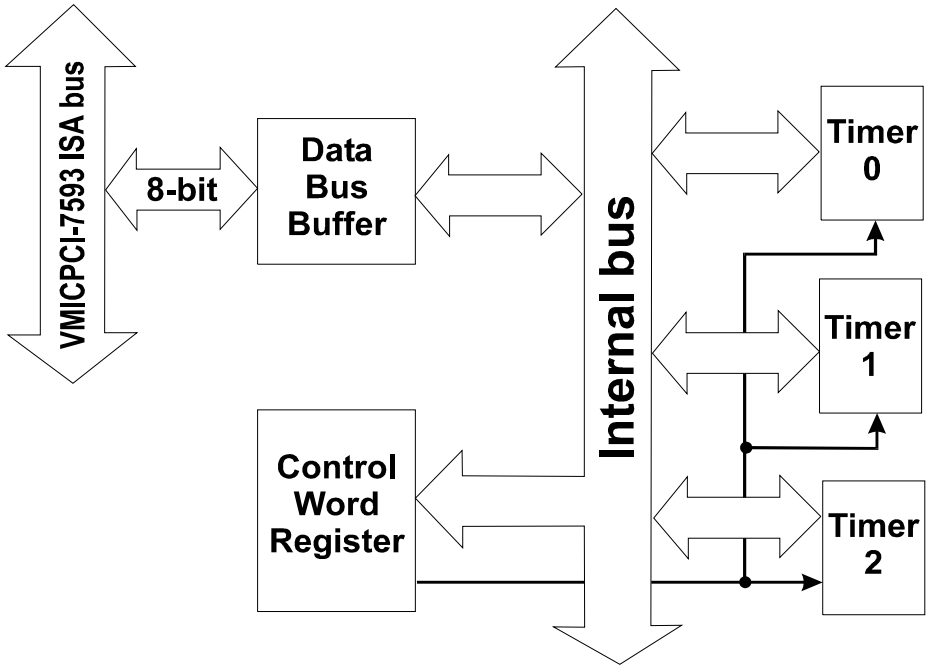


Figure 4-4 82C54 Diagram

The three Timers, Timer 0, 1, and 2, are functionally equivalent. Therefore only a single Timer will be described. Figure 4-5 is a block diagram of a Timer. Each Timer is functionally independent. Although the Control word is shown in the Timer block diagram, it is not a part of the Timer, but its contents directly affect how the Timer functions.

The status register, as shown in Figure 4-5, when latched, contains the present contents of the Control Word Register and the present state of the output and load count flag (The Status Word is available via the Read-Back command, see the Section titled Reading on page 51). The Timer is labelled TE (Timer Element). It is a 16-bit synchronous presettable down counter.

The blocks labelled OL_M and OL_L are 8-bit Output Latches (OL). The subscripts M and L stand for Most Significant byte and Least Significant byte. These latches usually track the TE but when commanded will latch and hold the present count until the CPU reads the count. When the latched count is read, the OL registers will continue to track the TE. When reading the OL registers, two 8-bit accesses must be performed to retrieve the complete 16-bit value of the Timer as only one latch at a time is enabled. The TE cannot be read, the count is read from the OL registers.

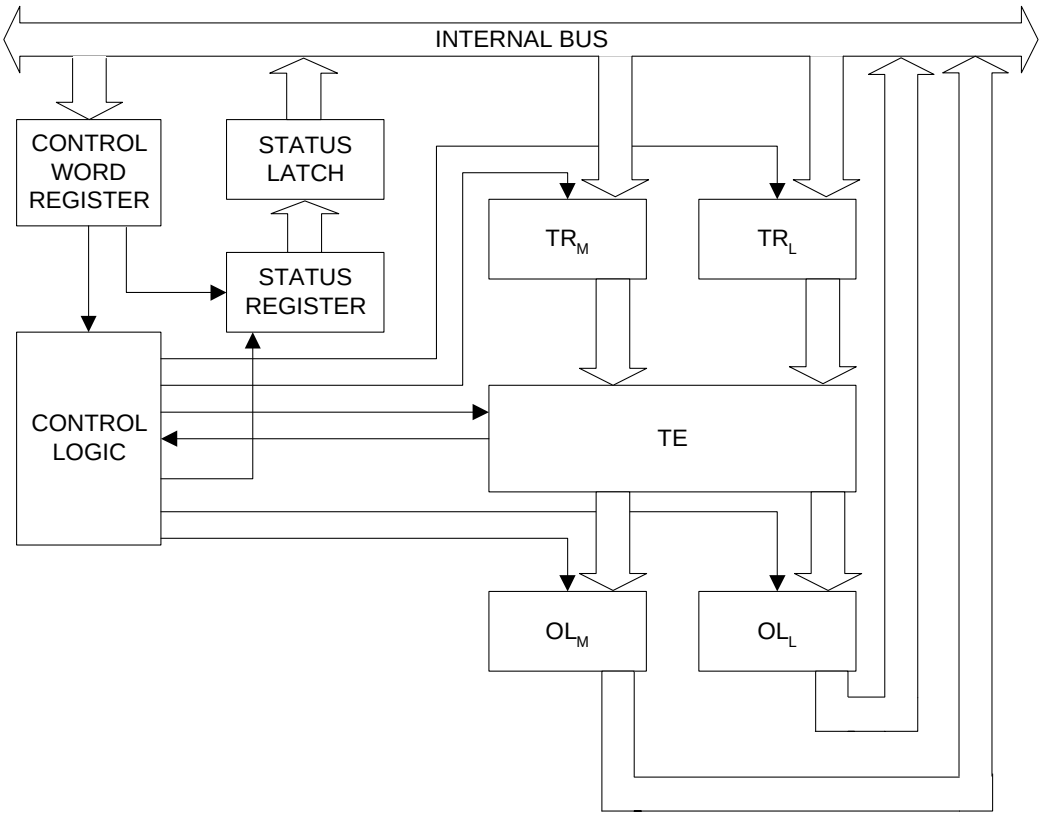


Figure 4-5 Internal Timer Diagram

There are two 8-bit registers labeled TR_M and TR_L (Timer Register). The subscripts M and L stand for Most Significant byte and Least Significant byte. When a new count is written to the Timer the count is loaded into the TR and later transferred to the TE. The Control logic lets one 8-bit TR register be written to at a time. Two 8-bit writes must be performed to load a complete 16-bit count value. Both TR bytes are transferred to the TE at the same time. The TE cannot be directly written to by the user, the count is written to the TR registers then latched to the TE.

Writing

The Timers are programmed by first writing a Control Word and then the initial count. The format of the Control Word is shown in Tables 4-2 through 4-6. All Control Words are written into the Control Word Register while the initial counts are written into the individual Timer registers. The format of the initial count is determined by the Control Word.

Table 4-2 Control Word Format

D7	D6	D5	D4	D3	D2	D1	D0
ST1	ST0	RW1	RW0	M2	M1	M0	BCD

Table 4-3 ST - Select Timer

ST1*	ST0*	Description
0	0	Select Timer 0
0	1	Select Timer 1
1	0	Select Timer 2
1	1	Read-Back Command (See Reading section on page 1-51)
* The ST bits specify which Timer (0, 1, or 2) the Control word refers to or whether this is a Read-Back command		

Table 4-4 RW - Read/Write

RW1*	RW0*	Description
0	0	Timer Latch Command (see Reading section)
0	1	Read / Write least significant byte only
1	0	Read / Write most significant byte only
1	1	Read / Write least significant byte first, then most significant
* The RW bits specify whether this is a Timer Latch command or the byte ordering of the Read / Write transaction.		

Table 4-5 M - Mode

M2*	M1*	M0*	Description
0	0	0	Mode 0
0	0	1	Mode 1
X	1	0	Mode 2
X	1	1	Mode 3
1	0	0	Mode 4
1	0	1	Mode 5
* Only Mode 2 is described in this manual.			

Table 4-6 BCD

BCD*	Description
0	Binary Timer 16-bits
1	Binary Coded Decimal (BCD) Timer (4 Decades)
* The BCD bit specifies whether the Timer count value is in Binary or BCD.	

When programming the 82C54, only two rules need to be followed.

1. For each Timer, the Control Word must be written first.
2. The initial count must follow the format specified in the Control Word (least significant byte only, most significant byte only, or least significant byte then most significant byte). As long as these rules are adhered to, any programming sequence is acceptable.

Reading

There are two methods for reading the timers: the Timer Latch Command and the Read-Back Command.

Timer Latch Command

The Timer Latch Command allows the reading of a Timer ‘on the fly’ without affecting the timing in process.

Like a Control Word, the Timer Latch Command is written to the Control Word Register (I/O Address $503, see Table 4-1). The Select Timer bits (ST1, ST0, see Table 4-3) select one of the three timers while the Read/Write bits (RW1,RW0, see Table 4-4) select the Timer Latch Command, RW1 = 0 and RW0 = 0. The selected Timer's count is latched into the 0L registers at the time of the Timer Latch Command. The count is held in the 0L latches until it is read. Multiple Timer Latch Commands can be used to latch more than one Timer. Again, each Timer's count is held latched until it is read.

Read-Back Command

The Read-Back Command allows the user to view the Timer count, the Timer Mode, the current state of the OUT pin, and the Load Flag of the selected Timer. Like a Control Word, the Read-Back Command is written into the Control Word Register and has the format shown in Tables 4-7 and 4-8. The Command applies to the Timer(s) selected by setting the corresponding bits Cnt2, Cnt1, Cnt0 = 1.

Table 4-7 Read-Back Command Format

D7	D6	D5	D4	D3	D2	D1	D0
1	1	$\overline{\text{Count}}$	$\overline{\text{Status}}$	Cnt2	Cnt1	Cnt0	0

Table 4-8 Read-Back Command Description

Bit	Description
D5: $\overline{\text{Count}}$	Latch count of selected Timer(s)
D4: $\overline{\text{Status}}$	Latch status of selected Timer(s)
D3: Cnt2	Select Timer 2
D2: Cnt1	Select Timer 1
D1: Cnt0	Select Timer 0
D0	Reserved, must be 0

The Read-Back Command can be used to latch several Timer counts by setting the $\overline{\text{Count}}$ bit = 0 and selecting the Timers. This is the same as using multiple Timer Latch Commands. Again each Timer's latched count will be held until it is read.

The Read-Back command can also be used to latch the timer status by setting the $\overline{\text{Status}}$ bit = 0 and selecting the Timers. Status of a Timer is accessed by a read from that Timer (see Table 4-1 on page 47). If more than one Timer Status Read-Back command is issued without reading the status, all but the first is ignored.

The format of the Timer Status byte is shown in Tables 4-9 and 4-10.

Table 4-9 Status Byte

D7	D6	D5	D4	D3	D2	D1	D0
OUT	LOAD	RW1	RW0	M2	M1	M0	BCD

Table 4-10 Status Byte Description

Bit	Description
D7: OUT	Current state of Timers OUT pin
D6: LOAD	Count loaded into Timer
D5-D0	Timer Programmed Mode

Bit D7 contains the state of the Timers OUT pin. This allows viewing of the Timer's OUT pin via software.

Bit D6 indicates when the count written to the Timer is actually loaded into the Timer register. The exact time of the loading depends on the Mode the Timer is in and is defined in the Mode Definitions section. The count cannot be read from the Timer until it has been loaded. If a count is read before this time, the value read will not be the new count just written. Refer to Table 4-11.

Bits D5 through D0 contain the Timer's programmed mode exactly, bit for bit, like the Timer Control Words bits D5 through D0. See Table 4-2 on page 50.

Table 4-11 LOAD Bit Operation

Action	Causes
1. Write to the Control Word Register [1]	LOAD bit = 1
2. Write count to Timer [2]	LOAD bit = 1
3. New count loaded into Timer	LOAD bit = 0

[1]Only the Timer specified in the Control Word will have its LOAD bit set to 1. LOAD bits of other Timers are not affected.

[2]If the Timer is programmed for two byte counts (least significant then most significant), the LOAD bit will go to 1 when the second byte is written.

Both the count and status of the specified Timer(s) can be latched at the same time by setting both the Count bit (D5) and Status bit (D4) to zero (0) in the Read-Back command. If this technique is used, the first read operation of the Timer will return the status while the next one or two reads (depending on whether the Timer is programmed for one or two bytes) will return the count. Succeeding reads will return unlatched counts.

Mode Definitions

The VMICPCI-7593 utilizes an 82C54 Timer / Counter for its Timers. 82C54 Timer / Counters can be programmed to function in six different modes (numbered Mode 0 through Mode 5). The VMICPCI-7593 Timers are hardware configured to operate using Mode 2. Only Mode 2 is defined.

Mode 2 functions as a divide by N counter. Once a Control Word and an initial count are written to the Timer the initial count is loaded on the next Clock cycle. When the count decrements to 1 an interrupt is generated. The Timer then reloads the initial count and the process repeats. This Mode is periodic. For an initial count of N, the sequence repeats every N CLK cycles. An initial count of 1 is illegal.

Writing a new count while the Timer is counting does not affect the current sequence. The new count will be loaded at the end of current sequence.

Flash Disk

The VMICPCI-7593 features an on-board 16 Mbyte flash mass storage system. This Flash Disk appears to the user as an intelligent ATA (IDE) disk drive with the same functionality and capabilities as a “rotating media” IDE hard drive. The VMICPCI-7593 BIOS includes an option to allow the board to boot from the Flash Disk.

Configuration

The Flash Disk resides on the VMICPCI-7593 as the secondary IDE bus master device (the secondary IDE bus slave device is not assignable). The default setting in the AWARD BIOS ‘STANDARD CMOS SETUP’ screen is the ‘AUTO’ setting. In the AWARD BIOS ‘PERIPHERAL SETUP’ screen, the secondary PCI IDE interface must be enabled for the Flash Disk to be functional. Refer to Appendix D, Award-Basic Input/Output System (BIOS) for additional details.

Figure 4-6 maps the configuration possibilities for a typical system consisting of the VMICPCI-7593 with a resident Flash Disk, a hard drive attached to the Primary IDE interface, and a floppy drive attached to the floppy interface.

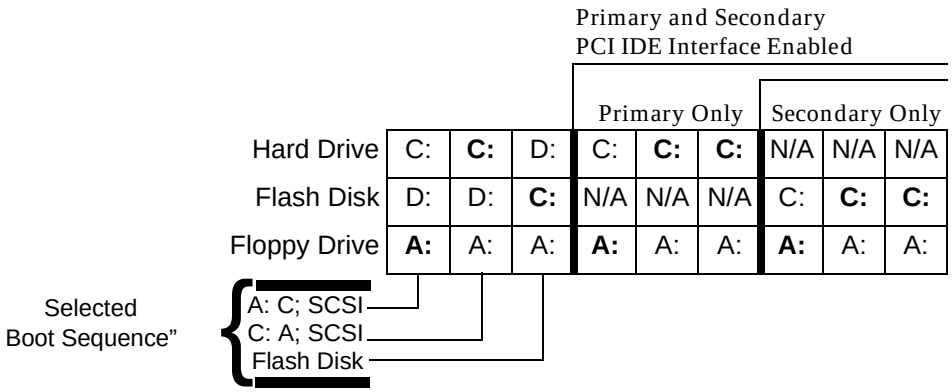


Figure 4-6 Typical System Configuration

The Primary and Secondary PCI IDE Interfaces are controlled (enabled or disabled) in the Integrated Peripheral Setup screen of the AWARD BIOS. The First Boot Device is selected in the BIOS Features Setup screen.

Figure 4-6 identifies the drive letter assigned to each physical device, and indicates in bold lettering the device booted from in each configuration, using devices that were bootable. Bootable being a device on which an operating system has been installed, or formatted as a system disk using MS-DOS.

Note

If during the configuration efforts a drive device is identified as a USER setting in the Standard CMOS Setup screen, as occurs after using the Auto-Detect Hard Drives function of the BIOS, then disabled (the device's IDE interface disabled) in the Peripheral Setup screen, the following error may occur:

Primary (or Secondary) Master HHD Error
Run Setup
Press <ESC> to Resume

Press the Escape key as directed to resume the boot procedure. The settings should be changed to prevent the reoccurrence of this error condition

Functionality

The Flash Disk performs identically to a standard IDE hard drive. Reads and writes to the device are performed using the same methods, utilizing DOS command line entries or the file managers resident in the chosen operating system.

Advanced Configuration

The previous discussion is based on using the IDE disk devices formatted as one large partition per device. Some applications may require the use of multiple partitions. The following discussion of these partitions includes special procedures that must be followed to create multiple partitions on the VMICPCI-7593 IDE disk devices (including the resident Flash Disk).

Partitions may be either a primary or an extended partition. An extended partition may be subdivided farther into logical partitions. Each device may have up to four main partitions, one of which may be an extended partition. However, if multiple primary partitions are created, only one partition may be active at one time. Data in the non-active partitions are not accessible.

Following the creation of the partitioning scheme, the partitions can be formatted to contain the desired file system.

As discussed earlier, a typical system consists of the VMICPCI-7593 with its resident Flash Disk configured as the Secondary IDE device, a hard drive attached to the Primary IDE interface, and a floppy drive attached to the floppy interface.

Using this configuration, it may be desirable to have a logical device on either IDE device, configured as a bootable device, allowing the selection of the first boot device by way of the Advanced CMOS Setup screen. Using this capability, a user could have a system configured with multiple operating systems that would be selectable by assigning the IDE logical device as the boot device.

The DOS utility FDISK is commonly used to configure the partition structure on a hard drive. Comments on the following page pertain to partitioning efforts using FDISK.

CAUTION: Deleting a partition will erase all the data previously stored in that partition.

The Flash Disk will be configured as a single partition device as delivered from the factory. The following sample sequence illustrates a proven method for creating two 8 Mbyte partitions, with one as an active primary partition. Take note of the instructions to exit FDISK. This has been shown to be an important step in a successful partitioning effort.

1. Power up the VMICPCI-7593, and enter the CMOS set-up.
2. Set IDE HDD Master to “Not Installed”.
3. Set Flash Disk Master to “AUTO”.
4. Set boot device to floppy.
5. Boot DOS from the floppy, verify that the System Configuration Screen shows only the Flash Disk.
6. Run FDISK.
7. Delete all current partitions (any data currently stored in the partitions will be lost).
8. Exit FDISK, this will cause a reboot, then run FDISK again.
9. Create a 8 Mbyte primary partition.
10. Create a 8 Mbyte extended partition.
11. Set-up a logical device for the 8 Mbyte extended partition.

Note

If only one partition is required it will be a 16 Mbyte partition.

12. Set the Primary partition as an active partition.
13. Exit FDISK.

Note

If an operating system has been installed on the Flash Disk that modifies the Master Boot Record (MBR), then the following step is required to rewrite the MBR for DOS.

14. Run FDISK /MBR
15. Run FORMAT C: (use the extension /s option if you want the Flash Disk as a bootable DOS device.)
16. Format D: (this is only required if two partitions were created).
17. Reset the CPU, and enter the CMOS set-up.

18. Set Primary Master to “AUTO”.
19. Set boot device to desired boot source.

Drive letter assignments for a simple system were illustrated in Figure 4-6. Understanding the order the operating system assigns drive letters is necessary for these multiple partition configurations. The operating system assigns drive letter C: to the active primary partition on the first hard disk (the boot device). Drive D: is assigned to the first recognized primary partition on the next hard disk. The operating system will continue to assign drive letters to the primary partitions in an alternating fashion between the two drives. The next logical partitions will be assigned drive letters starting on the first hard drive lettering each logical device sequentially until all are assigned a drive letter, then doing the same sequential lettering of each logical partition on the second hard disk.



Drive letter changes caused by adding an addition drive or changing the initial partitioning scheme may cause difficulties with an operating system installed prior to the changes. Plan your configuration prior to installing the operating system to minimize difficulties.

Watchdog Timer

The VMICPCI-7593 utilizes a Dallas DS1384 Watchdog Timekeeping Controller as its Watchdog Timer. The device provides a Time of Day feature, a Watchdog Alarm and an Non-Volatile SRAM controller. The Time of Day feature found within the DS1384 device is explained in this section. The actual Time of Day registers used by the VMICPCI-7593 are located at the standard PC / AT I / O address. The Time of Day feature in the DS1384 Watchdog Timer is available for use by the user at their discretion.

