VMIPMC-7441
Real-Time Timers/Flash Memory

Product Manual

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

500-757441-000 Rev. C

Notice:

The Battery jumper, E5, must be in the Installed position for the Nonvolatile SRAM to work properly.

COPYRIGHT AND TRADEMARKS

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

Table of Contents

Table of Contents 5

List of Figures 7

List of Tables 9

Overview 11
- Organization of the Manual 13
- References 13
- Safety Summary 15
- Safety Symbols Used in This Manual 16

Chapter 1 - Theory of Operation 17
- PCI Interface 17
- Flash Memory 19
  - Operating System Boot Feature 19
    - Expansion ROM BIOS 19
- Timers 20
  - Timer Structure 20
  - Timer Functionality 20
  - Polling 23
  - Timer Status 26
  - Timer Read-Back 26

Chapter 2 - Programming 29
- Flash Programming Utilities 29
- Timer Control Registers 30
  - Programming 30
    - Timer Width Control/System State (TWSS) Register, Offset 30h 32
    - Timer Enable/Interrupt (TEI) Register: Offset 34h 35
    - Timer Interrupt Status (TIS) Register, Offset 38h 36
- Timer Software Provided 37
  - Timer Software Examples 37

Example 1: 37
Example 2: 38

Chapter 3 - Configuration and Installation 39

Shipping Configuration 39
Physical Installation of the VMIPMC-7441 41
Flash Configuration 42
Formatting Flash Disk 44
Default Configuration 45
Nonvolatile Static Random Access Memory (SRAM) 46
Data Retention Power 46
Battery Replacement 46

Chapter 4 - Maintenance 49

Index 51

List of Figures

Figure 1	Block Diagram of the VMIPMC-7441	12
Figure 1-1	Flash Memory Partitions	19
Figure 1-2	VMIPMC-7441 Timer Block Diagram	21
Figure 3-1	Jumper Locations on the VMIPMC-7441	42

List of Tables

Table 1-1	PCI Configuration Space Registers	18
Table 1-2	Counter Value/Cycle Time Range Table (X is Counter Value)	22
Table 1-3	Counter Value/Cycle Time Comparison Table	22
Table 1-4	Read-Back Commands	26
Table 1-5	16-bit Read/Mode Command Example	27
Table 1-6	32-bit Read/Mode Command Example	27
Table 2-1	Timer Section Address Map	31
Table 2-2	Timer Width Control/System State (TWSS) Register, Offset 30h	32
Table 2-3	Timer Mode Register Values	33
Table 2-4	16-bit Wide Timer Counter Value Load Example	34
Table 2-5	32-bit Wide Timer Counter Value Load Example	34
Table 2-6	Timer Enable/Interrupt (TEI) Register: Offset 34h	35
Table 2-7	Timer Interrupt Status (TIS) Register, Offset 38h	36
Table 3-1	VMIPMC-7441 Shipping Configuration	40
Table 3-2	EXPBOOT/FBOOT State Description	43
Table 3-3	Flash OS Boot Feature Jumper Options	44
Table 3-4	SRAM Battery Jumper Option	46

Overview

Introduction

The VMIPMC-7441 is a PCI Mezzanine Card (PMC) specifically designed to support real-time operating system requirements of CPU applications. The VMIPMC-7441 features include three independent 32-bit timers, 128 Kbyte of nonvolatile SRAM, and 4 Mbyte of flash memory. Reference the block diagram in that follows on page 12.

The VMIPMC-7441 provides three independent 32-bit timers. Timers can be used to provide accurate timing, freeing the operating system from software timing loops.

The 128 Kbyte of battery-backed SRAM memory is provided to maintain system critical data even when power is lost. When the host power is removed, the data can be retained for more than two years. This feature minimizes the down time required to reinitialize the system to its preshutdown state.

The VMIPMC-7441 includes 4 Mbyte of flash memory providing maximum flexibility for applications which require bootable operation. Additionally, the lower 1 Mbyte of the 4 Mbyte flash can be hardware write-protected. Typically the real-time operating system resides in this lower portion, while the user application resides in the upper 2 Mbyte of flash. The flash memory, coupled with the bootable feature, provides CPU diskless operation required by many embedded applications.

The VMIPMC-7441 is designed to meet all requirements, both physical and electrical, of the IEEE P1386.1 PCI Mezzanine Card (PMC) Specification and the IEEE P1386 Common Mezzanine Card (CMC) Specification. The user can connect this board to any VMIC processor board configured to receive a PMC-type card. The VMIPMC-7441 also conforms to Revision 2.1 of the PCI Local Bus Specification.

Board Features

- Single-wide PMC mezzanine with front-panel bezel and mounting hardware
- 4 Mbyte of Flash memory
- 128 Kbyte of nonvolatile SRAM
- Three independent 32-bit timers

- Write-protectable lower 1 Mbyte of Flash memory via an on-board hardware jumper
- Each timer can be software selected to provide a periodic PCI interrupt
- Each timer has 1 µs resolution
- NVRAM power-off data retention of greater than two years
- VMIC's VxWorks option provides Board Support Packages for a variety of VMIC CPU products
- DOS-based flash-in utility provided

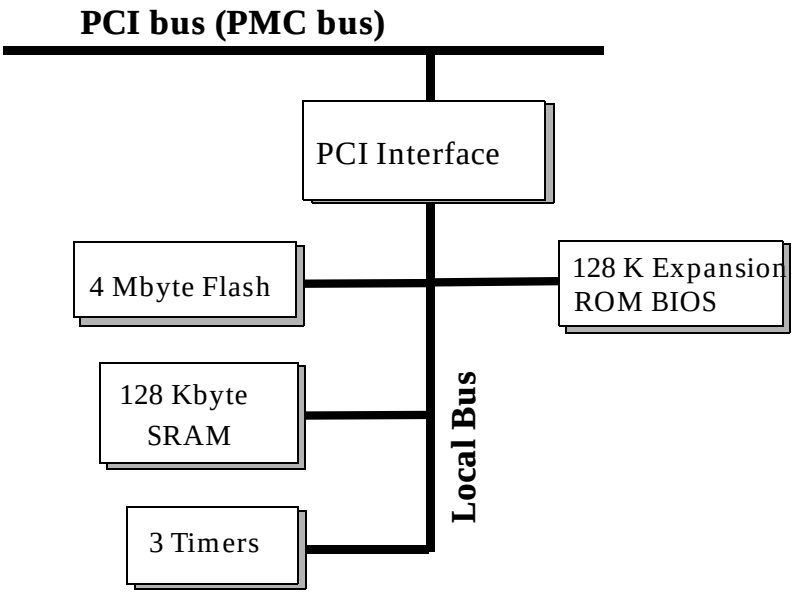


Figure 1 Block Diagram of the VMIPMC-7441

Organization of the Manual

This manual is composed of the following chapters and appendices:

Chapter 1 - Theory of Operation, describes the PCI interface, the use of Flash memory, and the operations of the Timer configurations.

Chapter 2 - Programming, describes the user-definable Flash memory and register settings of the VMIPMC-7441.

Chapter 3 - Configuration and Installation, describes physical installation and the Flash configuration of the product.

Chapter 4 -Maintenance, provides information relative to the care and maintenance of the unit.

