

**MCP750 Series Single Board
Computer**

**Programmer's Reference
Guide**

MCP750A/PG6

June 2001 Edition

© Copyright 1999, 2000, 2001 Motorola, Inc.

All rights reserved.

Printed in the United States of America.

PowerStack™ is a trademark of Motorola, Inc.

PowerPC™, PowerPC 603™, and PowerPC 604™ are trademarks of IBM Corp, and are used by Motorola, Inc. under license from IBM Corp.

AIX™ is a trademark of IBM Corp.

Timekeeper™ and Zeropower™ are trademarks of Thompson Components.

Motorola and the Motorola symbol are registered trademarks of Motorola, Inc.

All other products mentioned in this document are trademarks or registered trademarks of their respective holders.

Safety Summary

The following general safety precautions must be observed during all phases of operation, service, and repair of this equipment. Failure to comply with these precautions or with specific warnings elsewhere in this manual could result in personal injury or damage to the equipment.

The safety precautions listed below represent warnings of certain dangers of which Motorola is aware. You, as the user of the product, should follow these warnings and all other safety precautions necessary for the safe operation of the equipment in your operating environment.

Ground the Instrument.

To minimize shock hazard, the equipment chassis and enclosure must be connected to an electrical ground. If the equipment is supplied with a three-conductor AC power cable, the power cable must be plugged into an approved three-contact electrical outlet, with the grounding wire (green/yellow) reliably connected to an electrical ground (safety ground) at the power outlet. The power jack and mating plug of the power cable meet International Electrotechnical Commission (IEC) safety standards and local electrical regulatory codes.

Do Not Operate in an Explosive Atmosphere.

Do not operate the equipment in any explosive atmosphere such as in the presence of flammable gases or fumes. Operation of any electrical equipment in such an environment could result in an explosion and cause injury or damage.

Keep Away From Live Circuits Inside the Equipment.

Operating personnel must not remove equipment covers. Only Factory Authorized Service Personnel or other qualified service personnel may remove equipment covers for internal subassembly or component replacement or any internal adjustment. Service personnel should not replace components with power cable connected. Under certain conditions, dangerous voltages may exist even with the power cable removed. To avoid injuries, such personnel should always disconnect power and discharge circuits before touching components.

Use Caution When Exposing or Handling a CRT.

Breakage of a Cathode-Ray Tube (CRT) causes a high-velocity scattering of glass fragments (implosion). To prevent CRT implosion, do not handle the CRT and avoid rough handling or jarring of the equipment. Handling of a CRT should be done only by qualified service personnel using approved safety mask and gloves.

Do Not Substitute Parts or Modify Equipment.

Do not install substitute parts or perform any unauthorized modification of the equipment. Contact your local Motorola representative for service and repair to ensure that all safety features are maintained.

Observe Warnings in Manual.

Warnings, such as the example below, precede potentially dangerous procedures throughout this manual. Instructions contained in the warnings must be followed. You should also employ all other safety precautions which you deem necessary for the operation of the equipment in your operating environment.



To prevent serious injury or death from dangerous voltages, use extreme caution when handling, testing, and adjusting this equipment and its components.

Flammability

All Motorola PWBs (printed wiring boards) are manufactured with a flammability rating of 94V-0 by UL-recognized manufacturers.

EMI Caution



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

Lithium Battery Caution

This product contains a lithium battery to power the clock and calendar circuitry.



Danger of explosion if battery is replaced incorrectly. Replace battery only with the same or equivalent type recommended by the equipment manufacturer. Dispose of used batteries according to the manufacturer's instructions.



Il y a danger d'explosion s'il y a remplacement incorrect de la batterie. Remplacer uniquement avec une batterie du même type ou d'un type équivalent recommandé par le constructeur. Mettre au rebut les batteries usagées conformément aux instructions du fabricant.



Explosionsgefahr bei unsachgemäßem Austausch der Batterie. Ersatz nur durch denselben oder einen vom Hersteller empfohlenen Typ. Entsorgung gebrauchter Batterien nach Angaben des Herstellers.

CE Notice (European Community)

Motorola Computer Group products with the CE marking comply with the EMC Directive (89/336/EEC). Compliance with this directive implies conformity to the following European Norms:

EN55022 “Limits and Methods of Measurement of Radio Interference Characteristics of Information Technology Equipment”; this product tested to Equipment Class B

EN55024 “Information Technology Equipment-Immunity characteristics-Limits and methods of measurement”

Board products are tested in a representative system to show compliance with the above mentioned requirements. A proper installation in a CE-marked system will maintain the required EMC/safety performance.

In accordance with European Community directives, a “Declaration of Conformity” has been made and is available on request. Please contact your sales representative.

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.

Electronic versions of this material may be read online, downloaded for personal use, or referenced in another document as a URL to the Motorola Computer Group website. The text itself may not be published commercially in print or electronic form, edited, translated, or otherwise altered without the permission of Motorola, Inc.

It is possible that this publication may contain reference to or information about Motorola products (machines and programs), programming, or services that are not available 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.

Limited and 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 (b)(3) of the Rights in Technical Data clause at DFARS 252.227-7013 (Nov. 1995) and of the Rights in Noncommercial Computer Software and Documentation clause at DFARS 252.227-7014 (Jun. 1995).

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

Contents

About This Manual

| | |
|---------------------------------------|-------|
| Summary of Changes | xvi |
| Overview of Contents | xvi |
| Manual Terminology | xvii |
| Comments and Suggestions | xviii |
| Conventions Used in This Manual | xix |

CHAPTER 1 Board Description and Memory Maps

| | |
|------------------------------------------------------------|------|
| Introduction | 1-1 |
| Overview | 1-1 |
| Feature Summary | 1-2 |
| System Block Diagram | 1-2 |
| Functional Description | 1-5 |
| Overview | 1-5 |
| Programming Model | 1-6 |
| Memory Maps | 1-6 |
| Processor Memory Maps | 1-6 |
| PCI Memory Maps | 1-9 |
| CompactPCI Bus Mapping | 1-11 |
| DEC 21154 Transaction Forwarding | 1-11 |
| Upstream PCI Transactions | 1-11 |
| Falcon-Controlled System Registers | 1-11 |
| Falcon Registers | 1-12 |
| Falcon-Controlled System Registers | 1-12 |
| System Configuration Register (SYSCR) | 1-13 |
| Memory Configuration Register (MEMCR) | 1-15 |
| System External Cache Control Register (SXCCR) | 1-18 |
| Processor 0 External Cache Control Register (P0XCCR) | 1-18 |
| Processor 1 External Cache Control Register (P1XCCR) | 1-19 |
| CPU Control Register | 1-19 |
| ISA Local Resource Bus | 1-20 |
| VT82C586B PBC | 1-20 |
| PC87307VUL Super I/O Strapping | 1-20 |
| NVRAM/RTC & Watchdog Timer Registers | 1-21 |
| SROM | 1-22 |

| | |
|---------------------------------------------------------|------|
| Module Configuration and Status Registers..... | 1-22 |
| CPU Configuration Register | 1-23 |
| Base Module Feature Register | 1-24 |
| Base Module Status Register (BMSR)..... | 1-24 |
| Base Module Extended Features Register..... | 1-25 |
| SYSLED Status Register..... | 1-26 |
| Board Status Register..... | 1-27 |
| Board Last Reset Register..... | 1-28 |
| Z85230 ESCC and Z8536 CIO Registers and Port Pins | 1-28 |
| Z8536/Z85230 Registers | 1-29 |
| Z8536 CIO Port Pins | 1-30 |
| ISA DMA Channels | 1-32 |

CHAPTER 2 Raven PCI Host Bridge & Multi-Processor Interrupt Controller

| | |
|--------------------------------------------------|------|
| Introduction | 2-1 |
| Overview | 2-1 |
| Capability | 2-1 |
| Features | 2-2 |
| Block Diagram..... | 2-3 |
| Functional Description | 2-4 |
| PPC Bus Interface | 2-4 |
| PPC Map Decoders | 2-4 |
| PPC Write Posting..... | 2-5 |
| PPC Master..... | 2-6 |
| PPC Bus Timer..... | 2-8 |
| PCI Interface..... | 2-9 |
| PCI Map Decoders | 2-9 |
| PCI Configuration Space..... | 2-10 |
| PCI Write Posting | 2-10 |
| PCI Master | 2-11 |
| Generating PCI Memory and I/O Cycles | 2-11 |
| Generating PCI Configuration Cycles..... | 2-13 |
| Generating PCI Special Cycles | 2-13 |
| Generating PCI Interrupt Acknowledge Cycles..... | 2-14 |
| Endian Conversion | 2-14 |
| When PPC Devices are Big-Endian | 2-14 |
| When PPC Devices are Little-Endian | 2-15 |
| Cycles Originating From PCI..... | 2-16 |
| Error Handling..... | 2-16 |
| Watchdog Timers..... | 2-18 |

| | |
|-------------------------------------------------------------|------|
| PCI/PPC Contention Handling | 2-20 |
| Transaction Ordering | 2-21 |
| Registers..... | 2-22 |
| PPC Registers | 2-23 |
| Vendor ID/Device ID Registers | 2-24 |
| Revision ID Register | 2-25 |
| General Control-Status/Feature Registers | 2-25 |
| Prescaler Adjust Register..... | 2-27 |
| PPC Error Enable Register | 2-28 |
| PPC Error Status Register..... | 2-30 |
| PPC Error Address Register | 2-32 |
| PPC Error Attribute Register - MERAT..... | 2-32 |
| PCI Interrupt Acknowledge Register | 2-34 |
| PPC Slave Address (0, 1 and 2) Registers..... | 2-35 |
| PPC Slave Address (3) Register | 2-35 |
| PPC Slave Offset/Attribute (0, 1 and 2) Registers | 2-36 |
| PPC Slave Offset/Attribute (3) Registers | 2-37 |
| WDTxCNTL Registers..... | 2-38 |
| WDTxSTAT Registers | 2-40 |
| General Purpose Registers..... | 2-41 |
| PCI Registers | 2-41 |
| Vendor ID/Device ID Registers | 2-43 |
| PCI Command/Status Registers..... | 2-43 |
| Revision ID/Class Code Registers..... | 2-45 |
| Header Type Register | 2-45 |
| I/O Base Register..... | 2-46 |
| Memory Base Register | 2-46 |
| PCI Slave Address (0, 1, 2 and 3) Registers..... | 2-47 |
| PCI Slave Attribute/ Offset (0, 1, 2 and 3) Registers | 2-48 |
| CONFIG_ADDRESS | 2-49 |
| PCI I/O CONFIG_ADDRESS Register | 2-49 |
| PCI I/O CONFIG_DATA Register | 2-51 |
| Raven Interrupt Controller Implementation..... | 2-51 |
| Introduction..... | 2-51 |
| The Raven Interrupt Controller (Raven MPIC) Features | 2-51 |
| Architecture | 2-52 |
| CSR's Readability | 2-52 |
| Interrupt Source Priority | 2-53 |
| Processor's Current Task Priority..... | 2-53 |
| Nesting of Interrupt Events..... | 2-53 |
| Spurious Vector Generation | 2-53 |

| | |
|--------------------------------------------------------|------|
| Interprocessor Interrupts (IPI) | 2-54 |
| 8259 Compatibility | 2-54 |
| Raven-Detected Errors | 2-54 |
| Timers | 2-55 |
| Interrupt Delivery Modes | 2-55 |
| Block Diagram Description | 2-56 |
| Program Visible Registers | 2-56 |
| Interrupt Pending Register (IPR) | 2-56 |
| Interrupt Selector (IS) | 2-58 |
| Interrupt Request Register (IRR) | 2-58 |
| In-Service Register (ISR) | 2-58 |
| Interrupt Router | 2-59 |
| MPIC Registers | 2-60 |
| Raven MPIC Registers | 2-60 |
| Feature Reporting Register | 2-64 |
| Global Configuration Register | 2-65 |
| Vendor Identification Register | 2-66 |
| Processor Init Register | 2-66 |
| IPI Vector/Priority Registers | 2-67 |
| Spurious Vector Register | 2-68 |
| Timer Frequency Register | 2-68 |
| Timer Current Count Registers | 2-69 |
| Timer Basecount Registers | 2-70 |
| Timer Vector/Priority Registers | 2-71 |
| Timer Destination Registers | 2-72 |
| External Source Vector/Priority Registers | 2-72 |
| External Source Destination Registers | 2-74 |
| Raven-Detected Errors Vector/Priority Register | 2-74 |
| Raven-Detected Errors Destination Register | 2-75 |
| Interprocessor Interrupt Dispatch Registers | 2-76 |
| Current Task Priority Registers | 2-77 |
| Interrupt Acknowledge Registers | 2-77 |
| End-of-Interrupt Registers | 2-78 |
| Programming Notes | 2-78 |
| Disabling Watchdog Timers | 2-78 |
| External Interrupt Service | 2-79 |
| Reset State | 2-80 |
| Operation | 2-80 |
| Interprocessor Interrupts | 2-80 |
| Dynamically Changing I/O Interrupt Configuration | 2-81 |
| EOI Register | 2-81 |
| Interrupt Acknowledge Register | 2-81 |

| | |
|-----------------------------------|------|
| 8259 Mode | 2-82 |
| Current Task Priority Level | 2-82 |
| Architectural Notes | 2-82 |

CHAPTER 3 Falcon ECC Memory Controller Chip Set

| | |
|-------------------------------------------------------------|------|
| Introduction | 3-1 |
| Bit Ordering Convention | 3-1 |
| Features | 3-1 |
| Block Diagrams | 3-2 |
| Functional Description | 3-6 |
| Performance | 3-6 |
| Four-beat Reads/Writes | 3-6 |
| Single-beat Reads/Writes | 3-7 |
| DRAM Speeds | 3-7 |
| ROM/Flash Speeds | 3-11 |
| PowerPC 60x Bus Interface | 3-13 |
| Responding to Address Transfers | 3-13 |
| Completing Data Transfers | 3-13 |
| Data Parity | 3-13 |
| Cache Coherency | 3-14 |
| Cache Coherency Restrictions | 3-14 |
| L2 Cache Support | 3-14 |
| ECC | 3-15 |
| Cycle Types | 3-15 |
| Error Reporting | 3-15 |
| Error Logging | 3-17 |
| ROM/Flash Interface | 3-17 |
| Refresh/Scrub | 3-21 |
| Blocks A and/or B Present, Blocks C and D Not Present | 3-21 |
| Blocks A and/or B Present, Blocks C and/or D Present | 3-22 |
| Chip Defaults | 3-23 |
| External Register Set | 3-23 |
| CSR Accesses | 3-24 |
| Programming Model | 3-25 |
| CSR Architecture | 3-25 |
| Register Summary | 3-30 |
| Detailed Register Bit Descriptions | 3-32 |
| Vendor/Device Register | 3-32 |
| Revision ID/General Control Register | 3-33 |
| DRAM Attributes Register | 3-35 |

| | |
|------------------------------------------|------|
| DRAM Base Register..... | 3-37 |
| CLK Frequency Register..... | 3-37 |
| ECC Control Register | 3-38 |
| Error Logger Register | 3-42 |
| Error_Address Register | 3-44 |
| Scrub/Refresh Register..... | 3-45 |
| Refresh/Scrub Address Register | 3-46 |
| ROM A Base/Size Register..... | 3-47 |
| ROM B Base/Size Register..... | 3-50 |
| ROM Speed Control Register | 3-52 |
| Data Parity Error Logger Register | 3-53 |
| Data Parity Error Address Register..... | 3-55 |
| Data Parity Error Data Register | 3-55 |
| 32-Bit Counter..... | 3-56 |
| Power-Up Reset Status Register 1 | 3-56 |
| Power-Up Reset Status Register 2 | 3-57 |
| External Register Set..... | 3-57 |
| Software Considerations..... | 3-58 |
| Parity Checking on the PowerPC Bus | 3-58 |
| Programming ROM/Flash Devices | 3-58 |
| Writing to the Control Registers..... | 3-58 |
| Sizing DRAM..... | 3-59 |
| ECC Codes | 3-62 |
| Data Paths | 3-64 |

CHAPTER 4 Programming Details

| | |
|--------------------------------------|------|
| Introduction | 4-1 |
| PCI Arbitration | 4-1 |
| Interrupt Handling | 4-1 |
| Raven MPIC | 4-3 |
| 8259 Interrupts | 4-4 |
| ISA DMA Channels..... | 4-7 |
| Exceptions | 4-7 |
| Sources of Reset | 4-7 |
| Soft Reset | 4-8 |
| Error Notification and Handling..... | 4-8 |
| Endian Issues | 4-10 |
| Processor/Memory Domain..... | 4-11 |
| Raven's Involvement..... | 4-12 |
| PCI Domain..... | 4-12 |

| | |
|--------------------------------|------|
| PCI-Ethernet | 4-12 |
| ROM/Flash Initialization | 4-12 |
| Determining CPU Type | 4-13 |
| Determining Base Type | 4-13 |

APPENDIX A Related Documentation

| | |
|-----------------------------------------|-----|
| Motorola Computer Group Documents | A-1 |
| Manufacturers' Documents | A-1 |
| Related Specifications | A-4 |
| URLs | A-6 |

List of Figures

| | |
|-----------------------------------------------------------------------------|------|
| Figure 1-1. MCP750 Series System Block Diagram | 1-4 |
| Figure 2-1. Raven Block Diagram | 2-3 |
| Figure 2-2. PCI Spread I/O Cycle Mapping | 2-12 |
| Figure 2-3. Big- to Little-Endian Data Swap | 2-15 |
| Figure 2-4. Raven MPIC Block Diagram | 2-57 |
| Figure 3-1. Falcon Pair Used with DRAM in a System | 3-3 |
| Figure 3-2. Falcon Internal Data Paths (Simplified) | 3-4 |
| Figure 3-3. Overall DRAM Connections | 3-5 |
| Figure 3-4. Data Path for Reads from the Falcon Internal CSRs | 3-25 |
| Figure 3-5. Data Path for Writes to the Falcon Internal CSRs | 3-26 |
| Figure 3-6. Memory Map for Byte Reads to the CSR | 3-27 |
| Figure 3-7. Memory Map for Byte Writes to the Internal Register Set | 3-28 |
| Figure 3-8. Memory Map for 4-Byte Reads to the CSR | 3-29 |
| Figure 3-9. Memory Map for 4-Byte Writes to the Internal Register Set | 3-29 |
| Figure 3-10. PowerPC Data to DRAM Data Correspondence | 3-65 |
| Figure 4-1. MCP750 Series Interrupt Architecture | 4-2 |
| Figure 4-2. PBC Interrupt Handler Block Diagram | 4-5 |
| Figure 4-3. Big-Endian Mode | 4-10 |
| Figure 4-4. Little-Endian Mode | 4-11 |

List of Tables

| | |
|---------------------------------------------------------------|------|
| Table 1-1. MCP750 Series Features Summary | 1-2 |
| Table 1-2. Default Processor Memory Map..... | 1-7 |
| Table 1-3. PREP Memory Map Example..... | 1-8 |
| Table 1-4. Raven PPC Register Values for PREP Memory Map..... | 1-9 |
| Table 1-5. PCI PREP Memory Map..... | 1-10 |
| Table 1-6. Raven PCI Register Values for PREP Memory Map..... | 1-10 |
| Table 1-7. System Register Summary..... | 1-12 |
| Table 1-8. System Configuration Register..... | 1-13 |
| Table 1-9. System Clock Speed | 1-13 |
| Table 1-10. System External Cache Size | 1-14 |
| Table 1-11. Processor Status Values..... | 1-14 |
| Table 1-12. Memory Configuration Register | 1-15 |
| Table 1-13. Memory Speed | 1-15 |
| Table 1-14. ROM/FLASH Type..... | 1-16 |
| Table 1-15. L2 Memory Type | 1-16 |
| Table 1-16. L2 Core-To-Cache Ratio..... | 1-17 |
| Table 1-17. Bank A FLASH Memory Size | 1-17 |
| Table 1-18. System External Cache Control Register..... | 1-18 |
| Table 1-19. Processor 0 External Cache Control Register..... | 1-18 |
| Table 1-20. Processor 1 External Cache Control Register..... | 1-19 |
| Table 1-21. CPU Control Register | 1-19 |
| Table 1-22. Strap Pins Configuration for the VT82C586B | 1-20 |
| Table 1-23. Strap Pins Configuration for the PC87307VUL | 1-21 |
| Table 1-24. M48T559 Access Registers | 1-21 |
| Table 1-25. AT24C04 Serial EEPROM Device Address Field..... | 1-22 |
| Table 1-26. Module Configuration and Status Registers | 1-23 |
| Table 1-27. Old CPU Configuration Register..... | 1-23 |
| Table 1-28. Base Module Feature Register | 1-24 |
| Table 1-29. Base Module Status Register | 1-24 |
| Table 1-30. Base Module Type | 1-25 |
| Table 1-31. Base Module Extended Features Register | 1-25 |
| Table 1-32. SYSLED Status Register | 1-26 |
| Table 1-33. Board Status Register..... | 1-27 |
| Table 1-34. Board Last Reset Register..... | 1-28 |

| | |
|-------------------------------------------------------------------------------------------------------------------|------|
| Table 1-35. Z85230 Access Registers | 1-29 |
| Table 1-36. Z8536 CIO Port Pins Assignment | 1-30 |
| Table 1-37. Interpretation of MID3-MID0 | 1-31 |
| Table 1-38. PBC DMA Channel Assignments | 1-32 |
| Table 2-1. CHRP Compliant Memory Map..... | 2-4 |
| Table 2-2. PPC Transfer Types | 2-8 |
| Table 2-3. PCI Command Codes | 2-11 |
| Table 2-4. Address Modification for Little Endian Transfers | 2-16 |
| Table 2-5. WDTxCNTL Programming | 2-19 |
| Table 2-6. Raven PPC Register Map | 2-23 |
| Table 2-7. Raven PCI Configuration Register Map | 2-42 |
| Table 2-8. Raven PCI I/O Register Map..... | 2-42 |
| Table 2-9. Raven MPIC Register Map | 2-61 |
| Table 3-1. PowerPC 60x Bus to DRAM Access Timing when Configured for 70ns Fast Page Devices | 3-7 |
| Table 3-2. PowerPC 60x Bus to DRAM Access Timing when Configured for 60ns Fast Page Devices | 3-8 |
| Table 3-3. PowerPC 60x Bus to DRAM Access Timing when Configured for 50ns EDO Devices..... | 3-10 |
| Table 3-4. PowerPC 60x Bus to ROM/Flash Access Timing when configured for 180ns Devices | 3-11 |
| Table 3-5. PowerPC 60x Bus to ROM/Flash Access Timing when configured for 120ns Devices | 3-11 |
| Table 3-6. PowerPC 60x Bus to ROM/Flash Access Timing when configured for 75ns Devices | 3-12 |
| Table 3-7. PowerPC 60x Bus to ROM/Flash Access Timing when configured for 45ns Devices | 3-12 |
| Table 3-8. Error Reporting..... | 3-16 |
| Table 3-9. PowerPC 60x to ROM/Flash Address Mapping when ROM/Flash is 16 Bits Wide (8 Bits per Falcon) | 3-19 |
| Table 3-10. PowerPC 60x to ROM/Flash Address Mapping when ROM/Flash is 64 Bits Wide (32 Bits per Falcon) | 3-20 |
| Table 3-11. Register Summary | 3-30 |
| Table 3-12. ram spd1, ram spd0 and DRAM Type..... | 3-34 |
| Table 3-13. Block_A/B/C/D Configurations | 3-36 |
| Table 3-14. rtest encodings | 3-45 |
| Table 3-15. ROM/Flash Block A Size Encoding | 3-48 |
| Table 3-16. rom_a_rv and rom_b_rv Encoding..... | 3-48 |
| Table 3-17. Read/Write to ROM/Flash..... | 3-49 |

| | |
|----------------------------------------------------------------|------|
| Table 3-18. Rom Speed Bit Encodings | 3-52 |
| Table 3-19. Sizing Addresses | 3-61 |
| Table 3-20. PowerPC 60x Address to DRAM Address Mappings | 3-61 |
| Table 3-21. Syndrome Codes Ordered by Bit in Error | 3-62 |
| Table 3-22. Single-Bit Errors Ordered by Syndrome Code | 3-63 |
| Table 3-23. PowerPC Data to DRAM Data Mapping..... | 3-66 |
| Table 4-1. RavenMPIC Interrupt Assignments..... | 4-3 |
| Table 4-2. PBC ISA Interrupt Assignments..... | 4-6 |
| Table 4-3. Reset Sources and Devices Affected | 4-8 |
| Table 4-4. Error Notification and Handling | 4-9 |
| Table 4-5. ROM/FLASH Bank Default | 4-13 |
| Table A-1. Motorola Computer Group Documents | A-1 |
| Table A-2. Manufacturers' Documents | A-2 |
| Table A-3. Related Specifications | A-4 |

About This Manual

The *MCP750 Series Single Board Computer Programmer's Reference Guide* provides brief board level information, complete memory maps, and detailed ASIC chip information including register bit descriptions for the MCP750 series Single Board Computers.

This manual is intended for anyone who wants to program SBC 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.

The following table is a list of models associated with this manual.

| Model Number | Description |
|--------------|---------------------------------------|
| MCP750-1222 | 233 MHz, 1MB L2, 16MB ECC, TMCP700 |
| MCP750-1222A | 233 MHz, 1MB L2, 16MB ECC, INTR |
| MCP750-1223 | 233 MHz, 1MB L2, 16MB ECC, TMCP700 |
| MCP750-1232 | 233 MHz, 1MB L2, 32MB ECC, TMCP700 |
| MCP750-1242 | 233 MHz, 1MB L2, 64MB ECC, TMCP700 |
| MCP750-1242A | 233 MHz, 1MB L2, 64MB ECC, INTR |
| MCP750-1252 | 233 MHz, 1MB L2, 128MB ECC, TMCP700 |
| MCP750-1252A | 233 MHz, 1MB L2, 128MB ECC, INTR |
| MCP750-1262 | 233 MHz, 1MB L2, 256MB ECC, TMCP700 |
| MCP750-1262A | 233 MHz, 1MB L2, 256MB ECC, INTR |
| MCP750-1332 | 366 MHz, 1MB L2, 32MB ECC, 9MB FLASH |
| MCP750-1342 | 366 MHz, 1MB L2, 64MB ECC, 9MB FLASH |
| MCP750-1352 | 366 MHz, 1MB L2, 128MB ECC, 9MB FLASH |
| MCP750-1362 | 366 MHz, 1MB L2, 256MB ECC, 9MB FLASH |
| MCP750-1432 | MCP750-466, 32MB SDRAM |
| MCP750-1442 | MCP750-466, 64MB SDRAM |
| MCP750-1452 | MCP750-466, 128MB SDRAM |
| MCP750-1462 | 466 MHz, 1MB L2, 256MB ECC DRAM |

Summary of Changes

The following table shows any changes made to this manual since its last two releases.

| Date | Changes |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| January 2001 | Warning added to ensure user is aware of need for boot-up software to take steps to guarantee the DRAM memory component power-up refresh requirements are met. The Warnings can be found on pages 3-35 and 3-59 . |
| April 2001 | Bit SD7 in the "Board Last Reset Register" in Table 1-34 on page 1-28 was changed from PBRST to "Reserved," and the description of the PBRST bit was removed. The wording under the section titled Sources of Reset on page 4-7 was changed to read "six" potential sources of reset, rather than "eight." Two items were deleted from the list immediately following: item 3 (PRST#) and item 5 (FAL#) . Two rows were also deleted in Table 4-3 on page 4-8 : row 3 - PRST# and row 5 - FAL#. |
| June 2001 | Certain Z8536 CIO port pin assignment descriptions were changed. This affected Table 1-36 on page 1-30 . Other minor changes were made to reflect current regulatory specifications. |

Overview of Contents

[Chapter 1, *Board Description and Memory Maps*](#), describes the board level hardware features. It starts with a board level overview and features list and continues on to provide the board's memory maps.

[Chapter 2, *Raven PCI Host Bridge & Multi-Processor Interrupt Controller*](#), describes the architecture and usage of the Raven PCI Host Bridge & Multiprocessor Controller.

[Chapter 3, *Falcon ECC Memory Controller Chip Set*](#), provides a functional description and programming model for the Falcon DRAM controller ASIC.

[Chapter 4, *Programming Details*](#), provides details of several programming functions that are not tied to any specific ASIC chip.

[Appendix A, *Related Documentation*](#), lists all documentation related to this product.

Manual Terminology

Throughout this manual, special character symbols are used to identify the numeric format of data and address parameters. These symbols precede the parameters and identify their 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.

Note In some instances, an underscore character (_) following the signal name is used to indicate an active low signal.

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 for MPC60x chips are defined as follows:

- ❑ A *byte* is eight bits, numbered 0 through 7, with bit 0 being the least significant.
- ❑ A *half-word* is 16 bits, numbered 0 through 15, with bit 0 being the least significant.

-
- ❑ A *word* or *single word* is 32 bits, numbered 0 through 31, with bit 0 being the least significant.
 - ❑ A *double word* is 64 bits, numbered 0 through 63, with bit 0 being the least significant.

Refer to [Chapter 4, *Programming Details*](#), for *Endian Issues*. It covers which parts of the MCP750 series use *big-endian* byte ordering, and which use *little-endian* byte ordering.

The terms *control bit* and *status bit* are used extensively in this document. The term *control bit* is used to describe a bit in a register that can be set and cleared under software control. The term *true* is used to indicate that a bit is in the state that enables the function it controls. The term *false* is used to indicate that the bit is in the state that disables the function it controls. 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 term *status bit* is used to describe a bit in a register that reflects a specific condition. The status bit can be read by software to determine operational or exception conditions.

Comments and Suggestions

Motorola welcomes and appreciates your comments on its documentation. We want to know what you think about our manuals and how we can make them better. Mail comments to:

Motorola Computer Group
Reader Comments DW164
2900 S. Diablo Way
Tempe, Arizona 85282

You can also submit comments to the following e-mail address:
reader-comments@mcg.mot.com

In all your correspondence, please list your name, position, and company. Be sure to include the title and part number of the manual and tell how you used it. Then tell us your feelings about its strengths and weaknesses and any recommendations for improvements.

Conventions Used in This Manual

The following typographical conventions are used in this document:

bold

is used for user input that you type just as it appears; it is also used for commands, options and arguments to commands, and names of programs, directories and files.

italic

is used for names of variables to which you assign values. Italic is also used for comments in screen displays and examples, and to introduce new terms.

`courier`

is used for system output (for example, screen displays, reports), examples, and system prompts.

<**Enter**>, <**Return**> or <**CR**>

<**CR**> represents the carriage return or Enter key.

CTRL

represents the Control key. Execute control characters by pressing the Ctrl key and the letter simultaneously, for example, **Ctrl-d**.

Board Description and Memory Maps

1

Introduction

This manual provides programming information for the MCP750 Single Board Computers (SBCs) based on the MPC750 processor and the RAM300 mezzanine modules. Extensive programming information is provided in this manual for several Application-Specific Integrated Circuit (ASIC) devices used on the boards. Reference information is also included in [Appendix A, Related Documentation](#), for the Large Scale Integration (LSI) devices used on the boards along with sources for additional information.

This chapter describes the board level hardware features. It starts with a board level overview and features list and continues on to provide the board's memory maps (the major feature of this chapter).

Programmable registers in the MCP750 series that reside in ASICs are covered in the chapters applicable to those ASICs. For your information, [Chapter 2, Raven PCI Host Bridge & Multi-Processor Interrupt Controller](#), covers the Raven chip. [Chapter 3, Falcon ECC Memory Controller Chip Set](#), covers the Falcon chip set. [Chapter 4, Programming Details](#), covers certain programming features, such as interrupts and exceptions. [Appendix A, Related Documentation](#), lists all related documentation.

Overview

The MCP750 series SBCs provide many standard features required by a computer system. This includes: Ethernet interface, keyboard interface, mouse interface, sync and async serial ports, parallel port, boot Flash memory, and up to 256MB of ECC DRAM.

Feature Summary

There are many models based on the MCP750 series architecture. The following table summarizes the major features:

Table 1-1. MCP750 Series Features Summary

| Feature | Description |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Processors | Single MPC750 processor. Bus clock frequencies up to 66 MHz |
| L2 Cache | 1MB of backside external cache memory |
| Flash | 4MB or 8MB (64-bit wide), with socketed 1MB (16-bit wide) |
| DRAM | 16MB to 256MB, ECC protected (single-bit correction, double-bit detection) Two-way Interleaved |
| NVRAM | 8KB |
| RTC | MK48T59/559 device |
| Peripheral Support | Two async serial ports Two sync/async serial ports One parallel printer port 10BaseT/100BaseTX Ethernet interface One PS/2 keyboard and one PS/2 mouse interface One PS/2 floppy port Primary & secondary EIDE Ports (primary has <i>Compact Flash</i> interface on motherboard) |
| PMC Slots | Single 32/64-bit slot |
| Miscellaneous | RESET/ABORT switch Status LEDs |

System Block Diagram

The MCP750 provides a 1MB external cache option. The Falcon chip set controls both the boot Flash memory and the ECC DRAM. The Raven ASIC functions as the 64-bit PCI host bridge and the MPIC interrupt controller.

PCI devices include:

- ❑ Ethernet
- ❑ PCI-to-PCI bridge for CompactPCI bus interface (optional second bridge located on companion card)
- ❑ PCI-to-ISA/IDE/USB bridge
- ❑ One PMC slot

Standard I/O functions (serial, parallel, FDD, keyboard and mouse) are provided by the Super I/O device which resides on the ISA bus.

The NVRAM/RTC provides NVRAM and an RTC with battery backup. A 512 x 8 Serial EEPROM is also provided via an I²C interface off of the PBC.

The general system block diagram for the MCP750 series is shown on the following page:

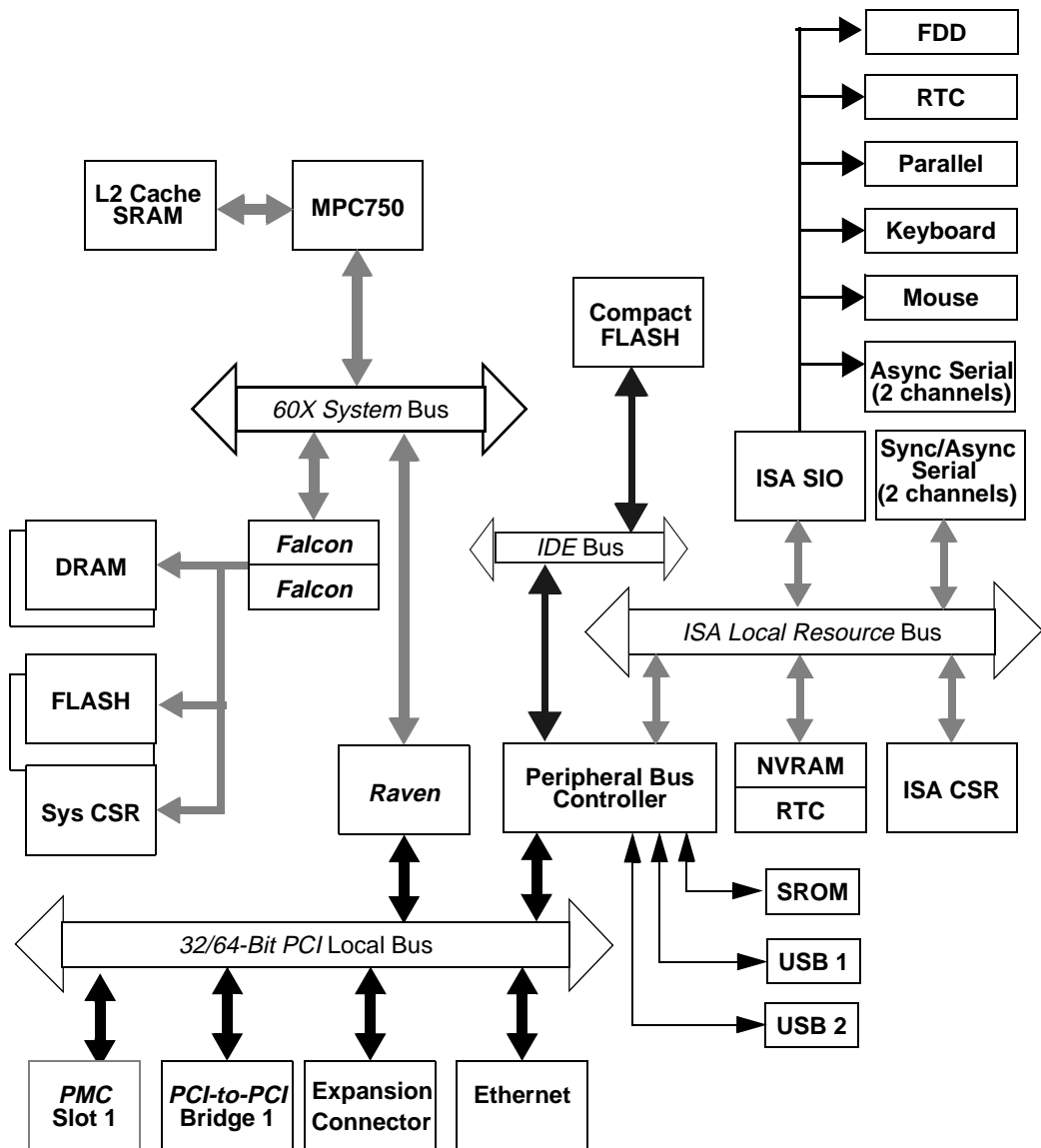


Figure 1-1. MCP750 Series System Block Diagram

Functional Description

Overview

The MCP750 is a single-slot SBC based on the PowerPlus architecture. It consists of the following:

- ❑ MPC750 processor with L2 cache
- ❑ Raven PCI Bridge & Interrupt Controller
- ❑ ECC Memory Controller Falcon chipset
- ❑ 5MB or 9MB of Flash memory
- ❑ IDE Compact Flash memory
- ❑ 16MB to 256MB of ECC-protected DRAM
- ❑ Interface to a CompactPCI bus
- ❑ Various I/O peripherals.

I/O peripheral devices on the PCI bus include: 10/100BaseT Ethernet interface, USB host controller, a 512 x 8 Serial EEPROM, an ISA master/slave interface, a Fast EIDE interface, and one PMC Slot.

Functions provided from the ISA bus are two asynchronous and two synchronous/asynchronous serial ports, keyboard and mouse ports, a floppy disk controller, a printer port, a real-time clock, and NVRAM.

The MCP750 interfaces to a CompactPCI bus using a DEC 21154 PCI-to-PCI bridge device. This device provides a 64-bit primary and a 64-bit secondary interface allowing full 64-bit data access between CompactPCI bus devices and the Raven host/PCI bridge. This bus is capable of driving 7 CompactPCI slots.

The 64-bit local PCI bus is routed to CompactPCI connect J4. This allows a second CompactPCI bus interface to be implemented on a companion board by connecting a PCI-to-PCI bridge to the local PCI bus via J4 on the companion board. This implementation would require a custom backplane to connect the J4 signals of the MCP750 SBC to the companion board J4.

Front panel connectors on the MCP750 series board include:

- ❑ RJ45 connector for 10BaseT/100BaseTX Ethernet
- ❑ 9-pin D-sub connector for one asynchronous serial channel
- ❑ Two 4-pin USB connectors

Three additional serial ports along with the two USB ports; the floppy disk drive port; and the IDE, keyboard, mouse, and parallel ports are routed to J3 and J5 for rear I/O.

The I/O option supported by the MCP750 series is via the TMCP700 transition module. This module provides the enhanced P1284 parallel port interface and the full synchronous support for serial ports 3 and 4.

The MCP750 series contains one IEEE1386.1 PCI Mezzanine Card (PMC) slot. This slot is 32/64-bit capable and supports both front and rear I/O. PMC I/O pins 1 through 64 of the PMC connector J14 are routed to the J3 User I/O Connector.

Programming Model

Memory Maps

The following sections describe the memory maps for the MCP750 series SBCs.

Processor Memory Maps

The processor memory map is controlled by the Raven ASIC and the Falcon chipset. The Raven ASIC and the Falcon chipset have flexible programming map decoder registers to customize the system to fit many different applications.

Default Processor Memory Map

After a reset, the Raven ASIC and the Falcon chipset provide the default processor memory map as shown in the following table.

Table 1-2. Default Processor Memory Map

| Processor Address | | Size | Definition | Notes |
|-------------------|-----------|-----------------|----------------------------|-------|
| Start | End | | | |
| 0000 0000 | 7FFF FFFF | 2G | Not mapped | |
| 8000 0000 | 8001 FFFF | 128K | PCI/ISA I/O Space | 1 |
| 8002 0000 | FEF7 FFFF | 2G - 16M - 640K | Not mapped | |
| FEF8 0000 | FEF8 FFFF | 64K | Falcon Registers | |
| FEF9 0000 | FEFE FFFF | 384K | Not mapped | |
| FEFF 0000 | FEFF FFFF | 64K | Raven Registers | |
| FF00 0000 | FFEF FFFF | 15M | Not mapped | |
| FFF0 0000 | FFFF FFFF | 1M | ROM/FLASH Bank A or Bank B | 2 |

Notes

1. This default map for PCI/ISA I/O space allows software to determine if the system is MPC105-based or Falcon/Raven-based by examining either the PHB Device ID or the CPU Type Register.
2. The first 1MB of ROM/FLASH memory Bank A appears at this range after a reset if the **rom_b_rv** control bit is cleared. If the **rom_b_rv** control bit is set, then this address range maps to ROM/FLASH Bank B.

Processor PREP Memory Map

The Raven/Falcon chipset can be programmed for PREP-compatible memory map. The following table shows the PREP memory map of the MCP750 series from the point of view of the processor.

Table 1-3. PREP Memory Map Example

| Processor Address | | Size | Definition | Notes |
|-------------------|-----------|-----------|-----------------------------------------------------------|-------|
| Start | End | | | |
| 0000 0000 | top_dram | dram_size | System Memory (onboard DRAM) | 1 |
| 8000 0000 | BFFF FFFF | 1G | Zero-Based PCI I/O Space: 0000 0000 - 3FFF FFFF | 2 |
| C000 0000 | FCFF FFFF | 1G - 48M | Zero-Based PCI/ISA Memory Space: 0000 0000 - 3CFF FFFF | 2 |
| FD00 0000 | FEF7 FFFF | 40.5M | Reserved | |
| FEF8 0000 | FEF8 FFFF | 64K | Falcon Registers | |
| FEF9 0000 | FEFE FFFF | 384K | Reserved | |
| FEFF 0000 | FEFF FFFF | 64K | Raven Registers | 5 |
| FF00 0000 | FF7F FFFF | 8M | ROM/FLASH Bank A | 1, 3 |
| FF80 0000 | FF8F FFFF | 1M | ROM/FLASH Bank B | 1, 3 |
| FF90 0000 | FFEF FFFF | 6M | Reserved | |
| FFF0 0000 | FFFF FFFF | 1M | ROM/FLASH Bank A or Bank B | 4 |

Notes

1. Programmable via Falcon chipset.
2. Programmable via Raven ASIC.
3. The actual size of each ROM/FLASH bank may vary.
4. The first 1MB of ROM/FLASH Bank A appears at this range after a reset if the **rom_b_rv** control bit is cleared. If the **rom_b_rv** control bit is set, then this address range maps to ROM/FLASH Bank B.

5. The only method to generate a PCI Interrupt Acknowledge cycle (8259 IACK) is to perform a read access to the Raven's PIACK register at 0xFEFF0030.

The following table shows the programmed values for the associated Raven PPC registers for the processor PREP memory map.

Table 1-4. Raven PPC Register Values for PREP Memory Map

| Address | Register Name | Register Value |
|-----------|-----------------|----------------|
| FEFF 0040 | MSADD0 | C000 FCFF |
| FEFF 0044 | MSOFF0 & MSATT0 | 4000 00C2 |
| FEFF 0048 | MSADD1 | 0000 0000 |
| FEFF 004C | MSOFF1 & MSATT1 | 0000 0002 |
| FEFF 0050 | MSADD2 | 0000 0000 |
| FEFF 0054 | MSOFF2 & MSATT2 | 0000 0002 |
| FEFF 0058 | MSADD3 | 8000 BFFF |
| FEFF 005C | MSOFF3 & MSATT3 | 8000 00C0 |

PCI Configuration Access

PCI Configuration accesses are accomplished via the **CONFIGADD** and **CONFIGDAT** registers. These two registers are implemented by the Raven ASIC. With the PREP memory map, the **CONFIG_ADD** register and the **CONFIG_DAT** register are located at 0x80000CF8 and 0x80000CFC, respectively.