The Non-Volatile SRAM is explained in the Battery Backed SRAM section of this manual.

The Watchdog Timer provides a Watchdog alarm window and interval timing between 0.01 and 99.99 seconds. If enabled, and if jumper E6 is loaded, the Watchdog alarm will reset the CPU to a known state if not accessed during the alarm window.

Figure 4-7 shows a generalized block diagram of how the Watchdog Timer is used in the VMICPCI-7593. The Watchdog Timer registers are memory-mapped in the bottom fourteen locations of battery-backed SRAM addresses $D8000 through $D800D. Table 4-12 on page 60 shows the address, content and the range of each Watchdog Register.

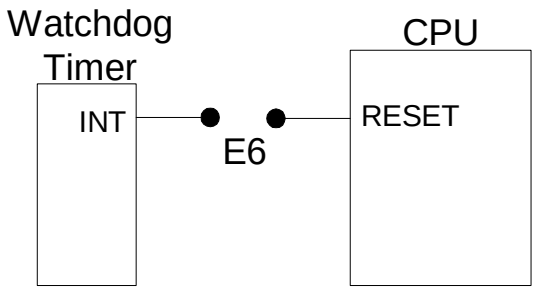


Figure 4-7 Watchdog Alarm Block

Table 4-12 Watchdog Registers

Register	Address	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Range
0	$D8000	0.1 Seconds (BCD)				0.01 Seconds (BCD)				00 - 99
1	$D8001	10 Seconds (BCD)				Seconds (BCD)				00 - 59
2	$D8002	10 Minutes (BCD)				Minutes (BCD)				00 - 59
3	$D8003	M	10 Minute Alarm (BCD)			Minute Alarm (BCD)				00 - 59
4	$D8004	0	12/24	AM/PM *	10 Hr	Hours (BCD)				
5	$D8005	M	12/24	AM/PM *	10 Hr	Hour Alarm (BCD)				
6	$D8006	0	0	0	0	Days (BCD)				01 - 07
7	$D8007	M	0	0	0	Day Alarm (BCD)				01 - 07
8	$D8008	0	0	10 Date (BCD)		Date (BCD)				01 - 31
9	$D8009	Eosc	1**	0	10 Mo	Months (BCD)				01 - 12
A	$D800A	10 Years (BCD)				Years (BCD)				00 - 99
B	$D800B	Te	Ipsw	Ibh/lo	Pu/lvl	Wam	Tdm	Waf	Tdf	
C	$D800C	0.1 Seconds (BCD)				0.01 Seconds (BCD)				00 - 99
D	$D800D	10 Seconds (BCD)				Seconds (BCD)				00 - 99

* In the 12 hour mode bit 5 determines AM (0) or PM (1). In the 24 hour mode bit 5 combines with Bit 4 to represent the 10 hour value.
** Bit 6 of Register 9 must be set to a 1. If set to a 0, an unused square wave will be generated in the circuit.

Registers 0 through A are Clock, Calendar, Time of Day Registers.
Register B is the Command Register.
Registers C and D are Watchdog Alarm Registers.

The Watchdog Timer contains 14 registers which are 8-bits wide. These registers contain all of the Time of Day, Alarm, Watchdog, Control, and Data information. The Clock Calendar, Alarm, and Watchdog Registers have both external (user accessible) and internal memory locations containing copies of the data. The external memory locations are independent of the internal functions except that they are updated periodically by the transfer of the incremented internal values. Registers 0, 1, 2, 4, 6, 8, 9, and A contain Time of Day and Data information in Binary Coded Decimal (BCD). Registers 3, 5, and 7 contain the Time of Day Alarm information in BCD. The Command Register (Register B) contains data in binary. The Watchdog Alarm Registers are Registers C and D and information stored in these registers is in BCD.

Time of Day Registers

Registers 0, 1, 2, 4, 6, 8, 9, and A contain Time of Day data in BCD.

Register 0 contains two Time of Day values. Bits 3 - 0 contain the 0.01 Seconds value with a range of 0 to 9 in BCD while Bits 7 - 4 contain the 0.1 Seconds value with a range of 0 to 9 in BCD. This Register has a total range of 0.00 to 0.99 Seconds.

Register 1 contains two Time of Day values. Bits 3 - 0 contain the 1 Seconds value with a range of 0 to 9 in BCD while Bits 7 - 4 contain the 10 Seconds value with a range of 0 to 5 in BCD. This Register has a total range of 0.0 to 59.0 Seconds. Bit 7 of this register will always be zero regardless of what value is written to it.

Register 2 contains two Time of Day values. Bits 3 - 0 contain the 0.01 Seconds value with a range of 0 to 9 in BCD while Bits 7 - 4 contain the 0.1 Seconds value with a range of 0 to 9 in BCD. This Register has a total range of 0.00 to 0.99 Seconds. Bit 7 of this register will always be zero regardless of what value is written to it.

Register 4 contains the Hours value of the Time of Day. The Hours can be represented in either 12 or 24 hour format depending on the state of Bit 6. When Bit 6 is set to a one (1) the format is 12 hour. When Bit 6 is set to a zero (0) the format is 24 hour. For the 12 hour format Bits 3 - 0 contain the 1 Hour value with a range of 0 to 9 in BCD and Bit 4 contains the 10 Hour value with a range of 0 to 1. In the 12 hour format Bit 5 is used as the AM / PM bit. When AM Bit 5 is a zero (0) and when PM bit 5 is a one (1). The total range of this register in the 12 hour format is 01 AM to 12 AM and 01 PM to 12 PM.

When Register 4 is in 24 hour format (Bit 6 is set to a zero (0)) Bits 3 - 0 contain the 1 Hour value with a range of 0 to 9 in BCD, Bit 5 combines with 4 to represent the 10 Hour value. The 10 Hour range is from 0 to 2. The total range of register 4 in the 24 hour format is 00 to 23 hours. Bit 7 of register 4 will always be zero regardless of what value is written to it and regardless of format (12 or 24 hour).

Register 6 contains the Days value of the Time of Day. Bits 2 - 0 contain the Days value with a range of 1 to 7 in BCD.

Register 8 contains two Time of Day values. Bits 3 - 0 contain the Date value with a range of 0 to 9 in BCD while Bits 5 - 4 contain the 10 Date value with a range of 0 to 3. This Register has a total range of 01 to 31. Bits 7 - 6 of this register will always be zero regardless of what value is written to it.

Register 9 contains two Time of Day values. Bits 3 - 0 contain the Months value with a range of 0 to 9 in BCD while Bits 4 contain the 10 Date value with a range of 0 to 1. This Register has a total range of 01 to 12. Bit 5 will always be zero regardless of what value is written to it. Bit 6 is unused but must be set to a 1. Bit 7, $\overline{\text{Eosc}}$, is the clock oscillator enable bit. When this bit is set to a zero (0) the oscillator is internally enabled. When set to a one (1) the oscillator is internally disabled. The oscillator via this bit is usually turned on once during system initialization but can be toggled on and off at the users discretion.

There are two techniques for reading the Time of Day from the Watchdog Timer. The first is to halt the external Time of Day registers from tracking the internal Time of Day registers by setting the Te bit (Bit 7 of the Command Register) to a logic zero (0) then reading the contents of the Time of Day registers. Using this technique eliminates the chance of the Time of Day changing while the read is taking place. At the end of the read, the Te bit is set to a logic one (1) allowing the external Time of Day registers to resume tracking the internal Time of Day Registers. No time is lost as the internal Time of Day Registers continue to keep time while the external Time of Day registers are halted. This is the recommended method.

The second technique for reading the Time of Day from the Watchdog Timer is to read the external Time of Day registers without halting the tracking of the internal Registers. This is not recommended as the registers may be updated while the reading is taking place, resulting in erroneous data being read.

Time of Day Alarm Registers

Registers 3, 5, and 7 are the Time of Day Alarm registers and are reformatted similar to Register 2, 4, and 6 respectively. Bit 7 of registers 3, 5, and 7 is a mask bit. The mask bits, when active (logic one (1)), disable the use of the particular Time of Day Alarm register in the determination of the Time of Day Alarm (see Table 4-13). When all the mask bits are low (0) an alarm will occur when Registers 2, 4, and 6 match the values found in registers 3, 5, and 7. When Register 7's mask bit is set to a logic one (1) register 6 will be disregarded in the determination of the Time of Day alarm and an alarm will occur everyday. When registers 7 and 5's mask bit is set to a logic one (1), Register 6 and 4 will be disregarded in the determination of the Time of Day alarm and an alarm will occur every hour. When Registers 7, 5 and 3's mask bit is set to a logic one (1), Register 6, 4, and 2 will be disregarded in the determination of the Time of Day alarm and an alarm will occur every minute (when register 1's seconds step from 59 to 00).

Table 4-13 Time of Day Alarm Registers

Register			Comment
Minutes	Hours	Days	
1	1	1	Alarm once per minute
0	1	1	Alarm when minutes match
0	0	1	Alarm when hours and minutes match
0	0	0	Alarm when hours, minutes, and days match

The Time of Day Alarm registers are read and written to in the same format as the Time of Day registers. The Time of Day alarm flag and interrupt are cleared when the alarm registers are read or written.

Watchdog Alarm Registers

Register C contains two Watchdog alarm values. Bits 3 - 0 contain the 0.01 Seconds value with a range of 0 to 9 in BCD while Bits 7 - 4 contain the 0.1 Seconds value with a range of 0 to 9 in BCD. This Register has a total range of 0.00 to 0.99 Seconds.

Register D contains two Watchdog Alarm values. Bits 3 - 0 contain the 1 Second value with a range of 0 to 9 in BCD while Bits 7 - 4 contain the 10 Seconds value with a range of 0 to 9 in BCD. This Register has a total range of 00.0 to 99.0 Seconds.

The Watchdog Alarm Registers can be read or written in any order. When a new value is entered or the Watchdog registers are read, the Watchdog Timer will start counting down from the entered value. When zero is reached the Watchdog Interrupt Output will go active. If jumper J30 is loaded, the CPU will reset to a known state, refer to Figure 4-7 on page 59. The Watchdog Timer count is reinitialized back to the entered value, the Watchdog flag bit is cleared, and the Watchdog interrupt output is cleared every time either of the registers are accessed. Periodic accesses to the Watchdog Timer will prevent the Watchdog Alarm from occurring. If access does not occur, the alarm will be repetitive. The Watchdog Alarm Register always reads the entered value. The actual countdown value is internal and not accessible to the user. Writing zero's to Registers C and D will disable the Watchdog Alarm feature.

Command Register

Register B is the Command Register. Within this register are mask bits, control bits, and flag bits. The following paragraphs describe each bit.

Te - Bit 7 Transfer Enable - This bit enables and disables the tracking of data between the internal and external registers. When set to a logic zero (0), tracking is disabled that is the data in the external register is frozen. When set to a logic one (1), tracking is enabled. This bit must be set to a logic one (1) to allow the external register to be updated.

Ipsw - Bit 6 Interrupt Switch - This bit toggles the Interrupt Output between the Time of Day Alarm and the Watchdog Alarm. When set to a logic zero (0), the Interrupt Output is from the Watchdog Alarm. When set to a logic one (1), the Interrupt Output is from the Time of Day Alarm.

Ibh/lo - Bit 5 Reserved - This bit should be set to a logic low (0).

Pu/lvl - Bit 4 Interrupt Pulse Mode or Level Mode - This bit determines whether the Interrupt Output will output as a pulse or a level. When set to a logic zero (0), Interrupt Output will be a level. When set to a logic one (1), Interrupt Output will be a pulse. In pulse mode the Interrupt Output will sink current for a minimum of 3 ms. This bit should be set to a logic one (1).

Wam - Bit 3 Watchdog Alarm Mask - Enables / Disables the Watchdog Alarm to Interrupt Output when Ipsw (Bit 6, Interrupt Switch) is set to logic zero (0). When set to a logic zero (0), Watchdog Alarm Interrupt Output will be enabled. When set to a logic one (1), Watchdog Alarm Interrupt Output will be disabled.

Tdm - Bit 2 Time of Day Alarm Mask - Enables / Disables the Time of Day Alarm to Interrupt Output when Ipsw (see Bit 6, Interrupt Switch) is set to logic one (1). When set to a logic zero (0), Time of Day Alarm Interrupt Output will be enabled. When set to a logic one (1), Time of Day Alarm Interrupt Output will be disabled.

Waf - Bit 1 Watchdog Alarm Flag - This is a read-only bit set to a logic one (1) when a Watchdog Alarm Interrupt occurs. This bit is reset when any of the Watchdog Alarm registers are accessed. When the Interrupt Output is set to Pulse Mode (see Bit 4, Interrupt Pulse Mode or Level Mode), the flag will be set to a logic one (1) only when the Interrupt Output is active.

Tdf - Bit 0 Time of Day Alarm Flag - This is a read-only bit set to a logic one (1) when a Time of Day Alarm Interrupt occurs. This bit is reset when any of the Time of Day Alarm registers are accessed. When the Interrupt Output is set to Pulse Mode (see Bit 4, Interrupt Pulse Mode or Level Mode), the flag will be set to a logic one (1) only when the Interrupt Output is active.

Battery Backed SRAM

The VMICPCI-7593 includes 32 Kbyte of battery-backed SRAM addressed at $D8000 to $DFFFF (the lower 14 bytes, $D8000 to $D800D are dedicated to Watchdog Timer registers and are unavailable for general use. See the Watchdog Timer section). The battery-backed SRAM can be accessed by the CPU at anytime, and can be used to store system data that must not be lost during power-off conditions.

CHAPTER

5

Maintenance

If a VMIC product malfunction, please verify the following:

1. Software resident on the product
2. System configuration
3. Electrical connections
4. Jumper or configuration options
5. Boards are fully inserted into their proper connector location
6. Connector pins are clean and free from contamination
7. No components or adjacent boards were disturbed when inserting or removing the board from the chassis
8. Quality of cables and I/O connections

If products must be returned, contact VMIC for a Return Material Authorization (RMA) Number. **This RMA Number must be obtained prior to any return**.

VMIC Customer Service is available at: 1-800-240-7782.
Or E-mail us at customer.service@vmic.com

Maintenance Prints

User level repairs are not recommended. The drawings and diagrams in this manual are for reference purposes only.

Appendix

A

Connector Pinouts

Contents

J1 Connector Pinout 71
J2 Connector Pinout 72
J3 Connector Pinout 73
Ethernet Connector Pinout (J6 and J7) 74
Keyboard and Mouse Connectors and Pinout. 75
Floppy Interface Connector Pinout 76
EIDE Hard Drive Connector Pinout 77
PMC Connector Pinout 78

Introduction

The VMICPCI-7593 PC / AT-Compatible CompactPCI® Controller has several connectors for its I / O ports. Figure A-1 shows the locations of the connectors on the VMICPCI-7593. Wherever possible, the VMICPCI-7593 uses connectors and pinouts typical for any desktop PC. This ensures maximum compatibility with a varity of systems.

Connector diagrams in this appendix are generally shown in a natural orientation with the controller board mounted in a CPCI bus chassis.

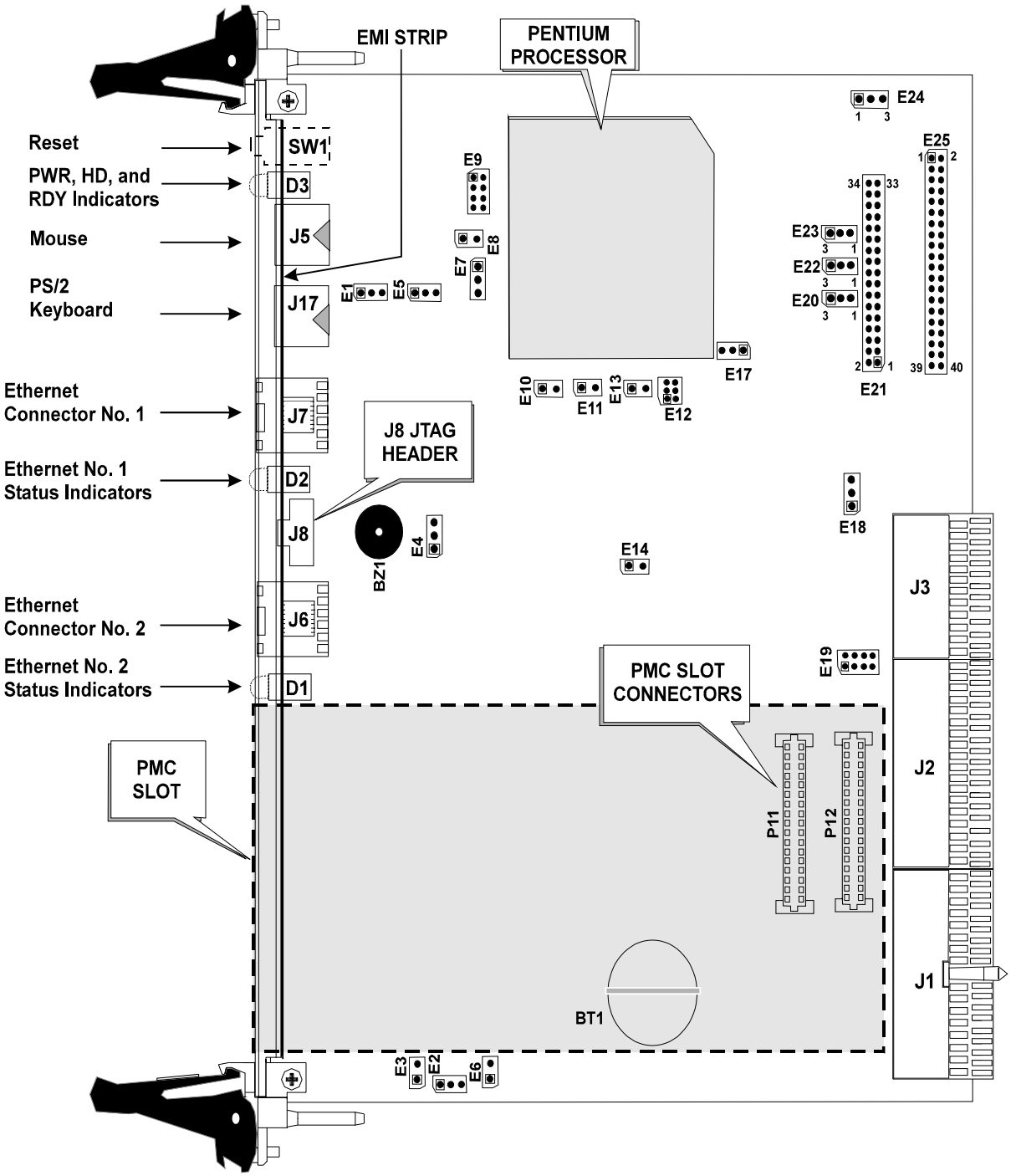


Figure A-1 VMICPCI-7593 Connector and Jumper Locations

J1 Connector Pinout

The VMICPCI-7593 utilitizes a high-density 110-pin, low inductance, and controlled impedance connector. This connector meets the IEC-1076 international standard for CompactPCI connectors. An additional external metal shield is required. The large number of ground pins ensures adequate shielding and grounding for low ground bounce and reliable operation in noisy environments. The key prevents mis-alignment of the board when installing in the chassis. Figure A-2 below depicts the J1 connector, with Table A-1 showing the connector pinout.