References

For the most up-to-date physical description and specifications for the VMIPMC-7441, please refer to:

VMIC specification number:

800-757441-000

The following material applies to the VMIPMC-7441:

Microprocessor Products for Commercial and Military Digital Applications
Harris Semiconductor Literature Dept.
P.O. Box 883, MS 53-204
Melbourne, FL 32902
1-800-442-7747
FAX 407-724-7240

Flash Memory Products
1996 DataBook/Handbook
Advanced Micro Devices, Inc.
One AMD Place
P.O. Box 3453
Sunnyvale, CA 94088-3453
(408) 732-2400
(800) 538-8450
TWX: 910-339-9280
TELX: 34-6306
www.amd.com

The following IEEE Specifications apply to the VMIPMC-7441:

IEEE P1386.1 PMC Mezzanine Card (PMC) Specification
IEEE Standards Department
Copyright and Permissions
445 Hoes Lane, P.O. Box 1331
Piscataway, NJ 08855-1331

IEEE P1386 Common Mezzanine Card (CMC) Specification
IEEE Standards Department
Copyright and Permissions
445 Hoes Lane, P.O. Box 1331
Piscataway, NJ 08855-1331

PCI Local Bus Specification, Revision 2.1
PCI Special Interest Group
P.O.Box 14070
Portland, OR 97214
1-800-433-5177 (USA)
503-797-4207 (International)

Safety Summary

The following general safety precautions must be observed during all phases of the operation, service, and repair of this product. Failure to comply with these precautions or with specific warnings elsewhere in this manual violates safety standards of design, manufacture, and intended use of this product.

VMIC assumes no liability for the customer's failure to comply with these requirements.

Ground the System

To minimize shock hazard, the chassis and system cabinet must be connected to an electrical ground. A three-conductor AC power cable should be used. The power cable must either be plugged into an approved three-contact electrical outlet or used with a three-contact to two-contact adapter with the grounding wire (green) firmly connected to an electrical ground (safety ground) at the power outlet.

Do Not Operate in an Explosive Atmosphere

Do not operate the system in the presence of flammable gases or fumes. Operation of any electrical system in such an environment constitutes a definite safety hazard.

Keep Away from Live Circuits

Operating personnel must not remove product covers. Component replacement and internal adjustments must be made by qualified maintenance personnel. Do not replace components with power cable connected. Under certain conditions, dangerous voltages may exist even with the power cable removed. To avoid injuries, always disconnect power and discharge circuits before touching them.

Do Not Service or Adjust Alone

Do not attempt internal service or adjustment unless another person, capable of rendering first aid and resuscitation, is present.

Do Not Substitute Parts or Modify System

Because of the danger of introducing additional hazards, do not install substitute parts or perform any unauthorized modification to the product. Return the product to VMIC for service and repair to ensure that safety features are maintained.

Dangerous Procedure Warnings

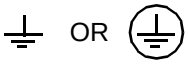
Warnings, such as the example below, precede only potentially dangerous procedures throughout this manual. Instructions contained in the warnings must be followed.

WARNING

Dangerous voltages, capable of causing death, are present in this system. Use extreme caution when handling, testing, and adjusting.

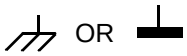
Safety Symbols Used in This Manual

Indicates dangerous voltage (terminals fed from the interior by voltage exceeding 1000 V are so marked).



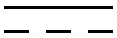
Protective conductor terminal. For protection against electrical shock in case of a fault. Used with field wiring terminals to indicate the terminal which must be connected to ground before operating equipment.

Low-noise or noiseless, clean ground (earth) terminal. Used for a signal common, as well as providing protection against electrical shock in case of a fault. Before operating the equipment, terminal marked with this symbol must be connected to ground in the manner described in the installation (operation) manual.



Frame or chassis terminal. A connection to the frame (chassis) of the equipment which normally includes all exposed metal structures.

Alternating current (power line).



Direct current (power line).

Alternating or direct current (power line).

The STOP symbol informs the operator the that a practice or procedure should not be performed. Actions could result in injury or death to personnel, or could result in damage to or destruction of part or all of the system.

The WARNING sign denotes a hazard. It calls attention to a procedure, a practice, a condition, which, if not correctly performed or adhered to, could result in injury or death to personnel.

The CAUTION sign denotes a hazard. It calls attention to an operating procedure, a practice, or a condition, which, if not correctly performed or adhered to, could result in damage to or destruction of part or all of the system.

The NOTE sign denotes important information. It calls attention to a procedure, a practice, a condition or the like, which is essential to highlight.

CHAPTER 1

Theory of Operation

Contents

PCI Interface 17
Flash Memory 19
Timers 20

Introduction

This section reviews the features provided with the VMIPMC-7441. The PCI interface is described. The product design, which allows for the booting of a user-defined operating system from Flash memory is described; and, the chapter concludes with a description of the operation of the product timers and control of the timers.

PCI Interface

The VMIPMC-7441 is compliant with both Peripheral Communication Interface (PCI) Specification 2.1 and PCI Mezzanine Card (PMC) P1386.1 Specification. The PCI Specification 2.1 requires that any interface to the PCI bus contain a PCI Configuration Register space as shown in Table 1-1. The registers are located in the configuration space and contain the base addresses of the four PCI memory windows utilized by the VMIPMC-7441.

Table 1-1 PCI Configuration Space Registers

31 16	15 00	Register Address
Device ID	Vendor ID	00h
Status	Command	04h
Class Code	Revision ID	08h
BIST / Header Type	Latency Timer / Cache Line Size	0Ch
PCI Base Address 0 for Memory-Mapped Configuration Registers		10h
PCI Base Address 1 for I/O-Mapped Configuration Registers		14h
PCI Base Address for the 4 Mbyte Flash		18h
PCI Base Address for the 128 Kbyte NVRAM		1Ch
PCI Base Address for the Timers		20h
Reserved		24h
Reserved		28h
Reserved	Reserved	2Ch
PCI Base Address for Expansion ROM BIOS		30h
Reserved		34h
Reserved		38h
Max_Lat / Min_gnt	Interrupt Pin / **Interrupt Line**	3Ch

Flash Memory

Flash memory provides to a system an in-circuit programmable, nonvolatile memory option similar to EEPROM, but with larger densities. The VMIPMC-7441 provides 4 Mbyte of Flash memory. The configuration allows for 32-, 16-, and 8-bit wide memory accesses. The PCI memory base address for the Flash memory is found in the VMIPMC-7441 PCI Configuration Register space at [*address*] offset 0x18 (reference Table 1-1 on page 18).

Operating System Boot Feature

The VMIPMC-7441 is designed with hardware and software features that allow booting to an Operating System (OS) resident in the lower portion of the Flash memory. Boot capability is provided by a 128 Kbyte local ROM (Expansion ROM BIOS), and configuration / write-protection jumpers.

When jumpered for boot operation (see Table 3-2 on page 43), the Expansion ROM BIOS configures the host CPU to view the lower 1.44 Mbyte of the Flash memory as a bootable drive (Drive A: or Drive B:).

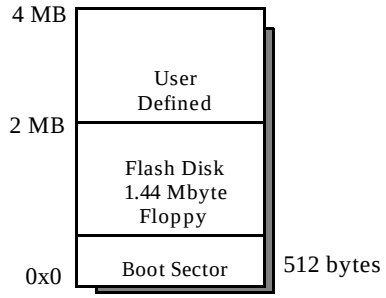


Figure 1-1 Flash Memory Partitions

Expansion ROM BIOS

When the EXPBOOT jumper is removed (not installed), the Expansion ROM BIOS is executed by the host CPU BIOS during bootup. The Expansion ROM BIOS redirects the CPU BIOS floppy drive handler to the Expansion ROM floppy handler. This handler is accessed either by host BIOS int13 or int40 depending on whether a hard drive is installed.

Once the Expansion ROM BIOS runs, the Lower Flash can be assessed as a write-protected A: or B: drive depending on the FBOOT jumper. If the Flash is set up as a drive, the user can still access the other floppy drive. The Expansion ROM BIOS int handler chains to the original int handler if the disk access is not for the Flash disk.