PCI Memory Maps

The PCI memory map is controlled by the Raven ASIC and the 21154 PCI-to-PCI bridge. The Raven ASIC and the 21154 bridge have flexible programming Map Decoder registers to customize the system to fit many different applications.

Default PCI Memory Map

After a reset, the Raven ASIC and the 21154 bridge turn all the PCI slave map decoders off. Software must program the appropriate map decoders for a specific environment.

PCI PREP Memory Map

The following table shows a PCI memory map of the MCP750 series that is PREP-compatible from the point of view of the PCI local bus.

Table 1-5. PCI PREP Memory Map

| PCI Address | | Size | Definition | Notes |
|-------------|-----------|------------|----------------------|-------|
| Start | End | | | |
| 0000 0000 | 00FF FFFF | 16M | PCI/ISA Memory Space | |
| 0100 0000 | 7FFF FFFF | 2G - 16M | PCI Memory Space | |
| 8000 0000 | FBFF FFFF | 2G - 64M | Onboard ECC DRAM | 1 |
| FC00 0000 | FC03 FFFF | 256K | RavenMPIC | 1 |
| FC04 0000 | FFFF FFFF | 64M - 256K | PCI Memory Space | |

Note

1. Programmable via the Raven's PCI Configuration registers.

The following table shows the programmed values for the associated Raven PCI registers for the PREP-compatible memory map.

Table 1-6. Raven PCI Register Values for PREP Memory Map

| Configuration Address Offset | Configuration Register Name | Register Value |
|------------------------------|-----------------------------|----------------|
| \$14 | RavenMPIC MBASE | FC00 0000 |
| \$80 | PSADD0 | 8000 FBFF |
| \$84 | PSOFF0 & PSATT0 | 8000 00FX |
| \$88 | PSADD1 | 0000 0000 |
| \$8C | PSOFF1 & PSATT1 | 0000 0000 |
| \$90 | PSADD2 | 0000 0000 |
| \$94 | PSOFF2 & PSATT2 | 0000 0000 |
| \$98 | PSADD3 | 0000 0000 |
| \$9C | PSOFF3 & PSATT3 | 0000 0000 |

CompactPCI Bus Mapping

CompactPCI Bus Mapping involves DEC 21154 Transaction Forwarding and Upstream PCI Transactions. These functions are described below.

DEC 21154 Transaction Forwarding

The processor will access devices on the CompactPCI bus by using transaction forwarding provided by the DEC 21154 PCI-to-PCI bridge. Transaction forwarding within the 21154 is based on address ranges defined in the 21154 base and limit registers. The 21154 provide registers for I/O, memory and pre-fetchable memory spaces. These registers define the address range for which PCI transactions are forwarded downstream from the primary PCI bus to the CompactPCI bus (secondary bus). All devices on the CompactPCI bus must be configured for addressing within this defined range. Conversely, these registers also define the addresses for which transactions will be forwarded upstream. Any CompactPCI bus address, generated by a CompactPCI bus master, not in the defined memory range will be forwarded upstream from the CompactPCI bus to the Primary PCI bus. There is no address translation between CompactPCI busses and the Primary PCI bus. Following a reset, the PCI-to-PCI bridge devices (21154) address maps are disabled.

Upstream PCI Transactions

Upstream transactions from a CompactPCI bus to the Primary PCI bus can be used by a bus master on a CompactPCI bus to access the onboard memory, an onboard PCI device or a device on the other CompactPCI bus. An upstream transaction presented on the Primary PCI bus will be accepted by the Raven ASIC as a DRAM access if it is in the address range defined by the Raven map registers.

Falcon-Controlled System Registers

The Falcon chipset consists of two identical Falcon ASIC devices, the upper Falcon and the lower Falcon. The upper Falcon connects to the upper half of the system data bus, DH00 through DH31, and the lower Falcon connects to lower half of the system data bus, DL00 through DL31.

Falcon Registers

The base address of the Falcon chipset's Registers is hard coded to the address \$FEF80000. Accesses to these registers are mapped differently depending on whether they are reads or writes. For reads, the data read on the upper half of the data bus comes from the upper while data read on the lower half of the data bus comes from the lower. For writes, internal register or test SRAM data written on the upper half of the data bus goes to the upper and is automatically copied by hardware to the lower. Internal register or test SRAM data written on the lower half of the data bus does not go to either in the pair, however the access is terminated normally with TA#.

Falcon-Controlled System Registers

The Falcon chipset latches the states of the DRAM data lines onto the PR_STAT1 registers. The MCP750 uses these status registers to provide the system configuration information. In addition, the Falcon chipset performs the decode and control for an external register port. This function is used by the MCP750 to provide access to registers, external to Falcon, which provide interrupt identification, reset identification, LED control, and CPU control.

Table 1-7. System Register Summary

| BIT # ----> | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
|-----------------|---------------------------------------------------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| FEF80400 | System Configuration Register (Upper Falcon's PR_STAT1) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF80404 | Memory Configuration Register (Lower Falcon's PR_STAT1) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF88000 | System External Cache Control Register | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF88100 | Processor 0 External Cache Control Register | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF88200 | Processor 1 External Cache Control Register | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF88300 | CPU Control Register | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

The following subsections describe these system registers in detail.

System Configuration Register (SYSCR)

The states of the RD[0:31] DRAM data pins, which have weak internal pull-ups, are latched by the upper Falcon chip at a rising edge of the power-up reset and stored in this System Configuration Register to provide some information about the system. Configuration is accomplished with external pull-down resistors. This 32-bit read-only register is defined as follows:

Table 1-8. System Configuration Register

| Register | System Configuration Register - \$FEF80400 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|--------------------------------------------|---|---|---|---|---|---|---|--------|---|----|----|-------|----|----|----|------------|----|----|----|------------|----|----|----|-----|----|----|----|-----|----|----|----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Field | SYSID | | | | | | | | SYSCLK | | | | SYSXC | | | | POSTA T | | | | PISTA T | | | | | | | | | | | |
| Operation | READ ONLY | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | \$FC | | | | | | | | X | | | | X | | | | X | | | | X | | | | \$F | | | | \$F | | | |

SYSID System Identification. This field specifies the type of the overall system configuration so that the software may appropriately handle any software visible differences. For the MCP750 series, this field returns a value of \$FC.

SYSCLK System Clock Speed. This field relays the system clock speed and the PCI clock speed information as follows:

Table 1-9. System Clock Speed

| SYSCLK Value | System Clock Speed | PCI Clock Speed |
|------------------|--------------------|-----------------|
| 0b0000 to 0b1100 | Reserved | Reserved |
| 0b1101 | 50 MHz | 25 MHz |
| 0b1110 | 60 MHz | 30 MHz |
| 0b1111 | 66.66 MHz | 33.33 MHz |

SYSXC System External Cache Size. This field reflects size of the look-aside cache on the system bus.

Table 1-10. System External Cache Size

| SYSXC Value | External Look-aside Cache Size |
|--------------------|---------------------------------------|
| 0b1111 | None |

P0STAT Processor 0 Status. This field is encoded as follows:

Table 1-11. Processor Status Values

| P0/1STAT Value | Processor 0/1 Present | External In-line Cache Size |
|-----------------------|------------------------------|------------------------------------|
| 0b0000 to 0b0011 | Reserved | Reserved |
| 0b0100 | YES | 1MB |
| 0b0101 | YES | 512KB |
| 0b0110 | YES | 256KB |
| 0b0111 | YES | None |
| 0b1000 to 0b1111 | NO | N/A |

Note P1STAT is not implemented and will therefore read as 0b1111.

Memory Configuration Register (MEMCR)

The states of the RD[00:31] DRAM data pins, which have weak internal pull-ups, are latched by the lower Falcon chip at a rising edge of the power-up reset and stored in this Memory Configuration Register to provide some information about the system memory. Configuration is accomplished with external pull-down resistors. This 32-bit read-only Register is defined as follows:

Table 1-12. Memory Configuration Register

| Register | Memory Configuration Register - \$FEF80404 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|--------------------------------------------|----|----|----|--------|----|----|--------|----|---------|---------|---------|----|----------|----------|----------|----------|----------|----------|----------|---------|---------|---------|---------|----|----|---------|---------|---------|----|----|----|
| Bit | 32 | 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 | 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 | 55 | 56 | 57 | 58 | 59 | 60 | 61 | 62 | 63 |
| Field | | | | | M_FREF | | | M_SPD1 | | R_A_TYP | R_A_TYP | R_A_TYP | | R_B_TYP0 | R_B_TYP1 | R_B_TYP2 | L2_TYPE0 | L2_TYPE1 | L2_TYPE2 | L2_TYPE3 | L2_PL10 | L2_PL11 | L2_PL12 | L2_PL13 | | | FL_SHP0 | FL_SHP1 | FL_SHP2 | | | |
| Operation | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | 1 | 1 | 1 | X | 1 | 1 | X | X | 1 | X | X | X | 1 | X | X | X | X | X | X | X | X | X | X | X | 1 | 1 | X | X | X | 1 | 1 | 1 |

M_FREF Block A/B/C/D Fast Refresh. When this bit is set, it indicates that a DRAM block requires a faster refresh rate. If any of the four blocks requires faster refresh rate, then the **ram_ref** control bit should be set.

M_SPD[0:1] Memory Speed. This field relays the memory speed information as follows:

Table 1-13. Memory Speed

| M_SPD[0:1] | DRAM Speed | DRAM Type |
|------------|------------|-----------|
| 0b00 | 70ns | Fast Page |
| 0b01 | 60ns | Fast Page |
| 0b10 | Reserved | Reserved |
| 0b11 | 50ns | EDO |

These two bits reflect the combined status of the four blocks of DRAM. Initialization software uses this information to program the **ram_spd0** and **ram_spd1** control bits in the Falcon's Chip Revision Register.

R_A/B_TYP[0:1] ROM/Flash Type. This field is encoded as follows:

Table 1-14. ROM/FLASH Type

| ROM_A/B_TYP[0:2] | ROM/FLASH Type |
|-------------------------|----------------------------------------------------|
| 0b000 to 0b101 | Reserved |
| 0b110 | Intel 16-bit wide FLASH with 16K Bottom Boot Block |
| 0b111 | Unknown type (i.e., ROM/FLASH Sockets) |

Note The device width is different from the width of the FLASH bank. If the bank width is 64-bit and the device width is 16-bit then the FLASH bank consists of four FLASH devices.

L2_TYPE[0:3] L2 Memory Type. This field is encoded as follows:

Table 1-15. L2 Memory Type

| L2_TYPE[0:3] | Configuration |
|---------------------|------------------------------------------------------------------------------|
| 0b0000 | Late Write Sync, 1 ns hold, Differential clock, Parity |
| 0b0001 | Pipelined Sync Burst, 0.5 ns hold, No Differential clock, Parity |
| 0b0010 | Late Write Sync, 1 ns hold, Differential clock, No parity |
| 0b0011 | Pipelined Sync Burst, 0.5 ns hold, No Differential clock, No parity |
| 0b0010-0b1111 | Reserved |

L2_PLL[0:3] L2 Core-To-Cache Ratio. This field is encoded as follows:

Table 1-16. L2 Core-To-Cache Ratio

| L2_PLL[0:3] | Ratio (Core Freq Divisor) |
|--------------------|--------------------------------------|
| 0b0000 | Disabled |
| 0b0001 | 1:1 (1) |
| 0b0010 | 3:2 (1.5) |
| 0b0011 | 2:1 (2) |
| 0b0100 | 5:2 (2.5) |
| 0b0101 | 3:1 (3) |
| 0b0110-0b1111 | Reserved |

FLSHP[0:2] Bank A FLASH memory size. This field is encoded as follows:

Table 1-17. Bank A FLASH Memory Size

| FLASH Size | FLSHP0_ | FLSHP1_ | FLSHP2_ |
|-------------------|----------------|----------------|----------------|
| 1 MB | 0 | 0 | 0 |
| 2 MB | 0 | 0 | 1 |
| 4 MB | 0 | 1 | 0 |
| 8 MB | 0 | 1 | 1 |
| 16 MB | 1 | 0 | 0 |
| 32 MB | 1 | 0 | 1 |
| 64 MB | 1 | 1 | 0 |
| No FLASH | 1 | 1 | 1 |

System External Cache Control Register (SXCCR)

The System Cache Control Register is not implemented on the MCP750 series product, but it is documented for backward compatibility of the firmware/software. This 8-bit register is defined as follows:

Table 1-18. System External Cache Control Register

| Register | System External Cache Control Register - \$FEF88000 | | | | | | | |
|-----------|-----------------------------------------------------|----------|----------|----------|---|---|---|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Field | Reserved | Reserved | Reserved | Reserved | | | | |
| Operation | R/W | | | | | | | |
| Reset | X | X | X | X | X | X | X | X |

Processor 0 External Cache Control Register (P0XCCR)

This register has no function on the MCP750 board, since it does not have In-line Cache, but it is documented for backward compatibility of the firmware/software. This 8-bit register is defined as follows:

Table 1-19. Processor 0 External Cache Control Register

| Register | Processor 0 External Cache Control Register - FEF88100h | | | | | | | |
|-----------|---------------------------------------------------------|----------|---|---|---|---|---|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Field | P0XC_CFG | P0XC_DIS | | | | | | |
| Operation | R/W | | | | | | | |
| Reset | x | x | x | x | x | x | X | X |

P0XC_CFG N/A

P0XC_DIS N/A

Processor 1 External Cache Control Register (P1XCCR)

This register has no function on the MCP750 board since it does not have In-line Cache, but it is documented for backward compatibility of the firmware/software. This 8-bit register is defined as follows:

Table 1-20. Processor 1 External Cache Control Register

| Register | Processor 1 External Cache Control Register - FEF88200h | | | | | | | |
|-----------|---------------------------------------------------------|----------|---|---|---|---|---|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Field | P1XC_CFG | P1XC_DIS | | | | | | |
| Operation | R/W | | | | | | | |
| Reset | x | x | x | x | X | X | X | X |

P1XC_CFG N/A

P1XC_DIS N/A

CPU Control Register

The CPU Control Register is accessed via the RD[32:39] data lines of the upper Falcon device. This 8-bit register is defined as follows:

Table 1-21. CPU Control Register

| Register | CPU Control Register - \$FEF88300 | | | | | | | |
|-----------|-----------------------------------|--------|---------|---------|---|---|---|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Field | Reserved | LEMODE | P1_TBEN | P0_TBEN | | | | |
| Operation | R/W | R/W | R/W | R/W | R | R | R | R |
| Reset | 0 | 0 | 1 | X | X | X | X | X |

LEMODE Little-endian Mode. This bit must be set in conjunction with the LEND bit in the Raven for little-endian mode.

P0/1_TBEN Processor Time Base Enable. When this bit is cleared, the TBEN pin of Processor will be driven low. **P1_TBEN** is not used since there is only one processor.

ISA Local Resource Bus

VT82C586B PBC

The PBC contains ISA Bridge I/O registers for various functions. These registers are actually accessible from the PCI bus. Refer to the *VT82C586B Data Sheet* for details. The following Table shows the hardware strapping for the PBC device.

Table 1-22. Strap Pins Configuration for the VT82C586B

| Pins | Reset Configuration |
|------|------------------------------------------|
| XD0 | 0 - Disable internal keyboard controller |
| XD1 | 0 - Disable internal mouse |
| XD2 | 0 - Disable internal RTC |
| SPKR | 1 - IDE I/O base address is flexible |

PC87307VUL Super I/O Strapping

The PC87307VUL Super I/O provides the following functions to the MCP750 series: a keyboard interface, a PS/2 mouse interface, two async serial ports, a parallel port, and a floppy drive port. Refer to the PC87307VUL Data Sheet, listed in [Appendix A, Related Documentation](#), for additional details and programming information.

The following table shows the hardware strapping for the Super I/O device:

Table 1-23. Strap Pins Configuration for the PC87307VUL

| Pins | Reset Configuration |
|-------------------|-----------------------------------------------------------|
| CFG0 | 0 - FDC, KBC and RTC wake up inactive. |
| CFG1 | 1 - Xbus Data Buffer (XDB) enabled. |
| CFG3, CFG2 | 00 - Clock source is 24 MHz fed via X1 pin. |
| BADDR1, BADDR2 | 11 - PnP Motherboard, Wake in Config State. Index \$002E. |
| SELCS | 1 - CS0# on CS0# pin. |

NVRAM/RTC & Watchdog Timer Registers

The M48T559 provides the MCP750 series with 8K of non-volatile SRAM, a time-of-day clock, and a watchdog timer. Accesses to the M48T559 are accomplished via three registers: The NVRAM/RTC Address Strobe 0 Register, the NVRAM/RTC Address Strobe 1 Register, and the NVRAM/RTC Data Port Register. The NVRAM/RTC Address Strobe 0 Register latches the lower 8 bits of the address and the NVRAM/RTC Address Strobe 1 Register latches the upper 5 bits of the address.

Table 1-24. M48T559 Access Registers

| PCI I/O Address | Function |
|-----------------|---------------------------------------|
| 0000 0074 | NVRAM/RTC Address Strobe 0 (A7 - A0) |
| 0000 0075 | NVRAM/RTC Address Strobe 1 (A15 - A8) |
| 0000 0077 | NVRAM/RTC Data Register |

The NVRAM and RTC is accessed through the above three registers. When accessing a NVRAM/RTC location, follow the following procedure:

1. Write the low address (A7-A0) of the NVRAM to the NVRAM/RTC STB0 register.

2. Write the high address (A15-A8) of the NVRAM to the NVRAM/RTC STB1 register.
3. Read or write the NVRAM/RTC Data Port.

Note Refer to the *M48T559 Data Sheet* for additional details and programming information.

SROM

The MCP750 SBC contains a 512 x 8 2-wire Serial CMOS EEPROM (AT24C04) that may be used to store various configuration information. This device is addressed via an I²C interface. The I²C interface is implemented using two general purpose I/O pins of the PBC device GPIO register. GPIO bit 1 will provide the I²C clock and GPIO bit 2 will provide the I²C data. The PBC device does not implement any I²C protocol in hardware so the clock/data protocol must be implemented in software.

The AT24C04 device address pins are tied low so the device addressing is as follows:

Table 1-25. AT24C04 Serial EEPROM Device Address Field

| AT24C04 Serial EEPROM Device Address Field | | | | | | | |
|--------------------------------------------|---|---|---|----------------|---|----|-----|
| 1 | 0 | 1 | 0 | 0 | 0 | P0 | R/W |
| Start Sequence | | | | Memory Address | | | R/W |
| MSB | | | | | | | LSB |

P0 Memory page address bit.

R/W Memory Read/Write bit. This bit is set for read operations and cleared for write operations.

Module Configuration and Status Registers

Three registers provide the configuration and status information about the board. These registers are listed in the following table:

Table 1-26. Module Configuration and Status Registers

| ISA I/O Address | Function |
|-----------------|-------------------------------------------------------|
| 0000 0800 | CPU Configuration Register |
| 0000 0802 | Base Module Feature Register |
| 0000 0803 | Base Module Status Register |
| 0000 0810 | Base Module Extended Features Register |
| 0000 0820 | SYSLED Register |
| 0000 0824 | Board Status Register |
| 0000 0828 | Board Last Reset Register |
| 0000 082C | Hardware Link Override Register (For prototypes only) |

The following sub sections describe these registers in detail.

CPU Configuration Register

The CPU Configuration Register is an 8-bit register located at ISA I/O address x0800.

Table 1-27. Old CPU Configuration Register

| Register | Old CPU Configuration Register - \$FE000800 | | | | | | | |
|-----------|---------------------------------------------|-----|-----|-----|-----|-----|-----|-----|
| Bit | SD7 | SD6 | SD5 | SD4 | SD3 | SD2 | SD1 | SD0 |
| Field | CPUTYPE | | | | | | | |
| Operation | R | | | | R | | | |
| Reset | \$E | | | | \$F | | | |

CPUTYPE CPU Type. This field will always read as \$E for the MCP750 series. The *System Configuration Register (SYSCR)* should be used for additional information.

Base Module Feature Register

The Base Module Feature Register is an 8-bit register providing the configuration information about the MCP750 SBC. This read-only register is located at ISA I/O address x0802.

Table 1-28. Base Module Feature Register

| Register | Base Module Feature Register - Offset \$0802 | | | | | | | |
|-----------|----------------------------------------------|-------|----------|-------|----------|----------|-------|----------|
| Bit | SD7 | SD6 | SD5 | SD4 | SD3 | SD2 | SD1 | SD0 |
| Field | Reserved | SCCP_ | Reserved | PMCP_ | Reserved | Reserved | LANP_ | Reserved |
| Operation | R | R | R | R | R | R | R | R |
| Reset | 1 | X | 1 | X | 1 | 1 | X | 1 |

SCCP_ Z85230 ESCC Present. If set, there is no onboard sync serial support. If cleared, there is onboard support for sync serial interface.

PMCP_ PMC Slot Present. If set, there is no PCI Mezzanine Card installed in the PMC Slot. If cleared, the PMC/PCIX Slot 2 contains a PCI Mezzanine Card or a PCI device.

LANP_ Ethernet Present. If set, there is no Ethernet transceiver interface. If cleared, there is on-board Ethernet support.

Base Module Status Register (BMSR)

The Base Module Status Register is an 8-bit read-only register located at ISA I/O address x0803.

Table 1-29. Base Module Status Register

| Register | Base Module Status Register - Offset \$0803 | | | | | | | |
|-----------|---------------------------------------------|-----|-----|-----|-----|-----|-----|-----|
| Bit | SD7 | SD6 | SD5 | SD4 | SD3 | SD2 | SD1 | SD0 |
| Field | BASE_TYPE | | | | | | | |
| Operation | R | | | | | | | |
| Reset | N/A | | | | | | | |

BASE_TYPE Base Module Type. This eight bit field is used to provide the category of the base module and is defined as follows:

Table 1-30. Base Module Type

| BASE_TYPE Value | Base Module Type |
|-----------------|------------------|
| \$0 to \$DF | Reserved |
| \$E0 | Standard MCP750 |
| \$FA to FF | Reserved |

Base Module Extended Features Register

The Base Module Extended Features Register is an 8-bit register providing the configuration and status information that is unique to the MCP750 SBC. This read-only register is located at ISA I/O address x0810h.

Table 1-31. Base Module Extended Features Register

| Register | Base Module Extended Features Register - Offset 0810h | | | | | | | |
|-----------|-------------------------------------------------------|----------|----------|----------|------|---------|--------|--------|
| Bit | SD7 | SD6 | SD5 | SD4 | SD3 | SD2 | SD1 | SD0 |
| Field | USBV1_OK | USBV0_OK | Reserved | Reserved | CFP_ | NVRAMP_ | LLINT_ | PPBIP_ |
| Operation | R | R | R | R | R | R | R | R |
| Reset | x | x | 1 | 1 | x | x | X | x |

USBV1_OK USB BUS 1 Voltage OK. If set, the polyswitch providing the 5 volts for USB Bus 1 is OK. If cleared, the polyswitch is open.

USBV0_OK USB BUS 0 Voltage OK. If set, the polyswitch providing the 5 volts for USB Bus 0 is OK. If cleared, the polyswitch is open.

CFP_ CompactFlash™ Present. If set, there is no CompactFlash Memory Card installed on the Memory Mezzanine. If cleared, the CompactFlash is installed.

NVRAMP_ NVRAM Present. If set, there is no M48T559 NVRAM/RTC device installed on board. If cleared, the device is installed.

LLINT_ Low Level Interrupts. If set, this bit indicates that this is a Rev 01C board which implemented an active high edge-sensitive interrupt for the Z85230/Z8536. If cleared, this is a later board which implements an active low level-sensitive ISA interrupt for the Z85230/Z8536 interrupts (see Table 4-2).

PPB1P_ PCI-to-PCI Bridge 1 Present. If set, there is no 21154 bridge device installed for CompactPCI Bus 1. If cleared, the device is installed.

SYSLED Status Register

The SYSLED register provides information about the system. This register is located at ISA I/O address x0820.

Table 1-32. SYSLED Status Register

| Register | SYSLED Status Register - Offset 0820h | | | | | | | |
|-----------|---------------------------------------|-----|-----|-----|-----|--------|----------|----------|
| Bit | SD7 | SD6 | SD5 | SD4 | SD3 | SD2 | SD1 | SD0 |
| Field | | | | | | SYSEN_ | Reserved | Reserved |
| Operation | R | R | R | R | R | R | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

SYSEN_ System Slot Identification. If set, the board is not installed in a CompactPCI system slot. If cleared, the board is installed in a system slot and will provide clocks for arbitration of the peripheral slots. The MCP750 must always be installed in a system slot.

Board Status Register

The Board Status register provides information about the current operating condition of the MCP750 SBC. This register is located at ISA I/O address x0824.

Table 1-33. Board Status Register

| Register | Board Status Register - Offset 0824h | | | | | | | |
|-----------|--------------------------------------|-----|-----|-------|-------|--------|--------|--------|
| Bit | SD7 | SD6 | SD5 | SD4 | SD3 | SD2 | SD1 | SD0 |
| Field | | | | LANOK | USBOK | CPCIOK | BOOTOK | ONLINE |
| Operation | R | R | R | R/W | R/W | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

LANOK LAN (Ethernet) Channel OK. This read/write bit is under software control and is used to indicate the state of the Ethernet channel. If set, the channel is operational. If cleared, the channel is uninitialized or has failed.

USBOK USB Channels OK. This read/write bit is under software control and is used to indicate the state of the USB channels. If set, the channels are operational. If cleared, the channels are uninitialized or have failed.

CPCIOK CompactPCI OK. This read/write bit is under software control and is used to indicate the state of the CompactPCI interfaces. If set, the interfaces are operational. If cleared, one of the busses is uninitialized or has failed.

BOOTOK BOOT OK. This read/write bit is under software control and is used to indicate the state of the BOOT diagnostics. If set, the BOOT diagnostic routine has executed and passed. If cleared, the diagnostics have not yet run or have failed.

ONLINE Board On-line. This read/write bit is under software control and is used to indicate the state of the overall condition of the board. If set, the board is ready and on-line. If cleared, the board is off-line.

Board Last Reset Register

The Board Last Reset register provides the source of the most recent reset. This read-only register is located at ISA I/O address x0828. This register is loaded with a new value when a reset occurs.

Table 1-34. Board Last Reset Register

| Register | Board Last Reset Register - Offset 0828h | | | | | | | |
|-----------|------------------------------------------|--------|--------|----------|----------|------|-------|-------|
| Bit | SD7 | SD6 | SD5 | SD4 | SD3 | SD2 | SD1 | SD0 |
| Field | Reserved | SWSRST | SWHRST | Reserved | Reserved | WDT2 | FPBTN | PWRON |
| Operation | R | R | R | R | R | R | R | R |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

PWRON Power-On Reset. If set, a power-on reset has occurred or an undervoltage reset has occurred on 3.3v or 5v.

FPBTN Front Panel Push Button Reset. If set, a front panel push button reset has occurred.

WDT2 Watchdog Timer Level 2 Reset. If set, a level 2 Watchdog timer reset has occurred.

SWHRST Software Hard Reset. If set, a software initiated hard reset has occurred via the PBC Port 92 Fast Reset bit of the SA Test Mode register.

SWSRST Software Soft Reset. If set, a software initiated soft reset has occurred via the Processor Init Register of the Raven MPIC.

Z85230 ESCC and Z8536 CIO Registers and Port Pins

The Z85230 ESCC is used to provide the two additional sync/async serial ports. The PCLK which can be used to derive the baud rates, is 5 MHz. Refer to the SCC User's Manual for programming information on the Z85230 ESCC device.

The Z8536 CIO is used to provide the modem control lines not provided by the Z85230 ESCC. Refer to the Z8536 Data Sheet, listed in [Appendix A, Related Documentation](#), for programming information.

Z8536/Z85230 Registers

Accesses to the Z8536 CIO and the Z85230 ESCC are accomplished via Port Control and Port Data Registers. A Pseudo IACK Register is also defined to retrieve interrupt vectors from these devices. The Z8536 CIO has higher priority than the Z85230 ESCC in the interrupt daisy chain. The following list the registers associated with accessing these two registers:

Table 1-35. Z85230 Access Registers

| PCI I/O Address | Function |
|-----------------|----------------------------------------|
| 0000 0840 | Z85230: Port B (Serial Port 4) Control |
| 0000 0841 | Z85230: Port B (Serial Port 4) Data |
| 0000 0842 | Z85230: Port A (Serial Port 3) Control |
| 0000 0843 | Z85230: Port A (Serial Port 3) Data |
| 0000 084F | Z85230/Z8536 Pseudo IACK |
| 0000 0844 | Z8536 CIO: Port C's Data Register |
| 0000 0845 | Z8536 CIO: Port B's Data Register |
| 0000 0846 | Z8536 CIO: Port A's Data Register |
| 0000 0847 | Z8536 CIO: Control Register |
| 0000 084F | Z85230/Z8536 Pseudo IACK |

Z8536 CIO Port Pins

The assignment for the port pins of the Z8536 CIO is as follows:

Table 1-36. Z8536 CIO Port Pins Assignment

| Port Pin | Signal Name | Direction | Descriptions |
|----------|-------------|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| PA0 | TM3_MID0 | Input | Port 3 Test Mode when IDREQ_ = 0; Module ID Bit 0 when IDREQ_ = 1. |
| PA1 | DSR3_MID1 | Input | Port 3 Data Set Ready when IDREQ_ = 0; Module ID Bit 1 when IDREQ_ = 1 |
| PA2 | RI3_ | Input | Port 3 Ring Indicator |
| PA3 | LLB3_MODSEL | Output | Port 3 Local Loopback (IDREQ_ = 0) or Port Select (IDREQ_ = 1): IDREQ_ = 1 & MODSEL = 1 =>Port 3 ID Select IDREQ_ = 1 & MODSEL = 0 => Port 4 ID Select |
| PA4 | RLB3_ | Output | Port 3 Remote Loopback |
| PA5 | DTR3_ | Output | Port 3 Data Terminal Ready |
| PA6 | BRDFAIL | Output | Board Fail: When set will cause FAIL LED to be lit |
| PA7 | IDREQ_ | Output | Module ID Request |
| PB0 | TM4_MID2 | Input | Port 4 Test Mode when IDREQ_ 0; Module ID Bit 2 when IDREQ_ = 1. |
| PB1 | DSR4_MID3 | Input | Port 4 Data Set Ready when IDREQ_ = 0; Module ID Bit 3 when IDREQ_ = 1. |
| PB2 | RI4 | Input | Port 4 Ring Indicator |
| PB3 | LLB4_ | Output | Port 4 Local Loopback |
| PB4 | RLB4_ | Output | Port 4 Remote Loopback |
| PB5 | DTR4_ | Output | Port 4 Data Terminal Ready |
| PB6 | ENUM_ | Input | Status of CPCI ENUM# signal. Used in hot swap systems. |
| PB7 | ABORT_ | Input | Status of ABORT# signal |

Table 1-36. Z8536 CIO Port Pins Assignment (Continued)

| Port Pin | Signal Name | Direction | Descriptions |
|----------|-------------|-----------|--------------|
| PC0 | Reserved | I/O | Reserved |
| PC1 | Reserved | I/O | Reserved |
| PC2 | Reserved | I/O | Reserved |
| PC3 | Reserved | I/O | Reserved |

Note The direction and the polarity of the Z8536's port pins are software programmable.

The module ID signals, which are only valid when IDREQ_ is asserted, indicate the type of the serial module that is installed on either Port 3 or Port 4 of the transition module. The following table shows how to interpret the MID3-MID0 signals:

Table 1-37. Interpretation of MID3-MID0

| IDREQ_ | LLB3_ MODSEL | MID3 | MID2 | MID1 | MID0 | Serial Module Type | Module Assembly Number |
|--------|-----------------|------|------|------|------|------------------------|---------------------------|
| 1 | X | X | X | X | X | Invalid module ID | |
| 0 | 0 | 0 | 0 | 0 | 0 | Module 3: EIA232 DCE | 01-W3876B01 |
| 0 | 0 | 0 | 0 | 0 | 1 | Module 3: EIA232 DTE | 01-W3877B01 |
| 0 | 0 | 0 | 0 | 1 | 0 | Module 3: EIA530 DCE | 01-W3878B01 |
| 0 | 0 | 0 | 0 | 1 | 1 | Module 3: EIA530 DTE | 01-W3879B01 |
| 0 | 0 | 1 | 1 | 1 | 1 | Module 3 Not Installed | |
| 0 | 1 | 0 | 0 | 0 | 0 | Module 4: EIA232 DCE | 01-W3876B01 |
| 0 | 1 | 0 | 0 | 0 | 1 | Module 4: EIA232 DTE | 01-W3877B01 |
| 0 | 1 | 0 | 0 | 1 | 0 | Module 3: EIA530 DCE | 01-W3878B01 |
| 0 | 1 | 0 | 0 | 1 | 1 | Module 3: EIA530 DTE | 01-W3879B01 |
| 0 | 1 | 1 | 1 | 1 | 1 | Module 4 Not Installed | |

Note Because IDREQ_ and MID3-MID0 signals go through the P2MX (P2 multiplexing) function must wait for the MID3-MID0 to become valid after asserting IDREQ_. The waiting time should be about 4 microseconds because the sampling rate is about 1.6 microsecond with a 10 MHz MXCLK clock.

ISA DMA Channels

There are seven ISA DMA channels in the PBC. Channels 0 through 3 support only 8-bit DMA devices while Channels 5 through 7 support only 16-bit DMA devices. These DMA channels are assigned as follows:

Table 1-38. PBC DMA Channel Assignments

| PBC Priority | PBC DRQ | DRQ SOURCE | PBC DACK | DACK DESTINATION | DMA Assignment |
|--------------|---------|----------------|----------|------------------|----------------------------------------------|
| Highest | DRQ0 | 85230 w_REQA | DACK0 | ISA Control PLD | Serial Port 3 Receive (Z85230 Port A Rx) |
| | DRQ1 | 85230 DTR_REQA | DACK1 | ISA Control PLD | Serial Port 3 Transmitter (Z85230 Port A Tx) |
| | DRQ2 | 87307 DRQ2 | DACK2 | 87307 DACK2 | Floppy Drive Controller |
| | DRQ3 | 87307 DRQ3 | DACK3 | 87307 DACK3 | Parallel Port |
| | | | | | Not available - Cascaded from DMA1 |
| | DRQ5 | 85230 W_REQB | DACK5 | ISA Control PLD | Serial Port 4 Receiver (Z85230 Port B Rx) |
| | DRQ6 | 85230 DTR_REQB | DACK6 | ISA Control PLD | Serial Port 4 Transmitter (Z85230 Port B Tx) |
| Lowest | DRQ7 | | | | Not Used |

Note Because the Z85230 is an 8-bit device and Channels 5 and 6 are 16-bit DMA Channels, only every other byte (the even bytes) from memory is valid.

Raven PCI Host Bridge & Multiprocessor Interrupt Controller

2

Introduction

Overview

This document describes the architecture and usage of the Raven PCI Host Bridge & Multiprocessor Controller (hereinafter referred to as Raven). The Raven is intended to provide PPC750 compliant devices access to devices residing on the PCI Local Bus in a very efficient manner.

Note Throughout the remainder of this chapter, the PPC750 bus will be referred to as the PPC bus. The PCI Local Bus will be referred to as PCI.

It is important to note that no manufacturer currently has plans to support the PPC bus directly. Therefore, some alternative I/O bus will be necessary in any PowerPC product. This I/O bus must be robust and efficient enough to handle the high bandwidth, burst oriented traffic required for Ethernet, SCSI, graphics, and VMEbus interfaces.

PCI is a high performance 32-bit or 64-bit, burst mode, synchronous bus capable of transfer rates of 132MB/sec in 32-bit mode or 264MB/sec in 64-bit mode using a 33 MHz clock.

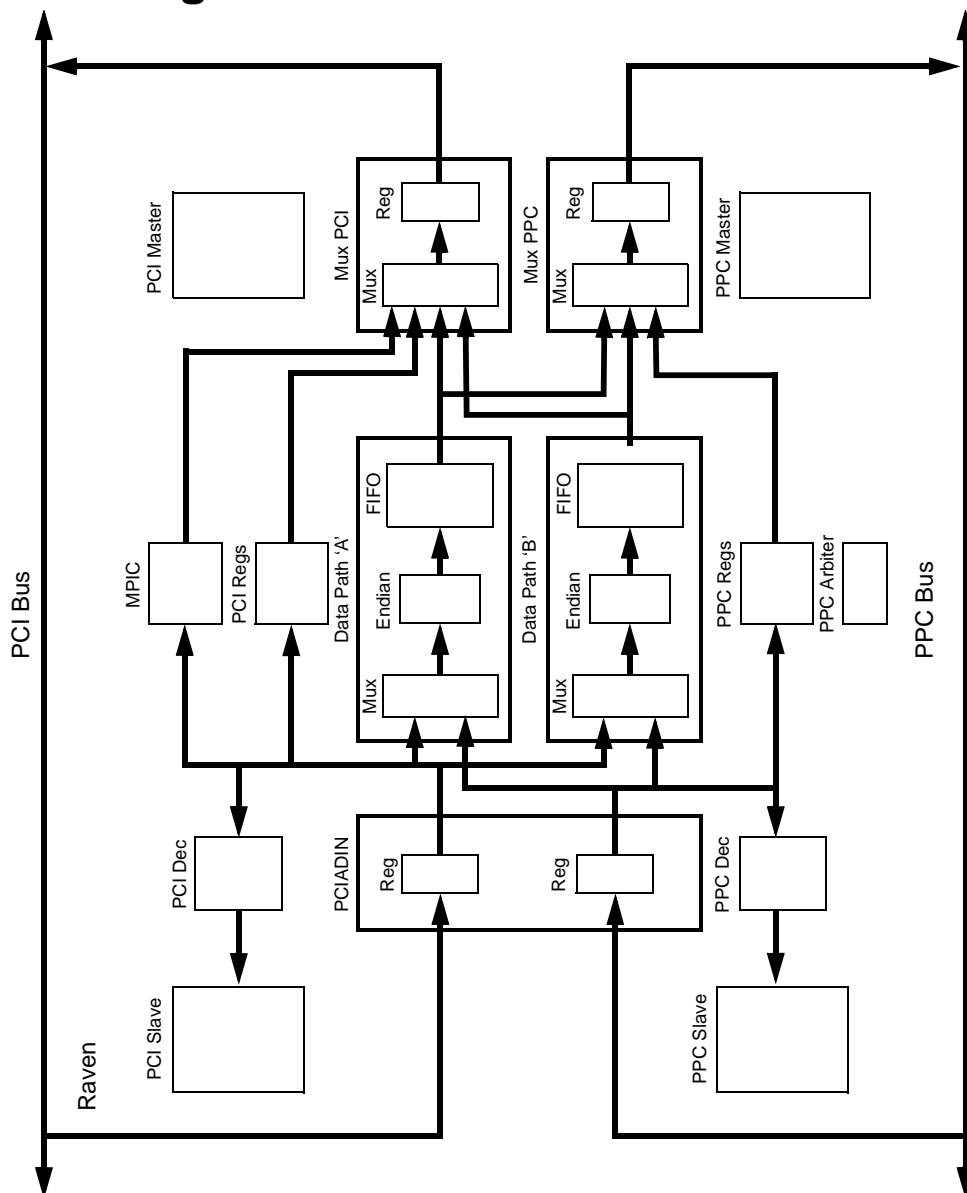
Capability

The Raven provides a high throughput interface between multiple PPC750 processors and 32/64-bit PCI local bus. It is capable of supporting up to two PPC750 processors and contain a multiprocessing interrupt structure to efficiently distribute interrupts dynamically between these processors.

Features

- ❑ PPC Bus Interface:
 - Direct interface to PPC750 processors
 - 64-bit data bus, 32-bit address bus
 - Optional bus arbitration logic supporting up to three bus masters
 - Four independent software programmable slave map decoders
 - Multi-level write post FIFO for writes to PCI
 - Support for PPC bus clock speeds up to 66 MHz
 - Selectable big- or little-endian operation
- ❑ PCI Interface:
 - PCI Rev. 2.0 compliant
 - 32-bit or 64-bit address/data bus
 - Support for accesses to all four PCI address spaces
 - Single-level write posting buffers for writes to the PPC bus
 - Read-ahead buffer for reads from the PPC bus
 - Four independent software programmable slave map decoders
- ❑ Interrupt Controller:
 - MPIC compliant.
 - Support for 16 external interrupt sources and two processors
 - Multiprocessor interrupt control allowing any interrupt source to be directed to either processor
 - Multilevel cross processor interrupt control for multiprocessor synchronization
 - Four 31 bit tick timers
- ❑ Two 64-bit general purpose registers for cross-processor messaging

Block Diagram



1914 9610

Figure 2-1. Raven Block Diagram

Functional Description

PPC Bus Interface

The PPC Bus Interface is designed to be coupled directly to up to two PPC750 microprocessors as well as a memory/cache subsystem. It uses a subset of the capabilities of the PPC750 bus protocol.

PPC Map Decoders

The Raven address decoders have been designed to be as flexible as possible to provide a wide range of addressing possibilities. There are five address map decoders in the Raven which determine the PPC bus addresses to which the Raven will respond: the PPC Register File Decoder, and four programmable decoders. The following table shows a typical CHRP compliant memory map:

Table 2-1. CHRP Compliant Memory Map

| PPC Address | Function |
|-----------------------|------------------------------------|
| \$00000000-\$7FFFFFFF | System Memory (2G) |
| \$80000000-\$FCFFFFFF | PCI Memory (2G - 48M) |
| \$FD000000-\$FDFFFFFF | ISA Memory (16M) |
| \$FE000000-\$FE7FFFFF | Discontiguous PCI IO (8M) |
| \$FE800000-\$FEBFFFFF | Contiguous PCI IO (4M) |
| \$FEC00000-\$FEF7FFFF | reserved (3.5M) |
| \$FEF80000-\$FEF8FFFF | Falcon 0 Registers (64K) |
| \$FEF90000-\$FEF9FFFF | Falcon 1 Registers (64K) |
| \$FEFA0000-\$FEFAFFFF | Falcon 2 Registers (64K) |
| \$FEFB0000-\$FEFBFFFF | Falcon 3 Registers (64K) |
| \$FEFC0000-\$FEFEFFFF | reserved (192K) |
| \$FEFF0000-\$FEFFFFFF | Raven Registers (64K) (EXT00 => 0) |
| \$FF000000-\$FFFFFFFF | System ROM/Flash (16MB) |

The PPC Register File decoder determines the address location of the Raven's PPC registers from the PPC bus. These registers may be accessed using only 1-, 2-, 3-, 4-, or 8-byte operations. The location of the PPC register file is fixed beginning at PPC address \$FEFE0000 or \$FEFF0000, depending on the state of the EXT01 bit at the time RST* is released. If the EXT01 pin is sampled in the low state, the PPC register file will start at address \$FEFE0000. If the EXT01 pin is sampled in the high state, the PPC register file will start at address \$FEFF0000. All references to the PPC register file within this specification will assume a base address of FEFF0000. Raven registers are described in detail later in this chapter.

The Raven includes four programmable decoders which control accesses from the PPC bus to the PCI bus. These decoders provide a window into the PCI bus from the PPC bus. The most significant 16 bits of the PPC address are compared with the address range of each map decoder, and if the address falls within the specified range, the access is passed on to PCI. For each map, there is an associated set of attributes. These attributes are used to enable read accesses, enable write accesses, enable write posting, and define the PCI transfer characteristics. Each map decoder also includes a programmable 16-bit address offset. The offset is added to the 16 most significant bits of the PPC address, and the result is used as the PCI address. This offset allows PCI devices to reside at any PCI address, independent of the PPC address map.

Note Care should be taken to ensure that all programmable decoders decode unique address ranges. Overlapping address ranges will lead to undefined operation.