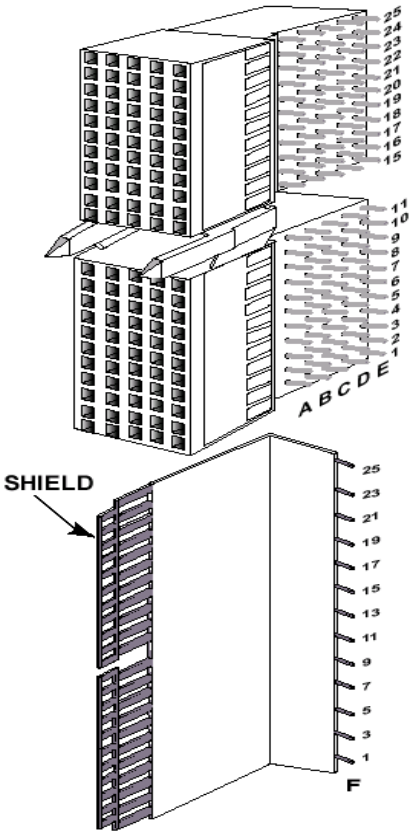


Figure A-2: J1 Connector

Table A-1 J1 Connector Pinout

Pin No.	Row A	Row B	Row C	Row D	Row E	Row F
25	+5 V	C_REQ64#	C_ENUM	CPCI_3.3	+5 V	GND
24	C_AD[1]	+5 V	CPCI_VIO	C_AD[0]	C_ACK64#	N/C
23	CPCI_3.3	C_AD[4]	C_AD[3]	+5 V	C_AD[2]	GND
22	C_AD[7]	GND	CPCI_3.3	C_AD[6]	C_AD[5]	N/C
21	CPCI_3.3	C_AD[9]	C_AD[8]	GND	C_C/BE0#	GND
20	C_AD[12]	GND	CPCI_VIO	C_AD[11]	C_AD[10]	N/C
19	CPCI_3.3	C_AD[15]	C_AD[14]	GND	C_AD[13]	GND
18	C_SERR#	GND	CPCI_3.3	C_PAR	C_C/BE1#	N/C
17	CPCI_3.3	C_SDONE	C_SB0#	GND	C_PERR#	GND
16	C_DEVSEL#	GND	CPCI_VIO	C_STOP#	C_LOCK#	N/C
15	CPCI_3.3	C_FRAME	C_IRDY#	GND	C_TRDY#	GND
12 through 14 are lost to the keying area						
11	C_AD[18]	C_AD[17]	C_AD[16]	GND	C_C/BE2#	GND
10	C_AD[21]	GND	CPCI_3.3	C_AD[20]	C_AD[19]	N/C
9	C_CBE3#	C_IDSEL	C_AD[23]	GND	C_AD[22]	GND
8	C_AD[26]	GND	CPCI_VIO	C_AD[25]	C_AD[24]	N/C
7	C_AD[30]	C_AD[29]	C_AD[28]	GND	C_AD[27]	GND
6	C_REQ#	GND	CPCI_3.3	C_CLK	C_AD[31]	N/C
5	N/C	N/C	C_RST#	GND	C_GNT#	GND
4	N/C	GND	CPCI_VIO	C_INTP	C_INTS#	N/C
3	C_INTA#	C_INTB#	C_INTC#	+5 V	C_INTD#	GND
2	C_TCK	+5 V	C_TMS	N/C	C_TDI	N/C
1	+5 V	-12 V	C_TRST#	+12 V	+5 V	GND

J2 Connector Pinout

The VMICPCI-7593 J2 connector is a 32 -bit, 2mm “Hard Metric” CompactPCI connector, with 22 rows, of 5 pins each. J2 is required for system slot CPUs. An additional external metal shield is also used, labeled row F. This connector’s controlled impedance minimizes unwanted signal reflections. Figure A-3 illustrates the J2 connector, and Table A-2 is the connector pinout.

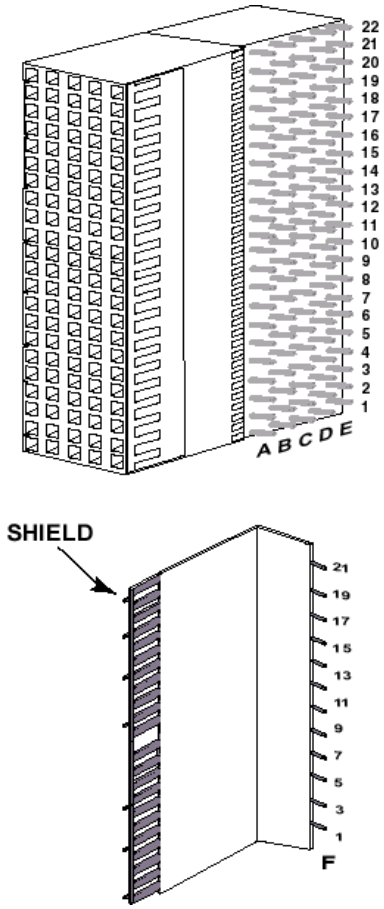


Figure A-3: J2 Connector

Table A-2 J2 Connector Pinout

Pin No.	Row A	Row B	Row C	Row D	Row E	Row F
22	N/C	N/C	N/C	N/C	N/C	N/C
21	C_CLK[6]	GND	N/C	N/C	N/C	GND
20	C_CLK[5]	GND	N/C	GND	N/C	N/C
19	GND	GND	N/C	N/C	N/C	GND
18	N/C	N/C	N/C	GND	N/C	N/C
17	N/C	GND	C-PRST#	C_REQ6#	C_GNT6#	GND
16	N/C	N/C	C_DEG#	GND	N/C	N/C
15	N/C	GND	C_FAL#	C_REQ5#	C_GNT5#	GND
14	N/C	N/C	N/C	GND	N/C	N/C
13	N/C	GND	CPCI_VIO	N/C	N/C	GND
12	N/C	N/C	N/C	GND	N/C	N/C
11	N/C	GND	CPCI_VIO	N/C	N/C	GND
10	N/C	N/C	N/C	GND	N/C	N/C
9	N/C	GND	CPCI_VIO	N/C	N/C	GND
8	N/C	GND	N/C	GND	N/C	N/C
7	N/C	GND	CPCI_VIO	N/C	N/C	GND
6	N/C	N/C	N/C	GND	N/C	N/C
5	C_C/BE5#	GND	CPCI_VIO	C_C/BE4#	C_PAR64	GND
4	CPCI_VIO	N/C	C_C/BE7#	GND	C_C/BE6#	N/C
3	C_CLK[4]	GND	C_GNT3#	C_REQ4#	C_GNT4#	GND
2	C_CLK[2]	C_CLK[3]	GND	C_GNT2#	C_REQ3#	N/C
1	C_CLK[1]	GND	C_REQ1#	C_GNT1#	C_REQ2#	GND

* CPCI_VIO - The VMICPCI-7593 is a universal VIO design.

* The VMICPCI-7593 supports a 32-bit CPCI bus.

*J2 Row C pins 15 and 16 are not supported.

J3 Connector Pinout

The J3 connector is 19 rows of 5 pins each, 2mm “Hard Metric” CompactPCI connector. An additional external metal shield is also used, labeled row F. Figure A-4 illustrates the J3 connector, and Table A-3 is the connector pinout. This connector is also used to route the serial, floppy, and harddrive signals to the backplane I/O.

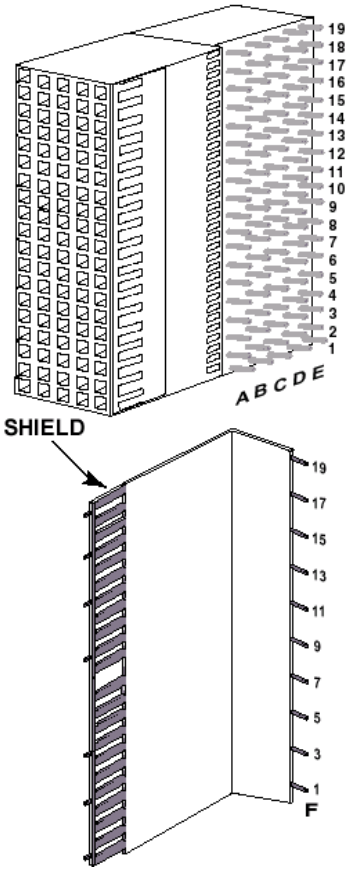


Figure A-4: J3 Connector

Table A-3: J3 Connector Pinout

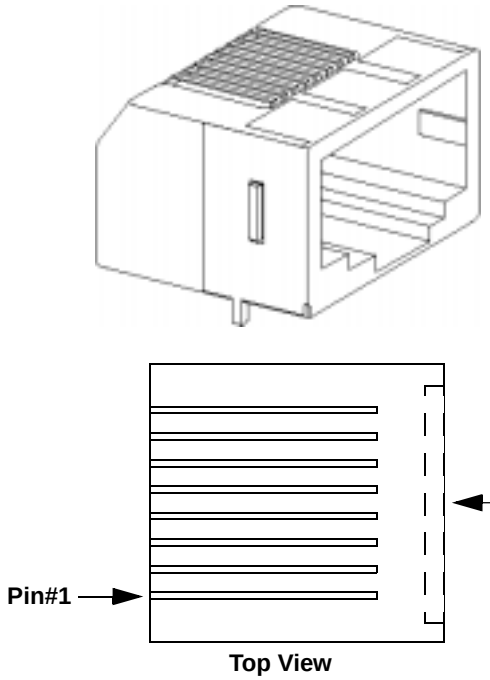
Pin No.	Row A	Row B	Row C	Row D	Row E	Row F
19	GND	ACK#	BUSY	PE	SLCT	GND
18	PD4	PD5	PD6	PD7	GND	N/C
17	PD1	INIT#	PD2	SLIN#	PD3	GND
16	N/C	STB#	AFD#	PD0	ERR#	N/C
15	N/C	N/C	N/C	N/C	N/C	GND
14	GND	RI COM1	DTR COM1	DCD COM1	TD COM1	N/C
13	RTS COM1	RD COM1	DSR COM1	CTS COM1	N/C	GND
12	RTS COM2	N/C	+5 V	N/C	+12 V	N/C
11	GND	RI COM2	DTR COM2	CTS COM2	TD COM2	GND
10	DAP0	RD COM2	DSR COM2	DCD COM2	DAP2PR	N/C
9	DAP1	REDWC#	IDEIORDY	IDESELA	IDEIOW0#	GND
8	DIR#	INDEX#	IDECS01#	DOP[8]	DOP[6]	N/C
7	SIDE1#	IDEIOR0#	HD_ACTA#	DOP[9]	DOP[7]	GND
6	RDATA#	TRK0#	IDECS03#	DOP[10]	DOP[2]	N/C
5	DRATE0#	WDATA#	IDEIRQ0	DOP[11]	DOP[3]	GND
4	DRVSB#	STEP#	IDE0ACK0#	DOP[12]	DOP[4]	N/C
3	DSKCHG#	MOTEB#	IOCS16A#	DOP[13]	DOP[5]	GND
2	WPT#	DRVSA#	IDEDRQ0	DOP[14]	DOP[0]	N/C
1	WGATE#	MOTEA#	IDERST#	DOP[15]	DOP[1]	GND

Note

Backplane designs should route J3 straight through to rear J3. The VMIACC-0559 board can then be utilitized.

Ethernet Connector Pinout (J6 and J7)

The pinout diagram for the two Ethernet 10BaseT and 100BaseTx connectors are shown in Figure A-5.

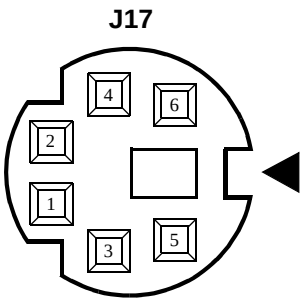


ETHERNET CONNECTOR (10BaseT, 100BaseTx)		
PIN	**Signal Name**	
1	TD+	Transmit Data
2	TD-	Transmit Data
3	RD+	Receive Data
4	NC	No Connection
5	NC	No Connection
6	RD-	Receive Data
7	NC	No Connection
8	NC	No Connection

Figure A-5: Ethernet Connector and Pinout

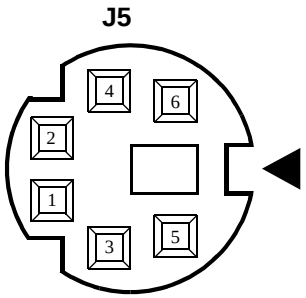
Keyboard and Mouse Connectors and Pinout

The keyboard and mouse connectors are standard 6-pin female mini-DIN PS/2 connectors as shown in Figure A-6 and A-7.



KEYBOARD CONNECTOR (J17)		
PIN	**DIR**	**FUNCTION**
1	In/Out	Data
2		Reserved
3		Ground
4		+5 V
5	Out	Clock
6		Reserved
Shield		Chassis Ground

Figure A-6 Keyboard Connector and Pinout

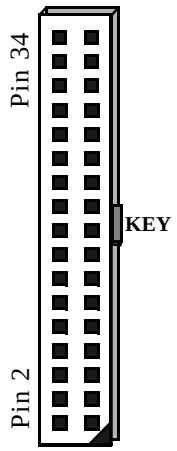


MOUSE CONNECTOR (J5)		
PIN	**DIR**	**FUNCTION**
1	In/Out	Data
2		Reserved
3		Ground
4		+5 V
5	Out	Clock
6		Reserved
Shield		Chassis Ground

Figure A-7 Mouse Connector and Pinout

Floppy Interface Connector Pinout

The floppy drive connector is a dual-row 34-pin header connector and is located at site E21 on Figure A-1. Pin 1 marks the beginning of the odd-numbered row. Most standard PC-compatible floppy disk drives use this pinout, which is illustrated in Figure A-8. The user should exercise caution when aligning the floppy drive connector and cable.

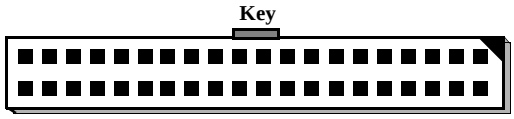


FLOPPY DRIVE HEADER		
PIN	**DIRECTION**	**FUNCTION**
1		Ground
2	From Controller	Drive Density 0 (not connected on some controllers)
3		Ground
4		Reserved
5		Ground
6	From Controller	Drive Density 1 (not connected on some controllers)
7		Ground
8	From Drive	Index
9		Ground
10	From Controller	Motor Enable A
11		Ground
12	From Controller	Drive Select B
13		Ground
14	From Controller	Drive Select A
15		Ground
16	From Controller	Motor Enable B
17		Ground
18	From Controller	Step Motor Direction
19		Ground
20	From Controller	Step Pulse
21		Ground
22	From Controller	Write Data
23		Ground
24	From Controller	Write Enable
25		Ground
26	From Drive	Track 0
27		Ground
28	From Drive	Write Protect
29		Ground
30	From Drive	Read Data
31		Ground
32	From Controller	Select Head 1
33		Ground
34	From Drive	Disk Change

Figure A-8 Floppy Drive Header and Pinout

EIDE Hard Drive Connector Pinout

Figure A-9 describes the pin assignment for the 40-pin EIDE / ATA hard drive header connector. The connector is located at the E25 site in Figure A-1. Like the floppy header connector it has an odd-numbered row and an even-numbered row. The user should exercise caution to align the hard drive connector and cable. If this connector is implemented, the jumper E18 must be placed in the 1-2 position.



PIN	DIRECTION	DESCRIPTION	PIN	DIRECTION	DESCRIPTION
1	Out	Reset Drive	2	Out	Signal Ground
3	In / Out	Bidirectional Data 07	4	In / Out	Bidirectional Data 08
5	In / Out	Bidirectional Data 06	6	In / Out	Bidirectional Data 09
7	In / Out	Bidirectional Data 05	8	In / Out	Bidirectional Data 10
9	In / Out	Bidirectional Data 04	10	In / Out	Bidirectional Data 11
11	In / Out	Bidirectional Data 03	12	In / Out	Bidirectional Data 12
13	In / Out	Bidirectional Data 02	14	In / Out	Bidirectional Data 13
15	In / Out	Bidirectional Data 01	16	In / Out	Bidirectional Data 14
17	In / Out	Bidirectional Data 00	18	In / Out	Bidirectional Data 15
19	Out	Signal Ground	20	None	Unused, Keying Position
21		Reserved	22	Out	Signal Ground
23	Out	Write Strobe	24	Out	Signal Ground
25	Out	Read Strobe	26	Out	Signal Ground
27		Reserved	28	Out	Address Latch Enable
29		Reserved	30	Out	Signal Ground
31	In	Interrupt Request #14	32	In	16-bit Data Word Size
33	Out	Address Line #1	34	In	Diagnostic Test Passed
35	Out	Address Line #0	36	Out	Address Line #2
37	Out	Chip Select #0	38	Out	Chip Select #1
39	In	Slave/Activity Status	40	Out	Signal Ground

Figure A-9 EIDE Hard Drive Connector and Header Pinout

Note

If the VMIACC-0559 is implemented for IDE connection E18 must be placed in the 2-3 position.

PMC Connector Pinout

PMC P11 Connector and Pinout

The PCI Mezzanine Card (PMC) carries the same signals as the PCI standard; however, the PMC standard uses a completely different form factor. Table A-4 and A-5 are the pinouts for the PMC connectors (P11 and P12).

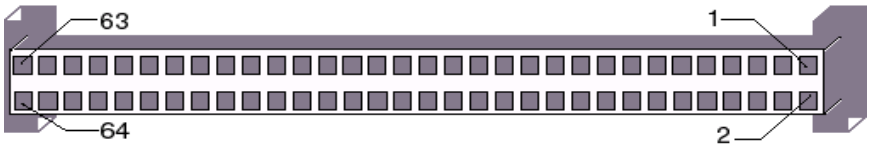


Table A-4: PMC P11 Connector Pinout

PMC Connector (P11)				PMC Connector (P11)			
Left Side		Right Side		Left Side		Right Side	
Pin	Name	Pin	Name	Pin	Name	Pin	Name
1	GND	2	-12	33	FRAME#	34	GND
3	GND	4	INTA#	35	GND	36	IRDY#
5	INTB#	6	INTC#	37	DEVSEL#	38	+5 V
7	BMODE1A	8	+5 V	39	GND	40	LOCK#
9	INTD#	10	NC	41	NC	42	NC
11	GND	12	NC	43	PAR	44	GND
13	CLK	14	GND	45	+5 V	46	AD[15]
15	GND	16	GNT#	47	AD[12]	48	AD[11]
17	REQ#	18	+5 V	49	AD[9]	50	+5 V
19	+5 V	20	AD[31]	51	GND	52	C/BE0#
21	AD[28]	22	AD[27]	53	AD[6]	54	AD[5]
23	AD[25]	24	GND	55	AD[4]	56	GND
25	GND	26	C/BE3#	57	+5 V	58	AD[3]
27	AD[22]	28	AD[21]	59	AD[2]	60	AD[1]
29	AD[19]	30	+5 V	61	AD[0]	62	+5 V
31	+5 V	32	AD[17]	63	GND	64	REQ64#

PMC P12 Connector and Pinout

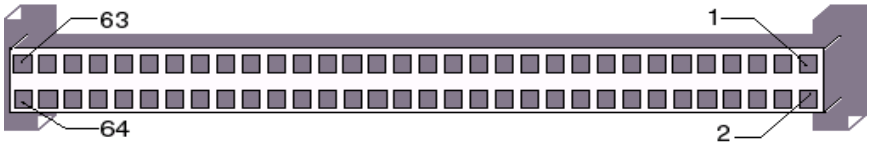


Table A-5: PMC P12 Connector Pinout

PMC Connector (P12)				PMC Connector (P12)			
Left Side		Right Side		Left Side		Right Side	
Pin	Name	Pin	Name	Pin	Name	Pin	Name
1	+12 V	2	+5 V	33	GND	34	NC
3	GND	4	NC	35	TRDY	36	+3.3 V
5	+5 V	6	GND	37	GND	38	STOP#
7	GND	8	NC	39	PERR#	40	GND
9	NC	10	NC	41	+3.3 V	42	SERR#
11	PRSNT2	12	+3.3 V	43	C/BE1#	44	GND
13	RST#	14	GND	45	AD[14]	46	AD[13]
15	+3.3 V	16	GND	47	GND	48	AD[10]
17	NC	18	GND	49	AD[8]	50	+3.3 V
19	AD[30]	20	AD[29]	51	AD[7]	52	NC
21	GND	22	AD[26]	53	+3.3 V	54	NC
23	AD[24]	24	+3.3 V	55	NC	56	GND
25	IDSEL	26	AD[23]	57	NC	58	NC
27	+3.3 V	28	AD[20]	59	GND	60	NC
29	AD[18]	30	GND	61	ACK64#	62	+3.3 V
31	AD[16]	32	C/BE2#	63	GND	64	NC

Appendix

B

System Driver Software

Contents

Driver Software Installation 82
Windows for Workgroups™ OS (Version 3.11) 83
Windows® 95 OS 85
Windows NT® OS (Version 4.0) 87

Introduction

The VMICPCI-7593 provides two Local Area Network (LAN) accesses by means of on-board PCI-based adapters and associated software drivers. High-performance LAN operation including 10BaseT and 100BaseTx, is provided by two unique DEC 21143 Ethernet controller chips.

