Single Board Computers Programmer's Reference Guide (Part 1 of 2)

VMESBCA1/PG1

Notice

While reasonable efforts have been made to assure the accuracy of this document, Motorola, Inc. assumes no liability resulting from any omissions in this document, or from the use of the information obtained therein. Motorola reserves the right to revise this document and to make changes from time to time in the content hereof without obligation of Motorola to notify any person of such revision or changes.

It is possible that this publication may contain reference to, or information about Motorola products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that Motorola intends to announce such Motorola products, programming, or services in your country.

Restricted Rights Legend

If the documentation contained herein is supplied, directly or indirectly, to the U.S. Government, the following notice shall apply unless otherwise agreed to in writing by Motorola, Inc.

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c)(1)(ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013.

Motorola, Inc.
Computer Group
2900 South Diablo Way
Tempe, Arizona 85282

Preface

This manual provides board level information and detailed ASIC chip information including register bit descriptions for the MVME166, MVME167, MVME176, MVME177, and MVME187 Single Board Computers. The information in this manual applies to the single board computers listed in the following table:

MVME166 Models	MVME167 Models	MVME176 Models	MVME177 Models	MVME187 Models
MVME166-011a	MVME167-001a	MVME176-001a	MVME177-001a	MVME187-001a
MVME166-012a	MVME167-002a	MVME176-002a	MVME177-002a	MVME187-002a
MVME166-013a	MVME167-003a	MVME176-003a	MVME177-003a	MVME187-003a
MVME166-014a	MVME167-004a	MVME176-004a	MVME177-004a	MVME187-004a
MVME166-015a	MVME167-031a	MVME176-005a	MVME177-005a	MVME187-023a
MVME166-016a	MVME167-032a	MVME176-006a	MVME177-006a	MVME187-024a
	MVME167-033a		MVME177-011a	MVME187-031a
	MVME167-034a		MVME177-012a	MVME187-032a
	MVME167-035a		MVME177-013a	MVME187-033a
	MVME167-036a		MVME177-014a	MVME187-034a
			MVME177-015a	MVME187-035a
			MVME177-016a	MVME187-036a

The letter “a” in the model number indicates the major revision level.

Notes

This document is bound in two parts. Part 1 (VMESBCA1/PG*x*) contains Chapters 1 through 4. Part 2 (VMESBCA2/PG*x*) contains Chapters 5 through 9.

This manual replaces the *MVME166/167/187 Single Board Computers Programmer’s Reference Guide*, MVME187PG/D3, and its supplement, MVME187PG/D3A1. They are obsolete.

This manual is intended for anyone who wants to program these boards in order to design OEM systems, supply additional capability to an existing compatible system, or work in a lab environment for experimental purposes.

A basic knowledge of computers and digital logic is assumed.

To use this manual, you should be familiar with the publications listed in *Related Documentation* below.

Related Documentation

The following publications are applicable to the Single Board Computers and may provide additional helpful information. If not shipped with this product, they may be purchased by contacting your local Motorola sales office. Non-Motorola documents may be obtained from the sources listed.

Document Title	Motorola Publication Number
MVME166 Single Board Computer User's Manual	MVME166/D
MVME167 Single Board Computer User's Manual	MVME167/D
MVME176 Single Board Computer Installation and Use Manual	VME176A/IH
MVME177 Single Board Computer Installation and Use Manual	VME177A/IH
MVME167Bug Debugging Package User's Manual	MVME167BUG/D
MVME177Bug Diagnostics User's Manual	V177DIAA/UM
Debugging Package for Motorola 68K CISC CPUs User's Manual (Parts 1 and 2)	68KBUG1/D and 68KBUG2/D
MVME187 RISC Single Board Computer User's Manual	MVME187/D
MVME187Bug Debugging Package User's Manual	MVME187BUG/D
Debugging Package for Motorola 88K RISC CPUs User's Manual	88KBUG1/D and 88KBUG2/D
Single Board Computers SCSI Software User's Manual	SBCSCSI/D
MVME712-06/07/09 I/O Distribution Board Set User's Manual	MVME712IO/D
MVME712-10 Transition Module User's Manual	MVME712-10/D
MVME712M Transition Module and P2 Adapter Board User's Manual	MVME712M/D
MVME712-12, MVME712-13, MVME712A, MVME712AM, and MVME712B Transition Modules and LCP2 Adapter Board User's Manual	MVME712A/D

Document Title	Motorola Publication Number
MC88100 RISC Microprocessor User's Manual	MC88100UM
MC88200 Cache/Memory Management Unit (CMMU) User's Manual	MC88200UM
M68040 Microprocessors User's Manual	M68040UM
M68060 Microprocessors User's Manual	M68060UM
M68000 Family Reference Manual	M68000FR

Note

Although not shown in the above list, each Motorola Computer Group manual publication number is suffixed with characters which represent the revision level of the document, such as "/*xx*2" (the second revision of a manual); a supplement bears the same number as a manual but has a suffix such as "/*xx*2A1" (the first supplement to the second edition of the manual).

The following publications are available from the sources indicated:

Versatile Backplane Bus: VMEbus, ANSI/IEEE Std 1014-1987, The Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017 (VMEbus Specification). (This is also *Microprocessor System Bus for 1 to 4 Byte Data*, IEC 821 BUS, Bureau Central de la Commission Electrotechnique Internationale; 3, rue de Varembé, Geneva, Switzerland.)

Z85230 Serial Communications Controller Data Sheet, order number DC-8293-02, Zilog Inc., 210 East Hacienda Drive, Campbell, CA 95008-6600.

IEEE Standard for Multiplexed High-Performance Bus Structure: VSB, ANSI/IEEE Std 1096-1988, The Institute of Electrical and Electronics Engineers, Inc., 345 East 47th Street, New York, NY 10017 (VSB Specification). (This is also *Parallel Sub-syste Bus of the IEC 821 VMEbus*, IEC 822 VSB, Bureau Central de la Commission Électrotechnique Internationale; 3, rue de Varembé, Geneva, Switzerland.)

ANSI Small Computer System Interface-2 (SCSI-2), Draft Document X3.131-198X, Revision 10c; Global Engineering Documents, 15 Inverness Way East, Englewood, CO 80112-5704.

CL-CD2400/2401 Four-Channel Multi-Protocol Communications Controller Data Sheet, order number 542400-003; Cirrus Logic, Inc., 3100 West Warren Ave., Fremont, CA 94538.

82596CA Local Area Network Coprocessor Data Sheet, order number 290218; and *82596 User's Manual*, order number 296853; Intel Corporation, Literature Sales, P.O. Box 58130, Santa Clara, CA 95052-8130.

DS1643 Nonvolatile Timekeeping RAM, Dallas Semiconductor Data Manual, 4401 South Beltwood Parkway, Dallas, Texas 75244-3292.

NCR 53C710 SCSI I/O Processor Data Manual, order number NCR53C710DM, and *NCR 53C710 SCSI I/O Processor Programmer's Guide*, order number NCR53C710PG; NCR Corporation, Microelectronics Products Division, 1635 Aeroplaza Dr., Colorado Springs, CO 80916.

MK48T08(B)/MK48T18(B) Timekeeper™ *and 8Kx8 Zeropower*™ *RAM* data sheet in *Static RAMs Databook*, order number DBSRAM71; SGS-THOMSON Microelectronics; North & South American Marketing Headquarters, 1000 East Bell Road, Phoenix, AZ 85022-2699.

i28F008 Flash Memory Data Sheet, order number 290435, *i28F020 Flash Memory Data Sheet*, order number 290245, *i28F008SA Software Drivers Application Note,* order number 292095, and *i28F008SA Automation and Algorithms Application Note*, order number 292099; Intel Literature Sales, P.O. Box 7641, Mt. Prospect, IL 60056-7641.

MC68230 Parallel Interface Timer (PI/T) Data Sheet, order number MC68230/D, Motorola Semiconductor Products, Inc., LDC, Broadway Bldg. BB100, P.O. Box 20924, Phoenix, AZ 85036-0924.

Manual Terminology

Throughout this manual, a convention is used which precedes data and address parameters by a character identifying the numeric format as follows:

$	dollar	specifies a hexadecimal character
%	percent	specifies a binary number
&	ampersand	specifies a decimal number

For example, "12" is the decimal number twelve, and "$12" is the decimal number eighteen.

Unless otherwise specified, all address references are in hexadecimal.

An asterisk (*) following the signal name for signals which are *level significant* denotes that the signal is *true* or valid when the signal is low.

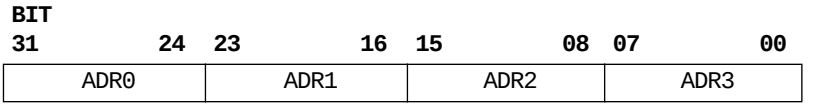
An asterisk (*) following the signal name for signals which are *edge significant* denotes that the actions initiated by that signal occur on high to low transition.

In this manual, *assertion* and *negation* are used to specify forcing a signal to a particular state. In particular, assertion and assert refer to a signal that is active or true; negation and negate indicate a signal that is inactive or false. These terms are used independently of the voltage level (high or low) that they represent.

Data and address sizes are defined as follows:

- A *byte* is eight bits, numbered 0 through 7, with bit 0 being the least significant.
- A *two-byte* is 16 bits, numbered 0 through 15, with bit 0 being the least significant. For the MVME166, MVME167, MVME176, MVME177, and other CISC boards, this is called a *word*. For the MVME187 and other RISC boards, this is called a *half-word*.
- A *four-byte* is 32 bits, numbered 0 through 31, with bit 0 being the least significant. For the MVME166, MVME167, MVME176, MVME177, and other CISC boards, this is called a *longword*. For the MVME187 and other RISC boards, this is called a *word*.

Throughout this manual, it is assumed that the MPU on the MVME187 always programs the CMMUs with *big-endian* byte ordering, as shown below. Any attempt to use small-endian byte ordering immediately renders the MVME187Bug debugger unusable.



The terms *control bit* and *status bit* are used extensively in this document:

control bit	A bit in a register that can be set and cleared under software control.
true	Indicates that a bit is in the state that enables the function it controls
false	Indicates that the bit is in the state that disables the function it controls
status bit	A bit in a register that reflects a specific condition. The status bit can be read by software to determine operational or exception conditions.

In all tables, the terms 0 and 1 are used to describe the actual value that should be written to the bit, or the value that it yields when read.

The computer programs stored in the Read Only Memory of this device contain material copyrighted by Motorola Inc., first published 1990, and may be used only under a license such as the License for Computer Programs (Article 14) contained in Motorola's Terms and Conditions of Sale, Rev. 1/79.

This equipment generates, uses, and can radiate electromagnetic energy. It may cause or be susceptible to electromagnetic interference (EMI) if not installed and used in a cabinet with adequate EMI protection.

Printed in the United States of America

June 1996

Contents

Chapter 1 Programming Issues

Chapter 2 Hardware Configuration

Chapter 3 Memory Maps

Chapter 4 VMEchip2

Chapter 5 VSBchip2

Chapter 6 PCCchip2

Chapter 7 MEMC040

Chapter 8 MCECC

Figures

Tables

Programming Issues 1

Introduction

The Single Board Computers are complex boards that interface to the VMEbus, VSB, and SCSI bus. These multiple bus interfaces raise the issue of cache coherency and support of indivisible cycles. There are also many sources of bus error. This chapter discusses these topics in addition to the programming interface to each device on the SBC, interrupt handling, and the use of bus timers.

Chapter 2 describes the configurable features of the Single Board Computers.

Chapter 3 describes memory maps for the local devices.

Specific programming information, including information on the programmable registers, is provided for the Application-Specific Integrated Circuit (ASIC) devices in the chapter dedicated to the ASIC, as follows:

Chapter 4	VMEchip2
Chapter 5	VSBchip2
Chapter 6	PCCchip2
Chapter 7	MEMC040 memory controller
Chapter 8	MCECC memory controller

Chapter 9 contains diagrams showing connections between the Single Board Computers' printer and serial ports and the MVME712*x* transition boards.

Programming Interfaces

The next sections describe the programming interface to features of the Single Board Computers. Unless the section calls out a specific Single Board Computer, the discussion applies to all Single Board Computers described in this manual.

MC68040 MPU

The MC68040 processor is used on the MVME166/167. The MC68040 has on-chip instruction and data caches and a floating point processor. Refer to the M68040 user's manual for more information.

MC68060 MPU

The MC68060 processor is used on the MVME176/177. The MC68060 has on-chip instruction and data caches and a floating point processor. Refer to the M68060 user's manual for more information.

M88000 MPU

The MVME187 is based on the M88000 family and uses one MC88100 MPU and two MC88200 or MC88204 CMMUs. One CMMU is used for the data cache and one is used for the instruction cache. Refer to the MC88100 and MC88200 user's manuals for more information.

Data Bus Structure

The Local Bus for all Single Board Computers described in this manual is a 32-bit synchronous bus, which is based on an MC68040-compatible bus and which supports burst transfers. Throughout this manual this bus is referred to as the Local Bus. The various Local Bus master and slave devices use the Local Bus to communicate. The Local Bus is arbitrated by priority type arbiter. The priority of the Local Bus masters from highest to lowest is:

Highest priority	82596CA LAN
	CD2401 serial (through the PCCchip2)
	53C710 SCSI
	VSB (MVME166/176 only)
	VMEbus
Lowest priority	MPU

In the general case, any master can access any slave; however, not all combinations pass the common sense test. Refer to the specific section of this manual and to the user's guide for each device to determine its port size, data bus connection, and any restrictions that apply when accessing the device.

EPROMs on the MVME167/176/177/187

All models except the MVME166 include 44-pin PLCC/CLCC EPROM sockets for EPROMs, as follows:

Model	Sockets	Banks
MVME167, 187	4	2
MVME176/177	2	1

The EPROM sockets support 64K x 16, 128K x 16, and 256K x 16 devices. Many devices may be used, including 27C102JK, 27C202JK, M27C4002, or 27C4096.

The EPROMs are organized as 32-bit wide banks that support 8-, 16-, and 32-bit read accesses. (The MVME166 has Flash memories; the MVME176/177 has Flash memories in addition to the EPROM.)

MVME167 and MVME187

The EPROMs are mapped to Local Bus address 0 following a Local Bus reset. This allows the MC68040 to access the stack pointer and execution address following a reset. It also allows the MC88100 to start executing code at address 0 following a reset. The EPROMs are controlled by the VMEchip2. The map decoder, access time, and when they appear at address 0, is programmable. Refer to the chapter *VMEchip2* for more detail.

MVME176/177

The EPROMs on the MVME176/177 share 2MB of memory with the first 2MB of Flash memory. The EPROM can co-exist with 2MB of Flash or you can program all 4MB as Flash memory. Only 1MB of

EPROM is writable. Its contents are duplicated in the second 1MB section, which can be read but not written to. The Flash and EPROM configuration is controlled by a jumper as described in Chapter 2, *Hardware Configuration*, and GPIO bit 2, as described in the chapter *VMEchip2*.

The EPROMs are mapped to Local Bus address 0 following a Local Bus reset.This allows the MC68060 to access the reset vector and execution address following a reset.

Flash Memory on the MVME176/177

The MVME176/177 includes four 28F008SA Flash memory devices. The 32-bit wide Flash can support 8-, 16-, and 32-bit accesses.The Flash can be used for the onboard debugger firmware, which can be downloaded from I/O resources such as Ethernet, SCSI, serial port, or VMEbus. Flash write-protection is programmable by setting the control bit (GPIO bit 1) in the VMEchip2 GPIO register after downloading.

When the Flash memory is used with EPROM, only the top or bottom 2MB of Flash memory is visible at any one time. For accessing the shadowed area of Flash, the 176/177Bug provides the **SFLASH** command.

The MVME176/177 is shipped with the top 2MB of Flash memory and EPROM mapped as illustrated by Map 2 in Figure 1-1.

The 176/177Bug is shipped in EPROM. To map all 4MB of Flash and retain access to the 176/177Bug, perform the following steps:

1. Map Flash and EPROM as shown in Map 3 in Figure 1-1.
2. Copy the 176/177Bug into the bottom 2MB of Flash memory.
3. Remap Flash memory as shown in Map 1 in Figure 1-1.

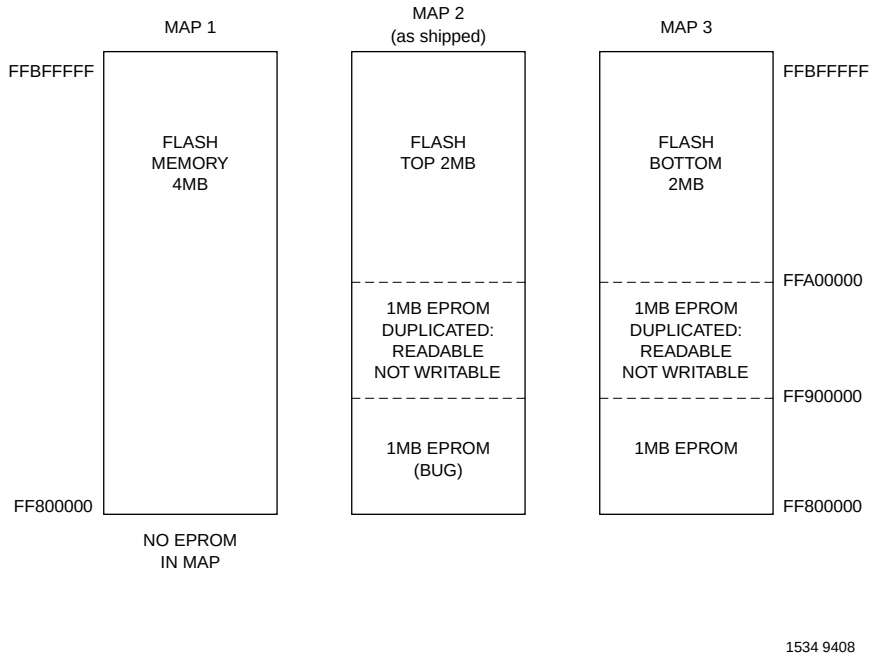


Figure 1-1. MVME176/177 Flash and EPROM Memory Mapping Schemes

Flash Memory and Download EPROM on the MVME166

The MVME166 includes four 28F020 Flash memory devices and a download EPROM. These parts replace the four EPROM sockets used on the MVME167/187. The Flash parts are programmable on the MVME166 board and the programming code is provided in the download EPROM. The Flash devices provide 1MB of ROM. The download EPROM provides 128KB of ROM. The download EPROM is mapped to Local Bus address 0 following a Local Bus reset. This allows the MC68040 to access the stack pointer and execution address following a reset. The download EPROM appears at 0 until the DR0 bit is cleared in the PCCchip2 chip. The Flash devices are controlled by the VMEchip2 and the download EPROM is controlled by the PCCchip2. The PC0 bit in the MC68230 PI/T chip must be low to enable writes to Flash.

The EPROM contains the BootBug product (166BBug). Because Flash memory can be electronically erased, the EPROM firmware is a subset of the regular debugger product. It contains enough functionality from the debugger to permit downloading of object code (via VMEbus, serial port, SCSI bus, or the network) and reprogramming of the Flash memory.

A jumper on the MVME166 (J3, pins 7 and 8) controls the operation of the BootBug:

- If the jumper is in place, the BootBug (which always executes at power-up and reset) passes execution to the full debugger contained in Flash memory.
- If the jumper is removed, execution continues (with diminished functionality) in the BootBug.

Before you perform any SCSI, VMEbus, or Ethernet I/O with the MVME166, it may be necessary to define some parameters (e.g., SCSI ID, Ethernet address, VMEbus mapping). For details on configuring the MVME166, refer to the **setup** command description in the *MVME167Bug Debugging Package User's Manual*.

SRAM

The Single Board Computers include 128KB of 32-bit wide static RAM that supports 8-, 16-, and 32-bit wide accesses. The SRAM allows the debugger to operate and limited diagnostics to be executed without the DRAM mezzanine. The SRAM is controlled by the VMEchip2, and the access time is programmable. Refer to the chapter *VMEchip2* for more detail. The boards are populated with 100 ns SRAMs.

The Single Board Computers provide for battery backup for the SRAM. (The backup circuitry is standard on the MVME166/176/177 and optionally available on other models.) The battery backup function is provided by a Dallas DS1210S.

The MVME166 and MVME187 implementations support primary and secondary back up power sources. The boards include jumpers that allow the power inputs of the DS1210S to be connected to either the VMEbus +5 V STDBY pin or to one cell of the onboard battery. For example, the primary backup source may be from the VMEbus +5 V STDBY pin and the secondary source may be the onboard battery.

The MVME167 and MVME176/177 implement a single backup source. You can select one of the following: +5V standby, the onboard battery, or no backup power.

The jumpers for each board are described in the chapter *Hardware Configuration*.

Onboard DRAM

The DRAM map decoder can be programmed to accommodate different base address(es) and sizes of mezzanine boards. The onboard DRAM is disabled by a Local Bus reset and must be programmed before the DRAM can be accessed.

Most DRAM devices require some number of access cycles before the DRAMs are fully operational. Normally this requirement is met by the onboard refresh circuitry and normal DRAM initialization. However, software should insure a minimum of 10 initialization cycles are performed to each bank of RAM.

Detailed programming information is available in the chapters on the memory options.

Battery Backed Up RAM and Clock

The Dallas DS1643/SGS-THOMSON MK48T18 is an 8-bit device; however, the interface provided by the PCCchip2 supports 8-, 16-, and 32-bit accesses to the DS1643/MK48T18. No interrupts are generated by the clock. Refer to the chapter *PCCchip2* and to the DS1643/MK48T18 data sheet for detailed programming and battery life information.

VMEbus Interface

The Local Bus to VMEbus interface, the VMEbus to Local Bus interface, and the local-VMEbus DMA controller functions are provided by the VMEchip2. The VMEchip2 can also provide the VMEbus system controller functions. Refer to the chapter *VMEchip2*, for detailed programming information.

VME Subsystem Bus (VSB) Interface

The Local Bus to VSB interface and the VSB to Local Bus interface are provided by the VSBchip2, only on the MVME166/176 board. The VSB uses the P2 connector of the MVME166/176. Refer to the chapter *VSBchip2* for detailed programming information for the VSBchip2.

I/O Interfaces

The Single Board Computers provides onboard I/O for many system applications. The I/O functions include serial ports, parallel (printer) port, Ethernet transceiver interface, and SCSI mass storage interface.

Serial Port Interface

The CD2401 serial controller chip (SCC) is used to implement the serial ports. On all Single Board Computers, the interface provided by the PCCchip2 allows the 16-bit CD2401 to appear at contiguous addresses; however, accesses to the CD2401 must be 8 or 16 bits; 32-bit accesses are not permitted. Refer to the CD2401 data sheet and to the chapter *PCCchip2* for detailed programming information.

On all Single Board Computers, the CD2401 supports DMA operations to local memory. Because the CD2401 does not support a retry operation necessary to break VMEbus or VSB dual port lockup conditions, the CD2401 DMA controllers should not be programmed to access the VMEbus or VSB. The hardware does not restrict the CD2401 to onboard DRAM.

The next subsections describe the two implementations of the serial ports.

Serial Port Interface (MVME167/177/187)

The four serial ports support the standard baud rates (110 to 38.4K baud). The serial ports are different functionally because of the limited number of pins on the P2 I/O connector:

- Serial port 1 is a minimum function asynchronous port. It uses RXD, CTS, TXD, and RTS.
- Serial ports 2 and 3 are full function asynchronous ports. They use RXD, CTS, DCD, TXD, RTS, and DTR.
- Serial port 4 is a full function asynchronous or synchronous port. It can operate at synchronous bit rates up to 64 k bits per second. It uses RXD, CTS, DCD, TXD, RTS, and DTR. It also interfaces to the synchronous clock signal lines. Refer to the chapter *Printer and Serial Port Connections* for drawings of the serial port interface connections.

All four of these serial ports use EIA-232-D drivers and receivers located on the main board, and all the signal lines are routed to the I/O connector. The configuration headers are located on the main board and the MVME712 series transition board. An external I/O transition board such as the MVME712*x* should be used to provide configuration headers and industry-standard connectors.

Note The board hardware of these Single Board Computers ties the DTR signal from the CD2401 to the pin labeled RTS at connector P2. Likewise, RTS from the CD2401 is tied to DTR on P2. Therefore, when programming the CD2401, assert DTR when you want RTS, and RTS when you want DTR.

Serial Port Interface (MVME166/176 Only)

The four serial ports on the MVME166/176 are functionally the same. All serial ports are full function asynchronous or synchronous ports. They can operate at synchronous bit rates up to 64 k bits per second. They use RXD, CTS, DCD, TXD, RTS, DTR, and DSR. They also interface to the synchronous clock signal lines.

Additional control signals are provided for each serial port by the MC68230 Parallel Interface/Timer. These include local loopback control, self test control, and ring indicator. The ring indicator signal can be programmed to generate a Local Bus interrupt. Refer to the MC68230 section for additional information. Refer to chapter 9 for drawings of the serial port interface connections. Note that the usable functionality of the serial ports depends on the transition module used.

All four serial ports use a TTL interface to the transition board. This allows the interface specific drivers to be located on the transition board. This allows more flexibility in configuring the serial ports for different interfaces like EIA-232-D or V.35. An external I/O transition module such as the MVME712-10 should be used to provide configuration headers, interface drivers, and industry-standard connectors.

MC68230 Parallel Interface/Timer (MVME166/176 Only)

The MVME166/176 provides an MC68230 parallel interface/timer (PI/T) chip. When the MVME166/176 is used with the MVME712-10 transition module or the MVME712-06/07/09 I/O distribution board set, the MC68230 is used to provide additional control lines for the serial ports. These include local loopback, self test, and ring indicator. The ring indicator signals can be programmed to generate Local Bus interrupts. Refer to the chapter *Printer and Serial Port Connections* in this manual, and to the MVME712-10 transition module manual, for more information.

The base address of the MC68230 is $FFF45E00, and because it is an 8-bit device it appears only at odd addresses. Space for the MC68230 was created by dividing the area occupied by redundant

copies of the CD2401 registers into eight segments. The CD2401 is still addressed at $FFF45000 to $FFF451FF. Addresses $FFF45200 to $FFF45BFF are reserved, and if accessed on an MVME166/176 cause a Local Bus time-out error, if the Local Bus timer is enabled. The address range from $FFF45C00 to $FFF45DFF always returns a Local Bus time-out error if the Local Bus timer is enabled. The CD2401 appears redundantly from $FFF45200 to $FFF45FFF on the MVME167/177/187.

The presence of the MC68230 can be determined by reading address $FFF45C00. If a time-out error occurs, then the board is an MVME166/176 and the MC68230 is present. If a time-out does not occur, then the board is an MVME167/177/187 and the MC68230 is not present. The Local Bus time-out timer in the VMEchip2 must be enabled for this test.

The MC68230 may be used for general purpose I/O when the MVME166/176 is not used with the MVME712 family of transition modules. Because the outputs are unbuffered and unprotected, these signals should be used with caution. The port A signal lines PA<7..0> are connected to the front panel connector J9. The port A signal lines can be programmed as inputs or outputs. The port B signal lines PB<3..0> are connected to the port H signal lines H<4..1> and the front panel connector J9. This allows these four lines to be inputs or outputs or receive interrupts. The port B signal line PB<7> is also connected to the front panel connector J9. When used with the MVME712 family of transition modules, the PB<7> signal line is used to read the configuration of the serial ports. Timer interrupts from the MC68230 are not supported on the MVME166/176. The MC68230 is connected to a 10 MHz clock. The PC0 bit in the MC68230 PI/T chip must be low to enable writes to Flash memory.

Parallel (Printer) Interface

All models have a parallel (printer) interface. The interface is provided by the PCCchip2. Refer to the chapter *PCCchip2* for detailed programming information. Refer to the chapter *Printer and Serial Port Connections* for drawings of the printer port interface connections.

Ethernet Interface

The 82596CA is used to implement the Ethernet transceiver interface. The 82596CA accesses local RAM using DMA operations to perform its normal functions. Because the 82596CA has small internal buffers and the VMEbus has an undefined latency period, buffer overrun may occur if the DMA is programmed to access the VMEbus or VSB. Therefore, the 82596CA should not be programmed to access the VMEbus or VSB.

Each Single Board Computer is assigned an Ethernet Station Address. The address is $08003E2*xxxxx* where *xxxxx* is the unique 5-nibble number assigned to a board.

Each board has an Ethernet Station Address displayed on a label attached to the VMEbus P2 connector. In addition, the six bytes including the Ethernet address are stored in the configuration area of the BBRAM. That is, 08003E2*xxxxx* is stored in the BBRAM. At an address of $FFFC1F2C, the upper four bytes (08003E2*x*) can be read. At an address of $FFFC1F30, the lower two bytes (*xxxx*) can be read. Refer to the BBRAM, TOD Clock memory map description in the chapter *Memory Maps*. The MVME166 BootBug has the capability to retrieve or set the Ethernet address, as do the other Single Board Computer debuggers.

If the data in the BBRAM is lost, use the number on the VMEbus P2 connector label to restore it.

The Ethernet transceiver interface is located on the main board of the Single Board Computers, and the industry standard connector is located on the MVME712*x* transition board.

Support functions for the 82596CA are provided by the PCCchip2. Refer to the 82596CA user's guide and to the chapter *PCCchip2* for detailed programming information.

SCSI Interface

The Single Board Computers provide for mass storage subsystems through the industry-standard SCSI bus. These subsystems may include hard and floppy disk drives, streaming tape drives, and other mass storage devices. The SCSI interface is implemented using the NCR 53C710 SCSI I/O controller.

Support functions for the 53C710 are provided by the PCCchip2. Refer to the 53C710 user's guide and to the chapter *PCCchip2* for detailed programming information.

SCSI Termination

The system configurer must ensure that the SCSI bus is properly terminated at both ends. This is done on each system as follows:

MVME167 MVME177 MVME187	Sockets are provided for the terminators on the P2 transition board. If the SCSI bus ends at the P2 transition board, then termination resistors must be installed on the P2 transition board. +5V power to the SCSI bus TERM power line and termination resistors is provided through a fuse located on the P2 transition board (for the MVME167 and MVME187), but is provided through a fuse on the MVME712 transition board and a diode on the P2 transition board (for the MVME177).
MVME166 MVME176	SCSI bus termination is provided on the main board. The terminators are enabled/disabled by a jumper. If the SCSI bus ends at the MVME166/176, the SCSI terminators must be enabled by installing the jumper.

Local Resources

The Single Board Computers include many resources for the local processor. These include tick timers, software programmable hardware interrupts, watchdog timer, and Local Bus time-out.

Programmable Tick Timers

Four 32-bit programmable tick timers with 1 µsec resolution are provided, two in the VMEchip2 and two in the PCCchip2. The tick timers can be programmed to generate periodic interrupts to the processor. Refer to the chapters *VMEchip2* and *PCCchip2*, respectively, for detailed programming information.

Watchdog Timer

A watchdog timer function is provided in the VMEchip2. When the watchdog timer is enabled, it must be reset by software within the programmed time or it times out. The watchdog timer can be programmed to generate a SYSRESET signal, local reset signal, or board fail signal if it times out. Refer to the chapter *VMEchip2* for detailed programming information.

Software-Programmable Hardware Interrupts

Eight software-programmable hardware interrupts are provided by the VMEchip2. These interrupts allow software to create a hardware interrupt. Refer to the chapter *VMEchip2* for detailed programming information.

Local Bus Time-out

The Single Board Computers provide a time-out function for the Local Bus. When the timer is enabled and a Local Bus access times out, a Transfer Error Acknowledge (TEA) signal is sent to the Local Bus master. The time-out value is selectable by software for 8 µsec, 64 µsec, 256 µsec, or infinite. The Local Bus timer does not operate during VMEbus or VSB bound cycles.