PPC Write Posting

The PPC write FIFO stores up to eight data beats in any combination of *single* and *burst* transactions. If write posting is enabled, Raven stores the data necessary to complete an PPC write transfer to the PCI bus and immediately acknowledges the transaction on the PPC bus. This frees the PPC bus from waiting for the potentially long PCI arbitration and transfer. The PPC bus may be used for more useful work while the Raven manages the completion of the write posted transaction on PCI.

If the write post FIFO is full, any other accesses to the Raven are delayed (AACK* will not be asserted) until there is room in the FIFO to store the complete transaction.

All write posted transfers will be completed before a non-write posted read or write is begun to assure that all transfers are completed in the order issued. All write posted transfers will also be completed before any access to the Raven's registers is begun.

PPC Master

Wherever possible, the PPC master will attempt to consolidate data movement into a pair of burst transfers called couplets. If there is not enough data movement to perform a couplet, the PPC master will attempt singular burst transfers. The PPC master will perform single beat transfers as required during all non-cache aligned writes and some non-cache aligned reads. A 64-bit by 16 entry FIFO is used to hold data between the PCI slave and the PPC master to ensure that optimum data throughput is maintained. While the PCI slave is filling the FIFO with one cache line worth of data, the PPC master can be moving another cache line worth onto the PPC bus. This will allow the PCI slave to receive long block transfers without stalling.

The PPC master has an optional read ahead mode controlled by the RAEN bit in the PCISATT_x registers that allows the PPC master to prefetch data in bursts and store it in the FIFO. The contents of the FIFO will then be used to satisfy the data requirements for the remainder of the PCI read transaction. A second control bit within the PCISATT_x registers called TDIS (Threshold Disable) determines how the PPC master will continue to fill the FIFO during prefetched reads. The following table shows the read ahead options.

| TDIS | RAEN | PCI Command | Initial Read Size | Continuation | Subsequent Read Size |
|------|------|---------------|-------------------|---------------------------------------|----------------------|
| x | 0 | Read | 1 cache line | PCI received data and FRAME* asserted | 1 cache line |
| | | Read Line | | | |
| 0 | 1 | Read | 4 cache lines | FIFO <= 1 cache line | 2 cache lines |
| | | Read Line | | | |
| | x | Read Multiple | | | |
| 1 | 1 | Read | 4 cache lines | PCI received data and FRAME* asserted | 2 cache lines |
| | | Read Line | | | |
| | x | Read Multiple | | | |

Upon completion of a prefetched read transaction, any residual read data left within the FIFO will be invalidated (discarded). The Raven does not have a mechanism for snooping the PPC bus for transactions associated with the prefetched read data within its FIFO, therefore caution should be exercised when using the prefetch option within coherent memory space.

The user should select a PCI command type, a transaction size and a TDIS bit setting that minimizes wasted PPC bus bandwidth due to unnecessary prefetching. For example, there will be no unnecessary prefetching if the user always performs a burst read of 4 cache lines and the TDIS option is enabled. If the user wishes to perform very long burst transactions, then the TDIS option should be disabled since the benefits of a long uninterrupted transaction far exceed the penalty of a few unused prefetch cycles.

The PPC master will never perform prefetch reads beyond the address range mapped within the PCI slave map decoders. As an example, assume Raven has been programmed to respond to PCI address range \$10000000 through \$1001FFFF with an offset of \$2000. The PPC master will perform its last read on the PPC bus at cache line address \$3001FFFC or word address \$3001FFF8.

The PPC bus transfer types generated by the PPC master depend on the PCI command code and the INV/GBL bits in the PCISATTx registers. The GBL bit determines whether or not the GBL* signal is asserted for all portions of a transaction and is fully independent of the PCI command code and INV bit. Table 2-2 shows the relationship between PCI command codes and the INV bit:

Table 2-2. PPC Transfer Types

| PCI Command Code | INV | PPC Transfer Type | PPC Transfer Size | TT0-TT4 |
|-----------------------------------------------------------|-----|-------------------------------|-------------------|---------|
| Memory Read, Memory Read Multiple, Memory Read Line | 0 | Read | Burst/Single Beat | 01010 |
| Memory Read, Memory Read Multiple, Memory Read Line | 1 | Read With Intent to Modify | Burst/Single Beat | 01110 |
| Memory Write, Memory Write and Invalidate | x | Write with Kill | Burst | 00110 |
| Memory Write, Memory Write and Invalidate | x | Write with Flush | Single Beat | 00010 |

The PPC master incorporates an optional operating mode called Bus Hog. When Bus Hog is enabled, the PPC master will continually request the PPC bus for the entire duration of each PCI transfer. When Bus Hog is not enabled, the PPC master will structure its bus request actions around its desire to perform couplets. This means the bus request will be deasserted between couplets. Caution should be exercised when using this mode since the over-generosity of bus ownership to the PPC master can be detrimental to the host CPU's performance. The Bus Hog mode can be controlled by the BHOG bit within the GCSR. The default state for BHOG is disabled.

PPC Bus Timer

The PPC bus timer allows the current bus master to recover from a lock-up condition caused when no slave responds to the transfer request.

The time-out length of the bus timer is determined by the MBT field in the Global Control/Status Register.

The bus timer starts ticking at the beginning of an address transfer (TS* asserted), and if the address transfer is not terminated (AACK* asserted) before the time-out period has passed, the Raven will assert the MATO bit in the PPC Error Status Register, latch the PPC address in the PPC Error Address Register, and then immediately assert AACK*.

The MATO bit may be configured to generate an interrupt or a machine check through the MEREN register.

The timer is disabled if the transfer is intended for PCI. PCI bound transfers will be timed by the PCI master.

PCI Interface

The Raven PCI Interface is designed to connect directly to a PCI Local Bus compliant I/O bus.

The PCI interface may operate at any clock speed up to 33 MHz. The PCLK input must be externally synchronized with the MCLK input, and the frequency of the PCLK input must be exactly half the frequency of the MCLK input.

PCI Map Decoders

The Raven contains four programmable decoders which provide windows into the PPC bus from the PCI bus. The most significant 16 bits of the PCI address is compared with the address range of each map decoder, and if the address falls within the specified range, the access is passed on to the PPC bus. For each map, there is an independent set of attributes. These attributes are used to enable read accesses, enable write accesses, enable write posting, and define the PPC bus transfer characteristics. Each map decoder also includes a programmable 16-bit address offset.

The offset is added to the 16 most significant bits of the PCI address, and the result is used as the PPC address. This offset allows devices to reside at any PPC address, independent of the PCI address map.

All Raven address decoders are prioritized so that programming multiple decoders to respond to the same address will not be a problem. When the PCI address falls into the range of more than one decoder, only the highest priority one will respond. The decoders are prioritized as shown below.

| Decoder | Priority |
|-------------|----------|
| PCI Slave 0 | highest |
| PCI Slave 1 | |
| PCI Slave 2 | |
| PCI Slave 3 | lowest |

PCI Configuration Space

The Raven does not have an IDSEL pin. An internal connection is made within the Raven that logically associates the assertion of IDSEL with the assertion of either AD30 or AD31. The exact association depends on the state of the EXT02 pin when the RST* pin is released. If EXT02 is sampled low, the Raven will associate AD30 with IDSEL. If EXT02 is sampled high, the Raven will associate AD31 with IDSEL.

PCI Write Posting

If write posting is enabled, the Raven stores the target address, attributes, and up to 128 bytes of data from one PCI write transaction and immediately acknowledges the transaction on the PCI bus. This allows the slower PCI to continue to transfer data at its maximum bandwidth, and the faster PPC bus to accept data in high performance cache-line burst transfers.

Only one PCI transaction may be write posted at any given time. If the Raven is busy processing a previous write posted transaction when a new PCI transaction begins, the next PCI transaction will be delayed (TRDY* will not be asserted) until the previous transaction has completed.

If during a transaction the write post buffer gets full, subsequent PCI data transfers will be delayed (TRDY* will not be asserted) until the Raven has removed some data from the FIFO. Under normal conditions, the Raven should be able to empty the FIFO faster than the PCI bus can fill it.

PCI Configuration cycles intended for internal Raven registers will also be delayed if Raven is busy so that control bits which may affect write posting do not change until all write posted transactions have completed.

PCI Master

The PCI master, in conjunction with the capabilities of the PPC slave, will attempt to move data in either single beat or burst transactions. All single beat transactions will be subdivided into one or two 32-bit transfers, depending on the alignment and size of the transaction. The PCI master will attempt to transfer all burst transactions in 64-bit mode. If at any time during the transaction the PCI target indicates it can not support 64-bit mode, the PCI master will continue to transfer the remaining data in 32-bit mode.

The PCI Command Codes generated by the PCI master depend on the PPC transfer type, TBST*, and the MEM field in the MSATTx registers. Refer to Table 2-3.

Table 2-3. PCI Command Codes

| PPC Transfer Type | TBST* | MEM | PCI Command |
|-----------------------|-------|-----|-------------------------|
| Write w/ Flush | x | 1 | 0111 (Memory Write) |
| Write w/ Flush Atomic | x | 0 | 0011 (I/O Write) |
| Write w/ Kill | | | |
| Graphics Write | | | |
| Read | 0 | 1 | 1110 (Memory Read Line) |
| Read w/ ITM | 1 | 1 | 0110 (Memory Read) |
| Read w/ ITM Atomic | x | 0 | 0010 (I/O Read) |
| Graphics Read | | | |

Generating PCI Memory and I/O Cycles

Each programmable slave may be configured to generate PCI I/O or memory accesses through the MEM and IOM fields in its Attribute register as shown below.

| MEM | IOM | PCI Cycle Type |
|-----|-----|----------------|
| 1 | x | Memory |
| 0 | 0 | Contiguous I/O |
| 0 | 1 | Spread I/O |

The IBM CHRP specification describes two approaches for handling PCI I/O addressing: contiguous or spread address modes. When the MEM bit is cleared, the IOM bit is used to select between these two modes whenever a PCI I/O cycle is to be performed.

When MEM is clear or IOM is clear, the Raven will take the PPC address, apply the offset specified in the MSOFFx register, and map the result directly to PCI.

When MEM is clear and IOM is set, the Raven will take the PPC address, apply the offset specified in the MSOFFx register, and map the result to PCI as shown in the figure below:

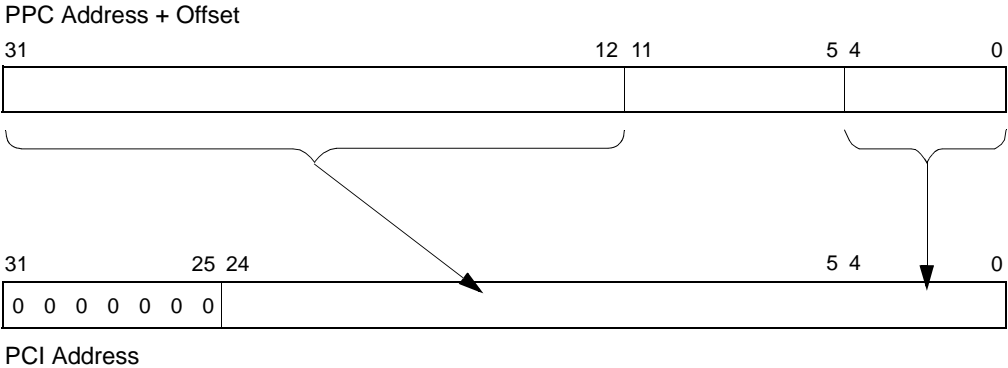


Figure 2-2. PCI Spread I/O Cycle Mapping

This CHRP compliant spread I/O mode allows each PCI device's I/O registers to reside on a different PPC memory page, so device drivers can be protected from each other using memory page protection.

All I/O accesses must be performed within natural word boundaries. Any I/O access that is not contained within a natural word boundary will result in unpredictable operation. For example, an I/O transfer of 4 bytes starting at address \$80000010 is considered a valid transfer. An I/O transfer of 4 bytes starting at address \$80000011 is considered an invalid transfer since it crosses the natural word boundary at address \$80000013/\$80000014.

Generating PCI Configuration Cycles

Mechanism one (as just described above) is utilized to generate configuration cycles. Two 32 bit PCI I/O ports at \$CF8 and \$CFC are used to access PCI configuration space. One of the four PPC Slave Address Registers is used to gain access to \$CF8 and \$CFC. Note that MSADD3 is initialized at reset to access PCI I/O space with an PPC address of \$80000000.

The resource at \$CF8 is a 32 bit configuration address port and is referred to as the CONFIG_ADDRESS register. The resource at \$CFC is a 32 bit configuration data port and is referred to as the CONFIG_DATA register.

Accessing a PCI functions' configuration port is a two step process:

1. Write the bus number, physical device number, function number and register number to the CONFIG_ADDRESS register.
2. Perform an I/O read from or a write to the CONFIG_DATA register.

Generating PCI Special Cycles

To prime Raven to generate a special cycle, the host processor must write a 32 bit value to the CONFIG_ADDRESS register. The contents of the write are defined later in this chapter under the CONFIG_ADDRESS register definition. After the write to \$CF8 has been accomplished, the next write to the CONFIG_DATA register causes the Raven to generate a special cycle on the PCI bus. The write data is driven onto AD[31:0] during the special cycle's data phase.

Generating PCI Interrupt Acknowledge Cycles

Performing a read from the PIACK register will initiate a single PCI Interrupt Acknowledge cycle. Any single byte or combination of bytes may be read from, and the actual byte enable pattern used during the read will be passed on to the PCI bus. Upon completion of the PCI interrupt acknowledge cycle, the Raven will present the resulting vector information obtained from the PCI bus as read data.

Endian Conversion

The Raven supports both big- and little-endian data formats. Since PCI is inherently little-endian, conversion is necessary if all PPC devices are configured for big-endian operation. The Raven may be programmed to perform the Endian conversion described below.

When PPC Devices are Big-Endian

When all PPC devices are operating in big-endian mode, all data to/from PCI must be swapped such that PCI looks big-endian from the PPC bus's perspective. Refer to the following figure.

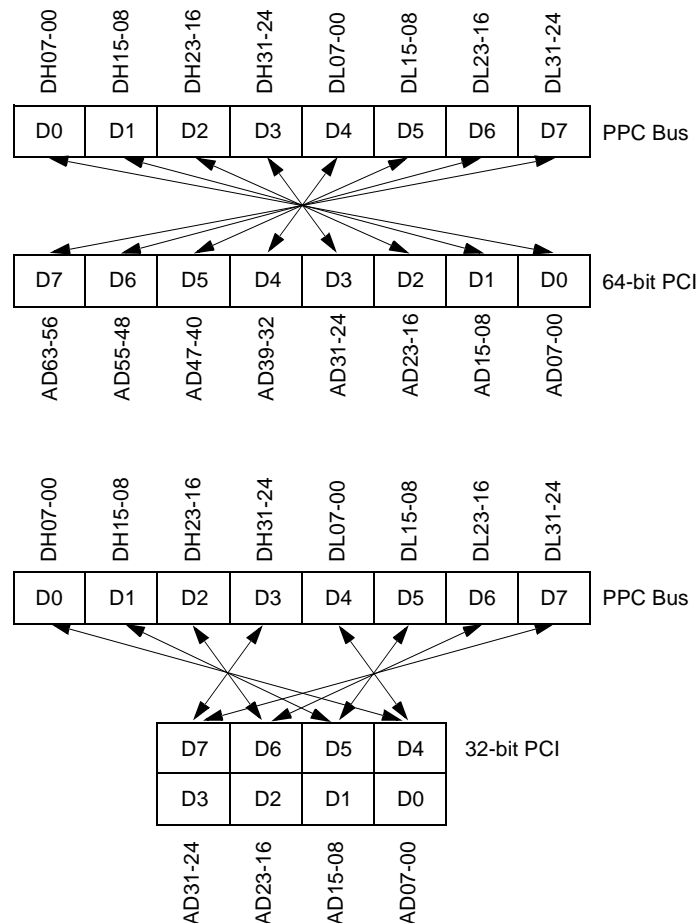


Figure 2-3. Big- to Little-Endian Data Swap

When PPC Devices are Little-Endian

When all PPC devices are operating in little-endian mode, the PPC address must be modified to remove the exclusive-ORing applied by PPC60x processors before being passed on to PCI. The three low order processor bus address bits are exclusive-ORed with a three-bit value that depends on the length of the operand, as shown in the following table.

Table 2-4. Address Modification for Little Endian Transfers

| Data Length (bytes) | Address Modification |
|--------------------------------|-----------------------------|
| 1 | XOR with 111 |
| 2 | XOR with 110 |
| 4 | XOR with 100 |
| 8 | no change |

Note The only legal data lengths supported in little-endian mode are 1, 2, 4, or 8-byte aligned transfers.

Cycles Originating From PCI

For bus cycles initiated by PCI masters, the PCI address will be modified the same way the MCP60x processor does in little-endian mode. The modification will be the same as that described in the section above. Since this method has some difficulties dealing with unaligned transfers, the Raven will break up all unaligned PCI transfers into multiple aligned transfers on the PPC bus.

Error Handling

The Raven is capable of detecting and reporting the following errors to one or more PPC masters:

- ☐ PPC address bus time-out
- ☐ PCI master signalled master abort
- ☐ PCI master received target abort
- ☐ PCI parity error
- ☐ PCI system error

Each of these error conditions will cause an error status bit to be set in the PPC Error Status Register. If a second error is detected while any of the error bits is set, the OVFL bit is asserted, but none of the error bits are changed. Each bit in the PPC Error Status Register may be cleared by writing a 1 to it; writing a 0 to it has no effect. New error bits may be set only when all previous error bits have been cleared.

When any bit in the PPC Error Status register is set, the Raven will attempt to latch as much information as possible about the error in the PPC Error Address and Attribute Registers. Information is saved as follows:

| Error Status | Error Address and Attributes |
|--------------|------------------------------|
| MATO | From PPC bus |
| SMA | From PCI bus |
| RTA | From PCI bus |
| PERR | Invalid |
| SERR | Invalid |

Each MERST error bit may be programmed to generate a machine check and/or a standard interrupt. The error response is programmed through the PPC Error Enable Register on a source by source basis. When a machine check is enabled, either the MID field in the PPC Error Attribute Register or the DFLT bit in the MEREN Register determine the master to which the machine check is directed. For errors in which the master who originated the transaction can be determined, the MID field is used, provided the MID is %00 (processor 0), %01 (processor 1), or %10 (processor 2). For errors not associated with a particular PPC master, or associated with masters other than processor 0, 1, or 2, the DFLT bit is used. One example of an error condition which cannot be associated with a particular PPC master would be a PCI system error.

Watchdog Timers

Raven features two watchdog timers called Watchdog Timer 1 (WDT1) and Watchdog Timer 2 (WDT2). Although both timers are functionally equivalent, each timer operates completely independent of each other. WDT1 and WDT2 are initialized at reset to a time-out value of 512 milliseconds and 576 milliseconds respectively.

The timers are designed to be reloaded by software at any time. When not being loaded, the timer will continuously decrement itself until either reloaded by software or a count of zero is reached. If a timer reaches a count of zero, an output signal will be asserted and the count will remain at zero until reloaded by software or Raven's reset is asserted.

The output of WDT1 is routed to MPIC external interrupt IRQ4. The output of WDT2 is logically onboard to provide a hard reset. Each output is delayed an additional 4.5 seconds (typical), using logic external to Raven. This external delay is fixed and not programmable.

Each timer is composed of a prescaler and a counter. The prescaler determines the resolution of the timer, and is programmable to any value between 1 and 16 microseconds. The counter counts in the units provided by the prescaler. For example, the watchdog timer would reach a count of zero within 24 us if the prescaler was programmed to 2 us and the counter was programmed to 12.

The watchdog timers are controlled by registers mapped within the PPC control register space. Each timer has a WDTxCNTL register and a WDTxSTAT register. The WDTxCNTL register can be used to start or stop the timer, write a new reload value into the timer, or cause the timer to initialize itself to a previously written reload value. The WDTxSTAT register is used to read the instantaneous count value of the watchdog timer.

Programming of the Watchdog Timers is performed through the WDTxCNTL register and is a two step process.

1. Arm the WDTxCNTL register by writing PATTERN_1 into the KEY field. Only the KEY byte lane may be selected during this process. The WDTxCNTL register will not arm itself if any of the other byte lanes are selected or the KEY field is written with any

other value than PATTERN_1. The operation of the timer itself remains unaffected by this write.

2. Write the new programming information to the WDTxCNTL register. The KEY field byte lane must be selected and must be written with PATTERN_2 for the write to take affect. The effects on the WDTxCNTL register depend on the byte lanes that are written to during step 2 and are shown in the following table.

Table 2-5. WDTxCNTL Programming

| Byte Lane Selection | | | | Results | | | |
|---------------------|--------------|--------|-------|-------------------------|----------------------------|-------------------------|----------------------------|
| KEY | ENAB/ RES | RELOAD | | WDT | | WDTxCNTL Register | |
| 0:7 | 8:15 | 16:23 | 24:31 | Prescaler/Enable | Counter | RES/ENAB | RELOAD |
| No | x | x | x | No Change | No Change | No Change | No Change |
| Yes | No | No | x | Update from RES/ENAB | Update from RELOAD | No Change | No Change |
| Yes | No | x | No | Update from RES/ENAB | Update from RELOAD | No Change | No Change |
| Yes | Yes | No | x | Update from data bus | Update from RELOAD | Update from data bus | No Change |
| Yes | Yes | x | No | Update from data bus | Update from RELOAD | Update from data bus | No Change |
| Yes | Yes | Yes | Yes | Update from data bus | Update from data bus | Update from data bus | Update from data bus |

The WDTxCNTL register will always become unarmed after the second write regardless of byte lane selection. Reads may be performed at any time from the WDTxCNTL register and will not affect the write arming sequence.

PCI/PPC Contention Handling

The Raven has a stall detection mechanism that detects when there is a possible resource contention problem (that is, deadlock) as a result of overlapping PPC and PCI initiated transactions. The PPC Slave and the PCI Slave functions contain the logic needed to implement this feature.

The PCI Slave function contributes to the stall detection mechanism by issuing a stall signal to the PPC Slave function whenever it is currently processing a transaction that must have control of the PPC bus before the transaction can be completed. The events that activate this signal are:

- ❑ PCI read cycle
- ❑ PCI non-posted write cycle
- ❑ PCI posted write cycle with flush-before-read (FLBRD) enabled
- ❑ PCI posted write cycle and internal FIFO full

The PPC Slave function determines its future actions based on the stall signal and the current PPC bus activity. If the PPC Slave function determines there will be contention between a cycle completing on the PPC bus and an incoming PCI cycle, the PPC Slave will issue a retry for the current PPC transaction. This retry will free up the PPC bus and allow the PCI initiated transaction to complete. An idle PPC bus obviously gives immediate access to the pending PCI initiated transaction.

If the PPC bus is currently supporting a read cycle of any type, the cycle will be terminated with a retry. Note that if the read cycle is across a mod-4 address boundary (that is, from address 0x...02, 3 bytes), it is possible that a portion of the read could have been completed before the stall condition was detected. The previously read data will be discarded and the current transaction will be retried.

If the PPC bus is currently supporting a posted write transaction, the transaction will be allowed to complete since this type of transaction is guaranteed completion. If the PPC bus is currently supporting a non-posted write transaction, the transaction will be terminated with a retry.

Note A mod-4 non-posted write transaction could be interrupted between write cycles, and thereby result in a partially completed write cycle. It is recommended that write cycles to write-sensitive non-posted locations be performed on mod-4 address boundaries.

The Raven has a programmable option to guarantee all PCI write posted transactions are completed before an PPC initiated read transaction may be allowed to complete. This option is controlled by the FLBRD bit in the GCSR register. If this bit is set, all PPC read transactions will be retried until all posted PCI write transactions have completed. It is recommended that this option be disabled, and the FLBRD bit be left in the default (disabled) state.

Transaction Ordering

The PCI Specification 2.1, listed in [Appendix A, Related Documentation](#), states that posted write buffers in both directions must be flushed before completing a read in either direction. Raven supports this by providing two optional FIFO flushing options. The PPCFBR (PPC Flush Before Read) bit within the GCSR register controls the flushing of PCI write posted data when performing PPC originated read transactions. The PCIFBR bit within the GCSR register controls the flushing of PPC write posted data when performing PCI originated read transactions. The PCIFBR and PPCFBR functions are completely independent of each other, however both functions must be enabled to guarantee full compliance with PCI Specification 2.0.

When the PPCFBR bit is set, Raven will handle read transactions originating from the PPC bus in the following manner:

- ❑ Write posted transactions originating from the processor bus are flushed by the nature of the FIFO architecture. The Raven will hold the processor with wait states until the PCI bound FIFO is empty.
- ❑ Write posted transactions originated from the PCI bus are flushed in the following manner. The PCI slave sets a signal called *pcis_fbrabt* anytime it has committed to performing a posted write transaction.

This signal will remain asserted until the PPC bound FIFO count has reached zero.

The PPC slave address decode logic settles out several clocks after the assertion of TS*, at which time the PPC slave can determine the transaction type. If it is a read and PPCFBR is enabled, the PPC slave will look at the *pcis_fbrabt* signal. If this signal is active, the PPC slave will retry the processor.

When the PCIFBR bit is set, Raven will handle read transactions originating from the PCI bus in the following manner:

- ❑ Write posted transactions originating from the PCI bus are flushed by the nature of the FIFO architecture. The Raven will hold the PCI master with wait states until the PPC bound FIFO is empty.
- ❑ Write posted transactions originated from the PPC bus are flushed in the following manner. The PPC slave will set a signal called *ppcs_fbrabt* anytime it has committed to performing a posted write transaction. This signal will remain asserted until the PCI bound FIFO count has reached zero.

The PCI slave decode logic settles out several clocks after the assertion of FRAME*, at which time the PCI slave can determine the transaction type. If it is a read and PCIFBR is enabled, the PCI slave will look at the *ppcs_fbrabt* signal. If this signal is active, the PCI slave will retry the PCI master.

Registers

This section provides a detailed description of all Raven registers. The Raven ASIC functions as the PPC to PCI Host Bridge. It also contains an interrupt controller, the Raven MPIC. The Raven ASIC has three sets of registers: the *PPC Registers*, the *PCI Registers*, and the *MPIC Registers*. The PPC Registers are described first; the PCI Configuration Registers next, and the MPIC Register set last.

The following conventions are used in the Raven register charts.

- ❑ R Read Only field.

- R/W Read/Write field.
- S Writing a ONE to this field sets this field.
- C Writing a ONE to this field clears this field.

PPC Registers

The Raven PPC register map is shown in the following table.

Table 2-6. Raven PPC Register Map

| Bit ---> | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | |
|------------|----------|---|---|---|---|---|---|---|-------|---|---|---|---|---|---|---|-------|---|---|---|---|---|---|---|----------|---|---|---|---|---|---|--|
| \$FEFF0000 | VENID | | | | | | | | | | | | | | | | DEVID | | | | | | | | | | | | | | | |
| \$FEFF0004 | | | | | | | | | REVID | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0008 | GCSR | | | | | | | | | | | | | | | | FEAT | | | | | | | | | | | | | | | |
| \$FEFF000C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0010 | | | | | | | | | | | | | | | | | | | | | | | | | PADJ | | | | | | | |
| \$FEFF0014 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0018 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF001C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0020 | ERRTST | | | | | | | | | | | | | | | | ERREN | | | | | | | | | | | | | | | |
| \$FEFF0024 | | | | | | | | | | | | | | | | | | | | | | | | | ERRST | | | | | | | |
| \$FEFF0028 | ERRAD | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF002C | | | | | | | | | | | | | | | | | ERRAT | | | | | | | | | | | | | | | |
| \$FEFF0030 | PIACK | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0034 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0038 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF003C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0040 | PPCSADD0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0044 | PPCSOFF0 | | | | | | | | | | | | | | | | | | | | | | | | PPCSATT0 | | | | | | | |
| \$FEFF0048 | PPCSADD1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF004C | PPCSOFF1 | | | | | | | | | | | | | | | | | | | | | | | | PPCSATT1 | | | | | | | |

Table 2-6. Raven PPC Register Map (Continued)

| Bit ----> | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 |
|------------|---------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|----------|---|---|---|---|---|---|---|
| \$FEFF0050 | PPCSADD2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0054 | PPCSOFF2 | | | | | | | | | | | | | | | | | | | | | | | | PPCSATT2 | | | | | | | |
| \$FEFF0058 | PPCSADD3 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF005C | PPCSOFF3 | | | | | | | | | | | | | | | | | | | | | | | | PPCSATT3 | | | | | | | |
| \$FEFF0060 | WDT1CNTL | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0064 | | | | | | | | | | | | | | | | | | | | | | | | | WDT1STAT | | | | | | | |
| \$FFE0068 | WDT2CNTL | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF006C | | | | | | | | | | | | | | | | | | | | | | | | | WDT2STAT | | | | | | | |
| \$FEFF0070 | GPREG0(Upper) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0074 | GPREG0(Lower) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF0078 | GPREG1(Upper) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| \$FEFF007C | GPREG1(Lower) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Vendor ID/Device ID Registers

| Address | \$FEFF0000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 |
| Name | VENID | | | | | | | | | | | | | | | | DEVID | | | | | | | | | | | | | | | |
| Operation | R | | | | | | | | | | | | | | | | R | | | | | | | | | | | | | | | |
| Reset | \$1057 | | | | | | | | | | | | | | | | \$4801 | | | | | | | | | | | | | | | |

VENID Vendor ID. This register identifies the manufacturer of the device. This identifier is allocated by the PCI SIG to ensure uniqueness. \$1057 has been assigned to Motorola. This register is duplicated in the PCI Configuration Registers.

DEVID Device ID. This register identifies this particular device. The Raven will always return \$4801. This register is duplicated in the PCI Configuration Registers.

Revision ID Register

| Address | \$FEFF0004 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|-------|---|---|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 |
| Name | | | | | | | | | | | REVID | | | | | | | | | | | | | | | | | | | | | |
| Operation | R | | | | | | | | | | R | | | | | | | | | | R | | | | | | | | | | | |
| Reset | \$00 | | | | | | | | | | \$03 | | | | | | | | | | \$00 | | | | | | | | | | | |

REVID Revision ID. This register identifies the Raven revision level. This register is duplicated in the PCI Configuration Registers.

General Control-Status/Feature Registers

| Address | \$FEFF0008 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|--------|------|--------|------|------|-----|------|---|---|---|------|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 |
| Name | GCSR | | | | | | | | | | | | | | | | FEAT | | | | | | | | | | | | | | | |
| | LEND | | | PCIFBR | BHOG | PPCFBR | PBT1 | PBT0 | P64 | OPIC | | | | MID1 | MID0 | EXT15 | EXT14 | EXT13 | EXT12 | EXT11 | EXT10 | EXT09 | EXT08 | EXT07 | EXT06 | EXT05 | EXT04 | EXT03 | EXT02 | EXT01 | EXT00 | |
| Operation | R/W | R | R | R/W | R/W | R/W | R/W | R/W | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | 0/1 | |

LEND Endian Select. If set, the PPC bus is operating in little endian mode. The PPC address will be modified as described in the section *When PPC Devices are Little-Endian*. When LEND is clear, the PPC bus is operating in big-endian mode, and all data to/from PCI is swapped as described in the section *When PPC Devices are Big-Endian*.

PCIFBR PCI Flush Before Read. If set, the Raven will guarantee that all PPC initiated posted write transactions will be completed before any PCI initiated read transactions will be allowed to complete. When PCIFBR

is clear, there will be no correlation between these transaction types and their order of completion. Please refer to the section on [PCI/PPC Contention Handling](#) for more information.

BHOG Bus Hog. If set, the Raven PPC master will operate in the Bus Hog mode. Bus Hog mode means the PPC master will continually request the PPC bus for the entire duration of each PCI transfer. If Bus Hog is not enabled, the PPC master will request the bus in a normal manner. Please refer to the section on [PPC Master](#) for more information.

PPCFBR PPC Flush Before Read. If set, the Raven will guarantee that all PCI initiated posted write transactions will be completed before any PPC initiated read transactions will be allowed to complete. When PPCFBR is clear, there will be no correlation between these transaction types and their order of completion. Please refer to the section on [PCI/PPC Contention Handling](#) for more information.

PBTx PPC Bus Time-out. This field specifies the PPC bus time-out length. The time-out length is encoded as follows:

| MBT | Time Out Length |
|-----|-----------------|
| 00 | 256 μ sec |
| 01 | 64 μ sec |
| 10 | 8 μ sec |
| 11 | disabled |

P64 64-bit PCI Mode Enable. If set, the Raven is connected to a 64-bit PCI bus. Please refer to the *Hardware Configuration* section in the *MCP750 Compact PCI Single Board Computer Installation and Use Guide*, listed in [Appendix A, Related Documentation](#), for more information on how this bit gets set.

OPIC OpenPIC Interrupt Controller Enable. If set, the Raven internal OpenPIC interrupt controller is enabled. If cleared, Raven detected errors will be passed on to processor 0 INT pin. Please refer to the *Hardware Configuration* section in the *MCP750 Compact PCI Single Board Computer Installation and Use Guide* for more information on how this bit gets set.

MIDx Master ID. This field is encoded as shown below to indicate who is currently the PPC bus master. This information is obtained by sampling the CPUID pins. In a multiprocessor environment, these bits allow software to determine on which processor it is currently running. The internal PPC arbiter encodes this field as follows:

| MID | Current PPC Data Bus Master |
|-----|-----------------------------|
| 00 | device on ABG0* |
| 01 | device on ABG1* |
| 10 | device on ABG2 |
| 11 | Raven |

FEAT Feature Register. Each bit in this register reflects the state of one of the external interrupt input pins on the rising edge of RESET*. This register may be used to report hardware configuration parameters to system software.

Prescaler Adjust Register

| Address | \$FEFF0010 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 |
| Name | | | | | | | | | | | | | | | | | | | | | | | | | PADJ | | | | | | | |
| Operation | R | | | | | | | | R | | | | | | | | R | | | | | | | | R/W | | | | | | | |
| Reset | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | \$BE | | | | | | | |

PADJ Prescaler Adjust. This register is used to specify a scale factor for the prescaler to ensure that the time base for the bus timer is 1 MHz. The scale factor is calculated as follows:

$$\text{PADJ} = 256 - \text{Clk},$$

where Clk is the frequency of the CLK input in MHz. The following table shows the scale factors for some common CLK frequencies.

| Frequency | PADJ |
|-----------|------|
| 66 | \$BE |
| 50 | \$CE |
| 40 | \$D8 |
| 33 | \$DF |
| 25 | \$E7 |

PPC Error Enable Register

| Address | \$FEFF0020 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|------|------|------|------|------|------|------|---|---|----|----|----|----|----|----|-------|-------|-------|-------|-------|------|------|----|----|-------|-------|-------|-------|------|------|-----|-----|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | | |
| Name | ERRTST | | | | | | | | | | | | | | | | ERREN | | | | | | | | | | | | | | | | | |
| | DPE0 | DPE1 | DPE2 | DPE3 | DPE4 | DPE5 | DPE6 | DPE7 | | | | | | | | | DFLT | PATOM | PDPEM | PERRM | SERRM | SMAM | RTAM | | | PATOI | PDPEI | PERRI | SERRI | SMAI | RTAI | | | |
| Operation | R/w | R/w | R/w | R/w | R/w | R/w | R/w | R/w | R | | | | | | | | R | R/w | R/w | R/w | R/w | R/w | R/w | R | R | R/w | R/w | R/w | R/w | R/w | R/w | R/w | R/w | |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | | | | | | | | | \$00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

DPEx Data Parity Error Enable. These bits are used for test reasons to purposely inject data parity errors whenever Raven is sourcing PPC data. A data parity error will be created on the corresponding PPC data parity bus if a bit is set. For example, setting DPE0 will cause DP0 to be generated incorrectly. If the bit is cleared, Raven will generate correct data parity. These bits only have meaning if PPC data parity mode is enabled.

DFLT Default PPC Master ID. This bit determines which MCHK* pin will be asserted for error conditions in which the PPC master ID cannot be determined or the Raven was the PPC master. For example, in event of a PCI parity error for a transaction in which the Raven's PCI master was not involved, the PPC master ID cannot be determined. When DFLT is set, MCHK1* is used. When DFLT is clear, MCHK0* will be used.

PATOM PPC Address Bus Time-out Machine Check Enable. When this bit is set, the PATO bit in the ERRST register will be used to assert the MCHK output to the current address bus master. When this bit is clear, MCHK will not be asserted.

PDPEM PPC Data Parity Error Machine Check Enable. When this bit is set, the PDPE bit in the ERRST register will be used to assert the MCHK output to the current address bus master. When this bit is clear, MCHK will not be asserted.

PERRM PCI Parity Error Machine Check Enable. When this bit is set, the PERR bit in the ERRST register will be used to assert the MCHK output to bus master 0. When this bit is clear, MCHK will not be asserted.

SERRM PCI System Error Machine Check Enable. When this bit is set, the PERR bit in the ERRST register will be used to assert the MCHK output to bus master 0. When this bit is clear, MCHK will not be asserted.

SMAM PCI Signalled Master Abort Machine Check Enable. When this bit is set, the SMA bit in the ERRST register will be used to assert the MCHK output to the bus master which initiated the transaction. When this bit is clear, MCHK will not be asserted.

RTAM PCI Master Received Target Abort Machine Check Enable. When this bit is set, the RTA bit in the ERRST register will be used to assert the MCHK output to the bus master which initiated the transaction. When this bit is clear, MCHK will not be asserted.

PATOI PPC Address Bus Time-out Interrupt Enable. When this bit is set, the PATO bit in the ERRST register will be used to assert an interrupt through the OpenPIC interrupt controller. When this bit is clear, no interrupt will be asserted.

PDPEI PPC Data Parity Error Interrupt Enable. When this bit is set, the PDPE bit in the ERRST register will be used to assert an interrupt through the OpenPIC interrupt controller. When this bit is clear, no interrupt will be asserted.

PERRI PCI Parity Error Interrupt Enable. When this bit is set, the PERR bit in the ERRST register will be used to assert an interrupt through the OpenPIC interrupt controller. When this bit is clear, no interrupt will be asserted.

SERRI PCI System Error Interrupt Enable. When this bit is set, the PERR bit in the ERRST register will be used to assert an interrupt through the OpenPIC interrupt controller. When this bit is clear, no interrupt will be asserted.

SMAI PCI Master Signalled Master Abort Interrupt Enable. When this bit is set, the SMA bit in the ERRST register will be used to assert an interrupt through the OpenPIC interrupt controller. When this bit is clear, no interrupt will be asserted.

RTAI PCI Master Received Target Abort Interrupt Enable. When this bit is set, the RTA bit in the ERRST register will be used to assert an interrupt through the OpenPIC interrupt controller. When this bit is clear, no interrupt will be asserted.

PPC Error Status Register

| Address | \$FEFF0024 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|------|---|----|----|----|----|----|----|------|----|----|----|----|----|----|----|-------|----|------|------|------|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|---|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | | | | | | | | | |
| Name | | | | | | | | | | | | | | | | | | | | | | | | | ERRST | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | OVF | | PATO | PDPE | PERR | SERR | SMA | RTA | | | | | | | | | |
| Operation | R | | | | | | | | R | | | | | | | | R | | | | | | | | R/C | R | R/C | R/W | R/C | R/C | R/C | R/C | R/C | R/C | R/C | R/C | R/C | R/C | R/C | | |
| Reset | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

OVF Error Status Overflow. This bit is set when any error is detected and any of the error status bits are already set. It may be cleared by writing a 1 to it; writing a 0 to it has no effect.

PATO PPC Address Bus Time-out. This bit is set when the PPC address bus timer times out. It may be cleared by writing it to a 1; writing it to a 0 has no effect. When the PATOM bit in the ERREN register is set, the assertion of this bit will assert MCHK to the master designated by the MID field in the ERRAT register. When the PATOI bit in the ERREN register is set, the assertion of this bit will assert an interrupt through the MPIC interrupt controller.

PDPE PPC Data Parity Error. This bit is set when Raven detects a data bus parity error. This bit is only valid if PPC data bus parity mode is enabled. It may be cleared by writing it to a 1; writing it to a 0 has no effect. When the PDPEM bit in the ERREN register is set, the assertion of this bit will assert MCHK to the master designated by the MID field in the ERRAT register. When the PDPEI bit in the ERREN register is set, the assertion of this bit will assert an interrupt through the OpenPIC interrupt controller.

PERR PCI Parity Error. This bit is set when the PCI PERR* pin is asserted. It may be cleared by writing it to a 1; writing it to a 0 has no effect. When the PERRM bit in the ERREN register is set, the assertion of this bit will assert MCHK to the master designated by the DFLT bit in the ERRAT register. When the PERRI bit in the ERREN register is set, the assertion of this bit will assert an interrupt through the OpenPIC interrupt controller.

SERR PCI System Error. This bit is set when the PCI SERR* pin is asserted. It may be cleared by writing it to a 1; writing it to a 0 has no effect. When the SERRM bit in the ERREN register is set, the assertion of this bit will assert MCHK to the master designated by the DFLT bit in the ERRAT register. When the SERRI bit in the ERREN register is set, the assertion of this bit will assert an interrupt through the OpenPIC interrupt controller.

SMA PCI Master Signalled Master Abort. This bit is set when the PCI master signals master abort to terminate a PCI transaction. It may be cleared by writing it to a 1; writing it to a 0 has no effect. When the SMAM bit in the ERREN register is set, the assertion of this bit will assert MCHK to the master designated by the MID field in the ERRAT register. When the SMAI bit in the ERREN register is set, the assertion of this bit will assert an interrupt through the OpenPIC interrupt controller.

RTA PCI Master Received Target Abort. This bit is set when the PCI master receives target abort to terminate a PCI transaction. It may be cleared by writing it to a 1; writing it to a 0 has no effect. When the RTAM bit in the ERREN register is set, the assertion of this bit will assert MCHK to the master designated by the MID field in the ERRAT register. When the RTAI bit in the ERREN register is set, the assertion of this bit will assert an interrupt through the OpenPIC interrupt controller.

PPC Error Address Register

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Address | \$FEFF0028 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | EERAD | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | R | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | \$00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

EERAD PPC Error Address. This register captures the PPC address when the PATO bit is set in the EERST register. It captures the PCI address when the SMA or RTA bits are set in the EERST register. Its contents are not defined when the PDPE, PERR, or SERR bits are set in the EERST register.

PPC Error Attribute Register - MERAT

If the PDPE, PERR, or SERR bits are set in the MERST register, the contents of the MERAT register are zero. If the MATO bit is set, the register is defined by the following figure:

| Address | \$FEFF002C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|---|---|-------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | | | | | | |
| Name | | | | | | | | | | | | | | | | | | | | | MERAT | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | R | | | | | | | | | | R | | | | | | | | | | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | |
| Reset | \$00 | | | | | | | | | | \$00 | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

MIDx PPC Master ID. This field contains the ID of the PPC master which originated the transfer in which the error occurred. The encoding scheme is identical to that used in the GCSR register.

TBST Transfer Burst. This bit is set when the transfer in which the error occurred was a burst transfer.

TSIZx Transfer Size. This field contains the transfer size of the PPC transfer in which the error occurred.

TTx Transfer Type. This field contains the transfer type of the PPC transfer in which the error occurred.

If the SMA or RTA bit are set, the register is defined as follows:

| Address | \$FEFF002C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|---|-------|---|------|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|---|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 3 | 3 | | | |
| Name | | | | | | | | | | | | | | | | | | | MERAT | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | WP | R | MIDI | MIDO | COMM3 | COMM2 | COMM1 | COMM0 | BYTE7 | BYTE6 | BYTE5 | BYTE4 | BYTE3 | BYTE2 | BYTE1 | BYTE0 | |
| Operation | R | | | | | | | | | R | | | | | | | | | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | |
| Reset | \$00 | | | | | | | | | \$00 | | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

WP Write Post Completion. This bit is set when the PCI master detects an error while completing a write post transfer.

MIDx PPC Master ID. This field contains the ID of the PPC master which originated the transfer in which the error occurred. The encoding scheme is identical to that used in the GCSR register

COMMx PCI Command. This field contains the PCI command of the PCI transfer in which the error occurred.

BYTEx PCI Byte Enable. This field contains the PCI byte enables of the PCI transfer in which the error occurred. A set bit designates a selected byte.