If the FBOOT jumper is configured for the Flash as the A: drive, the host BIOS int19 bootstrap is replaced by the Expansion BIOS int19 handler. This handler is executed by the host CPU BIOS to load the OS. The VMIPMC-7441 int19 handler copies the first 512 bytes from the Flash *drive* and jumps to it. The OS loader residing at the location then loads the OS via the int13 or int40 access.

Timers

There are many occasions in an industrial environment where the generation of accurate timing is required. The use of software alone to generate timing loops is awkward and wastes processor power. The hardware timers on-board the VMIPMC-7441 are designed to offload from software the task of generating timing loops. Instead of generating software loops, the software engineer can configure each of the VMIPMC-7441 timers to generate a periodic interrupt.

Timer Structure

The VMIPMC-7441 Timer segment contains three Timers and a Timer Control section as illustrated in Figure 1-2 on page 21. Each Timer is configured via software registers located within the Timer Control section. Each Timer is set up to be either 32 or 16 bits wide depending on the required time duration. Furthermore, as illustrated in the block diagram, each Timer consists of three different 16-bit counters (Scale Counter, Upper Counter, and Lower Counter).

The Timer Control section is the core of the Timer Structure. It contains the Timer Control circuitry, the Interrupt Status Block, and the Timer Control Registers. The Timer Control circuitry manages each Timer and signals the Interrupt Control Block when a Timer has timed out. The Interrupt Status Block maintains the Timer Interrupt Status (TIS) register and actually generates the interrupt to the host system (reference Table 2-1 on page 31). The Timer Control Registers contain all the registers which control the Timer process.

There are three Timer Control Registers within the Timer Control Section:

- The *Timer Width/System State Register* determines whether the Timers are 32 or 16 bits wide and also provides some board-level status based on the state of the jumpers.
- The *Timer Enable/Interrupt Register* enables counting within each timer and controls the Timer Interrupt masks.
- The *Timer Interrupt Status Register* provides the Timer status.

Timer Functionality

Timer setup involves the following sequence:

- Setup Timer width
- Load Counter Value into the Timer
- Enable the Timer

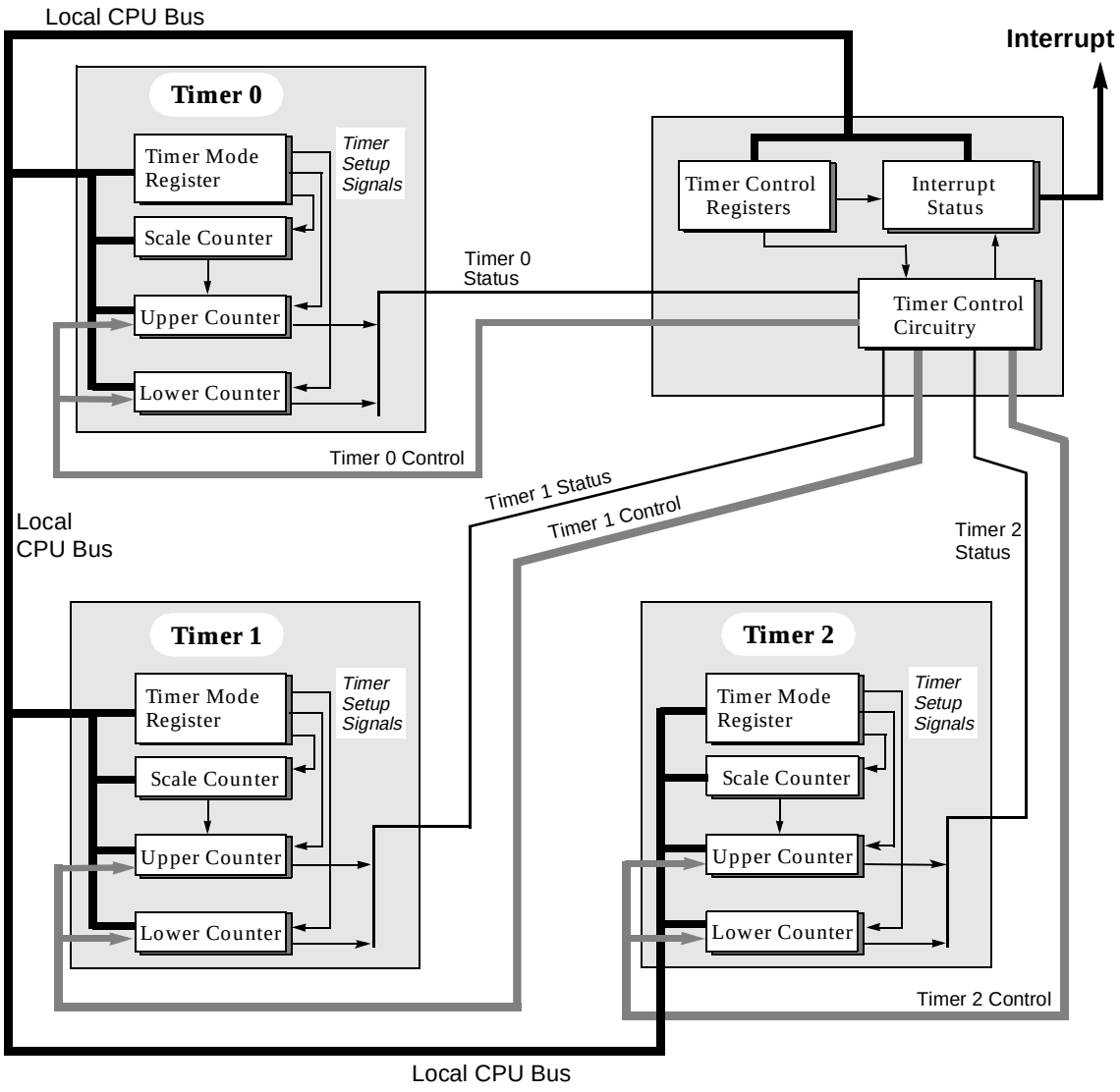


Figure 1-2 VMIPMC-7441 Timer Block Diagram

The Timers are set up by default to be 32 bits wide, but must be set to 16 bits wide for a time period of 65,538 µs or less. The Counter Value is loaded by first writing a Timer Mode byte (see Table 2-3 on page 33) to the Timer and then writing the actual Counter Value to the Timer. The Counter Value is defined in Table 2-4 on page 34 for 16-bit operation, or Table 2-5 on page 34 for 32-bit operation. After the mode and values are initialized, the timer is enabled.

At the end of the programmed time interval, the timer signals the Timer Control circuitry. The Timer Control circuitry sets a bit in the Timer Interrupt Status (TIS) register (reference Table 2-6 on page 35). The Timer Control circuitry then reinitializes the time to the original counter value and the Timer starts counting again. The Timer continues to loop in this fashion until disabled.

The host must read the TIS Register to determine the Timer status. The host can choose to poll the TIS register or wait for an interrupt. An interrupt is available for use by the three Timers. Each Timer's interrupt can be individually masked (reference Table 2-6 on page 35). The set bits and the pending interrupt are automatically cleared when the TIS register is read by the host. If more than one Timer is enabled, more than one bit could be set in the TIS Register, but only one interrupt is issued.

Each Timer is designed to provide an interrupt at repetitive time intervals based on the value placed in the timer. Each Timer has a resolution of 1.0 µs. A value of x placed in the timer generates an interrupt every (x+2)* 1.0 µs. For example, a value of 8 placed in the timer has an cycle time of 10.0 µs ([8 + 2] * 1.0 µs).

Each Timer has a cycle time range of between 3 µs to 71.58 minutes. Table 3-2 on page 43 shows the possible counter value ranges and their respective cycle time ranges. See Table 1-3 for a more concise description of the Counter Values and their corresponding cycle times.