VMEbus bound cycles are timed by the VMEbus access timer and the VMEbus global timer. Refer to the chapter *VMEchip2* for detailed programming information.

VSB bound cycles are timed by the VSB access timer, the VSB transfer timer, and if its serial arbiter is enabled, by the VSB arbitration timer. Refer to the chapter *VSBchip2* for detailed programming information.

Interrupt Handling

Because the M68000-based systems use hardware-vectored interrupts while the MC88100 does not, interrupts are handled differently on the Single Board Computers based on those processors. The C040 bit in the PCCchip2 General Control Register (address $FFF42002) should be set when the board MPU is from the M68000 family. It should be cleared when the MPU is an MC88100. For more information, refer to "Interrupt Prioritizer" (MVME187) in the chapter *PCCchip2*.

Most interrupt sources are level and base vector programmable. Interrupt vectors from the PCCchip2, VSBchip2, and the VMEchip2 have two sections:

Base value	Can be set by the processor, usually the upper four bits
Lower bits	Set according to the particular interrupt source

There is an onboard daisy chain of interrupt sources, prioritized as follows:

Highest priority	Interrupts from the PCCchip2
Next	Interrupt sources from the VSBchip2
Lowest priority	Interrupt sources from the VMEchip2

The interrupt scheme varies according to the processor:

MC68040 mode	Seven-level prioritized, hardware-vectored interrupt scheme that is standard in the M68000 family
MC88100 mode	All interrupt sources are combined into one interrupt request to the MC88100 by the PCCchip2. When an interrupt occurs, the MC88100 reads the interrupt level from the PCCchip2 Interrupt Priority Level Register. The MC88100 then reads the corresponding IRQ Level register as shown in Table 3-2, *I/O Devices Memory Map*. This read causes a hardware IACK cycle to be performed and returns the associated vector as the value read.

The Local Bus distinguishes interrupt acknowledge cycles from other cycles by placing the binary value %11 on TT1-TT0. It also specifies the level that is being acknowledged using TM2-TM0. The interrupt handler selects which device within that level is being acknowledged.

Interrupt Programming Examples

This section demonstrates how to use interrupts on Single Board Computers.

Read this entire section before performing any of these procedures.

The first subsection gives an example of how to generate and handle a VMEchip2 Tick Timer 1 interrupt on M68000-based Single Board Computers. Specific values have been given for the register writes.

The second subsection talks about how interrupts are handled on the MVME187. There are substantial differences in the way interrupts are handled on the MVME187 and the M68000-based Single Board Computers.

M68000 VMEchip2 Tick Timer 1 Periodic Interrupt Example

A. Set up Tick Timer 1.

Step	Register	Address	Action and Reference
1.	Prescaler Control Register	$FFF4004C	If not already initialized by the debugger, initialize as follows: Prescaler Register = 256 - **Bclock** (MHz). This gives a 1 MHz clock to the tick timers. **Bclock** is the bus clock rate, such as 25 MHz. 256 - 25 = $E7.
2.	Tick Timer 1 Compare Register	$FFF40050	For periodic interrupts, set the Compare Register value = **Period** (µs). For example, if you want an interrupt every millisecond, set the register value to 1000 ($3E8). Refer to the Tick Timer 1 Compare Register description in the chapter *VMEchip2*.
3.	Tick Timer 1 Counter Register	$FFF40054	Write a 0 to clear.
4.	Tick Timer 1 Control Register	$FFF40060 (8 bits)	Write $07 to this register (set bits 0, 1, and 2). This enables the Tick Timer 1 counter to increment, resets the count to 0 on compare, and clears the overflow counter.

B. Set up the Local Bus interrupter:

Step	Register	Address	Action and Reference
5.	Vector Base Register	$FFF40088 (8 of 32 bits)	If not already initialized by the debugger, set interrupt base register 0 by writing to bits 28-31. Refer to the Vector Base Register description and to Table 4-4, *Local Bus Interrupter Summary*, in the chapter *VMEchip2*.
6.	Interrupt Level Register 1 (bits 0-7)	$FFF40078 (8 of 32 bits)	Write desired level of Tick Timer 1 interrupt to bits 0-2.
7.	Local Bus Interrupter Enable Register	$FFF4006C (8 of 32 bits)	Set bit 24 (ETIC1) to 1 to enable Tick Timer 1 interrupts.
8.	I/O Control Register 1	$FFF40088 (8 of 32 bits)	Write a 1 to bit 23 to enable interrupts from the VMEchip2. A zero masks *all* interrupts from the VMEchip2.

Periodic Tick Timer 1 interrupts now occur, so you need an interrupt handler. Section C gives the details, as follows.

C. How to set up an interrupt handler routine. (Your interrupt handler should include the following features.)

Step	Action and Reference
1.	Be sure the M68000 vector base register is set up. Set the proper M68000 exception vector location so the processor vectors to your interrupt handler location. You can determine proper exception vector location to set from the M68000 vector base register, the VMEchip2 base register, and Table 4-4, *Local Bus Interrupter Summary*, from which you can determine the actual interrupt vector given on a Tick Timer 1 interrupt. Lower the M68000 mask so the vector level you programmed is accepted. The *interrupt handler itself* should include the following (steps 2 through 5).
2.	Confirm the Tick Timer 1 interrupt occurred, by reading the status of bit 24 of the Interrupter Status Register at $FFF40068. A high indicates an interrupt present.
3.	Clear Tick Timer 1 interrupt by writing a one to bit 24 of the Interrupt Clear Register at $FFF40074.
4.	Increment a software counter to keep track of the number of interrupts, if desired. Output a character or some other action (such as toggling the FAIL LED) on an appropriate count, such as 1000.
5.	Return from the exception.

MVME187 Interrupt Handling

The M88000 architecture currently does not support prioritized interrupts like the M68000 family. A single interrupt request is connected to the MC88100, and all device interrupt requests are summed into this signal. Software prioritization is required to implement interrupt priority encoding. Logic on the MVME187 assists the 88K software by encoding the interrupt priority in hardware.

A summary of the interrupt handling protocol for the MVME187 follows. Step numbers refer to circled items on Figure 1-2, which illustrates the protocol.

Step	Action and Reference
1.	A device requests interrupt service from the MPU. The hardware logic on the MVME187 PCCchip2 encodes this request on the INT signal connected to the MC88100. INT is asserted if the following is true: a. PCCchip2 Master Interrupt Enable is on, and b. External (Device) Interrupt Request Level is greater than the Interrupt Mask Level Register and greater than 0.
2.	The MC88100 saves its current context in order to service the interrupt request. The M88000 architecture requires that nested exceptions be handled totally in software. The MVME187 logic provides an external prioritizer for the hardware interrupts to assist in nested interrupt architecture. Software applications (operating systems, etc.) are encouraged to take advantage of the interrupt prioritizer. An example of how to use the prioritizer is given in the following steps. After the context of the processor is saved, the MC88100 is able to mask out interrupts with priorities lower than the current request under service. This masking operation is optional but recommended.
3.	The MC88100 then (through the Data CMMU) reads the Interrupt Priority Level Register on the PCCchip2 to determine the priority level of the interrupt request. At this time, the priority level is determined and the corresponding interrupt mask may be written to the PCCchip2, which masks any interrupts of equal or lower priority. This masking allows the application software (or operating system) to unfreeze shadowing and quickly re-enable exception handling. Without the Interrupt Mask Level Register and its associated interrupt prioritization, interrupt prioritization would have to be handled totally through software.
4.	The PCCchip2 Interrupt Priority Level Register contains a 3-bit value corresponding to the level of the interrupt requesting service. The MC88100 determines (through a table lookup or some other clever means) the address that emulates an IACK cycle on the Local Bus corresponding to this priority. The MC88100/Data CMMU drives this address on the bus (e.g., through a LOAD instruction) which generates an MC68040-like IACK cycle on the Local Bus.

Step	Action and Reference
5.	The IACK cycle causes a “vector” to be passed to the Data CMMU/MC88100 that can be used to index a table of interrupt service routine pointers. The initialization of the MVME187 module determines the interrupt priority and the device interrupt vectors, relieving considerable software overhead to determine interrupt priority.

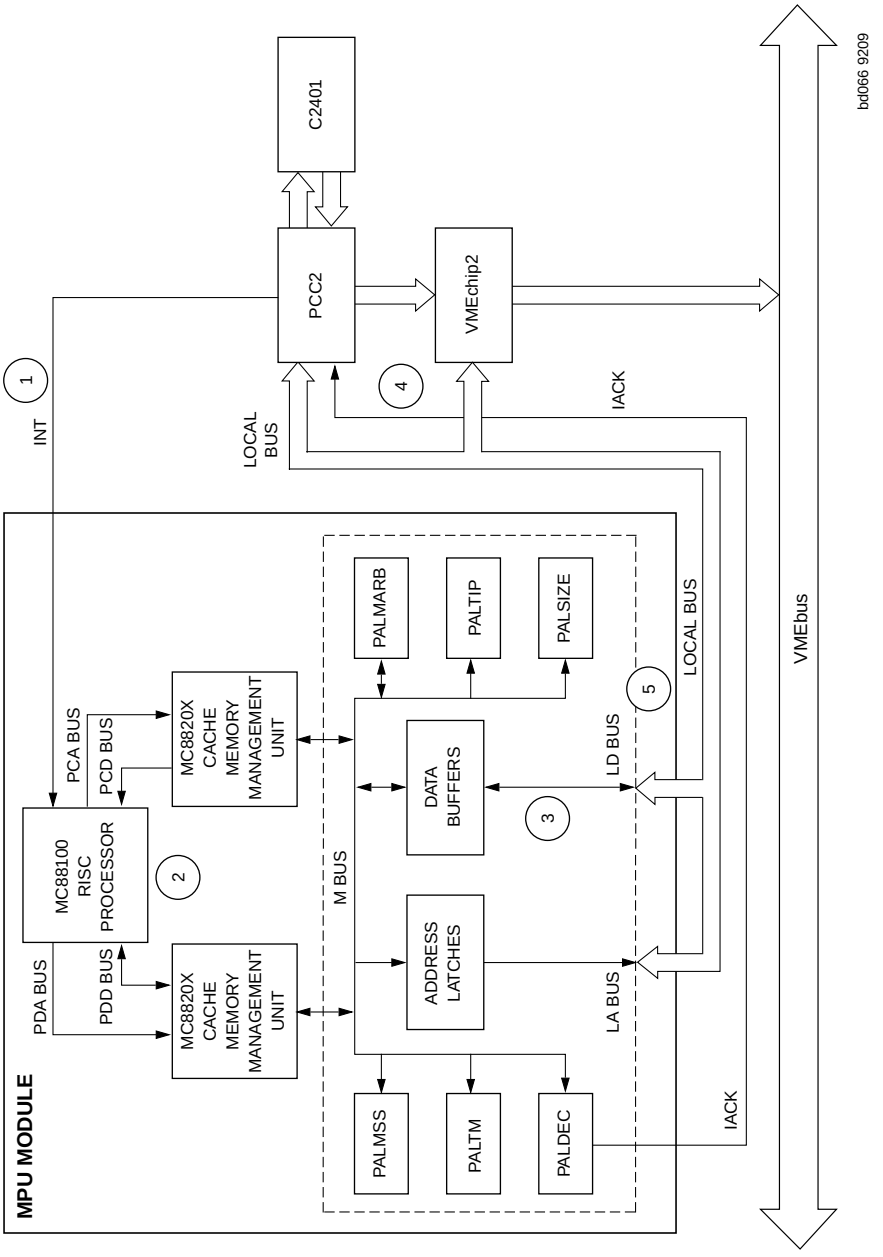


Figure 1-2. MVME187 Interrupt Handling Protocol

Cache Coherency

The next subsections describe the way each processor manages cache coherency.

Cache Coherency, MVME166/167

The MC68040 has the ability to watch Local Bus cycles executed by other Local Bus masters such as the SCSI DMA controller, the LAN, the VMEchip2 DMA controller, the VMEbus to Local Bus controller, and the VSB to Local Bus controller. This bus snooping capability is described in the *M68040 Microprocessors User's Manual* sections on *Cache Coherency* and *Bus Snooping Operation.*

When snooping is enabled, the MC68040 MPU can source data and invalidate cache entries as required by the current cycle. The MPU cannot watch VMEbus or VSB cycles that do not access the Local Bus. Software must ensure that data shared by multiple processors is kept in un-cached memory. The software must also mark all onboard I/O areas as cache inhibited and serialized.

Cache Coherency, MVME176/177

The MC68060 has the ability to watch the external bus during accesses by other bus masters, maintaining coherency between the MC68060's caches and external memory systems.To maintain cache coherency, the MC68060 provides automatic snoop-invalidation when it is not the bus master. When an external cycle is marked as snoopable, the bus snooper checks the caches and invalidates the matching data.

Unlike the MC68040, the MC68060 cannot source or sink cache data during alternate bus master accesses. Therefore, the MVME176/177 uses a single snoop control line – SC1. Snoop control bits for SC0 must be set to 0.

MC68060 cache coherency and bus snooping capabilities are described in the *M68060 Microprocessors User's Manual* sections on *Cache Coherency* and *Bus Snooping Operation.*

Cache Coherency, MVME187

Snooping is not supported on the MVME187. Software must ensure that data shared by multiple processors is kept in un-cached memory. The software must also mark all onboard I/O areas as cache inhibited and serialized. Snoop control bits in the 53C710, VMEchip2, VSBchip2, and PCCchip2 must be set to 0 (snoop disabled).

Using Bus Timers

This section illustrates the use of bus timers by describing the sequence of events when the MPU on one Single Board Computer accesses the Local Bus memory on another Single Board Computer using the VMEbus. A similar sequence of events could be described if the access occurred over the VSB. This scenario involves three bus timers, which normally should be set to quite different values:

Local bus timer	Measures the time an access to an onboard resource takes
VMEbus access timer	Measures the time from when the VMEbus request has been initiated to when a VMEbus grant has been obtained
Global VMEbus timer	Measures the time from when a VMEbus cycle begins to when it completes

The sequence begins when the MPU asserts a request for the Local Bus. The MPU must wait until the Local Bus is released by the current bus master before its cycle can begin. When the MPU is granted the Local Bus, it begins its cycle and the Local Bus timer starts counting. It continues to count until an address decode of the VMEbus address space is detected and then the timer stops. This is normally a very short period of time. In fact, all Local Bus non-error bus accesses are normally very short, such as the time to access onboard memory. Therefore, it is recommended this timer be set to a small value, such as 8 µsec.

The next timer to take over when one Single Board Computer accesses another is the VMEbus access timer. This measures the time between when the VMEbus has been address decoded and hence a VMEbus request has been made, and when VMEbus mastership has been granted. Because we have found in the past that some VME systems can become very busy, we recommend this time-out be set at a large value, such as 32 msec. For debug purposes this value can also be set to infinity.

Once the VMEbus has been granted, a third timer takes over. This is the global VMEbus timer. This timer starts when a transfer actually begins (DS0 or DS1 goes active) and ends when that transfer completes (DS0 or DS1 goes inactive). This time should be longer than any expected legitimate transfer time on the bus. We normally set it to 256 µsec. This timer can also be disabled for debug purposes.

Before a Single Board Computer access to another Single Board Computer can complete, however, the VMEchip2 on the accessed board must decode a slave access and request the Local Bus of the second board. When the Local Bus is granted (any in-process onboard transfers have completed), then the Local Bus timer of the accessed board starts. Normally, this is also set to 8 µsec. When the memory has the data available, a transfer acknowledge signal (TA) is given. This translates into a DTACK signal on the VMEbus which is then translated into a TA signal to the first requesting processor, and the transfer is complete.

If the VMEbus global timer expires on a legitimate transfer, the VMEbus to Local Bus controller in the VMEchip2 may become confused and the VMEchip2 may misbehave. Therefore the bus timer values must be set correctly. The correct settings may depend on the system configuration.

Indivisible Cycles

The Single Board Computers perform operations that require indivisible read-modify-write (RMW) memory accesses. These RMW sequences occur when the MMU modifies table entries or when the MPU executes one of the single-cycle instructions listed in Table 1-1.

Table 1-1. Single-Cycle Instructions

MPU	Instructions
MC68040	CAS, CAS2, TAS
MC68060	CAS CAS2 and misaligned CAS instructions are emulated by software (see NOTE)
MC88100	XMEM

Note Software emulation of CAS2 and misaligned CAS instructions is performed by the MC68060 Software Package, which is included in all Motorola-supplied operating systems for the MVME176/177. Contact your sales office for information about obtaining the MC68060 Software Package for use with other operating systems.

The Single Board Computers do not fully support all RMW operations in all possible cases. The modules make the following assumptions and support a limited subset of RMW instructions:

- The Single Board Computers support single-address RMW cycles.
- Multiple-address RMW cycles are not guaranteed indivisible and may cause illegal VMEbus cycles.

- Lock cycles caused by MMU table walks on the VMEbus do not cause illegal VMEbus cycles but they are not guaranteed do be indivisible.

On M68000-based systems, aligned CAS and all TAS cycles are always single-address RMW operations, while misaligned CAS and CAS2 operations and operations in the MMU can be multiple-address RMW cycles. The VMEbus does not support multiple-address RMW cycles and there is no defined protocol for supporting multiple-address RMW cycles that start onboard and then access offboard resources. Because it is not possible to tell if the processor is executing a single- or multiple-address read-modify-write cycle, software should only execute single-address RMW instructions. For efficiency, all CAS instructions should be aligned.

No Supervisor Stack Pointer on MC68060

On the MC68060, use of the supervisor stack pointer is reserved for system programming functions. All application software must be written to run in user mode. Such software will migrate to any M68000 platform without modification.

Programs that were written for platforms like the MC68040 that use the supervisor stack pointer must be recompiled before you can run them on an MC68060-based Single Board Computer, such as the MVME176/177.

Sources of Local BERR*

A TEA* signal (indicating a bus error) is returned to the Local Bus master when a Local Bus time-out occurs, a DRAM parity error occurs and parity checking is enabled, or a VME bus error occurs during a VMEbus access.

The sources of Local Bus errors on the Single Board Computers are described in the next subsections.

Local Bus Time-out

A Local Bus Time-out occurs whenever a Local Bus cycle does not complete within the programmed time (VMEbus bound cycles are not timed by the Local Bus timer). If the system is configured properly, this should only happen if software accesses a non-existent location within the onboard address range.

VMEbus Access Time-out

A VMEbus Access Time-out occurs whenever a VMEbus bound transfer does not receive a VMEbus bus grant within the programmed time. This is usually caused by another bus master holding the bus for an excessive period of time.

VMEbus BERR*

A VMEbus BERR* occurs when the BERR* signal line is asserted on the VMEbus while a Local Bus master is accessing the VMEbus. VMEbus BERR* should occur only if: an initialization routine samples to see if a device is present on the VMEbus and it is not, software accesses a non-existent device within the VMEbus range, incorrect configuration information causes the VMEchip2 to incorrectly access a device on the VMEbus (such as driving LWORD* low to a 16-bit board), a hardware error occurs on the VMEbus, or a VMEbus slave reports an access error (such as parity error).

Local DRAM Parity Error

When parity checking is enabled, the current bus master receives a bus error if it is accessing the local DRAM and a parity error occurs.

VMEchip2

An 8- or 16-bit write to the LCSR in the VMEchip2 causes a local BERR*.

VSBchip2 BERR*

The VSBchip2 on the MVME166/176 will return TEA* when specific VSB errors occur. Refer to the chapter *VSBchip2* for more information on bus errors from the VSB.

Bus Error Processing

Because different conditions can cause bus error exceptions, the software must be able to distinguish the source. To aid in this, status registers are provided for every Local Bus master. The next section describes the various causes of bus error and the associated status registers.

Generally, the bus error handler can interrogate the status bits and proceed with the result. However, an interrupt can happen during the execution of the bus error handler (before an instruction can write to the status register to raise the interrupt mask). If the interrupt service routine causes a second bus error, the status that indicates the source of the first bus error may be lost. The software must be written to deal with this.

Error Conditions

This section lists the various error conditions that are reported by the Single Board Computer hardware. A subheading identifies each error condition; a standard format provides the following information:

- Description of the error
- How notification of the error is made
- Status register(s) containing information about the error
- Comments pertaining to the error

MPU Parity Error

Description:	A DRAM parity error.
MPU Notification:	TEA is asserted during an MPU DRAM access.
Status:	Bit 9 of the MPU Status and DMA Interrupt Count Register in the VMEchip2 at address $FFF40048.
Comments:	After memory has been initialized, this error normally indicates a hardware problem.

MPU Offboard Error

Description:	An error occurred while the MPU was attempting to access an offboard resource.
MPU Notification:	TEA is asserted during offboard access.
Status:	Bit 8 of the MPU Status and DMA Interrupt Count Register. Address $FFF40048
Comments:	This can be caused by a VMEbus time-out, a VMEbus BERR, or a Single Board Computer VMEbus access time-out. The latter is the time from when the VMEbus has been requested to when it is granted.

MPU TEA - Cause Unidentified

Description:	An error occurred while the MPU was attempting an access.
MPU Notification:	TEA is asserted during an MPU access.
Status:	Bit 10 of the MPU Status and DMA Interrupt Count Register. Address $FFF40048
Comments:	No status was given as to the cause of the TEA assertion.

MPU Local Bus Time-out

Description:	An error occurred while the MPU was attempting to access a local resource.
MPU Notification:	TEA is asserted during the MPU access.
Status:	Bit 7 of the MPU Status and DMA Interrupt Count Register (actually in the DMAC Status Register). Address $FFF40048
Comments:	The Local Bus timer timed out. This usually indicates the MPU tried to read or write an address at which there was no resource. Otherwise, it indicates a hardware problem.

DMAC VMEbus Error

Description:	The DMAC experienced a VMEbus error during an attempted transfer.
MPU Notification:	DMAC interrupt (when enabled)
Status:	The VME bit is set in the DMAC Status Register (address $FFF40048 bit 1).
Comments:	This indicates the DMAC attempted to access a VMEbus address at which there was no resource or the VMEbus slave returned a BERR signal.

DMAC Parity Error

Description:	Parity error while the DMAC was reading DRAM.
MPU Notification:	DMAC interrupt (when enabled)
Status:	The DLPE bit is set in the DMAC Status Register (address $FFF40048 bit 5).
Comments:	If the TBL bit is set (address $FFF40048 bit 2) the error occurred during a command table access, otherwise the error occurred during a data access.

DMAC Offboard Error

Description:	Error encountered while the Local Bus side of the DMAC was attempting to go to the VMEbus.
MPU Notification:	DMAC interrupt (when enabled)
Status:	The DLOB bit is set in the DMAC Status Register (address $FFF40048 bit 4).
Comments:	This is normally caused by a programming error. The Local Bus address of the DMAC should not be programmed with a Local Bus address that maps to the VMEbus. If the TBL bit is set (address $FFF40048 bit 2) the error occurred during a command table access, otherwise the error occurred during a data access.

DMAC LTO Error

Description:	A Local Bus time-out (LTO) occurred while the DMAC was Local Bus master.
MPU Notification:	DMAC interrupt (when enabled)
Status:	The DLTO bit is set in the DMAC Status Register (address $FFF40048 bit 3).
Comments:	This indicates the DMAC attempted to access a Local Bus address at which there was no resource. If the TBL bit is set (address $FFF40048 bit 2) the error occurred during a command table access, otherwise the error occurred during a data access.

DMAC TEA - Cause Unidentified

Description:	An error occurred while the DMAC was Local Bus master and additional status was not provided.
MPU Notification:	DMAC interrupt (when enabled)
Status:	The DLBE bit is set in the DMAC Status Register (address $FFF40048 bit 6).

Comments: An 8- or 16-bit write to the LCSR in the VMEchip2 causes this error. If the TBL bit is set (address $FFF40048 bit 2) the error occurred during a command table access, otherwise the error occurred during a data access.

SCC Retry Error

Description:	Local Bus Retry occurred due to VMEbus Dual Port Lock or LAN-wanted-Bus while the SCC was Local Bus master.
MPU Notification:	SCC Transmit Interrupt or SCC Receive Interrupt
Status:	SCC Transmit Interrupt Status Register SCC Transmit Current Buffer Address Register SCC Receive Interrupt Status Register High SCC Receive Current Buffer Address Register PCCchip2 SCC Error Status Register ($FFF4201C)
Comments:	The DMA controllers in the SCC should not be programmed to access the VMEbus. Refer to the *Serial Port Interface* section in this chapter. SCC Transmit and Receive interrupt enables are controlled in the SCC and in the PCCchip2.

SCC Parity Error

Description:	Parity Error detected while the SCC was reading DRAM.
MPU Notification:	SCC Transmit Interrupt or SCC Receive Interrupt
Status:	SCC Transmit Interrupt Status Register SCC Transmit Current Buffer Address Register SCC Receive Interrupt Status Register High SCC Receive Current Buffer Address Register PCCchip2 SCC Error Status Register ($FFF4201C)
Comments:	SCC Transmit and Receive interrupt enables are controlled in the SCC and in the PCCchip2.

SCC Offboard Error

Description:	Error encountered while the SCC was attempting to go to the VMEbus.
MPU Notification:	SCC Transmit Interrupt or SCC Receive Interrupt

Status:	SCC Transmit Interrupt Status Register SCC Transmit Current Buffer Address Register SCC Receive Interrupt Status Register High SCC Receive Current Buffer Address Register PCCchip2 SCC Error Status Register ($FFF4201C)
Comments:	SCC Transmit and Receive interrupt enables are controlled in the SCC and in the PCCchip2.

SCC LTO Error

Description:	Local Bus Time-out occurred while the SCC was Local Bus master.
MPU Notification:	SCC Transmit Interrupt or SCC Receive Interrupt
Status:	SCC Transmit Interrupt Status Register SCC Transmit Current Buffer Address Register SCC Receive Interrupt Status Register High SCC Receive Current Buffer Address Register PCCchip2 SCC Error Status Register ($FFF4201C)
Comments:	SCC Transmit and Receive interrupt enables are controlled in the SCC and in the PCCchip2.

LAN Parity Error

Description:	Parity error while the LANCE was reading DRAM
MPU Notification:	PCCchip2 Interrupt (LAN ERROR IRQ)
Status:	PCCchip2 LAN Error Status Register ($FFF42028)
Comments:	The LANCE has no ability to respond to TEA so the error interrupt and status are provided in the PCCchip2. Control for the interrupt is in the PCCchip2 LAN Error Interrupt Control Register ($FFF4202B).

LAN Offboard Error

Description:	Error encountered while the LANCE was attempting to go to the VMEbus.
MPU Notification:	PCCchip2 Interrupt (LAN ERROR IRQ)
Status:	PCCchip2 LAN Error Status Register ($FFF42028)
Comments:	The LANCE has no ability to respond to TEA so the error interrupt and status are provided in the PCCchip2. Control for the interrupt is in the PCCchip2 LAN Error Interrupt Control Register ($FFF4202B).

LAN LTO Error

Description:	Local Bus Time-out occurred while the LANCE was Local Bus master.
MPU Notification:	PCCchip2 Interrupt (LAN ERROR IRQ)
Status:	PCCchip2 LAN Error Status Register ($FFF42028)
Comments:	The LANCE has no ability to respond to TEA so the error interrupt and status are provided in the PCCchip2. Control for the interrupt is in the PCCchip2 LAN Error Interrupt Control Register ($FFF4202B).

SCSI Parity Error

Description:	Parity error detected while the 53C710 was reading DRAM.
MPU Notification:	53C710 Interrupt
Status:	53C710 DMA Status Register 53C710 DMA Interrupt Status Register PCCchip2 SCSI Error Status Register ($FFF4202C)
Comments:	53C710 interrupt enables are controlled in the 53C710 and in the PCCchip2.

SCSI Offboard Error

Description:	Error encountered while the 53C710 was attempting to go to the VMEbus.
MPU Notification:	53C710 Interrupt
Status:	53C710 DMA Status Register 53C710 DMA Interrupt Status Register PCCchip2 SCSI Error Status Register ($FFF4202C)
Comments:	53C710 interrupt enables are controlled in the 53C710 and in the PCCchip2.

SCSI LTO Error

Description:	Local Bus Time-out occurred while the 53C710 was Local Bus master.
MPU Notification:	53C710 Interrupt
Status:	53C710 DMA Status Register 53C710 DMA Interrupt Status Register PCCchip2 SCSI Error Status Register ($FFF4202C)
Comments:	53C710 interrupt enables are controlled in the 53C710 and in the PCCchip2.

Hardware Configuration 2

Introduction

This chapter describes the hardware configuration features of the Single Board Computers.

SCSI Termination

The system configurer must ensure that the SCSI bus is properly terminated at both ends. This is done on each system as follows:

MVME166 MVME176	SCSI bus termination is provided on the main board. The terminators are enabled or disabled by a jumper. If the SCSI bus ends at the MVME166/176, the SCSI terminators must be enabled by installing the jumper, as described in the section, *Configuration Jumpers, MVME166* and *Configuration Jumpers, MVME177*.
MVME167 MVME177 MVME187	Sockets are provided for the terminators on the P2 transition board. If the SCSI bus ends at the P2 transition board, then termination resistors must be installed on the P2 transition board. +5V power to the SCSI bus TERM power line and termination resistors is provided through a fuse located on the P2 transition board (for the MVME167 and MVME187), but is provided through a fuse on the MVME712 transition board and a diode on the P2 transition board (for the MVME177).

2

Connectors

The Single Board Computers have two 96-position DIN connectors, P1 and P2, which provide connections as follows:

MVME166 MVME176	P1 rows A, B, C and P2 row B	provide the VMEbus interconnection
	P2 rows A and C	provide the connection to the VSB
MVME167 MVME177 MVME187	P1 rows A, B, C and P2 row B	provide the VMEbus interconnection
	P2 rows A and C	provide the connection to the SCSI bus, serial ports, Ethernet, and printer

All models have a 20-pin connector mounted behind the front panel. When the board is enclosed in a chassis and the front panel is not visible, this connector allows the reset, abort and LED functions to be extended to the control panel of the system, where they are visible.

The MVME166/176 has a 68-pin mini D ribbon shielded connector for the SCSI bus interface and a 100-pin mini D ribbon shielded connector for the serial ports, Ethernet, and printer. Refer to the support manual for the specific board for detailed connector pinout information.

Fuses

All boards supply power to various I/O devices. The power sources are fused on the main board. The fuses are included to protect the board in case any of the power sources are shorted. If a fuse is blown, the board may behave in a erratic manner or stop functioning completely. If any of the onboard I/O devices behave this way, the fuses should be checked. There are several LEDs to monitor the status of the fuses.

MVME166 Fuses

The MVME166 provides +12V, -12V, and +5V to the transition boards through fuses F1, F3, and F4. The fused +5V is also provided to the 20-pin remote reset connector. These voltage sources are used by the transition boards to power the serial port drivers and any LAN transceivers connected to the transition board. The RPWR LED is lit when all three voltages are available.

The MVME166 provides +5V to the SCSI bus TERMPWR signal through fuse F2 located near the front panel SCSI bus connector. The TPWR LED monitors the SCSI bus TERMPWR signal. Because any devices on the SCSI bus can provide TERMPWR, the LED does not directly indicate the condition of the fuse. If the LED is not illuminated, the fuse should be checked.

MVME176 Polyswitches

The MVME176 provides +12V, -12V, and +5V to the transition boards through polyswitches F2, F3, or F4. The +5V is also provided to the 20-pin remote reset connector through polyswitch F4. These voltage sources are used by the transition boards to power the serial port drivers and any LAN transceivers connected to the transition board. The RPWR LED is lit when all three voltages are available.

The MVME176 provides +5V to the SCSI bus TERMPWR signal through polyswitch F1. The TPWR LED monitors the SCSI bus TERMPWR signal.