PCI Interrupt Acknowledge Register

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Address | \$FEFF0030 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | PIACK | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | R | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | \$00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

PIACK PCI Interrupt Acknowledge. Performing a read from this register will initiate a single PCI Interrupt Acknowledge cycle. Any single byte or combination of bytes may be read from, and the actual byte enable pattern used during the read will be passed on to the PCI bus. Upon completion of the PCI interrupt acknowledge cycle, the Raven will present the resulting vector information obtained from the PCI bus as read data.

PPC Slave Address (0, 1 and 2) Registers

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------------|-------------------------------------------------------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Address | MSADD0 - \$FEFF0040 MSADD1 - \$FEFF0048 MSADD2 - \$FEFF0050 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 |
| Name | MSADDx | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | START | | | | | | | | | | | | | | | | END | | | | | | | | | | | | | | | | | |
| Operation | R/W | | | | | | | | | | | | | | | | R/W | | | | | | | | | | | | | | | | | |
| Reset | \$0000 | | | | | | | | | | | | | | | | \$0000 | | | | | | | | | | | | | | | | | |

To initiate a PCI cycle from the PPC bus, the PPC address must be greater than or equal to the START field and less than or equal to the END field.

START Start Address. This field determines the start address of a particular memory area on the PPC bus which will be used to access PCI bus resources. The value of this field will be compared with the upper 16 bits of the incoming PPC address.

END End Address. This field determines the end address of a particular memory area on the PPC bus which will be used to access PCI bus resources. The value of this field will be compared with the upper 16 bits of the incoming PPC address.

PPC Slave Address (3) Register

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------------|---------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Address | MSADD3 - \$FEFF0058 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0 | 1 | 2 | 3 |
| Name | MSADD3 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | START | | | | | | | | | | | | | | | | END | | | | | | | | | | | | | | | | | |
| Operation | R/W | | | | | | | | | | | | | | | | R/W | | | | | | | | | | | | | | | | | |
| Reset | \$8000 | | | | | | | | | | | | | | | | \$8080 | | | | | | | | | | | | | | | | | |

To initiate a PCI cycle from the PPC bus the PPC address must be greater than or equal to the START field and less than or equal to the END field.

END_End Address. This field determines the end address of a particular memory area on the PPC bus which will be used to access PCI bus resources. The value of this field will be compared with the upper 16 bits of the incoming PPC address.

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|----------------------------------------------------------------------------------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|--------|-----|---|-----|-----|-----|
| Address | MSOFF0/MSATT0 - \$FEFF0044 MSOFF1/MSATT1 - \$FEFF004C MSOFF2/MSATT2 - \$FEFF0054 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | | | | |
| Name | MSOFFx | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | MSATTx | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | R | W | | | | MEM |
| Operation | R/W | | | | | | | | | | | | | | | R | | | | | | | | | | | | | | | R/W | R/W | R | R/W | R/W | R/W |
| Reset | \$0000 | | | | | | | | | | | | | | | \$00 | | | | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 |

MSOFFx PPC Slave Offset. This register contains a 16-bit offset that is added to the upper 16 bits of the PPC address to determine the PCI address used for transfers from the PPC bus to PCI. This offset allows PCI resources to reside at addresses that would not normally be visible from the PPC bus.

REN Read Enable. If set, the corresponding PPC slave is enabled for read transactions.

WEN Write Enable. If set, the corresponding PPC slave is enabled for write transactions.

WPEN Write Post Enable. If set, write posting is enabled for the corresponding PPC slave.

MEM PCI Memory Cycle. If set, the corresponding PPC slave will generate transfers to or from PCI memory space. When clear, the corresponding PPC slave will generate transfers to or from PCI I/O space using the addressing mode defined by the IOM field.

IOM PCI I/O Mode. If set, the corresponding PPC slave will generate PCI I/O cycles using spread addressing as defined in the section on *Generating PCI Memory and I/O Cycles*. When clear, the corresponding PPC slave will generate PCI I/O cycles using contiguous addressing. This field only has meaning when the MEM bit is clear.

PPC Slave Offset/Attribute (3) Registers

| Address | MSOFF3/MSATT3 - \$FEFF005C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|----------------------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|--------|-----|------|----|----|----|-----|----|----|----|----|----|----|----|----|----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | MSOFF3 | | | | | | | | | | | | | | | | MSATT3 | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | REN | WEN | WPEN | | | | IOM | | | | | | | | | |
| Operation | R/W | | | | | | | | | | | | | | | | R | R/W | R | R | R | R | R/W | | | | | | | | | |
| Reset | \$8000 | | | | | | | | | | | | | | | | \$00 | 1 | 1 | 0 | 0 | 0 | 0 | | | | | | | | | |

MSOFF3 PPC Slave Offset. This register contains a 16-bit offset that is added to the upper 16 bits of the PPC address to determine the PCI address used for transfers from the PPC bus to PCI. This offset allows PCI resources to reside at addresses that would not normally be visible from the PPC bus. It is initialized to \$8000 to facilitate a zero-based access to PCI space.

REN Read Enable. If set, the corresponding PPC slave is enabled for read transactions.

WEN Write Enable. If set, the corresponding PPC slave is enabled for write transactions.

WPEN Write Post Enable. If set, write posting is enabled for the corresponding PPC slave.

IOM PCI I/O Mode. If set, the corresponding PPC slave will generate PCI I/O cycles using spread addressing as defined in the section on [Generating PCI Memory and I/O Cycles](#). When clear, the corresponding PPC slave will generate PCI I/O cycles using contiguous addressing.

WDTxCNTL Registers

| Address | WDT1CNTL - \$FEFF0060 WDT2CNTL - \$FEFF0068 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------------------------------------------|---|---|---|---|---|---|---|------|-----|----|------------|----|--------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | WDTxCNTL | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | KEY | | | | | | | | ENAB | ARM | | RES | | RELOAD | | | | | | | | | | | | | | | | | | |
| Operation | W | | | | | | | | R/W | R | R | R/W | | R/W | | | | | | | | | | | | | | | | | | |
| Reset | \$00 | | | | | | | | 1 | 0 | 00 | \$7 or \$8 | | \$FF | | | | | | | | | | | | | | | | | | |

KEY Key. This field is used during the two step arming process of the Control Register. This field is sensitive to the following data patterns:

PATTERN_1 = \$55

PATTERN_2 = \$AA

The Control register will be in the armed state if PATTERN_1 is written to the KEY field. The Control register will be changed if in the armed state and PATTERN_2 is written to the KEY field. An incorrect sequence of patterns will cause the Control register to be in the unarmed state.

A value of all zeros will always be returned within the KEY field during read cycles.

ENAB ENABLE. This field determines whether or not the WDT is enabled. If a one is written to this bit, the timer will be enabled. A zero written to this bit will disable the timer. The ENAB bit may only be modified on the second step of a successful two step arming process.

ARM ARMED. This read-only bit indicates the armed state of the register. If this bit is a zero, the register is unarmed. If this bit is a one, the register is armed for a write.

RES RESOLUTION. This field determines the resolution of the timer. The RES field may only be modified on the second step of a successful two step arming process. The following table shows the different options associated with this bit.

RELOAD RELOAD. This field written with a value that will be used to reload the timer. The RELOAD field may only be modified on the second step of a successful two step arming process.

| RES | Timer Resolution | Approximate Max Time |
|------|------------------|----------------------|
| 0000 | 1 us | 64 msec |
| 0001 | 2 us | 128 msec |
| 0010 | 3 us | 192 msec |
| 0011 | 4 us | 256 msec |
| 0100 | 5 us | 320 msec |
| 0101 | 6 us | 384 msec |
| 0110 | 7 us | 448 msec |

| RES | Timer Resolution | Approximate Max Time |
|------|------------------|----------------------|
| 0111 | 8 us | 512 msec |
| 1000 | 9 us | 576 msec |
| 1001 | 10 us | 640 msec |
| 1010 | 11 us | 704 msec |
| 1011 | 12 us | 768 msec |
| 1100 | 13 us | 832 msec |
| 1101 | 14 us | 896 msec |
| 1110 | 15 us | 960 msec |
| 1111 | 16 us | 1024 msec |

Note Each Watchdog timer output has an additional 4.5 second delay provided by logic external to Raven. This delay is fixed and not programmable. The delayed WDT1 output is routed to MPIC external interrupt IRQ4. The delayed WDT2 output is logically ORed onboard to provide a hard reset.

WDTxSTAT Registers

| Address | WDT1STAT - \$FEFF0064 WDT2STAT - \$FEFF006C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------------------------------------------|---|---|---|---|---|---|---|------|---|----|----|----|----|----|----|----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| | | | | | | | | | | | | | | | | | WDTxSTAT | | | | | | | | | | | | | | | |
| Name | | | | | | | | | | | | | | | | | COUNT | | | | | | | | | | | | | | | |
| Operation | R | | | | | | | | R | | | | | | | | R | | | | | | | | | | | | | | | |
| Reset | \$00 | | | | | | | | \$00 | | | | | | | | \$FF | | | | | | | | | | | | | | | |

COUNT Count. This read-only field reflects the instantaneous counter value of the WDT.

General Purpose Registers

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|--------------------------------------------------------------------------------------------------------------------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Address | GPREG0 (Upper) - \$FEFF0070 GPREG0 (Lower) - \$FEFF0074 GPREG1 (Upper) - \$FEFF0078 GPREG1 (Lower) - \$FEFF007C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | GPREGx | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | R/W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | \$00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

These general purpose read/write registers are provided for inter-process message passing or general purpose storage. They do not control any hardware.

PCI Registers

The PCI Configuration Registers are compliant with the configuration register set described in the *PCI Local Bus Specification, Revision 2.1*. The CONFIG_ADDRESS and CONFIG_DATA registers described in this section are accessed within PCI I/O space.

All write operations to reserved registers will be treated as no-ops. That is, the access will be completed normally on the bus and the data will be discarded. Read accesses to reserved or unimplemented registers will be completed normally and a data value of 0 returned.

The Raven PCI Configuration Register map is shown in [Table 2-7](#). The Raven PCI I/O Register map is shown in [Table 2-8](#).

| 3 1 | 3 0 | 2 9 | 2 8 | 2 7 | 2 6 | 2 5 | 2 4 | 2 3 | 2 2 | 2 1 | 0 | 1 9 | 1 8 | 1 7 | 1 6 | 1 5 | 1 4 | 1 3 | 1 2 | 1 1 | 1 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | <--- Bit |
|---------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---|---|---|--------|---|---|---|---|-------------|---|----------|
| DEVID | | | | | | | | | | | | | | | VENID | | | | | | | | | | | | | | | \$00 | | |
| PSTAT | | | | | | | | | | | | | | | PCOMM | | | | | | | | | | | | | | | \$04 | | |
| CLASS | | | | | | | | | | | | | | | | | | | | REVID | | | | | | | | | | \$08 | | |
| | | | | | | | | | | HEADER | | | | | | | | | | | | | | | | | | | | \$0C | | |
| IOBASE | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$10 | | |
| MEMBASE | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$14 | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$18 - \$7F | | |
| PSADD0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$80 | | |
| PSOFF0 | | | | | | | | | | | | | | | | | | | | | | | | | PSATT0 | | | | | \$84 | | |
| PSADD1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$88 | | |
| PSOFF1 | | | | | | | | | | | | | | | | | | | | | | | | | PSATT1 | | | | | \$8C | | |
| PSADD2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$90 | | |
| PSOFF2 | | | | | | | | | | | | | | | | | | | | | | | | | PSATT2 | | | | | \$94 | | |
| PSADD3 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$98 | | |
| PSOFF3 | | | | | | | | | | | | | | | | | | | | | | | | | PSATT3 | | | | | \$9C | | |

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|----------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|-------|---|---|---|----------|
| 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | <--- Bit |
| 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| CONFIG_ADDRESS | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$CF8 | | | | |
| CONFIG_DATA | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$CFC | | | | |

Vendor ID/Device ID Registers

| Offset | \$00 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|--------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | DEVID | | | | | | | | | | | | | | | | VENID | | | | | | | | | | | | | | | |
| Operation | R | | | | | | | | | | | | | | | | R | | | | | | | | | | | | | | | |
| Reset | \$4801 | | | | | | | | | | | | | | | | \$1057 | | | | | | | | | | | | | | | |

VENID Vendor ID. This register identifies the manufacturer of the device. This identifier is allocated by the PCI SIG to ensure uniqueness. \$1057 has been assigned to Motorola. This register is duplicated in the PPC Registers.

DEVID Device ID. This register identifies the particular device. The Raven will always return \$4801. This register is duplicated in the PPC Registers.

PCI Command/Status Registers

| Offset | \$04 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|-------|-------|-------|-------|-------|---------|---------|------|------|----|----|----|----|----|----|----|-------|----|----|----|----|----|-----|------|------|---|---|---|---|---|------|-------|------|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| Name | PSTAT | | | | | | | | | | | | | | | | PCOMM | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | RCVPE | SIGSE | RCVMA | RCVTA | SIGTA | SELTIM1 | SELTIM0 | DPAR | FAST | | | | | | | | | | | | | | | SERR | PERR | | | | | | MSTR | MEMSP | IOSP |
| Operation | R/C | R/C | R/C | R/C | R/C | R | R | R/C | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R/W | R | R/W | R | R | R | R | R | R/W | R/W | R/W |
| Reset | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | |

IOSP I/O Space Enable. If set, the Raven will respond to PCI I/O accesses when appropriate. If cleared, the Raven will not respond to PCI I/O space accesses.

MEMSP Memory Space Enable. If set, the Raven will respond to PCI memory space accesses when appropriate. If cleared, the Raven will not respond to PCI memory space accesses.

MSTR Bus Master Enable. If set, the Raven may act as a master on PCI. If cleared, the Raven may not act as a PCI master.

PERR Parity Error Response. If set, the Raven will check parity on all PCI transfers. If cleared, the Raven will ignore any parity errors that it detects and continue normal operation.

SERR System Error Enable. This bit enables the SERR* output pin. If clear, the Raven will never drive SERR*. If set, the Raven will drive SERR* active when a system error is detected.

FAST Fast Back-to-Back Capable. This bit indicates that the Raven is capable of accepting fast back-to-back transactions with different targets.

DPAR Data Parity Detected. This bit is set when three conditions are met: 1) the Raven asserted PERR* itself or observed PERR* asserted; 2) the Raven was the PCI master for the transfer in which the error occurred; 3) the PERR bit in the PCI Command Register is set. This bit is cleared by writing it to 1; writing a 0 has no effect.

SELTIM DEVSEL Timing. This field indicates that the Raven will always assert DEVSEL* as a 'medium' responder.

SIGTA Signalled Target Abort. This bit is set by the PCI slave whenever it terminates a transaction with a target-abort. It is cleared by writing it to 1; writing a 0 has no effect.

RCVTA Received Target Abort. This bit is set by the PCI master whenever its transaction is terminated by a target-abort. It is cleared by writing it to 1; writing a 0 has no effect.

RCVMA Received Master Abort. This bit is set by the PCI master whenever its transaction (except for Special Cycles) is terminated by a master-abort. It is cleared by writing it to 1; writing a 0 has no effect.

SIGSE Signaled System Error. This bit is set whenever the Raven asserts SERR*. It is cleared by writing it to 1; writing a 0 has no effect.

RCVPE Detected Parity Error. This bit is set whenever the Raven detects a parity error, even if parity error checking is disabled (see bit PERR in the *PCI Command Register*). It is cleared by writing it to 1; writing a 0 has no effect.

Revision ID/Class Code Registers

| Offset | \$08 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|----------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|-------|---|---|---|---|---|---|---|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | CLASS | | | | | | | | | | | | | | | | | | | | | | | | REVID | | | | | | | |
| Operation | R | | | | | | | | | | | | | | | | | | | | | | | | R | | | | | | | |
| Reset | \$060000 | | | | | | | | | | | | | | | | | | | | | | | | \$03 | | | | | | | |

REVID Revision ID. This register identifies the Raven revision level. This register is duplicated in the PPC Registers.

CLASS Class Code. This register identifies Raven as the following:

| | | |
|--------------------|------|-------------------|
| Base Class Code | \$06 | PCI Bridge Device |
| Subclass Code | \$00 | PCI Host Bridge |
| Program Class Code | \$00 | Not Used |

Header Type Register

| Offset | \$0C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------|---|---|---|---|---|---|---|--------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | | | | | | | | | HEADER | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | R | | | | | | | | R | | | | | | | | R | | | | | | | | R | | | | | | | |
| Reset | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | |

HEADER Header Type. This register identifies Raven as the following:

| | | |
|-------------|------|--------------------------------------|
| Header Type | \$00 | Single Function Configuration Header |
|-------------|------|--------------------------------------|

[illegible]

| Offset | \$14 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|---------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|--------|----|----|----|----|----|----|---|---|---|---|---|---------------|------------|------------|-------------|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | MEMBASE | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | MEMBA | | | | | | | | | | | | | | | | | | | | | | | | | | | IO/MEM PRE | MTYP0 R | MTYP1 R | IO/MEM R | |
| Operation | R/W | | | | | | | | | | | | | | | R | | | | | | | | | | | | R | R | R | R | |
| Reset | \$0000 | | | | | | | | | | | | | | | \$0000 | | | | | | | | | | | | 0 | 0 | 0 | 0 | |

This register controls the mapping of the MPIC control registers in PCI memory space.

IO/MEM I/O Space Indicator. This bit is hard-wired to a logic zero to indicate PCI memory space.

MTYPx Memory Type. These bits are hard-wired to zero to indicate that the MPIC registers can be located anywhere in the 32-bit address space.

PRE Prefetch. This bit is hard-wired to zero to indicate that the MPIC registers are not prefetchable.

MEMBA Memory Base Address. These bits define the memory space base address of the MPIC control registers. The MBASE decoder is disabled when the MBASE value is zero.

PCI Slave Address (0, 1, 2 and 3) Registers

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------------|------------------------------------------------------------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|--------|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Offset | PSADD0 - \$80 PSADD1 - \$88 PSADD2 - \$90 PSADD3 - \$98 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | PSADDx | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | START | | | | | | | | | | | | | | | | END | | | | | | | | | | | | | | | |
| Operation | R/W | | | | | | | | | | | | | | | | R/W | | | | | | | | | | | | | | | |
| Reset | \$0000 | | | | | | | | | | | | | | | | \$0000 | | | | | | | | | | | | | | | |

To initiate an PPC cycle from the PCI bus the PCI address must be greater than or equal to the START field and less than or equal to the END field.

START Start Address. This field determines the start address of a particular memory area on the PCI bus which will be used to access PPC bus resources. The value of this field will be compared with the upper 16 bits of the incoming PCI address.

END End Address. This field determines the end address of a particular memory area on the PCI bus which will be used to access PPC bus resources. The value of this field will be compared with the upper 16 bits of the incoming PCI address.

PCI Slave Attribute/ Offset (0, 1, 2 and 3) Registers

| Offset | PSATT0/PCISOFF0 - \$84 PSATT1/PCISOFF1 - \$8C PSATT2/PCISOFF2 - \$94 PSATT3/PCISOFF3 - \$9C | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------------------------------------------------------------------------------------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|--------|-----|------|------|------|-----|-----|-----|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| Name | PCISOFFx | | | | | | | | | | | | | | | | | | | | | | | | PSATTx | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | REN | WEN | WPEN | RAEN | TDIS | GBL | INV | |
| Operation | R/W | | | | | | | | | | | | | | | | R | | | | | | | | R/w | R/w | R/w | R/w | Rw | R | R/w | R/w |
| Reset | \$0000 | | | | | | | | | | | | | | | | \$00 | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

INV Invalidate Enable. If set, the PPC master will issue a transfer type code which specifies the current transaction should cause an invalidate for each PC transaction originated by the corresponding PCI slave. The transfer type codes generated are shown in [Table 2-3](#).

GBL Global Enable. If set, the PPC master will assert the GBL* pin for each PPC transaction originated by the corresponding PCI slave.

TDIS Threshold Disable. If set, the PPC master FIFO threshold mechanism is disabled. See the section titled *PPC Master* for more information on the relationship between TDIS and RAEN.

RAEN Read Ahead Enable. If set, read ahead is enabled for the corresponding PCI slave.

WPEN Write Post Enable. If set, write posting is enabled for the corresponding PCI slave.

WEN Write Enable. If set, the corresponding PCI slave is enabled for write transactions.

REN Read Enable. If set, the corresponding PCI slave is enabled for read transactions.

PCIOFFx PCI Slave Offset. This register contains a 16-bit offset that is added to the upper 16 bits of the PCI address to determine the PPC address used for transfers from PCI to the PPC bus. This offset allows PPC resources to reside at addresses that would not normally be visible from PCI.

CONFIG_ADDRESS

The routing of the PPC data bus to and from the CONFIG_ADDRESS register depends on the endian bit setting. Refer to the sections on [Endian Conversion](#), and the LEND bit in the GCSR. The following register diagrams have two additional rows of information. These rows indicate the source bit positions on the PPC data bus when data is written to this register. A read from the CONFIG_ADDRESS register will return contents to these locations. One row defines the source for big endian operation and the second is for little endian operation. LEND is the little endian control bit in the GCSR.

The MSADD3, MSOFF3, and MSATT3 are initialized at reset so software can access PCI configuration address and data space without changing Raven registers.

PCI I/O CONFIG_ADDRESS Register

| Offset | \$CF8 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------|----------------|---|---|---|---|---|---|-----|---|---|---|-----|---|---|---|-----|---|---|---|-----|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | CONFIG_ADDRESS | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | EN | | | | | | | BUS | | | | DEV | | | | FUN | | | | REG | | | | | | | | | | | | |

| Offset | \$CF8 | | | | | | | |
|-----------|-------|------------|------------|------------|-----|------------|---|---|
| Operation | R/W | R | R/W | R/W | R/W | R/W | R | R |
| Reset | 1 | \$00 | \$00 | \$00 | \$0 | \$00 | 0 | 0 |
| LEND = 0 | | PPC[39:32] | PPC[47:40] | PPC[55:48] | | PPC[63:56] | | |
| LEND = 1 | | PPC[31:24] | PPC[23:16] | PPC[15:8] | | PPC[7:0] | | |

REG Register Number. For PCI Configuration cycles, bits 7 through 2 identify the target double word within the target function's configuration space. Bits 1 and 0 must always be zero for a type 0 configuration cycle. These bits are copied to the PCI AD bus during the address phase on a Configuration cycle. This field must be all zeros for Special cycles.

FUN Function Number. For PCI Configuration cycles, bits 10 through 8 identify the function number within the target physical PCI device. These bits are copied to the PCI AD bus during the address phase on a Configuration cycle. This field must be all ones for Special cycles.

DEV Device Number. For PCI Configuration cycles, bits 15 through 11 identify the target physical PCI device number. Raven does a decode of the Device Number field to assert the appropriate IDSEL line. Values of \$01 through \$0a and \$1f are illegal entries for the device number. The Raven will drive all 0's in bit position AD11 through AD31 if a illegal device id is initialized into the configuration address register. A value of \$0B sets PCI AD bit 11 (IDSEL 11) during the address phase of a Configuration cycle. A value of \$0C sets AD bit 12. As the device number increments the AD bit increments until a the value of \$1E sets AD bit 30. A value of \$00 in device number field will select AD bit 31. The device number field must be all ones for Special cycles.

BUS Bus Number. For PCI Configuration cycles or Special cycles, bits 23 through 16 identify the bus number. Raven is always connected to PCI bus number zero. Bits 23 through 16 must be zero for a Configuration cycle or Special cycle. Raven will execute a type 1 translation cycle if the bus number is set to a value not equal to zero.

EN Enable. Bit 31 must be set to a one, enabling the translation of a subsequent host bus I/O access to the CONFIG_DATA register into a configuration access on the PCI bus. If bit 31 is zero and the processor initiates an I/O read from or write to the CONFIG_DATA register, the transaction is passed through to the PCI bus as a PCI I/O transaction.

PCI I/O CONFIG_DATA Register

| Offset | \$CFC | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|-------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | CONFIG_DATA | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | R/W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | \$00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

If the CONFIG_ADDRESS register is initialized for a PCI Configuration cycle, an access to the CONFIG_DATA register will write to or read from a PCI configuration space location. If the CONFIG_ADDRESS register is initialized for a PCI Special cycle, a word write to the CONFIG_DATA register will generate a special cycle on the PCI bus.

Raven Interrupt Controller Implementation

Introduction

The Raven Interrupt Controller (Raven MPIC) Features

- ❑ MPIC programming model
- ❑ Support for two processors
- ❑ Support for 16 external interrupts
- ❑ Support for 15 programmable Interrupt & Processor Task priority levels

- ❑ Support for the connection of an external 8259 for ISA/AT compatibility
- ❑ Distributed interrupt delivery for external I/O interrupts
- ❑ Direct/Multicast interrupt delivery for Interprocessor and timer interrupts
- ❑ Four Interprocessor Interrupt sources
- ❑ Four timers
- ❑ Processor initialization control

Architecture

The Raven PCI Slave implements two address decoders for placing the Raven MPIC registers in PCI IO or PCI Memory space. Access to these registers require MPC and PCI bus mastership. These accesses include interrupt and timer initialization and interrupt vector reads.

The Raven MPIC receives interrupt inputs from 16 external sources, four interprocessor sources, four timer sources, and one Raven internal error detection source. The externally sourced interrupts 1 through 15 have two modes of activation: low level or active high positive edge. External interrupt 0 can be either level or edge activated with either polarity. The interprocessor and timers interrupts are event activated.

CSR's Readability

Unless explicitly specified, all registers are readable and return the last value written. The exceptions are the IPI dispatch registers and the EOI registers which return zeros on reads, the interrupt source ACT bit which returns current interrupt source status, the interrupt acknowledge register which returns the vector of the highest priority interrupt which is currently pending, and reserved bits which returns zeros. The interrupt acknowledge register is also the only register which exhibits any read side-effects.

Interrupt Source Priority

Each interrupt source is assigned a priority value in the range from 0 to 15, where 15 is the highest. In order for delivery of an interrupt to take place, the priority of the source must be greater than that of the destination processor. Therefore, setting a source priority to zero inhibits that interrupt.

Processor's Current Task Priority

Each processor has a task priority register which is set by system software to indicate the relative importance of the task running on that processor. The processor will not receive interrupts with a priority level equal to or lower than its current task priority. Therefore, setting the current task priority to 15 prohibits the delivery of all interrupts to the associated processor.

Nesting of Interrupt Events

A processor is guaranteed never to have an in-service interrupt preempted by an equal or lower priority source. An interrupt is considered to be in service from the time its vector is returned during an interrupt acknowledge cycle until an EOI is received for that interrupt. The EOI cycle indicates the end of processing for the highest priority in-service interrupt.

Spurious Vector Generation

Under certain circumstances the Raven MPIC will not have a valid vector to return to the processor during an interrupt acknowledge cycle. In these cases, the spurious vector from the spurious vector register will be returned. The following cases would cause a spurious vector fetch.

- ❑ INT is asserted in response to an externally sourced interrupt which is activated with level sensitive logic and the asserted level is negated before the interrupt is acknowledged.
- ❑ INT is asserted for an interrupt source which is masked using the mask bit in the Vector-Priority register before the interrupt is acknowledged.

Interprocessor Interrupts (IPI)

Processor 0 and 1 can generate interrupts which are targeted for the other processor or both processors. There are four Interprocessor Interrupts (IPI) channels. The interrupts are initiated by writing a bit in the IPI dispatch registers. If subsequent IPIs are initiated before the first is acknowledged, only one IPI will be generated. The IPI channels deliver interrupts in the Direct Mode and can be directed to more than one processor.

8259 Compatibility

The Raven MPIC provides a mechanism to support PC-AT compatible chip sets using the 8259 interrupt controller architecture. After power-on reset, the Raven MPIC defaults to 8259 pass-through mode. In this mode, interrupts from external source number 0 (the interrupt signal from the 8259 is connected to this external interrupt source on the Raven MPIC) are passed directly to processor 0. If the pass-through mode is disabled, the 8259 interrupts are delivered using the priority and distribution mechanisms of the Raven MPIC.

The Raven MPIC does not interact with the vector fetch from the 8259 interrupt controller.

Raven-Detected Errors

Raven-detected errors are grouped together and sent to the interrupt logic as a singular interrupt source. The interrupt delivery mode for this interrupt is distributed.

For system implementations where the Raven MPIC controller is not used, the Raven-Detected Error condition will be made available by a signal which is external to the Raven ASIC. Presumably this signal would be connected to an externally sourced interrupt input of a MPIC controller in a different device. Since the MPIC specification defines external I/O interrupts to operate in the distributed mode, the delivery mode of this error interrupt should be consistent.

Timers

There is a divide by eight pre-scaler which is synchronized to the Raven clock (MPC processor clock). The output of the prescaler enables the decrement of the four timers. The timers may be used for system timing or to generate periodic interrupts. Each timer has four registers which are used for configuration and control. They are identified as follows:

- ❑ Current Count Register
- ❑ Base Count Register
- ❑ Vector-Priority Register
- ❑ Destination Register

Interrupt Delivery Modes

The direct and distributed interrupt delivery modes are supported. Note that the direct deliver mode has sub modes of multicast or non-multicast. The Interprocessor Interrupts (IPIs) and Timer interrupts operate in the direct delivery mode. The externally sourced or I/O interrupts operate in the distributed mode.

In the direct delivery mode, the interrupt is directed to one or both processors. If it is directed to two processors (that is, multicast), it will be delivered to two processors. The interrupt is delivered to the processor when the priority of the interrupt is greater than the priority contained in the task register for that processor, and when the priority of the interrupt is greater than any interrupt which is in-service for that processor. An interrupt is considered to be in service from the time its vector is returned during an interrupt acknowledge cycle until an EOI is received for that interrupt. The EOI cycle indicates the end of processing for the highest priority in- service interrupt.

In the distributed delivery mode, the interrupt is pointed to one or more processors but it will be delivered to only one processor. Therefore, for externally sourced or I/O interrupts, multicast delivery is not supported. The interrupt is delivered to a processor when the priority of the interrupt is greater than the priority contained in the task register for that processor, and when the priority of the interrupt is greater than any

interrupt which is in-service for that processor, and when the priority of that interrupt is the highest of all interrupts pending for that processor, and when that interrupt is not in-service for the other processor. If both destination bits are set for each processor, the interrupt will be delivered to the processor that has a lower task register priority.

Note Because a deadlock condition can occur when the task register priorities for each processor are the same and both processors are targeted for interrupt delivery, the interrupt will be delivered to processor 0 or processor 1 as determined by the TIE mode.

Block Diagram Description

The description of the block diagram focuses on the theory of operation for the interrupt delivery logic. If the preceding section is a satisfactory description of the interrupt delivery modes and the reader is not interested the logic implementation, this section can be skipped.

Program Visible Registers

These are the registers which software can access. They are described in detail in the Register section.

Interrupt Pending Register (IPR)

The interrupt signals to Raven MPIC are qualified and synchronized to the clock by the IPR. If the interrupt source is internal to the Raven ASIC or external with their Sense bit = 0 (edge sensitive), a bit is set in the IPR. That bit is cleared when the interrupt associated with that bit is acknowledge. If the interrupt source is external and level activated, the output from the IPR is not negated until the level into the IPR is negated.

Externally sourced interrupts are qualified based upon their Sense and/or Pol bits in the Vector-Priority register. IPI and Timer Interrupts are generated internally to the Raven ASIC and are qualified by their Destination bit. Since the internally generated interrupts use direct delivery mode with multicast capability, there are two bits in the IPR, one for each processor, associated with each IPI and Timer interrupt source.

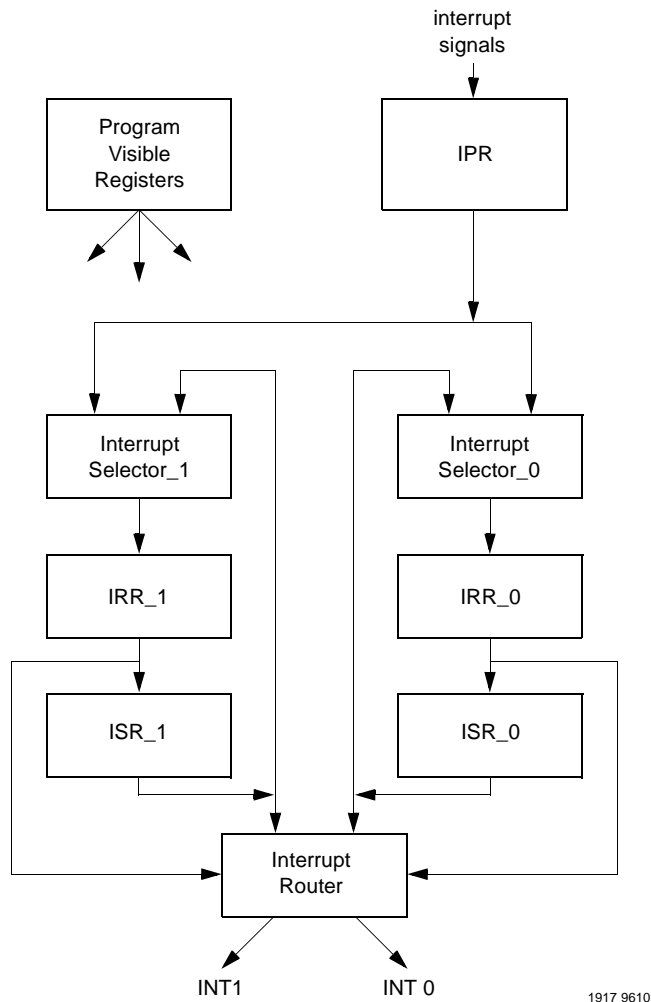


Figure 2-4. Raven MPIC Block Diagram

The MASK bits from the Vector-Priority registers are used to qualify the output of the IPR. Therefore, if an interrupt condition is detected when the MASK bit is set, that interrupt will be requested when the MASK bit is lowered.

Interrupt Selector (IS)

There is a Interrupt Selector (IS) for each processor. The IS receives interrupt requests from the IPR. If the interrupt request are from an external source, they are qualified by the destination bit for that interrupt and processor. If they are from an internal source, they have been qualified. The output of the IS will be the highest priority interrupt that has been qualified. This output is the priority of the selected interrupt and its source identification. The IS will resolve an interrupt request in two Raven clock ticks.

The IS also receives a second set of inputs from the ISR. During the End Of Interrupt cycle, these inputs are used to select which bits are to be cleared in the ISR.

Interrupt Request Register (IRR)

There is a Interrupt Request Register (IRR) for each processor. The IRR always passes the output of the IS except during Interrupt Acknowledge cycles. This guarantees that the vector which is read from the Interrupt Acknowledge Register is not changing due to the arrival of a higher priority interrupt. The IRR also serves as a pipeline register for the two tick propagation time through the IS.

In-Service Register (ISR)

There is a In-Service Register (ISR) for each processor. The contents of the ISR is the priority and source of all interrupts which are in-service. The ISR receives a bit-set command during Interrupt Acknowledge cycles and a bit-clear command during End Of Interrupt cycles.

The ISR is implemented as a 40 bit register with individual bit set and clear functions. Fifteen bits are used to store the priority level of each interrupt which is in-service. Twenty-five bits are used to store the source identification of each interrupt which is in service. Therefore, there is one bit for each possible interrupt priority and one bit for each possible interrupt source.

Interrupt Router

The Interrupt Router (IR) monitors the outputs from the ISRs, Current Task Priority Registers, Destination Registers, and the IRRs to determine when to assert a processor's INT pin.

When considering the following rule sets, it is important to remember that there are two types of inputs to the Interrupt Selectors. If the interrupt is a distributed class interrupt, there is a single bit in the IPR associated with this interrupt and it is delivered to both Interrupt Selectors. This IPR bit is qualified by the destination register contents for that interrupt before the Interrupt Selector compares its priority to the priority of all other requesting interrupts for that processor. If the interrupt is programmed to be edge sensitive, the IPR bit is cleared when the vector for that interrupt is returned when the Interrupt Acknowledge register is examined. On the other hand, if the interrupt is a direct/multicast class interrupt, there are two bits in the IPR associated with this interrupt. One bit for each processor. Then one of these bits are delivered to each Interrupt Selector. Since this interrupt source can be multicast, each of these IPR bits must be cleared separately when the vector is returned for that interrupt to a particular processor.

If one of the following sets of conditions are true, the interrupt pin for processor 0 is driven active.

❑ Set 1

- The source ID in IRR_0 is from an external source.
- The destination bit for processor 1 is a 0 for this interrupt.
- The priority from IRR_0 is greater than the highest priority in ISR_0.
- The priority from IRR_0 is greater than the contents of task register_0.

❑ Set 2

- The source ID in IRR_0 is from an external source.
- The destination bit for processor 1 is a 1 for this interrupt.
- The source ID in IRR_0 is not present in ISR_1.

- The priority from IRR_0 is greater than the highest priority in ISR_0.
 - The priority from IRR_0 is greater than the Task Register_0 contents.
 - The contents of Task Register_0 is less than the contents of Task Register_1.
- ❑ Set 3
- The source ID in IRR_0 is from an internal source.
 - The priority from IRR_0 is greater than the highest priority in ISR_0.
 - The priority from IRR_0 is greater than the Task Register_0 contents.

There is a possibility for a priority tie between the two processors when resolving external interrupts. In that case the interrupt will be delivered to processor 0 or processor 1 as determined by the TIE mode bit. This case is not defined in the above rule set.

MPIC Registers

The following conventions are used in the Raven register charts:

- ❑ R Read Only field.
- ❑ R/W Read/Write field.
- ❑ S Writing a one (1) to this field sets this field.
- ❑ C Writing a one (1) to this field clears this field.

Raven MPIC Registers

The Raven MPIC register map is shown in the following table. The Off field is the address offset from the base address of the Raven MPIC registers in the MPC-IO or MPC-MEMORY space. Note that this map does not depict linear addressing. The Raven PCI-SLAVE has two decoders for generating the Raven MPIC select. These decoders will

generate a select and acknowledge all accesses which are in a reserved 256Kbyte range. If the index into that 256K block does not decode a valid Raven MPIC register address, the logic will return \$00000000.

The registers are 8-, 16-, or 32-bits accessible.

Table 2-9. Raven MPIC Register Map

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-------------------------------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---|---|---|---|---|-------------|---|---------|---|---|-----|
| 3 1 | 3 0 | 2 9 | 2 8 | 2 7 | 2 6 | 2 5 | 2 4 | 2 3 | 2 2 | 2 1 | 2 0 | 1 9 | 1 8 | 1 7 | 1 6 | 1 5 | 1 4 | 1 3 | 1 2 | 1 1 | 1 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Off |
| FEATURE REPORTING REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01000 | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GLOBAL CONFIGURATION REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01020 | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| MPIC VENDOR IDENTIFICATION REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01080 | | | |
| PROCESSOR INIT REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01090 | | | |
| IPI0 VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$010a0 | | | |
| IPI1 VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$010b0 | | | |
| IPI2 VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$010c0 | | | |
| IPI3 VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$010d0 | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | SP REGISTER | | \$010e0 | | | |
| TIMER FREQUENCY REPORTING REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$010f0 | | | |
| TIMER 0 CURRENT COUNT REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01100 | | | |
| TIMER 0 BASE COUNT REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01110 | | | |
| TIMER 0 VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01120 | | | |
| TIMER 0 DESTINATION REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01130 | | | |
| TIMER 1 CURRENT COUNT REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01140 | | | |
| TIMER 1 BASE COUNT REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01150 | | | |
| TIMER 1VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01160 | | | |
| TIMER 1DESTINATION REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01170 | | | |
| TIMER 2 CURRENT COUNT REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01180 | | | |
| TIMER 2 BASE COUNT REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$01190 | | | |
| TIMER 2 VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$011a0 | | | |
| TIMER 2 DESTINATION REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$011b0 | | | |

Table 2-9. Raven MPIC Register Map (Continued)

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------------------------------------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---------------------|---------|---|-----|
| 3 1 | 3 0 | 2 9 | 2 8 | 2 7 | 2 6 | 2 5 | 2 4 | 2 3 | 2 2 | 2 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | Off |
| INT. SRC. 12 VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$10180 | | | |
| INT. SRC. 12 DESTINATION REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$10190 | | | |
| INT. SRC. 13 VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$101a0 | | | |
| INT. SRC. 13 DESTINATION REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$101b0 | | | |
| INT. SRC. 14 VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$101c0 | | | |
| INT. SRC. 14 DESTINATION REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$101d0 | | | |
| INT. SRC. 15 VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$101e0 | | | |
| INT. SRC. 15 DESTINATION REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$101f0 | | | |
| RAVEN DETECTED ERRORS VECTOR-PRIORITY REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$10200 | | | |
| RAVEN DETECTED ERRORS DESTINATION REGISTER | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$10210 | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| IPI 0 DISPATCH REGISTER PROC. 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$20040 | | | |
| IPI 1 DISPATCH REGISTER PROC. 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$20050 | | | |
| IPI 2 DISPATCH REGISTER PROC. 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$20060 | | | |
| IPI 3 DISPATCH REGISTER PROC. 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$20070 | | | |
| CURRENT TASK PRIORITY REGISTER PROC. 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$20080 | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | IACK REGISTER P0 | \$200a0 | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | EOI REGISTER P0 | \$200b0 | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| IPI 0 DISPATCH REGISTER PROC. 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$21040 | | | |
| IPI 1 DISPATCH REGISTER PROC. 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$21050 | | | |
| IPI 2 DISPATCH REGISTER PROC. 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$21060 | | | |
| IPI 3 DISPATCH REGISTER PROC. 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | \$21070 | | | |

Global Configuration Register

| Offset | \$01020 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|----------------------|---|-----|-----|---|------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|--|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | |
| | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | |
| Name | GLOBAL CONFIGURATION | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | RESET | | | T | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | C | R | R/W | R/W | | R | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | 0 | 0 | 0 | \$0 | | \$00 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

R RESET CONTROLLER. Writing a one to this bit forces the controller logic to be reset. This bit is cleared automatically when the reset sequence is complete. While this bit is set, the values of all other register are undefined.

M CASCADE MODE. Allows cascading of an external 8259 pair connected to the first interrupt source input pin (0). In the pass through mode, interrupt source 0 is passed directly through to the processor 0 INT pin. Raven MPIC is essentially disabled. In the mixed mode, 8259 interrupts are delivered using the priority and distribution mechanism of Raven MPIC. The Vector/Priority and Destination registers for interrupt source 0 are used to control the delivery mode for all 8259 generated interrupt sources.

| M | MODE |
|---|--------------|
| 0 | Pass Through |
| 1 | Mixed |

T Tie Mode. Writing a one to this register bit will cause a tie in external interrupt processing to, swap back and forth between processor 0 and 1. The first tie in external interrupt processing always goes to Processor 0 after a reset. When this register bit is set to 0, a tie in external interrupt processing will always go to processor 0 (Mode used on Version \$02 of MPIC).

| T | MODE |
|---|-----------------------------|
| 0 | Processor 0 always selected |
| 1 | Swap between Processor's |

Vendor Identification Register

| Offset | \$01080 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|-----------------------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|---|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | VENDOR IDENTIFICATION | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | STP | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | R | | | | | | | | R | | | | | | | | R | | | | | | | | R | | | | | | | | |
| Reset | \$00 | | | | | | | | \$02 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | |

There are two fields in the Vendor Identification Register which are not defined for the Raven MPIC implementation but are defined in the MPIC specification. They are the vendor identification and device ID fields.

STP STEPPING.The stepping or silicon revision number is initially 0.

Processor Init Register

| Offset | \$01090 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|----------------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|-----|-----|--|--|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | |
| | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | |
| Name | PROCESSOR INIT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | P1 | P0 | | |
| Operation | R | | | | | | | | R | | | | | | | | R | | | | | | | | R | | | | | | | | R/W | R/W | | |
| Reset | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | 0 | 0 | | |

P1 PROCESSOR 1. Writing a 1 to P1 will assert the Soft Reset input of processor 1. Writing a 0 to it will negate the SRESET signal.

P0 PROCESSOR 0. Writing a 1 to P0 will assert the Soft Reset input of processor 0. Writing a 0 to it will negate the SRESET signal.

The Soft Reset input to the 604 is negative edge-sensitive.