Table 1-2 Counter Value / Cycle Time Range Table (X is Counter Value)

Counter Value (HEX)		Cycle Time (µs)			
From	To	From	To	Timer Width Setup	Description
0001	0000	3	65,538	16 bits	Time Value is (X+2) µs.
0001 0001	0000 0000	65,539	4,295,032,834	32 bits	Time Value is (X+2) µs.

Table 1-3 Counter Value / Cycle Time Comparison Table

Counter Value (HEX)	Cycle Time (µs)	Timer Width Setup
0001	3	16-bit
FFFF	65,537	16-bit
0000	65,538	16-bit
0001 0001	65,539	32-bit
0001 FFFF	131,073	32-bit

Table 1-3 Counter Value / Cycle Time Comparison Table (Continued)

Counter Value (HEX)	Cycle Time (µs)	Timer Width Setup
0001 0000	131,074	32-bit
0002 0001	131,075	32-bit
0002 FFFF	196,609	32-bit
0002 0000	196,610	32-bit
FFFF 0001	4,294,901,763	32-bit
FFFF FFFF	4,294,967,297	32-bit
FFFF 0000	4,294,967,298	32-bit
0000 0001	4,294,967,299	32-bit
0000 FFFF	4,295,032,833	32-bit
0000 0000	4,295,032,834	32-bit

The value 0001 (HEX) represents the shortest time duration; whereas, 0000 (HEX) represents the largest.

Polling

The VMICPMC-7441 Timers can be used as polled Timers. Two incidental characteristics of the timers must be kept in mind while polling. First, the timers, when counting in 16 or 32 bit mode, will always transition through an all 0xF state immediately prior to reinitialzation. For example, a typical count series for a 16 bit timer with an initial count of 0x0005 will be as follows:

State	Polled Count
6	0x0005
5	0x0004
4	0x0003
3	0x0002
2	0x0001
1	0x0000
0	0xFFFF
6	0x0005
5	0x0004

A typical series count for a 32 bit timer through the transition with an initial count of 0x00020140 will be as follows:

State	Polled Count
3	0x00000002
2	0x00000001
1	0x00000000
0	0xFFFFFFFF
131393	0x00020140
131392	0x0002013F

VMIC recommends that a value of one (1) be added to the polled value to obtain the correct count removing the all 0xF state. The 16 bit example with one (1) added to the polled value would be as follows:

State	Polled Count	Polled Count + 1
6	0x0005	0x0006
5	0x0004	0x0005
4	0x0003	0x0004
3	0x0002	0x0003
2	0x0001	0x0002
1	0x0000	0x0001
0	0xFFFF	0x0000
6	0x0005	0x0006
5	0x0004	0x0005

The 32 count example with one (1) added to the polled value would be as follows:

State	Polled Count	Polled Count +1
3	0x00000002	0x00000003
2	0x00000001	0x00000002
1	0x00000000	0x00000001
0	0xFFFFFFFF	0x00000000
131393	0x00020140	0x00020141
131392	0x0002013F	0x00020140

The second characteristic concerns overlapping counts when an initial count contains 0x0000 or 0xFFFF. Overlapping is when two numbers in a Timer count series appear to be the same but actually represent two different values. Count overlaps will occur when 0x0000 or 0xFFFF is loaded as the initial count in a 16 bit Timer. An overlap will occur when 0x0000 is loaded in either upper or lower words of a 32 bit Timer. An overlap will occur when 0xFFFF is loaded in both the upper and lower word of a 32 bit Timer.

	32 Bit Mode			16 Bit Mode
	Both Upper & Lower Word	Uper Word Only	Lower Word Only	
0x0000	X			X
0xFFFF	X	X	X	X

There are two reasons for the overlapping. First, 0x0000 represents two different values in the 82C54. The 82C54's largest loadable value is 0x0000 (which represents 0x10000, 65536 in decimal) but it actually decrements down through to 0x0000 (which is 0x00000, 0 in decimal). Second, because the Timers transition through an all 0xF state (as explained in the first characteristic) 0xFFFF represents a large value (65535) as well as the transitional value.

When polling the Timers VMIC recommends avoiding any initial value that uses 0x0000 or 0xFFFF in the manner described above. If the use of 0x0000 or 0xFFFF cannot be avoided VMIC recommends polling the Timer multiple times to determine the correct time.

Timer Status

As previously mentioned, each Timer is set to be polled or can be programmed to cause an interrupt as a result of timer expiration. Specific Upper and Lower Counters can be read to determine elapsed time since the previous read. However, the Timer / Interrupt Status register will most often be used to monitor Timer activity.

The Timer Interrupt / Status register is used to clear timer interrupts as well as to determine Timer rollover when interrupts are not being used. The Interrupt / Status bits are set when a Timer has rolled over. If the specific timer is set up to cause interrupts, the action of the bit being set causes an interrupt. Timers continue to count after the rollover (expire).

Timer Interrupt / Status register bits are automatically cleared as a result of the read. A rollover can be detected (or interrupt cleared) without an additional write being required to *clear* the bit.

All bits that are read as a **1** during the register read are cleared. This is desirable since all three timers share a single interrupt on the PCI bus. If multiple timer channels are configured to cause interrupts, the Timer Interrupt Service routine must be written to handle multiple bits being set within the Timer Interrupt / Status register.

Timer Read-Back

Once enabled, the Timers can be read to determine their present count. This is done by first issuing a Read-Back command to the particular Timer's TMR Register. The format of the Read-Back command is determined by whether the Timer to be read is 32 or 16 bits wide (see Table 1-4). The Read-Back command latches the Timer's present count into an output register (the Lower Counter only if it is in 16-bit mode or both the Lower and Upper Counter if it is in 32-bit mode). The count is read by reading the Timer Counters, Upper and / or Lower, LSB first and then MSB.

Table 1-4 Read-Back Commands

Mode	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value in HEX
16-bit Mode	1	1	0	1	0	1	0	0	D4
32-bit Mode	1	1	0	1	1	1	0	0	DC

The latched count is retained in the output registers until they are read. If several Read-Back commands are issued to the same Timer without reading the count, all but the first are ignored.

Table 1-5 shows an example sequence of reading the 16-bit count from Timer 0 set in 16-bit mode. When a Timer is setup in 16 bit mode, only the Lower Counter is used.

Table 1-6 shows an example sequence of reading the 32-bit count from Timer 1 set in

Table 1-5 16-bit Read/Mode Command Example

Step	Address offset (HEX)	Data (HEX)	Description
1	0C	D4	Write Read-Back command to Timer 0's TMR Register
2	04	LSB	Read the LSB of Lower Counter
3	04	MSB	Read the MSB of Lower Counter

32-bit mode. When a Timer is set up in 32-bit mode, all three Counters, Upper, Lower, and Scale, are used. However, only the Lower and Upper Counters need to be read.

Table 1-6 32-bit Read/Mode Command Example

Step	Address offset (HEX)	Data (HEX)	Description
1	1C	DC	Write Read-Back command to Timer 1's TMR Register
2	14	LSB	Read the LSB of the Lower Counter
3	14	MSB	Read the MSB of the Lower Counter
4	18	LSB	Read the LSB of the Upper Counter
5	18	MSB	Read the MSB of the Upper Counter

CHAPTER 2

Programming

Contents

Flash Programming Utilities 29
Timer Control Registers 30
Timer Software Provided 37

Introduction

This chapter directs the user to the Flash programming utilities available with the VMIPMC-7441 and also describes in detail the programming of the Timer Control registers of the product timers. The programming material is followed by example product software.

Flash Programming Utilities

Flash memory is normally in a read-only state. However, the VMIPMC-7441 package includes utilities which allow a user to erase and program this Flash memory. These utilities are included on a provided 3.5-inch, 1.44 Mbyte, DOS floppy disk titled: *Flash Utilities and Timer Example Code (*Part Number 320-500014-001). An included `readme.txt` file explains how to use the utilities.