MVME167/177/187 Fuses

These Single Board Computer models provide +5V power to the 20-pin remote reset connector J3 through fuse F1 located near the connector. This voltage source is only used when the LED functions are extended and there is no onboard monitor.

The models provide +12V power to the transition boards through fuse F2 located between the VMEbus P1 and P2 connectors. This voltage source is used to power LAN transceivers connected to the

MVME712*x* transition boards and for pullup resistors on some serial port lines. The +12V LED on the front panel is used to monitor this voltage source.

The P2 transition board used with these models provides +5V to SCSI bus TERMPWR signal through a fuse. There is no monitor LED for this fuse. The MVME712M transition board and some SCSI bus terminators provide monitor LEDs.

Note The MVME177 uses polyswitches, which reset automatically when the board is powered down and the temperature cools.

Configuration Jumpers

The Single Board Computers were designed to provide software control for most options. Some options cannot be done in software, however, so are done by jumpers on headers. The next subsections describe the jumper settings for the Single Board Computers.

Configuration Jumpers, MVME166

This section describes the jumpers used on the MVME166.

Table 2-1. Configuring MVME166 Headers

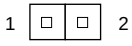
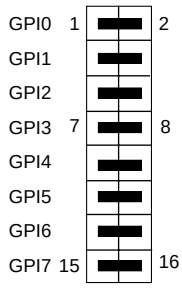
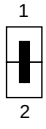
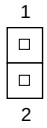
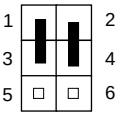
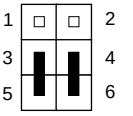
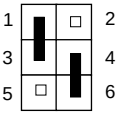
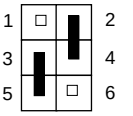
Header Number	Header Description	Configuration	Jumpers		Notes
J2	Onboard SCSI bus terminator	Enabled	1 -- 2 (Factory configuration)		1
		Disabled	None		
J3	General purpose software readable	GPI0 - GPI2: User-definable GPI3: Normal debugger GPI4 - GPI7: User-definable	1 -- 2 (GPI0) 3 -- 4 (GPI1) 5 -- 6 (GPI2) 7 -- 8 (GPI3) 9 -- 10 (GPI4) 11 -- 12 (GPI5) 13 -- 14 (GPI6) 15 -- 16 (GPI7) (Factory configuration)		2, 3
J6	VMEbus system controller function	Enabled	1 -- 2 (Factory configuration)		4
		Disabled	None		

Table 2-1. Configuring MVME166 Headers (Continued)

Header Number	Header Description	Configuration	Jumpers		Notes
J7	SRAM backup power source select	Primary source = VMEbus +5V STBY Secondary source = VMEbus +5V STBY	1 -- 3 2 -- 4 (Factory configuration)		5
		Primary source = onboard battery Secondary source = onboard battery	3 -- 5 4 -- 6		
		Primary source = VMEbus +5V STBY Secondary source = onboard battery	1 -- 3 4 -- 6		
		Primary source = onboard battery Secondary source = VMEbus +5V STBY	2 -- 4 3 -- 5		

MVME166 Header Notes:	
1.	The MVME166 provides terminators for the SCSI bus. Setting a jumper on J2 enables the SCSI terminator. Removing the jumper disables the terminator.
2.	The general purpose readable jumpers on header J3 can be read as I/O control register 3 (at $FFF40088, bits 0-7) in the VMEchip2 LCSR (see Chapter 4, *VMEchip2*). The bit values are read as a 1 when the jumper is off, and as a 0 when the jumper is on.
3.	On the MVME166, a jumper installed on pins 7 and 8 of J3 (bit 3) selects the normal debugger; if the jumper is removed, BootBug is enabled.

MVME166 Header Notes: (Continued)	
4.	The MVME166 can be the VMEbus system controller. The system controller function is enabled or disabled by jumpers on header J6. When the MVME166 is system controller, the SCON LED is turned on. The VMEchip2 may be configured as a system controller when J6 is jumpered as shown.
5.	Header J7 is used to select the power source used to back up the SRAM on the MVME166 when the backup battery is installed. **Caution** Do not remove all jumpers from J7. This may disable the SRAM. If you remove the battery, you must install jumpers on J7 between pins 1 and 3 and pins 2 and 4, as shown in the *Factory configuration* drawing above.

2

Configuration Jumpers, MVME167

This section describes the jumpers used on the MVME167.

Table 2-2. Configuring MVME167 Headers

Header Number	Header Description	Configuration	Jumpers		Notes
J1	General purpose software readable jumpers	User-definable, software-readable general purpose	16 -- 15 (GPI7) 14 -- 13 (GPI6) 12 -- 11 (GPI5) 10 -- 9 (GPI4) 8 -- 7 (GPI3) 6 -- 5 (GPI2) 4 -- 3 (GPI1) 2 -- 1 (GPI0) (Factory configuration)		1
J2	System controller header	System controller	1 -- 2 (Factory configuration)		2
		Not system controller	None		

Table 2-2. Configuring MVME167 Headers (Continued)

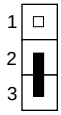
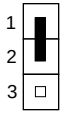
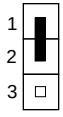
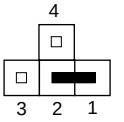
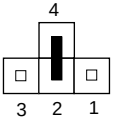
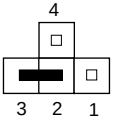
Header Number	Header Description	Configuration	Jumpers		Notes
J6	Serial Port 4 clock configuration select headers	Receive RTXC4	2 -- 3 (Factory configuration)		3
		Drive RTXC4	1 -- 2		
J7		Receive TRXC4	2 -- 3 (Factory configuration)		3
		Drive TRXC4	1 -- 2		

Table 2-2. Configuring MVME167 Headers (Continued)

Header Number	Header Description	Configuration	Jumpers		Notes
J8	SRAM backup power source select header (optional)	VMEbus +5V STBY	2 -- 1 (Factory configuration when optional battery is present)		4
		Backup power disabled	4 -- 2		
		Backup from optional battery	3 -- 2		

MVME167 Header Notes:	
1.	The general purpose readable jumpers on header J1 can be read as I/O control register 3 (at $FFF40088, bits 0-7) in the VMEchip2 LCSR (see Chapter 4, *VMEchip2*). The bit values are read as a 1 when the jumper is off, and as a 0 when the jumper is on.
2.	The MVME167 can be the VMEbus system controller. The system controller function is enabled or disabled by jumpers on header J2. When the MVME167 is system controller, the SCON LED is turned on. The VMEchip2 may be configured as a system controller when J2 is jumpered as shown.

MVME167 Header Notes: (Continued)	
3.	Serial port 4 can be configured to use clock signals provided by the RTXC4 and TRXC4 signal lines. Headers J6 and J7 on the MVME167 configure serial port 4 to drive or receive RTXC4 and TRXC4, respectively. Both jumpers should be set the same way. Factory configuration is with port 4 set to receive both signals. The remaining configuration of the clock lines is accomplished using the Serial Port 4 Clock Configuration Select header on the MVME712M transition board. Refer to the *MVME712M Transition Module and P2 Adapter Board User's Manual* for configuration of that header.
4.	Optional header J8 is used to select the power source used to back up the SRAM on the MVME167 when the optional backup battery is installed. **Caution** Do not remove all jumpers from J8. This may disable the SRAM. If you remove the battery, you must install jumpers on J8 between pins 2 and 4, as shown for *Backup power disabled*.

2

Configuration Jumpers, MVME177

This section describes the jumpers used on the MVME177.

Table 2-3. Configuring MVME177 Headers

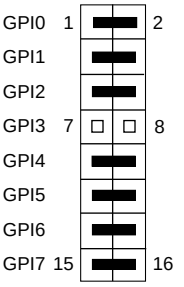
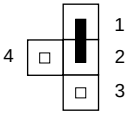
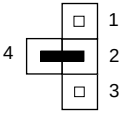
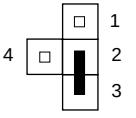
Header Number	Header Description	Configuration	Jumpers		Notes
J1	General purpose software readable jumpers	GPI0 - GPI2: User-definable GPI3: Reserved GPI4 - GPI7: User-definable	1 -- 2 (GPI0) 3 -- 4 (GPI1) 5 -- 6 (GPI2) 9 -- 10 (GPI4) 11 -- 12 (GPI5) 13 -- 14 (GPI6) 15 -- 16 (GPI7) (Factory configuration)		1, 2
J2	SRAM backup power source select header	VMEbus +5V STBY	2 -- 1 (Factory configuration)		3
		Backup power disabled	4 -- 2		
		Backup from battery	3 -- 2		

Table 2-3. Configuring MVME177 Headers (Continued)

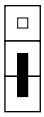
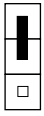
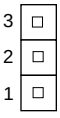
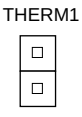
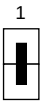
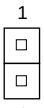
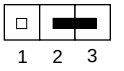
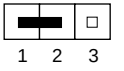
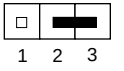
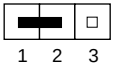
Header Number	Header Description	Configuration	Jumpers		Notes
J6	System controller header	System controller	1 -- 2 (Factory configuration)		4
		Auto system controller	2 -- 3		
		Not system controller	None		
J7	Thermal sensing pins	Connected to MC68060 internal thermal resistor	None (Factory configuration)		5
J8	EPROM/Flash configuration jumper	1MB EPROM and 2MB Flash enabled	1 -- 2 (Factory configuration)		6
		4MB Flash enabled	None		

Table 2-3. Configuring MVME177 Headers (Continued)

Header Number	Header Description	Configuration	Jumpers		Notes
J9	Serial Port 4 clock configuration select headers	Receive RTXC4	2 -- 3 (Factory configuration)		7
		Drive RTXC4	1 -- 2		
J10		Receive TRXC4	2 -- 3 (Factory configuration)		7
		Drive TRXC4	1 -- 2		

MVME177 Header Notes :	
1.	The general purpose readable jumpers on header J1 can be read as I/O control register 3 (at $FFF40088, bits 0-7) in the VMEchip2 LCSR (see Chapter 4, *VMEchip2*). The bit values are read as a 1 when the jumper is off, and as a 0 when the jumper is on.
2.	On the MVME177, pins 7 and 8 (bit 3) are removed for board ID and the bit value is reserved.

MVME177 Header Notes (Continued):	
3.	Header J2 is used to select the power source used to back up the SRAM on the MVME177 when the backup battery is installed. **Caution** — Do not remove all jumpers from J2. This may disable the SRAM. If you remove the battery, you must install jumpers on J2 between pins 2 and 4, as shown for *Backup power disabled*.
4.	The MVME177 can be the VMEbus system controller. The system controller function is enabled, disabled, or configured for automatic select by jumpers on header J6. If set for AUTO SCON, the MVME177 determines if it is the system controller by its position on the bus. If the MVME177 is in the first slot from the left, it configures itself as the system controller. When the MVME177 is system controller, the SCON LED is turned on. The VMEchip2 may be configured as a system controller when J6 is jumpered as shown. **Caution** — Do not jumper J6 to Auto System Controller. At this time, this feature is not functioning properly. Set up jumpers on J6 only as System Controller or Not System Controller. **Note** — AUTO SCON only works with a non-active backplane.
5.	The thermal sensing pins, THERM1 and THERM2, are connected to an internal thermal resistor and provide information about the average temperature of the processor. Refer to the *M68000 Microprocessors User's Manual* for additional information on the use of these pins.
6.	The FLASH jumper, J8, is used to select the Flash memory and EPROM configuration on the MVME177. If the board is configured for 1MB EPROM and 2MB Flash memory, the VMEchip2 GPIO bits can be programmed to select the first or second 2MB of Flash. See Chapter 1, *Programming Issues*, and Chapter 4, *VMEchip2*, for more information on Flash memory. You can also use the 177Bug **SFLASH** command to map the Flash memory.
7.	Serial port 4 can be configured to use clock signals provided by the RTXC4 and TRXC4 signal lines. Headers J9 and J10 on the MVME177 configure serial port 4 to drive or receive RTXC4 and TRXC4, respectively. Both jumpers should be set the same way. Factory configuration is with port 4 set to receive both signals. The remaining configuration of the clock lines is accomplished using the Serial Port 4 Clock Configuration Select header on the MVME712M transition board. Refer to the *MVME712M Transition Module and P2 Adapter Board User's Manual* for configuration of that header.

2

Configuration Jumpers, MVME187

This section describes the jumpers used on the MVME187.

Table 2-4. Configuring MVME187 Headers

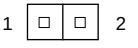
Header Number	Header Description	Configuration	Jumpers		Notes
J1	General purpose software readable jumpers	User-definable, software-readable general purpose	16 -- 15 (GPI7) 14 -- 13 (GPI6) 12 -- 11 (GPI5) 10 -- 9 (GPI4) 8 -- 7 (GPI3) 6 -- 5 (GPI2) 4 -- 3 (GPI1) 2 -- 1 (GPI0) (Factory configuration)		1
J2	System controller header	System controller	1 -- 2 (Factory configuration)		2
		Not system controller	None		

Table 2-4. Configuring MVME187 Headers (Continued)

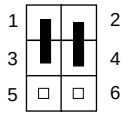
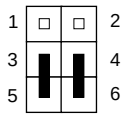
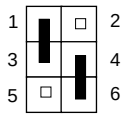
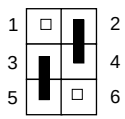
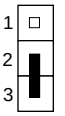
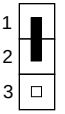
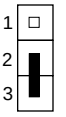
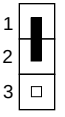
Header Number	Header Description	Configuration	Jumpers		Notes
J6	SRAM backup power source select header (optional)	Primary source = VMEbus +5V STBY Secondary source = VMEbus +5V STBY	1 -- 3 2 -- 4 (Factory configuration)		3
		Primary source = onboard battery Secondary source = onboard battery	3 -- 5 4 -- 6		
		Primary source = VMEbus +5V STBY Secondary source = onboard battery	1 -- 3 4 -- 6		
		Primary source = onboard battery Secondary source = VMEbus +5V STBY	2 -- 4 3 -- 5		

Table 2-4. Configuring MVME187 Headers (Continued)

Header Number	Header Description	Configuration	Jumpers		Notes
J7	Serial Port 4 clock configuration select headers	Receive TRXC4	2 -- 3 (Factory configuration)		4
		Drive TRXC4	1 -- 2		
J8		Receive RTXC4	2 -- 3 (Factory configuration)		4
		Drive RTXC4	1 -- 2		

MVME187 Header Notes:	
1.	The general purpose readable jumpers on header J1 can be read as I/O control register 3 (at $FFF40088, bits 0-7) in the VMEchip2 LCSR (see Chapter 4, *VMEchip2*). The bit values are read as a 1 when the jumper is off, and as a 0 when the jumper is on.
2.	The MVME187 can be the VMEbus system controller. The system controller function is enabled or disabled by jumpers on header J2. When the MVME187 is system controller, the SCON LED is turned on. The VMEchip2 may be configured as a system controller when J2 is jumpered as shown.

MVME187 Header Notes: (Continued)	
3.	Optional header J6 is used to select the power source used to back up the SRAM on the MVME187 when the optional backup battery is installed. **Caution** Do not remove all jumpers from J6. This may disable the SRAM. If you remove the battery, you must install jumpers on J6 between pins 1 and 3 and pins 2 and 4, as shown in the *Factory configuration* drawing above.
4.	Serial port 4 can be configured to use clock signals provided by the TRXC4 and RTXC4 signal lines. Headers J7 and J8 on the MVME187 configure serial port 4 to drive or receive TRXC4 and RTXC4, respectively. Both jumpers should be set the same way. Factory configuration is with port 4 set to receive both signals. The remaining configuration of the clock lines is accomplished using the Serial Port 4 Clock Configuration Select header on the MVME712M transition board. Refer to the *MVME712M Transition Module and P2 Adapter Board User's Manual* for configuration of that header.

2

Configuration Jumpers, MVME176

This section describes the jumpers used on the MVME176.

Table 2-5. MVME176 Headers

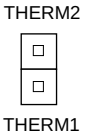
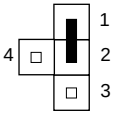
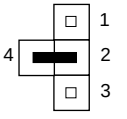
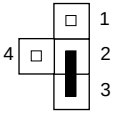
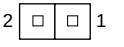
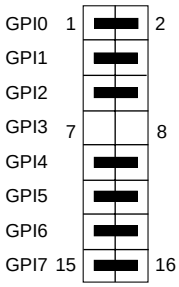
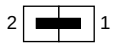
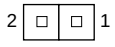
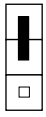
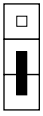
Header Number	Header Description	Configuration	Jumpers		Notes
J2	Thermal sensing pins	Connected to MC68060 internal thermal resistor	None (Factory configuration)		1
J3	EPROM/Flash select header	Mixed EPROM/Flash	1 -- 2 (Factory configuration)		2
		All Flash	None		
J4	SRAM backup power source select header	VMEbus +5V STBY	2 -- 1 (Factory configuration)		3
		Backup power disabled	4 -- 2		
		Backup from battery	3 -- 2		
J6	Serial arbitration select header	Parallel	1 -- 2		
		Serial	None (Factory configuration)		

Table 2-5. MVME176 Headers (Continued)

Header Number	Header Description	Configuration	Jumpers		Notes
J7	General purpose software readable jumper header	GPI0 - GPI2: User-definable GPI3: Board ID GPI4 - GPI7: User-definable	1 -- 2 (GPI0) 3 -- 4 (GPI1) 5 -- 6 (GPI2) 9 -- 10 (GPI4) 11 -- 12 (GPI5) 13 -- 14 (GPI6) 15 -- 16 (GPI7) (Factory configuration)		4, 5
J8	SCSI bus terminator enable header	Enabled	1 -- 2 (Factory configuration)		
		Disabled	None		
J9	VMEbus system controller select header	System controller	1 -- 2 (Factory configuration)		6
		Auto system controller	2 -- 3		
		Not system controller	None		

Notes:	
1.	The thermal sensing pins, THERM1 and THERM2, are connected to an internal thermal resistor and provide information about the average temperature of the processor. Refer to the *M68060 Microprocessors User's Manual* for additional information on the use of these pins.
2.	Header J8 is used to select the EPROM/Flash memory configuration for the MVME176. If the board is configured for mixed EPROM/Flash (1MB EPROM and 2MB Flash), the VMEchip2 GPIO bits can be programmed to select the first or second 2MB of Flash. See Chapter 4 for more information on Flash memory. You may also use the 176Bug SFLASH command to map the Flash memory.
3.	Header J4 is used to select the power source used to back up the SRAM on the MVME176 when the backup battery is installed. **Caution** — Do not remove the jumper from J4. This may disable the SRAM. If you remove the battery, install a jumper across pins 2 and 4 of header J4 as shown for Backup Power Disabled.
4.	Header J7 can be read as I/O control register 3 (at $FFF40088, bits 0-7) in the VMEchip2 LCSR (see Chapter 4, *VMEchip2*). The bit values are read as a one when the jumper is off, and as a zero when the jumper is on.
5.	GPI3 is not user-definable; it is used for board ID.
6.	Header J9 is used to enable/disable VMEbus system controller function. If the auto system controller configuration is selected, the MVME176 determines if it is the system controller by its position on the bus. If the MVME176 is in the first slot from the left, it configures itself as the system controller. The SCON LED is turned on when the MVME176 is the system controller.

Memory Maps 3

Introduction

This chapter provides memory map definitions for devices on the Single Board Computers, except for the ASICS listed below. The chapter that describes the ASIC includes the memory map for the ASIC and all the programmable registers.

Chapter 4	VMEchip2
Chapter 5	VSBchip2
Chapter 6	PCCchip2
Chapter 7	MEMC040 memory controller
Chapter 8	MCECC memory controller

There are two aspects to memory maps:

- The mapping of all resources as viewed by local bus masters (local bus memory map). This mapping is described in detail in later sections of this chapter.
- The mapping of onboard resources as viewed by external masters (VMEbus memory map or VSB memory map). This mapping is described briefly in the next two sections and in detail in the previously listed chapters.

VMEbus Memory Map

The VMEchip2 includes a user-programmable map decoder for the VMEbus to local bus interface. The map decoder allows you to program the starting and ending address and the modifiers to which the MVME166/167/176/177/187 responds.

The VMEchip2 also includes a user-programmable map decoder for the GCSR. The GCSR map decoder allows you to program the starting address of the GCSR in the VMEbus short I/O space.

Chapter 4, *VMEchip2* includes programming information on the VMEbus to local bus map decoders.

VSB Memory Map

The VSBchip2, on the MVME166/176, includes a user-programmable map decoder for the VSB to local bus interface. This map decoder allows VSB masters access to devices on the local bus.

Chapter 5, *VSBchip2* includes programming information on the VSB to local bus map decoders.

Local Bus Memory Map

The local bus memory map is split into different address spaces by the transfer type (TT) signals. The local resources respond to the normal access and interrupt acknowledge codes.

Normal Address Range

The memory map of devices that respond to the normal address range is shown in the following tables. The normal address range is defined by the Transfer Type (TT) signals on the local bus as follows:

MVME166 MVME167 MVME176 MVME177	Transfer Types 0, 1, and 2
MVME187	Transfer Types 0 and 1

Table 3-1, *Local Bus Memory Map*, lists the entire map from $00000000 to $FFFFFFFF. Many areas of the map are user-programmable, and suggested uses are shown in the table. The cache inhibit function is programmable in the MMUs. The onboard I/O space must be marked cache inhibit and serialized in its page table.

Table 3-2, *I/O Devices Memory Map*, further defines the map for the local I/O devices portion of the local bus Main Memory Map.

Detailed I/O Memory Maps

Table 3-3, *Cirrus Logic CD2401 Serial Port Memory Map*, through Table 3-9, *TOD Clock Memory Map*, give the detailed memory maps for:

Table 3-3	CD2401 serial chip
Table 3-4	MC68230 PI/T chip
Table 3-5	82596CA Ethernet chip
Table 3-6	53C710 SCSI chip
Table 3-7	DS1643/MK48T18 BBRAM/TOD clock
Table 3-8	BBRAM configuration area
Table 3-9	TOD clock

Note Manufacturers' errata sheets for the various chips are available by contacting your local Motorola sales representative. A non-disclosure agreement may be required.

Table 3-1. Local Bus Memory Map

Address Range		Devices Accessed	Port Size	Size	Software Cache Inhibit	Notes
$00000000 - DRAMSIZE		User Programmable (Onboard DRAM)	D32	DRAMSIZE	N	1, 2
DRAMSIZE - $FF7FFFFF		User Programmable (VMEbus or VSB)	D32/D16	3GB	?	3, 4
$FF800000 -	$FF8FFFFF	FLASH (MVME166)	D32	1MB	N	1
	$FFBFFFFF	ROM (MVME167/187)	D32	4MB	N	1
	$FFBFFFFF	EPROM and Flash (176/177)	D32	1 MB EPROM 4 MB Flash	N	1
$FFC00000 - $FFDFFFFF		Reserved	--	2MB	--	5
$FFE00000 - $FFE1FFFF		Onboard SRAM (default)	D32	128KB	N	6
$FFE20000 - $FFEFFFFF		Onboard SRAM (repeated)	D32	896KB	N	6
$FFF00000 - $FFFEFFFF		Local I/O Devices (Refer to next table)	D32-D8	960 KB (1MB - 64KB)	Y	3
$FFFF0000 - $FFFFFFFF		User Programmable (VMEbus A16)	D32/D16	64KB	?	2, 4

Notes 1. Onboard EPROM on the MVME167/187 (Download EPROM on the MVME166) appears at $00000000 - ROMSIZE following a local bus reset. The EPROM appears at 0 until the ROM0 bit is cleared in the VMEchip2. The ROM0 bit is located at address $FFF40030 bit 20. The EPROM must be disabled at 0 before the DRAM is enabled. The VMEchip2, VSBchip2, and DRAM map decoders are disabled by a local bus reset. (The Download EPROM appears at 0 until the DR0 bit is cleared in the PCCchip2. The DR0 bit is located at address $FFF42000 bit 15.)

On the MVME176/177, Flash/EPROM devices appear at $FF800000 - $FFBFFFFF and also appear at $00000000 - $003FFFFF if the ROM0 bit in the VMEchip2 EPROM

control register is high (ROM0=1). ROM0 is set to 1 after each reset. ROM0 bit must be cleared before other resources (DRAM or SRAM) can be mapped in this range ($00000000 - $003FFFFF). The VMEchip2 and DRAM map decoders are disabled by a local bus reset. On the MVME176/177, the Flash/EPROM memory is mapped at $00000000 - $003FFFFF by hardware default through the VMEchip2. Refer to the Flash and EPROM descriptions in Chapter 1, *Programming Issues*.

2. This area is user-programmable. The suggested use is shown in the table. The DRAM decoder is programmed in the MEMC040 or MCECC chip, and the local-to-VMEbus decoders are programmed in the VMEchip2. The local-to-VSB decoders are programmed in the VSBchip2.

3. Size is approximate.

4. Cache inhibit depends on devices in area mapped.

5. This area is not decoded. If these locations are accessed and the local bus timer is enabled, the cycle times out and is terminated by a TEA signal.

6. MVME166/176/177 SRAM has battery backup. SRAM battery backup is optional for the MVME167/187.

Table 3-2. I/O Devices Memory Map

Address Range	Devices Accessed	Port Size	Size	Notes
$FFF00000 - $FFF3FFFF	reserved	--	256KB	5
$FFF40000 - $FFF400FF	VMEchip2 (LCSR)	D32	256B	1,4
$FFF40100 - $FFF401FF	VMEchip2 (GCSR)	D32-D8	256B	1,4
$FFF40200 - $FFF40FFF	reserved	--	3.5KB	5,7
$FFF41000 - $FFF41FFF	VSBchip2	D32-D8	4KB	1,10
$FFF42000 - $FFF42FFF	PCCchip2	D32-D8	4KB	1
$FFF43000 - $FFF430FF	MEMC040/MCECC #1	D8	256B	1
$FFF43100 - $FFF431FF	MEMC040/MCECC #2	D8	256B	1
$FFF43200 - $FFF43FFF	MEMC040s/MCECCs (repeated)	--	3.5KB	1,7
$FFF44000 - $FFF44FFF	reserved	--	4KB	5
$FFF45000 - $FFF451FF	CD2401 (Serial Comm. Cont.)	D16-D8	512B	1,9
$FFF45200 - $FFF45DFF	reserved	--	3KB	7,9
$FFF45E00 - $FFF45FFF	MC68230	--	512B	1,9
$FFF46000 - $FFF46FFF	82596CA (LAN)	D32	4KB	1,8
$FFF47000 - $FFF47FFF	53C710 (SCSI)	D32/D8	4KB	1
$FFF48000 - $FFF4FFFF	reserved	--	32KB	5
$FFF50000 - $FFF6FFFF	reserved	--	128KB	5
$FFF70000 - $FFF76FFF	reserved	--	28KB	6
$FFF77000 - $FFF77FFF	CODE CMMU	D32	4KB	1,2
$FFF78000 - $FFF7EFFF	reserved	--	28KB	6
$FFF7F000 - $FFF7FFFF	DATA CMMU	D32	4KB	1,2
$FFF80000 - $FFF9FFFF	reserved (download EPROM for 166 only)	--	128KB	11
$FFFA0000 - $FFFBFFFF	reserved	--	128KB	5
$FFFC0000 - $FFFCFFFF	DS1643/MK48T18 (BBRAM, TOD Clock)	D32-D8	64KB	1
$FFFD0000 - $FFFDFFFF	reserved	--	64KB	5
$FFFE0007	IACK LEVEL 1	D8	1 byte	2,3
$FFFE000B	IACK LEVEL 2	D8	1 byte	2,3
$FFFE000F	IACK LEVEL 3	D8	1 byte	2,3
$FFFE0013	IACK LEVEL 4	D8	1 byte	2,3
$FFFE0017	IACK LEVEL 5	D8	1 byte	2,3
$FFFE001B	IACK LEVEL 6	D8	1 byte	2,3
$FFFE001F	IACK LEVEL 7	D8	1 byte	2,3
$FFFE0020 - $FFFEFFFF	IACK LEVELS (repeated)	--	64KB	2,7

Notes 1. For a complete description of the register bits, refer to the data sheet for the specific chip. For a more detailed memory map refer to the following detailed peripheral device memory maps.

2. Code and data CMMUs are only on the MVME187. On the MVME187, a read anywhere from location $FFFE0004 through $FFFE001C causes an interrupt acknowledge cycle at the specified level. Refer to the chapter, *PCCchip2*, for information on reading the current interrupt level and setting the interrupt mask.

On the MVME166/167/176/177, this area does not return an acknowledge signal. If the local bus timer is enabled, the access times out and is terminated by a TEA signal.

3. Byte reads should be used to read the interrupt vector. These locations do not respond when an interrupt is not pending. If the local bus timer is enabled, the access times out and is terminated by a TEA signal.

4. Writes to the LCSR in the VMEchip2 must be 32 bits. LCSR writes of 8 or 16 bits terminate with a TEA signal. Writes to the GCSR may be 8, 16 or 32 bits. Reads to the LCSR and GCSR may be 8, 16 or 32 bits.

5. This area does not return an acknowledge signal. If the local bus timer is enabled, the access times out and is terminated by a TEA signal.

6. This area does return an acknowledge signal.

7. Size is approximate.

8. Port commands to the 82596CA must be written as two 16-bit writes: upper word (two-byte) first and lower word (two-byte) second.

9. The MC68230 is included on the MVME166/176 only. The CD2401 appears repeatedly from $FFF45200 to $FFF45FFF on the MVME167/177/187. The area from $FFF45200 to $FFF45DFF does not return an acknowledge on the MVME166/176. If the local bus timer is enabled, the access times out and is terminated by a TEA signal.

10. The VSBchip2 is included on the MVME166/176 only. On all other models this area does not return an acknowledge signal. If the local bus timer is enabled, the access times out and is terminated by a TEA signal.

11. Download EPROM on the MVME166 only. EPROM and Flash memory can coexist. On the MVME167/176/177/187, this area does return an acknowledge signal.