IPI Vector/Priority Registers

| Offset | IPI 0 - \$010A0 IPI 1 - \$010B0 IPI 2 - \$010C0 IPI 3 - \$010D0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|--------------------------------------------------------------------------|----|-----|----|-------|----|----|----|----|----|----|----|-------|----|----|----|------|----|----|----|----|----|---|---|--------|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | IPI VECTOR/PRIORITY | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | MASK | | ACT | | | | | | | | | | PRIOR | | | | | | | | | | | | VECTOR | | | | | | | |
| Operation | R/W | | R | | R | | | | | | | | R/W | | | | R | | | | | | | | R/W | | | | | | | |
| Reset | 1 | | 0 | | \$000 | | | | | | | | \$0 | | | | \$00 | | | | | | | | \$00 | | | | | | | |

MASK MASK. Setting this bit disables any further interrupts from this source. If the mask bit is cleared while the bit associated with this interrupt is set in the IPR, the interrupt request will be generated.

ACT ACTIVITY. The activity bit indicates that an interrupt has been requested or that it is in-service. The ACT bit is set to a one when its associated bit in the Interrupt Pending Register or In-Service Register is set.

PRIOR PRIORITY. Interrupt priority 0 is the lowest and 15 is the highest. Note that a priority level of 0 will not enable interrupts.

VECTOR VECTOR. This vector is returned when the Interrupt Acknowledge register is examined during a request for the interrupt associated with this vector.

Spurious Vector Register

| Offset | \$010E0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|---------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|--------|---|---|---|---|---|---|---|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | | | | | | | | | | | | | | | | | | | | | | | | | VECTOR | | | | | | | |
| Operation | R | | | | | | | | R | | | | | | | | R | | | | | | | | R/W | | | | | | | |
| Reset | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | \$FF | | | | | | | |

VECTOR VECTOR. This vector is returned when the Interrupt Acknowledge register is read during a spurious vector fetch.

Timer Frequency Register

| Offset | \$010F0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|-----------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | TIMER FREQUENCY | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | R/W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | \$00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

This register is used to report the frequency (in Hz) of the clock source for the global timers. Following reset, this register contains zero. system initialization code must initialize this register to one-eighth the MPIC clock frequency. For the Raven implementation of MPIC, a typical value would be \$7de290 which is 66/8 MHz or 8.25 MHz.

Timer Current Count Registers

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|------------------|----------------------------------------------------------------------------------|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Offset | Timer 0 - \$01100 Timer 1 - \$01140 Timer 2 - \$01180 Timer 3 - \$011C0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | TIMER CURRENT COUNT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | T | CC | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | R | R | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | 0 | \$00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

T TOGGLE. This bit toggles when ever the current count decrements to zero. This bit gets cleared when a value is written to the corresponding base count register and the CI bit in the corresponding base count register transitions from a 1 to a 0.

CC CURRENT COUNT. The current count field decrements while the Count Inhibit bit is the Base Count Register is zero. When the timer counts down to zero, the Current Count register is reloaded from the Base Count register and the timer's interrupt becomes pending in the MPIC processing.

Timer Basecount Registers

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|----------------------------------------------------------------------------------|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|---|---|---|---|---|---|---|---|---|
| Offset | Timer 0 - \$01110 Timer 1 - \$01150 Timer 2 - \$01190 Timer 3 - \$011D0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | TIMER BASECOUNT | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | CI | BC | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | R/W | R/W | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | 1 | \$00000000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

CI COUNT INHIBIT. Setting this bit to one inhibits counting for this timer. Setting this bit to zero allows counting to proceed.

BC BASE COUNT. This field contains the 31 bit count for this timer. When a value is written into this register and the CI bit transitions from a 1 to a 0, it is copied into the corresponding Current Count register and the toggle bit in the Current Count register is cleared. When the timer counts down to zero, the Current Count register is reloaded from the Base Count register and the interrupt becomes pending in MPIC processing.

Timer Vector/Priority Registers

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|----------------------------------------------------------------------------------|-----|-------|----|----|----|----|----|----|----|-------|----|----|----|------|----|----|----|----|----|----|----|--------|---|---|---|---|---|---|---|---|---|
| Offset | Timer 0 - \$01120 Timer 1 - \$01160 Timer 2 - \$011A0 Timer 3 - \$011E0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | TIMER VECTOR/PRIORITY | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | MASK | ACT | | | | | | | | | PRIOR | | | | | | | | | | | | VECTOR | | | | | | | | | |
| Operation | R/W | R | R | | | | | | | | R/W | | | | R | | | | | | | | R/W | | | | | | | | | |
| Reset | 1 | 0 | \$000 | | | | | | | | \$0 | | | | \$00 | | | | | | | | \$00 | | | | | | | | | |

MASK MASK. Setting this bit disables any further interrupts from this source. If the mask bit is cleared while the bit associated with this interrupt is set in the IPR, the interrupt request will be generated.

ACT ACTIVITY. The activity bit indicates that an interrupt has been requested or that it is in-service. The ACT bit is set to a one when its associated bit in the Interrupt Pending Register or In-Service Register is set.

PRIOR PRIORITY. Interrupt priority 0 is the lowest and 15 is the highest. Note that a priority level of 0 will not enable interrupts.

VECTOR VECTOR. This vector is returned when the Interrupt Acknowledge register is examined upon acknowledgement of the interrupt associated with this vector.

Timer Destination Registers

| Offset | Timer 0 - \$01130 Timer 1 - \$01170 Timer 2 - \$011B0 Timer 3 - \$011F0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|----------------------------------------------------------------------------------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|-----|-----|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | TIMER DESTINATION | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | P1 | P0 |
| Operation | R | | | | | | | | R | | | | | | | | R | | | | | | | | R | | | | R/W | R/W |
| Reset | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | 0 | 0 |

This register indicates the destinations for this timer's interrupts. Timer interrupts, operate in the Directed delivery interrupt mode. This register may specify multiple destinations (multicast delivery).

P1 PROCESSOR 1. The interrupt is directed to processor 1.

P0 PROCESSOR 0. The interrupt is directed to processor 0.

External Source Vector/Priority Registers

| Offset | Int Src 0 - \$10000 Int Src 1-> Int Src15 - \$10020 -> \$101E0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|-------------------------------------------------------------------|-----|-------|----|----|----|----|----|----|----|-----|-------|----|----|-------|------|----|----|----|--------|----|----|---|---|---|---|---|---|---|---|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | EXTERNAL SOURCE VECTOR/PRIORITY | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | MASK | ACT | | | | | | | | | POL | SENSE | | | PRIOR | | | | | VECTOR | | | | | | | | | | | | |
| Operation | R/W | R | R | | | | | | | | R/W | R/W | R | R | R/W | R | | | | R/W | | | | | | | | | | | | |
| Reset | 1 | 0 | \$000 | | | | | | | | 0 | 0 | 0 | 0 | \$0 | \$00 | | | | \$00 | | | | | | | | | | | | |

MASK MASK. Setting this bit disables any further interrupts from this source. If the mask bit is cleared while the bit associated with this interrupt is set in the IPR, the interrupt request will be generated.

ACT ACTIVITY. The activity bit indicates that an interrupt has been requested or that it is in-service. The ACT bit is set to a one when its associated bit in the Interrupt Pending Register or In-Service Register is set.

POL POLARITY. This bit sets the polarity for external interrupts. Setting this bit to a zero enables active low or negative edge. Setting this bit to a one enables active high or positive edge. Only External Interrupt Source 0 uses this bit in this register. For external interrupt sources 1 through 15. This bit is hard-wired to 0.

SENSE SENSE. This bit sets the sense for external interrupts. Setting this bit to a zero enables edge sensitive interrupts. Setting this bit to a one enables level sensitive interrupts. For external interrupt sources 1 through 15, setting this bit to a zero enables positive edge triggered interrupts. Setting this bit to a one enables active low level triggered interrupts.

PRIOR PRIORITY. Interrupt priority 0 is the lowest and 15 is the highest.

Note A priority level of 0 will not enable interrupts.

VECTOR VECTOR. This vector is returned when the Interrupt Acknowledge register is examined upon acknowledgement of the interrupt associated with this vector.

External Source Destination Registers

| Offset | Int Src 0 - \$10010 Int Src 1-> Int Src 15 - \$10030 -> \$101F0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------|--------------------------------------------------------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

This register indicates the possible destinations for the external interrupt sources. These interrupts operate in the Distributed interrupt delivery mode.

P1 PROCESSOR 1. The interrupt is pointed to processor 1.

P0 PROCESSOR 0. The interrupt is pointed to processor 0.

Raven-Detected Errors Vector/Priority Register

| Offset | \$10200 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|---------------------------------------|-----|-------|---|---|---|---|---|---|---|-------|---|---|-------|-----|------|---|---|---|---|---|---|--------|------|---|---|---|---|---|---|---|---|---|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | |
| Name | RAVEN DETECTED ERRORS VECTOR/PRIORITY | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | MASK | ACT | | | | | | | | | SENSE | | | PRIOR | | | | | | | | | VECTOR | | | | | | | | | | |
| Operation | R/W | R | R | | | | | | | | R | R | R | R | R/W | R | | | | | | | | R/W | | | | | | | | | |
| Reset | 1 | 0 | \$000 | | | | | | | | 0 | 1 | 0 | 0 | \$0 | \$00 | | | | | | | | \$00 | | | | | | | | | |

MASK MASK. Setting this bit disables any further interrupts from this source. If the mask bit is cleared while the bit associated with this interrupt is set in the IPR, the interrupt request will be generated.

ACT ACTIVITY. The activity bit indicates that an interrupt has been requested or that it is in-service. The ACT bit is set to a one when its associated bit in the Interrupt Pending Register or In-Service Register is set.

SENSE SENSE. This bit sets the sense for the internal Raven Detected Errors interrupt. This bit is hard wired to 1 to enable active low level sensitive interrupts.

PRIOR PRIORITY. Interrupt priority 0 is the lowest and 15 is the highest. Note that a priority level of 0 will not enable interrupts.

VECTOR VECTOR. This vector is returned when the Interrupt Acknowledge register is examined upon acknowledgement of the interrupt associated with this vector.

Raven-Detected Errors Destination Register

| Offset | \$10210 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|----------------------------------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|----|----|---|--|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | |
| | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | | | | |
| Name | RAVEN DETECTED ERROR DESTINATION | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | P1 | P0 | | |
| Operation | R | | | | | | | | R | | | | | | | | R | | | | | | | | R | | | | | | | | W | W | | |
| Reset | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | 0 | 0 | | |

This register indicates the possible destinations for the Raven detected error interrupt source. These interrupts operate in the Distributed interrupt delivery mode.

P1 PROCESSOR 1. The interrupt is pointed to processor 1.

P0 PROCESSOR 0. The interrupt is pointed to processor 0.

Interprocessor Interrupt Dispatch Registers

| Offset | Processor 0 \$20040, \$20050, \$20060, \$20070 Processor 1 \$21040, \$21050,\$21060, \$21070 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|-------------------------------------------------------------------------------------------------|----|----|----|----|----|----|----|------|----|----|----|----|----|----|----|------|----|----|----|----|----|---|---|------|---|---|---|----|----|---|---|
| Bit | 31 | 30 | 29 | 28 | 27 | 26 | 25 | 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | IPI DISPATCH | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | p1 | p0 | | |
| Operation | R | | | | | | | | R | | | | | | | | R | | | | | | | | R | | | | W | W | | |
| Reset | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | 0 | 0 | | |

There are four Interprocessor Interrupt Dispatch Registers. Writing to an IPI Dispatch Register with the P0 and/or P1 bit set causes an interprocessor interrupt request to be sent to one or more processors.

Note Each IPI Dispatch Register has two addresses. These registers are considered to be per processor registers and there is one address per processor. Reading these registers returns zeros.

P1 PROCESSOR 1. The interrupt is directed to processor 1.

P0 PROCESSOR 0. The interrupt is directed to processor 0.

Current Task Priority Registers

| Offset | Processor 0 \$20080 Processor 1 \$21080 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------|--------------------------------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

There is one Task Priority Register per processor. Priority levels from 0 (lowest) to 15 (highest) are supported. Setting the Task Priority Register to 15 masks all interrupts to this processor. Hardware will set the task register to \$F when it is reset or when the Init bit associated with this processor is written to a one.

TP TASK PRIORITY. Task priority of processor

Interrupt Acknowledge Registers

| Offset | Processor 0 \$200A0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|---------------------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|------|---|---|---|---|---|---|---|--------|---|---|---|---|---|---|---|
| | Processor 1 \$210A0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Name | | | | | | | | | | | | | | | | | | | | | | | | | VECTOR | | | | | | | |
| Operation | R | | | | | | | | R | | | | | | | | R | | | | | | | | R | | | | | | | |
| Reset | \$00 | | | | | | | | \$00 | | | | | | | | \$00 | | | | | | | | \$FF | | | | | | | |

On PowerPC-based systems, Interrupt Acknowledge is implemented as a read request to a memory-mapped Interrupt Acknowledge register. Reading the Interrupt Acknowledge register returns the interrupt vector corresponding to the highest priority pending interrupt. Reading this register also has the following side effects.

- ❑ The associated bit in the Interrupt Pending Register is cleared.
- ❑ Reading this register will update the In-Service register.

Reading this register without a pending interrupt will return a value of \$FF hex.

VECTOR VECTOR. This vector is returned when the Interrupt Acknowledge register is read.

End-of-Interrupt Registers

| Offset | Processor 0 \$200B0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--------|---------------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Processor 1 \$210B0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 3 | 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

EOI END OF INTERRUPT. There is one EOI register per processor. EOI Code values other than 0 are currently undefined. Data values written to this register are ignored; zero is assumed. Writing to this register signals the end of processing for the highest priority interrupt currently in service by the associated processor. The write operation will update the In-Service register by retiring the highest priority interrupt. Reading this register returns zeros.

Programming Notes

Disabling Watchdog Timers

Both Raven Watchdog timers on the MCP750 are enabled, following a board reset, with a time-out period of approximately 5 seconds (0.5 second Raven default time-out, plus 4.5 seconds of external delay). PPCBUG 3.4 disables both Watchdog timers in Raven during initialization following power up or reset.

If the timers are enabled and not reset before the time-out period:

- Timer 1 will generate MPIC interrupt 4
- Timer 2 will generate a board reset

If PPCBUG is replaced on the board, the replacement software must disable or reset these timers during initialization before the time-out period.

External Interrupt Service

The following summarizes how an external interrupt is serviced:

1. An external interrupt occurs.
2. The processor state is saved in the machine status save/restore registers. A new value is loaded into the Machine State Register (MSR). The External Interrupt Enable bit in the new MSR (MSR_{ee}) is set to zero. Control is transferred to the O/S external interrupt handler.
3. The external interrupt handler calculates the address of the Interrupt Acknowledge register for this processor (MPIC Base Address + 0x200A00 + (processor ID shifted left 12 bits)).
4. The external interrupt handler issues an Interrupt Acknowledge request to read the interrupt vector from the MPIC. If the interrupt vector indicates the interrupt source is the 8259, the interrupt handler issues a second Interrupt Acknowledge request to read the interrupt vector from the 8259. The Raven MPIC does not interact with the vector fetch from the 8259.
5. The interrupt handler saves the processor state and other interrupt-specific information in system memory and re-enables for external interrupts (the MSR_{ee} bit is set to 1). Raven MPIC blocks interrupts from sources with equal or lower priority until an End-of-Interrupt is received for that interrupt source. Interrupts from higher priority interrupt sources continue to be enabled. If the interrupt source was the 8259, the interrupt handler issues an EOI request to the MPIC. This resets the In-Service bit for the 8259 within the Raven MPIC and allows it to recognize higher priority interrupt requests, if any, from the 8259. If none of the nested interrupt modes of the 8259 are enabled, the interrupt handler issues an EOI request to the 8259.
 - a. The device driver interrupt service routine associated with this interrupt vector is invoked.

- b. If the interrupt source was not the 8259, the interrupt handler issues an EOI request for this interrupt vector to the MPIC. If the interrupt source was the 8259 and any of the nested interrupt modes of the 8259 are enabled, the interrupt handler issues an EOI request to the 8259.

Normally, interrupts from ISA devices are connected to the 8259 interrupt controller. ISA devices typically rely on the 8259 Interrupt Acknowledge to flush buffers between the ISA device and system memory. If interrupts from ISA devices are directly connected to the Raven MPIC (bypassing the 8259), the device driver interrupt service routine must read status from the ISA device to ensure buffers between the device and system memory are flushed.

Reset State

After a power-on reset the Raven MPIC state is:

- ❑ Current task priority for all CPUs set to 15.
- ❑ All interrupt source priorities set to zero.
- ❑ All interrupt source mask bits set to a one.
- ❑ All interrupt source activity bits cleared.
- ❑ Processor Init Register is cleared.
- ❑ All counters stopped and interrupts disabled.
- ❑ Controller mode set to 8259 pass-through.

Operation

Interprocessor Interrupts

Four Interprocessor Interrupt (IPI) channels are provided for use by all processors. During system initialization, the IPI vector/priority registers for each channel should be programmed to set the priority and vector returned for each IPI event. During system operation, a processor may generate an IPI by writing a destination mask to one of the IPI dispatch registers.

Note that each IPI dispatch register is shared by both processors. Each IPI dispatch register has two addresses but they are shared by both processors. That is, there is a total of four IPI dispatch registers in the Raven MPIC.

The IPI mechanism may be used for self interrupts by programming the dispatch register with the bit mask for the originating processor.

Dynamically Changing I/O Interrupt Configuration

The interrupt controller provides a mechanism for safely changing the vector, priority, or destination of I/O interrupt sources. This is provided to support systems which allow dynamic configuration of I/O devices. In order to change the vector, priority, or destination of an active interrupt source, the following sequence should be performed:

1. Mask the source using the MASK bit in the vector/priority register.
2. Wait for the activity bit (ACT) for that source to be cleared.
3. Make the desired changes.
4. Unmask the source.

This sequence ensures that the vector, priority, destination, and mask information remain valid until all processing of pending interrupts is complete.

EOI Register

Each processor has a private EOI register which is used to signal the end of processing for a particular interrupt event. If multiple nested interrupts are in service, the EOI command terminates the interrupt service of the highest priority source. Once an interrupt is acknowledged, only sources of higher priority will be allowed to interrupt the processor until the EOI command is received. This register should always be written with a value of zero which is the nonspecific EOI command.

Interrupt Acknowledge Register

Upon receipt of an interrupt signal, the processor may read this register to retrieve the vector of the interrupt source which caused the interrupt.

8259 Mode

The 8259 mode bits control the use of an external 8259 pair for PC-AT compatibility. Following a reset, this mode is set for pass-through, which essentially disables the advanced controller and passes an 8259 input on external interrupt source 0 directly through to processor zero. During interrupt controller initialization this channel should be programmed for mixed mode in order to take advantage of the interrupt delivery modes.

Current Task Priority Level

Each processor has a separate Current Task Priority Level register. The system software uses this register to indicate the relative priority of the task running on the corresponding processor. The interrupt controller will not deliver an interrupt to a processor unless it has a priority level which is greater than the current task priority level of that processor. This value is also used in determining the destination for interrupts which are delivered using the distributed delivery mode.

Architectural Notes

The hardware and software overhead required to update the task priority register synchronously with instruction execution may far outweigh the anticipated benefits of the task priority register. To minimize this overhead, the interrupt controller architecture should allow the task priority register to be updated asynchronously with respect to instruction execution. Lower priority interrupts may continue to occur for an indeterminate number of cycles after the processor has updated the task priority register. If this is not acceptable, the interrupt controller architecture should recommend that, if the task priority register is not implemented with the processor, the task priority register should be updated only when the processor enter or exits an idle state.

Only when the task priority register is integrated within the processor, (such that it can be accessed as quickly as the MSRee bit, for example), should the architecture require the task priority register to be updated synchronously with instruction execution.

Introduction

This chapter provides a functional description and programming model for the Falcon DRAM controller ASIC (hereinafter referred to as Falcon). This chip is designed for use on the PowerPC family of boards. It is used in sets of two to provide the interface between the PowerPC 60x bus (also called MPC60x bus or MPC bus) and a 144-bit ECC-DRAM memory system. It also provides an interface to ROM/Flash.

This chapter also contains information on using the device in a system along with programming and testing it in a system.

Bit Ordering Convention

All Falcon based signals are named using big-endian bit ordering (bit 0 is the most significant bit).

Features

- ❑ DRAM Interface
 - Double-bit error detect/Single-bit error correct on 72-bit basis.
 - 4 blocks with up to 1GB each (using 64Mx1 devices).
 - Programmable base address for each block.
 - Two-way interleave factor.
 - Built-in Refresh/Scrub.
- ❑ Error Notification for DRAM
 - Software programmable Interrupt on Single/Double-Bit Error.
 - Error address and Syndrome Log Registers for Error Logging.
 - Does not provide TEA_ on Double-Bit Error (chip does not have a TEA_ pin.)

- ❑ ROM/Flash Interface
 - Two blocks with each block being 16 bits wide (8 bits per Falcon), or 64 bits wide (32 bits per Falcon).
 - Software programmable access time for each block.

Block Diagrams

[Figure 3-1](#) depicts a Falcon pair as it would be connected in a system. [Figure 3-2](#) shows the Falcon's internal data paths. [Figure 3-3](#) shows the overall DRAM connections.

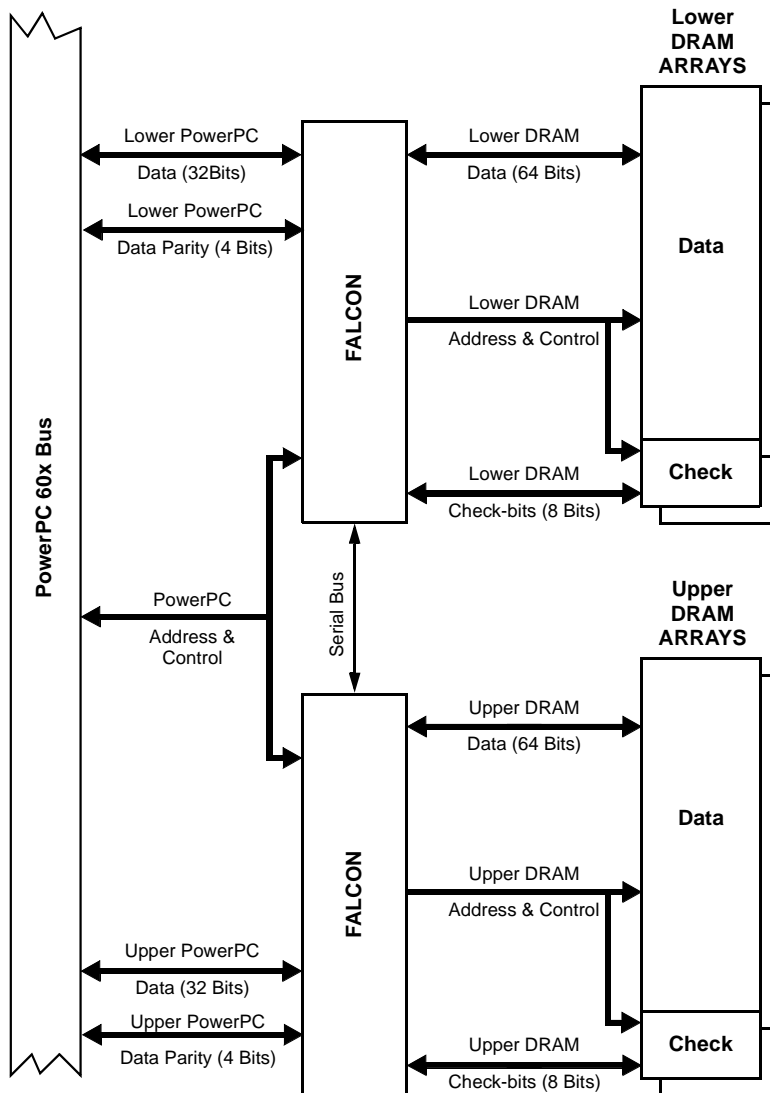


Figure 3-1. Falcon Pair Used with DRAM in a System

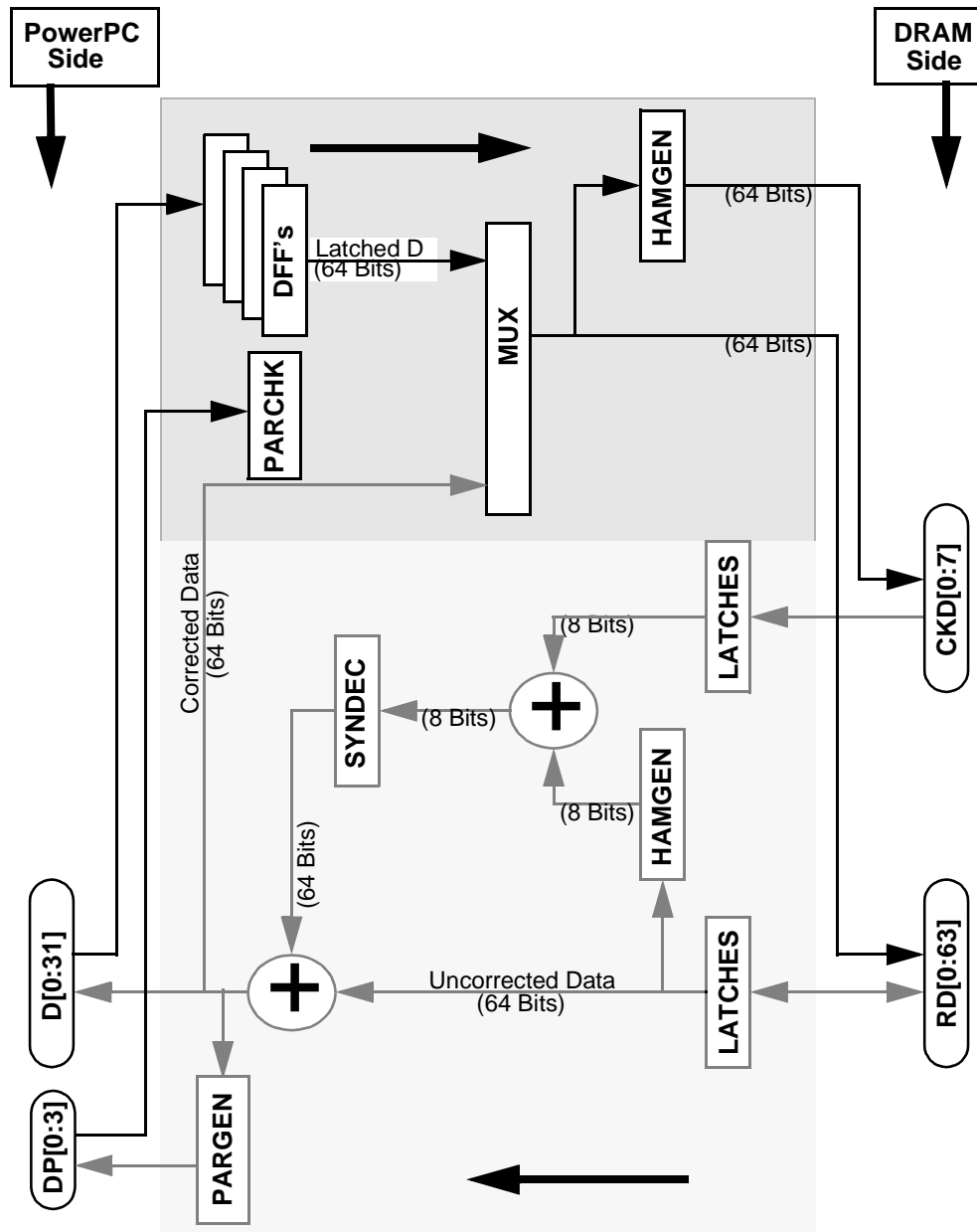


Figure 3-2. Falcon Internal Data Paths (Simplified)

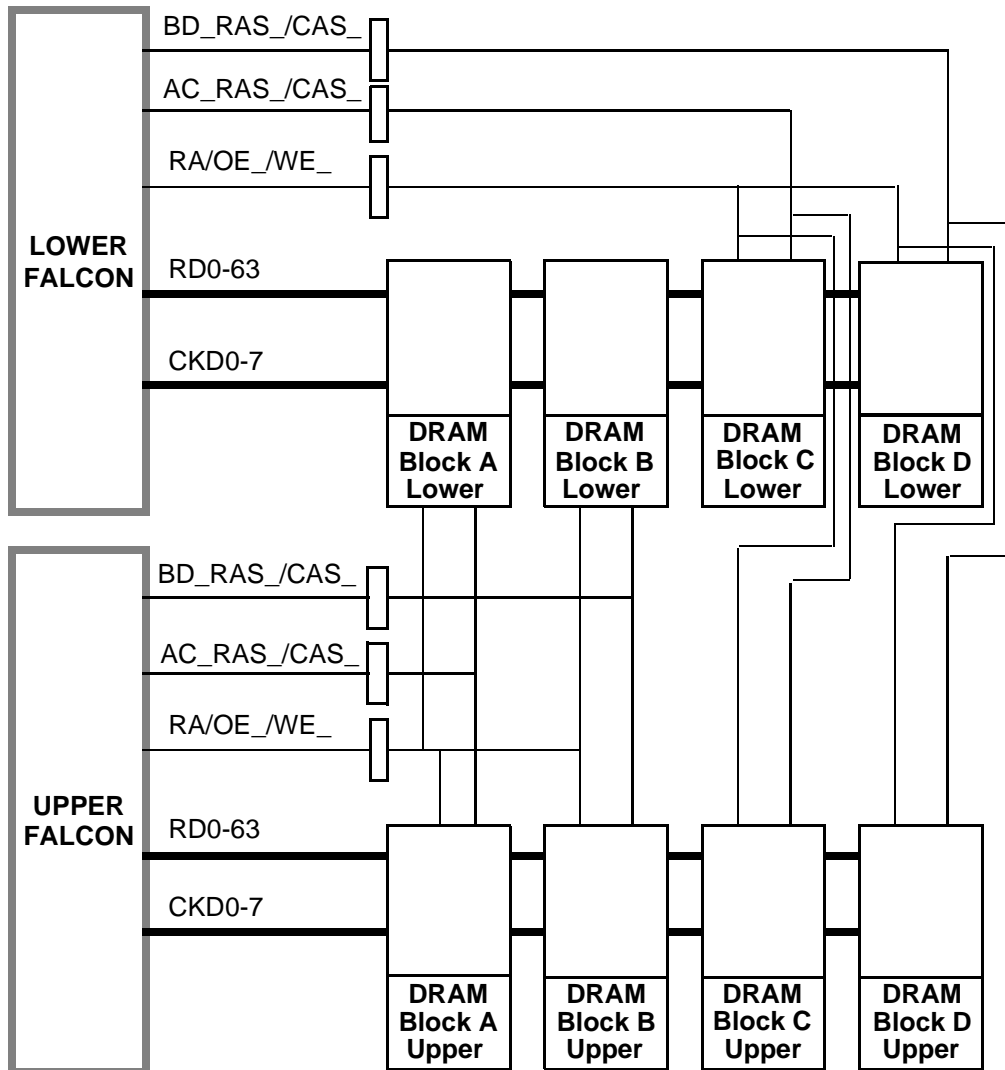


Figure 3-3. Overall DRAM Connections

Functional Description

The following sections describe the logical function of the ASIC. The Falcon is designed to be used as a set of two chips. A pair of Falcons works with $x1$ or wider DRAM memory devices to form a memory system for the PowerPC 60x bus. A pair of Falcons that is connected to implement a memory control function is referred to in this document as a Falcon pair.

Performance

Four-beat Reads/Writes

The Falcon pair is specifically designed to provide maximum performance for cache line (four-beat) cycles to and from the PowerPC 60x bus at 66 MHz. This is done by providing a two-way interleave between the 64-bit PowerPC 60x data bus and the 128-bit (144 with check-bits) DRAM bus. When a PowerPC 60x bus master begins a quad-aligned, four-beat read to DRAM, the Falcon pair accesses the full 144-bit width of DRAM at once so that when the DRAM access time is reached, not only is the first 64-bit double-word of data ready to be transferred to the PowerPC 60x bus master, but so is the next. While the Falcon pair is presenting the first two double-words to the PowerPC 60x bus, it cycles CAS without cycling RAS to obtain the next two double-words. The Falcon pair transfers the next two double-words to the PowerPC 60x bus after 0 or more idle clocks.

The Falcon pair also takes advantage of the fact that PowerPC 60x processors can do address pipelining. Many times while a data cycle is finishing, the PowerPC 60x processor begins a new address cycle. The Falcon pair can begin the next DRAM access earlier when this happens, thus shortening the access time. Further savings come when the new address cycle is to an address close enough to the previous one that it falls within the same row in the DRAM array. When this happens, the Falcon pair can transfer the data for the next cycle by cycling CAS without cycling RAS.

Single-beat Reads/Writes

Single-beat cycles to and from the PowerPC 60x bus do not achieve data rates as high as do four-beat cycles. The Falcon pair does take advantage of the PowerPC 60x address pipelining as much as possible for single-beat accesses.

Single-beat writes are the slowest kind of accesses because they require that the Falcon pair perform a read cycle then a write cycle to the DRAM in order to complete. Fortunately, in most 60x systems, single-beat accesses can be held to a minimum especially with data cache and copyback modes in place.

DRAM Speeds

The Falcon pair can be configured for 3 different speeds of DRAM: 50ns, 60ns, and 70ns. When the Falcon pair is configured for 50ns DRAMs, it assumes that the devices are Extended Data Out (EDO) parts. When the Falcon pair is configured for 70ns DRAMs, it assumes that the devices are fast page mode parts. When the pair is configured for 60ns DRAMs, it allows the devices to be either fast page or EDO parts. Performance summaries using the different devices are shown in [Table 3-1](#), [Table 3-2](#), and [Table 3-3](#).

Table 3-1. PowerPC 60x Bus to DRAM Access Timing when Configured for 70ns Fast Page Devices

| ACCESS TYPE | CLOCK PERIODS REQUIRED FOR: | | | | Total Clocks |
|---------------------------------------------------|-----------------------------|----------|----------|----------|--------------|
| | 1st Beat | 2nd Beat | 3rd Beat | 4th Beat | |
| 4-Beat Read after Idle (Quad-word aligned) | 10 | 1 | 3 | 1 | 15 |
| 4-Beat Read after Idle (Quad-word misaligned) | 10 | 4 | 1 | 1 | 16 |
| 4-Beat Read after 4-Beat Read (Quad-word aligned) | $9/3^1$ | 1 | 3 | 1 | 14/8 |
| 4-Beat Read after 4-Beat Read (misaligned) | $7/2^1$ | 4 | 1 | 1 | 13/8 |
| 4-Beat Write after Idle | 4 | 1 | 1 | 1 | 7 |

Table 3-1. PowerPC 60x Bus to DRAM Access Timing when Configured for 70ns Fast Page Devices (Continued)

| ACCESS TYPE | CLOCK PERIODS REQUIRED FOR: | | | | Total Clocks |
|-----------------------------------------------------|-----------------------------|----------|----------|----------|--------------|
| | 1st Beat | 2nd Beat | 3rd Beat | 4th Beat | |
| 4-Beat Write after 4-Beat Write (Quad-word aligned) | 10/6 ¹ | 1 | 1 | 1 | 13/9 |
| 1-Beat Read after Idle | 10 | - | - | - | 10 |
| 1-Beat Read after 1-Beat Read | 11/7 ¹ | - | - | - | 11/7 |
| 1-Beat Write after Idle | 4 | - | - | - | 4 |
| 1-Beat Write after 1-Beat Write | 15/11 ¹ | - | - | - | 15/11 |

Notes

1. These numbers assume that the PowerPC 60x bus master is doing address pipelining with TS_ occurring at the minimum time after AACK_ is asserted. Also, the two numbers shown in the 1st beat column are for page miss/page hit.
2. In some cases, the numbers shown are averages and specific instances may be longer or shorter.

Table 3-2. PowerPC 60x Bus to DRAM Access Timing when Configured for 60ns Fast Page Devices

| ACCESS TYPE | CLOCK PERIODS REQUIRED FOR: | | | | Total Clocks |
|---------------------------------------------------|-----------------------------|----------|----------|----------|--------------|
| | 1st Beat | 2nd Beat | 3rd Beat | 4th Beat | |
| 4-Beat Read after Idle (Quad-word aligned) | 9 | 1 | 2 | 1 | 13 |
| 4-Beat Read after Idle (Quad-word misaligned) | 9 | 3 | 1 | 1 | 14 |
| 4-Beat Read after 4-Beat Read (Quad-word aligned) | 7/3 ¹ | 1 | 2 | 1 | 11/7 |
| 4-Beat Read after 4-Beat Read (misaligned) | 6/2 ¹ | 3 | 1 | 1 | 11/7 |

Table 3-2. PowerPC 60x Bus to DRAM Access Timing when Configured for 60ns Fast Page Devices (Continued)

| ACCESS TYPE | CLOCK PERIODS REQUIRED FOR: | | | | Total Clocks |
|-----------------------------------------------------|-----------------------------|----------|----------|----------|--------------|
| | 1st Beat | 2nd Beat | 3rd Beat | 4th Beat | |
| 4-Beat Write after Idle | 4 | 1 | 1 | 1 | 7 |
| 4-Beat Write after 4-Beat Write (Quad-word aligned) | 7/3 ¹ | 1 | 1 | 1 | 10/6 |
| 1-Beat Read after Idle | 9 | - | - | - | 9 |
| 1-Beat Read after 1-Beat Read | 9/6 ¹ | - | - | - | 9/6 |
| 1-Beat Write after Idle | 4 | - | - | - | 4 |
| 1-Beat Write after 1-Beat Write | 13/10 ¹ | - | - | - | 13/10 |

Notes

1. These numbers assume that the PowerPC 60x bus master is doing address pipelining with TS_ occurring at the minimum time after AACK_ is asserted. Also, the two numbers shown in the 1st beat column are for page miss/page hit.
2. In some cases, the numbers shown are averages and specific instances may be longer or shorter.

Table 3-3. PowerPC 60x Bus to DRAM Access Timing when Configured for 50ns EDO Devices

| ACCESS TYPE | CLOCK PERIODS REQUIRED FOR: | | | | Total Clocks |
|-----------------------------------------------------|-----------------------------|----------|----------|----------|--------------|
| | 1st Beat | 2nd Beat | 3rd Beat | 4th Beat | |
| 4-Beat Read after Idle (Quad-word aligned) | 8 | 1 | 1 | 1 | 11 |
| 4-Beat Read after Idle (Quad-word misaligned) | 8 | 2 | 1 | 1 | 12 |
| 4-Beat Read after 4-Beat Read (Quad-word aligned) | $5/2^1$ | 1 | 1 | 1 | 8/5 |
| 4-Beat Read after 4-Beat Read (misaligned) | $4/2^1$ | 2 | 1 | 1 | 8/6 |
| 4-Beat Write after Idle | 4 | 1 | 1 | 1 | 7 |
| 4-Beat Write after 4-Beat Write (Quad-word aligned) | $4/3^1$ | 1 | 1 | 1 | 7/6 |
| 1-Beat Read after Idle | 8 | - | - | - | 8 |
| 1-Beat Read after 1-Beat Read | $7/5^1$ | - | - | - | 7/5 |
| 1-Beat Write after Idle | 4 | - | - | - | 4 |
| 1-Beat Write after 1-Beat Write | $9/7^1$ | - | - | - | 9/7 |

Notes

1. These numbers assume that the PowerPC 60x bus master is doing address pipelining with TS_ occurring at the minimum time after AACK_ is asserted. Also, the two numbers shown in 1st beat column are for page miss/page hit.
2. In some cases, the numbers shown are averages and specific instances may be longer or shorter.

ROM/Flash Speeds

The Falcon pair provides the interface for two blocks of ROM/Flash. Access times to ROM/Flash are programmable for each block. Access times are also affected by block width. The access times for ROM/Flash are shown in [Table 3-4](#), [Table 3-5](#), [Table 3-6](#), and [Table 3-7](#).

Table 3-4. PowerPC 60x Bus to ROM/Flash Access Timing when configured for 180ns Devices

| ACCESS TYPE | CLOCK PERIODS REQUIRED FOR: | | | | | | | | Total Clocks | |
|-------------------------------|-----------------------------|------------|------------|------------|------------|------------|------------|------------|-----------------|------------|
| | 1st Beat | | 2nd Beat | | 3rd Beat | | 4th Beat | | | |
| | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits |
| 4-Beat Read | 68 | 20 | 64 | 16 | 64 | 16 | 64 | 16 | 260 | 68 |
| 4-Beat Write | N/A | | | | | | | | N/A | |
| 1-Beat Read (1 byte) | 20 | 20 | - | - | - | - | - | - | 20 | 20 |
| 1-Beat Read (2 to 8 bytes) | 68 | 20 | - | - | - | - | - | - | 68 | 20 |
| 1-Beat Write | 19 | 19 | - | - | - | - | - | - | 19 | 19 |

Table 3-5. PowerPC 60x Bus to ROM/Flash Access Timing when configured for 120ns Devices

| ACCESS TYPE | CLOCK PERIODS REQUIRED FOR: | | | | | | | | Total Clocks | |
|-------------------------------|-----------------------------|------------|------------|------------|------------|------------|------------|------------|-----------------|------------|
| | 1st Beat | | 2nd Beat | | 3rd Beat | | 4th Beat | | | |
| | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits |
| 4-Beat Read | 52 | 16 | 48 | 12 | 48 | 12 | 48 | 12 | 196 | 52 |
| 4-Beat Write | N/A | | | | | | | | N/A | |
| 1-Beat Read (1 byte) | 16 | 16 | - | - | - | - | - | - | 16 | 16 |
| 1-Beat Read (2 to 8 bytes) | 52 | 16 | - | - | - | - | - | - | 52 | 16 |
| 1-Beat Write | 19 | 19 | - | - | - | - | - | - | 19 | 19 |

Table 3-6. PowerPC 60x Bus to ROM/Flash Access Timing when configured for 75ns Devices

| ACCESS TYPE | CLOCK PERIODS REQUIRED FOR: | | | | | | | | Total Clocks | |
|-------------------------------|-----------------------------|------------|------------|------------|------------|------------|------------|------------|-----------------|------------|
| | 1st Beat | | 2nd Beat | | 3rd Beat | | 4th Beat | | | |
| | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits |
| 4-Beat Read | 40 | 13 | 36 | 9 | 36 | 9 | 36 | 9 | 148 | 40 |
| 4-Beat Write | N/A | | | | | | | | N/A | |
| 1-Beat Read (1 byte) | 13 | 13 | - | - | - | - | - | - | 13 | 13 |
| 1-Beat Read (2 to 8 bytes) | 40 | 13 | - | - | - | - | - | - | 40 | 13 |
| 1-Beat Write | 19 | 19 | - | - | - | - | - | - | 19 | 19 |

Table 3-7. PowerPC 60x Bus to ROM/Flash Access Timing when configured for 45ns Devices

| ACCESS TYPE | CLOCK PERIODS REQUIRED FOR: | | | | | | | | Total Clocks | |
|-------------------------------|-----------------------------|------------|------------|------------|------------|------------|------------|------------|-----------------|------------|
| | 1st Beat | | 2nd Beat | | 3rd Beat | | 4th Beat | | | |
| | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits | 16 Bits | 64 Bits |
| 4-Beat Read | 32 | 11 | 28 | 7 | 28 | 7 | 28 | 7 | 116 | 32 |
| 4-Beat Write | N/A | | | | | | | | N/A | |
| 1-Beat Read (1 byte) | 11 | 11 | - | - | - | - | - | - | 11 | 11 |
| 1-Beat Read (2 to 8 bytes) | 32 | 11 | - | - | - | - | - | - | 32 | 11 |
| 1-Beat Write | 19 | 19 | - | - | - | - | - | - | 19 | 19 |

PowerPC 60x Bus Interface

The Falcon pair has a PowerPC slave interface only. It has no PowerPC master interface. The slave interface is the mechanism for all accesses to DRAM, ROM/Flash, and internal and external register sets.

Responding to Address Transfers

When the Falcon pair detects an address transfer that it is to respond to, it asserts `AACK_` immediately if there is no uncompleted PowerPC 60x bus data transfer in process. If there is one in process, then the Falcon pair waits and asserts `AACK_` coincident with the uncompleted data transfer's last data beat if the Falcon pair is the slave for the previous data. If it is not, it holds off `AACK_` until the `CLOCK` after the previous data transfer's last data beat.