To optimize performance of each of these PCI-based subsystems, the VMICPCI-7593 is provided with software drivers compatible with DOS, Windows for Workgroups™ Version 3.11, Windows® 95 OS, and Windows NT® operating systems. The following paragraphs provide instructions for loading and installing the adapter software.

Driver Software Installation

In order to properly use the LAN adapters of the VMICPCI-7593, the user must install the driver software provided as distribution diskettes with the unit. Detailed instructions for installation of the drivers during installation of Windows for Workgroups Version 3.11, Windows 95 OS, or Windows NT (Versions 3.5x and 4.0) operating systems are described in the following sections.



A video adapter is required for installation of the drives. The VMICPCI-7593 does not provide on-board video, therefore the user can either utilize the PMC site for video or install a CompactPCI VGA video card in one of the other backplane peripheral slots.

Windows for Workgroups™ OS (Version 3.11)

The VMICPCI-7493 provides two Local Area Network (LAN), the following procedures can be used for setup of either LAN.

1. Format and load the IDE hard drive with MS-DOS®.
2. Begin installation of Windows for Workgroups 3.11 OS, following the instruction provided by Microsoft for Express Setup. When you reach the 'WINDOWS FOR WORKGROUPS NETWORK SETUP' screen, it will show that 'NO NETWORK IS INSTALLED'.
3. The following steps will load the DEC Ethernet drivers, and configure the DEC 21143 adapter.

Note

If you do not require LAN operation, click the mouse on 'CONTINUE' and skip steps 4 through 12. Otherwise from the main 'NETWORK SETUP' screen, click the mouse on the 'NETWORKS' button.

4. Under 'NETWORKS', click the mouse on the circle to install 'MICROSOFT WINDOWS NETWORK'.
5. Then Click the mouse on 'OK'.
6. The 'NETWORK SETUP SCREEN' appears again with the option for 'SHARING'. Click on 'SHARING' and choose the options that fit your system requirements by placing an x in the boxes shown. Then click on 'OK'.
7. Again at the 'NETWORK SETUP SCREEN', click on 'CONTINUE'.
8. Under 'ADD NETWORK ADAPTER' click the mouse on 'UNLISTED or UPDATED NETWORK ADAPTER'. Then click the mouse on 'OK'.
9. Insert the VMICPCI-7593 distribution disk marked 320-500018-002 into drive A: and type: `A:\32bit\WFW311\.` Then click the mouse on 'OK'.
10. Under 'UNLISTED' or 'UPDATED NETWORK ADAPTER', choose 'Digital Semiconductor 21143-based 10/100 mpbs Ethernet Controller (ND153)'. Then click the mouse on 'OK'.
11. Under 'MICROSOFT WINDOWS NETWORK NAMES', enter the network names you want for computer name, group name, etc. Then click the mouse on 'OK'.
12. Windows for Workgroups OS should now continue with regular installation. During the remaining installation steps, use the full path name, A:\32bit\WFW311\, whenever prompted for the DEC 21143 driver diskette.
13. In order for the network to be setup properly, it is necessary to setup the proper connection type for your system.

Proceed with the following steps to set your network connection type.

Note

If you are not running a network, please skip steps 14 through 18.

14. From the 'PROGRAM MANAGER' screen, double-click on the 'NETWORK' icon.
15. Then double-click on 'NETWORK SETUP' icon.
16. Under 'NETWORK SETUP', double-click on 'Digital Semiconductor 21143-based 10/100 mpbs Ethernet Controller (ND153)'.
17. Under 'ADVANCED NETWORK-ADAPTER SETTINGS', choose 'CONNECTION TYPE' and choose the 'CONNECTION TYPE VALUE' of *'AUTO SENSE TYPE'*. Click on 'SET' and then 'OK'.
18. Under 'NETWORK SETUP', click on 'OK'. If the network connection type has changed, then click 'OK' at the next screen for the information message about 'SYSTEM.INI' and 'PROTOCOL.INI' and click on 'RESTART COMPUTER' for the new network settings to take effect.

The unit should now be configured for operation in the WINDOWS FOR WORKGROUPS 3.11 OS environment.

Windows® 95 OS

1. Format the hard drive with MS-DOS.
2. Begin installation of Windows 95 OS, following the instructions provided by the Windows 95 OS manual.
3. When you reach the 'WINDOWS 95 SETUP WIZARD SCREEN', choose 'TYPICAL' under 'SETUP OPTIONS' and then click on 'NEXT'.
4. When you reach the 'ANALYZING YOUR COMPUTER' screen, place an X in the box for 'NETWORK ADAPTER', then click on 'NEXT'.
5. Under the 'WINDOWS COMPONENTS SCREEN', select 'INSTALL THE MOST COMMON COMPONENTS' and then click on 'NEXT'.
6. Continue with the installation until Windows 95 OS is completely installed and has rebooted.

Note

If you do not require LAN operation, skip steps 7 through 24.

7. From the main Windows 95 OS screen, click on 'START'.
8. Then click on 'SETTINGS' and then 'CONTROL PANEL'.
9. Double-click on the 'SYSTEM' icon and select the 'DEVICE MANAGER' tab.
10. Double-click on 'OTHER DEVICES' and then double-click on 'PCI ETHERNET CONTROLLER'.
11. Select the 'DRIVER' tab and then select 'CHANGE DRIVER'.
12. In the 'SELECT HARDWARE TYPE' screen, choose 'NETWORK ADAPTERS' and click on 'OK'.
13. Insert the Diskette marked '320-500013-002' into drive A:
14. In the 'SELECT DEVICE' window, click on 'HAVE DISK' and type `A:\32bit\WIN95\INF` and then click on 'OK'.
15. Under 'SELECT NETWORK ADAPTERS' choose 'DIGITAL SEMICONDUCTOR 21143-BASED 10/100 MPBS ETHERNET CONTROLLER', then click on 'OK'.
16. Under 'PCI ETHERNET CONTROL PROPERTIES' select 'OK'. Then the system will prompt you for computer and workgroup names. Type in the names you wish to use in the spaces shown and then choose 'CLOSE'.
17. Windows 95 OS will then prompt for diskettes. Follow all instructions.
18. At the 'SYSTEM PROPERTIES' window, click on 'OK'.
19. The driver files are located at A:\32bit\WIN95\. If prompted for the Windows 95 OS system disk, indicate the location of the files. For example, if you are loadingfrom a CD-ROM located a D:, the desire response is D:\WIN95.

20. From ‘CONTROL PANEL’, double-click on the ‘NETWORK’ icon.
21. Under ‘NETWORK’, click on ‘FILE AND PRINT SHARING’ and choose the appropriate items for your system, click on ‘OK’.
22. Under the ‘NETWORK’ window, double-click on ‘Digital Semiconductor 21143-based 10/100 mpbs Ethernet Controller (ND153)’.
23. At the ‘NETWORK’ window, click on ‘OK’. When prompted, insert the diskettes needed to complete the network installation.
24. When the system prompts you to restart your computer, click on ‘YES’ for the network settings to take effect.

The unit should now be properly configured for operation in Windows 95 OS.

Windows NT® OS (Version 4.0)

Windows NT 4.0 OS includes drivers for the on-board LAN and video adapters. The following steps are required to configure the LAN for operation:

1. Follow the normal Windows NT 4.0 OS installation until you reach the 'WINDOWS NT WORKSTATION SETUP' window which states that 'WINDOWS NT NEEDS TO KNOW HOW THIS COMPUTER SHOULD PARTICIPATE ON A NETWORK'.
2. Place a dot next to 'THIS COMPUTER WILL PARTICIPATE ON A NETWORK'.
3. Place a check mark next to 'WIRED TO THE NETWORK' and then click on 'NEXT'.
4. At the next screen, click on the 'SELECT FROM LIST' button.
5. Click on the 'HAVE DISK' button.
6. Insert the distribution disk marked '320-500018-002' into drive A:.
7. Type A:\32bit\wnt40\NDIS40 and click on 'OK'.
8. In the "SELECT OEM OPTION', choose 'Digital Semiconductor 21143-based 10/100 mpbs Ethernet Controller (NDIS4)', then click 'OK'.
9. Select the above entry on the displayed list, click on 'NEXT'.
10. Select the NetBEUI Protocol (only) or to disregard the protocol, click on 'NEXT'.
11. Click 'CONTINUE' to allow an Autosense connection type.
12. Step through the remaining screens, providing the data pertinent to your network.
13. Continue through the setup procedure until the 'DETECTED DISPLAY' window appears, click on 'OK' to continue.
14. In the 'DISPLAY PROPERTIES' window, click on 'TEST'.

    If the display test is successful, click on 'OK' to continue. If the display test is not successful, you may have to adjust the display parameter to find a functional setting, for example a lower resolution or lower number of colors.
15. Continue with the procedure for the 'Windows NT Setup' window. Click on 'RESTART COMPUTER'.
16. When the computer reboots, double-click on 'MY COMPUTER' window.
17. Proceed as directed, removing the driver disk from the floppy drive, and the computer will restart to activate the new settings. When the system reboots, the display settings may be configured for your display type.

The unit should now be configured for operation under Windows NT 4.0 OS.

Appendix

C

LANWorks BIOS

Contents

System BIOS Setup Utility 90
BIOS Features Setup 91
LANWorks BIOS Setup 92

Introduction

The VMICPCI-7593 includes the LANWorks BIOS options which allows the VMICPCI-7593 to be booted from a network. This appendix describes the LANWorks BIOS Setup screen, and the procedures to enable this option.

Remote Ethernet Booting

The VMICPCI-7593 is capable of booting from a server over the Ethernet. Utilizing LANWorks BootWare®. BootWare gives you the ability to remotely boot the VMICPCI-7593 using Netware, TCP/IP, or RPL network protocols. The Ethernet must be connected through the LAN 1, RJ-45 front panel connector to boot remotely. This feature allows users to create systems without the worry of disk drive reliability, or the extra cost of adding Flash drives.

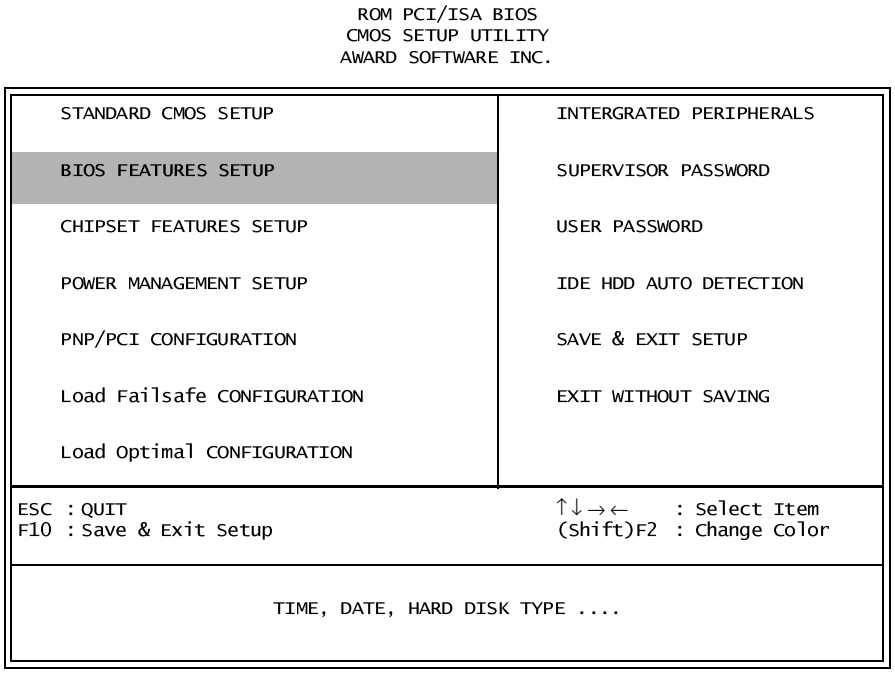
BootWare Features:

- Netware, TCP/IP, RPL Compatible
- Unparalleled boot sector virus protection
- Detailed boot configuration screens
- Comprehensive diagnostics
- Optional disabling of local boots
- Dual-boot option lets users select network or local booting

The following are the actual windows displayed when entering the LANWorks BIOS.

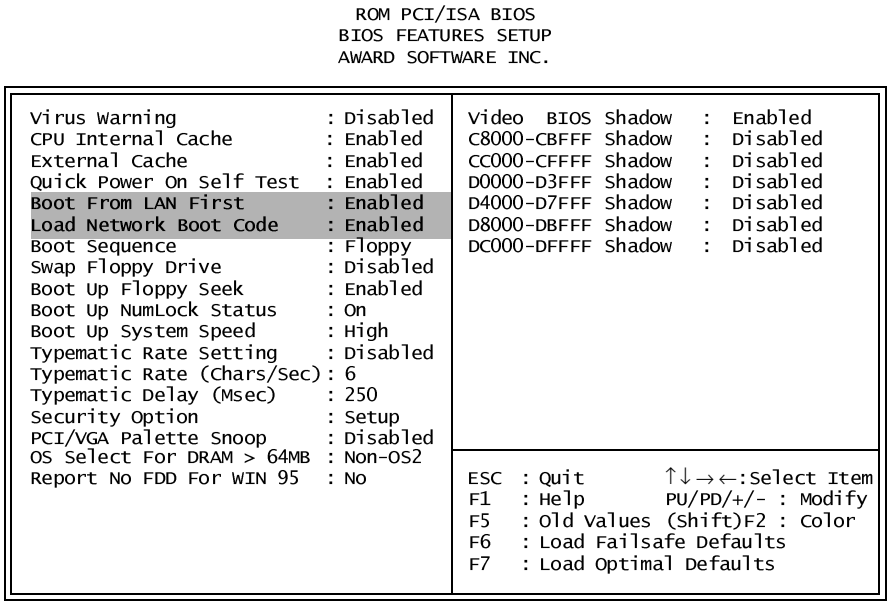
System BIOS Setup Utility

To enable the LANWorks BIOS option reboot the VMICPCI-7593 and when prompted, press the Delete key to access the System BIOS Setup Utility screen shown below.



Using the arrow keys, highlight *BIOS Features Setup,* and press the Return key. This will display the BIOS Features Setup screen.

BIOS Features Setup

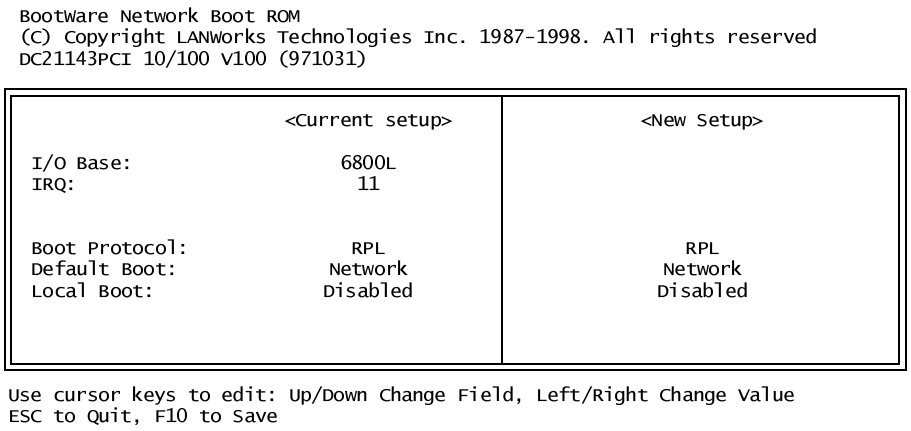


Using the arrow keys, enable *Boot From LAN First* and *Load Network Boot Code*. Exit the BIOS setup, saving changes.

When prompted, press “Control-Alt-B” as the VMICPCI-7593 reboots. This will activate the LANWorks BIOS setup screen.

LANWorks BIOS Setup

Below is the screen in which the various options for booting through LANWorks are set.



Boot Protocol

The Boot Protocol required is dependent on the network system being utilized, check with your network administrator for the correct protocol. The following options are available in the Boot Protocol field:

- **RPL** **(Default)**
- TCP/IP BootP Used for Real Time operating systems
- TCP/IP DACP
- Netware 802.3
- Netware 802.2
- Netware th_TI

Default Boot

The following options are available in the Default Boot field:

- **Network (Default)** - Selects Network Boot as the default boot device
- Local - Selects a Local Drive as the default boot device

Local Boot

The following options are available in the Local Boot field:

- **Disabled (Default)** - Disables Local Boot from a Hard Drive or Floppy
- Enabled - Enables Local Boot from a Hard Drive or Floppy

Appendix

D

Award - BIOS

Contents

System BIOS Setup Utility 94
Standard CMOS Setup 95
BIOS Features Setup 97
Chipset Features Setup 100
Power Management. 103
PnP / PCI Configuration 106
Integrated Peripherals. 108

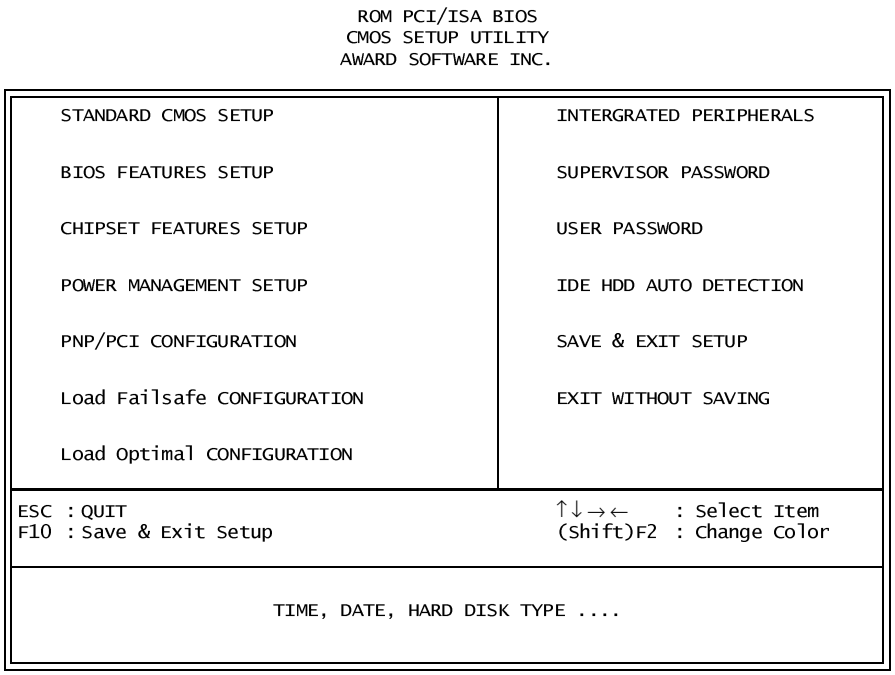
Introduction

The VMICPCI-7593 utilizes the BIOS (Basic Input / Output system) in the same manner as other PC / AT compatible computers. This appendix describes the menus and options associated with the VMICPCI-7593 BIOS.