Timer Control Registers

The VMIPMC-7441 Timer segment contains three Timers and a Timer Control section. reference the illustration in Figure 1-2 on page 21. Each Timer is configured via software registers located within the Timer Control section. The Timer Control section is the core of the Timer structure. It contains the Timer Control circuitry, the Interrupt Status Block, and the Timer Control Registers. The Timer Control Registers contain all the registers which control the Timer process. These register are defined as:

- The *Timer Width/System State Register (Offset 30h)* determines whether the Timers are 32 or 16 bits wide and also provides some board-level status based on the state of the jumpers.
- The *Timer Enable/Interrupt Register (Offset 34h)* enables counting within each timer and controls the Timer Interrupt masks.
- The *Timer Interrupt Status Register (Offset 38h)* provides the Timer status.

A detailed description of the programming of these registers follows.

Programming

Upon powerup of the VMIPMC-7441, the Timers are in an undefined state. Each Timer must be set up and enabled before it can be used. Each timer is completely independent of the others. Any times not used do not need to be set up.

The VMIPMC-7441 includes three Timers. Each Timer is implemented using an Intel 82C54 timer/counter chip. Each 82C54 contains three 16-bit counters. These counters are designated within the VMIPMC-7441 as the Scale Counter, the Lower Counter, and the Upper Counter.

Table 2-1 defines the Timer Section Address Map. All offsets are based from the PCI memory base address for the Timers. This base address is referred to a PCI base address and can be found in the VMIPMC-7441 PCI Configuration Register space at offset 0x20 (reference Table 1-1 on page 18). All data is transferred via the LSB (lower 8 bits) of the PCI data bus. All registers labeled *Register* are 8 bits wide. The Scale Counter, Lower Counter, and Upper Counter are 16 bits wide, and they are accessed using a byte-wide port using a defined sequence.

Depending on the desired cycle time, the upper counter and lower counter are cascaded to form a 32-bit timer. The cascading is controlled by the timer width register. In 32-bit mode, the scale counter is used to prescale the upper counter.

Table 2-1 Timer Section Address Map

Address Offset AD[5...0]	Segment	Description
00	Timer 0	Scale Counter (SC0)
04	Timer 0	Lower Counter (LC0)
08	Timer 0	Upper Counter (UC0)
0C	Timer 0	Timer Mode Register (TMR0)
10	Timer 1	Scale Counter (SC1)
14	Timer 1	Lower Counter (LC1)
18	Timer 1	Upper Counter (UC1)
1C	Timer 1	Timer Mode Register (TMR1)
20	Timer 2	Scale Counter (SC2)
24	Timer 2	Lower Counter (LC2)
28	Timer 2	Upper Counter (UC2)
2C	Timer 2	Timer Mode (TMR2) Register
30	Timer Control	Timer Width / System State (TWSS) Register
34	Timer Control	Timer Enable / Interrupt (TEI) Register
38	Timer Control	Timer Interrupt Status (TIS) Register
3C	Timer Control	Expansion ROM Read Enable Register

The Timers have two width modes, 16 and 32-bit wide, the choice of which is determined by the Cycle Time required by the Timer (see Table 1-3 on page 22). For a Cycle Time of 65,538 µs or less, the Timer is to be 16 bits wide. For a Cycle Time greater than 65,538 µs, the Timer is to be 32 bits wide.

The Timer must be 16 bits wide to load a cycle time equal to or less than 65,538 µs (see Table 1-3 on page 22).

The Timer width is controlled by the Timer Width Bit Field (bits 2 to 0) of the Timer Width / System State (TWSS) Register (see Table 2-2). This register is located at offset 0x30 from the Timer PCI memory base address. Each of the bits correspond to one of the three Timers. Bit 0 corresponds to Timer 0, Bit 1 corresponds to Timer 1, and Bit 2 corresponds to Timer 2. When the bit is set to a 1 (high), then the Timer is 32 bits wide, when the bit is set to a 0 (low), then the Timer is 16 bits wide. At powerup, the Timer Width Bits default to 32 bits wide state.

Timer Width Control/System State (TWSS) Register, Offset 30h

The Timer Width Control / System State (TWSS) Register, Offset 30h (Table 2-2) is used to provide board-level state information and control the width of the cascaded timers.

Table 2-2 Timer Width Control/System State (TWSS) Register, Offset 30h

Bit	Description	Read/ Write	Default
7	**Expansion ROM Bootable:** Displays the state of the EXPBOOT jumper. The jumper is used to enable / disable execution of the VMIPMC-7441 Expansion ROM BIOS during boot-up. If the value is 1, the EXPBOOT jumper is removed and the first Expansion ROM read after a hardware reset contains the actual contents of the first byte of the Expansion ROM. If the value is 0, the jumper is installed and the first Expansion ROM read after a hardware reset, contains all zeros and not the first byte of the Expansion ROM.	Read Only	Jumper State
6	**Flash Memory Bootable:** Displays the state of the FBOOT jumper. This jumper is used by Expansion ROM BIOS to determine if Flash is a bootable DOS Disk A: or DOS disk B:. If the value is 1, the jumper is removed and the Flash is set up to be a bootable DOS drive A:. If the value is 0 the jumper is installed and the Flash is set up to be DOS drive B:.	Read Only	Jumper State
5	**Flash Write Enable Control:** Displays the state of the FWECTRL jumper. This jumper is used by board circuitry to enable / disable writing to lower 1 Mbyte of the Bulk Flash memory. If the value is 1, the jumper is removed and writing to the bottom 1 Mbyte of Bulk Flash is disabled. If the value is 0, the jumper is installed and writing to the bottom 1 Mbyte of Bulk Flash memory is enabled.	Read Only	Jumper State
4 to 3	**Reserved**	Read Only	0
2	**Timer 2 Width:** Used by the Timer State Machine logic. Establishes whether the Timer is 32 or 16 bits wide. Each bit corresponds to a Timer. If the bit is set to a 1 (high), then the timer will be 32 bits wide. If the bit is set to a 0 (low), then the timer will be 16 bits wide.	R / W	1
1	Timer 1 Width	R / W	1
0	Timer 0 Width	R / W	1

The 16-bit width mode is implemented using one 16-bit counter of an 82C54 chip, specifically the Lower Counter. The 32-bit width mode is implemented using two 16-bit counters of an 82C54 chip. The most significant portion of the 32-bit Timer is referred to as the Upper Counter, while the least significant portion is referred to as the Lower Counter. The Upper Counter is clocked using the Scale Counter.

A single timer mode register is used to control the modes and loading of the scale, lower and upper counter within a 82C54 timer.

Before each individual Counter is loaded with its Counter Value, a unique counter-specific control byte must be written to the Timer Mode Register (TMRx). Table 2-3 shows the Timer Mode Bytes. More specifically, before a Counter Value is loaded into the Scale Counter, the Scale Timer Mode byte (36) must be written to the Timer Mode Register. Before a Counter Value is loaded into the Lower Counter the Lower Timer Mode byte (7A) must be written to the Timer Mode Register, etc.

Table 2-3 Timer Mode Register Values

Counter	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value in HEX
Scale	0	0	1	1	0	1	1	0	36
Lower	0	1	1	1	1	0	1	0	7A
Upper	1	0	1	1	1	0	1	0	BA

The loading of the actual Counter Value is different based on whether the Timer is 32 or 16 bits wide. For a Timer setup to be 16 bits wide, a 16-bit Counter Value is loaded into the Lower Counter (LCx). For a Timer setup to be 32 bits wide, the Lower Counter (LCx) is loaded with the least significant portion of the 32-bit Counter Value, while the Upper Counter (UCx) is loaded with the most significant portion of the 32-bit Counter Value. Also, when a Timer is 32 bits wide, the Scale Counter (SCx) must be loaded with zeros.