Completing Data Transfers

If an address transfer to the Falcon pair will have an associated data transfer, the Falcon pair begins a read or write cycle to the accessed entity (DRAM/ROM/Flash/Internal and External Register) as soon as the entity is free. If the data transfer will be a read, the Falcon pair begins providing data to the PowerPC 60x bus as soon as the entity has data ready and the PowerPC 60x data bus is granted. If the data transfer will be a write, the Falcon pair begins latching data from the PowerPC data bus as soon as any previously latched data is no longer needed and the PowerPC 60x data bus is available.

Data Parity

The Falcon pair has 8 DP pins (4 per Falcon) for generating and checking PowerPC 60x data bus parity. In addition, each Falcon has a `DPERR_` pin to provide real-time data parity error notification for its half of the PowerPC 60x data bus.

During read cycles that access the Falcon pair, the pair generates the correct value on `DP0-DP7` so that each data byte lane along with its corresponding DP signal has odd parity. This can be changed on a lane basis to even parity by software bits that can force the generation of wrong (even) parity.

During write cycles to the Falcon pair, each Falcon in the pair checks each of its four PowerPC 60x data byte lanes and the corresponding DP signal for odd parity. If any of the four lanes has even parity, that Falcon logs the error in the CSR and can generate a machine check if so enabled. In addition to logging the error, that Falcon also pulses its DPERR_ signal true for the duration of one clock period, two clock periods after the TA_ during which the error data is captured.

While normal (default) operation is for the Falcon to check data parity only on writes to itself, it can also be programmed to check data parity on all reads or writes to any device on the PowerPC 60x bus

Cache Coherency

The Falcon supports cache coherency to DRAM only. It does this by monitoring the ARTRY_ control signal on the PowerPC 60x bus and behaving appropriately when it is asserted. When ARTRY_ is asserted, if the access is a DRAM read, the Falcon does not source the data for that access. If the access is a DRAM write, the Falcon does not write the data for that access. Depending upon when the retry occurs, the Falcon may cycle the DRAM even though the data transfer does not happen.

Cache Coherency Restrictions

The PowerPC 60x GBL_ signal must not be asserted in the CSR areas.

L2 Cache Support

The Falcon pair provides support for a look-aside L2 cache by implementing a hold-off input, L2CLM_. On cycles that select the Falcon pair, the Falcon pair samples L2CLM_ on the second rising edge of CLOCK after the assertion of TS_. If L2CLM_ is high, the Falcon pair responds normally to the cycle. If it is low, the Falcon pair ignores the cycle.

ECC

The Falcon pair performs single-bit error correction and double-bit error detection for DRAM. (No checking is provided for ROM/Flash.) The 64-bit wide PowerPC 60x data bus is divided into upper (DH0-DH31) and lower (DL0-DL31) halves. Each half is routed through a Falcon which multiplexes it with half of the DRAM data bus. Each Falcon connects to 64 DRAM data-bits and to 8 DRAM check-bits. The total DRAM array width is 144 bits ($2 \times [64+8]$).

Cycle Types

To support ECC, the Falcon pair always deals with DRAM using full width (144-bit) accesses. When the PowerPC 60x bus master requests any size read of DRAM, the Falcon pair reads 144 bits at least once. When the PowerPC 60x bus master requests a four-beat write to DRAM, the Falcon pair writes all 144 bits twice. When the PowerPC 60x bus master requests a single-beat write to DRAM, the Falcon pair performs a 144-bit wide read cycle to DRAM, merges in the appropriate PowerPC 60x bus write data, and writes 144 bits back to DRAM.

Error Reporting

The Falcon pair checks data from the DRAM during single- and four-beat reads, during single-beat writes, and during scrubs. [Table 3-8](#) shows the actions it takes for different errors during these accesses.

It is important to note that the Falcon pair does not assert TEA_ on double-bit errors. In fact, the Falcon pair does not have a TEA_ signal pin and it assumes that the system does not implement TEA_. The Falcon can, however, assert machine check (MCP_) on double-bit error.

Table 3-8. Error Reporting

| Error Type | Single-Beat /Four-Beat Read | Single-Beat Write | Four-Beat Write | Scrub |
|--------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Single-Bit Error | <p>Terminate the PowerPC 60x bus cycle normally.</p> <p>Provide corrected data to the PowerPC 60x bus master.</p> <p>Assert INT_ if so enabled.</p> | <p>Terminate the PowerPC 60x bus cycle normally.</p> <p>Correct the data read from DRAM, merge with the write data, and write the corrected, merged data to DRAM.</p> <p>Assert INT_ if so enabled.</p> | N/A ¹ | <p>This cycle is not seen on the PowerPC 60x bus.</p> <p>Write corrected data back to DRAM if so enabled.</p> <p>Assert INT_ if so enabled.</p> |
| Double-Bit Error | <p>Terminate the PowerPC 60x bus cycle normally.</p> <p>Provide miss-corrected, raw DRAM data to the PowerPC 60x bus master.</p> <p>Assert INT_ if so enabled.</p> <p>Assert MCP_ if so enabled.</p> | <p>Terminate the PowerPC 60x bus cycle normally.</p> <p>May or may not perform the write portion of the read-modify-write cycle to DRAM.²</p> <p>Assert INT_ if so enabled.</p> <p>Assert MCP_ if so enabled.</p> | N/A ¹ | <p>This cycle is not seen on the PowerPC 60x bus.</p> <p>May or may not perform the write portion of the read-modify-write cycle to DRAM.²</p> <p>Assert INT_ if so enabled.</p> |
| Triple- (or greater) Bit Error | Some of these errors are detected correctly and are treated the same as double-bit errors. The rest could show up as “no error” or “single-bit error”, both of which are incorrect. | | | |

Notes

1. No opportunity for error since no read of DRAM occurs during a four-beat write.
2. The recommended connection for Falcon is for WE* to connect from upper Falcon to both upper and lower DRAM's of banks A and B and for WE* of lower Falcon to connect to both upper and lower DRAM's of banks C and D. With this configuration, the write portion of single-bit-writes and of scrubs *does* happen if the double-bit error is in the lower portion of Banks A or B or if it is in the upper portion of Banks C or D

Error Logging

ECC error logging is facilitated by the Falcon because of its internal latches. When an error (single or double-bit) occurs in the DRAMs to which a Falcon is connected, it records the address and syndrome bits associated with the data in error.

Each Falcon performs this logging function independently of the other. Once a Falcon has logged an error, it does not log any more until the **elog** control /status bit has been cleared by software unless the currently logged error is single-bit and a new double-bit error is encountered.

The logging of errors that occur during scrub can be enabled/disabled in software. Refer to the [Error Logger Register](#) in this chapter.

ROM/Flash Interface

The Falcon pair provides the interface for two blocks of ROM/Flash. Each block provides addressing and control for up to 64MB.

Note No ECC error checking is provided for the ROM/Flash.

The ROM/Flash interface allows each block to be individually configured by jumpers and/or by software as follows:

1. Access for each block is controlled by three software programmable control register bits: an overall enable, a write enable, and a reset vector enable. The overall enable controls normal read accesses. The write enable is used to program Flash devices. The reset vector enable controls whether the block is also enabled at \$FFF00000 - \$FFFFFFFF. The overall enable and write enable bits are always cleared at reset. The reset vector enable bit is cleared or set at reset depending on external jumper configuration. This allows the board designer to use external jumpers to enable/disable Block A/B ROM/Flash as the source of reset vectors.

2. The base address for each block is software programmable. At reset, Block A's base address is \$FF000000 and Block B's base address is \$FF800000.

As noted above, in addition to appearing at the programmed base address, the first 1MB of Block A/B also appears at \$FFF00000-\$FFFFFFFF if the reset vector enable bit is set.

3. The assumed size for each block is software programmable. It is initialized to its smallest setting at reset.
4. The access time for each block is software programmable
5. The assumed width for Block A/B is determined by an external jumper at reset time. It also is available as a status bit and cannot be changed by software.

When the width status bit is cleared, the block's ROM/Flash is considered to be 16 bits wide, where each Falcon interfaces to 8 bits. In this mode, the following rules are enforced:

- a. Only single-byte writes are allowed (all other sizes are ignored),
- and-
- b. All reads are allowed (multiple accesses are performed to the ROM/Flash devices when the read is for greater than one byte).

When the width status bit is set, the block's ROM/Flash is considered to be 64 bits wide, where each Falcon interfaces with 32 bits. In this mode, the following rules are enforced:

- c. Only aligned, 4-byte writes should be attempted (all other sizes are ignored)

- and -

- d. All reads are allowed (multiple accesses to the ROM/Flash device are performed for burst reads).

For additional about the ROM/Flash, refer to the *Programming Model* section in this chapter.

In order to place code correctly in the ROM/Flash devices, address mapping information is required. Table 3-9 shows how PowerPC 60x addresses map to the ROM/Flash addresses when ROM/Flash is 16 bits wide (8 bits per Falcon). Table 3-10 shows how they map when ROM/Flash is 64 bits wide (32 bits per Falcon).

Table 3-9. PowerPC 60x to ROM/Flash Address Mapping when ROM/Flash is 16 Bits Wide (8 Bits per Falcon)

| PowerPC 60x A0-A31 | ROM/Flash A22-A0 | ROM/Flash Selected |
|--------------------|------------------|--------------------|
| \$XX000000 | \$000000 | Upper |
| \$XX000001 | \$000001 | Upper |
| \$XX000002 | \$000002 | Upper |
| \$XX000003 | \$000003 | Upper |
| \$XX000004 | \$000000 | Lower |
| \$XX000005 | \$000001 | Lower |
| \$XX000006 | \$000002 | Lower |
| \$XX000007 | \$000003 | Lower |
| \$XX000008 | \$000004 | Upper |
| \$XX000009 | \$000005 | Upper |
| \$XX00000A | \$000006 | Upper |
| \$XX00000B | \$000007 | Upper |
| \$XX00000C | \$000004 | Lower |
| \$XX00000D | \$000005 | Lower |
| \$XX00000E | \$000006 | Lower |

Table 3-9. PowerPC 60x to ROM/Flash Address Mapping when ROM/Flash is 16 Bits Wide (8 Bits per Falcon) (Continued)

| PowerPC 60x A0-A31 | ROM/Flash A22-A0 | ROM/Flash Selected |
|--------------------|------------------|--------------------|
| \$XX00000F | \$000007 | Lower |
| \$XXFFFFFF8 | \$7FFFFFFC | Upper |
| \$XXFFFFFF9 | \$7FFFFFFD | Upper |
| \$XXFFFFFFA | \$7FFFFFFE | Upper |
| \$XXFFFFFFB | \$7FFFFFFF | Upper |
| \$XXFFFFFFC | \$7FFFFFFC | Lower |
| \$XXFFFFFFD | \$7FFFFFFD | Lower |
| \$XXFFFFFFE | \$7FFFFFFE | Lower |
| \$XXFFFFFFF | \$7FFFFFFF | Lower |

Table 3-10. PowerPC 60x to ROM/Flash Address Mapping when ROM/Flash is 64 Bits Wide (32 Bits per Falcon)

| PowerPC 60x A0-A31 | ROM/Flash A22-A0 | ROM/Flash Selected |
|--------------------|------------------|--------------------|
| \$X0000000 | \$000000 | Upper |
| \$X0000001 | \$000000 | Upper |
| \$X0000002 | \$000000 | Upper |
| \$X0000003 | \$000000 | Upper |
| \$X0000004 | \$000000 | Lower |
| \$X0000005 | \$000000 | Lower |
| \$X0000006 | \$000000 | Lower |
| \$X0000007 | \$000000 | Lower |
| \$X0000008 | \$000001 | Upper |
| \$X0000009 | \$000001 | Upper |
| \$X000000A | \$000001 | Upper |
| \$X000000B | \$000001 | Upper |
| \$X000000C | \$000001 | Lower |
| \$X000000D | \$000001 | Lower |

Table 3-10. PowerPC 60x to ROM/Flash Address Mapping when ROM/Flash is 64 Bits Wide (32 Bits per Falcon) (Continued)

| PowerPC 60x A0-A31 | ROM/Flash A22-A0 | ROM/Flash Selected |
|--------------------|------------------|--------------------|
| \$X000000E | \$000001 | Lower |
| \$X000000F | \$000001 | Lower |
| \$X3FFFFFF0 | \$7FFFFFFE | Upper |
| \$X3FFFFFF1 | \$7FFFFFFE | Upper |
| \$X3FFFFFF2 | \$7FFFFFFE | Upper |
| \$X3FFFFFF3 | \$7FFFFFFE | Upper |
| \$X3FFFFFF4 | \$7FFFFFFE | Lower |
| \$X3FFFFFF5 | \$7FFFFFFE | Lower |
| \$X3FFFFFF6 | \$7FFFFFFE | Lower |
| \$X3FFFFFF7 | \$7FFFFFFE | Lower |
| \$X3FFFFFF8 | \$7FFFFFFF | Upper |
| \$X3FFFFFF9 | \$7FFFFFFF | Upper |
| \$X3FFFFFFA | \$7FFFFFFF | Upper |
| \$X3FFFFFFB | \$7FFFFFFF | Upper |
| \$X3FFFFFFC | \$7FFFFFFF | Lower |
| \$X3FFFFFFD | \$7FFFFFFF | Lower |
| \$X3FFFFFFE | \$7FFFFFFF | Lower |
| \$X3FFFFFFF | \$7FFFFFFF | Lower |

Refresh/Scrub

Refresh/Scrub is done differently based on which DRAM blocks are populated.

Blocks A and/or B Present, Blocks C and D Not Present

The Falcon pair performs refresh by doing a burst of four RAS_ cycles approximately once every 60 μ s. This increases to once every 30 μ s when certain DRAM devices are used. (Controlled by the **ram_fref** bit in the

status registers.) RAS_ is asserted to both of Blocks A and B during each of the 4 cycles. Along with RAS_, the Falcon pair also asserts CAS_ with (OE_ then WE_) to one of the blocks during one of the four cycles. This forms a read-modify-write which is a scrub cycle to that location.

After each of the 4 cycles, the DRAM row address increments by one. When it reaches all 1s, it rolls over and starts over at 0. Each time the row address rolls over, the block that is scrubbed toggles between A and B. Every second time that the row address rolls over, whichever of the 4 cycles that is a scrub changes from 1st to 2nd, from 2nd to 3rd, from 3rd to 4th, or from 4th to 1st. Every eighth time that the row address rolls over, the column address increments by one. When the column address reaches all 1s, it rolls over and starts over at 0. Each time the column address rolls over, the SC1, SC0 bits in the scrub/refresh register increment by one.

Blocks A and/or B Present, Blocks C and/or D Present

The Falcon pair performs refresh by doing a burst of four RAS_ cycles approximately once every 30μs. This increases to once every 15μs when certain DRAM devices are used. (Controlled by the ram_fref bit in the status registers.) RAS_ is asserted to blocks A and B during the first cycle, to blocks C and D during the second cycle, back to blocks A and B during the third cycle and to blocks C and D during the fourth cycle. Along with RAS, the Falcon pair also asserts CAS_ with (OE_ then WE_) to one of the blocks during one of the four cycles. This forms a read-modify-write which is a scrub cycle to that location.

After the second and fourth cycles, the DRAM row address increments by one. When it reaches all 1s, it rolls over and starts over at 0. Each time the row address rolls over, the block that is scrubbed toggles between A/C and B/D. Every second time the row address rolls over, whichever of the 4 cycles that is a scrub changes from 1st to 2nd, from 2nd to 3rd, from 3rd to 4th, or from 4th to 1st. Every eighth time that the row address rolls over, the column address increments by one. When the column address reaches all 1's, it rolls over and starts over at 0. Each time the column address rolls over, the SC1, SC0 bits in the scrub/refresh register increment by one.

An entire refresh of DRAM is achieved every time the row address rolls over, and an entire scrub of DRAM is achieved every time the column address rolls over.

During scrub cycles, if the SWEN bit is cleared, the Falcon pair *does not* perform the write portion of the read-modify write cycle. If the SWEN bit is set, the Falcon pair *does* perform the write unless it encounters a double-bit error during the read.

If so enabled, single- and double-bit scrub errors are logged, and the PowerPC 60x bus master is notified via interrupt.

Chip Defaults

Based on certain hardware configuration options, some parameters need to be configured by software in the Falcon pair. These parameters include DRAM and ROM/Flash attributes. In order to set up these parameters correctly, software needs some way of knowing about the devices that are being used with the Falcon pair. One way of providing this information is by using the power-up status registers in the Falcon pair. At power-up reset, each Falcon latches the level on its RD0-RD63 signal pins into its power-up status registers. Since the RD signal pins are high impedance during reset, their power-up reset level can be controlled by pullup/pulldown resistors (they are pulled-up internally).

External Register Set

Each chip in the Falcon pair has an external register chip select pin which enables it to talk to an external set of registers. This interface is like the ROM/Flash interface but with less flexibility. It is intended for the system designer to be able to implement general-purpose status/control signals with this external set.

Note Refer to the *Programming Model* section in this chapter for a description of this register set.

CSR Accesses

An important part of the operation of a Falcon pair is that the value written to the internal control registers and SRAM in each of the two chips must be the same at all times. In order to facilitate this, writes to the pair itself are restricted to the upper Falcon only.

When software writes to the upper Falcon, hardware in the two chips shifts this same value into the lower Falcon before the cycle completion is acknowledged. The shifting is done in holding registers such that the actual update of the control register happens on the same CLOCK cycle in both chips. Writes to the upper Falcon can be single-byte or 4-byte. Writes to the lower Falcon are ignored.

This duplicating of writes from upper to lower applies to the Falcon's internal registers only. No duplication is performed for writes to DRAM, ROM/Flash, or the External Register set.

Programming Model

CSR Architecture

3

The Control and Status Register (CSR) set consists of the chip's internal register set and its external register set. The base address of the CSR is hard coded to the address \$FEF80000 (or \$FEF90000 if the SIO pin is low at reset).

Accesses to the CSR are mapped differently depending on whether they are reads or writes. For reads, CSR data read on the upper half of the data bus comes from the upper Falcon while CSR data read on the lower half of the data bus comes from the lower Falcon. Refer to [Figure 3-4](#).

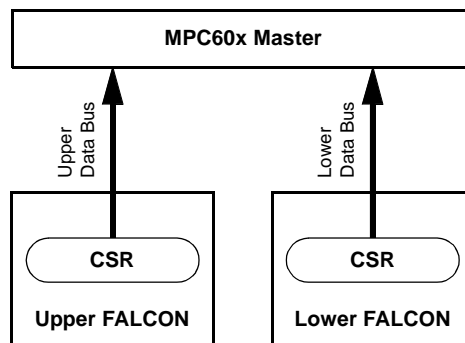


Figure 3-4. Data Path for Reads from the Falcon Internal CSRs

For writes, internal register data written on the upper half of the data bus goes to the upper Falcon and is automatically copied by hardware to the lower Falcon. Internal register data written on the lower half of the data bus does not go to either Falcon in the pair, but the access is terminated normally with TA_. Refer to [Figure 3-5](#).

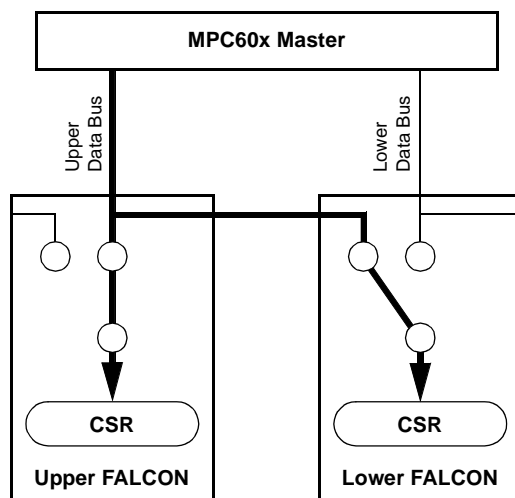


Figure 3-5. Data Path for Writes to the Falcon Internal CSRs

External register data that is written on the upper data bus goes through the upper Falcon, while data that is written on the lower data bus goes through the lower Falcon. Unlike the internal register set, there is no automatic copying of upper data to lower data for the external register set.

CSR read accesses can have a size of 1, 2, 4, or 8 bytes with any alignment. CSR write accesses are restricted to a size of 1 or 4 bytes and they must be aligned. Figure 3-6, Figure 3-7, Figure 3-8, and Figure 3-9 illustrate the memory map for the different kinds of access.

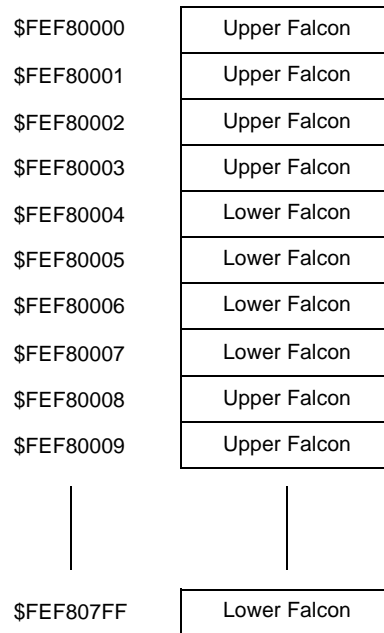


Figure 3-6. Memory Map for Byte Reads to the CSR

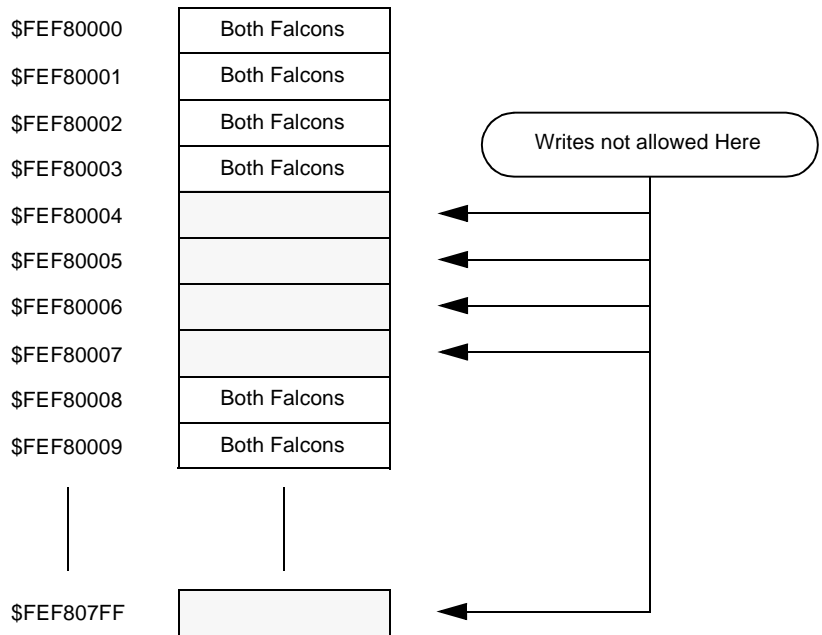


Figure 3-7. Memory Map for Byte Writes to the Internal Register Set

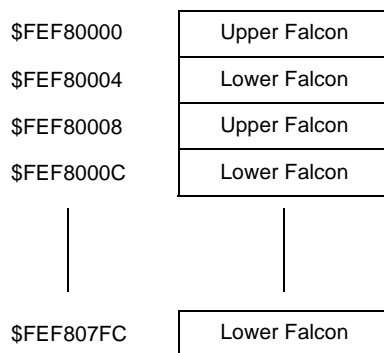
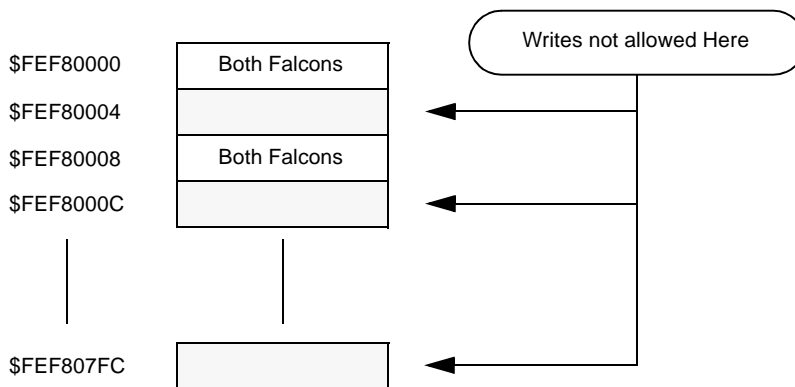


Figure 3-8. Memory Map for 4-Byte Reads to the CSR



1908 9609

Figure 3-9. Memory Map for 4-Byte Writes to the Internal Register Set

Register Summary

Table 3-11 shows a summary of the CSR.

Note The table only shows addresses for accesses to the upper Falcon. To get the addresses for accesses to the lower Falcon, add 4 to the address shown. Since the only way to write to the lower Falcon's internal register set is to duplicate what is written to the upper Falcon, only the addresses shown in the table should be used for writes to them. Writes to the external register set are not duplicated from upper to lower, so writes to them can be via the upper or lower Falcon.

Table 3-11. Register Summary

| BIT # ----> | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | | | | |
|-------------|------------------|------|---|---|------------------------------|------|------------|----------------|-------|---|----|-----------------|----|----|----------|------------|------------------|--------------------------------|-----------------|----|----------------------|------------|----|----|-----------|-----------------|----|-------|------------------------------|----|----|----|-------------------------------|--|--|--|
| FEF80000 | VENDID | | | | | | | | | | | | | | | DEVID | | | | | | | | | | | | | | | | | | | | |
| FEF80008 | | | | | | | | | REVID | | | | | | | | | | | | isa_hole aonly_en | | | | | | | | ram_spd0 ram_fref adis | | | | ram_spd1 ram_spd1 chipu | | | |
| FEF80010 | ram_a_en | | | | RAM A SIZ | | | ram_b_en | | | | RAM B SIZ | | | ram_c_en | | | | RAM C SIZ | | | ram_d_en | | | | RAM D SIZ | | | | | | | | | | |
| FEF80018 | RAM A BASE | | | | | | RAM B BASE | | | | | | | | | RAM C BASE | | | | | | RAM D BASE | | | | | | | | | | | | | | |
| FEF80020 | CLK FREQUENCY | | | | | | | | | | | | | | | | | | | | | por | | | | | | | | | | | | | | |
| FEF80028 | | | | | refdis rwcb derc | | | | | | | | | | | | | mien sien dpjen scien | | | | | | | | | | mcken | | | | | | | | |
| FEF80030 | elog | | | | esch esen embt esbt | | | ERROR_SYNDROME | | | | | | | | | esbkl0 esbkl1 | | | | scof | | | | SBE COUNT | | | | | | | | | | | |
| FEF80038 | ERROR_ADDRESS | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF80040 | scb0 | scb1 | | | | swen | | | | | | | | | | | | rtest0 rtest1 rtest2 | | | | | | | | | | | | | | | | | | |

Table 3-11. Register Summary (Continued)

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|----------------------|-----------------------|-------------|--------|---|----------|-----------|---|-------------|--------|------------|------------|----------|------------|------------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| FEF80048 | | ROW ADDRESS | | | | | | COL ADDRESS | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF80050 | ROM A BASE | | | | rom_a_64 | ROM A SIZ | | | | | | rom_a_lv | rom_a_en | rom_a_we | | | | | | | | | | | | | | | | | | |
| FEF80058 | ROM B BASE | | | | rom_b_64 | ROM B SIZ | | | | | | rom_b_lv | rom_b_en | rom_b_we | | | | | | | | | | | | | | | | | | |
| FEF80060 | | | | | | | | | | rom_a_spd0 | rom_a_spd1 | | rom_b_spd0 | rom_b_spd1 | | | | | | | | | | | | | | | | | | |
| FEF80068 | dpe_log | | DPE_TT | | DPE_DP | | | dpe_ckall | dpe_me | GWDP | | | | | | | | | | | | | | | | | | | | | | |
| FEF80070 | DPE_A | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF80078 | DPE_D | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF80100 | CTR32 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF80400 | PR_STAT1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF80500 | PR_STAT2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| FEF88000 FEF8FFF8 | EXTERNAL REGISTER SET | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| BIT # ----> | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |

Notes

1. All blank bit fields are reserved and read as zeros.
2. All status bits are shown in *italics*.
3. All control bits are shown with underline.
4. All control-and-status bits are shown with *italics* and underline.

Detailed Register Bit Descriptions

The following sections describe the registers and their bits in detail. The possible operations for each bit in the register set are as follows:

- ❑ R The bit is a read only status bit.
- ❑ R/W The bit is readable and writable.
- ❑ R/C The bit is cleared by writing a one to itself.

The possible states of the bits after local and power-up reset are as defined below.

- ❑ P The bit is affected by power-up reset (PURESET_).
- ❑ L The bit is affected by local reset (HRESET_).
- ❑ X The bit is not affected by reset.
- ❑ V The effect of reset on the bit is variable.

Vendor/Device Register

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Address | \$FEF80000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | VENDID | | | | | | | | | | | | | | | | DEVID | | | | | | | | | | | | | | | |
| Operation | READ ONLY | | | | | | | | | | | | | | | | READ ONLY | | | | | | | | | | | | | | | |
| Reset | X | | | | | | | | | | | | | | | | X | | | | | | | | | | | | | | | |

VENDID This read-only register contains the value \$1057. It is the vendor number assigned to Motorola Inc.

DEVID This read-only register contains the value \$4802. It is the device number for the Falcon.

Revision ID/General Control Register

| Address | \$FEF80008 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|-----------|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|------|-----|-----|------|----------|----------|----------|-------|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | | |
| Name | | | | | | | | | REVID | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | adis | ram fref | ram spd0 | ram spd1 | chipu |
| Operation | READ ZERO | | | | | | | | READ ONLY | | | | | | | | R | R | R | R | R | R | R | R | R | R | R/W | R/W | R/W | R/W | R/W | R/W | R/W | R |
| Reset | X | | | | | | | | \$03 | | | | | | | | X | X | X | X | X | X | X | X | X | V/P | 0 PL | X | X | X | 0 P | 1 P | 0 P | V/P |

REVID The REVID bits are hard-wired to indicate the revision level of the Falcon. The value for the first revision is \$01, for the second is \$02, for the third it is \$03.

aonly_en Normally, the Falcon pair responds to address-only cycles only if they fall within the address range of one of its enabled map decoders. When the **aonly_en** bit is set, the Falcon pair also responds to address-only cycles that fall outside of the range of its enabled map decoders provided they are not acknowledged by some other slave within 8 clock periods. **aonly_en** is read-only and reflects the level that was on the CKD4 pin at power-up reset.

isa_hole When it is set, **isa_hole** disables any of the DRAM or ROM/Flash blocks from responding to PowerPC accesses in the range from \$000A0000 to \$000BFFFF. This has the effect of creating a hole in the DRAM memory map for accesses to ISA. When **isa_hole** is cleared, there is no hole created in the memory map.

adis When **adis** is clear, fast page mode operation is used for back-to-back pipelined accesses to the same page within DRAM. When it is set, RAS is cycled between accesses. This bit should normally be cleared unless the Falcon has a problem operating that way.

ram fref Some DRAMs require that they be refreshed at the rate of 7.8 μ s per row rather than the standard 15.6 μ s per row. If any of the DRAM devices require the higher rate, then the **ram fref** bit should be left set, otherwise, it can be cleared.

ram spd0, ram spd1 Together **ram spd0, ram spd1** control DRAM timing used by the Falcon pair. They are encoded as shown in the following table:

Table 3-12. ram spd1, ram spd0 and DRAM Type

| ram spd0, ram spd1 | DRAM Speed | DRAM Type |
|--------------------|------------|-----------|
| %00 | 70ns | Fast Page |
| %01 | 60ns | Fast Page |
| %10 | - | Reserved |
| %11 | 50ns | EDO |

EDO refers to DRAMs that use an output latch on data. Sometimes these parts are referred to as Hyper-Page Mode DRAMs.

To ensure reliable operation, the system should always be configured so that these two bits are encoded to match the slowest devices that are used. Also, if any parts do not support EDO, then these bits must set for Page Mode. The only case in which it is permissible to set **ram spd0, ram spd1** for 50ns, EDO is when all parts are 50ns and all support EDO.

chipu indicates which of the two positions within the Falcon pair is occupied by this chip. When **chipu** is low, this chip is connected to the lower half of the PowerPC 60x data bus and it does not drive TA_ or AACK_. When **chipu** is high, this chip is connected to the upper half of the PowerPC 60x data bus, and it drives TA_ and AACK_. **chipu** reflects the level that was on the ERCS_ pin during power-up reset.

DRAM Attributes Register

| Address | \$FEF80010 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|------------|------------|------------|----------|---|---|---|---|------------|------------|------------|----------|---|---|---|---|------------|------------|------------|----------|---|---|---|---|------------|------------|------------|
| Bit | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Name | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 0 | ram a en | 0 | 0 | 0 | 0 | ram a siz0 | ram a siz1 | ram a siz2 | ram b en | 0 | 0 | 0 | 0 | ram b siz0 | ram b siz1 | ram b siz2 | ram c en | 0 | 0 | 0 | 0 | ram c siz0 | ram c siz1 | ram c siz2 | ram d en | 0 | 0 | 0 | 0 | ram d siz0 | ram d siz1 | ram d siz2 |
| | R/W | R | R | R | R | R/W | R/W | R/W | R/W | R | R | R | R | R/W | R/W | R/W | R/W | R | R | R | R | R/W | R/W | R/W | R/W | R | R | R | R | R/W | R/W | R/W |
| | 0 PL | X | X | X | X | 0 P | 0 P | 0 P | 0 PL | X | X | X | X | 0 P | 0 P | 0 P | 0 PL | X | X | X | X | 0 P | 0 P | 0 P | 0 PL | X | X | X | X | 0 P | 0 P | 0 P |



Warning

To satisfy DRAM component requirements before the memory is used at start-up, software must always wait at least 500µs between the initial setting of a bank's size bits, to a non-zero value, and the first accessing of that bank. These settings are in the DRAM Attributes Register (offset \$FEF80010). The delay is intended to make sure that the bank has been refreshed at least 8 times before it is used. The 500µs is sufficient as long as the CLK Frequency Register (offset \$FEF80020) is within a factor of 2 of matching the actual 60x clock frequency.

ram a/b/c/d en enables accesses to the corresponding block of DRAM when set, and disables them when cleared.

ram a/b/c/d siz0-2 These control bits define the size of their corresponding block of DRAM. [Table 3-13](#) shows the block configuration assumed by the Falcon pair for each value of **ram siz0-ram siz2**.

Table 3-13. Block_A/B/C/D Configurations

| ram a/b/c/d siz0-2 | Block SIZE | Devices Used | | | Technology | Comments |
|-----------------------|---------------|--------------|---|----------|------------|-------------------|
| %000 | 0MB | - | - | - | - | Block Not Present |
| %001 | 16MB | 36 | - | 1Mx4's | 4MB | |
| | | 8 | - | 1Mx18's | 16MB | |
| | | 4 | - | 1Mx36's | 4MB/1MB | SIMM/DIMM |
| %010 | 32MB | 18 | - | 2Mx8's | 16MB | |
| %011 | 64MB | 144 | - | 4Mx1's | 4MB | |
| | | 36 | - | 4Mx4's | 16MB | |
| | | 8 | - | 4Mx18's | 64MB | |
| | | 4 | - | 4Mx36's | 16MB/4MB | SIMM/DIMM |
| %100 | 128MB | 18 | - | 8Mx8's | 64MB | |
| %101 | 256MB | 144 | - | 16Mx1's | 16MB | |
| | | 36 | - | 16Mx4's | 64MB | |
| | | 4 | - | 16Mx36's | 64MB/16MB | SIMM/DIMM |
| %110 | 1024MB | 144 | - | 64Mx1's | 64MB | |
| %111 | 0MB | - | - | - | - | Reserved |

Note It is important that all of the **ram a/b/c/d siz0-2** bits be set to accurately match the actual size of their corresponding blocks. This includes clearing them to %000 if their corresponding blocks are not present. Failure to do so will cause problems with addressing and with scrub error logging.

DRAM Base Register

| Address | \$FEF80018 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|------------|---|----|----|----|----|----|----|------------|----|----|----|----|----|----|----|------------|----|----|----|----|----|----|----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | RAM A BASE | | | | | | | | RAM B BASE | | | | | | | | RAM C BASE | | | | | | | | RAM D BASE | | | | | | | |
| Operation | READ/WRITE | | | | | | | | READ/WRITE | | | | | | | | READ/WRITE | | | | | | | | READ/WRITE | | | | | | | |
| Reset | 0 PL | | | | | | | | 0 PL | | | | | | | | 0 PL | | | | | | | | 0 PL | | | | | | | |

RAM A/B/C/D BASE These control bits define the base address for their block's DRAM. RAM A/B/C/D BASE bits 0-7/8-15/16-23/24-31 correspond to PowerPC 60x address bits 0 - 7. For larger DRAM sizes, the lower significant bits of A/B/C/D BASE are ignored. This means that the block's base address will always appear at an even multiple of its size.

Notes Bit 0 is MSB.

The combination of **RAM_X_BASE** and **ram_x_siz** should never be programmed such that DRAM responds at the same address as the CSR, ROM/Flash, External Register Set, or any other slave on the PowerPC bus.

CLK Frequency Register

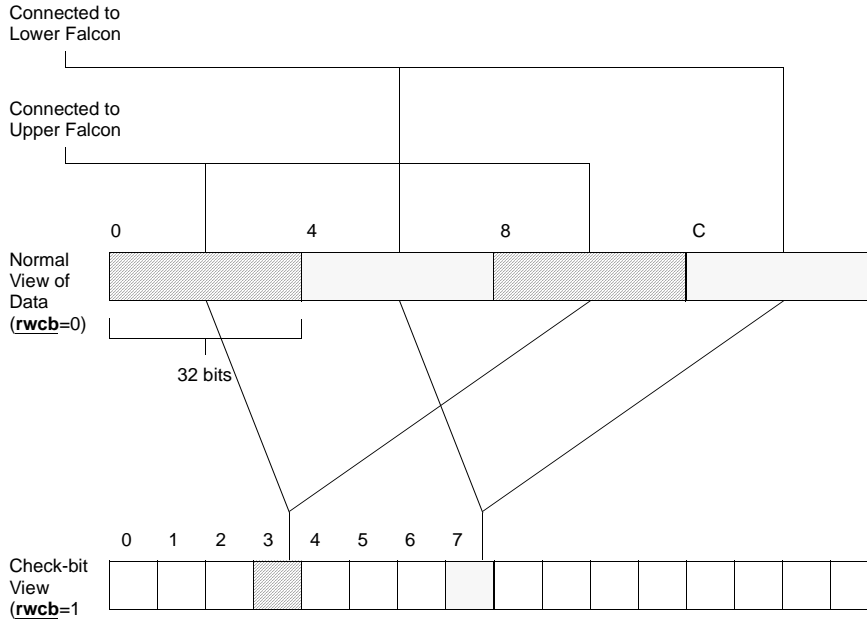
| Address | \$FEF80020 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------------|---|---|---|---|---|---|---|-----------|---|----|----|----|----|----|----|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | CLK FREQUENCY | | | | | | | | | | | | | | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | 0 | por |
| Operation | READ/WRITE | | | | | | | | READ ZERO | | | | | | | | READ ZERO | | | | | | | | R | R | R | R | R | R | R | R/C |
| Reset | 42 P | | | | | | | | X | | | | | | | | X | | | | | | | | X | X | X | X | X | X | X | 1 P |

CLK FREQUENCY These bits should be programmed with the hexadecimal value of the operating CLOCK frequency in MHz (that is, \$42 for 66 MHz). When these bits are programmed this way, the chip's prescale counter produces a 1 MHz

por is set by the occurrence of power up reset. It is cleared by writing a one to it. Writing a 0 to it has no effect.

| Address | \$FEF80028 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|--------|------|------|---|---|----|----|-------|-------|------|------|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|-------|--|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | |
| Name | 0 | 0 | 0 | 0 | 0 | reidis | rwcb | dere | 0 | 0 | 0 | 0 | scien | dpjen | sien | njen | | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | mcken | |
| Operation | R | R | R | R | R | R/W | R/W | R/W | R | R | R | R | R/W | R/W | R/W | R/W | READ ZERO | | | | | | | | | | R | R | R | R | R | R/W | |
| Reset | X | X | X | X | X | 0 PL | 0 PL | 1 PL | X | X | X | X | 0 PL | 0 PL | 0 PL | 0 PL | X | | | | | | | | | | | X | X | X | X | 0 PL | |

rwcb When set, **rwcb** causes reads and writes to DRAM from the PowerPC 60x bus to access check-bit data rather than normal data. The data path used for this mode is DH24-31 for check-bit data controlled by the upper Falcon, and DL24-31 for check-bit data controlled by the lower Falcon. Each 8-bit check-bit location services 64 bits of normal data. The 64 bits of data are all within the same Falcon. Each Falcon provides every other 32 bits of data in the normal mode. The following figure shows the relationship between normal data and check-bit data.



11707.00 9701

So, for example, the check-bits that correspond to the 64 bits of data found in normal mode (**rwcb**=0) at \$00001000-\$00001003 and \$00001008-\$0000100b are written and read in check-bit mode (**rwcb**=1) at location \$00001003.

Note If test software wishes to force a single-bit error to a location using the **rwcb** function, the scrubber may correct the location before the test software gets a chance to check for the single-bit error. This can be avoided by disabling scrub writes.

Also note that writing bad check-bits can set the **elog** bit in the Error Logger Register.

The writing of check-bits causes the Falcon to perform a read-modify-write to DRAM. If the location to which check-bits are being written has a single or double-bit error, data in the location may be altered by the write check-bits operation.

To prevent this, it is recommended that the **derc** bit also be set while the **rwcb** bit is set. A possible sequence for performing read-write check-bits is as follows:

1. Disable scrub writes by clearing the **swen** bit if it is set.
2. Make sure software is not using DRAM at this point, because while **rwcb** is set, DRAM will not function as normal memory.
3. Set the **derc** and **rwcb** bits in the Data Control register.
4. Perform the desired read and/or write check-bit operations.
5. Clear the **derc** and **rwcb** bits in the Data Control register.
6. Perform the desired testing related to the location/locations that have had their check-bits altered.
7. Enable scrub writes by setting the **swen** bit if it was set before.

derc Setting **derc** to one alters Falcon pair operation as follows:

1. During reads, data is presented to the PowerPC 60x data bus uncorrected from the DRAM array.
2. During single-beat writes, data is written without correcting single-bit errors that may occur on the read portion of the read-modify-write. Check-bits are generated for the data being written.
3. During single-beat writes, the write portion of the read-modify-write happens regardless of whether there is a multiple-bit error during the read portion. No correction of data is attempted. Check-bits are generated for the data being written.
4. During refresh/scrub cycles, if **swen** is set, a read-write to DRAM happens with no attempt to correct data bits. Check-bits are generated for the data being written.

derc is useful for initializing DRAM after power-up and for testing DRAM, but it should be cleared during normal system operation.

scien When **scien** is set, the rolling over of the **SBE COUNT** register causes the INT_ signal pin to pulse true.

dpien When **dpien** is set, the logging of a 60x data parity error causes the INT_ signal pin to pulse true.

sien When **sien** is set, the logging of a single-bit error causes the INT_ signal pin to pulse true.

mien When **mien** is set, the logging of a non-correctable error causes the INT_ signal pin to pulse true.

mcken When **mcken** is set, the detection of a multiple-bit error during a PowerPC read or write causes the Falcon to pulse its machine check interrupt request pin (MCP_) true. When **mcken** is cleared, the Falcon does not ever assert its MCP_ pin.

The Falcon never asserts its MCP_ pin in response to a multiple-bit error detected during a scrub cycle.



Caution

The INT_ and MCP_ pins are the only non-pollled notification that a multiple-bit error has occurred. The Falcon pair does not assert TEA as a result of a multiple bit error. In fact, the Falcon pair does not have a TEA_ signal pin and it assumes that the system does not implement TEA_.