System BIOS Setup Utility

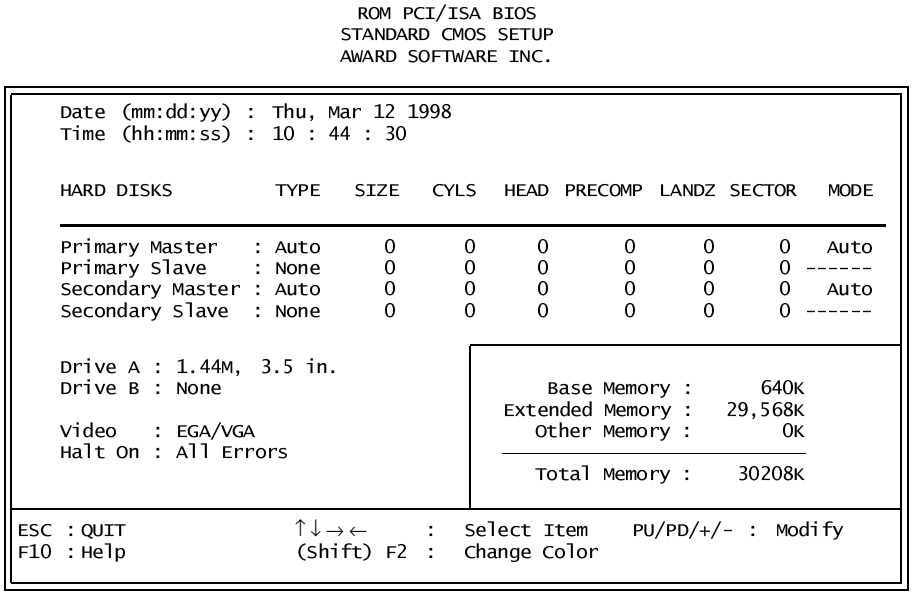
During system bootup, press the Delete key to access the Award *Elite*BIOS CMOS Setup Utility screen. From this screen, the user can select any section of the Award (system) BIOS for configuration, such as floppy drive configuration or system memory.

The parameters shown throughout this section are the default values.



Standard CMOS Setup

Selection of the first main menu item, the standard CMOS Setup, allows the user to set the system clock and calendar, record disk drive parameters, video subsystem type, and select the type of errors that will cause a system halt.



Setting The Date

Press the left or right arrow key to move the cursor to the desired field (month, day, year). Press the PgUp or PgDn key to step through the available choices, or type in the information. Note that the day of the week field is for information purposes only, and will be set based on the other date information.

Setting The Time

The time format is based on the 24-hour military-time clock. For example, 1 PM is 13:00:00. Press the left or right arrow key to move the cursor to the desired field (hour, minute, seconds). Press the PgUp or PgDn key to step through the available choices, or type in the information.

Primary Master/Slave

The VMICPCI-7593 has the capability of utilizing one IDE hard disk drive on the Primary Master bus. The default setting is Auto. The Primary Slave is not used with the VMICPCI-7593.

Secondary Master/Slave

The Secondary Master is the resident 8 Mbyte flash disk. The default setting is Auto. The Secondary Slave is not assignable.

Floppy Disk Drive

Floppy Drive A

The VMICPCI-7593 supports one floppy disk drive. The options are:

- None — No diskette drive installed
- 360K, 5.25 in — 5-1/4 inch PC-type standard drive; 360 kilobyte capacity
- 1.2M, 5.25 in — 5-1/4 inch AT-type high-density drive; 1.2 megabyte capacity
- 720K, 3.5 in — 3-1/2 double-sided drive; 720 kilobyte capacity
- 1.44M, 3.5 in — 3-1/2 inch double-sided drive; 1.44 megabyte capacity
- 2.88M, 3.5 in — 3-1/2 inch double-sided drive; 2.88 megabyte capacity

Use PgUp or Pgdn to select the floppy drive. The default is 1.44M, 3.5 inch.

Floppy Drive B

The VMICPCI-7593 does not support a second floppy drive. The default is None.

Halt On

During the power-on self-test (POST), the computer may be made to halt if the BIOS detects a hardware error. The options are:

- All Errors — If the BIOS detects any non-fatal error, POST stops and prompts you for corrective action
- No Errors — POST does not stop for any errors
- All, But Keyboard — Post does not stop for a keyboard error, but stops for all other errors
- All, But Diskette — POST does not stop for diskette drive errors, but stops for all other errors
- All, But Disk/Key — POST does not stop for a keyboard or disk error, but stops for all other errors

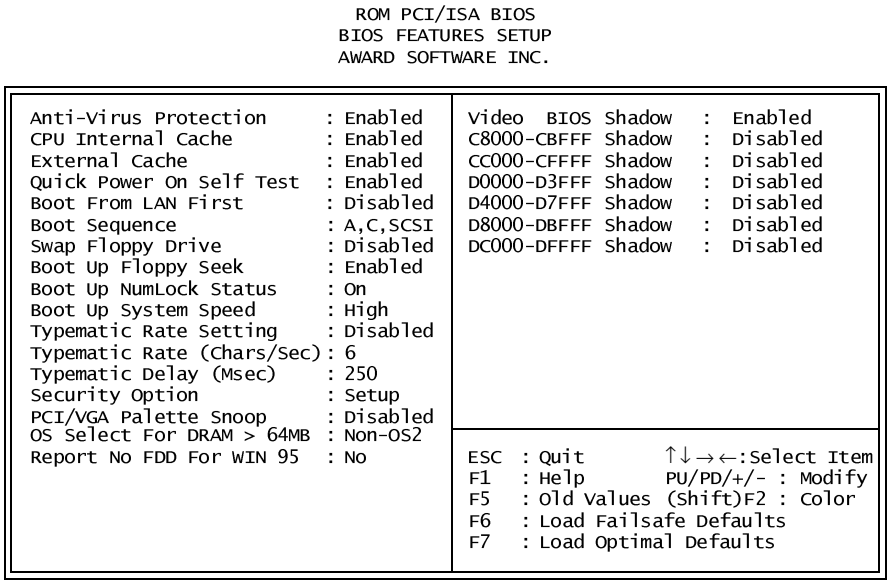
Use the PgUp or PgDn key to select the errors which will halt the system. The default is All Errors.

Memory

The Memory field at the lower right of the screen is for informational purposes only and can not be modified by the user. This field displays the total RAM installed in the system, and the amounts allocated to base, extended, and other (high) memory.

BIOS Features Setup

This screen, selected from the CMOS Setup Utility screen, allows the user to configure options that are in addition to the basic BIOS features.



Anti-Virus Protection

When enabled the system provides a warning if an attempt is made to write to the boot sector or the partition table of the hard disk drive the system is booted from. If a warning is received, it is recommended that an anti-virus program be run. Please note that this feature protects only the boot sector, not the entire hard drive. The default is Enabled.

CPU Internal Cache

Enabling the cache memory enhances the speed of the processor. When the CPU requests data, the system transfers the requested data from the main DRAM into the cache memory were it is stored until processed by the CPU. The default is Enabled.

External Cache

The external cache (512 K) provides additional cache for use by the CPU. This cache functions in the same manner as internal cache. The default is Enabled.

Quick Power On Self Test

When enabled, certain checks normally performed during the POST are omitted, decreasing the time required to run the POST. The default is Enabled.

Boot From LAN First

When enabled, this option allows the CPU to boot off of a connected network. The default is Disabled.

Boot Sequence

Determines the order in which the BIOS will seek a bootable drive. The default order is the floppy disk (A:), then the internal hard drive (C:), followed by the SCSI drive, if attached.

Swap Floppy Drive

The option is functional only in a system with two floppy drives. The VMICPCI-7593 supports only one floppy drive. Changing this option will have no effect on the system. The default is Disabled.

Boot Up Floppy Seek

When enabled, the BIOS will test the floppy to determine if the floppy drive has 40 or 80 tracks. Only 360K floppy drives have 40 tracks. It is recommended that this option be set to disabled unless a third party 360K floppy drive is being used in the system. The default is Enabled.

Boot Up NumLock Status

Toggle between On or Off to control the state of the NumLock key when the system boots. When toggled On, the numeric keypad generates numbers instead of controlling the cursor operations. The default is ON.

Boot Up System Speed

This option determines if the system boots at the default CPU speed (high) or at the AT bus speed (low). Some add-on peripherals or legacy software may require the system to be booted at the lower speed. The default is High.

Typematic Rate Setting

This option enables or disables the Typematic Rate and Delay settings. When disabled the values in the Typematic Rate and Delay are ignored. The default is Disabled.

Typematic Rate (Chars/Sec)

If the Typematic Rate Setting is enabled this determines the rate a character is repeated when a key is held down. The options are: 6, 8, 10, 12, 15, 20, 24, or 30 characters per second. The default is 6.

Typematic Delay (Msec)

If the Typematic Rate Setting is enabled this determines the delay before a character starts repeating when a key is held down. The options are: 250, 500, 750, or 1000 milliseconds. The default is 250.

Security Option

If a password has been set, this determines whether the password is required every time the system boots, or only when the Setup is accessed. The default is Setup.

PCI/VGA Palette Snoop

Enabling the video palette snoop allows the ISA add-in card to share a common palette with the graphics controller. The default is Disabled.

OS Select For DRAM>64MB

Select OS2 only if you are running OS/2 operating system with greater than 64 MB of RAM on the system. The default is Non-OS2.

Report No FDD For WIN 95

Selecting Yes for this option releases IRQ 6 when the system does not contain a floppy drive, for compatibility with Windows 95 logo certification. In the Integrated Peripherals screen, select Disabled for the Onboard FDC Controller field. The default is No.

Chipset Features Setup

This section describes features of the Intel 82430TX PCIset.

Advance Options



The parameters in this screen are for system designers, service personnel, and technically competent users only. Do not reset these values without a complete understanding of the consequences.

```
                    ROM PCI/ISA BIOS
                CHIPSET FEATURES SETUP
                AWARD SOFTWARE INC.

Auto Configuration          : Enabled
DRAM Timing                 : 70ns

DRAM Leadoff Timing         : 10/6/4
DRAM Read Burst (EDO/FP)    : X333/X444
DRAM Write Burst Timing     : X333
Fast EDO Lead Off           : Disabled
Refresh RAS# Assertion      : 5 Clks
Fast RAS to CAS Delay       : 2
DRAM Page Idle Timer        : 2 Clks
DRAM Enhanced Paging        : Enabled
Fast MA to RAS# Delay       : 2 Clks
SDRAM(CAS Lat/RAS-to-CAS)   : 2/2
SDRAM Speculative Read      : Disabled
System BIOS Cacheable       : Disabled
Video BIOS Cacheable        : Disabled
8 Bit I/O Recovery Time     : 1
16 Bit I/O Recovery Time    : 2
Memory Hole At 15M-16M      : Disabled
PCI 2.1 Compliance          : Disabled

ESC  : Quit        ↑↓→←:Select Item
F1   : Help        PU/PD/+/- : Modify
F5   : Old Values  (Shift)F2 : Color
F6   : Load Failsafe Defaults
F7   : Load Optimal Defaults
```

Auto Configuration

Auto Configuration selects predetermined optimal values of chipset parameters. When disabled, chipset parameters revert to the setup information stored in CMOS. The default is Enabled.

DRAM Timing

The value in this field depends on performance parameters of the installed memory chips (DRAM). Do not change the value from the factory setting unless new memory has been installed with a different performance rating than the original DRAMs. The default is 70ns.

DRAM Leadoff Timing

Select the combination of CPU clocks the DRAM on the system board requires before each read from or write to the memory. Changing the value of this setting from the default may cause memory errors. The default is 10/6/4.

DRAM Read Burst (EDO/FP)

Sets the timing for reads from EDO (Extended Data Output) or FP[M] (Fast Page Mode) memory. The lower the timing numbers, the faster the system addresses memory. Selecting timing numbers lower than the installed DRAM is able to support can result in memory errors. The default is x333/x444.

DRAM Write Burst Timing

Sets the timing for writes to memory. The lower the timing numbers, the faster the system addresses memory. Selecting timing numbers lower than the installed DRAM is able to support can result in memory errors. The default is x333.

Fast EDO Lead Off

Refresh RAS# Assertion

Select the number of clock cycles in which RAS# is asserted for refresh cycles. The default is 5 Clks.

Fast RAS to CAS Delay

This field inserts a timing delay between the CAS and RAS strobe signals, used when DRAM is written to, read from, or refreshed. The default is 2.

DRAM Page Idle Timer

Select the amount of time in HCLKs that the DRAM controller waits to close a DRAM page after the CPU becomes idle. The default is 2 Clks.

DRAM Enhanced Paging

When Enabled, the chipset keeps the page open until a page/row miss. When Disabled, the chipset uses additional information to keep the DRAM page open when the host may be “right back.” The default Enabled.

Fast MA to RAS# Delay

The values in this field are set by the system board designer, depending on the DRAM installed. Do not change the values in this field unless you change specifications of the installed DRAM or the installed CPU. The default is 2 Clks

SDRAM(CAS Lat/RAS-to-CAS)

You can select a combination of CAS latency and RAS-to-CAS delay in HCLKs of 2/2 or 3/3. The values in this field depend on the DRAM installed. Do not change the values in this field unless you change specifications of the installed DRAM or the installed CPU. The default is 2/2.

SDRAM Speculative Read

The chipset can “speculate” on a DRAM read address, thus reducing read latencies. The CPU issues a read request containing the data memory address. The DRAM controller receives the request. When this field is Enabled, the controller issues the read command slightly before it has finished decoding the data address. The default is Disabled.

System BIOS Cacheable

Selecting Enabled allows caching of the system BIOS ROM at F0000h-FFFFFh. If a program writes to this memory area, a system error may occur. The default is Disabled.

Video BIOS Cacheable

Selecting Enabled allows caching of the video BIOS ROM at C0000h to C7FFFh. If a program writes to this memory area, a system error may occur. The default is Disabled.

8 Bit I/O Recovery Time

The I/O recovery mechanism adds bus clock cycles between PCI-originated I/O cycles to the ISA bus. This delay takes place because the PCI bus is faster than the ISA bus. The default is 1.

16 Bit I/O Recovery Time

The I/O recovery mechanism adds bus clock cycles between PCI-originated I/O cycles to the ISA bus. This delay takes place because the PCI bus is faster than the ISA bus. The default is 2.

These two fields (8 Bit and 16 Bit) let you add recovery time (in bus clock cycles) for 8-bit and 16-bit I/O.

Memory Hole at 15M-16M

This area of system memory may be reserved for ISA adapter ROM. When this area is reserved, it cannot be cached. Refer to the documentation that came with the peripheral that require the use of this area of system memory for memory requirements. The default is Disabled.

PCI 2.1 Compliance

Select Enabled to support compliance with PCI specification version 2.1. The default is Disabled.

Power Management

This section discusses the power management features provided with the installed Award BIOS.

```
                    ROM PCI/ISA BIOS
                POWER MANAGEMENT SETUP
                AWARD SOFTWARE INC.

ACPI Function              : Disabled      ** Reload Global Timer Events **
Power Management           : Disabled      IRQ  [3-7,9-15],NMI: Enabled
PM Control by APM          : Yes           Primary IDE 0      : Disabled
Video Off Method           : V/H SYNC+Blank Primary IDE 1      : Disabled
Video Off After            : Standby       Secondary IDE 0    :Disabled
Modem Use IRQ              : 3             Secondary IDE 1     :Disabled
Doze Mode                  : Disabled      Floppy disk        : Disabled
Standby Mode               : Disabled      Serial Port        : Enabled
Suspend Mode               : Disabled      Parallel Port      : Disabled
HDD Power Down             : Disabled
Throttle Duty Cycle        : 62.5%
zz Active in Suspend       : Disabled
PCI/VGA Act-Monitor        : Enabled
Soft-off by PWR-BTTN       : Instant-Off
Power On by Ring           : Enabled
IRQ 8 Break Suspend        : Disabled

                                           ESC  : Quit       ↑↓→←:Select Item
                                           F1   : Help       PU/PD/+/- : Modify
                                           F5   : Old Values (Shift)F2 : Color
                                           F6   : Load Failsafe Defaults
                                           F7   : Load Optimal Defaults
```

ACPI Function

Select Enabled only if your computer's operating system supports the Advanced Configuration and Power Interface (ACPI) specification. Currently, Windows 98 and Windows NT 5.0 support ACPI. The default is Disabled.

Power Management

This option disables or sets the power management options. The options are:

- Max SavingMaximum power savings. This option is only available for SL CPUs.
  Inactivity period is 1 minute in each mode.
- User defineSet each mode individually. Select time-out periods in the PM timers section.
- Min SavingsMinimum power savings. Inactivity period is 1 hour in each mode (except the hard drive).
- DisabledTurns off all power management features.

The default is Disabled.

PM Control by APM

Advanced Power Management (APM) provides better power savings. The default is Yes.

Video Off Method

Determines the manner in which the monitor is blanked. The options are:

- V/H SYNC+BlankSystem turns off vertical and horizontal synchronization ports and writes blanks to the video buffer.
- DPMS SupportSelect this option if your monitor supports the Display Power Management Signaling (DPMS) standard of the Video Electronics Standards Association (VESA). Use the software supplied for your video subsystem to select video power management values.
- Blank ScreenSystem only writes blanks to the video buffer.

The default setting is V/H SYNC+Blank.

Video Off After

As the system moves from lesser to greater power-saving modes, select the mode in which you want the monitor to blank. The default is Standby.

Modem Use IRQ

Name the interrupt request (IRQ) line assigned to the modem (if any) on your system. Activity of the selected IRQ awakens the system. The default setting is 3.

Doze Mode

After the selected period of system inactivity (1 minute to 1 hour), the CPU clock runs at a slower speed while all other devices still operate at full speed. The default is Disabled.

Standby Mode

After the selected period of system inactivity (1 minute to 1 hour), the fixed disk drive and the video shut off while all other devices still operate at full speed. The default setting is Disabled.

Suspend Mode

After the selected period of system inactivity (1 minute to 1 hour), all devices except the CPU shut off. The default setting is Disabled.

HDD Power Down

After the selected period of drive inactivity (1 to 15 minutes), the hard disk drive powers down while all other devices remain active. The default is Disabled.

Throttle Duty Cycle

When the system enters Doze mode, the CPU clock runs only part of the time. You may select the percent of time that the clock runs. The default is 62.5%.

ZZ Active in Suspend

When Enabled, the ZZ signal is active during Suspend mode. The default is Disabled.

PCI/VGA Active Monitor

When Enabled, any video activity restarts the global timer for Standby mode. The default is Enabled.

Soft-Off by PWR-BTTN

When you select Instant Off or Delay 4 Sec., turning the system off with the on / off button places the system in a very low-power-usage state, either immediately or after 4 seconds, with only enough circuitry receiving power to detect power button activity or Resume by Ring activity. The default is Instant-Off.

Power On by Ring

When Enabled, an input signal on the serial Ring Indicator (RI) line (in other words, an incoming call on the modem) awakens the system from a soft off state. The default is Enabled.

IRQ8 Break Suspend

You can select Enabled or Disabled for monitoring of IRQ8 (the Real Time Clock) so it does not awaken the system from Suspend mode. The default is Disabled.

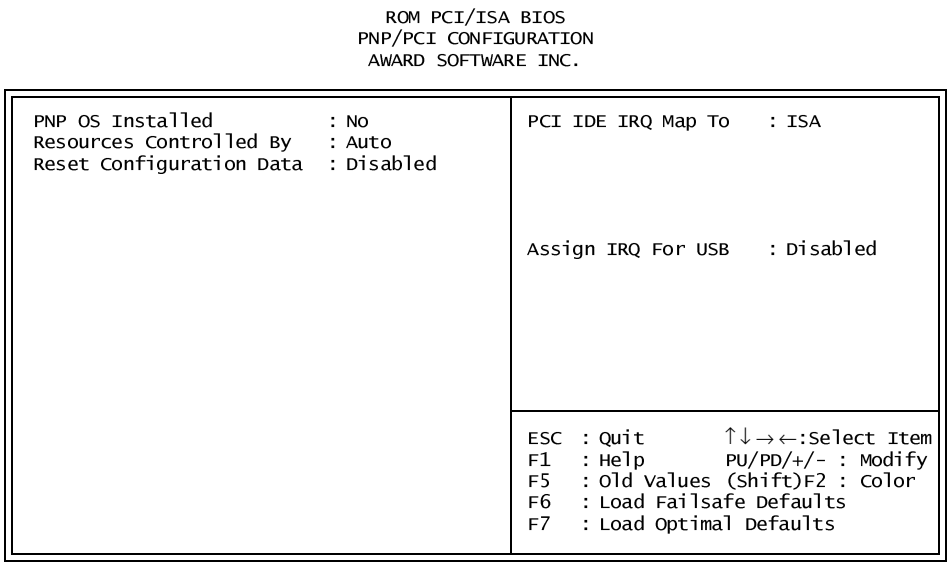
Reload Global Timer Events

When Enabled, an event occurring on each device listed below restarts the global timer for Standby mode.