Table 3-3. Cirrus Logic CD2401 Serial Port Memory Map

Base Address = $FFF45000

Register Description	Register Name	Offsets	Size	Access
Global Registers				
Global Firmware Revision Code Register	GFRCR	81	B	R
Channel Access Register	CAR	EE	B	R/W
Option Registers				
Channel Mode Register	CMR	1B	B	R/W
Channel Option Register 1	COR1	10	B	R/W
Channel Option Register 2	COR2	17	B	R/W
Channel Option Register 3	COR3	16	B	R/W
Channel Option Register 4	COR4	15	B	R/W
Channel Option Register 5	COR5	14	B	R/W
Channel Option Register 6	COR6	18	B	R/W
Channel Option Register 7	COR7	07	B	R/W
Special Character Register 1	SCHR1	1F	B	R/W Async
Special Character Register 2	SCHR2	1E	B	R/W Async
Special Character Register 3	SCHR3	1D	B	R/W Async
Special Character Register 4	SCHR4	1C	B	R/W Async
Special Character Range low	SCRl	23	B	R/W Async
Special Character Range high	SCRh	22	B	R/W Async
LNext Character	LNXT	2E	B	R/W Async
Bit Rate and Clock Option Registers				
Receive Frame Address Register1	RFAR1	1F	B	R/W Sync
Receive Frame Address Register2	RFAR2	1E	B	R/W Sync
Receive Frame Address Register3	RFAR3	1D	B	R/W Sync
Receive Frame Address Register4	RFAR4	1C	B	R/W Sync
CRC Polynomial Select Register	CPSR	D6	B	R/W Sync
Receive Baud Rate Period Register	RBPR	CB	B	R/W
Receive Clock Option Register	RCOR	C8	B	R/W
Transmit Baud Rate Period Register	TBPR	C3	B	R/W
Transmit Clock Option Register	TCOR	C0	B	R/W

Table 3-3. Cirrus Logic CD2401 Serial Port Memory Map (Continued)

Base Address = $FFF45000

Register Description	Register Name	Offsets	Size	Access
Channel Command and Status Registers				
Channel Command Register	CCR	13	B	R/W
Special Transmit Command Register	STCR	12	B	R/W
Channel Status Register	CSR	1A	B	R
Modem Signal Value Registers	MSVR-RTS	DE	B	R/W
	MSVR-DTR	DF	B	R/W
Interrupt Registers				
Local Interrupt Vector Register	LIVR	09	B	R/W
Interrupt Enable Register	IER	11	B	R/W
Local Interrupting Channel Register	LICR	26	B	R/W
Stack Register	STK	E2	B	R
Receive Interrupt Registers				
Receive Priority Interrupt Level Register	RPILR	E1	B	R/W
Receive Interrupt Register	RIR	ED	B	R
Receive Interrupt Status Register	RISR	88	W (NOTE)	R/W
Receive Interrupt Status Register low	RISRl	89	B	R
Receive Interrupt Status Register high	RISRh	88	B	R
Receive FIFO Output Count	RFOC	30	B	R
Receive Data Register	RDR	F8	B	R
Receive End Of Interrupt Register	REOIR	84	B	W
Transmit Interrupt Registers				
Transmit Priority Interrupt Level Register	TPILR	E0	B	R/W
Transmit Interrupt Register	TIR	EC	B	R
Transmit Interrupt Status Register	TISR	8A	B	R
Transmit FIFO Transfer Count	TFTC	80	B	R
Transmit Data Register	TDR	F8	B	W
Transmit End Of Interrupt Register	TEOIR	85	B	W

Table 3-3. Cirrus Logic CD2401 Serial Port Memory Map (Continued)

Base Address = $FFF45000

Register Description	Register Name	Offsets	Size	Access
Modem Interrupt Registers				
Modem Priority Interrupt Level Register	MPILR	E3	B	R/W
Modem Interrupt Register	MIR	EF	B	R
Modem (/Timer) Interrupt Status Register	MISR	8B	B	R
Modem End Of Interrupt Register	MEOIR	86	B	W
DMA Registers				
DMA Mode Register (write only)	DMR	F6	B	W
Bus Error Retry Count	BERCNT	8E	B	R/W
DMA Buffer Status	DMABSTS	19	B	R
DMA Receive Registers				
A Receive Buffer Address Lower	ARBADRL	42	W	R/W
A Receive Buffer Address Upper	ARBADRU	40	W	R/W
B Receive Buffer Address Lower	BRBADRL	46	W	R/W
B Receive Buffer Address Upper	BRBADRU	44	W	R/W
A Receive Buffer Byte Count	ARBCNT	4A	W	R/W
B Receive Buffer Byte Count	BRBCNT	48	W	R/W
A Receive Buffer Status	ARBSTS	4F	B	R/W
B Receive Buffer Status	BRBSTS	4E	B	R/W
Receive Current Buffer Address Lower	RCBADRL	3E	W	R
Receive Current Buffer Address Upper	RCBADRU	3C	W	R
DMA Transmit Registers				
A Transmit Buffer Address Lower	ATBADRL	52	W	R/W
A Transmit Buffer Address Upper	ATBADRU	50	W	R/W
B Transmit Buffer Address Lower	BTBADRL	56	W	R/W
B Transmit Buffer Address Upper	BTBADRU	54	W	R/W
A Transmit Buffer Byte Count	ATBCNT	5A	W	R/W
B Transmit Buffer Byte Count	BTBCNT	58	W	R/W
A Transmit Buffer Status	ATBSTS	5F	B	R/W
B Transmit Buffer Status	BTBSTS	5E	B	R/W
Transmit Current Buffer Address Lower	TCBADRL	3A	W	R
Transmit Current Buffer Address Upper	TCBADRU	38	W	R

Table 3-3. Cirrus Logic CD2401 Serial Port Memory Map (Continued)

Base Address = $FFF45000

Register Description	Register Name	Offsets	Size	Access
Timer Registers				
Timer Period Register	TPR	DA	B	R/W
Receive Time-out Period Register	RTPR	24	W	R/W Async
Receive Time-out Period Regis low	RTPRl	25	B	R/W Async
Receive Time-out Period Register high	RTPRh	24	B	R/W Async
General Timer 1	GT1	2A	W	R Sync
General Timer 1 low	GT1l	2B	B	R Sync
General Timer 1 high	GT1h	2A	B	R Sync
General Timer 2	GT2	29	B	R Sync
Transmit Timer Register	TTR	29	B	R Async

Note This is a 16-bit register.

Table 3-4. MC68230 PI/T Register Map

MC68230 Base Address = $FFF45E00

Offset	Physical Address	Register Name	Register Description
$1	$FFF45E01	PGCR	Port General Control Register
$3	$FFF45E03	PSRR	Port Service Request Register
$5	$FFF45E05	PADDR	Port A Data Direction Register
$7	$FFF45E07	PBDDR	Port B Data Direction Register
$9	$FFF45E09	PCDDR	Port C Data Direction Register
$B	$FFF45E0B	PIVR	Port Interrupt Vector Register
$D	$FFF45E0D	PACR	Port A Control Register
$F	$FFF45E0F	PBCR	Port B Control Register
$11	$FFF45E11	PADR	Port A Data Register
$13	$FFF45E13	PBDR	Port B Data Register
$15	$FFF45E15	PAAR	Port A Alternate Register
$17	$FFF45E17	PBAR	Port B Alternate Register
$19	$FFF45E19	PCDR	Port C Data Register
$1B	$FFF45E1B	PSR	Port Status Register
$21	$FFF45E21	TCR	Timer Control Register
$23	$FFF45E23	TIVR	Timer Interrupt Vector Register
$27	$FFF45E27	CPRH	Counter Preload Register High
$29	$FFF45E29	CPRM	Counter Preload Register Middle
$2B	$FFF45E2B	CPRL	Counter Preload Register Low
$2F	$FFF45E2F	CNTRH	Counter Register High
$31	$FFF45E31	CNTRM	Counter Register Middle
$33	$FFF45E33	CNTRL	Counter Register Low
$35	$FFF45E35	TSR	Timer Status Register

Table 3-5. 82596CA Ethernet LAN Memory Map

82596CA Ethernet LAN Directly Accessible Registers

Address	Data Bits	
	D31 . . . D16	D15 . . . D0
$FFF46000	Upper Command Word	Lower Command Word
$FFF46004	MPU Channel Attention (CA)	

Notes

1. Refer to the MPU Port and MPU Channel Attention registers in Chapter 6.

2. After resetting, you must write the System Configuration Pointer to the command registers before writing to the MPU Channel Attention register. Writes to the System Configuration Pointer must be upper word first, lower word second.

Table 3-6. 53C710 SCSI Memory Map

Base Address = $FFF47000

Big Endian Mode	53C710 Register Address Map				SCRIPTs Mode and Little Endian Mode
00	SIEN	SDID	SCNTL1	SCNTL0	00
04	SOCL	SODL	SXFER	SCID	04
08	SBCL	SBDL	SIDL	SFBR	08
0C	SSTAT2	SSTAT1	SSTAT0	DSTAT	0C
10	DSA				10
14	CTEST3	CTEST2	CTEST1	CTEST0	14
18	CTEST7	CTEST6	CTEST5	CTEST4	18
1C	TEMP				1C
20	LCRC	CTEST8	ISTAT	DFIFO	20
24	DCMD	DBC			24
28	DNAD				28
2C	DSP				2C
30	DSPS				30
34	SCRATCH				34
38	DCNTL	DWT	DIEN	DMODE	38
3C	ADDER				3C

Note Accesses may be 8-bit or 32-bit, but not 16-bit.

Table 3-7. DS1643/MK48T18 BBRAM/TOD Clock Memory Map

Address Range	Description	Size (Bytes)	Notes
$FFFC0000 - $FFFC0FFF	User Area	4096	1,3
$FFFC1000 - $FFFC10FF	Networking Area	256	1, 4
$FFFC1100 - $FFFC16F7	Operating System Area	1528	1, 5
$FFFC16F8 - $FFFC1EF7	Debugger Area	2048	1, 6
$FFFC1EF8 - $FFFC1FF7	BBRAM Configuration Area	256	1, 7
$FFFC1FF8 - $FFFC1FFF	TOD Clock	8	2, 8

Notes

1. Defined by software.

2. Defined by the chip hardware.

3. Reserved for user data.

   If you use the **NIOT** debugger command, the **ENV** command Network Autoboot Configuration Parameters need to be saved/retained somewhere in this address range. Refer to the *68KBUG* or *88KBUG* debugging package manuals, or your individual board installation manual, for details.

4. Used by Motorola networking software.

5. Used by the SYSTEM V/68 or SYSTEM V/88 operating system.

6. Used by the debuggers for the respective Single Board Computers.

7. Configuration area defined in Table 3-8, *BBRAM Configuration Area Memory Map*.

8. Memory map detailed in Table 3-9, *TOD Clock Memory Map*.

Table 3-8. BBRAM Configuration Area Memory Map

Address Range	Description	Size (Bytes)
$FFFC1EF8 - $FFFC1EFB	Version	4
$FFFC1EFC - $FFFC1F07	Serial Number	12
$FFFC1F08 - $FFFC1F17	Board ID	16
$FFFC1F18 - $FFFC1F27	PWA	16
$FFFC1F28 - $FFFC1F2B	Speed	4
$FFFC1F2C - $FFFC1F31	Ethernet Address	6
$FFFC1F32 - $FFFC1F33	Reserved	2
$FFFC1F34 - $FFFC1F35	SCSI ID	2
$FFFC1F36 - $FFFC1F3D	System ID	8
$FFFC1F3E - $FFFC1F45	Mezzanine Board 1 PWB	8
$FFFC1F46 - $FFFC1F4D	Mezzanine Board 1 Serial Number	8
$FFFC1F4E - $FFFC1F55	Mezzanine Board 2 PWB	8
$FFFC1F56 - $FFFC1F5D	Mezzanine Board 2 Serial Number	8
$FFFC1F5E - $FFFC1FF6	Reserved	153
$FFFC1FF7	Checksum	1

Note The data structure of the configuration bytes starts at $FFFC1EF8 and is as follows:

```
struct brdi_cnfg {
       char   version[4];
       char   serial[12];
       char   id[16];
       char   pwa[16];
       char   speed[4];
       char   ethernet_adr[6];
       char   fill[2];
       char   lscsiid[2];
       char   sysid[8];
       char   brd1_pwb[8];
       char   brd1_serial[8];
       char   brd2_pwb[8];
```

```
        char    brd2_serial[8];
        char    reserved[153];
        char    cksum[1];
}
```


The fields are defined as follows:

1. Four bytes are reserved for the revision or version of this structure. This revision is stored in ASCII format, with the first two bytes being the major version numbers and the last two bytes being the minor version numbers. For example, if the version of this structure is 1.0, this field contains:

   `0100`

2. Twelve bytes are reserved for the serial number of the board in ASCII format. For example, this field could contain:

   `000000470476`

3. Sixteen bytes are reserved for the board ID in ASCII format. For example, for a 16 MB, 25 MHz MVME167 board, this field contains:

   `MVME167-003B`

   (The 12 characters are followed by four blanks.)

4. Sixteen bytes are reserved for the printed wiring assembly (PWA) number assigned to this board in ASCII format. This includes the `01-W` prefix. This is for the main logic board if more than one board is required for a set. Additional boards in a set are defined by a structure for that set. For example, for a 16 MB, 25 MHz MVME167 board at revision A, the PWA field contains:

   `01-W3899B03A`

   (The 12 characters are followed by four blanks.)

5. Four bytes contain the speed of the board in MHz. The first two bytes are the whole number of MHz and the second two bytes are fractions of MHz. For example, for a 25.00 MHz board, this field contains:

   `2500`

6. Six bytes are reserved for the Ethernet address. The address is stored in hexadecimal format. (Refer to the detailed description earlier in Chapter 1.) If the board does not support Ethernet, this field is filled with zeros.
7. These two bytes are reserved.
8. Two bytes are reserved for the local SCSI ID. The SCSI ID is stored in ASCII format.
9. Eight bytes are reserved for the systems serial ID, for boards used in a system.
10. Eight bytes are reserved for the printed wiring board (PWB) number assigned to the first mezzanine board in ASCII format. This does *not* include the `01-W` prefix. For example, for a 16MB parity mezzanine at revision E, the PWB field contains:

    ```
    3690B03E
    ```

11. Eight bytes are reserved for the serial number assigned to the first mezzanine board in ASCII format.
12. Eight bytes are reserved for the printed wiring board (PWB) number assigned to the optional second mezzanine board in ASCII format.
13. Eight bytes are reserved for the serial number assigned to the optional second mezzanine board in ASCII format.
14. Growth space (153 bytes) is reserved. This pads the structure to an even 256 bytes. System-specific items, such as size of system side, and systems side version, may go here.
15. The final one byte of the area is reserved for a checksum (as defined in the user's manual for the board debugging package), for security and data integrity of the configuration area of the NVRAM. This data is stored in hexadecimal format.

Table 3-9. TOD Clock Memory Map

Address	Data Bits								Function	
	D7	D6	D5	D4	D3	D2	D1	D0		
$FFFC1FF8	W	R	S	--	--	--	--	--	CONTROL	
$FFFC1FF9	ST	--	--	--	--	--	--	--	SECONDS	00
$FFFC1FFA	x	--	--	--	--	--	--	--	MINUTES	00
$FFFC1FFB	x	x	--	--	--	--	--	--	HOUR	00
$FFFC1FFC	x	FT	x	x	x	--	--	--	DAY	01
$FFFC1FFD	x	x	--	--	--	--	--	--	DATE	01
$FFFC1FFE	x	x	x	--	--	--	--	--	MONTH	01
$FFFC1FFF	--	--	--	--	--	--	--	--	YEAR	00

Notes W = Write Bit R = Read Bit S = Sign Bit
ST = Stop Bit FT = Frequency Test x = Unused

VMEchip2 4

Introduction

This chapter defines the VMEchip2, local bus to VMEbus interface chip.

The VMEchip2 interfaces as MC68040-compatible local bus (Local Bus) to the VMEbus. In addition to the VMEbus defined functions, the VMEchip2 includes a Local Bus to VMEbus DMA controller, VME board support features, and Global Control and Status Registers (GCSR) for interprocessor communications.

Summary of Features

This section lists the major features of the VMEchip2 chip.

- Local Bus to VMEbus Interface:
  - Programmable Local Bus map decoder.
  - Programmable short, standard and extended VMEbus addressing.
  - Programmable AM codes.
  - Programmable 16-bit and 32-bit VMEbus data width.
  - Software-enabled write posting mode.
  - Write post buffer (one cache line or one four-byte).
  - Automatically performs dynamic bus sizing for VMEbus cycles.
  - Software-configured VMEbus access timers.
  - Local bus to VMEbus Requester:
    Software-enabled FAIR request mode;
    Software-configured release modes:
    Release-When-Done (RWD), and
    Release-On-Request (ROR); and
    Software-configured BR0*-BR3* request levels.

- VMEbus Bus to Local Bus Interface:
  - Programmable VMEbus map decoder.
  - Programmable AM decoder.
  - Programmable Local Bus snoop enable.
  - Simple VMEbus to Local Bus address translation.
  - 8-bit, 16-bit, and 32-bit VMEbus data width.
  - 8-bit, 16-bit, and 32-bit block transfer.
  - Standard and extended VMEbus addressing.
  - Software-enabled write posting mode.
  - Write post buffer (17 four-bytes in BLT mode, two four-bytes in non-BLT mode).
  - An eight four-byte read ahead buffer (BLT mode only).
- 32-Bit Local Bus to VMEbus DMA Controller:
  - Programmable 16-bit, 32-bit, and 64-bit VMEbus data width.
  - Programmable short, standard and extended VMEbus addressing.
  - Programmable AM code.
  - Programmable Local Bus snoop enable.
  - A 16 four-byte FIFO data buffer.
  - Supports up to 4 GB of data per DMA request.
  - Automatically adjusts transfer size to optimize bus utilization.
  - DMA complete interrupt.
  - DMAC command chaining is supported by a singly-linked list of DMA commands.
  - VMEbus DMA controller requester:
    Software-enabled FAIR request modes;
    Software-configured release modes:
    Release-On-Request (ROR), and

Release-On-End-Of-Data (ROEOD);
Software-configured BR0-BR3 request levels, and
Software enabled bus-tenure timer.

- VMEbus Interrupter:
  - Software-configured IRQ1-IRQ7 interrupt request level.
  - 8-bit software-programmed status/ID register.
- VMEbus System Controller:
  - Arbiter with software-configured arbitration modes:
    Priority (PRI).
    Round-Robin-Select (RRS).
    Single-level (SGL).
  - Programmable arbitration timer.
  - IACK daisy-chain driver.
  - Programmable bus timer.
  - SYSRESET logic.
- Global Control Status Register Set:
  - Four location monitors.
  - Global control of locally detected failures.
  - Global control of local reset.
  - Four global attention interrupt bits.
  - A chip ID and revision register.
  - Four 16-bit dual-ported general purpose registers.
- Interrupt Handler:
  - All interrupts are level-programmable.
  - All interrupts are maskable.
  - All interrupts provide a unique vector.
  - Software and external interrupts.
- Watchdog timer.
- Two 32-bit tick timers.

- Map decoder and control for two banks of EPROM (Flash memory on the MVME166/176).
- Map decoder and control for one bank of static RAM.
- Support for RESET and ABORT switches.

Functional Blocks

The following sections provide an overview of the functions provided by the VMEchip2. See Figure 4-1 for a block diagram of the VMEchip2. A detailed programming model for the local control and status registers (LCSR) is provided in the following section. A detailed programming model for the global control and status registers (GCSR) is provided in the next section.

Local Bus to VMEbus Interface

The Local Bus to VMEbus interface allows Local Bus masters access to global resources on the VMEbus. This interface includes a *local bus slave*, a *write post buffer*, and a *VMEbus master*.

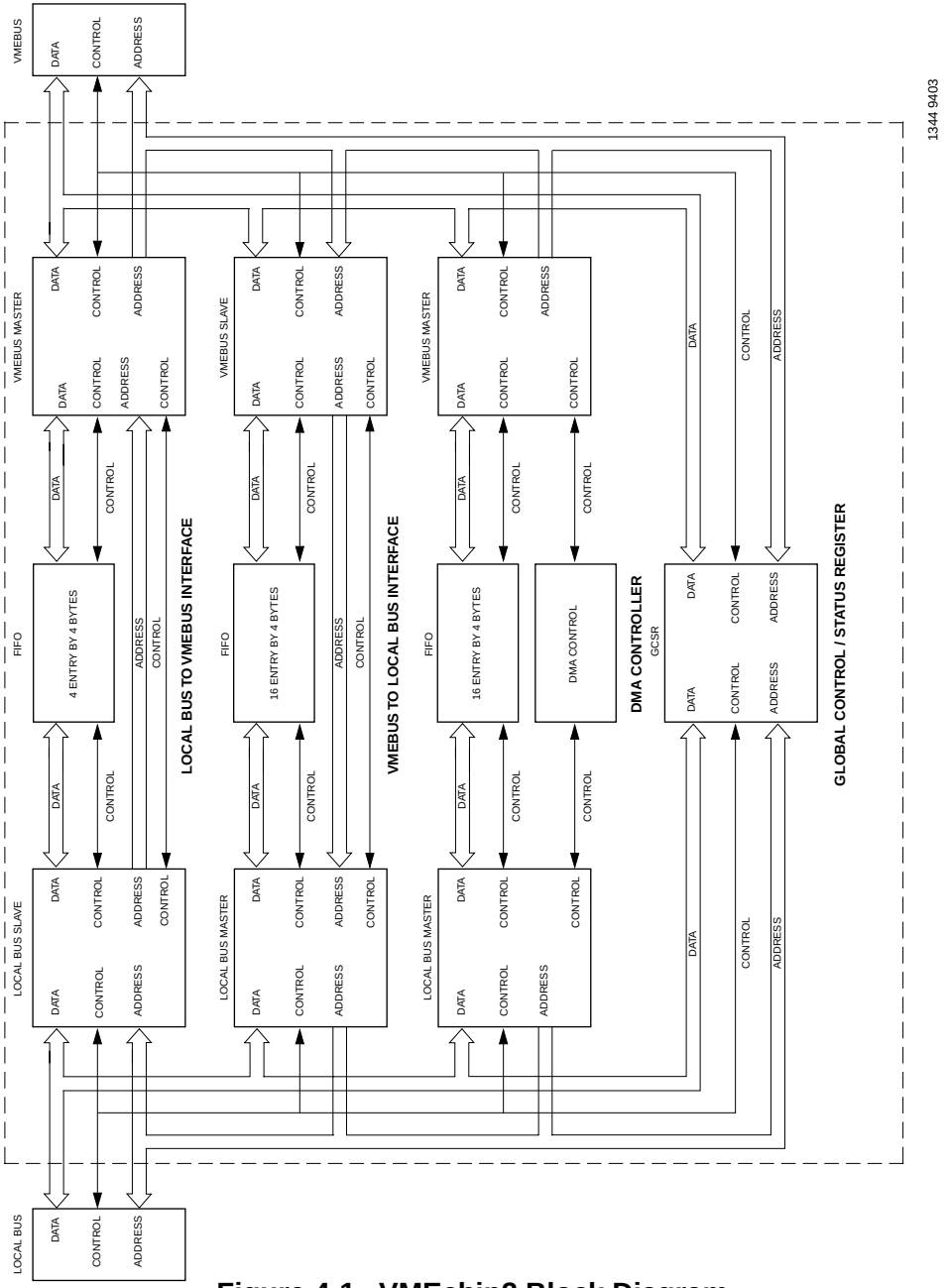


Figure 4-1. VMEchip2 Block Diagram

Using programmable map decoders with programmable attribute bits, the Local Bus to VMEbus interface can be configured to provide the following VMEbus capabilities:

Addressing capabilities: A16, A24, A32
Data transfer capabilities: D08, D16, D32

The *local bus slave* includes six Local Bus map decoders for accessing the VMEbus. The first four map decoders are general purpose programmable decoders, while the other two are fixed and are dedicated for I/O decoding.

The first four map decoders compare Local Bus address lines A31 through A16 with a 16-bit start address and a 16-bit end address. When an address in the selected range is detected, a VMEbus select is generated to the VMEbus master. Each map decoder also has eight attribute bits and an enable bit. The attribute bits are for VMEbus AM codes, D16 enable, and write post (WP) enable.

The fourth map decoder also includes a 16-bit alternate address register and a 16-bit alternate address select register. This allows any or all of the upper 16 address bits from the Local Bus to be replaced by bits from the alternate address register. The feature allows the Local Bus master to access any VMEbus address.

Using the four programmable map decoders, separate VMEbus maps can be created, each with its own attributes. For example, one map can be configured as A32, D32 with write posting enabled while a second map can be A24, D16 with write posting disabled.

The first I/O map decoder decodes Local Bus addresses $FFFF0000 through $FFFFFFFF as the short I/O A16/D16 or A16/D32 area, and the other provides an A24/D16 space at $F0000000 to $F0FFFFFF and an A32/D16 space at $F1000000 to $FF7FFFFF.

Supervisor/non-privileged and program/data space is determined by attribute bits. Write posting may be enabled or disabled for each decoder I/O space and this map decoder may be enabled or disabled.

When *write posting* is enabled, the VMEchip2 stores the Local Bus address and data and then acknowledges the Local Bus master. The Local Bus is then free to perform other operations while the VMEbus master requests the VMEbus and performs the requested operation.

The write post buffer stores one byte, two-byte, four-byte or one cache line (four four-bytes). Write posting should only be enabled when bus errors are not expected. If a bus error is returned on a write posted cycle, the local processor is interrupted, if the interrupt is enabled. The address of the error is not saved. Normal memory never returns a bus error on a write cycle. However, some VMEbus ECC memory cards perform a read-modify-write operation and therefore may return a bus error if there is an error on the read portion of a read-modify-write. Write posting should not be enabled when this type of memory card is used. Also, memory should not be sized using write operations if write posting is enabled. I/O areas that have holes should not be write posted if software may access non-existent memory. Using the programmable map decoders, write posting can be enabled for "safe" areas and disabled for areas which are not "safe".

Block transfer is not supported because the Local Bus block transfer capability is not compatible with the VMEbus.

The *VMEbus master* supports dynamic bus sizing. When a local device initiates a quad-byte access to a VMEbus slave that only has the D16 data transfer capability, the chip executes two double-byte cycles on the VMEbus, acknowledging the local device after all requested four-bytes have been accessed. This enhances the portability of software because it allows software to run on the system regardless of the physical organization of global memory.

Using the Local Bus map decoder attribute register, the AM code that the master places on the VMEbus can be programmed under software control.

The VMEchip2 includes a software-controlled VMEbus access timer, and it starts ticking when the chip is requested to do a VMEbus data transfer or an interrupt acknowledge cycle. The timer

stops ticking once the chip has started the data transfer on the VMEbus. If the data transfer does not begin before the timer times out, the timer drives the Local Bus error signal, and sets the appropriate status bit in the Local Control and Status Register (LCSR). Using control bits in the LCSR, the timer can be disabled, or it can be enabled to drive the Local Bus error signal after 64 µs, 1 ms, or 32 ms.

The VMEchip2 includes a software-controlled VMEbus write post timer, and it starts ticking when a data transfer to the VMEbus is write posted. The timer stops ticking once the chip has started the data transfer on the VMEbus. If this does not happen before the timer times out, the chip aborts the write posted cycle and send an interrupt to the Local Bus interrupter. If the write post bus error interrupt is enabled in the Local Bus interrupter, the local processor is interrupted to indicate a write post time-out has occurred. The write post timer has the same timing as the VMEbus access timer.

Local Bus to VMEbus Requester

The requester provides all the signals necessary to allow the Local Bus to VMEbus master to request and be granted use of the VMEbus. The chip connects to all signals that a VMEbus requester is required to drive and monitor.

Requiring no external jumpers, the chip provides the means for software to program the requester to request the bus on any one of the four bus request levels, automatically establishing the bus grant daisy-chains for the three inactive levels.

The requester requests the bus if any of the following conditions occur:

1. The Local Bus master initiates either a data transfer cycle or an interrupt acknowledge cycle to the VMEbus.
2. The chip is requested to acquire control of the VMEbus as signaled by the DWB input signal pin.
3. The chip is requested to acquire control of the VMEbus as signaled by the DWB control bit in the LCSR.

The Local Bus to VMEbus requester in the VMEchip2 implements a FAIR mode. By setting the LVFAIR bit, the requester refrains from requesting the VMEbus until it detects its assigned request line in its negated state.

The Local Bus to VMEbus requester attempts to release the VMEbus when the requested data transfer operation is complete, the DWB pin is negated, the DWB bit in the LCSR is negated and the bus is not being held by a lock cycle. The requester releases the bus as follows:

1. When the chip is configured in the release-when-done (RWD) mode, the requester releases the bus when the above conditions are satisfied.
2. When the chip is configured in the release-on-request (ROR) mode, the requester releases the bus when the above conditions are satisfied and there is a bus request pending on one of the VMEbus request lines.

To minimize the timing overhead of the arbitration process, the Local Bus to VMEbus requester in the VMEchip2 executes an early release of the VMEbus. If it is about to release the bus and it is executing a VMEbus cycle, the requester releases BBSY before its associated master completes the cycle. This allows the arbiter to arbitrate any pending requests, and grant the bus to the next requester, at the same time that the active master completes its cycle.

VMEbus to Local Bus Interface

The VMEbus to Local Bus interface allows an off-board VMEbus master access to onboard resources. The VMEbus to Local Bus interface includes the V*MEbus slave*, *write post buffer*, and *local bus master*.

Adhering to the IEEE 1014-87 VMEbus Standard, the *slave* can withstand address-only cycles, as well as address pipelining, and respond to unaligned transfers. Using programmable map decoders, it can be configured to provide the following VMEbus capabilities:

Addressing capabilities:	A24, A32
Data transfer capabilities:	D08(EO), D16, D32, D8/BLT, D16/BLT, D32/BLT, D64/BLT (BLT = block transfer)

The slave can be programmed to perform *write posting* operations. When in this mode, the chip latches incoming data and addressing information into a staging FIFO and then acknowledges the VMEbus write transfer by asserting DTACK. The chip then requests control of the Local Bus and independently accesses the local resource after it has been granted the Local Bus. The write-posting pipeline is two deep in the non-block transfer mode and 16 deep in the block transfer mode.

To significantly improve the access time of the slave when it responds to a VMEbus block read cycle, the VMEchip2 contains a 16 four-byte deep read-ahead pipeline. When responding to a block read cycle, the chip performs block read cycles on the Local Bus to keep the FIFO buffer full. Data for subsequent transfers is then retrieved from the on-chip buffer, significantly improving the response time of the slave in the block transfer mode.

The VMEchip2 includes an on-chip map decoder that allows software to configure the global addressing range of onboard resources. The decoder allows the local address range to be partitioned into two separate banks, each with its own start and end address (in increments of 64KB), as well as set each bank's address modifier codes and write post enable and snoop enable.

Each map decoder includes an alternate address register and an alternate address select register. These registers allow any or all of the upper 16 VMEbus address lines to be replaced by signals from the alternate address register. This allows the address of local resources to be different from their VMEbus address.

The alternate address register also provides the upper eight bits of the local address when the VMEbus slave cycle is A24.

The *local bus master* requests the Local Bus and executes cycles as required. To reduce Local Bus loading and improve performance it always attempts to transfer data using a burst transfer as defined by the Local Bus.

When snooping is enabled, the Local Bus master requests the cache controller in the CPU to monitor the Local Bus addresses. (This action will vary by CPU.)

Local Bus to VMEbus DMA Controller

The DMA Controller (DMAC) operates in conjunction with the Local Bus master, the VMEbus master, and a 16 four-byte FIFO buffer. The DMA controller has a 32-bit local address counter, 32-bit table address counter, a 32-bit VMEbus address counter, a 32-bit byte counter, and control and status registers. The Local Control and Status Register (LCSR) provides software with the ability to control the operational modes of the DMAC. Software can program the DMAC to transfer up to 4GB of data in the course of a single DMA operation. The DMAC supports transfers from any Local Bus address to any VMEbus address. The transfers may be from one byte to 4GB in length.

To optimize Local Bus use, the DMAC automatically adjusts the size of individual data transfers until 32-bit transfers can be executed. Based on the address of the first byte, the DMAC transfers a single-byte, a double-byte, or a mixture of both, and then continues to execute quad-byte block transfer cycles. When the DMAC is set for 64-bit transfers, the octal-byte transfers takes place. Based on the address of the last byte, the DMAC transfers a single-byte, a double-byte, or a mixture of both to end the transfer.

Using control register bits in the LCSR, the DMAC can be configured to provide the following VMEbus capabilities:

Addressing capabilities: A16, A24, A32
Data transfer capabilities: D16, D32, D16/BLT, D32/BLT, D64/BLT (BLT = block transfer)

Using the DMA AM control register, the address modifier code that the VMEbus DMA controller places on the VMEbus can be programmed under software control. In addition, the DMAC can be programmed to execute block-transfer cycles over the VMEbus.

Complying with the VMEbus specification, the DMAC automatically terminates block-transfer cycles whenever a 256-byte (D32/BLT) or 2-KB (D64/BLT) boundary is crossed. It does so by momentarily releasing AS and then, in accordance with its bus release/bus request configuration, initiating a new block-transfer cycle.

To optimize VMEbus use, the DMAC automatically adjusts the size of individual data transfers until 64-bit transfers (D64/BLT mode), 32-bit transfers (D32 mode) or 16-bit transfers (D16 mode) can be executed. Based on the address of the first byte, the DMAC transfers single-byte, double-byte, or a mixture of both, and then continues to execute transfer cycles based on the programmed data width. Based on the address of the last byte, the DMAC transfers single-byte, double-byte, or a mixture of both to end the transfer.

To optimize Local Bus use when the VMEbus is operating in the D16 mode, the data FIFO converts D16 VMEbus transfers to D32 Local Bus transfers. The FIFO also aligns data if the source and destination addresses are not aligned so the Local Bus and VMEbus can operate at their maximum data transfer sizes.

To allow other boards access to the VMEbus, the DMAC has bus tenure timers to limit the time the DMAC spends on the VMEbus and to ensure a minimum time off the VMEbus. Since the Local Bus is generally faster than the VMEbus, other Local Bus masters may use the Local Bus while the DMAC is waiting for the VMEbus.

The DMAC also supports command chaining through the use of a singly-linked list built in local memory. Each entry in the list includes a VMEbus address, a Local Bus address, a byte count, a control word, and a pointer to the next entry. When the command

chaining mode is enabled, the DMAC reads and executes commands from the list in local memory until all commands are executed.

The DMAC can be programmed to send an interrupt request to the Local Bus interrupter when any specific table entry has completed. In addition the DMAC always sends an interrupt request at the normal completion of a request or when an error is detected. If the DMAC interrupt is enabled in the DMAC, the Local Bus is interrupted.

To allow increased flexibility in managing the bus tenure to optimize bus usage as required by the system configuration, the chip contains control bits that allow the DMAC time on and off the bus to be programmed. Using these control bits, software can instruct the DMA Controller to acquire the bus, maintain mastership for a specific amount of time, and then, after relinquishing it, refrain from requesting it for another specific amount of time.

DMAC VMEbus Requester

The chip contains an independent VMEbus requester associated with the DMA Controller. This allows flexibility in instituting different bus tenure policies for the single-transfer oriented master, and the block-transfer oriented DMA controller. The DMAC requester provides all the signals necessary to allow the on-chip DMA Controller to request and be granted use of the VMEbus.

Requiring no external jumpers, the chip provides the means for software to program the DMAC requester to request the bus on any one of the four bus request levels, automatically establishing the bus grant daisy-chains for the three inactive levels.

The DMAC requester requests the bus as required to transfer data to or from the FIFO buffer.

The requester implements a FAIR mode. By setting the DFAIR bit, the requester refrains from requesting the bus until it detects its assigned request line in its negated state.

The requester releases the bus when requested to by the DMA controller. The DMAC always releases the VMEbus when the FIFO is full (VMEbus to Local Bus) or empty (Local Bus to VMEbus). The DMAC can also be programmed to release the VMEbus when another VMEbus master requests the bus, when the time on timer has expired, or when the time on timer has expired and another VMEbus master is requesting the bus. To minimize the timing overhead of the arbitration process, the DMAC requester executes an early release of the bus. If it is about to release the bus and it is executing a VMEbus cycle, the requester releases BBSY before its associated VMEbus master completes the cycle. This allows the arbiter to arbitrate any pending requests, and grant the bus to the next requester, at the same time that the DMAC completes its cycle.

Tick and Watchdog Timers

The VMEchip2 has two 32-bit tick timers and a watchdog timer. The tick timers run on a 1 MHz clock which is derived from the Local Bus clock by the prescaler.

Prescaler

The prescaler is used to derive the various clocks required by the tick timers, VME access timers, reset timer, bus arbitration timer, Local Bus timer, and VMEbus timer. The prescaler divides the Local Bus clock to produce the constant-frequency clocks required. Software is required to load the appropriate constant, depending upon the Local Bus clock, following reset to ensure proper operation of the prescaler.

Tick Timer

The VMEchip2 includes two general purpose tick timers. These timers can be used to generate interrupts at various rates or the counters can be read at various times for interval timing. The timers have a resolution of 1 µs and when free running, they roll over every 71.6 minutes.

Each tick timer has a 32-bit counter, a 32-bit compare register, a 4-bit overflow register, an enable bit, an overflow clear bit, and a clear-on-compare enable bit. The counter is readable and writable at any time and when enabled in the free run mode, it increments every 1 µs. When the counter is enabled in the clear-on-compare mode, it increments every 1 µs until the counter value matches the value in the compare register. When a match occurs, the counter is cleared. When a match occurs, in either mode, an interrupt is sent to the Local Bus interrupter and the overflow counter is incremented. An interrupt to the Local Bus is only generated if the tick timer interrupt is enabled by the Local Bus interrupter. The overflow counter can be cleared by writing a one to the overflow clear bit.

Tick timer one or two can be programmed to generate a pulse on the VMEbus IRQ1 interrupt line at the tick timer period. This provides a broadcast interrupt function which allows several VME boards to receive an interrupt at the same time. In certain applications, this interrupt can be used to synchronize multiple processors. This interrupt is not acknowledged on the VMEbus. This mode is intended for specific applications and is not defined in the VMEbus specification.

Watchdog Timer

The watchdog timer has a 4-bit counter, four clock select bits, an enable bit, a local reset enable bit, a SYSRESET enable bit, a board fail enable bit, counter reset bit, WDTO status bit, and WDTO status reset bit.

When enabled, the counter increments at a rate determined by the clock select bits. If the counter is not reset by software, the counter reaches its terminal count. When this occurs, the WDTO status bit is set; and if the local or SYSRESET function is enabled, the selected reset is generated; if the board fail function is enabled, the board fail signal is generated.

VMEbus Interrupter

The interrupter provides all the signals necessary to allow software to request interrupt service from a VMEbus interrupt handler. The chip connects to all signals that a VMEbus interrupter is required to drive and monitor.

Requiring no external jumpers, the chip provides the means for software to program the interrupter to request an interrupt on any one of the seven interrupt request lines. In addition, the chip controls the propagation of the acknowledge on the IACK daisy-chain.

The interrupter operates in the release-on-acknowledge (ROAK) mode. An 8-bit control register provides software with the means to dynamically program the status/ID information. Upon reset, this register is initialized to a status/ID of $0F (the uninitialized vector in the 68K-based environment).

The VMEbus interrupter has an additional feature not defined in the VMEbus specification. The VMEchip2 supports a broadcast mode on the IRQ1 signal line. When this feature is used, the normal IRQ1 interrupt to the Local Bus interrupter should be disabled and the edge-sensitive IRQ1 interrupt to the Local Bus interrupter should be enabled. All boards in the system which are not participating in the broadcast interrupt function should not drive or respond to any signals on the IRQ1 signal line.

There are two ways to broadcast an IRQ1 interrupt. The VMEbus interrupter in the VMEchip2 may be programmed to generate a level one interrupt. This interrupt must be cleared using the interrupt clear bit in the control register because the interrupt is never acknowledged on the VMEbus. The VMEchip2 allows the output of one of the tick timers to be connected to the IRQ1 interrupt signal line on the VMEbus. When this function is enabled, a pulse appears on the IRQ1 signal line at the programmed interrupt rate of the tick timer.

VMEbus System Controller

With the exception of the optional SERCLK Driver and the Power Monitor, the chip includes all the functions that a VMEbus System Controller must provide. The System Controller is enabled/disabled with the aid of an external jumper.

Arbiter

The arbitration algorithm used by the chip arbiter is selected by software. All three arbitration modes defined in the VMEbus Specification are supported: Priority (PRI), Round-Robin-Select (RRS), as well as Single (SGL). When operating in the PRI mode, the arbiter asserts the BCLR line whenever it detects a request for the bus whose level is higher that the one being serviced.

The chip includes an arbitration timer, preventing a bus lock-up when no requester assumes control of the bus after the arbiter has issued a grant. Using a control bit, this timer can be enabled or disabled. When enabled, it assumes control of the bus by driving the BBSY signal after 256 µseconds, releasing it after satisfying the requirements of the VMEbus specification, and then re-arbitrating any pending bus requests.

IACK Daisy-Chain Driver

Complying with the latest revision of the VMEbus specification, the System Controller includes an IACK Daisy-Chain Driver, ensuring that the timing requirements of the IACK daisy-chain are satisfied.

Bus Timer

The Bus Timer is enabled/disabled by software to terminate a VMEbus cycle by asserting BERR if any of the VMEbus data strobes is maintained in its asserted state for longer than the programmed time-out period. The time-out period can be set to 8, 64, or 256 µsecs. The bus timer terminates an unresponded VMEbus cycle only if both it and the system controller are enabled.

In addition to the VMEbus timer, the chip contains a Local Bus timer. This timer asserts the local TEA when the Local Bus cycle maintained in its asserted state for longer that the programmed time-out period. This timer can be enabled or disabled under software control. The time-out period can be programmed for 8, 64, or 256 µsecs.

Reset Driver

The chip includes both a global and a local reset driver. When the chip operates as the VMEbus system controller, the reset driver provides a global system reset by asserting the VMEbus signal SYSRESET. A SYSRESET may be generated by the RESET switch, a power up reset, a watch dog time-out, or by a control bit in the LCSR. SYSRESET remains asserted for at least 200 msec, as required by the VMEbus specification.

Similarly, the chip provides an input signal and a control bit to initiate a local reset operation. By setting a control bit, software can maintain a board in a reset state, disabling a faulty board from participating in normal system operation. The local reset driver is enabled even when the chip is not the system controller. A local reset may be generated by the RESET switch, a power up reset, a watch dog time-out, a VMEbus SYSRESET, or a control bit in the GCSR.

Local Bus Interrupter and Interrupt Handler

There are 31 interrupt sources in the VMEchip2, as defined in Table 4-4, *Local Bus Interrupter Summary*. Each of the 31 interrupts can be enabled to generate a Local Bus interrupt at any level. For example, VMEbus IRQ5 can be programmed to generate a level 2 Local Bus interrupt.

The VMEbus AC fail interrupter is an edge-sensitive interrupter connected to the VMEbus ACFAIL signal line. This interrupter is filtered to remove the ACFAIL glitch which is related to the BBSY glitch.

The ABORT switch interrupter is an edge-sensitive interrupter connected to the ABORT switch. This interrupter is filtered to remove switch bounce.

The SYS fail interrupter is an edge-sensitive interrupter connected to the VMEbus SYSFAIL signal line.

The write post bus error interrupter is an edge-sensitive interrupter connected to the Local Bus to VMEbus write post bus error signal line.

The external interrupter is an edge-sensitive interrupter connected to a signal pin on the VMEchip2.

The VMEbus IRQ1 edge-sensitive interrupter is an edge-sensitive interrupter connected to the VMEbus IRQ1 signal line. This interrupter is used when one of the tick timers is connected to the IRQ1 signal line. When this interrupt is acknowledged, the vector is provided by the VMEchip2 and a VMEbus interrupt acknowledge is not generated. When this interrupt is enabled, the VMEbus IRQ1 level-sensitive interrupter should be disabled.

The VMEchip2 VMEbus interrupter acknowledge interrupter is an edge-sensitive interrupter connected to the acknowledge output of the VMEbus interrupter. An interrupt is generated when an interrupt on the VMEbus from VMEchip2 is acknowledged by a VMEbus interrupt handler.

The tick timer interrupters are edge-sensitive interrupters connected to the output of the tick timers.

The DMAC interrupter is an edge-sensitive interrupter connected to the DMAC.

The GCSR SIG3-0 interrupters are edge-sensitive interrupters connected to the output of the signal bits in the GCSR.

The location monitor interrupters are edge-sensitive interrupters connected to the location monitor bits in the GCSR.

The software 7-0 interrupters can be set by software to generate interrupts.

The VMEbus IRQ7-1 interrupters are level-sensitive interrupters connected to the VMEbus IRQ7-1 signal lines.

The interrupt handler provides all logic necessary to identify and handle all local interrupts as well as VMEbus interrupts. When a local interrupt is acknowledged, a unique vector is provided by the chip. Edge-sensitive interrupters are not cleared during the interrupt acknowledge cycle and must by reset by software as required. If the interrupt source is the VMEbus, the interrupt handler instructs the VMEbus master to execute a VMEbus IACK cycle to obtain the vector from the VMEbus interrupter. The chip connects to all signals that a VMEbus handler is required to drive and monitor. On the Local Bus, the interrupt handler is designed to comply with the interrupt handling signaling protocol of the Local Bus.

Global Control and Status Registers

The VMEchip2 includes a set of registers that are accessible from both the VMEbus and the Local Bus. These registers are provided to aid in interprocessor communications over the VMEbus. These registers are fully described in a later section.

VMEboard Functions

The VMEchip2 also includes several functions that are generally used on VMEbus boards. The VMEchip2 includes a Local Bus map decoder and control logic for EPROM and Flash memory. (Chapter 1, *Programming Issues*, discusses the addressing of EPROM and Flash memory on each Single Board Computer.)

The VMEchip2 provides a Local Bus map decoder and control for static RAM. The SRAM space is 1MB and it is located at Local Bus address $FFE00000 to $FFEFFFFF.

The VMEchip2 provides eight general purpose input signal pins and four general purpose I/O pins.

LCSR Programming Model

This section defines the programming model for the Local Control and Status Registers (LCSR) in the VMEchip2. The Local Bus map decoder for the LCSR is included in the VMEchip2. The base address of the LCSR is $FFF40000 and the registers are 32 bits wide. Byte, two-byte and four-byte read operations are permitted: however, byte and two-byte write operations are not permitted. Byte and two-byte write operations return a TEA signal to the Local Bus. Read-modify-write operations should be used to modify a byte or a two-byte of a register.

Each register definition includes a table with 5 lines:

- Line 1 is the base address of the register and the number of bits defined in the table.
- Line 2 shows the bits defined by this table.
- Line 3 defines the name of the register or the name of the bits in the register.
- Line 4 defines the operations possible on the register bits as follows:

R	This bit is a read-only status bit.
R/W	This bit is readable and writable.
W/AC	This bit can be set and it is automatically cleared. This bit can also be read.
C	Writing a one to this bit clears this bit or another bit. This bit reads zero.
S	Writing a one to this bit sets this bit or another bit. This bit reads zero.

- Line 5 defines the state of the bit following a reset as follows:

P	The bit is affected by power-up reset.
S	The bit is affected by SYSRESET.
L	The bit is affected by local reset.
X	The bit is not affected by reset.

A summary of the LCSR is shown in Table 4-1 and Table 4-2.

Table 4-1. VMEchip2 Memory Map - LCSR Summary (Sheet 1 of 2)

VMEchip2 LCSR Base Address = $FFF40000

OFFSET:	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	SLAVE ENDING ADDRESS 1															
4	SLAVE ENDING ADDRESS 2															
8	SLAVE ADDRESS TRANSLATION ADDRESS 1															
C	SLAVE ADDRESS TRANSLATION ADDRESS 2															
10					ADDER 2	SNP 2		WP 2	SUP 2	USR 2	A32 2	A24 2	BLK D64 2	BLK 2	PRGM 2	DATA2
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
14	MASTER ENDING ADDRESS 1															
18	MASTER ENDING ADDRESS 2															
1C	MASTER ENDING ADDRESS 3															
20	MASTER ENDING ADDRESS 4															
24	MASTER ADDRESS TRANSLATION ADDRESS 4															
28	MAST D16 EN	MAST WP EN	MASTER AM 4						MAST D16 EN	MAST WP EN	MASTER AM 3					
2C	GCSR GROUP SELECT								GCSR BOARD SELECT				MAST 4 EN	MAST 3 EN	MAST 2 EN	MAST 1 EN
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
30											WAIT RMW	ROM ZERO	DMA TB SNP MODE		SRAM SPEED	
34																
38	DMA CONTROLLER															
3C	DMA CONTROLLER															
40	DMA CONTROLLER															
44	DMA CONTROLLER															
48		TICK 2/1	TICK IRQ 1 EN	CLR IRQ	IRQ STAT	VMEBUS INTERRUPT LEVEL			VMEBUS INTERRUPT VECTOR							

This sheet continues on facing page. ⟶

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SLAVE STARTING ADDRESS 1															
SLAVE STARTING ADDRESS 2															
SLAVE ADDRESS TRANSLATION SELECT 1															
SLAVE ADDRESS TRANSLATION SELECT 2															
				ADDER 1	SNP 1			WP 1	SUP 1	USR 1	A32 1	A24 1	BLK D64 1	BLK 1	PRGM 1	DATA1
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MASTER STARTING ADDRESS 1															
MASTER STARTING ADDRESS 2															
MASTER STARTING ADDRESS 3															
MASTER STARTING ADDRESS 4															
MASTER ADDRESS TRANSLATION SELECT 4															
MAST D16 EN	MAST WP EN	MASTER AM 2						MAST D16 EN	MAST WP EN	MASTER AM 1					
IO2 EN	IO2 WP EN	IO2 S/U	IO2 P/D	IO1 EN	IO1 D16 EN	IO1 WP EN	IO1 S/U	ROM SIZE		ROM BANK B SPEED			ROM BANK A SPEED		
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ARB ROBN	MAST DHB	MAST DWB		MST FAIR	MST RWD	MASTER VMEBUS		DMA HALT	DMA EN	DMA TBL	DMA FAIR	DM RELM		DMA VMEBUS	
DMA TBL INT	DMA LB SNP MODE			DMA INC VME	DMA INC LB	DMA WRT	DMA D16	DMA D64 BLK	DMA BLK	DMA AM 5	DMA AM 4	DMA AM 3	DMA AM 2	DMA AM 1	DMA AM 0
LOCAL BUS ADDRESS COUNTER															
VMEBUS ADDRESS COUNTER															
BYTE COUNTER															
TABLE ADDRESS COUNTER															
DMA TABLE INTERRUPT COUNT				MPU CLR STAT	MPU LBE ERR	MPU LPE ERR	MPU LOB ERR	MPU LTO ERR	DMA LBE ERR	DMA LPE ERR	DMA LOB ERR	DMA LTO ERR	DMA TBL ERR	DMA VME ERR	DMA DONE

1360 9403

← This sheet begins on facing page.

Table 4-2. VMEchip2 Memory Map - LCSR Summary (Sheet 2 of 2)

VMEchip2 LCSR Base Address = $FFF40000
OFFSET:

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
4C								ARB BGTO EN	DMA TIME OFF			DMA TIME ON			VME GLOBAL TIMER	
50	TICK TIMER 1															
54	TICK TIMER 1															
58	TICK TIMER 2															
5C	TICK TIMER 2															
60		SCON	SYS FAIL	BRD FAIL STAT	PURS STAT	CLR PURS STAT	BRD FAIL OUT	RST SW EN	SYS RST	WD CLR TO	WD CLR CNT	WD TO STAT	TO BF EN	WD SRST LRST	WD RST EN	WD EN
64	PRE															
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
68	AC FAIL IRQ	AB IRQ	SYS FAIL IRQ	MWP BERR IRQ	PE IRQ	IRQ1E IRQ	TIC2 IRQ	TIC1 IRQ	VME IACK IRQ	DMA IRQ	SIG3 IRQ	SIG2 IRQ	SIG1 IRQ	SIG0 IRQ	LM1 IRQ	LM0 IRQ
6C	EN IRQ 31	EN IRQ 30	EN IRQ 29	EN IRQ 28	EN IRQ 27	EN IRQ 26	EN IRQ 25	EN IRQ 24	EN IRQ 23	EN IRQ 22	EN IRQ 21	EN IRQ 20	EN IRQ 19	EN IRQ 18	EN IRQ 17	EN IRQ 16
70																
74	CLR IRQ 31	CLR IRQ 30	CLR IRQ 29	CLR IRQ 28	CLR IRQ 27	CLR IRQ 26	CLR IRQ 25	CLR IRQ 24	CLR IRQ 23	CLR IRQ 22	CLR IRQ 21	CLR IRQ 20	CLR IRQ 19	CLR IRQ 18	CLR IRQ 17	CLR IRQ 16
78		AC FAIL IRQ LEVEL				ABORT IRQ LEVEL				SYS FAIL IRQ LEVEL				MST WP ERROR IRQ LEVEL		
7C		VME IACK IRQ LEVEL				DMA IRQ LEVEL				SIG 3 IRQ LEVEL				SIG 2 IRQ LEVEL		
80		SW7 IRQ LEVEL				SW6 IRQ LEVEL				SW5 IRQ LEVEL				SW4 IRQ LEVEL		
84		SPARE IRQ LEVEL				VME IRQ 7 IRQ LEVEL				VME IRQ 6 IRQ LEVEL				VME IRQ 5 IRQ LEVEL		
88	VECTOR BASE REGISTER 0				VECTOR BASE REGISTER 1				MST IRQ EN	SYS FAIL LEVEL	AC FAIL LEVEL	ABORT LEVEL	GPIOEN			
8C																

This sheet continues on facing page. ⟶

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VME ACCESS TIMER		LOCAL BUS TIMER		WD TIME OUT SELECT				PRESCALER CLOCK ADJUST							
COMPARE REGISTER															
COUNTER															
COMPARE REGISTER															
COUNTER															
OVERFLOW COUNTER 2				X	CLR OVF 2	COC EN 2	TIC EN 2	OVERFLOW COUNTER 1				X	CLR OVF 1	COC EN 1	TIC EN 1
SCALER															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SW7 IRQ	SW6 IRQ	SW5 IRQ	SW4 IRQ	SW3 IRQ	SW2 IRQ	SW1 IRQ	SW0 IRQ	SPARE	VME IRQ7	VME IRQ6	VME IRQ5	VME IRQ4	VME IRQ3	VME IRQ2	VME IRQ1
EN IRQ 15	EN IRQ 14	EN IRQ 13	EN IRQ 12	EN IRQ 11	EN IRQ 10	EN IRQ 9	EN IRQ 8	EN IRQ 7	EN IRQ 6	EN IRQ 5	EN IRQ 4	EN IRQ 3	EN IRQ 2	EN IRQ 1	EN IRQ 0
SET IRQ 15	SET IRQ 14	SET IRQ 13	SET IRQ 12	SET IRQ 11	SET IRQ 10	SET IRQ 9	SET IRQ 8	X							
CLR IRQ 15	CLR IRQ 14	CLR IRQ 13	CLR IRQ 12	CLR IRQ 11	CLR IRQ 10	CLR IRQ 9	CLR IRQ 8	X							
X	P ERROR IRQ LEVEL			X	IRQ1E IRQ LEVEL			X	TIC TIMER 2 IRQ LEVEL			X	TIC TIMER 1 IRQ LEVEL		
X	SIG 1 IRQ LEVEL			X	SIG 0 IRQ LEVEL			X	LM 1 IRQ LEVEL			X	LM 0 IRQ LEVEL		
X	SW3 IRQ LEVEL			X	SW2 IRQ LEVEL			X	SW1 IRQ LEVEL			X	SW0 IRQ LEVEL		
X	VME IRQ 4 IRQ LEVEL			X	VMEB IRQ 3 IRQ LEVEL			X	VME IRQ 2 IRQ LEVEL			X	VME IRQ 1 IRQ LEVEL		
GPIOO				GPIOI				GPI							
X								MP IRQ EN	REV EROM	DIS SRAM	DIS MST	NO EL BBSY	DIS BSYT	EN INT	DIS BGN

1361 9403

← This sheet begins on facing page.

Programming the VMEbus Slave Map Decoders

This section includes programming information for the VMEbus to Local Bus map decoders.

The VMEbus to Local Bus interface allows off-board VMEbus masters access to local onboard resources. The address of the local resources as viewed from the VMEbus is controlled by the VMEbus slave map decoders, which are part of the VMEbus to Local Bus interface. Two VMEbus slave map decoders in the VMEchip2 allow two segments of the VMEbus to be mapped to the Local Bus. A segment may vary in size from 64KB to 4GB in increments of 64KB. Address translation is provided by the address translation registers which allow the upper 16 bits of the Local Bus address to be provided by the address translation address register rather than the upper 16 bits of the VMEbus.

Each VMEbus slave map decoder has the following registers: *address translation address register, address translation select register*, *starting address register, ending address register*, *address modifier select register*, and *attribute register*. The addresses and bit definitions of these registers are shown in the following tables.

The VMEbus slave map decoders described in this section are disabled by local reset, SYSRESET, or power-up reset. Caution must be used when enabling the map decoders or when modifying their registers after they are enabled. The safest time to enable or modify the map decoder registers is when the VMEchip2 is VMEbus master. The following procedure should be used to modify the map decoder registers: Set the DWB bit in the LCSR and then wait for the DHB bit in the LCSR to be set, indicating that VMEbus mastership has been acquired. The map decoder registers can then be modified and the VMEbus released by clearing the DWB bit in the LCSR. Because the VMEbus is held during this programming operation, the registers should be programmed quickly with interrupts disabled.

The VMEbus slave map decoders can be programmed, without obtaining VMEbus mastership, if they are disabled and the following procedure is followed: The address translation registers

and starting and ending address registers should be programmed first, and then the map decoders should be enabled by programming the address modifier select registers.

A VMEbus slave map decoder is programmed by loading the starting address of the segment into the *starting address register* and the ending address of the segment into the *ending address register*. If the VMEbus address modifier codes indicate an A24 VMEbus address cycle, then the upper eight bits of the VMEbus address are forced to zero before the compare. The address modifier select register should be programmed for the required address modifier codes. A VMEbus slave map decoder is disabled when the address modifier select register is cleared.

The *address translation registers* allow local resources to have different VMEbus and Local Bus addresses. Only address bits A31 through A16 may be modified.

The *address translation registers* also provide the upper eight Local Bus address lines when an A24 VMEbus cycle is used to accesses a local resource. The address translation register should be programmed with the translated address and the address translation select register should be programmed to enable the translated address. If address translation is not desired, then the address translation registers should be programmed to zero.

The *address translation address register* and the *address translation select register* operate in the following way: If a bit in the address translation select register is set, then the corresponding Local Bus address line is driven from the corresponding bit in the address translation address register. If the bit is cleared in the address translation select register, then the corresponding Local Bus address line is driven from the corresponding VMEbus address line. The most significant bit of the address translation select register corresponds to the most significant bit of address translation register and to A32 of the Local Bus and A32 of the VMEbus.

In addition to the address translation method previously described, the VMEchip2 used on the Single Board Computers includes an adder which can be used for address translation. When the adder is enabled, the Local Bus address is generated by adding the offset value to the VMEbus address lines VA<31..16>. The offset is the

value in the address translation/offset register. If the VMEbus transfer is A24, then the VMEbus address lines VA<31..24> are forced to 0 before the add. The adders are enable by setting bit 11 for map decoder 1 and bit 27 for map decoder 2 in register $FFF40010. The adders allow any size board to be mapped on any 64KB boundary. The adders are disabled and the address replacement method is used following reset.

Write posting is enabled for the segment by setting the write post enable bit in the *attribute register*. Local bus snooping for the segment is enabled by setting the snoop bits in the attribute register. The snoop bits in the attribute register are driven on to the Local Bus when the VMEbus to Local Bus interface is Local Bus master.

VMEbus Slave Ending Address Register 1

ADR/SIZ	$FFF40000 (16 bits of 32)		
BIT	31	. . .	16
NAME	Ending Address Register 1		
OPER	R/W		
RESET	0 PS		

This register is the ending address register for the first VMEbus to Local Bus map decoder.

VMEbus Slave Starting Address Register 1

ADR/SIZ	$FFF40000 (16 bits of 32)		
BIT	15	. . .	0
NAME	Starting Address Register 1		
OPER	R/W		
RESET	0 PS		

This register is the starting address register for the first VMEbus to Local Bus map decoder.

VMEbus Slave Ending Address Register 2

ADR/SIZ	$FFF40004 (16 bits of 32)		
BIT	31	. . .	16
NAME	Ending Address Register 2		
OPER	R/W		
RESET	0 PS		

This register is the ending address register for the second VMEbus to Local Bus map decoder.

VMEbus Slave Starting Address Register 2

ADR/SIZ	$FFF40004 (16 bits of 32)		
BIT	15	. . .	0
NAME	Starting Address Register 2		
OPER	R/W		
RESET	0 PS		

This register is the starting address register for the second VMEbus to Local Bus map decoder.

VMEbus Slave Address Translation Address Offset Register 1

ADR/SIZ	$FFF40008 (16 bits of 32)		
BIT	31	. . .	16
NAME	Address Translation Address Offset Register 1		
OPER	R/W		
RESET	0 PS		

This register is the address translation address register for the first VMEbus to Local Bus map decoder. It should be programmed to the Local Bus starting address. When the adder is engaged, this register is the offset value.

VMEbus Slave Address Translation Select Register 1

ADR/SIZ	$FFF40008 (16 bits of 32)		
BIT	15	. . .	0
NAME	Address Translation Select Register 1		
OPER	R/W		
RESET	0 PS		

This register is the address translation select register for the first VMEbus to Local Bus map decoder. The address translation select register value is based on the segment size (the difference between the VMEbus starting and ending addresses).

If the segment size is between the sizes shown in the table below, assume the larger size.

Segment Size	Address Translation Select Value	Segment Size	Address Translation Select Value
64KB	FFFF	32MB	FE00
128KB	FFFE	64MB	FC00
256KB	FFFC	128MB	F800
512KB	FFF8	256MB	F000
1MB	FFF0	512MB	E000
2MB	FFE0	1GB	C000
4MB	FFC0	2GB	8000
8MB	FF80	4GB	0000
16MB	FF00		

VMEbus Slave Address Translation Address Offset Register 2

ADR/SIZ	$FFF4000C (16 bits of 32)		
BIT	31	. . .	16
NAME	Address Translation Address Offset Register 2		
OPER	R/W		
RESET	0 PS		

This register is the address translation address register for the second VMEbus to Local Bus map decoder. It should be programmed to the Local Bus starting address. When the adder is enabled, this register is the offset value.

VMEbus Slave Address Translation Select Register 2

ADR/SIZ	$FFF4000C (16 bits of 32)		
BIT	15	. . .	0
NAME	Address Translation Select Register 2		
OPER	R/W		
RESET	0 PS		

This register is the address translation select register for the second VMEbus to Local Bus map decoder. The address translation select register value is based on the segment size (the difference between the VMEbus starting and ending addresses). If the segment size is between the sizes shown in the table below, assume the larger size.

Segment Size	Address Translation Select Value	Segment Size	Address Translation Select Value
64KB	FFFF	32MB	FE00
128KB	FFFE	64MB	FC00
256KB	FFFC	128MB	F800
512KB	FFF8	256MB	F000
1MB	FFF0	512MB	E000
2MB	FFE0	1GB	C000
4MB	FFC0	2GB	8000
8MB	FF80	4GB	0000
16MB	FF00		

VMEbus Slave Write Post and Snoop Control Register 2

ADR/SIZ	$FFF40010 (8 bits [4 used] of 32)							
BIT	31	30	29	28	27	26	25	24
NAME					ADDER2	SNP2		WP2
OPER					R/W	R/W		R/W
RESET					0 PS	0 PS		0 PS

This register is the slave write post and snoop control register for the second VMEbus to Local Bus map decoder.

WP2 When this bit is high, write posting is enabled for the address range defined by the second VMEbus slave map decoder. When this bit is low, write posting is disabled for the address range defined by the second VMEbus slave map decoder.

SNP2 These bits control the snoop enable lines to the Local Bus for the address range defined by the second VMEbus slave map decoder. These bits must be 0 on the MVME187. The snooping functions are:

- **0** Snoop inhibited
- **1** Write - Sink data
  Read - Supply dirty data and leave dirty

  This bit must be 0 on the MVME176/177
- **2** Write - Invalidate
  Read - Supply dirty data and mark invalid
- **3** Snoop inhibited

ADDER2 When this bit is high, the adder is used for address translation. When this bit is low, the adder is not used for address translation.

VMEbus Slave Address Modifier Select Register 2

ADR/SIZ	$FFF40010 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
NAME	SUP	USR	A32	A24	D64	BLK	PGM	DAT
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the address modifier select register for the second VMEbus to Local Bus map decoder. There are three groups of address modifier select bits: DAT, PGM, BLK and D64; A24 and A32; and USR and SUP. At least one bit must be set from each group to enable the map decoder.