Error Logger Register

| Address | \$FEF80030 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|------|------|------|------|----------------|---|----|----|----|----|----|----|----|----|-------|-------|----|----|-----|------------|-----------|----|----|----|----|----|----|----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | elog | 0 | 0 | 0 | escb | esen | embt | esbt | ERROR_SYNDROME | | | | | | | | 0 | 0 | eblk0 | eblk1 | 0 | 0 | 0 | scnf | SBE_COUNT | | | | | | | |
| Operation | R/C | R | R | R | R | R/W | R | R | READ ONLY | | | | | | | | R | R | R | R | R | R | R/C | READ/WRITE | | | | | | | | |
| Reset | 0P | X | X | X | X | 0PL | X | X | X | | | | | | | | X | X | X | X | X | X | 0P | 0 | | | | | | | | |

The Error Logger and Error Address Registers behave the same as the other registers, in that data written to the upper Falcon is automatically duplicated in the lower Falcon. They also behave the same as the other registers, in that status read from the upper Falcon pertains to the upper Falcon, and status read from the lower Falcon pertains to the lower Falcon. Unlike most of the other registers, however, it is normal for this status to differ between the two. This is due to the fact that each Falcon is connected to its own set of DRAMs. The upper Falcon can log an error during a cycle and the local Falcon not, or vice-versa. Or they can both log an error during the same cycle and have the attributes of the errors differ.

Because of the above, software needs to monitor both the upper and lower Falcon's Error Logger and Error Address Registers. This includes checking the **elog** bit from the upper Falcon and from the lower Falcon. When the upper Falcon logs an error, it updates its attribute bits (**escb**, **embt**, **esbt**, **ERROR_SYNDROME**, **eblk0**, **eblk1**, and **ERROR_ADDRESS**) to match the results of the read cycle for its portion of the DRAM array. When the lower Falcon logs an error, it updates its attribute bits to match the results of the read cycle for its portion of the DRAM array.

While the logging of errors by one Falcon in a pair does not affect the logging of errors by the other, writing to the Error Logger Register control bits affects both Falcons. This is of particular interest as regards the **elog** bit. Writing a one to the **elog** bit clears the **elog** bit for both the upper and

lower Falcons. Because of this, software needs to check the status of both upper and lower Error Logger and Error Address Registers before it clears the **elog** bits. Otherwise, it could miss a logged error.

elog When set, **elog** indicates that a single- or a multiple-bit error has been logged by its Falcon. If **elog** is set by a multiple-bit error, then no more errors will be logged until software clears it. If **elog** is set by a single-bit error, then no more single-bit errors will be logged until software clears it; however if **elog** is set by a single-bit error and a multiple-bit error occurs, the multiple-bit error will be logged and the single-bit error information overwritten. **elog** can only be set by the logging of an error and cleared by the writing of a one to itself or by power-up reset.

escb indicates the entity that was accessing DRAM at the last logging of a single- or multiple-bit error by its Falcon. If **escb** is 1, it indicates that the scrubber was accessing DRAM. If **escb** is 0, it indicates that the PowerPC 60x bus master was accessing DRAM.

esen When set, **esen** allows errors that occur during scrubs to be logged. When cleared, **esen** does not allow errors that occur during scrubs to be logged.

embt is set by the logging of a multiple-bit error in its Falcon. It is cleared by the logging of a single-bit error in its Falcon. It is undefined after power-up reset. A Falcon's syndrome code is meaningless if its **embt** bit is set.

esbt is set by the logging of a single-bit error in its Falcon. It is cleared by the logging of a multiple-bit error in its Falcon. When a Falcon logs a single-bit error, its syndrome code indicates which bit was in error. (Refer to the section on *ECC Codes*.)

ERROR_SYNDROME reflects the syndrome value at the last logging of an error by its Falcon. This eight-bit code indicates the position of the data error. When all the bits are zero, there was no error. Note that if the logged error was non-correctable, then these bits are meaningless. Refer to the section on *ECC Codes* for a decoding of the syndromes.

esblk0,esblk1 Together these two bits indicate which block of DRAM was being accessed when their Falcon logged a scrub error. **esblk0,esblk1** are 0,0 for Block A; 0,1 for Block B; 1,0 for Block C; and 1,1 for Block D.

scof is set by its **SBE COUNT** register rolling over from \$FF to \$00. It is cleared by software writing a 1 to it.

SBE COUNT This register keeps track of the number of single-bit errors that have occurred since it was last cleared. It counts up by one each time its half of the Falcon pair detects a single-bit error (independent of the state of the **elog** bit). It is cleared by power-up reset and by software writing all zeros to it. When **SBE COUNT** rolls over from \$FF to \$00, its Falcon sets the **scof** bit. It also pulses the **INT_** signal low if the **scien** bit is set.

Error_Address Register

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|---------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Address | \$FEF80038 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | ERROR_ADDRESS | | | | | | | | | | | | | | | | | | | | | | | | | | | 0 | 0 | 0 | 0 | |
| Operation | READ ONLY | | | | | | | | | | | | | | | | | | | | | | | | | | | R | R | R | R | |
| Reset | X P | | | | | | | | | | | | | | | | | | | | | | | | | | | X | X | X | X | |

ERROR_ADDRESS These bits reflect the value that corresponds to bits 0-27 of the PowerPC 60x address bus when their Falcon last logged an error during a PowerPC access to DRAM. They reflect the value of the DRAM row and column addresses if the error was logged during a scrub cycle. In this case, bits 2-14 correspond to row address signals 0-12 respectively and bits 15-27 correspond to column address signals 0-12 respectively. Refer to [Table 3-20 on page 3-61](#). It shows how PowerPC addresses correspond to DRAM row and column addresses.

Scrub/Refresh Register

| Address | \$FEF80040 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|------|---|---|---|---|---|------|---|---|----|----|----|--------|--------|--------|-----------|----|----|----|----|----|-----------|----|----|----|----|----|----|----|----|----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | scb0 | scb1 | 0 | 0 | 0 | 0 | 0 | swen | 0 | 0 | 0 | 0 | 0 | rtest0 | rtest1 | rtest2 | | | | | | | | | | | | | | | | |
| Operation | R | R | R | R | R | R | R | R/W | R | R | R | R | R | R/W | R/W | R/W | READ ZERO | | | | | | READ ZERO | | | | | | | | | |
| Reset | 0P | 0P | X | X | X | X | X | 0P | X | X | X | X | X | 0P | 0P | 0P | X | | | | | | X | | | | | | | | | |

scb0,scb1 These bits increment every time the scrubber completes a scrub of the entire DRAM. When these bits reach binary 11, they roll over to binary 00 and continue. These bits are cleared by power-up reset.

swen When set, **swen** allows the scrubber to perform write cycles. When cleared, **swen** prevents scrubber writes.

rtest0,1,2 The **rtest** bits enable certain refresh counter test modes. The table below shows their encodings.

Note These test modes are not intended to be used once the chip is in a system.

Table 3-14. rtest encodings

| rtest0,rtest1,rtest2 | Test Mode selected |
|----------------------|-----------------------------------|
| %000 | Normal Counter Operation |
| %001 | RA counts at 16x |
| %010 | RA counts at 256x |
| %011 | RA is always at roll value for CA |
| %100 | CA counts at 16x |
| %101 | CA counts at 256x |
| %110 | reserved |
| %111 | reserved |

Refresh/Scrub Address Register

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|-------------|---|---|---|---|---|---|----|----|----|----|----|----|----|----|-------------|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Address | \$FEF80048 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | 0 | 0 | 0 | ROW ADDRESS | | | | | | | | | | | | 0 | 0 | 0 | COL ADDRESS | | | | | | | | | | | | | |
| Operation | R | R | R | READ/WRITE | | | | | | | | | | | | R | R | R | READ/WRITE | | | | | | | | | | | | | |
| Reset | X | X | X | 0 P | | | | | | | | | | | | X | X | X | 0 P | | | | | | | | | | | | | |

ROW ADDRESS These bits form the row address counter used by the refresher/scrubber for all blocks of DRAM. The row address counter increments by one after each refresh/scrub cycle. When it reaches all 1s, it rolls back over to all 0s and continues counting. **ROW ADDRESS** is readable and writable for test purposes.

Note Within each block, the most significant bits of **ROW ADDRESS** are used only when their DRAM devices are large enough to require them.

COL ADDRESS These bits form the column address counter used by the refresher/scrubber for all blocks of DRAM. The counter increments by one every eighth time the **ROW ADDRESS** rolls over. **COL ADDRESS** is readable and writable for test purposes.

Note Within each block, the most significant bits of **COL ADDRESS** are only used when their DRAM devices are large enough to require them.

ROM A Base/Size Register

| Address | \$FEF80050 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|----|----------|------------|------------|------------|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----------|----------|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | ROM A BASE | | | | | | | | | | | rom_a_64 | rom_a_siz0 | rom_a_siz1 | rom_a_siz2 | | | | | | | | | | | | | | | | rom_a_en | rom_a_we |
| Operation | READ/WRITE | | | | | | | | | | | R | R/W | R/W | R/W | READ ZERO | | | | | | | | | | | | | | | R/W | R/W |
| Reset | \$FF0 PL | | | | | | | | | | | V P | 0 PL | 0 PL | 0 PL | X | | | | | | | | | | | | | | | 0 PL | 0 PL |

ROM A BASE These control bits define the base address for ROM/Flash Block A. **ROM A BASE** bits 0-11 correspond to PowerPC 60x address bits 0 - 11 respectively. For larger ROM/Flash sizes, the lower significant bits of **ROM A BASE** are ignored. This means that the block's base address will always appear at an even multiple of its size. **ROM A BASE** is initialized to \$FF0 at power-up or local bus reset.

Note In addition to the programmed address, the first 1MB of Block A also appears at \$FFF00000 - \$FFFFFFFF if the **rom_a_rv** bit is set and the **rom_b_rv** bit is cleared.

Also note that the combination of **ROM_A_BASE** and **rom_a_siz** should never be programmed such that ROM/Flash Block A responds at the same address as the CSR, DRAM, External Register Set, or any other slave on the PowerPC bus.

rom_a_64 indicates the width of ROM/Flash device/devices being used for Block A. When **rom_a_64** is cleared, Block A is 16 bits wide, where each Falcon interfaces to 8 bits. When **rom_a_64** is set, Block A is 64 bits wide, where each Falcon interfaces to 32 bits. **rom_a_64** matches the value that was on the CKD2 pin at power-up reset. It cannot be changed by software.

rom a siz The **rom a siz** control bits are the size of ROM/Flash for Block A. They are encoded as shown in Table 3-15.

Table 3-15. ROM/Flash Block A Size Encoding

| rom a siz | BLOCK SIZE |
|-----------|------------|
| %000 | 1MB |
| %001 | 2MB |
| %010 | 4MB |
| %011 | 8MB |
| %100 | 16MB |
| %101 | 32MB |
| %110 | 64MB |
| %111 | Reserved |

rom_a_rv and **rom_b_rv** determine which if either of Blocks A and B is the source of reset vectors or any other access in the range \$FFF00000 - \$FFFFFFFF as shown in the table below.

Table 3-16. rom_a_rv and rom_b_rv Encoding

| rom_a_rv | rom_b_rv | Result |
|----------|----------|-----------------------------------------------|
| 0 | 0 | Neither Block is the source of reset vectors. |
| 0 | 1 | Block B is the source of reset vectors. |
| 1 | 0 | Block A is the source of reset vectors. |
| 1 | 1 | Block B is the source of reset vectors. |

rom_a_rv is initialized at power-up reset to match the value on the CKD0 pin.

rom a en When **rom a en** is set, accesses to Block A ROM/Flash in the address range selected by **ROM A BASE** are enabled. When **rom a en** is cleared, they are disabled.

rom a we When **rom a we** is set, writes to Block A ROM/Flash are enabled. When **rom a we** is cleared, they are disabled.

Note If **rom_a_64** is cleared, only 1-byte writes are allowed. If **rom_a_64** is set, only 4-byte writes are allowed. The Falcon ignores other writes. If a valid write is attempted and **rom_a_we** is cleared, the write does not happen but the cycle is terminated normally. Refer to the following table for details of ROM/Flash accesses.

Table 3-17. Read/Write to ROM/Flash

| Cycle | Transfer Size | Alignment | rom_x_64 | rom_x_we | Falcon Response |
|-------|---------------------|------------|----------|----------|-----------------------------------------------|
| write | 1-byte | X | 0 | 0 | Normal termination, but no write to ROM/Flash |
| write | 1-byte | X | 0 | 1 | Normal termination, write occurs to ROM/Flash |
| write | 1-byte | X | 1 | X | No Response |
| write | 4-byte | Misaligned | X | X | No Response |
| write | 4-byte | Aligned | 0 | X | No Response |
| write | 4-byte | Aligned | 1 | 0 | Normal termination, but no write to ROM/Flash |
| write | 4-byte | Aligned | 1 | 1 | Normal termination, write occurs to ROM/Flash |
| write | 2,3,5,6,7,8,32-byte | X | X | X | No Response |
| read | X | X | X | X | Normal Termination |

ROM B Base/Size Register

| Address | \$FEF80058 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|----|----------|------------|------------|------------|-----------|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|---|----------|----------|----------|-----|------|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | | | | | | |
| Name | ROM B BASE | | | | | | | | | | | rom b 64 | rom b siz0 | rom b siz1 | rom b siz2 | | | | | | | | | | | | | | | | | 0 | 0 | rom_b_rv | rom_b en | rom b we | | |
| Operation | READ/WRITE | | | | | | | | | | | R | R/W | R/W | R/W | READ ZERO | | | | | | | | | | | | | | | | R | R | R | R | R | R/W | R/W |
| Reset | \$FF4 PL | | | | | | | | | | | V P | 0 PL | 0 PL | 0 PL | X | | | | | | | | | | | | | | | | X | X | X | X | X | V P | 0 PL |

ROM B BASE These control bits define the base address for ROM/Flash Block B. **ROM B BASE** bits 0-11 correspond to PowerPC 60x address bits 0 - 11 respectively. For larger ROM/Flash sizes, the lower significant bits of **ROM B BASE** are ignored. This means that the block's base address will always appear at an even multiple of its size. **ROM B BASE** is initialized to \$FF4 at power-up or local bus reset.

Note In addition to the programmed address, the first 1Mbyte of Block B also appears at \$FFF00000 - \$FFFFFFFF if the **rom_b_rv** bit is set.

Also note that the combination of **ROM_B_BASE** and **rom_b_siz** should never be programmed such that ROM/Flash Block B responds at the same address as the CSR, DRAM, External Register Set, or any other slave on the PowerPC bus.

rom_b_64 indicates the width of ROM/Flash device/devices being used for Block B. When **rom_b_64** is cleared, Block B is 16 bits wide, where each Falcon interfaces to 8 bits. When **rom_b_64** is set, Block B is 64 bits wide, where each Falcon interfaces to 32 bits. **rom_b_64** matches the inverse of the value that was on the CKD3 pin at power-up reset. It cannot be changed by software.

rom_b siz The **rom_b siz** control bits are the size of ROM/Flash for Block B. They are encoded as shown in the following table.

| rom_b siz | BLOCK SIZE |
|------------------|-------------------|
| %000 | 1MB |
| %001 | 2MB |
| %010 | 4MB |
| %011 | 8MB |
| %100 | 16MB |
| %101 | 32MB |
| %110 | 64MB |
| %111 | Reserved |

rom_b_rv and **rom_a_rv** determine which if either of Blocks A and B is the source of reset vectors or any other access in the range \$FFF00000 - \$FFFFFFFF as shown in [Table 3-16 on page 3-48](#).

rom_b_rv is initialized at power-up reset to match the inverse of the value on the CKD1 pin.

rom_b_en When **rom_b_en** is set, accesses to Block B ROM/Flash in the address range selected by **ROM B BASE** are enabled. When **rom_b_en** is cleared, they are disabled.

rom_b_we When **rom_b_we** is set, writes to Block B ROM/Flash are enabled. When **rom_b_we** is cleared, they are disabled. Refer back to [Table 3-17 on page 3-49](#) for more details.

ROM Speed Control Register

| Address | \$FEF80060 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|-----------|---|----|----|----|----|----|----|-----------|----|----|----|----|----|----|----|----|----|------------|------------|------|------------|------------|----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | | | | | | | | | | | | | | | | | | | | | | | | | 0 | 0 | rom a spd0 | rom a spd1 | 0 | rom b spd0 | rom b spd1 | |
| Operation | READ ZERO | | | | | | | | READ ZERO | | | | | | | | READ ZERO | | | | | | | | R | R | R/W | R/W | R | R/W | R/W | |
| Reset | X | | | | | | | | X | | | | | | | | X | | | | | | | | X | X | 0 PL | X | 0 PL | 0 PL | 0 PL | |

rom_a_spd0,1 determine the access timing used for ROM/Flash Block A. The encodings of these bits are as follows.

Table 3-18. Rom Speed Bit Encodings

| rom_a/b_spd0,1 | ROM Block A/B Access Time |
|----------------|---------------------------|
| %00 | 180ns |
| %01 | 120ns |
| %10 | 75ns |
| %11 | 45ns |

rom_b_spd0,1 determine the access timing used for ROM/Flash Block B. See table above.

Data Parity Error Logger Register

| Address | \$FEF80068 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---------|---------|---------|---------|---------|---|---|----|----|---------|---------|---------|---------|----|----|----|----|----|----|-----------|--------|------------|----|----|----|----|----|----|----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | dpelog | 0 | | dpe tt0 | dpe tt1 | dpe tt2 | dpe tt3 | dpe tt4 | 0 | 0 | 0 | 0 | dpe dp0 | dpe dp1 | dpe dp2 | dpe dp3 | 0 | 0 | 0 | 0 | 0 | 0 | dpe ckall | dpe me | GWDG | | | | | | | |
| Operation | R/C | R | | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R | R/W | R/W | READ/WRITE | | | | | | | |
| Reset | 0 P | | X | 0 P | 0 P | 0 P | 0 P | 0 P | X | X | X | X | 0 P | 0 P | 0 P | 0 P | X | X | X | X | X | X | 0 PL | 0 PL | 0 PL | | | | | | | |

The Data Parity Error Logger and Data Parity Error Address and Data registers are much like the Error Logger and Error Address registers in that it is normal for status to differ between the upper and lower Falcons. This is due to the fact that each Falcon is connected to its own half of the 60x data bus and data parity bus. The upper Falcon can log parity error during a cycle and the lower Falcon not, or vice-versa. Or they can both log a parity error during the same cycle and have the attributes of the errors differ.

Because of the above, software needs to monitor both the upper and lower Falcon's Data Parity Error Logger, Data Parity Error Address and Data Parity Error Data Registers. This includes checking the **dpe log** bit from the upper Falcon and from the lower Falcon. When the upper Falcon logs an error, it updates its attribute bits (**dpe0-3**, **DATA PARITY ERROR ADDRESS**, and **DATA PARITY ERROR DATA**) to match the results of the read cycle for the upper PowerPC 60x data bus. When the lower Falcon logs an error, it updates its attribute bits to match the results of the read cycle for the lower 60x data bus.

While the logging of data parity errors by one Falcon in a pair does not affect the logging of data parity errors by the other, writing to the Data Parity Error Logger control bits does affect both Falcons. This is of particular interest as regards the **dpe log** bit. Writing a one to the **dpe log** bit clears the **dpe log** bit for both the upper and lower Falcons. Because of this,

software needs to check the status of both upper and lower Data Parity Error Logger, Address, and Data registers before it clears the **dpelog** bits. Otherwise, it could miss a logged error.

dpelog is set when a parity error occurs on the Falcon's half of the 60x data bus during any 60x data cycle. It is cleared by writing a one to the upper Falcon's **dpelog** bit or by power-up reset.

dpe_tt is the value that was on the tt0-tt4 signals when the Falcon's **dpelog** bit was set.

dpe_dp is the value that was on the Falcon's dp0-dp3 signals when its **dpelog** bit was set.

When **dpe_ckall** is set, the Falcon checks data parity on all cycles in which ta_ is asserted. When **dpe_ckall** is cleared, the Falcon checks data parity on cycles when ta_ is asserted only during writes to the Falcon Pair.

Note The Falcon does not check parity during cycles in which there is a qualified artry_ at the same time as the ta_

When **dpe_me** is set, the transition of a Falcon's **dpelog** bit from false to true causes it to pulse its machine check interrupt request pin (MCP_) true. When **dpe_me** is cleared, the Falcon does not assert its MCP_ pin based on its **dpelog** bit.

The **GWDP0-7** bits are used to invert the value that is driven onto DP0-DP7 respectively during reads to the Falcon pair. This allows test software to generate wrong (even) parity on selected byte lanes. For example, to create a parity error on the PowerPC 60x DL24-DL31 and DP7 signals during Falcon reads, software should set **GWDP7**. Note that while the value on **GWDP** is duplicated on both the upper and lower Falcons, **GWDP0-3** affect only the upper Falcon and **GWDP4-7** affect only the lower Falcon.

Data Parity Error Address Register

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Address | \$FEF80070 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | DPE_A | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | READ ONLY | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | 0 P | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

DPE_A is the address of the last 60x data bus parity error that was logged by a Falcon. It is updated only when the Falcon's **dpelog** bit goes from 0 to 1.

Data Parity Error Data Register

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Address | \$FEF80078 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | DPE_D | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | READ ONLY | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | 0 P | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

DPE_D is the value on the Falcon's half of the PowerPC 60x data bus at the time it last logged a PowerPC 60x data bus parity error. **DPE_D** updates only when the associated **dpelog** bit goes from 0 to 1.

32-Bit Counter

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Address | \$FEF80100 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | CTR32 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | READ/WRITE | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | 0 PL | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

CTR32 is a 32-bit, free-running counter that increments once per microsecond if the CLK_FREQUENCY register has been programmed properly. Notice that **CTR32** is cleared by power-up and local reset. It does not exist in Revision 1 of Falcon.

Note When the system clock is a fractional frequency, such as 66.67 MHz, **CTR32** will count at a fractional amount faster or slower than 1 MHz, depending on the programming of the CLK_FREQUENCY Register.

Power-Up Reset Status Register 1

| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Address | \$FEF80400 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | PR_STAT1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | READ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | V P | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

PR_STAT1 (power-up reset status) reflects the value that was on the RD0-RD31 signal pins at power-up reset. This register is read-only.

Note For descriptions of how this register is used in the PowerPC series boards, refer to the [System Configuration Register \(SYSCR\)](#) on page 1-13 and the [Memory Configuration Register \(MEMCR\)](#) on page 1-15.

Power-Up Reset Status Register 2

| Address | \$FEF80500 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | PR_STAT2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | READ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | V P | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

PR_STAT2 (power-up reset status) reflects the value that was on the RD32-RD63 signal pins at power-up reset. This register is read-only.

External Register Set

| Address | \$FEF88000 - \$FEF8FFF8 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------|-------------------------|---|---|---|---|---|---|---|---|---|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|----|
| Bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |
| Name | EXTERNAL REGISTER SET | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Operation | READ/WRITE | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Reset | X PL | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

The **EXTERNAL REGISTER SET** is user provided and is external to the Falcon pair. The Falcon pair provides a static RAM style interface for the external registers. The external registers can be SRAM, ROM, Flash or some other user device. There can be one device connected to either the upper or lower Falcon, or there can be two devices with one connected to each Falcon. The devices can be 8, 16, or 32 bits wide. The data path to the external devices is via the RD32-RD63 pins. Reads to the external devices can be any size except burst.

Note If the devices are less than 32 bits wide, reads to unused data lanes will yield undefined data. Note that writes are restricted to one or 4-byte length only. 4-byte writes can be used for any size device, data should be placed on the correct portion of the data bus so that valid data is written to the device. Data duplication is turned off for the **EXTERNAL REGISTER SET** so writes can be to either the upper Falcon, or to the lower Falcon.

Note For descriptions of how these registers are used in the PowerPC series boards, refer to the [Processor 0 External Cache Control Register \(P0XCCR\)](#) on page 1-18 and the [CPU Control Register](#) on page 1-19.

Software Considerations

This section contains information that will be useful in programming a system that uses the Falcon pair.

Parity Checking on the PowerPC Bus

The Falcon does not generate parity on the PowerPC address bus. Because of this, the appropriate registers in the MPC60x should not be programmed to enable parity checking for the address bus.

The Falcon does generate parity on the PowerPC data bus. The appropriate registers in the MPC60x can be programmed to enable parity checking for the data bus.

Programming ROM/Flash Devices

Those who program devices to be controlled by the Falcon should make note of the address mapping that is shown in [Table 3-9 on page 3-19](#) and in [Table 3-10 on page 3-20](#). For example, when using 8-bit devices, the code will be split so that every other 4-byte segment goes in each device.

Writing to the Control Registers

Software should not change control register bits that affect DRAM operation while DRAM is being accessed. Because of pipelining, software should always make sure that the two accesses before and after the updating of critical bits are not DRAM accesses. A possible scenario for trouble would be to execute code out of DRAM while updating the critical DRAM control register bits. The preferred method is to be executing code out of ROM/Flash and avoiding DRAM accesses while updating these bits.

Since software has no way of controlling refresh accesses to DRAM, the hardware is designed so that updating control bits coincidentally with refreshes is not a problem. An exception to this is the **ROW_ADDRESS** and **COL_ADDRESS** bits. It is not intended that software write to these bits anyway.

As with DRAM, software should not change control register bits that affect ROM/Flash while the affected Block is being accessed. This generally means that the ROM/Flash size, base address, enable, write enable, etcetera, are changed only while executing initially in the reset vector area (\$FFF00000 - \$FFFFFFF).



To satisfy DRAM component requirements before the memory is used at start-up, software must always wait at least 500 μ s between the initial setting of a bank's size bits, to a non-zero value, and the first accessing of that bank. These settings are in the DRAM Attributes Register (offset \$FEF80010). The delay is intended to make sure that the bank has been refreshed at least 8 times before it is used. The 500 μ s is sufficient as long as the CLK Frequency Register (offset \$FEF80020) is within a factor of 2 of matching the actual PowerPC 60x clock frequency.

Sizing DRAM

The following routine can be used to size DRAM for the Falcon.

Initialize the Falcon control register bits to a known state as follows:

1. Clear the **isa_hole** bit.
2. Make sure that **ram_fref** and **ram_spd0,ram_spd1** are correct.
3. Set **CLK_FREQUENCY** to match the operating frequency.
4. Clear the **refdis, rwcb** bits.
5. Set the **derc** bit.
6. Clear the **scien, dpien, sien**, and **mien** bits.
7. Clear the **mcken** bit.

8. Clear the **swen** and **rtest0,rtest1,rtest2** bits.
9. Make sure that ROM/Flash Banks A and B are not enabled to respond in the range from \$00000000 to \$40000000.
10. Make sure that no other devices respond in the range from \$00000000 to \$40000000.

For each block:

1. Set the block's base address to \$00000000.
2. Enable the block and make sure that the other three blocks are disabled.
3. Set the block's size control bits. Start with the largest possible (1024MB).
4. Write differing 64-bit data patterns to certain addresses within the block. The data patterns do not matter as long as each 64-bit data pattern is unique. The addresses to be written vary depending on the size that is currently being checked and are specified in [Table 3-19 on page 3-61](#). [Table 3-20 on page 3-61](#) shows how PowerPC addresses correspond to DRAM row/column addresses.
5. Read back all of the addresses that have been written.

If all of the addresses still contain exactly what was written, then the block's size has been found. It is the size for which the block is currently programmed.

If any of the addresses do not match exactly, then the amount of memory is less than that for which it is currently programmed. Sizing needs to continue for this block by programming its control bits to the next smaller size and repeating steps 4 and 5.
6. If no match is found for any size, then the block is unpopulated and has a size of 0MB.

Each size that is checked has a specific set of locations that must be written and read. The following table shows the addresses that go with each size.

Table 3-19. Sizing Addresses

| 1024MB | 256MB | 128MB | 64MB | 32MB | 16MB |
|--------------------------|---------------------------------------------------------|--------------------------------------------------------------------------------------------------------------|------------------------------------------------------|--------------------------------------------------------------------------------------------------------------|------------|
| \$00000000 \$20000000 | \$00000000, \$02000000, \$08000000, \$0A000000 | \$00000000 \$00002000 \$02000000 \$02002000 \$04000000 \$04002000 \$06000000 \$06002000 | \$00000000 \$00002000 \$02000000 \$02002000 | \$00000000 \$00001000 \$00002000 \$00003000 \$01000000 \$01001000 \$01002000 \$01003000 | \$00000000 |

Table 3-20. PowerPC 60x Address to DRAM Address Mappings

| RA ----▶ Block Size ▼ | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|----------------------------------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 16MB | ROW | | A19 | A18 | A8 | A9 | A10 | A11 | A12 | A13 | A14 | A15 | A16 | A17 |
| | COL | | | | A18 | A19 | A20 | A21 | A22 | A23 | A24 | A25 | A26 | A27 |
| 32MB | ROW | | A18 | A7 | A8 | A9 | A10 | A11 | A12 | A13 | A14 | A15 | A16 | A17 |
| | COL | | | | A18 | A19 | A20 | A21 | A22 | A23 | A24 | A25 | A26 | A27 |
| 64MB | ROW | A18 | A6 | A7 | A8 | A9 | A10 | A11 | A12 | A13 | A14 | A15 | A16 | A17 |
| | COL | | | A6 | A18 | A19 | A20 | A21 | A22 | A23 | A24 | A25 | A26 | A27 |
| 128MB | ROW | A6 | A5 | A7 | A8 | A9 | A10 | A11 | A12 | A13 | A14 | A15 | A16 | A17 |
| | COL | | | A6 | A18 | A19 | A20 | A21 | A22 | A23 | A24 | A25 | A26 | A27 |
| 256MB | ROW | A4 | A5 | A7 | A8 | A9 | A10 | A11 | A12 | A13 | A14 | A15 | A16 | A17 |
| | COL | | A4 | A6 | A18 | A19 | A20 | A21 | A22 | A23 | A24 | A25 | A26 | A27 |
| 1024MB | ROW | A3 | A5 | A7 | A8 | A9 | A10 | A11 | A12 | A13 | A14 | A15 | A16 | A17 |
| | COL | A2 | A4 | A6 | A18 | A19 | A20 | A21 | A22 | A23 | A24 | A25 | A26 | A27 |

ECC Codes

When the Falcon reports a single-bit error, software can use the syndrome that was logged by the Falcon (upper or lower depending where the error occurred) to determine which bit was in error. [Table 3-21 on page 3-62](#) shows the syndrome for each possible single bit error. [Table 3-22 on page 3-63](#) shows the same information ordered by syndrome. In order to relate this information to PowerPC addresses and bit numbers, the user needs to understand how the Falcon pairs positions PowerPC data in DRAM. See the section on [Data Paths](#) for an explanation of this.

Note Tables 3-22 and 3-23 are the same whether the Falcon is configured as upper or as lower.

Table 3-21. Syndrome Codes Ordered by Bit in Error

| Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome |
|-----|----------|------|----------|------|----------|------|----------|------|----------|
| rd0 | \$4A | rd16 | \$92 | rd32 | \$A4 | rd48 | \$29 | ckd0 | \$01 |
| rd1 | \$4C | rd17 | \$13 | rd33 | \$C4 | rd49 | \$31 | ckd1 | \$02 |
| rd2 | \$2C | rd18 | \$0B | rd34 | \$C2 | rd50 | \$B0 | ckd2 | \$04 |
| rd3 | \$2A | rd19 | \$8A | rd35 | \$A2 | rd51 | \$A8 | ckd3 | \$08 |
| rd4 | \$E9 | rd20 | \$7A | rd36 | \$9E | rd52 | \$A7 | ckd4 | \$10 |
| rd5 | \$1C | rd21 | \$07 | rd37 | \$C1 | rd53 | \$70 | ckd5 | \$20 |
| rd6 | \$1A | rd22 | \$86 | rd38 | \$A1 | rd54 | \$68 | ckd6 | \$40 |
| rd7 | \$19 | rd23 | \$46 | rd39 | \$91 | rd55 | \$64 | ckd7 | \$80 |

Table 3-21. Syndrome Codes Ordered by Bit in Error (Continued)

| Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome |
|------|----------|------|----------|------|----------|------|----------|-----|----------|
| rd8 | \$25 | rd24 | \$49 | rd40 | \$52 | rd56 | \$94 | | |
| rd9 | \$26 | rd25 | \$89 | rd41 | \$62 | rd57 | \$98 | | |
| rd10 | \$16 | rd26 | \$85 | rd42 | \$61 | rd58 | \$58 | | |
| rd11 | \$15 | rd27 | \$45 | rd43 | \$51 | rd59 | \$54 | | |
| rd12 | \$F4 | rd28 | \$3D | rd44 | \$4F | rd60 | \$D3 | | |
| rd13 | \$0E | rd29 | \$83 | rd45 | \$E0 | rd61 | \$38 | | |
| rd14 | \$0D | rd30 | \$43 | rd46 | \$D0 | rd62 | \$34 | | |
| rd15 | \$8C | rd31 | \$23 | rd47 | \$C8 | rd63 | \$32 | | |

Table 3-22. Single-Bit Errors Ordered by Syndrome Code

| Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit |
|----------|------|----------|------|----------|------|----------|------|----------|------|----------|------|----------|------|----------|------|
| \$00 | - | \$20 | ckd5 | \$40 | ckd6 | \$60 | - | \$80 | ckd7 | \$A0 | - | \$C0 | - | \$E0 | rd45 |
| \$01 | ckd0 | \$21 | - | \$41 | - | \$61 | rd42 | \$81 | - | \$A1 | rd38 | \$C1 | rd37 | \$E1 | - |
| \$02 | ckd1 | \$22 | - | \$42 | - | \$62 | rd41 | \$82 | - | \$A2 | rd35 | \$C2 | rd34 | \$E2 | - |
| \$03 | - | \$23 | rd31 | \$43 | rd30 | \$63 | - | \$83 | rd29 | \$A3 | - | \$C3 | - | \$E3 | - |
| \$04 | ckd2 | \$24 | - | \$44 | - | \$64 | rd55 | \$84 | - | \$A4 | rd32 | \$C4 | rd33 | \$E4 | - |
| \$05 | - | \$25 | rd8 | \$45 | rd27 | \$65 | - | \$85 | rd26 | \$A5 | - | \$C5 | - | \$E5 | - |
| \$06 | - | \$26 | rd9 | \$46 | rd23 | \$66 | - | \$86 | rd22 | \$A6 | - | \$C6 | - | \$E6 | - |
| \$07 | rd21 | \$27 | - | \$47 | - | \$67 | - | \$87 | - | \$A7 | rd52 | \$C7 | - | \$E7 | - |
| \$08 | ckd3 | \$28 | - | \$48 | - | \$68 | rd54 | \$88 | - | \$A8 | rd51 | \$C8 | rd47 | \$E8 | - |
| \$09 | - | \$29 | rd48 | \$49 | rd24 | \$69 | - | \$89 | rd25 | \$A9 | - | \$C9 | - | \$E9 | rd4 |
| \$0A | - | \$2A | rd3 | \$4A | rd0 | \$6A | - | \$8A | rd19 | \$AA | - | \$CA | - | \$EA | - |
| \$0B | rd18 | \$2B | - | \$4B | - | \$6B | - | \$8B | - | \$AB | - | \$CB | - | \$EB | - |
| \$0C | - | \$2C | rd2 | \$4C | rd1 | \$6C | - | \$8C | rd15 | \$AC | - | \$CC | - | \$EC | - |
| \$0D | rd14 | \$2D | - | \$4D | - | \$6D | - | \$8D | - | \$AD | - | \$CD | - | \$ED | - |

Table 3-22. Single-Bit Errors Ordered by Syndrome Code (Continued)

| Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit | Syndrome | Bit |
|----------|------|----------|------|----------|------|----------|------|----------|------|----------|------|----------|------|----------|------|
| \$0E | rd13 | \$2E | - | \$4E | - | \$6E | - | \$8E | - | \$AE | - | \$CE | - | \$EE | - |
| \$0F | - | \$2F | - | \$4F | rd44 | \$6F | - | \$8F | - | \$AF | - | \$CF | - | \$EF | - |
| \$10 | ckd4 | \$30 | - | \$50 | - | \$70 | rd53 | \$90 | - | \$B0 | rd50 | \$D0 | rd46 | \$F0 | - |
| \$11 | - | \$31 | rd49 | \$51 | rd43 | \$71 | - | \$91 | rd39 | \$B1 | - | \$D1 | - | \$F1 | - |
| \$12 | - | \$32 | rd63 | \$52 | rd40 | \$72 | - | \$92 | rd16 | \$B2 | - | \$D2 | - | \$F2 | - |
| \$13 | rd17 | \$33 | - | \$53 | - | \$73 | - | \$93 | - | \$B3 | - | \$D3 | rd60 | \$F3 | - |
| \$14 | - | \$34 | rd62 | \$54 | rd59 | \$74 | - | \$94 | rd56 | \$B4 | - | \$D4 | - | \$F4 | rd12 |
| \$15 | rd11 | \$35 | - | \$55 | - | \$75 | - | \$95 | - | \$B5 | - | \$D5 | - | \$F5 | - |
| \$16 | rd10 | \$36 | - | \$56 | - | \$76 | - | \$96 | - | \$B6 | - | \$D6 | - | \$F6 | - |
| \$17 | - | \$37 | - | \$57 | - | \$77 | - | \$97 | - | \$B7 | - | \$D7 | - | \$F7 | - |
| \$18 | - | \$38 | rd61 | \$58 | rd58 | \$78 | - | \$98 | rd57 | \$B8 | - | \$D8 | - | \$F8 | - |
| \$19 | rd7 | \$39 | - | \$59 | - | \$79 | - | \$99 | - | \$B9 | - | \$D9 | - | \$F9 | - |
| \$1A | rd6 | \$3A | - | \$5A | - | \$7A | rd20 | \$9A | - | \$BA | - | \$DA | - | \$FA | - |
| \$1B | - | \$3B | - | \$5B | - | \$7B | - | \$9B | - | \$BB | - | \$DB | - | \$FB | - |
| \$1C | rd5 | \$3C | - | \$5C | - | \$7C | - | \$9C | - | \$BC | - | \$DC | - | \$FC | - |
| \$1D | - | \$3D | rd28 | \$5D | - | \$7D | - | \$9D | - | \$BD | - | \$DD | - | \$FD | - |
| \$1E | - | \$3E | - | \$5E | - | \$7E | - | \$9E | rd36 | \$BE | - | \$DE | - | \$FE | - |
| \$1F | - | \$3F | - | \$5F | - | \$7F | - | \$9F | - | \$BF | - | \$DF | - | \$FF | - |

Data Paths

Because of the Falcon pair architecture, data paths can be confusing. [Figure 3-10](#) attempts to show the placement of data that is written by a PowerPC master to DRAM. [Table 3-23](#) shows the same information in tabular format.

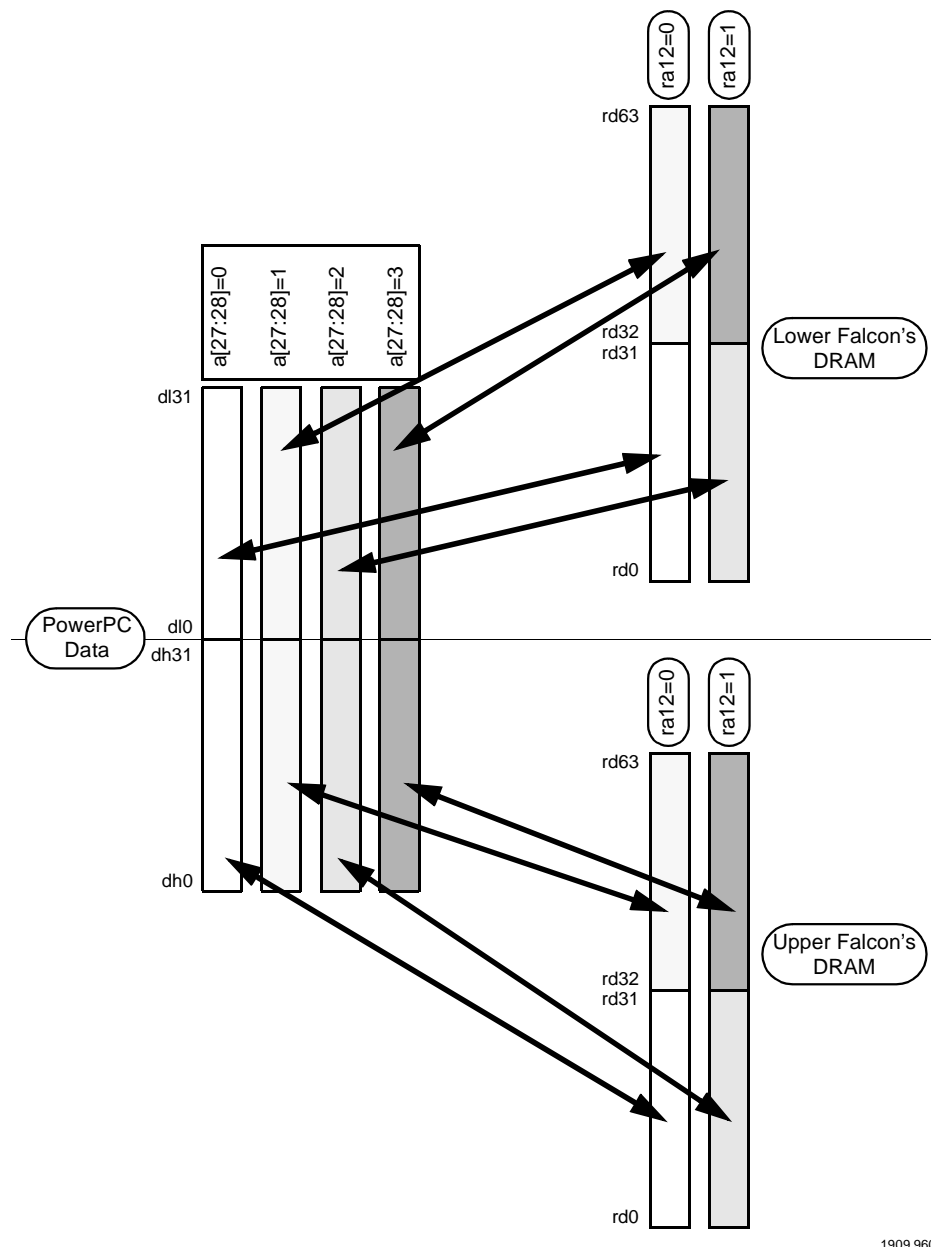


Figure 3-10. PowerPC Data to DRAM Data Correspondence

Table 3-23. PowerPC Data to DRAM Data Mapping

| PowerPC | | | DRAM Array | | |
|---------|-------|-----------|------------|-----------------------------|-----------------------------|
| A[27] | A[28] | Data Bits | RA[12] | Upper Falcon DRAM Data Bits | Lower Falcon DRAM Data Bits |
| 0 | 0 | dh[00:07] | 0 | rd[00:07] | - |
| 0 | 0 | dh[08:15] | 0 | rd[08:15] | - |
| 0 | 0 | dh[16:23] | 0 | rd[16:23] | - |
| 0 | 0 | dh[24:31] | 0 | rd[24:31] | - |
| 0 | 0 | dl[00:07] | 0 | - | rd[00:07] |
| 0 | 0 | dl[08:15] | 0 | - | rd[08:15] |
| 0 | 0 | dl[16:23] | 0 | - | rd[16:23] |
| 0 | 0 | dl[24:31] | 0 | - | rd[24:31] |
| 0 | 1 | dh[00:07] | 0 | rd[32:39] | - |
| 0 | 1 | dh[08:15] | 0 | rd[40:47] | - |
| 0 | 1 | dh[16:23] | 0 | rd[48:55] | - |
| 0 | 1 | dh[24:31] | 0 | rd[56:63] | - |
| 0 | 1 | dl[00:07] | 0 | - | rd[32:39] |
| 0 | 1 | dl[08:15] | 0 | - | rd[40:47] |
| 0 | 1 | dl[16:23] | 0 | - | rd[48:55] |
| 0 | 1 | dl[24:31] | 0 | - | rd[56:63] |
| 1 | 0 | dh[00:07] | 1 | rd[00:07] | - |
| 1 | 0 | dh[08:15] | 1 | rd[08:15] | - |
| 1 | 0 | dh[16:23] | 1 | rd[16:23] | - |
| 1 | 0 | dh[24:31] | 1 | rd[24:31] | - |
| 1 | 0 | dl[00:07] | 1 | - | rd[00:07] |
| 1 | 0 | dl[08:15] | 1 | - | rd[08:15] |
| 1 | 0 | dl[16:23] | 1 | - | rd[16:23] |
| 1 | 0 | dl[24:31] | 1 | - | rd[24:31] |
| 1 | 1 | dh[00:07] | 1 | rd[32:39] | - |
| 1 | 1 | dh[08:15] | 1 | rd[40:47] | - |

Table 3-23. PowerPC Data to DRAM Data Mapping (Continued)

| PowerPC | | | DRAM Array | | |
|---------|---|-----------|------------|-----------|-----------|
| 1 | 1 | dh[16:23] | 1 | rd[48:55] | - |
| 1 | 1 | dh[24:31] | 1 | rd[56:63] | - |
| 1 | 1 | dl[00:07] | 1 | - | rd[32:39] |
| 1 | 1 | dl[08:15] | 1 | - | rd[40:47] |
| 1 | 1 | dl[16:23] | 1 | - | rd[48:55] |
| 1 | 1 | dl[24:31] | 1 | - | rd[56:63] |

Introduction

This chapter provides details of several programming functions that are not tied to any specific ASIC chip.