- IRQ3-7, 9-15, NMI
- Primary IDE 0
- Primary IDE 1
- Secondary IDE 0
- Secondary IDE 1
- Floppy Disk
- Serial Port
- Parallel Port

PnP/PCI Configuration

This section describes the PNP / PCI options available.



PNP OS Installed

Select Yes if the system operating environment is Plug-and-Play aware (e.g., Windows 95). The default is No.

Resources Controlled By

The Plug-and-Play BIOS can automatically configure all the boot and Plug-and-Play compatible devices. If Auto is selected, all the interrupt request (IRQ) and DMA assignment and memory base address fields disappear, as the BIOS automatically assigns them. The default is Auto.

Reset Configuration Data

Select Enabled to reset Extended System Configuration Data (ESCD) when exiting Setup if a new add-on has been installed and the system reconfiguration has caused such a serious conflict that the operating system cannot boot. The default is Disabled.

IRQ *n* Assigned to

When resources are controlled manually, assign each system interrupt as one of the following types, depending on the type of device using the interrupt:

- Legacy ISA Devices compliant with the original PC AT bus specification, requiring a specific interrupt (such as IRQ4 for serial port 1).
- PCI/ISA PnP Devices compliant with the Plug-and-Play standard, whether designed for PCI or ISA bus architecture.

DMA *n* Assigned to

When resources are controlled manually, assign each system DMA channel as one of the following types, depending on the type of device using the interrupt:

- Legacy ISA Devices compliant with the original PC AT bus specification, requiring a specific DMA channel
- PCI/ISA PnP Devices compliant with the Plug-and-Play standard, whether designed for PCI or ISA bus architecture.

PCI IDE IRQ Map to

This field lets you select PCI IDE IRQ mapping or PC AT (ISA) interrupts. If your system does not have one or two PCI IDE connectors on the system board, select values according to the type of IDE interface(s) installed in your system (PCI or ISA). Standard ISA interrupts for IDE channels are IRQ14 for primary and IRQ15 for secondary. The default is ISA.

Primary/Secondary IDE INT#

Each PCI peripheral connection is capable of activating up to four interrupts: INT# A, INT# B, INT# C and INT# D. By default, a PCI connection is assigned INT# A. Assigning INT# B has no meaning unless the peripheral device requires two interrupt services rather than just one. Because the PCI/IDE interface in the chipset has two channels, it requires two interrupt services. The primary and secondary IDE INT# fields default to values appropriate for two PCI/IDE channels, with the primary PCI/IDE channel having a lower interrupt than the secondary. The defaults are:

- Primary IDE INT#A
- Secondary IDE INT#B

Used Mem base addr

Select a base address for the memory area used by any peripheral that requires high memory. The defaults are:

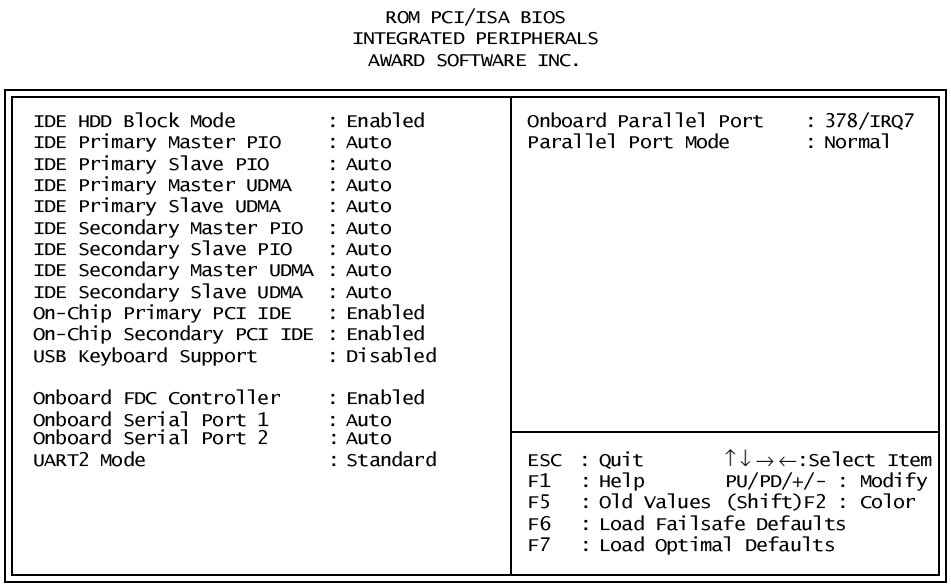
- Used MEM base addrN/A
- Assign IRQ For USBDisabled

Assign IRQ for USB

USB is not supported on this board. Assign IRQ for USB should always be Disabled.

Integrated Peripherals

This section describes the setup for integrated peripherals in the system.



IDE HDD Block Mode

Block mode is also called block transfer, multiple commands, or multiple sector read / write. If the IDE hard drive supports block mode, select Enabled for automatic detection of the optimal number of block read / writes per sector the drive can support. The default is Enabled.

IDE Primary Master PIO

This IDE PIO (Programmed Input / Output) field allows setting a PIO mode (0-4) for the IDE devices that the onboard IDE interface supports. Modes 0 through 4 provide successively increased performance. In Auto mode, the system automatically determines the best mode for each device. The default is Auto

IDE Primary Slave PIO

This IDE PIO (Programmed Input / Output) field allows setting a PIO mode (0-4) for the IDE devices that the onboard IDE interface supports. Modes 0 through 4 provide successively increased performance. In Auto mode, the system automatically determines the best mode for each device. The default is Auto

IDE Primary Master UDMA

Ultra DMA / 33 implementation is possible only if the IDE hard drive supports it and the operating environment includes a DMA driver (Windows 95 OSR2 or a third-party IDE bus master driver). If the hard drive and the operating system both support Ultra DMA / 33, select Auto to enable BIOS support. The default is Auto.

IDE Primary Slave UDMA

Ultra DMA / 33 implementation is possible only if the IDE hard drive supports it and the operating system includes a DMA driver (Windows 95 OSR2 or a third-party IDE bus master driver). If the hard drive and the operating system both support Ultra DMA / 33, select Auto to enable BIOS support. The default is Auto.

IDE Secondary Master PIO

This IDE PIO (Programmed Input / Output) field allows setting a PIO mode (0-4) for the IDE devices that the onboard IDE interface supports. Modes 0 through 4 provide successively increased performance. In Auto mode, the system automatically determines the best mode for each device. The default is Auto

IDE Secondary Slave PIO

This IDE PIO (Programmed Input / Output) field allows setting a PIO mode (0-4) for the IDE devices that the onboard IDE interface supports. Modes 0 through 4 provide successively increased performance. In Auto mode, the system automatically determines the best mode for each device. The default is Auto

IDE Secondary Master UDMA

Ultra DMA / 33 implementation is possible only if the IDE hard drive supports it and the operating environment includes a DMA driver (Windows 95 OSR2 or a third-party IDE bus master driver). If the hard drive and the operating system both support Ultra DMA / 33, select Auto to enable BIOS support. The default is Auto.

IDE Secondary Slave UDMA

Ultra DMA / 33 implementation is possible only if the IDE hard drive supports it and the operating system includes a DMA driver (Windows 95 OSR2 or a third-party IDE bus master driver). If the hard drive and the operating system both support Ultra DMA / 33, select Auto to enable BIOS support. The default is Auto.

On-Chip Primary PCI IDE

The integrated peripheral controller contains an IDE interface with support for two IDE channels. Select Enabled to activate each channel separately. The default is Enabled.

On-Chip Secondary PCI IDE

The integrated peripheral controller contains an IDE interface with support for two IDE channels. Select Enabled to activate each channel separately. When this option is disabled, the IDE Primary Master PIO and IDE Primary Slave PIO options in this screen are removed. The default is Enabled.

USB Keyboard Support

Select Enabled if the operating system contains a Universal Serial Bus (USB) controller and you have a USB keyboard. The default is Disabled.

Onboard FDC Controller

Select Enabled to activate the floppy disk controller (FDC) installed on the system board. If an add-in FDC was installed or the system has no floppy drive, select Disabled in this field. The default is Enabled.

Onboard Serial Port 1/2

Select an address and corresponding interrupt for the first and second serial ports. The default for both ports is Auto.

UART 2 Mode

Select an operating mode for the second serial port. Check the specifications of the infrared port to select the appropriate infrared mode. The options are:

- Standard Rs-232C serial port
- IrDA 1.0 IrDA-compliant serial infrared port
- MIR 0.57M Mid-infrared 0.57 MB port
- MIR 1.15M Mid-infrared 1.15 MB port
- FIR Fast infrared standard
- ASK IR Amplitude shift keyed infrared port

The default for this setting is Standard. All other setting are not supported and should not be used.

Onboard Parallel Port

Select an address and corresponding interrupt for the physical parallel (printer) port. The default setting is 378/IRQ7.

Parallel Port Mode

Select an operating mode for the onboard parallel (printer) port. The default setting is Normal.

Appendix

E

Device Configuration: I/O and Interrupt Control

Contents

BIOS Operations 112
Device Address Definition. 117
Device Interrupt Definition 119

Introduction

This appendix provides the user with the information needed to develop custom applications for the VMICPCI-7593. The CPU board on the VMICPCI-7593 is unique in that the BIOS can not be removed; it must be used in the initial boot cycle. A custom application, like a revised operating system for example, can only begin to operate after the BIOS has finished initializing the CPU. The VMICPCI-7593 will allow the user to either maintain the current BIOS configuration or alter this configuration to be more user specific, but this alteration can only be accomplished after the initial BIOS boot cycle has completed.

BIOS Operations

When the VMICPCI-7593 is powered on, control immediately jumps to the BIOS. The BIOS initiates a Power-on Self-Test (POST) program which instructs the microprocessor to initialize system memory as well as the rest of the system. The BIOS establishes the configuration of all on-board devices by initializing their respective I/O and Memory addresses and interrupt request lines. The BIOS then builds an interrupt vector table in main memory, which is used for interrupt handling. The default interrupt vector table and the default address map is described in Chapter 3 of this manual. Finally, the BIOS jumps to the selected boot device. This is the point at which a custom operating system could take over control of the board and proceed with a custom configuration and/or custom application. A user application could override the configuration set by the BIOS and reconfigure the system or it could accept what the BIOS initialized.

BIOS Control Overview

There are two areas on the VMICPCI-7593 in which the user must be familiar in order to override the initial BIOS configuration. These include the device addresses and the device interrupts. This appendix reviews the details of these addresses and interrupts, and provides a reference list for the individual devices used on the board.

The VMICPCI-7593 utilizes the high-performance Peripheral Component Interconnect (PCI) bus along with the Industrial Standard Architecture (ISA) bus. In general, the PCI bus is plug-and-play compatible. The components that are connected to the PCI bus are not always placed at a standard I/O or Memory address, nor are they connected to a standard interrupt request line as is the case with ISA bus devices. These PCI bus devices are re-established by the BIOS during every boot cycle, meaning that these devices will not always be located at the same address or connected to the same interrupt request line every time the CPU is booted. This appendix lists the defaults that are found by powering up a specific VMICPCI-7593.

Functional Overview

The block diagram included in Figure E-1 on page 113 illustrates the VMICPCI-7593 emphasizing the I/O features, including the PCI-to-PCI bridge.

The circled number in the upper right corner of a function block references the appropriate data book necessary for the programming of the function block.

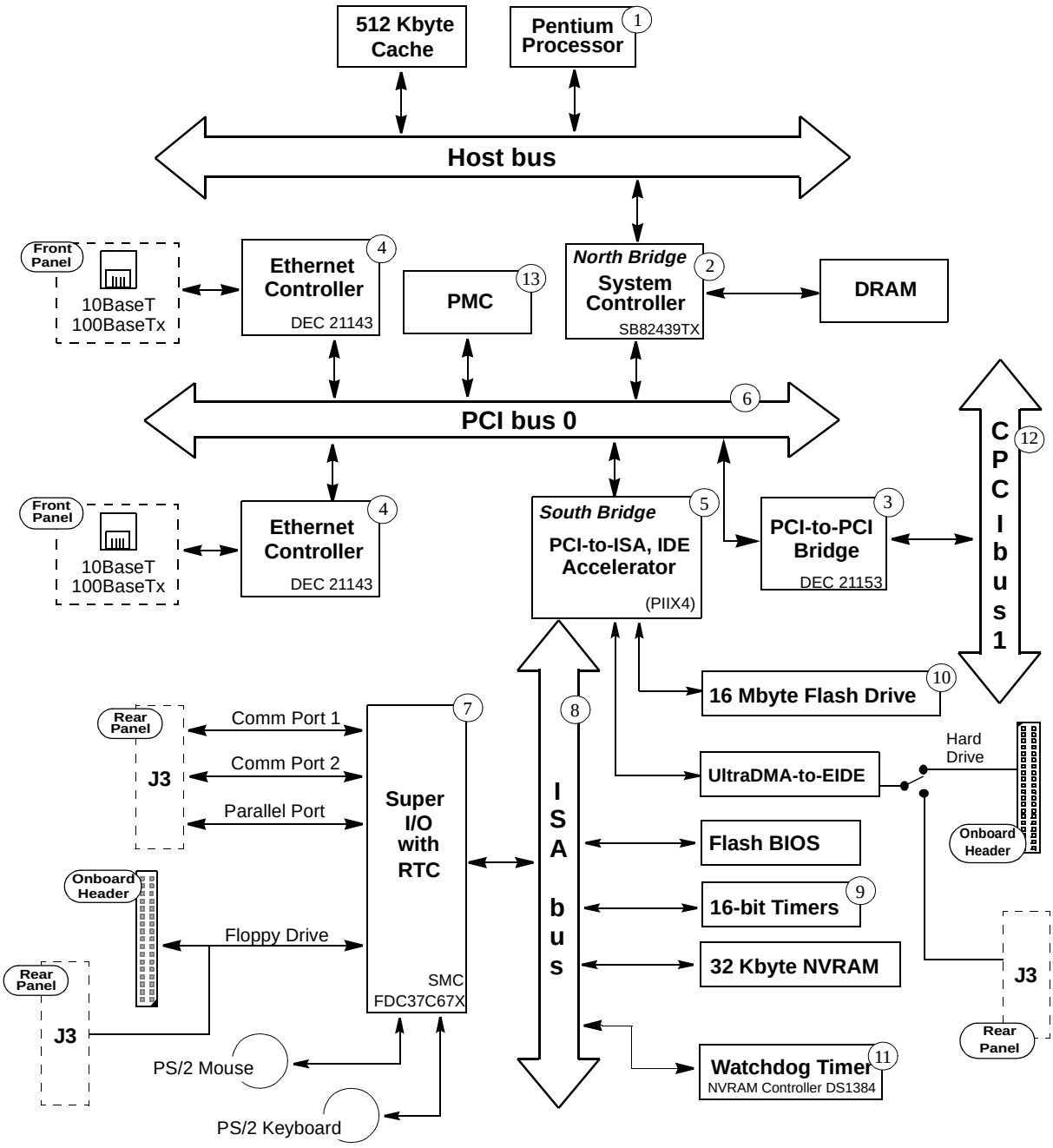


Figure E-1 VMICPCI-7593 Block Diagram

Data Book References

1. Pentium® Processor with MMX™ Technology

   January 1997, Order Number 243185-001
   Intel Corporation
   2200 Mission College Blvd.
   P.O. Box 58119
   Santa Clara, CA 95052-8119
   (408) 765-8080

2. Intel 82439TX PCIset System Controller (MTKE)

   Intel Corporation
   2200 Mission College Boulevard
   P.O. Box 58119
   Santa Clara, CA 95052-8119

3. Digital Semiconductor 21153 PCI-to-PCI Bridge

   Digital Equipment Corporation
   Maynard, Mass.
   1-800-332-2717
   www.digital.com

4. Digital Semiconductor 21143 10/100 Mb/s Ethernet LAN Controller

   Digital Equipment Corporation
   Maynard, Mass.
   1-800-332-2717
   www.digital.com

5. Intel 82430TX PCIset ISA Bridge

   82371AB PCI ISA IDE Xcellerator (PIIX4)
   2200 Mission College Boulevard
   P.O. Box 58119
   Santa Clara, CA 95052-8119

6. PCI Local Bus Specification, Rev. 2.1

   PCI Special Interest Group
   P.O. Box 14070
   Portland, OR 97214
   (800) 433-5177 (U.S.)
   (503) 797-4207 (International)
   (503) 234-6762 (FAX)

7. SMC FDC37C67X Enhanced Super I/O Controller

   SMC Component Products Division
   300 Kennedy Drive
   Hauppauge, NY 11788
   (516) 435-6000
   (516) 231-6004 (FAX)

8. ISA & EISA, Theory and Operation

   Solari, Edward
   Annabooks
   15010 Avenue of Science, Suite 101
   San Diego, CA 92128 USA
   ISBN 0-929392 -15-9

9. 82C54 CHMOS Programmable Internal Timer

   Intel Corporation
   2200 Mission College Boulevard
   P.O. Box 58119
   Santa Clara, CA 95052-8119

10. Flash ChipSet Product Manual

    SanDisk Corporation
    140 Caspian Court
    Sunnyvale, CA 94089-9820

11. DS 1384 Watchdog Timekeeping Controller

    Dallas Semiconductor
    4461 South Beltwood Pwky.
    Dallas, TX 75244-3292

12. *CompactPCI™* Specification PICMG 2.0 R2.1

    PCI Industrial Computer Manufactures' Group
    301 Edgewater Place
    Suite 220
    Wakefield, MA 01880
    (617) 224-1100
    (617) 224-1239 (FAX)
    www.pcimg.org (Web)

13. PCI Mezzanine Cards (PMC)

    IEEE Standards Department
    445 Hoes Lane, P.O. Box 1331
    Piscataway, NJ 08855-1331

14. VMICPCI-7594 Specification

    800-007593-000 Product Specification
    VMIC
    12090 South Memorial Parkway
    Huntsville, Alabama 35803-3308
    (800) 322-3616
    (256) 880-0444
    (256) 882-0859 (FAX)

Device Address Definition

The standard PC / AT architecture defines two distinctive types of address spaces for the devices and peripherals on the CPU board. These spaces have typically been named Memory address space and I / O address space. The boundaries for these areas are limited to the number of address bus lines that are physically located on the CPU board. The VMICPCI-7593 has 32 address bus lines located on the board, thereby defining the limit of the address space as 4 Gbyte. The standard PC / AT architecture defines Memory address space from zero to 4 Gbyte and the separate I / O address space from zero to 64 Kbyte.

ISA Devices

The ISA devices on the VMICPCI-7593 are configured by the BIOS at boot-up and fall under the realm of the standard PC / AT architecture. They are mapped in I / O address space within standard addresses and their interrupts are mapped to standard interrupt control registers. However, all of the ISA devices with the exception of the real-time clock, keyboard, and programmable timer are relocatable to almost anywhere within the standard 1 Kbyte of I / O address space. Table E-1 below defines the spectrum of addresses available for reconfiguration of ISA devices.

As previously stated, in the standard PC / AT system, all I / O devices are mapped in I / O address space; however, one exception exists. The Dynamic Random Access Memory (DRAM), Battery-Backed SRAM, and Watchdog Timer are addressed in Memory address space. The BIOS places DRAM at address zero and extends to the physical limit of the on-board DRAM.