Although each of the three Counters within an 82C54 Timer are 16 bits wide, they are addressed via a single 8-bit address location. To load a Counter Value into one of the Counters, one must write two bytes, representing the Least Significant Byte (LSB) and the Most Significant Byte (MSB) of a 16-bit Counter Value, to the same 8-bit address location.

Note

The Least Significant Byte must be written first and then the Most Significant Byte.

Table 2-4 shows an example sequence of setting up a Timer Mode Byte to the Timer Mode Register (TMR0, address offset 0C (HEX)) and writing a counter value of 0x45AD to the Lower Counter (LC0, address offset 04 (HEX)) of Timer 0. Timer 0 has been set up previously to be 16 bits wide.

Table 2-4 16-bit Wide Timer Counter Value Load Example

Step	Address Offset (HEX)	Data (HEX)	Description
1	0C	7A	Timer Mode Register (TMR0) byte setting up the Lower Counter of Timer 0.
2	04	AD	LSB byte of the counter value written to LC0.
3	04	45	MSB byte of the counter value written to LC0.

Table 2-5 shows an example sequence of loading a counter value to Timer 0, which has been previously set up to be 32 bits wide. First, the Scale Counter is loaded with zero. Then a counter value of 01BC 45AD (HEX) is loaded into the Timer.

Table 2-5 32-bit Wide Timer Counter Value Load Example

Step	Address Offset (HEX)	Data (HEX)	Description
1	0C	36	Timer Mode Register (TMR0) byte setting up the Scale Counter 0 (SC0).
2	00	00	LSB byte with a value of zero written to the SC0.
3	00	00	MSB byte with a value of zero written to the SC0.
4	0C	7A	Timer Mode Register (TMR0) byte setting up the Lower Counter 0 (LC0).
5	04	AD	LSB byte of the LSW of the counter value written to the LC0.
6	04	45	MSB byte of the LSW of the counter value written to the LC0.
7	0C	BA	Timer Mode Register (TMR0) byte setting up the Upper Counter 0 (UC0).
8	08	BC	LSB byte of the MSW of the counter value written to the UC0.
9	08	01	MSB byte of the MSW of the counter value written to the UC0.

Timer Enable/Interrupt (TEI) Register: Offset 34h

The Final step in programming a Timer involves setting up the Timer Interrupt Masks and enabling the Timer. The Timer Enable / Interrupt (TEI) Register (Table 2-6) is used to do both of these tasks. It controls the turning on and off of each Timer.

Bits 5 to 3 control the masking of the interrupt from each Timer. When an Interrupt Mask bit is set to zero (0), the interrupt is not masked. When the bit is set to one (1), then the Timer Interrupt is masked.

Bits 2 to 0 of the Timer Enable / Interrupt (TEI) Register are the enable bits for each Timer, respectively. Bit 0 enables Timer 0, Bit 1 enables Timer 1, etc. When the bit is set to zero (0), the particular Timer is disabled. When the bit is set to a one (1), then the particular Timer is enabled.

Table 2-6 Timer Enable/Interrupt (TEI) Register: Offset 34h

Field	Description	Read/Write	Default
7 to 6	Reserved	R / W	0
5	Timer 2 Interrupt Mask	R / W	0
4	Timer 1 Interrupt Mask	R / W	0
3	Timer 0 Interrupt Mask	R / W	0
2	Enables Timer 2	R / W	0
1	Enables Timer 1	R / W	0
0	Enables Timer 0	R / W	0

Timer Interrupt Status (TIS) Register, Offset 38h

If more than one Timer is enabled, the host needs to read the Timer Interrupt Status (TIS) Register (see Table 2-7) to determine which of the enabled Timers counted through its count cycle. The TIS Register, Offset 38h address, displays which timer rolled through its count (expired). See Table 2-1 on page 31 for additional address information. The register is cleared immediately upon being read. Bits 2 to 0 are the status bits for each Timer, respectively. Bit 0 is the status bit for Timer 0, bit 1 is the status bit for Timer 1; the sequence continues.

Table 2-7 Timer Interrupt Status (TIS) Register, Offset 38h

Field	Description	Read/Write	Default
7 to 3	Reserved	Read Only	0
2	Timer 2 interrupt status	Read Only	0
1	Timer 1 interrupt status	Read Only	0
0	Timer 0 interrupt status	Read Only	0

Timer Software Provided

Provided with the VMIPMC-7441 package is a 3.5-inch 1.44 Mbyte DOS floppy that contains examples of code written in C showing how the Timers are programmed. These examples are located under the directory TIMERS. Under the same directory is a readme.txt file, which explains the different files and their uses.

The Timer Enable Register enables counting within each timer.

Timer Software Examples

The following C software examples demonstrate how a Timer is initialized, loaded with a Timer value, and enabled. The first example sets a Timer up to be 16 bits wide, while the second sets the Timer up to be 32 bits wide (see Table 1-5 on page 27 and Table 1-3 on page 22). Example 1 loads Timer 1 with a cycle time of 256 µs, while Example 2 loads Timer 2 with a cycle time of 320 ms. Each example assumes all unused timers are disabled.

timers_base refers to the PCI base address for the Timers, reference Table 1-1 on page 18.

Example 1:

```
/* set Timer 1 to 16 bits wide mode */
/* This command writes the hex value 00 to the TWSS register, see Table 2-1 and */
/* Table 2-2, setting bit 1 in the TWSS register to zero                          */

fw_byte( timers_base + 0x30, 0x00 );

/* Load Timer 1 with hex value 0x00FE */
/* The following commands load Timer 1 with a hex value which will generate a cycle */
/* time of 256µs. Since Timer cycle time is equal to (X+2)µs, where X is the loaded */
/* counter value, the counter is loaded with a value of (256 - 2) = 254. 254 is 0xFE */
/* in hexadecimal. Notice that only the Lower Counter within the Timer is loaded */
/* since the Timer is in 16 bit mode */

fw_byte( timers_base + 0x1C, 0x7A ); // Write to Timer 1 Mode Register, see Table 2-3
fw_byte( timers_base + 0x14, 0xFE ); // Write 0xFE (lsb) to Timer 1 Lower Counter
fw_byte( timers_base + 0x14, 0x00 ); // Write 0x00 (msb) to Timer 1 Lower Counter

/* enable Timer 1 and interrupt */
/* The following command writes a value of 0x02 (hex) to the TEI register (see */
/* Table 2-1 and Table 2-6). This byte enables interrupts from Timer 1 and */
/* actually turns on Timer 1. */

fw_byte( timers_base + 0x34, 0x02 );
```