PCI Arbitration

There are six potential local PCI bus masters on the MCP750 SBC:

1. Raven ASIC (MPU/PCI bus bridge controller)
2. VIA 82C586B PBC (Peripheral Bus Controller) PCI/ISA bridge
3. DEC 21154 PCI-to-PCI bridge
4. DEC 21140 Ethernet Controller
5. PMC Slot (PCI mezzanine card)
6. External PCI bus master via J4 connector

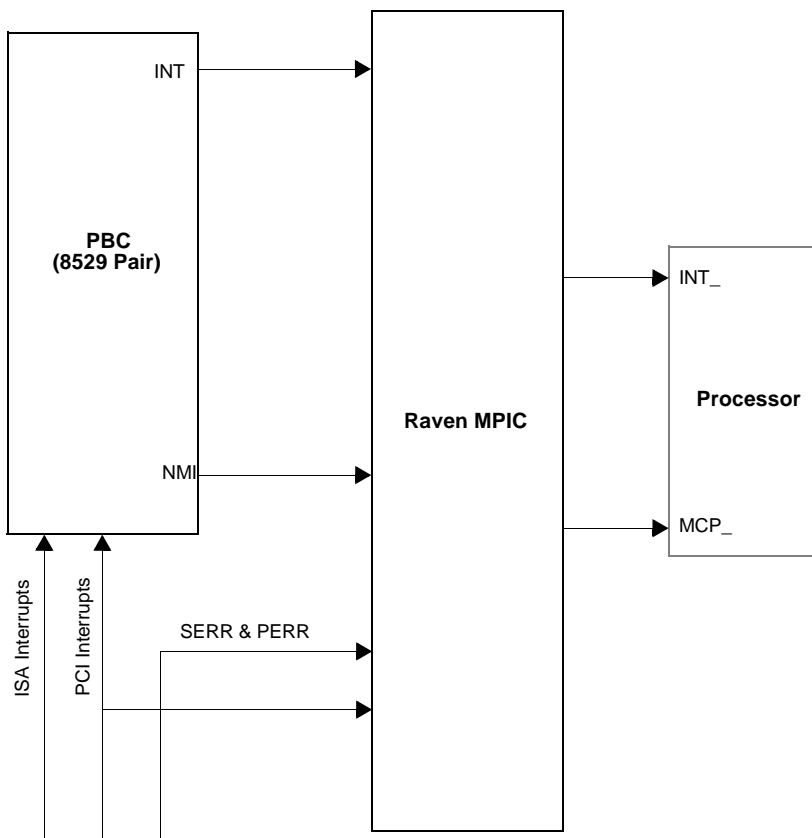
Interrupt Handling

The Raven ASIC provides an MPIC Interrupt Controller to handle various interrupt sources. It controls PHB (PCI Host Bridge) MPU/local bus interface functions on the MCP750, as well as performing interrupt handling. Sources of interrupts may be any of the following:

- ❑ The Raven ASIC itself (timer interrupts or transfer error interrupts)
- ❑ The processor 0 (processor self-interrupts)
- ❑ The Falcon chip set (memory error interrupts)
- ❑ The PCI bus (interrupts from PCI devices)

- ❑ The CPCI bus (interrupts from CPCI devices)
- ❑ Power monitor interrupts
- ❑ Watchdog timer interrupt
- ❑ The ISA bus (interrupts from ISA devices)

The figure below illustrates interrupt architecture on the MCP750.



11559.00 9609

Figure 4-1. MCP750 Series Interrupt Architecture

Raven MPIC

The Raven ASIC has a built-in interrupt controller that meets the Multi-Processor Interrupt Controller (MPIC) Specification. This MPIC supports up to two processors and 16 external interrupt sources. There are also six other interrupt sources inside the MPIC: two cross-processor interrupts and four timer interrupts. All ISA interrupts go through the 8259 pair in the PIB. The output of the PIB then goes through the MPIC in the Raven.

Refer to [Chapter 2, Raven PCI Host Bridge & Multi-Processor Interrupt Controller](#) for details on the Raven MPIC. The following table shows the interrupt assignments for the Raven MPIC on the MCP750 series:

Table 4-1. RavenMPIC Interrupt Assignments

| MPIC IRQ | Edge/Level | Polarity | Interrupt Source | Notes |
|----------|------------|----------|-------------------------|-------|
| IRQ0 | Level | High | PBC (8259) | 1 |
| IRQ1 | Edge | Low | Falcon-ECC Error | 2 |
| IRQ2 | Level | Low | PCI-Ethernet | |
| IRQ3 | Level | Low | PMC INT | 3 |
| IRQ4 | Level | Low | Watchdog Timer Level 1 | |
| IRQ5 | Level | Low | CompactPCI PRST# signal | |
| IRQ6 | Level | Low | CompactPCI FAL# signal | |
| IRQ7 | Level | Low | CompactPCI DEG# signal | |
| IRQ8 | Level | Low | CompactPCI Bus 1 INTA# | |
| IRQ9 | Level | Low | CompactPCI Bus 1 INTB# | |
| IRQ10 | Level | Low | CompactPCI Bus 1 INTC# | |
| IRQ11 | Level | Low | CompactPCI Bus 1 INTD# | |
| IRQ12 | Level | Low | CompactPCI Bus 2 INTA# | 4 |
| IRQ13 | Level | Low | CompactPCI Bus 2 INTB# | 4 |
| IRQ14 | Level | Low | CompactPCI Bus 2 INTC# | 4 |
| IRQ15 | Level | Low | CompactPCI Bus 2 INTD# | 4 |

Notes

1. Interrupt from the PCI/ISA Bridge.
2. Interrupt from the Falcon chipset for a Single and/or Double bit memory error.
3. This is the logical OR of the PMC slot interrupts INTA# through INTD#.
4. CompactPCI Bus 2 interrupts are provided by an optional companion card via J4.

8259 Interrupts

The PBC supports 15 interrupt requests. These 15 interrupts are ISA-type interrupts that are functionally equivalent to two 82C59 interrupt controllers. Except for IRQ0, IRQ1, IRQ2, IRQ8_, and IRQ13, each of the interrupt lines can be configured for either edge-sensitive mode or level-sensitive mode by programming the appropriate ELCR registers in the PBC. The PBC treats level-sensitive interrupts as active low signal inputs. Therefore, any interrupt connected to an IRQ input which is intended to be a level-sensitive interrupt must be an active low signal.

There is also support for four PCI interrupts, PIRQD_ through PIRQA_. The PBC has four PIRQ Route Control Registers to allow each of the PCI interrupt lines to be routed to any of twelve ISA interrupt lines (IRQ0, IRQ2, IRQ8_, and IRQ13 are reserved for ISA system interrupts). Since PCI interrupts are defined as level-sensitive/active low, software must program the selected IRQ(s) for level-sensitive mode. Note that more than one PCI interrupt can be routed to the same ISA IRQ line. The PBC can be programmed to handle the PCI interrupts if the Raven MPIC is either not present or not used.

The following figure shows the interrupt structure of the PBC.

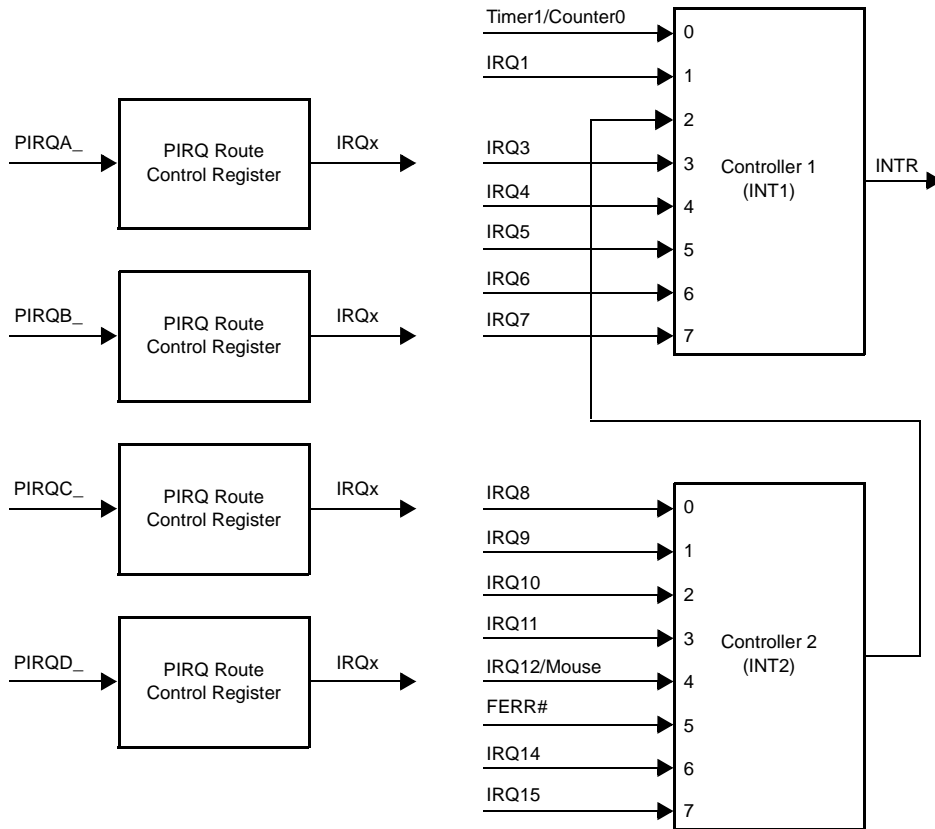


Figure 4-2. PBC Interrupt Handler Block Diagram

The assignments of the PCI and ISA interrupts supported by the PBC are as follows:

Table 4-2. PBC ISA Interrupt Assignments

| PRI | PBC IRQ Input | Routed to ISA IRQ | Controller | Edge/Level | Polarity | Interrupt Source | Notes |
|------|---------------|-------------------|------------|------------|----------|-----------------------------|---------------------|
| 1 | | IRQ0 | INT1 | Edge | High | Timer 1 / Counter 0 | 1 |
| 2 | MSCK/IRQ1 | IRQ1 | | Edge | High | Keyboard | |
| 3-10 | | IRQ2 | | Edge | High | Cascade Interrupt from INT2 | |
| 3 | RTCX1/IRQ8_ | IRQ8 | INT2 | Edge | Low | ABORT Switch Interrupt | 3 |
| 4 | IRQ9 | IRQ9 | | Level | Low | Z85230/Z8536 | 5 |
| 5 | PIRQA_ | IRQ10 | | Level | Low | 21140 Ethernet chip | 2 |
| 6 | | IRQ11 | | Level | Low | Internal USB controller | 2 |
| 7 | MSDT/IRQ12 | IRQ12 | | Edge | High | Mouse | |
| 8 | | IRQ13 | | Edge | High | Not Used | |
| 9 | IRQ14 | IRQ14 | | Edge | High | Primary IDE interface | |
| 10 | IRQ15 | IRQ15 | | Edge | High | Secondary IDE interface | |
| 11 | IRQ3 | IRQ3 | INT1 | Edge | High | COM2 (Async Serial Port 2) | |
| 12 | IRQ4 | IRQ4 | | Edge | High | COM1 (Async Serial Port 1) | |
| 13 | PIRQB_ | IRQ5 | | Level | Low | PMC | 2,4 |
| 14 | IRQ6 | IRQ6 | | Edge | High | Floppy | |
| 15 | IRQ7 | IRQ7 | | Edge | High | Parallel Port | |

Notes:

1. Internally generated by the PBC.
2. These interrupt sources must be routed to the appropriate ISA IRQ using the PBC interrupt routing registers.

3. The ABORT switch interrupt is OR'ed with the RTC interrupt. This is an inverting input on the PBC chip.
4. The PMC interrupt is the logical OR of PMC interrupts INTA# through INTD#.
5. On Rev 01C MCP750 boards, this interrupt must be programmed as an edge-sensitive interrupt because this is an active high interrupt on Rev 01C boards.

ISA DMA Channels

Refer to [Chapter 1, *Board Description and Memory Maps*](#), for information on the ISA DMA channels.

Exceptions

Sources of Reset

There are six potential sources of reset on the MCP750 series:

1. Power-On/undervoltage Reset
2. Front Panel Reset Switch
3. Watchdog Timer Reset via the MK48T559 Watchdog Timer or the Raven Watchdog Timer
4. Port 92 Register via the PBC
5. CompactPCI Bus 1 via the 21154 Bridge Control Register
6. Processor RISCwatch JTAG Emulator Interface HRESET# Signal (open collector).

The following table shows which devices are affected by various reset sources:

Table 4-3. Reset Sources and Devices Affected

| Device Affected | Processor (s) | Raven ASIC | Falcon Chipset | PCI Devices | ISA Devices | CompactPCI Busses |
|--------------------------------------------------|---------------|------------|----------------|-------------|-------------|-------------------|
| Power-On/undervoltage | 3 | 3 | 3 | 3 | 3 | 3 |
| Front Panel Reset Switch | 3 | 3 | 3 | 3 | 3 | 3 |
| Watchdog Timer (M48T559/Raven) | 3 | 3 | 3 | 3 | 3 | 3 |
| Software Hard Reset (PBC Port92 Register) | 3 | 3 | 3 | 3 | 3 | 3 |
| CompactPCI Reset (21154 Bridge Control Register) | | | | | | 3 |
| Processor RISCWatch JTAG HRESET# | 3 | 3 | 3 | | | |

Soft Reset

Software can assert the SRESET# pin of any processor by programming the Processor Init Register of the Raven MPIC appropriately.

Error Notification and Handling

The Raven and Falcon chipset can detect certain hardware errors and can be programmed to report these errors via the Raven MPIC interrupts or Machine Check Interrupt.

Note The TEA* signal is not used at all by the MCP750 series. The following table summarizes how the hardware errors are handled by the MCP750 series:

Table 4-4. Error Notification and Handling

| Cause | Action |
|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Single-bit ECC | <i>Store</i> : Write corrected data to memory <i>Load</i> : Present corrected data to the PPC master Generate interrupt via Raven MPIC if so enabled |
| Double-bit ECC | <i>Store</i> : Terminate the bus cycle normally without writing to DRAM <i>Load</i> : Present un-corrected data to the PPC master Generate interrupt via Raven MPIC if so enabled Generate Machine Check Interrupt to the Processor(s) if so enabled |
| PPC Bus Time Out | <i>Store</i> : Discard write data and terminate bus cycle normally <i>Load</i> : Present undefined data to the PPC master Generate interrupt via Raven MPIC if so enabled Generate Machine Check Interrupt to the Processor(s) if so enabled |
| PCI Target Abort | <i>Store</i> : Discard write data and terminate bus cycle normally <i>Load</i> : Return all 1's and terminate bus cycle normally Generate interrupt via Raven MPIC if so enabled Generate Machine Check Interrupt to the Processor(s) if so enabled |
| PCI Master Abort | <i>Store</i> : Discard write data and terminate bus cycle normally <i>Load</i> : Return all 1's and terminate bus cycle normally Generate interrupt via Raven MPIC if so enabled Generate Machine Check Interrupt to the Processor(s) if so enabled |
| PERR# Detected | Generate interrupt via Raven MPIC if so enabled Generate Machine Check Interrupt to the Processor(s) if so enabled |
| SERR# Detected | Generate interrupt via Raven MPIC if so enabled Generate Machine Check Interrupt to the Processor(s) if so enabled |

Endian Issues

The MCP750 series supports both little-endian and big-endian software. The PowerPC processor is inherently big-endian and PCI is inherently little-endian. The following figures show how the MCP750 handles the endian issue in big- and little-endian modes:

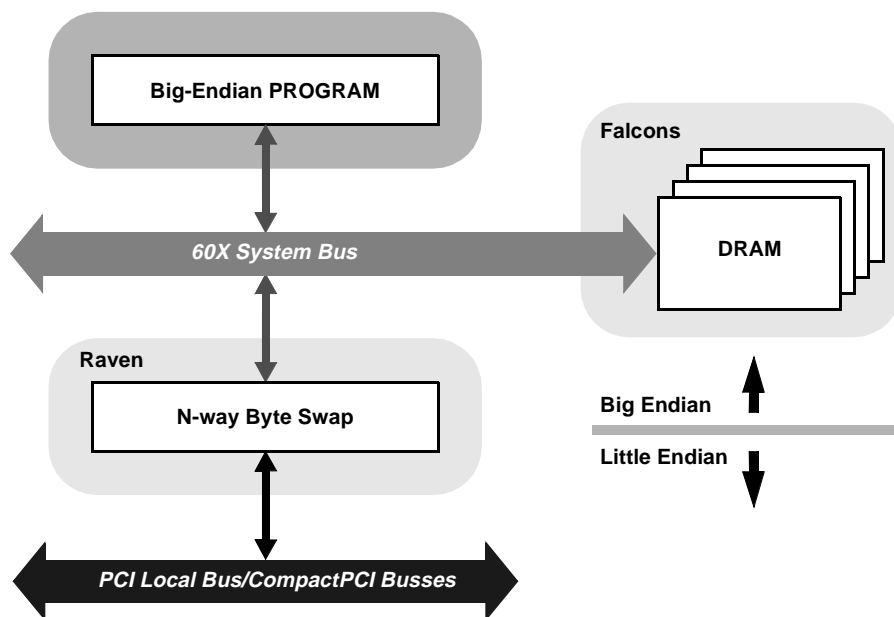


Figure 4-3. Big-Endian Mode

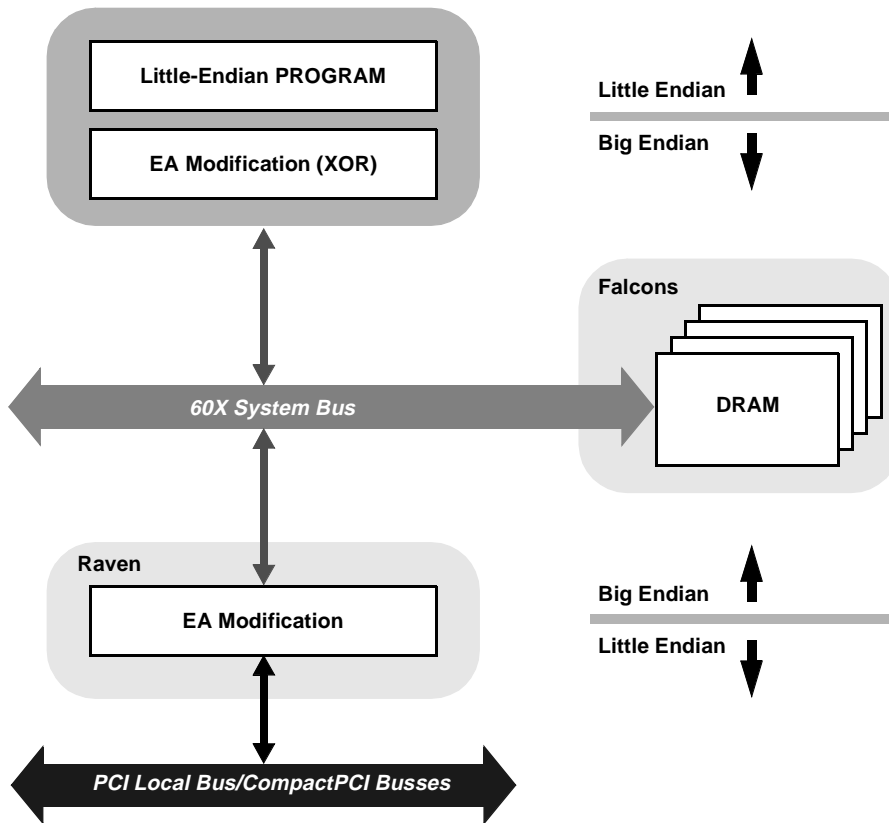


Figure 4-4. Little-Endian Mode

Processor/Memory Domain

The PPC750 processor can operate in both big-endian and little-endian mode. However, it always treats the external processor/memory bus as big-endian by performing *address rearrangement and reordering* when running in little-endian mode.

The PPC registers inside Raven, the registers inside the Falcon chipset, the DRAM, the ROM/FLASH and the system registers always appear as big-endian.

Raven's Involvement

Since PCI is little-endian, the Raven performs byte swapping in both directions (from PCI to memory and from the processor to PCI) to maintain address invariance when it is programmed to operate in big-endian mode with the processor and the memory sub-system.

In little-endian mode, it *reverse-rearranges* the address for PCI-bound accesses and *rearranges* the address for memory-bound accesses (from PCI). In this case, no byte swapping is done.

PCI Domain

The PCI bus is inherently little-endian and all devices connected directly to PCI will operate in little-endian mode, regardless of the mode of operation in the processor's domain.

PCI-Ethernet

Ethernet is byte stream oriented with the byte having the lowest address in memory being the first one to be transferred regardless of the endian mode. Since address invariance is maintained by the Raven in both little-endian and big-endian mode, there should be no endian issues for the Ethernet data. Big-endian software must still however be aware of the byte-swapping effect when accessing the registers of the PCI-Ethernet device.

ROM/Flash Initialization

There are two methods used to inject code into the Flash in Bank A: (1) In-circuit programming and (2) Loading it from the ROM/Flash Bank B. For the second method, the hardware must direct the Falcon chipset to map the FFF00000-FFFFFFFF address range to Bank B following a hard reset. Bank A then can be programmed by code from Bank B.

Software can determine the mapping of the FFF00000-FFFFFFFF address range by examining the **rom_b_rv** bit in the Falcon's ROM B Base/Size Register.

Table 4-5. ROM/FLASH Bank Default

| rom_b_rv | Default Mapping for FFF00000-FFFFFFFF |
|----------|---------------------------------------|
| 0 | ROM/FLASH Bank A |
| 1 | ROM/FLASH Bank B |

Determining CPU Type

The SYSID field in the System Configuration Register allows up to 256 types of CPU. This field is always FCh for the MCP750.

Determining Base Type

The initialization software can determine the types of the base modules by interrogating the Base Status Register which reside at offset 803h from the PCI I/O Base Address.

Motorola Computer Group Documents

The Motorola publications listed below are referenced in this manual. You can obtain paper or electronic copies of Motorola Computer Group publications by:

- ❑ Contacting your local Motorola sales office
- ❑ Visiting Motorola Computer Group's World Wide Web literature site, <http://www.motorola.com/computer/literature>

Table A-1. Motorola Computer Group Documents

| Document Title | Publication Number |
|----------------------------------------------------------------------|----------------------------|
| MCP750 CompactPCI Single Board Computer Installation and Use | MCP750A/IH |
| MCP750 CompactPCI Single Board Computer Programmer's Reference Guide | MCP750A/PG |
| TMCP700 Transition Module Installation and Use | TMCP700A/IH |
| PPCBUG Firmware Package User's Manual (Parts 1 and 2) | PPCBUGA1/UM PPCBUGA2/UM |
| PPCBUG Diagnostics Users Manual | PPCDIAA/UM |

To obtain the most up-to-date product information in PDF or HTML format, visit <http://www.motorola.com/computer/literature>

Manufacturers' Documents

For additional information, refer to the following table for manufacturers' data sheets or user's manuals. As an additional help, a source for the listed document is provided. Please note that, while these sources have been verified, the information is subject to change without notice.

Table A-2. Manufacturers' Documents (Continued)

| Document Title and Source | Publication Number |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------|
| 21154 PCI-to-PCI Bridge Data Sheet; Intel Corporation http://developer.intel.com/design/bridge/datashts/278108.htm | 278108.pdf |
| PC87308VUL (Super I/O Enhanced Sidewinder Lite) Floppy Disk Controller, Keyboard Controller, Real-Time Clock, Dual UARTs, IEEE 1284 Parallel Port, and IDE Interface National Semiconductor Corporation Telephone: 1-800-272-9959 http://www.national.com/pf/PC/PC87308.html | PC87308.html |
| M48T59 CMOS 8K x 8 TIMEKEEPER™ SRAM Data Sheet STMicroelectronics; http://eu.st.com/stonline/index.shtml | M48T59 |
| SCC (Serial Communications Controller) User's Manual (for Z85230 and other Zilog parts) http://www.zilog.com/pdfs/serial/scc_escs_iscc_manual/contents.html | SCC/ESCC User's Manual |
| Z8536 CIO Counter/Timer and Parallel I/O Unit Product Specification and User's Manual (in Z8000® Family of Products Data Book) http://www.zilog.com/products/zx80dev.html#um | DM10001176 |
| VT82C586B PIPC PCI Integrated Peripheral Controller http://www.viatech.com/pdf/productinfo/586b.pdf | VT82C586B.pdf |
| ATMEL Nonvolatile Memory Data Book Must request documentation at: http://www.atmel.com/atmel/support/ | AT24C04 |

Related Specifications

For additional information, refer to the following table for related specifications. As an additional help, a source for the listed document is provided. Please note that while these sources have been verified, the information is subject to change without notice.

Table A-3. Related Specifications

| Document Title and Source | Publication Number |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------|
| IEEE - Common Mezzanine Card Specification (CMC) Institute of Electrical and Electronics Engineers, Inc. http://standards.ieee.org/catalog/ | P1386 Draft 2.0 |
| IEEE - PCI Mezzanine Card Specification (PMC) Institute of Electrical and Electronics Engineers, Inc. http://standards.ieee.org/catalog/ | P1386.1 Draft 2.0 |
| Bidirectional Parallel Port Interface Specification Institute of Electrical and Electronics Engineers, Inc. http://standards.ieee.org/catalog/ | IEEE Standard 1284 |
| Peripheral Component Interconnect (PCI) Local Bus Specification, Revision 2.0, 2.1, 2.2 PCI Special Interest Group; http://www.pcisig.com/ | PCI Local Bus Specification |
| PowerPC Microprocessor Common Hardware Reference Platform: A System Architecture (CHRP), Version 1.0 Literature Distribution Center for Motorola Telephone: 1-800- 441-2447 FAX: (602) 994-6430 or (303) 675-2150 http://merchant.hibbertco.com/mtrlexit/ E-mail: ldcformotorola@hibbertco.com OR Morgan Kaufmann Publishers, Inc. Telephone: (415) 392-2665 Telephone: 1-800-745-7323 http://www.mkp.com/books_catalog/ | |

Table A-3. Related Specifications (Continued)

| Document Title and Source | Publication Number |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------|
| PowerPC Reference Platform (PRP) Specification, Third Edition, Version 1.0, Volumes I and II; International Business Machines Corporation http://www.ibm.com | MPR-PPC-RPU-02 |
| IEEE Standard for Local Area Networks: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications Institute of Electrical and Electronics Engineers, Inc. http://standards.ieee.org/catalog/ | IEEE 802.3 |
| Information Technology - Local and Metropolitan Networks - Part 3: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications Global Engineering Documents; http://global.ihs.com/index.cfm (for publications) <i>(This document can also be obtained through the national standards body of member countries.)</i> | ISO/IEC 8802-3 |
| Interface Between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange; Electronic Industries Alliance; http://global.ihs.com/index.cfm (for publications) | TIA/EIA-232 Standard |
| Compact PCI Specification http://www.picmg.com/ | CPCI Rev. 2.1 Dated 9/2/97 |
| PCI-to-PCI Bridge Specification PCI-ISA Specification http://www.picmg.com/ | Rev. 1.02 Rev. 2.0 |

URLs

The following URLs (uniform resource locators) may provide helpful sources of additional information about this product, related services, and development tools. Please note that, while these URLs have been verified, they are subject to change without notice.

- ❑ Motorola Computer Group, <http://www.motorola.com/computer>
- ❑ Motorola Computer Group OEM Services,
<http://www.motorola.com/computer/support>

Numerics

32-Bit Counter [3-56](#)
8259 compatibility [2-54](#)
8259 interrupts [4-4](#)
8259 mode [2-82](#)

A

A0-A31 [3-5](#)
access timing (DRAM) [3-7](#), [3-8](#), [3-10](#)
Access Timing (ROM) [3-11](#), [3-12](#)
address modification for little endian transfers [2-16](#)
address pipelining [3-6](#)
address transfers [3-13](#)
Application-Specific Integrated Circuit (ASIC) [1-1](#)
architectural notes [2-82](#)
architecture [2-52](#)
ARTRY_ [3-14](#)
assertion, definition [xxiii](#)
asterisk (*) [xxiii](#)

B

Base Module Feature Register [1-24](#)
Base Module Status Register (BMSR) [1-24](#)
big to little endian data swap [2-15](#)
big-endian [xxiv](#)
big-endian mode [4-10](#)
binary number [xxiii](#)
bit descriptions [3-32](#)
bit ordering convention [3-1](#)
block diagram [2-3](#)
block diagram description [2-56](#)

block diagrams [3-2](#)
blocks A and/or B present, blocks C and D not present [3-21](#)
blocks A and/or B present, blocks C and/or D present [3-22](#)
bus interface (60x) [3-13](#)
byte ordering [xxiv](#)
byte, definition [xxiii](#)

C

cache coherency [3-13](#)
cache coherency restrictions [3-14](#)
chip defaults [3-23](#)
CHRP compliant memory map [2-4](#)
CLK FREQUENCY [3-37](#)
clock frequency [3-37](#)
Column Address bits [3-46](#)
CONFIG_ADDRESS [2-49](#)
control bit descriptions [3-32](#)
control bit, definition [xxiv](#)
conventions, manual [xxiii](#)
CPU Configuration Register [1-23](#)
CPU Control Register [1-19](#)
CSR accesses [3-24](#)
CSR architecture [3-25](#)
CSR base address [1-12](#), [3-25](#)
CSR reads and writes [3-25](#)
CSR's readability [2-52](#)
current task priority level [2-82](#)
cycles originating from PCI [2-16](#)

D

data path diagram [3-65](#)

data path for reads from the Falcon internal CSRs [3-25](#)
 data path for writes to the Falcon internal CSRs [3-26](#)
 data path mapping [3-66](#)
 data paths [3-64](#)
 data transfers [3-13](#)
 DECchip 21140 LAN controller [4-1](#)
 decimal number [xxiii](#)
 default PCI memory map [1-9](#)
 default processor memory map [1-6](#)
 derc [3-40](#)
 Disable Error Correction control bit [3-40](#)
 double word, definition [xxiv](#)
 DRAM addressing [3-61](#)
 DRAM attributes register [3-35](#)
 DRAM Base Register [3-37](#)
 DRAM connection diagram [3-5](#)
 DRAM enable bits [3-35](#)
 DRAM size control bits [3-36](#)
 DRAM speed control bits [3-34](#)
 DRAM speeds [3-7](#)
 dynamically changing I/O interrupt configuration [2-81](#)

E

ECC [3-15](#)
 ECC codes [3-62](#)
 ECC Control Register [3-38](#)
 elog [3-43](#)
 embt [3-43](#)
 endian conversion [2-14](#)
 endian issues [4-10](#)
 End-of-Interrupt Registers [2-78](#)
 EOI Register [2-81](#)
 Error Address Register [3-44](#)
 error correction [3-15](#)
 Error Correction Codes [3-62](#)
 error detection [3-15](#)
 error handling [2-16](#)
 Error Logger Register [3-42](#)
 error logging [3-17](#)

error notification and handling [4-8, 4-9](#)
 error reporting [3-16](#)
 ERROR_ADDRESS [3-44](#)
 ERROR_SYNDROME [3-43](#)
 esbt [3-43](#)
 escb [3-43](#)
 esen [3-43](#)
 exceptions [4-7](#)
 external interrupt service [2-79](#)
 External Register Set [3-57](#)
 external register set [3-23](#)
 external register set reads and writes [3-25](#)
 External Source Destination Registers [2-74](#)
 External Source Vector/Priority Registers [2-72](#)

F

Falcon ECC memory controller chip set [3-1](#)
 Falcon internal data paths (simplified) [3-4](#)
 Falcon memory controller chip set [4-1](#)
 Falcon pair used with DRAM in a system [3-3](#)
 Falcon-controlled system registers [1-11](#)
 false, definition [xxiv](#)
 fast refresh control bit [3-34](#)
 Feature Reporting Register [2-64](#)
 features [2-2, 3-1](#)
 Flash (See ROM/Flash) [3-17](#)
 four-beat reads/writes [3-6](#)
 functional description [1-5, 2-4, 3-6](#)

G

General Control-Status/Feature Registers [2-25](#)
 General Purpose Registers [2-41](#)
 generating PCI configuration cycles [2-13](#)
 generating PCI interrupt acknowledge cycles [2-14](#)
 generating PCI memory and I/O cycles [2-11](#)
 generating PCI special cycles [2-13](#)
 Global Configuration Register [2-65](#)

H

half-word, definition [xxiii](#)

hexadecimal character [xxiii](#)

I

I/O Base Register [2-46](#)

In-Service Register (ISR) [2-58](#)

interpretation of MID3-MID0 [1-31](#)

Interprocessor Interrupt Dispatch Registers [2-76](#)

interprocessor interrupts [2-80](#)

interprocessor interrupts (IPI) [2-54](#)

Interrupt Acknowledge Register [2-81](#)

Interrupt Acknowledge Registers [2-77](#)

interrupt delivery modes [2-55](#)

Interrupt Enable Control Bits [3-41](#)

Interrupt Enable control bits [3-41](#)

interrupt handling [4-1](#)

Interrupt Pending Register (IPR) [2-56](#)

Interrupt Request Register (IRR) [2-58](#)

interrupt router [2-59](#)

interrupt selector (IS) [2-58](#)

interrupt source priority [2-53](#)

interrupt support

sources [4-1](#)

Interrupt Task Priority Registers [2-77](#)

introduction [2-1](#), [2-51](#), [3-1](#), [4-1](#)

IPI Vector/Priority Registers [2-67](#)

ISA bus [4-2](#)

ISA DMA channels [1-32](#), [4-7](#)

ISA local resource bus [1-20](#)

L

L2 cache support [3-14](#)

L2CLM_ [3-14](#)

Large Scale Integration (LSI) [1-1](#)

little-endian mode [4-11](#)

M

manual terminology [xxiii](#)

manufacturer's documents [A-1](#)

manufacturers' documents [A-1](#)

mcken [3-41](#)

Memory Base Register [2-46](#)

Memory Configuration Register (MEMCR) [1-15](#)

memory map for 4-byte reads to the CSR [3-29](#)

memory map for 4-byte writes to the internal register set and test SRAM [3-29](#)

memory map for byte reads to the CSR [3-27](#)

memory map for byte writes to the internal register set and test SRAM [3-28](#)

memory maps [1-6](#)

mien [3-41](#)

MK48T59 access registers [1-21](#)

module configuration and status registers [1-22](#)

Motorola Computer Group Documents [A-1](#)

MPC bus interface [2-4](#)

MPC bus timer [2-8](#)

MPC Error Address Register [2-32](#)

MPC Error Attribute Register - MERAT [2-32](#)

MPC Error Enable Register [2-28](#)

MPC Error Status Register [2-30](#)

MPC map decoders [2-4](#)

MPC master [2-6](#)

MPC registers [2-23](#)

MPC Slave Address (0,1 and 2) Registers [2-35](#)

MPC Slave Address (3) Register [2-35](#)

MPC Slave Offset/Attribute (0,1 and 2) Registers [2-36](#)

MPC Slave Offset/Attribute (3) Registers [2-37](#)

MPC transfer types [2-8](#)

MPC write posting [2-5](#)

MPIC registers [2-60](#)

MVME2600 series [1-1](#)

MVME2600 series features summary [1-2](#)

MVME2600 series interrupt architecture [4-2](#)

MVME2600 series system block diagram [1-4](#)

N

negation, definition [xxiii](#)
 nesting of interrupt events [2-53](#)
 NVRAM/RTC & Watchdog Timer Registers
[1-21](#)

O

operation [2-80](#)
 overall DRAM connections [3-5](#)
 overview [1-1](#), [2-1](#), [3-1](#)

P

parity checking [3-58](#)
 PC87308VUL Super I/O (ISASIO) strapping
[1-20](#)
 PCI arbitration [4-1](#)
 PCI command codes [2-11](#)
 PCI Command/ Status Registers [2-43](#)
 PCI configuration access [1-9](#)
 PCI configuration space [2-10](#)
 PCI domain [4-12](#)
 PCI I/O CONFIG_ADDRESS Register [2-49](#)
 PCI I/O CONFIG_DATA Register [2-51](#)
 PCI interface [2-9](#)
 PCI Interrupt Acknowledge Register [2-34](#)
 PCI map decoders [2-9](#)
 PCI master [2-11](#)
 PCI memory maps [1-9](#)
 PCI PREP memory map [1-10](#)
 PCI registers [2-41](#)
 PCI Slave Address (0,1,2 and 3) Registers
[2-47](#)
 PCI Slave Attribute/ Offset (0,1,2 and 3)
 Registers [2-48](#)
 PCI spread I/O cycle mapping [2-12](#)
 PCI write posting [2-10](#)
 PCI/MPC contention handling [2-18](#)
 PCI-Ethernet [4-12](#)
 PCI-graphics [4-12](#)
 performance [3-6](#)
 PIB interrupt handler block diagram [4-5](#)
 PIB PCI/ISA interrupt assignments [4-6](#)

PMC Slot

as PCI bus master [4-1](#)

Power monitor

as source of interrupt [4-2](#)

PowerPC 60x to ROM/Flash Address Map-
 ping when ROM/Flash is 16 Bits
 Wide (8 Bits per Falcon) [3-19](#)

PowerPC 60x to ROM/Flash Address Map-
 ping when ROM/Flash is 64 Bits
 Wide (32 Bits per Falcon) [3-20](#)

power-up reset status bit [3-38](#)

Power-Up Reset Status Register 1 [3-56](#)

Power-Up Reset Status Register 2 [3-57](#)

PR_STAT1 bits [3-56](#)

PR_STAT2 bits [3-57](#)

PREP memory map example [1-8](#)

Prescaler Adjust Register [2-27](#)

Processor Init Register [2-66](#)

processor memory maps [1-6](#)

processor PREP memory map [1-8](#)

processor/memory domain [4-11](#)

processor's current task priority [2-53](#)

program visible registers [2-56](#)

programming details [4-1](#)

programming model [1-6](#)

programming notes [2-78](#)

programming ROM/Flash [3-58](#), [3-59](#)

R

RAM A BASE [3-37](#)

RAM B BASE [3-37](#)

RAM C BASE [3-37](#)

RAM D BASE [3-37](#)

Raven block diagram [2-3](#)

Raven interrupt controller (RavenMPIC) fea-
 tures [2-51](#)

Raven MPC register map [2-23](#)

Raven MPC register values for PREP memo-
 ry map [1-9](#)

Raven MPU/PCI bus bridge controller ASIC
[4-1](#)

Raven PCI configuration register map [2-42](#)

- Raven PCI Host Bridge & Multi-Processor Interrupt Controller chip [2-1](#)
- Raven PCI I/O register map [2-42](#)
- Raven PCI register values for PREP memory map [1-10](#)
- Raven's involvement [4-12](#)
- Raven-detected errors [2-54](#)
- Raven-Detected Errors Destination Register [2-75](#)
- Raven-Detected Errors Vector/Priority Register [2-74](#)
- RavenMPIC [4-3](#)
- RavenMPIC block diagram [2-57](#)
- RavenMPIC interrupt assignments [4-3](#)
- RavenMPIC register map [2-61](#)
- RavenMPIC registers [2-60](#)
- Read/Write Checkbits control bit [3-38](#)
- reads/writes to ROM/Flash [3-49](#)
- refdis [3-38](#)
- Refresh Counter Test control bits [3-45](#)
- Refresh/Scrub [3-21](#)
- Refresh/Scrub Address Register [3-46](#)
- register bit descriptions [3-32](#)
- register summary [3-30](#)
- related specifications [A-4](#)
- requirements [2-1](#)
- reset sources and devices affected [4-8](#)
- reset state [2-80](#)
- revision ID bits [3-33](#)
- Revision ID Register [2-25](#)
- Revision ID/ Class Code Registers [2-45](#)
- ROM/Flash [3-17](#)
- ROM/Flash A Base Address control bits [3-47](#)
- ROM/Flash A Base/Size Register [3-47](#)
- ROM/Flash A size encoding bits [3-48](#)
- ROM/Flash A Width control bit [3-47](#)
- ROM/Flash B Base Address control bits [3-50](#)
- ROM/Flash B Base/Size Register [3-50](#)
- ROM/Flash B Width control bit [3-50](#)
- ROM/FLASH bank default [4-13](#)
- ROM/Flash initialization [4-12](#)
- ROM/Flash speeds [3-11](#)
- rom_a_64 bit [3-47](#)
- ROM_A_BASE [3-47](#)
- rom_a_en bit [3-48](#)
- rom_a_rv bit [3-48](#)
- rom_a_siz bit [3-48](#)
- rom_a_we bit [3-48](#)
- rom_b_64 bit [3-50](#)
- ROM_B_BASE bits [3-50](#)
- rom_b_en bit [3-51](#)
- rom_b_rv bit [3-51](#)
- rom_b_siz bit [3-51](#)
- rom_b_we bit [3-51](#)
- Row Address bits [3-46](#)
- rtest0-rtest2 [3-45](#)
- rwcw [3-38](#)

S

- SBE_COUNT [3-44](#)
- scb0,scb1 [3-45](#)
- scien [3-41](#)
- scof [3-44](#)
- Scrub Counter bits [3-45](#)
- Scrub Write Enable control bit [3-45](#)
- Scrub/Refresh Register [3-45](#)
- sien [3-41](#)
- Single Bit Error Counter [3-44](#)
- single word, definition [xxiv](#)
- single-beat reads/writes [3-7](#)
- small-endian [xxiv](#)
- soft reset [4-8](#)
- software considerations [3-58](#)
- sources of reset [4-7](#)
- specifications, related [A-4](#)
- spurious vector generation [2-53](#)
- Spurious Vector Register [2-68](#)
- SRAM base address [3-25](#)
- status bit descriptions [3-32](#)
- status bit, definition [xxiv](#)
- strap pins configuration for the PC87308VUL [1-21](#)
- swen [3-45](#)
- Symbios SYM53C825A SCSI controller [4-1](#)

syndrome codes [3-62](#)
System Configuration Register (SYSCR)
 [1-13](#)
System External Cache Control Register
 (SXCCR) [1-18](#)
system register summary [1-12](#)

T

Test SRAM [3-56](#)
tien [3-41](#)
Timer Basecount Registers [2-70](#)
Timer Current Count Registers [2-69](#)
Timer Destination Registers [2-72](#)
Timer Frequency Register [2-68](#)
Timer Vector/Priority Registers [2-71](#)
timers [2-55](#)
timing (DRAM access) [3-7](#), [3-8](#), [3-10](#)
Timing (ROM/Flash Access) [3-11](#), [3-12](#)
true, definition [xxiv](#)
trun [3-52](#)

U

Universe's involvement [4-12](#)
upper/lower chip status bit [3-34](#)
URLs (uniform resource locators) [A-6](#)

V

Vendor ID/ Device ID Registers [2-43](#)
Vendor ID/Device ID Registers [2-24](#)
Vendor Identification Register [2-66](#)
Vendor/Device Register [3-32](#)
VME Geographical Address Register
 (VGAR) [1-27](#)
VMEbus slave map [1-11](#)

W

W83C553 PIB registers [1-20](#)
Watchdog timer
 as type of interrupt [4-2](#)
when MPC devices are big-endian [2-14](#)
when MPC devices are little endian [2-15](#)
word, definition [xxiv](#)

writing to the control registers [3-58](#)

Z

Z8536 CIO port pins [1-30](#)
Z8536/Z85230 access registers [1-29](#)
Z8536/Z85230 registers [1-29](#)