Table E-1 ISA Device Mapping Configuration

Device	Memory Space	I/O Address Space	PIC Interrupt Options	Byte Address Boundary	Default
Floppy	N / A	[0x100 - 0xFF8]	IRQ1 - IRQ15	8	$3F0
Parallel Port	N / A	[0x100 - 0xFFC] [0x100 - 0xFF8]	IRQ1 - IRQ15	4 8	$378
Serial Port 1	N / A	[0x100 - 0xFF8]	IRQ1 - IRQ15	8	$3F8
Serial Port 2	N / A	[0x100 - 0xFF8]	IRQ1 - IRQ15	8	$2F8
Real-Time Clock	Nonrelocatable				
Keyboard	Nonrelocatable				
Auxiliary I / O	N / A	- Primary I / O [0x000 - 0xFFF] - Secondary I / O [0x000 - 0xFFF]	IRQ1, IRQ3-IRQ15	1 1	
Programmable Timer		Nonrelocatable 0X500	IRQ5	4	0X500
Watchdog Timer	Nonrelocatable 0XD8000	N / A	N / A	0XD	0XD8000
Battery-Backed SRAM	Nonrelocatable 0XD800E	N / A	N / A	(32K-0XD)	0XD800E

PCI Devices

PCI devices are fully configured under I/O and/or Memory address space. Table E-1 describes the PCI bus devices that are on-board the VMICPCI-7593 along with each device's configuration spectrum.

The PCI bus includes three physical address spaces. As with ISA bus, PCI bus supports Memory and I/O address space, but PCI bus includes an additional Configuration address space. This address space is defined to support PCI bus hardware configuration (refer to the PCI bus Specification for complete details on the configuration address space). PCI bus targets are required to implement Base Address registers in configuration address space to access internal registers or functions. The BIOS uses the Base Address register to determine how much space a device requires in a given address space and then assigns where in that space the device will reside. This functionality enables PCI devices to be located in either Memory or I/O address space.

Table E-2 PCI Device Mapping Configuration

Device	Memory Space	I/O Address Space	PIC Interrupt Options
PCI-to-PCI Bridge	Anywhere	N/A	PCI Defined
Ethernet	Anywhere	Anywhere	PCI Defined
Timer Interrupt Registers	N/A	Fixed	N/A
PMC	Anywhere	Anywhere	PCI Defined

Device Interrupt Definition

PC/AT Interrupt Definition

The interrupt hardware implementation on the VMICPCI-7593 is standard for computers built around the PC / AT architecture. The PC / AT evolved from the IBM PC / XT architecture. In the IBM PC / XT systems, only eight Interrupt Request (IRQ) lines exist, numbered from IRQ0 to IRQ7. These interrupt lines were included originally on a 8259A Priority Interrupt Controller (PIC) chip.

The IBM PC / AT computer added eight more IRQx lines, numbered IRQ8 to IRQ15, by cascading a second slave 8259A PIC into the original master 8259A PIC. The interrupt line IRQ2 at the master PIC was committed as the cascade input from the slave PIC. This master / slave architecture, the standard PC / AT interrupt mapping, is illustrated in Figure E-2 on page 120 within the PCI-to-ISA Bridge PIIX4 82371AB section of the diagram.

To maintain backward compatibility with PC / XT systems, IBM chose to use the new IRQ9 input on the slave PIC to operate as the old IRQ2 interrupt line on the PC / XT Expansion Bus. Thus, in AT systems, the IRQ9 interrupt line connects to the old IRQ2 pin on the AT Expansion Bus (or ISA bus).

The BIOS defines the PC / AT interrupt line to be used by each device. The BIOS writes to each of the two cascaded 8259A PIC chips an 8-bit vector which maps each IRQx to its corresponding interrupt vector in memory.

ISA Device Interrupt Map

The VMICPCI-7593 BIOS maps the IRQx lines to the appropriate device per the standard ISA architecture. Reference Figure E-2 on page 120. This initialization operation cannot be changed; however, a custom application could reroute the interrupt configuration after the BIOS has completed the initial configuration cycle.

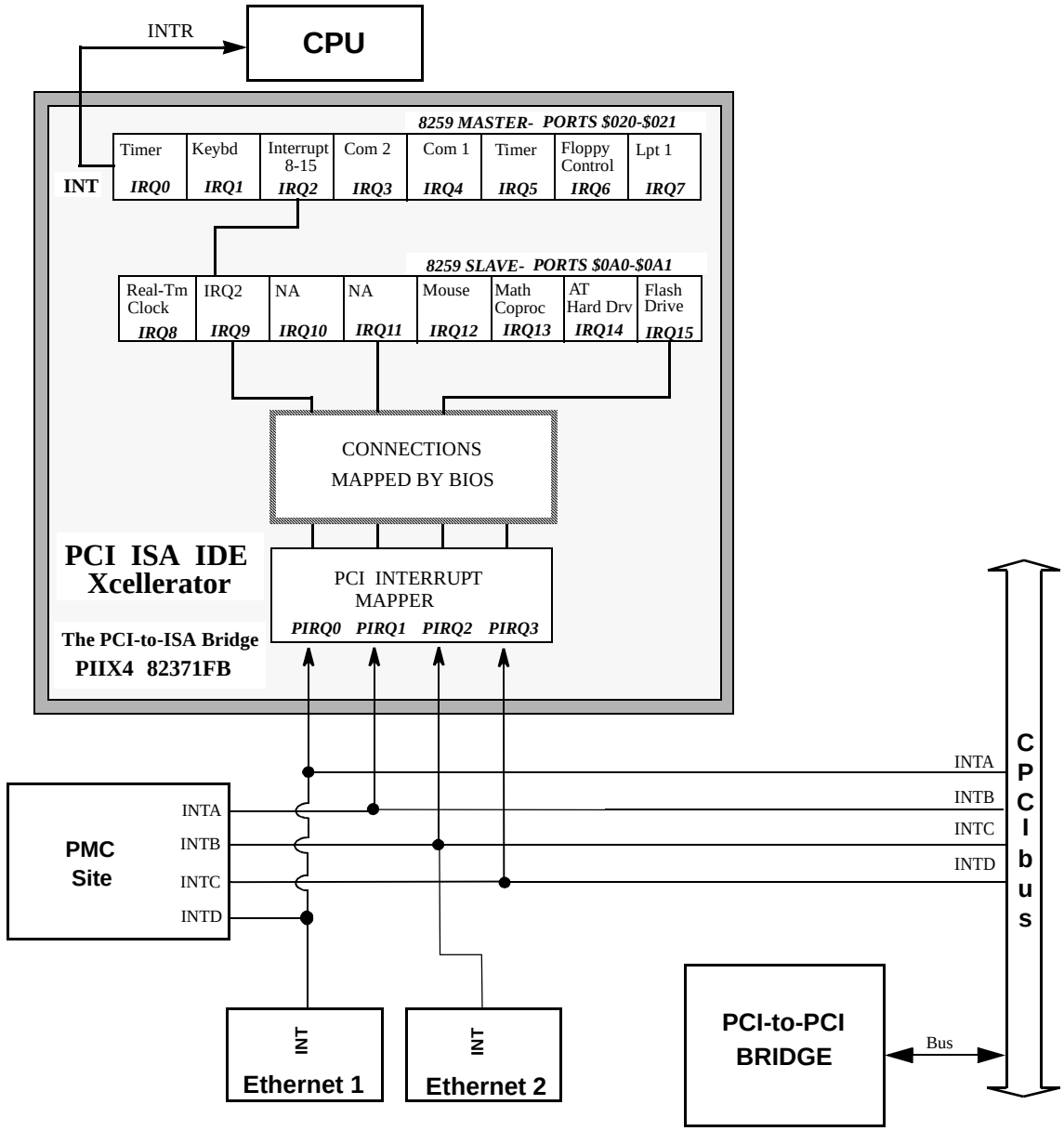


Figure E-2 BIOS Default Connections for the PC Interrupt Logic Controller

Note

See Table B-3 for the PCI Device Interrupt Mapping by the BIOS.

PCI Device Interrupt Map

The PCI bus-based external devices include the PMC site, Ethernet Controller, and the PCI-to-PCI bridge. The default BIOS maps these external devices to the PCI Interrupt Request (PIRQx) lines of the PIIX4. This mapping is illustrated in Figure E-2 on page 120 and is defined in Table E-3.

The device PCI interrupt lines (INTA through INTD) that are present on each device *cannot* be modified.

Table E-3 PCI Device Interrupt Mapping by the BIOS

DEVICE	COMPONENT	VENDOR ID	DEVICE ID	CPU ADDRESS MAP ID SELECT	DEVICE PCI INTERRUPT	MOTHER-BOARD PCI INTERRUPT MAPPER	DATA BOOK REF. #	Arbitration Level
Power Management	PIIX4 82371AB Function	0x8086	0x7113	AD18	N / A	N / A	2	N / A
PCI-to-ISA Bridge	PIIX4 82371AB Function 00	0x8086	0x7000	AD18	N / A	N / A	2	N / A
Ethernet Controller No. 2	DEC 21143	0x1011	0x0019	AD22	INTC	PIRQ2	5	REQ1
PCI IDE Controller	PIIX4 82371AB Function 01	0x8086	0x7111	AD18	N / A	N / A	2	N / A
PCI Host Bridge	Intel 430 TX	0x8086	0x7100	N / A	N / A	N / A	1	N / A
Ethernet Controller No.1	DEC 21143	0x1011	0x0019	AD19	INTA	PIRQ0	5	REQ2
PMC	N / A	N / A	N / A	AD27	INTB	PIRQ1	16	REQ0
Compact PCI Bridge	DEC 21153	0x1011	0x0025	AD26	N / A	N / A	4	REQ3

****** Not connected, for reference only
******* PIRQ4 interrupt is not enabled by the BIOS.

The motherboard accepts these PCI device interrupts through the PCI interrupt mapper function. The BIOS default maps the PCI Interrupt Request (PIRQx) external device lines to one of the available slave PIC Interrupt Request lines, IRQ (9, 10, 11, or 15). The BIOS default mapping of the PIRQx to the slave PIC is defined in Table E-4.

Table E-4 Default PIRQx to IRQx BIOS Mapping with all Devices Loaded

PCI INTx	PIC IRQx
PIRQ0	IRQ11
PIRQ1	IRQ10
PIRQ2	IRQ9
PIRQ3	N / C

Using the interrupt steering registers of the 82371FB PIIX3, the user can override the BIOS defaults and map any of the PCI interrupts (PIRQ0-3) to any of the following PIC IRQx (ISA) interrupts: IRQ15, 14, 12-9, or 7-3.



If PCI interrupts are remapped by the user, care must be taken to ensure that all ISA and PCI functions that require an interrupt are included.

Appendix

F

Sample C Software

Contents

Directory SRAM 124
Directory Timers 127
Directory WATCHDOG 143

Introduction

This appendix provides listings of a library of sample code that the programmer may utilize to build applications. These files are provided to the VMICPCI-7594 user on disk 320-500018-004, Sample Application C Code for the VMICPCI-7594, included in the distribution disk set.

Because of the wide variety of environments in which the VMICPCI-7594 operates, the samples provided in this appendix are not necessarily intended to be verbatim boilerplates. Rather, they are intended to give the end user an example of the standard structure of the operating code.

Directory SRAM

**File: T_SRAM.C

```
/*************************************************************************/
/* FILE: T_SRAM.C                                                        */
/*                                                                       */
/* Test battery backed SRAM with patterns and data=address.              */
/*                                                                       */
/*                                                                       */
/*************************************************************************/
#include    <stdlib.h>
#include    <stdio.h>
#include    <dos.h>
unsigned char far * b_ptr;
unsigned int  far * w_ptr;
unsigned long far * l_ptr;
unsigned int  far * buf_ptr;
static unsigned long pat[4] = {
  0x55555555,
  0xCCCCCCCC,
  0x66666666,
  0xFFFFFFFF
};
void main( void ) {
unsigned long i, x;
unsigned char bdat;
unsigned char brd;
unsigned int  wdat;
unsigned int  wrd;
unsigned long ldat;
unsigned long lrd;
  printf("\nTesting 32K SRAM DATA/~DATA B/W/L/ADDR .............");
  buf_ptr = (unsigned int far *) MK_FP( 0xD800, 0x18 );
  /* fill and test buf with DATA/~DATA BYTES 4 patterns */
  for( x = 0; x < 4; x++ ) {
    b_ptr = (unsigned char far *) buf_ptr;
    bdat = (unsigned char) pat[x];
    for( i = 0x18; i < 0x8000; i++ ) {
      *b_ptr++ = bdat;
      bdat = ~bdat;
    }
    b_ptr = (unsigned char far *) buf_ptr;
```

```
  bdat = (unsigned char) pat[x];
  for( i = 0x18; i < 0x8000; i++ ) {
    brd = *b_ptr++;
    if( bdat != brd ) {
      printf("FAILED\nBYTE DATA @ ADDR: %Fp  WR: %.2X  RD: %.2X\n",
              --b_ptr, bdat, brd );
      exit( 1 );
    }
    bdat = (~bdat) & 0xFF;
  }
}
/* fill and test buf with DATA/~DATA WORDS 4 patterns */
for( x = 0; x < 4; x++ ) {
  w_ptr = (unsigned int far *) buf_ptr;
  wdat = (unsigned int) pat[x];
  for( i = 0x18; i < 0x8000; i+=2 ) {
    *w_ptr++ = wdat;
    wdat = ~wdat;
  }
  w_ptr = (unsigned int far *) buf_ptr;
  wdat = (unsigned int) pat[x];
  for( i = 0x18; i < 0x8000; i+=2 ) {
    wrd = *w_ptr++;
    if( wdat != wrd ) {
      printf("FAILED\nWORD DATA @ ADDR: %Fp  WR: %.4X  RD: %.4X\n",
              --w_ptr, wdat, wrd );
      exit( 1 );
    }
    wdat = ~wdat;
  }
}
/* fill and test buf with DATA/~DATA LONGS 4 patterns */
for( x = 0; x < 4; x++ ) {
  l_ptr = (unsigned long far *) MK_FP( 0xD800, 0x18 ); ;
  ldat = (unsigned long) pat[x];
  for( i = 0x18; i < 0x8000; i+=4 ) {
    *l_ptr++ = ldat;
    ldat = ~ldat;
  }
  l_ptr = (unsigned long far *) MK_FP( 0xD800, 0x18 );
  ldat = (unsigned long) pat[x];
  for( i = 0x18; i < 0x8000; i+=4 ) {
    lrd = *l_ptr++;
    if( ldat != lrd ) {
```

```
        printf("FAILED\nLONG DATA @ ADDR: %Fp  WR: %.8X  RD: %.8X\n",
                --l_ptr, ldat, lrd );
        exit( 1 );
      }
      ldat = ~ldat;
    }
  }
  /* fill and test buf with DATA = ADD LONG */
  for( x = 0; x < 4; x++ ) {
    l_ptr = (unsigned long far *) MK_FP( 0xD800, 0x18 );
    for( i = 0x18; i < 0x8000; i+=4 ) {
      *l_ptr++ = i;
    }
    l_ptr = (unsigned long far *) MK_FP( 0xD800, 0x18 );
    for( i = 0x18; i < 0x8000; i+=4 ) {
      lrd = *l_ptr++;
      if( lrd != i ) {
        printf("FAILED\nDATA=ADDR @ ADDR: %Fp  WR: %.8X  RD: %.8X\n",
                --l_ptr, i, lrd );
        exit( 1 );
      }
    }
  }
  printf("PASSED\n\n");
  exit( 0 );
} /* end main */
```

Directory Timers

This directory contains sample code useful in the creation of applications involving the VMIVME-7593's three software controlled 16-bit timers. The code is written for the control of a single timer, but can be utilized in generating code for any timer configuration. The timers are described in Chapter 4 of the manual.

**File: CPU.H

```
/*
**  FILE: T7593T.H
**
*/
#define     DID_PWR_MGM     0x7113
#define     VID_PWR_MGM     0x8086
#define     IRQ5            0x0D
#define     GPI_T1          0x80    /* PIX General Purpose Input 15 (tmr 1)     */
#define     GPI_T2          0x40    /* PIX General Purpose Input 14 (tmr 2)     */
#define     GPI_T3          0x20    /* PIX General Purpose Input 13 (tmr 3)     */
#define     GPO_T1          0x40    /* PIX General Purpose Output 30 (tmr 1)    */
#define     GPO_T2          0x10    /* PIX General Purpose Output 28 (tmr 2)    */
#define     GPO_T3          0x08    /* PIX General Purpose Output 27 (tmr 3)    */
#define     GPO_CLR         0xA7    /* PIX General Purpose Output CLR TMRS      */
#define     TIMER_CNTR1     0x00    /* Timer counter 1 offset                   */
#define     TIMER_CNTR2     0x01    /* Timer counter 2 offset                   */
#define     TIMER_CNTR3     0x02    /* Timer counter 3 offset                   */
#define     TIMER_CNTL      0x03    /* Timer control offset                     */
/*****************************************************************************/
/* 8254 Control word                                                         */
/*****************************************************************************/
#define     CW_SC0          0x00    /* W Selcect counter 0                  */
#define     CW_SC1          0x40    /* W Selcect counter 1                  */
#define     CW_SC2          0x80    /* W Selcect counter 2                      */
#define     CW_RBC          0xC0    /* W Read back command                      */
#define     CW_CLC          0x00    /* W Cntr latch command (cnt/stat)          */
#define     CW_SLC          0x00    /* W Status latch command           */
#define     CW_LSB          0x10    /* W LSB only                               */
#define     CW_MSB          0x20    /* W MSB only                               */
#define     CW_LSBMSB       0x30    /* W LSB first then MSB             */
#define     CW_M0           0x00    /* W Mode 0                                 */
#define     CW_M1           0x02    /* W Mode 1                                 */
#define     CW_M2           0x04    /* W Mode 2                                 */
#define     CW_M3           0x06    /* W Mode 3                                 */
#define     CW_M4           0x08    /* W Mode 4                                 */
#define     CW_M5           0x0A    /* W Mode 5                                 */
#define     CW_BCD          0x01    /* W Binary Coded Decimal           */
```

```
#define     CW_RB_CNT       0x00    /* W Read back count                    */
#define     CW_RB_STAT      0x00    /* W Read back status              */
#define     CW_RB_C0        0x02    /* W Read back counter 0           */
#define     CW_RB_C1        0x04    /* W Read back counter 1           */
#define     CW_RB_C2        0x08    /* W Read back counter 2           */
```

** File: PCI.H

```
define TRUE  1
#define FALSE 0

#define CARRY_FLAG              0x01

/* PCI Functions */
#define PCI_FUNCTION_ID         0xB1
#define PCI_BIOS_PRESENT        0x01
#define FIND_PCI_DEVICE         0x02
#define FIND_PCI_CLASS_CODE     0x03
#define READ_CONFIG_BYTE        0x08
#define READ_CONFIG_WORD        0x09
#define READ_CONFIG_DWORD       0x0A
#define WRITE_CONFIG_BYTE       0x0B
#define WRITE_CONFIG_WORD       0x0C
#define WRITE_CONFIG_DWORD      0x0D

/* PCI Return codes */

#define SUCCESSFUL              0x00
#define NOT_SUCCESSFUL          0x01

/* PCI Config Space Regs */
#define PCI_CS_VENDOR_ID
#define PCI_CS_DEVICE_ID
#define PCI_CS_COMMAND
#define PCI_CS_STATUS
#define PCI_CS_REVISION_ID
#define PCI_CS_CLASS_CODE
#define PCI_CS_BASE_ADDRESS_0
#define PCI_CS_BASE_ADDRESS_1
#define PCI_CS_BASE_ADDRESS_2
#define PCI_CS_BASE_ADDRESS_3
#define PCI_CS_BASE_ADDRESS_4
#define PCI_CS_BASE_ADDRESS_5
#define PCI_CS_INTERRUPT_LINE
#define PCI_CS_INTERRUPT_PIN

/* Prototypes */

int find_pci_device(unsigned short device_id,
                    unsigned short vendor_id,
```

```
                    unsigned short index,
                    unsigned char *bus_number,
                    unsigned char *device_and_function);

int read_configuration_area(unsigned char function,
                            unsigned char bus_number,
                            unsigned char device_and_function,
                            unsigned char register_number,
                            unsigned long *data);

int write_configuration_area(unsigned char function,
                             unsigned char bus_number,
                             unsigned char device_and_function,
                             unsigned char register_number,
                             unsigned long value);

void outpd(unsigned short port, unsigned long value);

unsigned long inpd(unsigned short port);
```