DAT When this bit is high, the second map decoder responds to VMEbus data access cycles. When this bit is low, the second map decoder does not respond to VMEbus data access cycles.

PGM When this bit is high, the second map decoder responds to VMEbus program access cycles. When this bit is low, the second map decoder does not respond to VMEbus program access cycles.

BLK When this bit is high, the second map decoder responds to VMEbus block access cycles. When this bit is low, the second map decoder does not respond to VMEbus block access cycles.

D64 When this bit is high, the second map decoder responds to VMEbus D64 block access cycles. When this bit is low, the second map decoder does not respond to VMEbus D64 block access cycles.

A24 When this bit is high, the second map decoder responds to VMEbus A24 (standard) access cycles. When this bit is low, the second map decoder does not respond to VMEbus A24 access cycles.

A32 When this bit is high, the second map decoder responds to VMEbus A32 (extended) access cycles. When this bit is low, the second map decoder does not respond to VMEbus A32 access cycles.

USR When this bit is high, the second map decoder responds to VMEbus user (non-privileged) access cycles. When this bit is low, the second map decoder does not respond to VMEbus user access cycles.

SUP When this bit is high, the second map decoder responds to VMEbus supervisory access cycles. When this bit is low, the second map decoder does not respond to VMEbus supervisory access cycles.

VMEbus Slave Write Post and Snoop Control Register 1

ADR/SIZ	$FFF40010 (8 bits [4 used] of 32)							
BIT	15	14	13	12	11	10	9	8
NAME					ADDER1	SNP1		WP1
OPER					R/W	R/W		R/W
RESET					0 PS	0 PS		0 PS

This register is the slave write post and snoop control register for the first VMEbus to Local Bus map decoder.

WP1 When this bit is high, write posting is enabled for the address range defined by the first VMEbus slave map decoder. When this bit is low, write posting is disabled for the address range defined by the first VMEbus slave map decoder.

SNP1 These bits control the snoop enable lines to the Local Bus for the address range defined by the first VMEbus slave map decoder. These bits must be 0 on the MVME187. The snooping functions are:

0 Snoop inhibited

1 Write - Sink data
Read - Supply dirty data and leave dirty

This bit must be 0 on the MVME176/177.

2 Write - Invalidate
Read - Supply dirty data and mark invalid

3 Snoop inhibited

ADDER1 When this bit is high, the adder is used for address translation. When this bit is low, the adder is not used for address translation.

VMEbus Slave Address Modifier Select Register 1

ADR/SIZ	$FFF40010 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
NAME	SUP	USR	A32	A24	D64	BLK	PGM	DAT
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the address modifier select register for the first VMEbus to Local Bus map decoder. There are three groups of address modifier select bits: DAT, PGM, BLK and D64; A24 and A32; and USR and SUP. At least one bit must be set from each group to enable the first map decoder.

DAT When this bit is high, the first map decoder responds to VMEbus data access cycles. When this bit is low, the first map decoder does not responded to VMEbus data access cycles.

PGM When this bit is high, the first map decoder responds to VMEbus program access cycles. When this bit is low, the first map decoder does not respond to VMEbus program access cycles.

BLK When this bit is high, the first map decoder responds to VMEbus block access cycles. When this bit is low, the first map decoder does not respond to VMEbus block access cycles.

D64 When this bit is high, the first map decoder responds to VMEbus D64 block access cycles. When this bit is low, the first map decoder does not respond to VMEbus D64 block access cycles.

A24 When this bit is high, the first map decoder responds to VMEbus A24 (standard) access cycles. When this bit is low, the first map decoder does not respond to VMEbus A24 access cycles.

A32 When this bit is high, the first map decoder responds to VMEbus A32 (extended) access cycles. When this bit is low, the first map decoder does not respond to VMEbus A32 access cycles.

USR When this bit is high, the first map decoder responds to VMEbus user (non-privileged) access cycles. When this bit is low, the first map decoder does not respond to VMEbus user access cycles.

SUP When this bit is high, the first map decoder responds to VMEbus supervisory access cycles. When this bit is low, the first map decoder does not respond to VMEbus supervisory access cycles.

Programming the Local Bus to VMEbus Map Decoders

This section includes programming information on the Local Bus to VMEbus map decoders and the GCSR base address registers.

The Local Bus to VMEbus interface allows onboard Local Bus masters access to off-board VMEbus resources. The address of the VMEbus resources as viewed from the Local Bus is controlled by the Local Bus slave map decoders, which are part of the Local Bus

to VMEbus interface. Four of the six Local Bus to VMEbus map decoders are programmable, while the two I/O map decoders are fixed, as follows:

first I/O map decoder	A16/D16 or A16/D32 space at $FFFF0000 to $FFFFFFFF which is the VMEbus short I/O space
second I/O map decoder	A24/D16 space at $F000000 to $F0FFFFFF and A32/D16 space at $F1000000 to $FF7FFFFF

A programmable segment may vary in size from 64KB to 4GB in increments of 64KB. Address translation for the fourth segment is provided by the address translation registers which allow the upper 16 bits of the VMEbus address to be provided by the address translation address register rather than the upper 16 bits of the Local Bus.

Each of the four programmable Local Bus map decoders has a starting address, an ending address, an address modifier register with attribute bits, and an enable bit. The fourth decoder also has address translation registers. The addresses and bit definitions for these registers are in the tables below.

A Local Bus slave map decoder is programmed by loading the starting address of the segment into the starting address register and the ending address of the segment into the ending address register. The address modifier code is programmed in to the address modifier register. Because the Local Bus to VMEbus interface does not support VMEbus block transfers, block transfer address modifier codes should not be programmed.

The address translation register allows a Local Bus master to view a portion of the VMEbus that may be hidden by onboard resources or an area of the VMEbus may be mapped to two local address. For example, some devices in the I/O map may support write posting while others do not. The VMEbus area in question may be mapped to two Local Bus addresses, one with write posting enabled and one with write posting disabled. The address translation registers allow Local Bus address bits A31 through A16 to be modified. The address translation register should be programmed with the translated address, and the address translation select register

should be programmed to enable the translated address. If address translation is not desired, then the address translation registers should be programmed to zero.

The address translation address register and the address translation select register operate in the following way. If a bit in the address translation select register is set, then the corresponding VMEbus address line is driven from the corresponding bit in the address translation address register. If the bit is cleared in the address translation select register, then the corresponding VMEbus address line is driven from the corresponding Local Bus address line. The most significant bit of the address translation select register corresponds to the most significant bit of address translation address register and to A32 of the Local Bus and A32 of the VMEbus.

Write posting is enabled for the segment by setting the write post enable bit in the address modifier register. D16 transfers are forced by setting the D16 bit in the address modifier register. A segment is enabled by setting the enable bit. Segments should not be programmed to overlap.

The first I/O map decoder maps the Local Bus address range $FFFF0000 to $FFFFFFFF to the A16 (short I/O) map of the VMEbus. This segment may be enabled using the enable bit. Write posting may be enabled for this segment using the write post enable bit. The transfer size may be D16 or D32 as defined by the D16 bit in the control register.

The second I/O map decoder provides support for the other I/O map of the VMEbus. This decoder maps the Local Bus address range $F0000000 to $F0FFFFFF to the A24 map of the VMEbus and the address range $F1000000 to $FF7FFFFF to the A32 map of the VMEbus. The transfer size is always D16. This segment may be enabled using the enable bit. Write posting may be enabled using the write post enable bit.

The Local Bus map decoders should not be programmed such that more than one map decoder responds to the same Local Bus address or a map decoder conflicts with on board resources. However, the map decoders may be programmed to allow a VMEbus address to be accessed from more than one Local Bus address.

Local Bus Slave (VMEbus Master) Ending Address Register 1

ADR/SIZ	$FFF40014 (16 bits of 32)		
BIT	31	. . .	16
NAME	Ending Address Register 1		
OPER	R/W		
RESET	0 PS		

This register is the ending address register for the first Local Bus to VMEbus map decoder.

Local Bus Slave (VMEbus Master) Starting Address Register 1

ADR/SIZ	$FFF40014 (16 bits of 32)		
BIT	15	. . .	0
NAME	Starting Address Register 1		
OPER	R/W		
RESET	0 PS		

This register is the starting address register for the first Local Bus to VMEbus map decoder.

Local Bus Slave (VMEbus Master) Ending Address Register 2

ADR/SIZ	$FFF40018 (16 bits of 32)		
BIT	31	. . .	16
NAME	Ending Address Register 2		
OPER	R/W		
RESET	0 PS		

This register is the ending address register for the second Local Bus to VMEbus map decoder.

Local Bus Slave (VMEbus Master) Starting Address Register 2

ADR/SIZ	$FFF40018 (16 bits of 32)		
BIT	15	. . .	0
NAME	Starting Address Register 2		
OPER	R/W		
RESET	0 PS		

This register is the starting address register for the second Local Bus to VMEbus map decoder.

Local Bus Slave (VMEbus Master) Ending Address Register 3

ADR/SIZ	$FFF4001C (16 bits of 32)		
BIT	31	. . .	16
NAME	Ending Address Register 3		
OPER	R/W		
RESET	0 PS		

This register is the ending address register for the third Local Bus to VMEbus map decoder.

Local Bus Slave (VMEbus Master) Starting Address Register 3

ADR/SIZ	$FFF4001C (16 bits of 32)		
BIT	15	. . .	0
NAME	Starting Address Register 3		
OPER	R/W		
RESET	0 PS		

This register is the starting address register for the third Local Bus to VMEbus map decoder.

Local Bus Slave (VMEbus Master) Ending Address Register 4

ADR/SIZ	$FFF40020 (16 bits of 32)		
BIT	31	. . .	16
NAME	Ending Address Register 4		
OPER	R/W		
RESET	0 PS		

This register is the ending address register for the fourth Local Bus to VMEbus map decoder.

Local Bus Slave (VMEbus Master) Starting Address Register 4

ADR/SIZ	$FFF40020 (16 bits of 32)		
BIT	15	. . .	0
NAME	Starting Address Register 4		
OPER	R/W		
RESET	0 PS		

This register is the starting address register for the fourth Local Bus to VMEbus map decoder.

Local Bus Slave (VMEbus Master) Address Translation Address Register 4

ADR/SIZ	$FFF40024 (16 bits of 32)		
BIT	31	. . .	16
NAME	Address Translation Address Register 4		
OPER	R/W		
RESET	0 PS		

This register is the address translation address register for the fourth Local Bus to VMEbus bus map decoder.

Local Bus Slave (VMEbus Master) Address Translation Select Register 4

ADR/SIZ	$FFF40024 (16 bits of 32)		
BIT	15	. . .	0
NAME	Address Translation Select Register 4		
OPER	R/W		
RESET	0 PS		

This register is the address translation select register for the fourth Local Bus to VMEbus bus map decoder.

Local Bus Slave (VMEbus Master) Attribute Register 4

ADR/SIZ	$FFF40028 (8 bits of 32)							
BIT	31	30	29	28	27	26	25	24
NAME	D16	WP	AM					
OPER	R/W	R/W	R/W					
RESET	0 PS	0 PS	0 PS					

This register is the attribute register for the fourth Local Bus to VMEbus bus map decoder.

AM These bits define the VMEbus address modifier codes the VMEbus master uses for the segment defined by map decoder 4. Because the Local Bus to VMEbus interface does not support block transfers, the block transfer address modifier codes should not be used.

WP When this bit is high, write posting is enabled to the segment defined by map decoder 4. When this bit is low, write posting is disabled to the segment defined by map decoder 4.

D16 When this bit is high, D16 data transfers are performed to the segment defined by map decoder 4. When this bit is low, D32 data transfers are performed to the segment defined by map decoder 4.

Local Bus Slave (VMEbus Master) Attribute Register 3

ADR/SIZ	$FFF40028 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
NAME	D16	WP	AM					
OPER	R/W	R/W	R/W					
RESET	0 PS	0 PS	0 PS					

This register is the attribute register for the third Local Bus to VMEbus bus map decoder.

AM These bits define the VMEbus address modifier codes the VMEbus master uses for the segment defined by map decoder 3. Because the Local Bus to VMEbus interface does not support block transfers, the block transfer address modifier codes should not be used.

WP When this bit is high, write posting is enabled to the segment defined by map decoder 3. When this bit is low, write posting is disabled to the segment defined by map decoder 3.

D16 When this bit is high, D16 data transfers are performed to the segment defined by map decoder 3. When this bit is low, D32 data transfers are performed to the segment defined by map decoder 3.

Local Bus Slave (VMEbus Master) Attribute Register 2

ADR/SIZ	$FFF40028 (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	D16	WP	AM					
OPER	R/W	R/W	R/W					
RESET	0 PS	0 PS	0 PS					

This register is the attribute register for the second Local Bus to VMEbus bus map decoder.

AM These bits define the VMEbus address modifier codes the VMEbus master uses for the segment defined by map decoder 2. Since the Local Bus to VMEbus interface does not support block transfers, the block transfer address modifier codes should not be used.

WP When this bit is high, write posting is enabled to the segment defined by map decoder 2. When this bit is low, write posting is disabled to the segment defined by map decoder 2.

D16 When this bit is high, D16 data transfers are performed to the segment defined by map decoder 2. When this bit is low, D32 data transfers are performed to the segment defined by map decoder 2.

Local Bus Slave (VMEbus Master) Attribute Register 1

ADR/SIZ	$FFF40028 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
NAME	D16	WP	AM					
OPER	R/W	R/W	R/W					
RESET	0 PS	0 PS	0 PS					

This register is the attribute register for the first Local Bus to VMEbus bus map decoder.

AM These bits define the VMEbus address modifier codes the VMEbus master uses for the segment defined by map decoder 1. Because the Local Bus to VMEbus interface does not support block transfers, the block transfer address modifier codes should not be used.

WP When this bit is high, write posting is enabled to the segment defined by map decoder 1. When this bit is low, write posting is disabled to the segment defined by map decoder 1.

D16 When this bit is high, D16 data transfers are performed to the segment defined by map decoder 1. When this bit is low, D32 data transfers are performed to the segment defined by map decoder 1.

VMEbus Slave GCSR Group Address Register

ADR/SIZ	$FFF4002C (8 bits of 32)		
BIT	31	. . .	24
NAME	GCSR Group Address Register		
OPER	R/W		
RESET	$00 PS		

This register defines the group address of the GCSR as viewed from the VMEbus. The GCSR address is defined by the group address and the board address. Once enabled, the GCSR register should not be reprogrammed unless the VMEchip2 is VMEbus master.

GCSR Group These bits define the group portion of the GCSR address. These bits are compared with VMEbus address lines A8 through A15. The recommended group address for the MVME166/167 is $CC, for the MVME176/177 is $D4, and for the MVME187 is $CE.

VMEbus Slave GCSR Board Address Register

ADR/SIZ	$FFF4002C (4 bits of 32)						
BIT	23	. . .	20				
NAME	GCSR Board Address						
OPER	R/W						
RESET	$F PS						

This register defines the board address of the GCSR as viewed from the VMEbus. The GCSR address is defined by the group address and the board address. Once enabled, the GCSR register should not be reprogrammed unless the VMEchip2 is VMEbus master. The

value $F in the GCSR board address register disables the map. decoder. The map decoder is enabled when the board address is not $F.

GCSR Board These bits define the board number portion of the GCSR address. These bits are compared with VMEbus address lines A4 through A7. The GCSR is enabled by values $0 through $E. The address $XXFY in the VMEbus A16 space is reserved for the location monitors LM0 through LM3. Note: XX is the group address and Y is the location monitor (1,LM0; 3,LM1; 5,LM2; 7,LM3).

Local Bus To VMEbus Enable Control Register

ADR/SIZ	$FFF4002C (4 bits of 32)							
					19	18	17	16
NAME					EN4	EN3	EN2	EN1
OPER					R/W	R/W	R/W	R/W
RESET					0 PSL	0 PSL	0 PSL	0 PSL

This register is the map decoder enable register for the four programmable Local Bus to VMEbus map decoders.

EN1 When this bit is high, the first Local Bus to VMEbus map decoder is enabled. When this bit is low, the first Local Bus to VMEbus map decoder is disabled.

EN2 When this bit is high, the second Local Bus to VMEbus map decoder is enabled. When this bit is low, the second Local Bus to VMEbus map decoder is disabled.

EN3 When this bit is high, the third Local Bus to VMEbus map decoder is enabled. When this bit is low, the third Local Bus to VMEbus map decoder is disabled.

EN4 When this bit is high, the fourth Local Bus to VMEbus map decoder is enabled. When this bit is low, the fourth Local Bus to VMEbus map decoder is disabled.

Local Bus To VMEbus I/O Control Register

ADR/SIZ	$FFF4002C (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	I2EN	I2WP	I2SU	I2PD	I1EN	I1D16	I1WP	I1SU
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PS	0 PS	0 PS	0 PSL	0 PS	0 PS	0 PS

This register controls the VMEbus short I/O map and the F page ($F0000000 through $FF7FFFFF) I/O map.

I1SU When this bit is high, the VMEchip2 drives a supervisor address modifier code when the short I/O space is accessed. When this bit is low, the VMEchip2 drives a user address modifier code when the short I/O space is accessed.

I1WP When this bit is high, write posting is enabled to the VMEbus short I/O segment. When this bit is low, write posting is disabled to the VMEbus short I/O segment.

I1D16 When this bit is high, D16 data transfers are performed to the VMEbus short I/O segment. When this bit is low, D32 data transfers are performed to the VMEbus short I/O segment.

I1EN When this bit is high, the VMEbus short I/O map decoder is enabled. When this bit is low, the VMEbus short I/O map decoder is disabled.

I2PD When this bit is high, the VMEchip2 drives a program address modifier code when the F page is accessed. When this bit is low, the VMEchip2 drives a data address modifier code when the F page is accessed.

I2SU When this bit is high, the VMEchip2 drives a supervisor address modifier code when the F page is accessed. When this bit is low, the VMEchip2 drives a user address modifier code when the F page is accessed.

I2WP When this bit is high, write posting is enabled to the Local Bus F page. When this bit is low, write posting is disabled to the Local Bus F page.

I2EN When this bit is high, the F page ($F0000000 through $FF7FFFFF) map decoder is enabled. The F0 page is defined as A24/D16 on the VMEbus while the F1-FE pages are defined as A32/D16. When this bit is low, the F page is disabled.

ROM Control Register

ADR/SIZ	$FFF4002C (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
NAME	SIZE		BSPD			ASPD		
OPER	R/W		R/W			R/W		
RESET	0 PS		0 PS			0 PS		

This register is the ROM control register. The VMEchip2 provides a map decoder and control logic for two banks of ROM. The ROM size and speed are programmable. Bank A is always selected following reset to allow the processor to fetch the program counter and stack pointer. (Refer to the ROM0 bit in the EPROM Decoder, SRAM and DMA Control Register description later in this chapter.) The time from CS* to data valid, for the ROMs used with the VMEchip2, must be less than:

(T * (*local bus clocks* - 1) - 35)

where **T** is the Local Bus clock period, and *local bus clocks* is the programmed number of Local Bus clocks. For example, if the Local Bus clock is 33 MHz (30 ns), and the number of Local Bus clocks is 4, the access time of the ROMs must be less than:

(30 * (4-1) -35) = 55 ns

Note The VMEchip2 runs at half the MPU speed on the MVME176/177. For example, an MVME176/177 with a 50 MHz MPU will run the VMEchip2 at 25 MHz.

ASPD These bits define the number of Local Bus clocks for a bank A ROM cycle.

	Local Bus	Maximum EPROM/Flash Access Time		
ASPD	**Clocks**	**at 25 MHz**	**at 30 MHz**	**at 33 MHz**
0	11	365 ns	299 ns	265 ns
1	10	325 ns	265 ns	235 ns
2	9	285 ns	232 ns	205 ns
3	8	245 ns	199 ns	175 ns
4	7	205 ns	165 ns	145 ns
5	6	165 ns	132 ns	115 ns
6	5	125 ns	99 ns	85 ns
7	4	85 ns	65 ns	55 ns

BSPD These bits define the number of Local Bus clocks for a bank B ROM cycle.

	Local Bus	Maximum EPROM/Flash Access Time		
BSPD	**Clocks**	**at 25 MHz**	**at 30 MHz**	**at 33 MHz**
0	11	365 ns	299 ns	265 ns
1	10	325 ns	265 ns	235 ns
2	9	285 ns	232 ns	205 ns
3	8	245 ns	199 ns	175 ns
4	7	205 ns	165 ns	145 ns
5	6	165 ns	132 ns	115 ns
6	5	125 ns	99 ns	85 ns
7	4	85 ns	65 ns	55 ns

SIZE These bits define the size of the ROM chips. The ending address of bank A and the starting and ending address of bank B is defined by the ROM size. This allows the ROMs to be contiguous when both banks are equal in size.

0	8-megabit chips	Bank A	$FF800000 to $FF9FFFFF
		Bank B	$FFA00000 to $FFBFFFFF
1	4-megabit chips	Bank A	$FF800000 to $FF8FFFFF
		Bank B	$FF900000 to $FF9FFFFF
2	2-megabit chips	Bank A	$FF800000 to $FF87FFFF
		Bank B	$FF880000 to $FF8FFFFF
3	1-megabit chips	Bank A	$FF800000 to $FF83FFFF
		Bank B	$FF840000 to $FF87FFFF

Programming the VMEchip2 DMA Controller

This section includes programming information on the DMA controller, VMEbus interrupter, MPU status register, and Local Bus to VMEbus requester register.

The VMEchip2 features a Local Bus - VMEbus DMA controller (DMAC). The DMAC has two modes of operation: command chaining, and direct. In the direct mode, the Local Bus address, the VMEbus address, the byte count, and the control register of the DMAC are programmed and the DMAC is enabled. The DMAC transfers data, as programmed, until the byte count is zero or an error is detected. When the DMAC stops, the status bits in the DMAC status register are set and an interrupt is sent to the Local Bus interrupter. If the DMAC interrupt is enabled in the Local Bus interrupter, the Local Bus is interrupted. The time on and time off timers should be programmed to control the VMEbus bandwidth used by the DMAC.

A maximum of 4GB of data may be transferred with one DMAC command. Larger transfers can be accomplished using the command chaining mode. In the command chaining mode, a singly-linked list of commands is built in local memory and the table address register in the DMAC is programmed with the starting address of the list of commands. The DMAC control register is programmed and the DMAC is enabled. The DMAC executes commands from the list until all commands are executed or an error is detected. When the DMAC stops, the status bits are set in the DMAC status register and an interrupt is sent to the Local Bus interrupter. If the DMAC interrupt is enabled in the Local Bus interrupter, the Local Bus is interrupted. When the DMAC finishes processing a command in the list, and interrupts are enabled for that command, the DMAC sends an interrupt to the Local Bus interrupter. If the DMAC interrupt is enabled in the Local Bus interrupter, the Local Bus is interrupted.

The DMAC control is divided into two registers. The first register is only accessible by the processor. The second register can be loaded by the processor in the direct mode and by the DMAC in the command chaining mode.

Once the DMAC is enabled, the counter and control registers should not be modified by software. When the command chaining mode is used, the list of commands must be in local 32-bit memory and the entries must be four-byte aligned.

A DMAC command list includes one or more DMAC command packets. A DMAC command packet includes a control word that defines the VMEbus AM code, the VMEbus transfer size, the VMEbus transfer method, the DMA transfer direction, the VMEbus and Local Bus address counter operation, and the Local Bus snoop operation. The format of the control word is the same as the lower 16 bits of the control register. The command packet also includes a Local Bus address, a VMEbus address, a byte count, and a pointer to the next command packet in the list. The end of a command is indicated by setting bit 0 or 1 of next command address. The command packet format is shown in Table 4-3.

Table 4-3. DMAC Command Table Format

Entry	Function	
0 (bits 0-15)	--	Control Word
1 (bits 0-31)	Local Bus Address	
2 (bits 0-31)	VMEbus Address	
3 (bits 0-31)	Byte Count	
4 (bits 0-31)	Address of Next Command Packet	

DMAC Registers

This section provides addresses and bit level descriptions of the DMAC counters, control registers, and status registers. Other control functions are also included in this section.

PROM Decoder, SRAM and DMA Control Register

ADR/SIZ	$FFF40030 (8 bits [6 used] of 32)							
BIT	23	22	21	20	19	18	17	16
NAME			WAIT RMW	ROM0	TBLSC		SRAMS	
OPER			R/W	R/W	R/W		R/W	
RESET			0 PSL	1 PSL	0 PS		0 PS	

This register controls the EPROM decoder, the snoop control bits used by the DMAC when it is accessing table entries, and the access time of the SRAM (Static RAM, also known as slow RAM). The time from SRAM CS* to data valid, for the SRAMs used with the VMEchip2, must be less than:

(T * (*local bus clocks* -1) - 35)

where **T** is the Local Bus clock period, and *local bus clocks* is the programmed number of Local Bus clocks. For example, if the Local Bus clock is 33 MHz (30 ns), and the number of Local Bus clocks is 3, the access time of the SRAMs must be less than:

(30 * (3-1) -35) = 25 ns

Note The VMEchip2 runs at half the MPU speed on the MVME176/177. For example, an MVME176/177 with a 50 MHz MPU will run the VMEchip2 at 25 MHz.

SRAMS These bits define the number of Local Bus clocks for a static RAM cycle.

		Maximum SRAM Access Time		
SRAMS	**Local Bus Clocks**	**at 25 MHz**	**at 30 MHz**	**at 33 MHz**
0	6	165 ns	132 ns	115 ns
1	5	125 ns	99 ns	85 ns
2	4	85 ns	65 ns	55 ns
3	3	45 ns	32 ns	25 ns

TBLSC These bits control the snoop signal lines on the Local Bus when the DMAC is table walking. These bits must be 0 on the MVME187.

0 Snoop inhibited

1 Write - Sink data
Read - Supply dirty data and leave dirty

This bit must be 0 on the MVME176/177.

2 Write - Invalidate
Read - Supply dirty data and mark invalid

3 Snoop inhibited

ROM0 When this bit is set to 1, the EPROM decoder responds at $00000000 to $003FFFFF and $FF800000 to $FFBFFFFF.

When this bit is set to 0, the EPROM decoder responds only at $FF800000 to $FFBFFFFF.

Note ROM0 is set to 1 by power-up reset, SYSRESET, and local reset, causing ROM BANK A to provide reset vectors for the MPU.

WAIT RMW Not used.

Local Bus To VMEbus Requester Control Register

ADR/SIZ	$FFF40030 (8 bits [7 used] of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	ROBN	DHB	DWB		LVFAIR	LVRWD	LVREQL	
OPER	R/W	R	R/W		R/W	R/W	R/W	
RESET	0 PS	0 PS	0 PSL		0 PS	0 PS	0 PS	

This register controls the VMEbus request level, the request mode, and release mode for the Local Bus to VMEbus interface.

LVREQL These bits define the VMEbus request level. The request is only changed when the VMEchip2 is bus master. The VMEchip2 always requests at the old level until it becomes bus master and the new level takes effect. If the VMEchip2 is bus master when the level is changed, the new level does not take effect until the bus has been released and rerequested at the old level. The requester always requests the VMEbus at level 3 the first time following a SYSRESET.

- **0** Request level is 0
- **1** Request level is 1
- **2** Request level is 2
- **3** Request level is 3

LVRWD When this bit is high, the requester operates in the release-when-done mode. When this bit is low, the requester operates in the release-on-request mode.

LVFAIR When this bit is high, the requester operates in the fair mode. When this bit is low, the requester does not operate in the fair mode. In the fair mode, the requester waits until the request signal line for the selected level is inactive before requesting the VMEbus.

DWB When this bit is high, the VMEchip2 requests the VMEbus and does not release it. When this bit is low, the VMEchip2 releases the VMEbus according to the release mode programmed in the LVRWD bit. When the VMEbus has been acquired, the DHB bit is set.

DHB When this bit is high, the VMEbus has been acquired in response to the DWB bit being set. When the DWB bit is cleared, this bit is cleared.

ROBN When this bit is high, the VMEbus arbiter operates in the round robin mode. When this bit is low, the arbiter operates in the priority mode.

DMAC Control Register 1 (bits 0-7)

ADR/SIZ	$FFF40030 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
NAME	DHALT	DEN	DTBL	DFAIR	DRELM		DRELQ	
OPER	S	S	R/W	R/W	R/W		R/W	
RESET	0 PS	0 PS	0 PS	0 PS	0 PS		0 PS	

This control register is loaded by the processor; it is not modified when the DMAC loads new values from the command packet.

DREQL These bits define the VMEbus request level for the DMAC requester. The request is only changed when the VMEchip2 is bus master. The VMEchip2 always requests at the old level until it becomes bus master and the new level takes effect. If the VMEchip2 is bus master when the level is changed, the new level does not take effect until the bus has been released and rerequested at the old level. The requester always requests the VMEbus at level 3 the first time following a SYSRESET.

0 VMEbus request level 0
1 VMEbus request level 1
2 VMEbus request level 2
3 VMEbus request level 3

DRELM These bits define the VMEbus release mode for the DMAC requester. The DMAC always releases the bus when the FIFO is full (VMEbus to Local Bus) or empty (Local Bus to VMEbus).

- **0** Release when the time on timer has expired and a BRx* signal is active on the VMEbus.
- **1** Release when the time on timer has expired.
- **2** Release when a BRx* signal is active on the VMEbus.
- **3** Release when a BRx* signal is active on the VMEbus or the time on timer has expired.

DFAIR When this bit is high, the DMAC requester operates in the fair mode. It waits until its request level is inactive before requesting the VMEbus. When this bit is low, the DMAC requester does not operate in the fair mode.

DTBL The DMAC operates in the direct mode when this bit is low, and it operates in the command chaining mode when this bit is high.

DEN The DMAC is enabled when this bit is set high. This bit always reads 0.

DHALT When this bit is high, the DMAC halts at the end of a command when the DMAC is operating in the command chaining mode. When this bit is low, the DMAC executes the next command in the list.

DMAC Control Register 2 (bits 8-15)

ADR/SIZ	$FFF40034 (8 bits [7 used] of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	INTE	SNP			VINC	LINC	TVME	D16
OPER	R/W	R/W			R/W	R/W	R/W	R/W
RESET	0 PS	0 PS			0 PS	0 PS	0 PS	0 PS

This portion of the control register is loaded by the processor or by the DMAC when it loads the command word from the command packet. Because this register is loaded from the command packet in the command chaining mode, the descriptions here also apply to the control word in the command packet.

D16 When this bit is high, the DMAC executes D16 cycles on the VMEbus. When this bit is low, the DMAC executes D32 cycles on the VMEbus.

TVME This bit defines the direction in which the DMAC transfers data. When this bit is high, data is transferred to the VMEbus. When it is low, data is transferred to the Local Bus.

LINC When this bit is high, the Local Bus address counter is incremented during DMA transfers. When this bit is low, the counter is not incremented. This bit should normally be set high. In special situations such as transferring data to or from a FIFO, it may be desirable to not increment the counter.

VINC When this bit is high, the VMEbus address counter is incremented during DMA transfers. When this bit is low, the counter is not incremented. This bit should normally be set high. In special situations such as transferring data to or from a FIFO, it may be desirable to not increment the counter.

SNP These bits control the snoop signal lines on the Local Bus when the DMAC is Local Bus master and it is not accessing the command table. These bits must be 0 on the MVME187.

0 Snoop inhibited

1 Write - Sink data
Read - Supply dirty data and leave dirty
This bit must be 0 on the MVME176/177.

2 Write - Invalidate
Read - Supply dirty data and mark invalid

3 Snoop inhibited

INTE This bit is used only in the command chaining mode and it is only modified when the DMAC loads the control register from the control word in the command packet. When this bit in the command packet is set, an interrupt is sent to the Local Bus interrupter when the command in the packet has been executed. The Local Bus is interrupted if the DMAC interrupt is enabled.

DMAC Control Register 2 (bits 0-7)

ADR/SIZ	$FFF40034 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
NAME	BLK		VME AM					
OPER	R/W		R/W					
RESET	0 PS		0 PS					

This portion of the control register is loaded by the processor or the DMAC when it loads the command word from the command packet. Because this byte is loaded from the command packet in the command chaining mode, the descriptions here also apply to the control word in the command packet.

VME AM These bits define the address modifier codes the DMAC drives on the VMEbus when it is bus master. During non-block transfer cycles, bits 0-5 define the VMEbus address modifiers. During block transfers, bits 2-5 define VMEbus address modifier bits 2-5, and address modifier bits 0 and 1 are provided by the DMAC to indicate a block transfer. Block

transfer mode should not be set in the address modifier codes. The special block transfer bits should be set to enable block transfers. If non-block cycles are required to reach a 32- or 64-bit boundary, bits 0 and 1 are used during these cycles.

BLK These bits control the block transfer modes of the DMAC:

0 Block transfers disabled

1 The DMAC executes D32 block transfer cycles on the VMEbus. In the block transfer mode, the DMAC may execute byte and two-byte cycles at the beginning and ending of a transfer in non-block transfer mode.

2 Block transfers disabled

3 The DMAC executes D64 block transfer cycles on the VMEbus. In the block transfer mode, the DMAC may execute byte, two-byte and four-byte cycles at the beginning and ending of a transfer in non-block transfer mode.

DMAC Local Bus Address Counter

ADR/SIZ	$FFF40038 (32 bits)		
BIT	31	. . .	0
NAME	DMAC Local Bus Address Counter		
OPER	R/W		
RESET	0 PS		

In the direct mode, this counter is programmed with the starting address of the data in Local Bus memory.

DMAC VMEbus Address Counter

ADR/SIZ	$FFF4003C (32 bits)		
BIT	31	. . .	0
NAME	DMAC VMEbus Address Counter		
OPER	R/W		
RESET	0 PS		

In the direct mode, this counter is programmed with the starting address of the data in VMEbus memory.

DMAC Byte Counter

ADR/SIZ	$FFF40040 (32 bits)		
BIT	31	. . .	0
NAME	DMAC Byte Counter		
OPER	R/W		
RESET	0 PS		

In the direct mode, this counter is programmed with the number of bytes of data to be transferred.

Table Address Counter

ADR/SIZ	$FFF40044 (32 bits)		
BIT	31	. . .	0
NAME	Table Address Counter		
OPER	R/W		
RESET	0 PS		

In the command chaining mode, this counter should be loaded by the processor with the starting address of the list of commands. This register gets reloaded by the DMAC with the starting address of the current command. The last command in a list should have bits 0 and 1 set in the next command pointer.

VMEbus Interrupter Control Register

ADR/SIZ	$FFF40048 (8 bits [7 used] of 32)							
BIT	31	30	29	28	27	26	25	24
NAME		IRQ1S		IRQC	IRQS	IRQL		
OPER		R/W		S	R	S		
RESET		0 PS		0 PS	0 PS	0 PS		

This register controls the VMEbus interrupter.

IRQL These bits define the level of the VMEbus interrupt generated by the VMEchip2. A VMEbus interrupt is generated by writing the desired level to these bits. These bits always read 0 and writing 0 to these bits has no effect.

IRQS This bit is the IRQ status bit. When this bit is high, the VMEbus interrupt has not been acknowledged. When this bit is low, the VMEbus interrupt has been acknowledged. This is a read-only status bit.

IRQC This bit is VMEbus interrupt clear bit. When this bit is set high, the VMEbus interrupt is removed. This feature is only used when the IRQ1 broadcast mode

is used. Normal VMEbus interrupts should never be cleared. This bit always reads 0 and writing a 0 to this bit has no effect.

IRQ1S These bits control the function of the IRQ1 signal line on the VMEbus:

0 The IRQ1 signal from the interrupter is connected to the IRQ1 signal line on the VMEbus.

1 The output from tick timer 1 is connected to the IRQ1 signal line on the VMEbus.

2 The IRQ1 signal from the interrupter is connected to the IRQ1 signal line on the VMEbus.

3 The output from tick timer 2 is connected to the IRQ1 signal line on the VMEbus.

VMEbus Interrupter Vector Register

ADR/SIZ	$FFF40048 (8 bits of 32)		
BIT	23	1	16
NAME	INTERRUPTER VECTOR		
OPER	R/W		
RESET	$0F PS		

This register controls the VMEbus interrupter vector.

MPU Status and DMA Interrupt Count Register

ADR/SIZ	$FFF40048 (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	DMAIC				MCLR	MLBE	MLPE	MLOB
OPER	R				C	R	R	R
RESET	0 PS				0 PS	0 PS	0 PS	0 PS

This is the MPU status register and DMAC interrupt counter.

MLOB When this bit is set, the MPU received a TEA and the status indicated offboard. This bit is cleared by writing a one to the MCLR bit in this register.

MLPE When this bit is set, the MPU received a TEA and the status indicated a parity error during a DRAM data transfer. This bit is cleared by writing a one to the MCLR bit in this register.

MLBE When this bit is set, the MPU received a TEA and additional status was not provided. This bit is cleared by writing a one to the MCLR bit in this register.

MCLR Writing a one to this bit clears the MPU status bits 7, 8, 9 and 10 (MLTO, MLOB, MLPE, and MLBE) in this register.

DMAIC The DMAC interrupt counter is incremented when an interrupt is sent to the Local Bus interrupter. The value in this counter indicates the number of commands processed when the DMAC is operated in the command chaining mode. If interrupt count exceeds 15, the counter rolls over. This counter operates regardless of whether the DMAC interrupts are enabled. This counter is cleared when the DMAC is enabled.

DMAC Status Register

ADR/SIZ	$FFF40048 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
NAME	MLTO	DLBE	DLPE	DLOB	DLTO	TBL	VME	DONE
OPER	R	R	R	R	R	R	R	R
RESET	0 PS	0 PS	0 PS	0 PS	0 PS	0 PS	0 PS	0 PS

This is the DMAC status register.

DONE This bit is set when the DMAC has finished executing commands and there were no errors or the DMAC has finished executing command because the halt bit was set. This bit is cleared when the DMAC is enabled.

VME When this bit is set, the DMAC received a VMEbus BERR during a data transfer. This bit is cleared when the DMAC is enabled.

TBL When this bit is set, the DMAC received an error on the Local Bus while it was reading commands from the command packet. Additional information is provided in bits 3 - 6 (DLTO, DLOB, DLPE, and DLBE). This bit is cleared when the DMAC is enabled.

DLTO When this bit is set, the DMAC received a TEA and the status indicated a Local Bus time-out. This bit is cleared when the DMAC is enabled.

DLOB When this bit is set, the DMAC received a TEA and the status indicated offboard. This bit is cleared when the DMAC is enabled.

DLPE When this bit is set, the DMAC received a TEA and the status indicated a parity error during a DRAM data transfer. This bit is cleared when the DMAC is enabled.

DLBE When this bit is set, the DMAC received a TEA and additional status was not provided. This bit is cleared when the DMAC is enabled.

MLTO When this bit is set, the MPU received a TEA and the status indicated a Local Bus time-out. This bit is cleared by a writing a one to the MCLR bit in this register.

Programming the Tick and Watchdog Timers

The VMEchip2 has two 32-bit tick timers and one watchdog timer. This section provides addresses and bit level descriptions of the prescaler, tick timer, watchdog timer registers and various other timer registers.

VMEbus Arbiter Time-out Control Register

ADR/SIZ	$FFF4004C (8 bits [1 used] of 32)							
BIT	31	30	29	28	27	26	25	24
NAME								ARBTO
OPER								R/W
RESET								0 PS

This register controls the VMEbus arbiter time-out timer.

ARBTO When this bit is high, the VMEbus grant time-out timer is enabled. When this bit is low, the VMEbus grant timer is disabled. When the timer is enabled and the arbiter does not receive a BBSY signal within 256 µs after a grant is issued, the arbiter asserts BBSY and removes the grant. The arbiter then re-arbitrates any pending requests.

DMAC Timers and VMEbus Global Time-out Control Register

ADR/SIZ	$FFF4004C (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
NAME	TIME OFF			TIME ON			VGTO	
OPER	R/W			R/W			R/W	
RESET	0 PS			0 PS			0 PS	

This register controls the DMAC time off timer, the DMAC time on timer, and the VMEbus global time-out timer.

VGTO These bits define the VMEbus global time-out value. When DS0 or DS1 is asserted on the VMEbus, the timer begins timing. If the timer times out before the data strobes are removed, a BERR signal is sent to the VMEbus. The global time-out timer is disabled when the VMEchip2 is not system controller.

0 8 µs
1 64 µs
2 256 µs
3 The timer is disabled

TIME ON These bits define the maximum time the DMAC spends on the VMEbus:

0	16 µs	**4**	256 µs
1	32 µs	**5**	512 µs
2	64 µs	**6**	1024 µs
3	128 µs	**7**	When done (or no data)

TIME OFF These bits define the minimum time the DMAC spends off the VMEbus:

0	0 µs	**4**	128 µs
1	16 µs	**5**	256 µs
2	32 µs	**6**	512 µs
3	64 µs	7	1024 µs

VME Access, Local Bus and Watchdog Time-out Control Register

ADR/SIZ	$FFF4004C (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	VATO		LBTO		WDTO			
OPER	R/W		R/W		R/W			
RESET	0 PS		0 PS		0 PS			

WDTO These bits define the watchdog time-out period:

Bit Encoding	Time-out	Bit Encoding	Time-out
0	512 µs	8	128 ms
1	1 ms	9	256 ms
2	2 ms	10	512 ms
3	4 ms	11	1 s
4	8 ms	12	4 s
5	16 ms	13	16 s
6	32 ms	14	32 s
7	64 ms	15	64 s

LBTO These bits define the Local Bus time-out value. The timer begins timing when TS is asserted on the Local Bus. If TA or TAE is not asserted before the timer times out, a TEA signal is sent to the Local Bus. The timer is disabled if the transfer is bound for the VMEbus.

0 8 µs
1 64 µs
2 256 µs
3 The timer is disabled

VATO These bits define the VMEbus access time-out value. When a transaction is headed to the VMEbus and the VMEchip2 is not the current VMEbus master, the access timer begins timing. If the VMEchip2 has not received bus mastership before the timer times out and the transaction is not write posted, a TEA signal is sent to the Local Bus. If the transaction is write posted, a write post error interrupt is sent to the Local Bus interrupter.

0 64 µs
1 1 ms
2 32 ms
3 The timer is disabled

Prescaler Control Register

ADR/SIZ	$FFF4004C (8 bits of 32)		
BIT	7	. . .	0
NAME	Prescaler Adjust		
OPER	R/W		
RESET	$DF P		

The prescaler provides the various clocks required by the counters and timers in the VMEchip2. In order to specify absolute times from these counters and timers, the prescaler must be adjusted for different Local Bus clocks. The prescaler register should be programmed based on the following equation. This provides a 1MHz clock to the tick timers.

prescaler register = 256 - *B clock* (MHz)

For example, for operation at 20 MHz the prescaler value is $EC, at 25 MHz it is $E7, at 30 MHz it is $E2, and at 33 MHz it is $DF.

Note The VMEchip2 runs at half the MPU speed on the MVME176/177. For example, an MVME176/177 with a 50 MHz MPU will run the VMEchip2 at 25 MHz.

Non-integer Local Bus clocks introduce an error into the specified times for the various counters and timers. This is most notable in the tick timers. The tick timer clock can be derived by the following equation.

tick timer clock = *B clock* / (256 - *prescaler value*)

If the prescaler is not correctly programmed, the bus timers do not generate their specified values and the VMEbus reset time may be violated. The maximum clock frequency for the tick timers is the B clock divided by two. The prescaler register control logic does not allow the value 255 ($FF) to be programmed.

Tick Timer 1 Compare Register

ADR/SIZ	$FFF40050 (32 bits)		
BIT	31	. . .	0
NAME	Tick timer 1 Compare Register		
OPER	R/W		
RESET	0 P		

The tick timer 1 counter is compared to this register. When they are equal, an interrupt is sent to the Local Bus interrupter and the overflow counter is incremented. If the clear-on-compare mode is enabled, the counter is also cleared. For periodic interrupts, the following equation should be used to calculate the compare register value for a specific period (T).

$$\textit{compare register value} = T\ (\mu s)$$

When programming the tick timer for periodic interrupts, the counter should be cleared to zero by software and then enabled. If the counter does not initially start at zero, the time to the first interrupt may be longer or shorter than expected. Remember the rollover time for the counter is 71.6 minutes.

Tick Timer 1 Counter

ADR/SIZ	$FFF40054 (32 bits)		
BIT	31	. . .	0
NAME	Tick timer 1 Counter		
OPER	R/W		
RESET	0 P		

This is the tick timer 1 counter. When enabled, it increments every microsecond. Software may read or write the counter at any time.

Tick Timer 2 Compare Register

ADR/SIZ	$FFF40058 (32 bits)		
BIT	31	. . .	0
NAME	Tick timer 2 Compare Register		
OPER	R/W		
RESET	0 P		

The tick timer 2 counter is compared to this register. When they are equal, an interrupt is sent to the Local Bus interrupter and the overflow counter is incremented. If the clear-on-compare mode is enabled, the counter is also cleared. For periodic interrupts, the following equation should be used to determine the compare register value for a specific period.

compare register value = T (μs)

When programming the tick timer for periodic interrupts, the counter should be cleared to zero by software and then enabled. If the counter does not initially start at zero, the time to the first interrupt may be longer or shorter than expected. Remember the rollover time for the counter is 71.6 minutes.

Tick Timer 2 Counter

ADR/SIZ	$FFF4005C (32 bits)		
BIT	31	. . .	0
NAME	Tick timer 2 Counter		
OPER	R/W		
RESET	0 P		

This is the tick timer 2 counter. When enabled, it increments every microsecond. Software may read or write the counter at any time.

Board Control Register

ADR/SIZ	$FFF40060 (8 bits [7 used] of 32)							
BIT	31	30	29	28	27	26	25	24
NAME		SCON	SFFL	BRFLI	PURS	CPURS	BDFLO	RSWE
OPER		R	R	R	R	C	R/W	R/W
RESET		X	X	1 PSL	1 P	0 PS	1 PSL	1 P

RSWE When this bit is high, the RESET switch is enabled. When this bit is low, the RESET switch is disabled.

BDFLO When this bit is high, the VMEchip2 asserts the BRDFAIL signal pin. When this bit is low, this bit does not contribute to the BRDFAIL signal on the VMEchip2.

CPURS When this bit is set high, the power-up reset status bit is cleared. This bit is always read zero.

PURS This bit is set by a power-up reset. It is cleared by a write to the CPURS bit.

BRFLI When this status bit is high, the BRDFAIL signal pin on the VMEchip2 is asserted. When this status bit is low, the BRDFAIL signal pin on the VMEchip2 is not asserted. The BRDFAIL pin may be asserted by an external device, the BDFLO bit in this register, or a watchdog time-out.

SFFL When this status bit is high, the SYSFAIL signal line on the VMEbus is asserted. When this status bit is low, the SYSFAIL signal line on the VMEbus is not asserted.

SCON When this status bit is high, the VMEchip2 is configured as system controller. When this status bit is low, the VMEchip2 is not configured as system controller.

Watchdog Timer Control Register

ADR/SIZ	$FFF40060 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
NAME	SRST	WDCS	WDCC	WDTO	WDBFE	WDS/L	WDRSE	WDEN
OPER	S	C	C	R	R/W	R/W	R/W	R/W
RESET	0 PS	0	0	0 P	0 PSL	0 PSL	0 PSL	0 PSL

WDEN When this bit is high, the watchdog timer is enabled. When this bit is low, the watchdog timer is not enabled.

WDRSE When this bit is high, and a watchdog time-out occurs, a SYSRESET or LRESET is generated. The WDS/L bit in this register selects the reset. When this bit is low, a watchdog time-out does not cause a reset.

WDS/L When this bit is high and the watchdog timer has timed out and the watchdog reset enable (WDRSE bit in this register) is high, a SYSRESET signal is generated on the VMEbus which in turn causes LRESET to be asserted. When this bit is low and the watchdog timer has timed out and the watchdog reset enable (WDRSE bit in this register) is high, an LRESET signal is generated on the Local Bus.

WDBFE When this bit is high and the watchdog timer has timed out, the VMEchip2 asserts the BRDFAIL signal pin. When this bit is low, the watchdog timer does not contribute to the BRDFAIL signal on the VMEchip2.

WDTO When this status bit is high, a watchdog time-out has occurred. When this status bit is low, a watchdog time-out has not occurred. This bit is cleared by writing a one to the WDCS bit in this register.

WDCC When this bit is set high, the watchdog counter is reset. The counter must be reset within the time-out period or a watchdog time-out occurs.

WDCS When this bit is set high, the watchdog time-out status bit (WDTO bit in this register) is cleared.

SRST When this bit is set high, a SYSRESET signal is generated on the VMEbus. SYSRESET resets the VMEchip2 and clears this bit.

Tick Timer 2 Control Register

ADR/SIZ	$FFF40060 (8 bits [7 used] of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	OVF					COVF	COC	EN
OPER	R					C	R/W	R/W
RESET	0 PS					0 PS	0 PS	0 PS

EN When this bit is high, the counter increments. When this bit is low, the counter does not increment.

COC When this bit is high, the counter is reset to zero when it compares with the compare register. When this bit is low, the counter is not reset.

COVF The overflow counter is cleared when a one is written to this bit.

OVF These bits are the output of the overflow counter. The overflow counter is incremented each time the tick timer sends an interrupt to the Local Bus interrupter. The overflow counter can be cleared by writing a one to the COVF bit.

Tick Timer 1 Control Register

ADR/SIZ	$FFF40060 (8 bits [7 used] of 32)							
BIT	7	6	5	4	31	2	1	0
NAME	OVF					COVF	COC	EN
OPER	R					C	R/W	R/W
RESET	0 PS					0 PS	0 PS	0 PS

EN When this bit is high, the counter increments. When this bit is low, the counter does not increment.

COC When this bit is high, the counter is reset to zero when it compares with the compare register. When this bit is low, the counter is not reset.

COVF The overflow counter is cleared when a one is written to this bit.

OVF These bits are the output of the overflow counter. The overflow counter is incremented each time the tick timer sends an interrupt to the Local Bus interrupter. The overflow counter can be cleared by writing a one to the COVF bit.

Prescaler Counter

ADR/SIZ	$FFF40064 (32 bits)		
BIT	31	. . .	0
NAME	Prescaler Counter		
OPER	R/W		
RESET	0 P		

The VMEchip2 has a 32-bit prescaler that provides the clocks required by the various timers in the chip. Access to the prescaler is provided for test purposes. The counter is described here because it may be useful in other applications. The lower 8 bits of the prescaler counter increment to $FF at the Local Bus clock rate and then they are loaded from the prescaler adjust register. When the load occurs, the upper 24 bits are incremented. When the prescaler adjust register is correctly programmed, the lower 8 bits increment at the Local Bus clock rate and the upper 24 bits increment every microsecond. The counter may be read at any time.

Programming the Local Bus Interrupter

The Local Bus interrupter is used by devices that wish to interrupt the Local Bus. There are 31sources that can interrupt the Local Bus through the VMEchip2. In the general case, each interrupter has a level select register, an enable bit, a status bit, a clear bit, and for the software interrupts, a set bit. Each interrupter also provides a unique interrupt vector to the processor. The upper four bits of the vector are programmable in the vector base registers. The lower four bits are unique for each interrupter. There are two base registers, one for the first 16 interrupters, and one for the next 8 interrupters. The VMEbus interrupters provide their own vectors. A summary of the interrupts is shown in Table 4-4.

The status bit of an interrupter is affected by the enable bit. If the enable bit is low, the status bit is also low. Interrupts may be polled by setting the enable bit and programming the level to zero. This enables the status bit and prevents the Local Bus from being interrupted. The enable bit does not clear edge-sensitive interrupts. If necessary, edge-sensitive interrupts should be cleared, in order to remove any old interrupts, and then enabled. The master interrupt enable (MIEN) bit must be set before the VMEchip2 can generate any interrupts. The MIEN bit is in the I/O Control Register 1.

Table 4-4. Local Bus Interrupter Summary

Interrupt	Vector	Priority for Simultaneous Interrupts
VMEbus IRQ1	External	Lowest
VMEbus IRQ2	External	
VMEbus IRQ3	External	
VMEbus IRQ4	External	
VMEbus IRQ5	External	
VMEbus IRQ6	External	
VMEbus IRQ7	External	
Spare	$Y7	
Software 0	$Y8	
Software 1	$Y9	
Software 2	$YA	
Software 3	$YB	
Software 4	$YC	
Software 5	$YD	
Software 6	$YE	
Software 7	$YF	

Table 4-4. Local Bus Interrupter Summary (Continued)

Interrupt	Vector	Priority for Simultaneous Interrupts
GCSR LM0	$X0	
GCSR LM1	$X1	
GCSR SIG0	$X2	
GCSR SIG1	$X3	
GCSR SIG2	$X4	
GCSR SIG3	$X5	
DMAC	$X6	
VMEbus Interrupter Acknowledge	$X7	
Tick Timer 1	$X8	
Tick Timer 2	$X9	
VMEbus IRQ1 Edge-Sensitive	$XA	
External Input (parity error)	$XB	
VMEbus Master Write Post Error	$XC	
VMEbus SYSFAIL	$XD	
ABORT Switch	$XE	
VMEbus ACFAIL	$XF	Highest

Notes 1. X = The contents of vector base register 0.

2. Y = The contents of vector base register 1.

3. Refer to the Vector Base Register description later in this chapter for recommended Vector Base Register values.

Local Bus Interrupter Status Register (bits 24-31)

ADR/SIZ	$FFF40068 (8 bits of 32)							
BIT	31	30	29	28	27	26	25	24
NAME	ACF	AB	SYSF	MWP	PE	VI1E	TIC2	TIC1
OPER	R	R	R	R	R	R	R	R
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the Local Bus interrupter status register. When an interrupt status bit is high, a Local Bus interrupt is being generated. When an interrupt status bit is low, a local interrupt is not being generated. The interrupt status bits are:

TIC1 Tick timer 1 interrupt

TIC2 Tick timer 2 interrupt

VI1E VMEbus IRQ1 edge-sensitive interrupt

PE External interrupt (parity error)

MWP VMEbus master write post error interrupt

SYSF VMEbus SYSFAIL interrupt

AB ABORT switch interrupt

ACF VMEbus ACFAIL interrupt

Local Bus Interrupter Status Register (bits 16-23)

ADR/SIZ	$FFF40068 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
NAME	VIA	DMA	SIG3	SIG2	SIG1	SIG0	LM1	LM0
OPER	R	R	R	R	R	R	R	R
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the Local Bus interrupter status register. When an interrupt status bit is high, a Local Bus interrupt is being generated. When an interrupt status bit is low, a local interrupt is not being generated. The interrupt status bits are:

LM0 GCSR LM0 interrupt

LM1 GCSR LM1 interrupt

SIG0 GCSR SIG0 interrupt

SIG1 GCSR SIG1 interrupt

SIG2 GCSR SIG2 interrupt

SIG3 GCSR SIG3 interrupt

DMA DMAC interrupt

VIA VMEbus interrupter acknowledge interrupt

Local Bus Interrupter Status Register (bits 8-15)

ADR/SIZ	$FFF40068 (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	SW7	SW6	SW5	SW4	SW3	SW2	SW1	SW0
OPER	R	R	R	R	R	R	R	R
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the Local Bus interrupter status register. When an interrupt status bit is high, a Local Bus interrupt is being generated. When an interrupt status bit is low, a local interrupt is not being generated. The interrupt status bits are:

SW0 Software 0 interrupt

SW1 Software 1 interrupt

SW2 Software 2 interrupt

SW3 Software 3 interrupt

SW4 Software 4 interrupt

SW5 Software 5 interrupt

SW6 Software 6 interrupt

SW7 Software 7 interrupt

Local Bus Interrupter Status Register (bits 0-7)

ADR/SIZ	$FFF40068 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
NAME	SPARE	VME7	VME6	VME5	VME4	VME3	VME2	VME1
OPER	R	R	R	R	R	R	R	R
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the Local Bus interrupter status register. When an interrupt status bit is high, a Local Bus interrupt is being generated. When an interrupt status bit is low, a local interrupt is not being generated. The interrupt status bits are:

VME1 VMEbus IRQ1 Interrupt

VME2 VMEbus IRQ2 Interrupt

VME3 VMEbus IRQ3 Interrupt

VME4 VMEbus IRQ4 Interrupt

VME5 VMEbus IRQ5 Interrupt

VME6 VMEbus IRQ6 Interrupt

VME7 VMEbus IRQ7 Interrupt

SPARE This bit is not used

Local Bus Interrupter Enable Register (bits 24-31)

ADR/SIZ	$FFF4006C (8 bits of 32)							
BIT	31	30	29	28	27	26	25	24
NAME	EACF	EAB	ESYSF	EMWP	EPE	EVI1E	ETIC2	ETIC1
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the Local Bus interrupter enable register. When an enable bit is high, the corresponding interrupt is enabled. When an enable bit is low, the corresponding interrupt is disabled. The enable bit does not clear edge-sensitive interrupts or prevent the flip flop from being set. If necessary, edge-sensitive interrupters should be cleared to remove any old interrupts and then enabled.

ETIC1 Enable tick timer 1 interrupt

ETIC2 Enable tick timer 2 interrupt

EVI1E Enable VMEbus IRQ1 edge-sensitive interrupt

EPE Enable external interrupt (parity error)

EMWP Enable VMEbus master write post error interrupt

ESYSF Enable VMEbus SYSFAIL interrupt

EAB Enable ABORT switch interrupt

EACF Enable VMEbus ACFAIL interrupt

Local Bus Interrupter Enable Register (bits 16-23)

ADR/SIZ	$FFF4006C (8-bits)							
BIT	23	22	21	20	19	18	17	16
NAME	EVIA	EDMA	ESIG3	ESIG2	ESIG1	ESIG0	ELM1	ELM0
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is the Local Bus interrupter enable register. When an enable bit is high, the corresponding interrupt is enabled. When an enable bit is low, the corresponding interrupt is disabled. The enable bit does not clear edge-sensitive interrupts or prevent the flip flop from being set. If necessary, edge-sensitive interrupters should be cleared to remove any old interrupts and then enabled.

ELM0 Enable GCSR LM0 interrupt

ELM1 Enable GCSR LM1 interrupt

ESIG0 Enable GCSR SIG0 interrupt

ESIG1 Enable GCSR SIG1 interrupt

ESIG2 Enable GCSR SIG2 interrupt

ESIG3 Enable GCSR SIG3 interrupt

EDMA Enable DMAC interrupt

EVIA VMEbus interrupter acknowledge interrupt

Local Bus Interrupter Enable Register (bits 8-15)

ADR/SIZ	$FFF4006C (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	ESW7	ESW6	ESW5	ESW4	ESW3	ESW2	ESW1	ESW0
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This is the Local Bus interrupter enable register. When an enable bit is high, the corresponding interrupt is enabled. When an enable bit is low, the corresponding interrupt is disabled. The enable bit does not clear edge-sensitive interrupts or prevent the flip flop from being set. If necessary, edge-sensitive interrupters should be cleared to remove any old interrupts and then enabled.

ESW0 Enable software 0 interrupt

ESW1 Enable software 1 interrupt

ESW2 Enable software 2 interrupt

ESW3 Enable software 3 interrupt

ESW4 Enable software 4 interrupt

ESW5 Enable software 5 interrupt

ESW6 Enable software 6 interrupt

ESW7 Enable software 7 interrupt

Local Bus Interrupter Enable Register (bits 0-7)

ADR/SIZ	$FFF4006C (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
NAME	SPARE	EIRQ7	EIRQ6	EIRQ5	EIRQ4	EIRQ3	EIRQ2	EIRQ1
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This is the Local Bus interrupter enable register. When an enable bit is high, the corresponding interrupt is enabled. When an enable bit is low, the corresponding interrupt is disabled. The enable bit does not clear edge-sensitive interrupts or prevent the flip flop from being set. If necessary, edge-sensitive interrupters should be cleared to remove any old interrupts and then enabled.

EIRQ1 Enable VMEbus IRQ1 interrupt

EIRQ2 Enable VMEbus IRQ2 interrupt

EIRQ3 Enable VMEbus IRQ3 interrupt

EIRQ4 Enable VMEbus IRQ4 interrupt

EIRQ5 Enable VMEbus IRQ5 interrupt

EIRQ6 Enable VMEbus IRQ6 interrupt

EIRQ7 Enable VMEbus IRQ7 interrupt

SPARE SPARE

Software Interrupt Set Register (bits 8-15)

ADR/SIZ	$FFF40070 (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	SSW7	SSW6	SSW5	SSW4	SSW3	SSW2	SSW17	SSW07
OPER	S	S	S	S	S	S	S	S
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is used to set the software interrupts. An interrupt is set by writing a one to it. The software interrupt set bits are:

SSW0 Set software 0 interrupt

SSW1 Set software 1 interrupt

SSW2 Set software 2 interrupt

SSW3 Set software 3 interrupt

SSW4 Set software 4 interrupt

SSW5 Set software 5 interrupt

SSW6 Set software 6 interrupt

SSW7 Set software 7 interrupt

Interrupt Clear Register (bits 24-31)

ADR/SIZ	$FFF40074 (8 bits of 32)							
BIT	31	30	29	28	27	26	25	24
NAME	CACF	CAB	CSYSF	CMWP	CPE	CVI1E	CTIC2	CTIC1
OPER	C	C	C	C	C	C	C	C
RESET	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL	0 PSL

This register is used to clear the edge-sensitive interrupts. An interrupt is cleared by writing a one to its clear bit. The clear bits are defined below.

CTIC1 Clear tick timer 1 interrupt

CTIC2 Clear tick timer 2 interrupt

CVI1E Clear VMEbus IRQ1 edge-sensitive interrupt

CPE Clear external interrupt (parity error)

CMWP Clear VMEbus master write post error interrupt

CSYSF Clear VMEbus SYSFAIL interrupt

CAB Clear ABORT switch interrupt

CACF Clear VMEbus ACFAIL interrupt

Interrupt Clear Register (bits 16-23)

ADR/SIZ	$FFF40074 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
NAME	CVIA	CDMA	CSIG3	CSIG2	CSIG1	CSIG0	CLM1	CLM0
OPER	C	C	C	C	C	C	C	C
RESET	X	X	X	X	X	X	X	X

This register is used to clear the edge sensitive-interrupts. An interrupt is cleared by writing a one to its clear bit. The clear bits are defined below.

CLM0 Clear GCSR LM0 interrupt

CLM1 Clear GCSR LM1 interrupt

CSIG0 Clear GCSR SIG0 interrupt

CSIG1 Clear GCSR SIG1 interrupt

CSIG2 Clear GCSR SIG2 interrupt

CSIG3 Clear GCSR SIG3 interrupt

CDMA Clear DMA controller interrupt

CVIA Clear VMEbus interrupter acknowledge interrupt

Interrupt Clear Register (bits 8-15)

ADR/SIZ	$FFF40074 (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	CSW7	CSW6	CSW57	CSW4	CSW3	CSW2	CSW1	CSW0
OPER	C	C	C	C	C	C	C	C
RESET	X	X	X	X	X	X	X	X

This register is used to clear the edge software interrupts. An interrupt is cleared by writing a one to its clear bit. The clear bits are:

CSW0 Clear software 0 interrupt

CSW1 Clear software 1 interrupt

CSW2 Clear software 2 interrupt

CSW3 Clear software 3 interrupt

CSW4 Clear software 4 interrupt

CSW5 Clear software 5 interrupt

CSW6 Clear software 6 interrupt

CSW7 Clear software 7 interrupt

Interrupt Level Register 1 (bits 24-31)

ADR/SIZ	$FFF40078 (8 bits [6 used] of 32)							
BIT	31	30	29	28	27	26	25	24
NAME		ACF LEVEL				AB LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the abort interrupt and the ACFAIL interrupt.

AB LEVEL These bits define the level of the abort interrupt.

ACF LEVEL These bits define the level of the ACFAIL interrupt.

Interrupt Level Register 1 (bits 16-23)

ADR/SIZ	$FFF40078 (8 bits [6 used] of 32)							
BIT	23	22	21	20	19	18	17	16
NAME		SYSF LEVEL				WPE LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the SYSFAIL interrupt and the master write post bus error interrupt.

WPE LEVEL These bits define the level of the master write post bus error interrupt.

SYSF LEVEL These bits define the level of the SYSFAIL interrupt.

Interrupt Level Register 1 (bits 8-15)

ADR/SIZ	$FFF40078 (8 bits [6 used] of 32)							
BIT	15	14	13	12	11	10	9	8
NAME		PE LEVEL				IRQ1E LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the VMEbus IRQ1 edge-sensitive interrupt and the level of the external (parity error) interrupt.

IRQ1E LEVEL These bits define the level of the VMEbus IRQ1 edge-sensitive interrupt.

PE LEVEL These bits define the level of the external (parity error) interrupt.

Interrupt Level Register 1 (bits 0-7)

ADR/SIZ	$FFF40078 (8 bits [6 used] of 32)							
BIT	7	6	5	4	3	2	1	0
NAME		TICK2 LEVEL				TICK1 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the tick timer 1 interrupt and the tick timer 2 interrupt.

TICK1 LEVEL These bits define the level of the tick timer 1 interrupt.

TICK2 LEVEL These bits define the level of the tick timer 2 interrupt.

Interrupt Level Register 2 (bits 24-31)

ADR/SIZ	$FFF4007C (8 bits [6 used] of 32)							
BIT	31	30	29	28	27	26	25	24
NAME		VIA LEVEL				DMA LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the DMA controller interrupt and the VMEbus acknowledge interrupt.

DMA LEVEL These bits define the level of the DMA controller interrupt.

VIA LEVEL These bits define the level of the VMEbus interrupter acknowledge interrupt.

Interrupt Level Register 2 (bits 16-23)

ADR/SIZ	$FFF4007C (8 bits [6 used] of 32)							
BIT	23	22	21	20	19	18	17	16
NAME		SIG3 LEVEL				SIG2 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the GCSR SIG2 interrupt and the GCSR SIG3 interrupt.

SIG2 LEVEL These bits define the level of the GCSR SIG2 interrupt.

SIG3 LEVEL These bits define the level of the GCSR SIG3 interrupt.

Interrupt Level Register 2 (bits 8-15)

ADR/SIZ	$FFF4007C (8 bits [6 used] of 32)							
BIT	15	14	13	12	11	10	9	8
NAME		SIG1 LEVEL				SIG0 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the GCSR SIG0 interrupt and the GCSR SIG1 interrupt.

SIG0 LEVEL These bits define the level of the GCSR SIG0 interrupt.

SIG1 LEVEL These bits define the level of the GCSR SIG1 interrupt.

Interrupt Level Register 2 (bits 0-7)

ADR/SIZ	$FFF4007C (8 bits [6 used] of 32)							
BIT	7	6	5	4	3	2	1	0
NAME		LM1 LEVEL				LM0 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the GCSR LM0 interrupt and the GCSR LM1 interrupt.

LM0 LEVEL These bits define the level of the GCSR LM0 interrupt.

LM1 LEVEL These bits define the level of the GCSR LM1 interrupt.

Interrupt Level Register 3 (bits 24-31)

ADR/SIZ	$FFF40080 (8 bits [6 used] of 32)							
BIT	31	30	29	28	27	26	25	24
NAME		SW7 LEVEL				SW6 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the software 6 interrupt and the software 7 interrupt.

SW6 LEVEL These bits define the level of the software 6 interrupt.

SW7 LEVEL These bits define the level of the software 7 interrupt.

Interrupt Level Register 3 (bits 16-23)

ADR/SIZ	$FFF40080 (8 bits [6 used] of 32)							
BIT	23	22	21	20	19	18	17	16
NAME		SW5 LEVEL				SW4 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the software 4 interrupt and the software 5 interrupt.

SW4 LEVEL These bits define the level of the software 4 interrupt.

SW5 LEVEL These bits define the level of the software 5 interrupt.

Interrupt Level Register 3 (bits 8-15)

ADR/SIZ	$FFF40080 (8 bits [6 used] of 32)							
BIT	15	14	13	12	11	10	9	8
NAME		SW3 LEVEL				SW2 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the software 2 interrupt and the software 3 interrupt.

SW2 LEVEL These bits define the level of the software 2 interrupt.

SW3 LEVEL These bits define the level of the software 3 interrupt.

Interrupt Level Register 3 (bits 0-7)

ADR/SIZ	$FFF40080 (8 bits [6 used] of 32)							
BIT	7	6	5	4	3	2	1	0
NAME		SW1 LEVEL				SW0 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the software 0 interrupt and the software 1 interrupt.

SW0 LEVEL These bits define the level of the software 0 interrupt.

SW1 LEVEL These bits define the level of the software 1 interrupt.

Interrupt Level Register 4 (bits 24-31)

ADR/SIZ	$FFF40084 (8 bits [6 used] of 32)							
BIT	31	30	29	28	27	26	25	24
NAME		SPARE LEVEL				VIRQ7 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the VMEbus IRQ7 interrupt and the spare interrupt. The VMEbus level 7 (IRQ7) interrupt may be mapped to any Local Bus interrupt level.

VIRQ7 LEVEL These bits define the level of the VMEbus IRQ7 interrupt.

SPARE LEVEL These bits define the level of the spare interrupt.

Interrupt Level Register 4 (bits 16-23)

ADR/SIZ	$FFF40084 (8 bits [6 used] of 32)							
BIT	23	22	21	20	19	18	17	16
NAME		VIRQ6 LEVEL				VIRQ5 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the VMEbus IRQ5 interrupt and the VMEbus IRQ6 interrupt. The VMEbus level 5 (IRQ5) interrupt and the VMEbus level 6 (IRQ6) interrupt may be mapped to any Local Bus interrupt level.

VIRQ5 LEVEL These bits define the level of the VMEbus IRQ5 interrupt.

VIRQ6 LEVEL These bits define the level of the VMEbus IRQ6 interrupt.

Interrupt Level Register 4 (bits 8-15)

ADR/SIZ	$FFF40084 (8 bits [6 used] of 32)							
BIT	15	14	13	12	11	10	9	8
NAME		VIRQ4 LEVEL				VIRQ3 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the VMEbus IRQ3 interrupt and the VMEbus IRQ4 interrupt. The VMEbus level 3 (IRQ3) interrupt and the VMEbus level 4 (IRQ4) interrupt may be mapped to any Local Bus interrupt level.

VIRQ3 LEVEL These bits define the level of the VMEbus IRQ3 interrupt.

VIRQ4 LEVEL These bits define the level of the VMEbus IRQ4 interrupt.

Interrupt Level Register 4 (bits 0-7)

ADR/SIZ	$FFF40084 (8 bits [6 used] of 32)							
BIT	7	6	5	4	3	2	1	0
NAME		VIRQ2 LEVEL				VIRQ1 LEVEL		
OPER		R/W				R/W		
RESET		0 PSL				0 PSL		

This register is used to define the level of the VMEbus IRQ1 interrupt and the VMEbus IRQ2 interrupt. The VMEbus level 1 (IRQ1) interrupt and the VMEbus level 2 (IRQ2) interrupt may be mapped to any Local Bus interrupt level.

VIRQ1 LEVEL These bits define the level of the VMEbus IRQ1 interrupt.

VIRQ2 LEVEL These bits define the level of the VMEbus IRQ2 interrupt.

Vector Base Register

ADR/SIZ	$FFF40088 (8 bits of 32)							
BIT	31	30	29	28	27	26	25	24
NAME	VBR 0				VBR 1			
OPER	R/W				R/W			
RESET	0 PSL				0 PSL			

This register is used to define the interrupt base vectors.

VBR 1 These bits define the interrupt base vector 1.

VBR 0 These bits define the interrupt base vector 0.

Note Refer to Table 4-4, *Local Bus Interrupter Summary*, earlier in this chapter, for further information.

A suggested setting for the Vector Base Register for the VMEchip2 is: VBR0 = 6, VBR1 = 7 (i.e., setting the Vector Base Register at address $FFF40088 to $67*xxxxxx*). This produces a Vector Base0 of $60 corresponding to the “X” in Table 4-4, and a Vector Base1 of $70 corresponding to the “Y” in Table 4-4.

I/O Control Register 1

ADR/SIZ	$FFF40088 (8 bits of 32)							
BIT	23	22	21	20	19	18	17	16
NAME	MIEN	SYSFL	ACFL	ABRTL	GPOEN3	GPOEN2	GPOEN1	GPOEN0
OPER	R/W	R	R	R	R/W	R/W	R/W	R/W
RESET	0 PSL	X	X	X	0 PS	0 PS	0 PS	0 PS

This register is a general purpose I/O control register.

Bits 16-19 control the direction of the four General Purpose I/O pins (GPIO0-3).

Caution

Refer to the notes for your module in the I/O Control Register 2 description before programming any of the I/O Control Register bits.

GPOEN0 When this bit is low, the GPIO0 pin is an input. When this bit is high, the GPIO0 pin is an output.

GPOEN1 When this bit is low, the GPIO1 pin is an input. When this bit is high, the GPIO1 pin is an output.

GPOEN2 When this bit is low, the GPIO2 pin is an input. When this bit is high, the GPIO2 pin is an output.

GPOEN3 When this bit is low, the GPIO3 pin is an input. When this bit is high, the GPIO3 pin is an output.

ABRTL This bit indicates the status of the ABORT switch. When this bit is high, the ABORT switch is depressed. When this bit is low, the ABORT switch is not depressed.

ACFL This bit indicates the status of the ACFAIL signal line on the VMEbus. When this bit is high, the ACFAIL signal line is active. When this bit is low, the ACFAIL signal line is not active.

SYSFL This bit indicates the status of the SYSFAIL signal line on the VMEbus. When this bit is high, the SYSFAIL signal line is active. When this bit is low, the SYSFAIL signal line is not active.

MIEN When this bit is low, all interrupts controlled by the VMEchip2 are masked. When this bit is high, all interrupts controlled by the VMEchip2 are not masked.

I/O Control Register 2

ADR/SIZ	$FFF40088 (8 bits of 32)							
BIT	15	14	13	12	11	10	9	8
NAME	GPIOO3	GPIOO2	GPIOO1	GPIOO0	GPIOI3	GPIOI2	GPIOI1	GPIOI0
OPER	R/W	R/W	R/W	R/W	R	R	R	R
RESET	0 PSL	0 PS	0 PS	0 PS	X	X	X	X

This register is a general purpose I/O control register.

Bits 8-11 reflect the status of the four General Purpose I/O pins (GPIO0-3).

GPIOI0 When this bit is low, the GPIO0 pin is low. When this bit is high, the GPIO0 pin is high.

GPIOI1 When this bit is low, the GPIO1 pin is low. When this bit is high, the GPIO1 pin is high.

GPIOI2 When this bit is low, the GPIO2 pin is low. When this bit is high, the GPIO2 pin is high.

GPIOI3 When this bit is low, the GPIO3 pin is low. When this bit is high, the GPIO3 pin is high.

Bits 12-15 determine the driven level of the four General Purpose I/O pins (GPIO0-3) when they are defined as outputs.

GPIOO0 When this bit is low, the GPIO0 pin is driven low if it is defined as an output. When this bit is high, the GPIO0 pin is driven high if it is defined as an output.

GPIOO1 When this bit is low, the GPIO1 pin is driven low if it is defined as an output. When this bit is high, the GPIO1 pin is driven high if it is defined as an output.

GPIOO2 When this bit is low, the GPIO2 pin is driven low if it is defined as an output. When this bit is high, the GPIO2 pin is driven high if it is defined as an output.

GPIOO3 When this bit is low, the GPIO3 pin is driven low if it is defined as an output. When this bit is high, the GPIO3 pin is driven high if it is defined as an output.

Notes

MVME166	
GPIO Pin	**Description**
0	Used to monitor power to the I/O transition module (MVME712-10). When the bit is low, +5 Vdc, +12 Vdc, and -12 Vdc are available to the transition board. When high, one of the voltages is not available to the transition board. Should not be programmed as an output.
1	Connected to the Remote Reset connector (pin 16) May be used as an input or an output.
2	Controls bus error handling by the 82596CA interface logic. Refer to the 82596CA LAN Controller Interface section in the chapter, *PCCchip2*.
3	Monitors TERM (termination) power on the SCSI bus. If low, power is present. If high, power is not present. Should not be programmed as an output.

MVME167	
GPIO Pin	**Description**
0	Used to monitor power to the I/O transition module (MVME712-10). When the bit is low, +5 Vdc, +12 Vdc, and -12 Vdc are available to the transition board. When high, one of the voltages is not available to the transition board. Should not be programmed as an output.
1	Connected to the Remote Reset connector (pin 16). May be used as an input or an output.
2	Controls bus error handling by the 82596CA interface logic. Refer to the 82596CA LAN Controller Interface section in the chapter, *PCCchip2*.
3	Connected to the Remote Reset connector (pin 18).

MVME176	
GPIO Pin	**Description**
0	Used to monitor power to the I/O transition module (MVME712-10). When the bit is low, +5 Vdc, +12 Vdc, and -12 Vdc are available to the transition board. When high, one of the voltages is not available to the transition board. Should not be programmed as an output.
1	Flash write enable. Enabled when pin is programmed low. Disabled when pin is programmed high.
2	Flash Switch Control Upper (top) 2 MB Flash used when programmed high. Lower (bottom) 2 MB Flash used when programmed low.
3	Flash/EPROM jumper (input *only*) Flash and EPROM when pin is set low. Flash only when pin is set high.

MVME177	
GPIO Pin	**Description**
0	Used to monitor the +12 Vdc power to the LAN connector. When the GPIOI0 bit is low, +12 Vdc is present. When the GPIOI0 bit is high, +12Vdc is not present. Should not be programmed as an output
1	Flash write enable. Enabled when pin is programmed low. Disabled when pin is programmed high.
2	Flash Switch Control Upper (top) 2 MB Flash used when programmed high. Lower (bottom) 2 MB Flash used when programmed low.
3	Flash/EPROM jumper (input *only*) Flash and EPROM when pin is set low. Flash only when pin is set high.

MVME187	
GPIO Pin	**Description**
0	Used to monitor the +12 Vdc power to the LAN connector: When the GPIOI0 bit is high, +12Vdc is not present. When the GPIOI0 bit is low, +12 Vdc is present. Should not be programmed as an output.
1	Controls STAT LED: ON when pin is programmed high or as an input. OFF when pin is programmed low. Normally an output. If used as an input, STAT LED follows the input.
2	Controls bus error handling by the 82596CA interface logic. Refer to the 82596CA LAN Controller Interface section in the chapter, *PCCchip2*.
3	Connected to the Remote Reset connector (pin 18). May be used as an input or output.

I/O Control Register 3

ADR/SIZ	$FFF40088 (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
NAME	GPI7	GPI6	GPI5	GPI4	GPI3	GPI2	GPI1	GPI0
OPER	R	R	R	R	R	R	R	R
RESET	X	X	X	X	X	X	X	X

This register reflects the status of the eight General Purpose Input pins (GPI0-7). (Chapter 2, *Hardware Configuration,* describes the hardware jumpering of these pins.)

GPI0 When this bit is low, the GPI0 pin is low. When this bit is high, the GPI0 pin is high.

GPI1 When this bit is low, the GPI1 pin is low. When this bit is high, the GPI1 pin is high.

GPI2 When this bit is low, the GPI2 pin is low. When this bit is high, the GPI2 pin is high.

GPI3 When this bit is low, the GPI3 pin is low. When this bit is high, the GPI3 pin is high.

On the MVME166 only, if the jumper is in, you can execute 166Bug from the Flash memory; if the jumper is out, the 166Bug stays in the Boot ROM.

On the MVME176/177, the pins are removed and the signal is reserved.

GPI4 When this bit is low, the GPI4 pin is low. When this bit is high, the GPI4 pin is high.

GPI5 When this bit is low, the GPI5 pin is low. When this bit is high, the GPI5 pin is high.

GPI6 When this bit is low, the GPI6 pin is low. When this bit is high, the GPI6 pin is high.

GPI7 When this bit is low, the GPI7 pin is low. When this bit is high, the GPI7 pin is high.

Miscellaneous Control Register

ADR/SIZ	$FFF4008C (8 bits of 32)							
BIT	7	6	5	4	3	2	1	0
NAME	MPIRQEN	REVEROM	DISSRAM	DISMST	NOELBBSY	DISBSYT	ENINT	DISBGN
OPER	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
RESET	0 PSL	0 PSL	0 PSL	0 PS	0 PS	0 PS	0 PS	0 PS

DISBGN When this bit is high, the VMEbus BGIN filters are disabled. When this bit is low, the VMEbus BGIN filters are enabled. This bit should not be set.

ENINT When this bit is high, the Local Bus interrupt filters are enabled. When this bit is low, the Local Bus interrupt filters are disabled. This bit should not be set.

DISBSYT When this bit is low, the minimum VMEbus BBSY* time when the Local Bus master has been retried off the Local Bus is 32 Local Bus clocks.

When this bit is high, the minimum VMEbus BBSY* time when the Local Bus master has been retried off the Local Bus is 3 Local Bus clocks.

When a Local Bus master attempts to access the VMEbus and a VMEbus master attempts to access the Local Bus, a deadlock is created. The VMEchip2 detects this condition and requests the Local Bus master to give up the Local Bus and retry the cycle. This allows the VMEbus master to complete the cycle to the Local Bus.

If the VMEchip2 receives VMEbus mastership, the local master has not returned from the retry, and this bit is **high**, VMEchip2 drives VMEbus BBSY* for the minimum time (about 90 ns) and then releases the VMEbus. If the local master does not return from the retry within this 90 ns window, the board loses its turn on the VMEbus.

If the VMEchip2 receives VMEbus mastership, the local master has not returned from the retry, and this bit is **low**, VMEchip2 drives VMEbus BBSY* for a minimum of 32 Local Bus clocks, which allows the Local Bus master time to return from the retry and the board does not lose its turn on the VMEbus. For this reason, it is recommended that this bit remain low.

NOELBBSY When this bit is high, the early release feature of bus busy feature on the VMEbus is disabled. The VMEchip2 drives BBSY* low whenever VMEbus AS* is low.

When this bit is low, the early release feature of bus busy feature on the VMEbus is not disabled.

DISMST When this bit is high, the VME LED on the Single Board Computers is lit when Local Bus reset is asserted or the VMEchip2 is driving Local Bus busy.

When this bit is low, the VME LED is lit when Local Bus reset is asserted, the VMEchip2 is driving Local Bus busy, or the VMEchip2 is driving the VMEbus address strobe.

DISSRAM When this bit is high, the SRAM decoder in the VMEchip2 is disabled.

When this bit is low, the SRAM decoder in the VMEchip2 is enabled. This bit should not be set.

REVEROM MVME167/176/177/187: when this bit is high, the EPROM is disabled; when this bit is low, the EPROM is enabled.This bit should not be set.

MVME166: This bit is used to enable/disable the FLASH ROM. When this bit is high, the FLASH ROM is enabled. When this bit is low, the FLASH ROM is disabled. Set this bit after the ROM0 bit is cleared ($FFF40030 bit 20).

MPIRQEN This function is not used. This bit must not be set.

GCSR Programming Model

This section describes the programming model for the Global Control and Status Registers (GCSR) in the VMEchip2. The Local Bus map decoder for the GCSR registers is included in the VMEchip2. The Local Bus base address for the GCSR is $FFF40100. The registers in the GCSR are 16 bits wide and they are byte accessible from both the VMEbus and the Local Bus. The GCSR is located in the 16-bit VMEbus short I/O space and it responds to address modifier codes $29 or $2D. The address of the GCSR as viewed from the VMEbus depends upon the GCSR group select value XX and GCSR board select value Y programmed in the LCSR. The board value Y may be $0 through $E, allowing 15 boards in one group. The value $F is reserved for the location monitors.

The VMEchip2 includes four location monitors (LM0-LM3). The location monitors provide a broadcast signaling capability on the VMEbus. When a location monitor address is generated on the VMEbus, all location monitors in the group are cleared. The signal interrupts SIG0-SIG3 should be used to signal individual boards. The location monitors are located in the VMEbus short I/O space and the specific address is determined by the VMEchip2 group address. The location monitors LM0-LM3 are located at addresses $XXF1, $XXF3, $XXF5, and $XXF7 respectively. A location monitor cycle on the VMEbus is generated by a read or write to VMEbus short I/O address $XXFN, where XX is the group address and N is the specific location monitor address. When the VMEchip2 generates a location monitor cycle to the VMEbus, within its own group, the VMEchip2 DTACKs itself. A VMEchip2 cannot DTACK location monitor cycles to other groups.

The GCSR section of the VMEchip2 contains the following registers: a *chip ID register*, a *chip revision register*, a *location monitor status registe*r, an *interrupt control register*, a *board control register*, and six *general purpose registers*.

The *chip ID* and *revision registers* are provided to allow software to determine the ID of the chip and its revision level. The VMEchip2 has a chip ID of ten. ID codes zero and one are used by the old VMEchip. The initial revision of the VMEchip2 is zero. If mask changes are required, the revision level is incremented.

The *location monitor status register* provides the status of the location monitors. A location monitor bit is cleared when the VMEchip2 detects a VMEbus cycle to the corresponding location monitor address. When the LM0 or LM1 bits are cleared, an interrupt is set to the Local Bus interrupter. If the LM0 or LM1 interrupt is enabled in the Local Bus interrupter, then a Local Bus interrupt is generated. The location monitor bits are set by writing a one to the corresponding bit in the location monitor register. LM0 and LM1 can also be set by writing a one to the corresponding clear bits in the local interrupt clear register.

The *interrupt control register* provides four bits that allow the VMEbus to interrupt the Local Bus. An interrupt is sent to the Local Bus interrupter when one of the bits is set. If the interrupt is enabled in the Local Bus interrupter, then a Local Bus interrupt is generated. The interrupt bits are cleared by writing a one to the corresponding bit in the interrupt clear register.

The *board control register* allows a VMEbus master to reset the Local Bus, prevent the VMEchip2 from driving the SYSFAIL signal line, and detect if the VMEchip2 wants to drive the SYSFAIL signal line.

The six *general purpose registers* can be read and written from both the Local Bus and the VMEbus. These registers are provided to allow Local Bus masters to communicate with VMEbus masters. The function of these registers is not defined by this specification. The GCSR supports read-modify-write cycles such as TAS.

Caution

The GCSR allows a VMEbus master to reset the Local Bus. This feature is very dangerous and should be used with caution. The local reset feature is a partial system reset, not a complete system reset such as power-up reset or SYSRESET. When the Local Bus reset signal is

asserted, a Local Bus cycle may be aborted. The VMEchip2 is connected to both the Local Bus and the VMEbus and if the aborted cycle is bound for the VMEbus, erratic operation may result. Communications between the local processor and a VMEbus master should use interrupts or mailbox locations; reset should not be used in normal communications. Reset should be used only when the local processor is halted or the Local Bus is hung and reset is the last resort.

Programming the GCSR

A complete description of the GCSR is provided in the following tables. Each register definition includes a table with 5 lines:

- Line 1 is the base address of the register as viewed from the Local Bus and as viewed from the VMEbus, and the number of bits defined in the table.
- Line 2 shows the bits defined by this table.
- Line 3 defines the name of the register or the name of the bits in the register.
- Line 4 defines the operations possible on the register bits as follows:

  R This bit is a read-only status bit.
  R/W This bit is readable and writable.
  S/R Writing a one to this bit sets it. Reading it returns its current status.

- Line 5 defines the state of the bit following a reset as defined below:

  P This bit is affected by power-up reset.
  S The bit is affected by SYSRESET.
  L The bit is affected by Local Bus reset.
  X The bit is not affected by reset.

A summary of the GCSR is shown in Table 4-5.

Table 4-5. VMEchip2 Memory Map (GCSR Summary)

VMEchip2 GCSR Base Address = $FFF40100

Offsets		Bit Numbers															
VME -bus	**Local Bus**	**15**	**14**	**13**	**12**	**11**	**10**	**9**	**8**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	0	CHIP REVISION								CHIP ID							
2	4	LM3	LM2	LM1	LM0	SIG3	SIG2	SIG1	SIG0	RST	ISF	BF	SCON	SYSFL	X	X	X
4	8	GENERAL PURPOSE CONTROL AND STATUS REGISTER 0															
6	C	GENERAL PURPOSE CONTROL AND STATUS REGISTER 1															
8	10	GENERAL PURPOSE CONTROL AND STATUS REGISTER 2															
A	14	GENERAL PURPOSE CONTROL AND STATUS REGISTER 3															
C	18	GENERAL PURPOSE CONTROL AND STATUS REGISTER 4															
E	1C	GENERAL PURPOSE CONTROL AND STATUS REGISTER 5															

VMEchip2 Revision Register

ADR/SIZ	Local bus: $FFF40100/VMEbus: $XXY0 (8 bits)		
BIT	15	. . .	8
NAME	VMEchip2 Revision Register		
OPER	R		
RESET	01 PS		

This register is the VMEchip2 revision register. The revision level for the VMEchip2 starts at zero and is incremented if mask changes are required.

VMEchip2 ID Register

ADR/SIZ	Local bus: $FFF40100/VMEbus: $XXY0 (8 bits)		
BIT	7	. . .	0
NAME	VMEchip2 ID Register		
OPER	R		
RESET	10 PS		

This register is the VMEchip2 ID register. The ID for the VMEchip2 is 10.

VMEchip2 LM/SIG Register

ADR/SIZ	Local Bus: $FFF40104/VMEbus: $XXY2 (8 bits)							
BIT	15	14	13	12	11	10	9	8
NAME	LM3	LM2	LM1	LM0	SIG3	SIG2	SIG1	SIG0
OPER	R	R	R	R	S/R	S/R	S/R	S/R
RESET	1 PS	1 PS	1 PS	1 PS	0 PS	0 PS	0 PS	0 PS

This register is the VMEchip2 location monitor register and the interrupt register.

SIG0 The SIG0 bit is set when a VMEbus master writes a one to it. When the SIG0 bit is set, an interrupt is sent to the Local Bus interrupter. The SIG0 bit is cleared when the local processor writes a one to the SIG0 bit in this register or the CSIG0 bit in the local interrupt clear register.

SIG1 The SIG1 bit is set when a VMEbus master writes a one to it. When the SIG1 bit is set, an interrupt is sent to the Local Bus interrupter. The SIG1 bit is cleared when the local processor writes a one to the SIG1 bit in this register or the CSIG1 bit in the local interrupt clear register.

SIG2 The SIG2 bit is set when a VMEbus master writes a one to it. When the SIG2 bit is set, an interrupt is sent to the Local Bus interrupter. The SIG2 bit is cleared when the local processor writes a one to the SIG2 bit in this register or the CSIG2 bit in the local interrupt clear register.

SIG3 The SIG3 bit is set when a VMEbus master writes a one to it. When the SIG3 bit is set, an interrupt is sent to the Local Bus interrupter. The SIG3 bit is cleared when the local processor writes a one to the SIG3 bit in this register or the CSIG3 bit in the local interrupt clear register.

LM0 This bit is cleared by an LM0 cycle on the VMEbus. When this bit is cleared, an interrupt is set to the Local Bus interrupter. This bit is set when the local processor or a VMEbus master writes a one to the LM0 bit in this register or the CLM0 bit in local interrupt clear register.

LM1 This bit is cleared by an LM1 cycle on the VMEbus. When this bit is cleared, an interrupt is set to the Local Bus interrupter. This bit is set when the local processor or a VMEbus master writes a one to the LM1 bit in this register or the CLM1 bit in local interrupt clear register.

LM2 This bit is cleared by an LM2 cycle on the VMEbus. This bit is set when the local processor or a VMEbus master writes a one to the LM0 bit in this register.

LM3 This bit is cleared by an LM3 cycle on the VMEbus. This bit is set when the local processor or a VMEbus master writes a one to the LM3 bit in this register.

VMEchip2 Board Status/Control Register

ADR/SIZ	Local Bus: $FFF40104/VMEbus: $XXY2 (8 bits [5 used])							
BIT	7	6	5	4	3	2	1	0
NAME	RST	ISF	BF	SCON	SYSFL			
OPER	S/R	R/W	R	R	R			
RESET	0 PSL	0 PSL	1 PS	X	1 PSL			

This register is the VMEchip2 board status/control register.

SYSFL This bit is set when the VMEchip2 is driving the SYSFAIL signal.

SCON This bit is set if the VMEchip2 is system controller.

BF When this bit is high, the Board Fail signal is active. When this bit is low, the Board Fail signal is inactive. When this bit is set, the VMEchip2 drives SYSFAIL if the inhibit SYSFAIL bit is not set.

ISF When this bit is set, the VMEchip2 is prevented from driving the VMEbus SYSFAIL signal line. When this bit is cleared, the VMEchip2 is allowed to drive the VMEbus SYSFAIL signal line.

RST This bit allows a VMEbus master to reset the Local Bus. Refer to the note on local reset in the *GCSR Programming Model* section, earlier in this chapter. When this bit is set, a Local Bus reset is generated. This bit is cleared by the Local Bus reset.

General Purpose Register 0

ADR/SIZ	Local Bus: $FFF40108/VMEbus: $XXY4 (16 bits)		
BIT	15	. . .	0
NAME	General Purpose Register 0		
OPER	R/W		
RESET	0 PS		

This register is a general purpose register that allows a Local Bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

General Purpose Register 1

ADR/SIZ	Local Bus: $FFF4010C/VMEbus: $XXY6 (16 bits)		
BIT	15	. . .	0
NAME	General Purpose Register 1		
OPER	R/W		
RESET	0 PS		

This register is a general purpose register that allows a Local Bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

General Purpose Register 2

ADR/SIZ	Local Bus: $FFF40110/VMEbus: $XXY8 (16 bits)		
BIT	15	. . .	0
NAME	General Purpose Register 2		
OPER	R/W		
RESET	0 PS		

This register is a general purpose register that allows a Local Bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

General Purpose Register 3

ADR/SIZ	Local Bus: $FFF40114/VMEbus: $XXYA (16 bits)		
BIT	15	. . .	0
NAME	General Purpose Register 3		
OPER	R/W		
RESET	0 PS		

This register is a general purpose register that allows a Local Bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

General Purpose Register 4

ADR/SIZ	Local Bus: $FFF40118/VMEbus: $XXYC (16 bits)		
BIT	15	. . .	0
NAME	General Purpose Register 4		
OPER	R/W		
RESET	0 PS		

This register is a general purpose register that allows a Local Bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

General Purpose Register 5

ADR/SIZ	Local Bus: $FFF4011C/VMEbus: $XXYE (16 bits)		
BIT	15	. . .	0
NAME	General Purpose Register 5		
OPER	R/W		
RESET	0 PS		

This register is a general purpose register that allows a Local Bus master to communicate with a VMEbus master. The function of this register is not defined by the hardware specification.

Index

D

G

H

I

M

INDEX

INDEX

R

INDEX

S

INDEX

W

Z