Example 2:

```
/* set Timer 2 to 32 bits wide */
/* This writes the 0x04 (hex) to the TWSS register, see Table 2-1 and Table 2-6, */
/* setting bit 2 in the TWSS register to one                                      */

fw_byte( timers_base + 0x30, 0x04 );

/* Load Timer 2 with hex value 0x4E1FE */
/* The following commands load Timer 2 with a hex value which will generate a cycl*/
/* time of 320ms. Since Timer cycle time is equal to (X+2)µs, where X is the loaded */
/* counter value, the counter is loaded with a value of (320000 - 2) = 319999. 319999 */
/* is 0x4E1FE in hexadecimal. Notice that the Scale Counter, the Lower Counter and */
/* Upper Counter within the Timer is loaded since the Timer is in 32 bit mode. */
/* The Scale Counter must be loaded with 0x0000 when the Timer is in 32 bit mode */

fw_byte( timers_base + 0x2C, 0x36 ); // Write to Timer 2 Mode Register, see Table 2-3
fw_byte( timers_base + 0x20, 0x00 ); // Write 0x00 (lsb) to Timer 2 Scale Counter
fw_byte( timers_base + 0x20, 0x00 ); // Write 0x00 (msb) to Timer 2 Scale Counter
fw_byte( timers_base + 0x2C, 0x7A ); // Write to Timer 2 Mode Register, see Table 2-3
fw_byte( timers_base + 0x24, 0xFE ); // Write 0xFE (lsb) to Timer 2 Lower Counter
fw_byte( timers_base + 0x24, 0xE1 ); // Write 0xE1 (msb) to Timer 2 Lower Counter
fw_byte( timers_base + 0x2C, 0xBA ); // Write to Timer 2 Mode Register, see Table 2-3
fw_byte( timers_base + 0x28, 0x04 ); // Write 0x04 (lsb) to Timer 2 Upper Counter
fw_byte( timers_base + 0x28, 0x00 ); // Write 0x00 (msb) to Timer 2 Upper Counter

/* enable Timer 2 and interrupt */
/* The following command writes a value of 0x04 (hex) to the TEI register (see */
/* Table 2-1 and Table 2-6). This byte enables interrupts from Timer 2 and */
/* actually turns on Timer 2. */

fw_byte( timers_base + 0x34, 0x04 );
```

CHAPTER 3

Configuration and Installation

Contents

Shipping Configuration 39
Physical Installation of the VMIPMC-7441 41
Flash Configuration 42
Nonvolatile Static Random Access Memory (SRAM) 46

Introduction

The VMIPMC-7441 is designed to be used within any VMIC CPU system that allows the installation of a PCI Mezzanine Card (PMC) interface based on the P1386.1 Specification. In this chapter, the shipping configuration is described, followed by step-by-step instructions for the physical installation of the VMIPMC-7441. The Flash configuration involving the jumper settings and disk formatting follows; and, the chapter concludes with configuration notes involving the battery-backed SRAM.

Shipping Configuration

The VMIPMC-7441 is shipped with the configuration shown in Table 3-1. Verify that this is the actual configuration of the product prior to performing any Installation and / or Formatting procedure.

Table 3-1 VMIPMC-7441 Shipping Configuration

Header	Name	State
E 1	Note 1	Removed
E 3 1-2	EXPBOOT	Removed
E 3 3-4	FWECTRL	Installed
E 3 5-6	FBOOT	Installed
E 5	BATTERY	Removed

Note 1: Do not install. Reserved for factory use.

The Battery jumper, E5, must be in the Installed position for the Nonvolatile SRAM to work properly.

Physical Installation of the VMIPMC-7441

1. Four host mounting screws are attached to the VMIPMC-7441 that must be removed before the board can be installed into the CPU board. Two of the screws are installed in the standoffs, located closest to the PMC connectors. The other two are screwed into the Bezel. Remove the four screws (do not remove the screws that hold the Bezel and the standoffs to the PCB board).
2. Verify that the Battery jumper is in the Installed position. Nonvolatile SRAM does not work properly with the Battery jumper uninstalled.
3. Turn power off to the CPU board.
4. Place the CPU board in a position so that the PMC Interface is easily accessible.
5. With the component side of the VMIPMC-7441 board facing the CPU board and oriented such that the Front Panel Bezel of the PMC board faces the Front Panel of the CPU board, slip the Front Panel Bezel of the PMC board into the CPU PMC Front Panel port.
6. Taking care that the PMC alignment pin on the CPU board aligns with the alignment hole located near the PMC connectors in the VMIPMC-7441, connect the PMC board to the PMC connectors on the CPU board and press until the unit is snug.
7. Optional: Install the screws that were removed from the board in step 1 through the host CPU board into the VMIPMC-7441 standoffs and Bezel. Tighten until secure (do not overtighten). This step secures the VMIPMC-7441 to the host CPU.
8. Re-install the CPU board / VMIPMC-7441 assembly into the VMEbus chassis.

Figure 3-1 Jumper Locations on the VMIPMC-7441

Flash Configuration

There are three jumpers that are used in conjunction with the Flash OS Boot feature:

- Expansion ROM BIOS Boot (designated as EXPBOOT on the board, see Figure 3-1)
- Flash Boot (designated as FBOOT on the board)
- Flash Write Enable Control (designated as FWECTRL on the board)

The Expansion ROM Boot (EXPBOOT) jumper and the Flash Boot (FBOOT) jumper are used directly in conjunction with the Flash Boot process. The Expansion ROM Boot jumper is used to allow / disallow the execution of the Expansion ROM BIOS upon boot-up (see Table 3-2). When the jumper is installed, the Expansion ROM BIOS ***does not*** execute. When the jumper is removed, the Expansion ROM BIOS executes. The Expansion ROM BIOS, during the BIOS boot-up initialization phase, configures the host CPU based on the state of the Flash Boot (FBOOT) jumper.

The Flash Boot (FBOOT) jumper is examined when the Expansion ROM BIOS is executed. If the FBOOT jumper is installed, the Expansion ROM BIOS attempts to configure the Flash memory as DOS Drive B: (Drive B: is enabled in the host CPU BIOS). If the FBOOT jumper is removed, the Expansion ROM BIOS configures the Flash memory as DOS Drive A: and attempts to boot from the Flash memory.

Table 3-2 shows the various states of the EXPBOOT and FBOOT jumpers and the subsequent results on the boot-up process.

Table 3-2 EXPBOOT/FBOOT State Description

EXPBOOT	FBOOT	Description
Removed	Removed	Host boots to Flash-resident operating system. Expansion ROM BIOS executes and configures the Flash memory to be Drive A:. The system then attempts to boot from the Flash memory (see Note below).
Removed	Installed	Host does not boot to Flash resident operating system. Expansion ROM BIOS executes and configures the Flash memory to be Drive B: (If Drive B: is enabled in the host CPU BIOS).
Installed	NA	Host does not boot to Flash resident operating system. Expansion ROM BIOS does not execute. No system configuration is performed.

The Expansion ROM BIOS loads the first 512 bytes from the Flash memory and jumps to it for execution. The Expansion ROM BIOS does not examine the Flash memory to verify that the disk has a valid boot sector. Therefore, the user must ensure that a valid boot sector is installed on the Flash memory before trying to boot from it. The user must run the supplied Flash utilities to install the boot image in the Flash memory (see the *Flash Programming Utilities* section on page 29).

The Flash Write Enable Control jumper (designated as FWECTRL on the board) is used to protect the lower 1 Mbyte of Flash memory from accidental programming. If the FWECTRL jumper is installed, the lower 1 Mbyte of the Flash memory can be programmed. If the FWECTRL jumper is removed, the lower 1 Mbyte of the Flash memory cannot be programmed.

The states of all three of the jumpers can be examined via software by reading the Timer register named Timer Width / System State (TWSS). Bits 7 to 5 show the state of EXPBOOT, FBOOT, and FWECTRL, respectively. This register is located at offset 0x30 from the Timer section base address (see the *Timers* section on page 20 and Table 2-2 on page 32).

Table 3-3 Flash OS Boot Feature Jumper Options

Name	Jumper	Description
EXPBOOT	E 3 1 - 2	The Expansion ROM Boot jumper is used to allow / disallow the execution of the Expansion ROM BIOS upon CPU boot-up. When the jumper is installed, the Expansion ROM BIOS will not execute. When the jumper is removed, the Expansion ROM BIOS will execute.
FBOOT	E 3 5 - 6	The Flash Boot jumper is examined by the Expansion ROM BIOS to determine whether the Flash memory is configured to be write-protected Drive A: or Drive B:. If the FBOOT jumper is installed, the Flash memory will be configured to be Drive B:. If the FBOOT jumper is removed, the Flash memory will be configured to be Drive A:.
FWECTRL	E 3 3 - 4	The Flash Write Enable Control jumper is used to protect the lower 1 Mbyte of Flash memory from accidental programming. If the FWECTRL jumper is installed, the lower 1 Mbyte of the Flash memory can be programmed. If the FWECTRL jumper is removed, the lower 1 Mbyte of the Flash memory cannot be programmed.

Formatting Flash Disk

The VMIPMC-7441 is shipped with the Flash memory unformatted. The following procedure details how to format the Flash memory as a Flash disk. The procedure requires MS-DOS version 5.0 or greater to run and assumes that the jumpers are in their shipped state (reference Figure 3-1 on page 40). Furthermore, the board should be installed according to the procedure in the *Physical Installation of the VMIPMC-7441* section on page 41. This procedure makes use of a utility found on the provided 3.5-inch floppy disk labeled *Flash Utilities and Timer Example Code*.

The VMIPMC-7441 board is configured such that the Flash memory is hooked as Drive B: regardless of whether Flash memory is formatted or unformatted.

To format the Flash memory as a Flash disk, do the following:

1. Turn on the computer and verify that the BIOS Tables enable Drive B: as a 1.44 Mbyte floppy.
2. Exit out of the BIOS Tables and reboot the computer.
3. Verify that the provided Flash Utilities disk labeled *Flash Utilities and Timer Example Code* is secure in Drive A:.
4. Change drives to Drive A: by typing **A:** and pressing the <Enter> key.

Refer to the `Readme.txt` file located on the Utilities Disk provided with the VMIPMC-7441 for a description of the menu selections available from the *pgmflash.exe* utility.

5. Run the Flash programming utility by entering **pgmflash** and pressing the Enter key.
6. Choose option A - PROGRAM FLASH DISK FROM BOOT DISK and follow the instructions.
7. Once complete with instructions in step 6, choose option X - EXIT TO RETURN TO DOS.
8. The Flash memory is now a Flash Disk and can be accessed as Drive B: by entering **B:** and pressing Enter.
9. To boot from the Flash Disk as Drive A:
   - Turn the system power off
   - Remove the VMIPMC-7441 from the processor board
   - Remove FBOOT jumper
   - Reinstall the VMIPMC-7441 board
   - Reboot the computer

The Flash Disk uses only the lower 1.44 Mbyte of the available Flash memory.

If the installed OS replaces the BIOS, the OS must have a driver to access the Flash Disk.

Default Configuration

When the Flash device is shipped, the default configuration designates the Flash as Drive B:. However, the Flash disk is not formatted and must be formatted before accessing it. This is done via the program Flash utilities provided. Once the Flash disk is formatted, the user can set up the Flash as Drive A:.

When Flash is set up as Drive A:, it attempts to boot from Flash memory. This cannot be overridden with the host CPU BIOS settings. The VMIPMC-7441 Expansion ROM BIOS does not check the Flash boot sector before loading it and jumping to it.

It is the user's responsibility to install a valid boot image to Flash before trying to boot from Flash. Additionally, if the OS that is booting replaces the host CPU BIOS, then the Flash is no longer accessible as a drive without a OS-specific driver.

Nonvolatile Static Random Access Memory (SRAM)

The VMIPMC-7441 has 128 Kbyte of battery-backed SRAM. The SRAM allows 32-, 16-, and 8-bit wide read or write memory accesses. The PCI memory base address for the Nonvolatile SRAM is found in the VMIPMC-7441 PCI Configuration Register space at offset 0x1C.

Data Retention Power

A 3.0 V 20 mm x 3.2 mm Lithium coin cell battery provides for power-off data retention allowing the SRAM contents to be retained for more than two years. This battery is a commonly used coin cell which can be easily removed and replaced. The battery is held on the VMIPMC-7441 board by the battery retainer B1 located next to the battery jumper E5, see Figure 3-1 on page 42. The battery jumper, E5, is provided to connect / disconnect the battery power to the SRAM.

Note

The battery jumper is not installed when the VMIPMC-7441 is shipped. Therefore, it must be installed to enable SRAM operation.

The battery and battery jumper, E5, (see Figure 3-1 on page 42) must be installed for SRAM to operate.

Table 3-4 SRAM Battery Jumper Option

Name	Jumper Designator	Description
Battery	E5	Disconnects / connects the battery power to the SRAM. With the jumper removed, the battery power is disconnected from the SRAM and the SRAM feature is disabled. With the jumper installed, the battery power is connected to the SRAM.

Battery Replacement

The following instructions explain how to remove and replace a battery from the VMIPMC-7441 board.

Prior to replacing the battery, the SRAM data must be backed up; otherwise, the data will be lost.

Installing a new battery onto the VMIPMC-7441:

1. Back up SRAM data presently in memory.
2. Turn the system power off and remove the VMIPMC-7441 board from the host motherboard.
3. Apply pressure to the center eject tab of the battery retainer located on the side of the retainer closest to the battery jumper. This pressure can be administered via your finger or a pen point. The opposite side of the battery should pop out of the retainer.
4. Remove the old battery.
5. Slide a new 3.0 V 20 mm x 3.2 mm lithium coin cell battery under the center eject tab of the battery retainer. The positive side of the battery should be facing up.
6. Press the battery into the battery retainer with your finger until it snaps firmly in place.
7. Visually verify battery jumper E5 is installed. SRAM does not operate with the battery jumper not installed.
8. Reinstall VMIPMC-7441 board onto host motherboard.

CHAPTER
4

Maintenance

This section provides information relative to the care and maintenance of VMIC's products. If the products malfunction, verify the following:

1. Software
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 of adjacent boards are 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**.

Maintenance Prints

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

Index

A

Architectural Diagram 19

B

battery jumper, E5 46
battery replacement 46
BIOS 19
Block Diagram 12
Boot Jumper Options 44
Boot State Description 43

C

configuration register space 17
Counter access 30
Counter Value 33
CPU BIOS settings 45
cycle time 30
cycle time range 22

D

default configuration 45

E

E5 jumper 46
Expansion ROM BIOS 19, 32, 42, 43
Expansion ROM Read Enable Register 31
EXPBOOT 32, 42

F

FBOOT 19, 32, 42
Flash Boot 42
Flash memory 11, 19, 32, 45
Flash utilities 29, 43
Flash Write Enable Control 32, 42
Flash Write Enable Control jumper 43
FWECTRL 32, 42, 43

I

Intel 82C54 30, 33

L

Least Significant Byte 33
Lower Counter 20

M

Most Significant Byte 33

O

Offset 30h 32
Offset 34h 35
Offset 38h 36
Operating System 19, 43

P

PCI Configuration Register Space 19
PCI Configuration Space Registers 18
PCI Local Bus Specification 11
PCI Mezzanine Card Specification 11

R

Register
  Timer Enable/Interrupt (TEI) Register 20, 30, 31, 35
  Timer Interrupt Status (TIS) Register 20, 22, 30, 31, 36
  Timer Mode (TMR2) Register 31

Timer Mode (TMRx) Register 33
Timer Width Control/System State (TWSS) Register 32
Timer Width/System State (TWSS) Register 20, 30, 31, 32
register space 19
Return Material Authorization (RMA) number 49
roll-over (expire) 26

S

safety precautions 15
Safety Symbols 16
Scale Counter 20
SRAM 11, 46

T

Timer 11, 30
Address Map 30
Comparison Table 22
configuration 30
Control circuitry sets 22
Mode Register Values 33
Width Bit Field 32
Timer 2 width 32
Timer Block Diagram 21
timer control 20
Timer Example Code 29

U

Upper Counter 20