**File: PCI.C

```
#include <dos.h>
#include <stddef.h>
#include "pci.h"

#define HIGH_BYTE(ax) (ax >> 8)
#define LOW_BYTE(ax) (ax & 0xff)

int find_pci_device(unsigned short device_id,
            unsigned short vendor_id,
            unsigned short index,
            unsigned char *bus_number,
            unsigned char *device_and_function)
{
   int ret_status;
   unsigned short ax, bx, flags;
   _CX = device_id;
   _DX = vendor_id;
   _SI = index;
   _AH = PCI_FUNCTION_ID;
   _AL = FIND_PCI_DEVICE;

   geninterrupt(0x1a);

   ax = _AX;
   bx = _BX;
   flags = _FLAGS;

   if ((flags & CARRY_FLAG) == 0)
   {
     ret_status = HIGH_BYTE(ax);
     if (ret_status == SUCCESSFUL)
     {
       if (bus_number != NULL) *bus_number = HIGH_BYTE(bx);
       if (device_and_function != NULL) *device_and_function = LOW_BYTE(bx);
     }
   }
   else
   {
     ret_status = NOT_SUCCESSFUL;
   }
   return(ret_status);
}
```

```
int read_configuration_area(unsigned char function,
                            unsigned char bus_number,
                            unsigned char device_and_function,
                            unsigned char register_number,
                            unsigned long *data)
{
   int ret_status;
   unsigned short ax, flags;
   unsigned long ecx;

   _BH = bus_number;
   _BL = device_and_function;
   _DI = register_number;
   _AH = PCI_FUNCTION_ID;
   _AL = function;

   geninterrupt(0x1a);

   ecx = _ECX;
   ax = _AX;
   flags = _FLAGS;
   if ((flags & CARRY_FLAG) == 0)
   {
     ret_status = HIGH_BYTE(ax);
     if (ret_status == SUCCESSFUL)
     {
       *data = ecx;
     }
   }
   else
   {
     ret_status = NOT_SUCCESSFUL;
   }
   return(ret_status);
}

int write_configuration_area(unsigned char function,
                     unsigned char bus_number,
                     unsigned char device_and_function,
                     unsigned char register_number,
                     unsigned long value)
{
   int ret_status;
   unsigned short ax, flags;
```

```
   _BH = bus_number;
   _BL = device_and_function;
   _ECX = value;
   _DI = register_number;
   _AH = PCI_FUNCTION_ID;
   _AL = function;

   geninterrupt(0x1a);

   ax = _AX;
   flags = _FLAGS;
   if ((flags & CARRY_FLAG) == 0)
   {
     ret_status = HIGH_BYTE(ax);
   }
   else
   {
     ret_status = NOT_SUCCESSFUL;
   }
   return(ret_status);
}
void outpd(unsigned short port, unsigned long value)
{
   _DX = port;
   _EAX = value;
   __emit__(0x66, 0xEF);
}
unsigned long inpd(unsigned short port)
{
   _DX = port;
   __emit__(0x66, 0xED);
   return(_EAX);
}
/*
```

**File: T_Timers.C

```
/*****************************************************************************/
/*  FILE:   T_TIMERS.C                                                       */
/*          Test Timers (TIC is jumper selectable for 500 ns or 1 us)        */
/*                                                                           */
/*****************************************************************************/
#include  <stdlib.h>
#include  <stdio.h>
#include  <string.h>
#include  <conio.h>
#include  <ctype.h>
#include  <dos.h>
#include  "pci.h"
#include  "cpu.h"
/* TT.C function prototypes */
void do_exit( int );
/* TIMERS.C function prototypes */
void far interrupt irq_rcvd( void );
void init_timer_int( void );
void restore_orig_int( void );
void load_counter( int, unsigned int );
void read_counter( int, unsigned int *, unsigned char * );
/* global variables */
unsigned char bus, dev_func;
/* the following globals are used in other files as 'extern' variables */
unsigned char tmr_status, t1_stat, t2_stat, t3_stat;
unsigned int tmr_cnt, t1_cnt, t2_cnt, t3_cnt;
unsigned long t1_count;      /* counts no. of times timer 1 ISR entered */
unsigned long t2_count;      /* counts no. of times timer 2 ISR entered */
unsigned long t3_count;      /* counts no. of times timer 3 ISR entered */
unsigned int pwr_mgm_base;
unsigned int gpi_base;
unsigned int gpo_base;
unsigned int timer_base;
unsigned char pic1_org;
unsigned char gpo_org;
void main( int argc, char * argv[] )
{
  unsigned long t1, t2, t3;
  int test_int;
  unsigned long temp_dword;
  char user[80];
  timer_base = 0x500;
```

```
/* try to locate the power management device on the PCI bus */
test_int = find_pci_device(DID_PWR_MGM, VID_PWR_MGM, 0,
               &bus, &dev_func);
if(test_int != SUCCESSFUL)
{
   printf("\nUnable to locate power management device on PCI bus\n");
   do_exit( 1 );
}
/* get base address from config area */
test_int = read_configuration_area(READ_CONFIG_DWORD,
                                   bus, dev_func, 0x40, &temp_dword);
if(test_int != SUCCESSFUL)
{
 printf("\nUnable to read POWER MGM. BASE ADDRESS @ 0x40 in config space\n");
   do_exit( 1 );
}
pwr_mgm_base = temp_dword & 0x0000FFC0;
gpi_base = pwr_mgm_base + 0x31; /* PIX general purpose input bits 8-15 */
gpo_base = pwr_mgm_base + 0x37; /* PIX general purpose output bits 24-31*/
disable();
/* Read 8259 slave Programmable Interrupt controller */
pic1_org = inp(0x21) & 0xFF; /* slave mask bits */
/* disable interrupt 5 */
outp(0x21, (pic1_org | 0x20) ); /* 0 = enable 1 = disable */
enable();
gpo_org = inp( gpo_base ) & 0xFF;
/* setup timers interrupt service routine */
init_timer_int();
/*
** verify all three counters can generate an interrupt (counters 1,2,3)
*/
printf("\nTesting all three 16 bit counters for interrupt ....");
/* setup for interrupts to occur */
t1_count = 0;
t2_count = 0;
t3_count = 0;
t1 = 0;
t2 = 0;
t3 = 0;
tmr_status = 0;
test_int = 100;
/* load counters */
load_counter( 1, 0xFFFF );
do
```

```
{
  if( t1_count ) {
    t1++;
    break;
  }
  test_int--;
  delay( 1 );
} while( test_int );
/* disable timers by reloading the control word */
outp( timer_base + TIMER_CNTL, (CW_SC0 | CW_LSBMSB | CW_M2) );
tmr_status = 0;
test_int = 100;
load_counter( 2, 0xFFFF );
do
{
  if( t2_count ) {
    t2++;
    break;
  }
  test_int--;
  delay( 1 );
} while( test_int );
/* disable timers by reloading the control word */
outp( timer_base + TIMER_CNTL, (CW_SC1 | CW_LSBMSB | CW_M2) );
tmr_status = 0;
test_int = 100;
load_counter( 3, 0xFFFF );
do
{
  if( t3_count ) {
    t3++;
    break;
  }
  test_int--;
  delay( 1 );
} while( test_int );
/* disable timers by reloading the control word */
outp( timer_base + TIMER_CNTL, (CW_SC2 | CW_LSBMSB | CW_M2) );
/* clear all three status bits in GPI */
outp( gpo_base, ( gpo_org & GPO_CLR ) );
/* set all three GPO outputs to 1 to allow int status registers to function */
outp( gpo_base, ( gpo_org | GPO_T1 | GPO_T2 | GPO_T3 ) );
if( t1 && t2 && t3 )
{
```

```
    printf("PASSED\n");
  }
  else
  {
    printf("FAILED\n");
    if( !t1 ) printf("TIMER 1 failed\n");
    if( !t2 ) printf("TIMER 2 failed\n");
    if( !t3 ) printf("TIMER 3 failed\n");
    do_exit( 2 );
  }
  /* do orderly exit */
  do_exit( 3 );
} /* end main */
void do_exit( int xit_code )
{
  if( xit_code > 1 )  restore_orig_int();
  outp( gpo_base, gpo_org );
  if( xit_code == 3 ) xit_code = 0;
  exit( xit_code );
} /* do_exit */
```

**File: Timer.C

```
/*
**  FILE: TIMERS.C
**
*/
#include  <stdlib.h>
#include  <stdio.h>
#include  <dos.h>
#include  <ctype.h>
#include  <conio.h>
#include  "cpu.h"
/* function prototypes */
void far interrupt irq_rcvd( void );
void init_timer_int( void );
void restore_orig_int( void );
void load_counter( int, unsigned int );
void read_counter( int, unsigned int *, unsigned char * );
/* global variables */
extern unsigned long t1_count;    /* timer 1 count */
extern unsigned long t2_count;    /* timer 2 count */
extern unsigned long t3_count;    /* timer 3 count */
extern unsigned char tmr_status;
extern unsigned int gpi_base;
extern unsigned int gpo_base;
extern unsigned int timer_base;
extern unsigned char pic1_org;
extern unsigned char gpo_org;
void far interrupt (* old_vect)(void);
/*****************************************************************/
/*  init_timer_int()                                             */
/*                                                               */
/*  purpose: Using the interrupt assigned, the original vector is  */
/*           saved and the vector to the new ISR is installed. The */
/*           programmable-interrupt-controller (PIC) is enabled.   */
/*                                                               */
/*  Prerequisite: The interrupt line to be used must have          */
/*                already been loaded in the global variable.      */
/*                                                               */
/*****************************************************************/
/*  parameters: none                                             */
/*****************************************************************/
/*  return value: none                                           */
/*****************************************************************/
```

```
void init_timer_int( void )
{
  disable();
  old_vect = getvect( IRQ5 ); /* save vector for IRQ5 */
  setvect( IRQ5, irq_rcvd );
  /* enable interrupt 5 */
  outp(0x21, (pic1_org & 0xDF) ); /* 0 = enable 1 = disable */
  /* clear all three GPO inputs */
  outp( gpo_base, ( gpo_org & GPO_CLR ) );
  /* set all three GPO outputs to 1 to allow int status registers to function */
  outp( gpo_base, (gpo_org | GPO_T1 | GPO_T2 | GPO_T3 ) );
  enable();
} /* init_timer_int */
/*********************************************************************/
/*  restore_orig_int()                                               */
/*                                                                   */
/*  purpose: Using the interrupt assigned, the original vector is  */
/*           restored and the programmable-interrupt-controller    */
/*           is disabled.                                            */
/*                                                                   */
/*  Prerequisite: The interrupt line to be used must have          */
/*                already been loaded in the global variable.      */
/*                                                                   */
/*********************************************************************/
/*  parameters: none                                                 */
/*********************************************************************/
/*  return value: none                                               */
/*********************************************************************/
void restore_orig_int( void )
{
  disable();
  outp(0x21, pic1_org);
  setvect( IRQ5, old_vect );
  enable();
} /* restore_orig_int */
/*********************************************************************/
/*  load_counter()                                                   */
/*                                                                   */
/*  purpose: Loads the appropriate counter with the count passed   */
/*                                                                   */
/*                                                                   */
/*********************************************************************/
/*  parameters: int counter = 1, 2, 3 for COUNTER 1, 2, or 3       */
/*              unsigned int count = count to be loaded            */
/*********************************************************************/
/*  return value: none                                               */
/*********************************************************************/
```

```
void load_counter( int counter, unsigned int count )
{
  int lsb, msb;
  lsb = count & 0xff;
  msb = count >> 8;
  switch( counter )
  {
    case 1: /* select counter 1, LSB then MSB, mode 2 */
      outp( timer_base + TIMER_CNTL, (CW_SC0 | CW_LSBMSB | CW_M2) );
      outp( timer_base + TIMER_CNTR1, (unsigned char) lsb );
      outp( timer_base + TIMER_CNTR1, (unsigned char) msb );
    break;
    case 2: /* select counter 2, LSB then MSB, mode 2 */
      outp( timer_base + TIMER_CNTL, (CW_SC1 | CW_LSBMSB | CW_M2) );
      outp( timer_base + TIMER_CNTR2, (unsigned char) lsb );
      outp( timer_base + TIMER_CNTR2, (unsigned char) msb );
    break;
    case 3: /* select counter 3, LSB then MSB, mode 2 */
      outp( timer_base + TIMER_CNTL, (CW_SC2 | CW_LSBMSB | CW_M2) );
      outp( timer_base + TIMER_CNTR3, (unsigned char) lsb );
      outp( timer_base + TIMER_CNTR3, (unsigned char) msb );
    break;
  }
} /* load_counter */
/*****************************************************************/
/*  read_counter()                                               */
/*                                                               */
/*  purpose: Reads the appropriate counter in the appropriate    */
/*           bank with the remainin count and status.            */
/*                                                               */
/*                                                               */
/*****************************************************************/
/*  parameters: int counter = 1, 2, 3 for COUNTER 1, 2, or 3     */
/*              unsigned int * count = remaining count           */
/*              unsigned char * status = counter status          */
/*****************************************************************/
/*  return value: none                                           */
/*****************************************************************/
void read_counter( int counter,
                   unsigned int * count, unsigned char * status )
{
  int lsb, msb;
  switch( counter )
  {
    case 1: /* select counter 1, LSB then MSB */
      outp( timer_base + TIMER_CNTL, ( CW_RBC | CW_RB_CNT | CW_RB_STAT | CW_RB_C0 )
);
```

```
      *status = inp( timer_base + TIMER_CNTR1 ) & 0xFF;
      lsb = inp( timer_base + TIMER_CNTR1 ) & 0xFF;
      msb = inp( timer_base + TIMER_CNTR1 ) & 0xFF;
      msb = msb << 8;
      *count = ( lsb | msb );
    break;
    case 2: /* select counter 2, LSB then MSB */
      outp( timer_base + TIMER_CNTL, ( CW_RBC | CW_RB_CNT | CW_RB_STAT | CW_RB_C1 )
);
      *status = inp( timer_base + TIMER_CNTR2 ) & 0xFF;
      lsb = inp( timer_base + TIMER_CNTR2 ) & 0xFF;
      msb = inp( timer_base + TIMER_CNTR2 ) & 0xFF;
      msb = msb << 8;
      *count = ( lsb | msb );
    break;
    case 3: /* select counter 3, LSB then MSB */
      outp( timer_base + TIMER_CNTL, ( CW_RBC | CW_RB_CNT | CW_RB_STAT | CW_RB_C2 )
);
      *status = inp( timer_base + TIMER_CNTR3 ) & 0xFF;
      lsb = inp( timer_base + TIMER_CNTR3 ) & 0xFF;
      msb = inp( timer_base + TIMER_CNTR3 ) & 0xFF;
      msb = msb << 8;
      *count = ( lsb | msb );
    break;
  }
} /* read_counter */
/*********************************************************************/
/*  irq_rcvd()                                                       */
/*                                                                   */
/*  purpose: Interrupt service routine used to service any of the  */
/*           counters on the 7593.                                   */
/*                                                                   */
/*                                                                   */
/*********************************************************************/
/*  parameters: none                                                 */
/*********************************************************************/
/*  return value: none                                               */
/*********************************************************************/
void interrupt irq_rcvd(void)
{
  disable();
  asm {
    .386P
    push  eax
```

```
    push  ebx
  }
  tmr_status = inp( gpi_base ) & 0xFF;
  /* increment counts and clear status */
  if( tmr_status & GPI_T1 ) {
    t1_count++;
    outp( gpo_base, (gpo_org & (~GPO_T1)) );  /* clear timer 1 status bit */
  }
  if( tmr_status & GPI_T2 ) {
    t2_count++;
    outp( gpo_base, (gpo_org & (~GPO_T2)) );  /* clear timer 2 status bit */
  }
  if( tmr_status & GPI_T3 ) {
    t3_count++;
    outp( gpo_base, (gpo_org & (~GPO_T3)) );  /* clear timer 3 status bit */
  }
  outp( gpo_base, (gpo_org | GPO_T1 | GPO_T2 | GPO_T3) ); /* enable status */
  /* Non specific end of interrupt to PIC */
  outp(0x20, 0x20);  /* Master end of irq command */
  asm {
    .386P
    pop  ebx
    pop  eax
  }
  enable();
}
```

Directory WATCHDOG

This directory contains sample code useful in the creation of applications involving the VMICPCI-7593's watchdog timer function as described in Chapter 4.

**File: Watchdog.H

```
/*
** DS1384 REGISTER OFFSETS
*/
                                /* 7  6  5  4  3  2  1  0           */
#define      CLK_MSEC    0x00    /* 00-99                            */
#define      CLK_SEC     0x01    /* 00-59 0                          */
#define      CLK_MIN     0x02    /* 00-59 0                          */
#define      CLK_MINAL   0x03    /* 00-59 M                          */
#define      CLK_HRS     0x04    /* 01-12+A/P OR 00-23               */
#define      CLK_HRSAL   0x05    /* 01-12+A/P OR 00-23               */
#define      CLK_DAY     0x06    /* 01-07 0  0  0  0  0              */
#define      CLK_DAYAL   0x07    /* 01-07 M  0  0  0  0              */
#define      CLK_DATE    0x08    /* 01-31 0  0                       */
#define      CLK_MONTH   0x09    /* 01-12       0                    */
#define      CLK_YRS     0x0A    /* 00-99                            */
#define      WD_CMD      0x0B    /* command register                 */
#define      WD_MSEC     0x0C    /* milli second watchdog time       */
#define      WD_SEC      0x0D    /* seconds watchdog time            */

/*
** DS1384 COMMAND REGSITER BIT DEFINITIONS
*/

#define      WD_TE       0x80    /* transfer enable 1 - allow updates  */
#define      WD_IPSW     0x40    /* interrupt switch 0 - WD out INTA   */
#define      WD_IBHL     0x20    /* int. B output 0 - current sink     */
#define      WD_PU       0x10    /* pulse/level 1 - 3 ms pulse         */
#define      WD_WAM      0x08    /* watchdog alarm mask 0 - active     */
#define      WD_TDM      0x04    /* time-of-day alarm mask 0 - active  */
#define      WD_WAF      0x02    /* watchdog alram flag                */
#define      WD_TDF      0x01    /* time-of-day flag                   */
```

**File: WDT0_RST.C

```
/*************************************************************************/
/* FILE: WDTO_RST.C                                                      */
/*                                                                       */
/* Setup watchdog to issue reset on time out.                            */
/*                                                                       */
/*                                                                       */
/*************************************************************************/
#include    <stdlib.h>
#include    <stdio.h>
#include    <dos.h>
#include    <time.h>
#include    <conio.h>
#include    <ctype.h>
#include    "watchdog.h"
unsigned char far * wd_ptr;
time_t t;
char usr[80];
char reg_b;
void main( void ) {
  /* Install Jumper E6 */
  wd_ptr = (unsigned char far *) MK_FP( 0xD800, 0 );
  /* set WatchDog Alarm Mask 1 - deactivated and update with 0 time */
  *(wd_ptr + WD_CMD) = ( WD_TE | WD_WAM );
  *(wd_ptr + WD_MSEC) = 0; /* load with 0 to disable */
  *(wd_ptr + WD_SEC) = 0;  /* load with 0 to disable */
  *(wd_ptr + WD_CMD) = ( WD_TE | WD_WAM ); /* allow update with 0 time */
  *(wd_ptr + WD_CMD) = WD_WAM; /* set watchdog alarm mask to 1 */
  *(wd_ptr + WD_MSEC) = 0x99; /* 00.99 seconds */
  *(wd_ptr + WD_SEC) = 0x99;  /* 99.00 seconds */
  *(wd_ptr + WD_CMD) = ( WD_TE | WD_PU ); /* set for 3 ms pulse */
  printf("Reset time out in 99.99 seconds\n\n");
  time(&t);
  printf("START DATE & TIME: %.24s\n\n", ctime(&t) );
  do {
    time(&t);
    printf("CURRENT DATE & TIME: %.24s\r", ctime(&t) );
    delay( 250 );
  } while( !kbhit() );
} /* end main */
```

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP