RM7000A

RM7000A™ Microprocessor with On-Chip Secondary Cache

Data Sheet

Proprietary and Confidential

Released

Issue 3, October 2001

Legal Information

Copyright

PMC-2002227 (R3)

Disclaimer

Trademarks

Patents

The technology discussed is protected by one or more of the following Patents.

U.S. Patent Numbers 5,953,748, 5,953,748, 5,953,748

Relevant patent applications and other patents may also exist.

Contacting PMC-Sierra

PMC-Sierra, Inc.
8555 Baxter Place Burnaby, BC
Canada V5A 4V7

Tel: +1 (604) 415-6000
Fax: +1 (604) 415-6200

Document Information: document@pmc-sierra.com
Corporate Information: info@pmc-sierra.com
Technical Support: apps@pmc-sierra.com
Web Site: http://www.pmc-sierra.com

Revision History

Issue No.	Issue Date	Details of Change
3	September 2001	Updated power consumption values.
2	May 2001	Changed pin AC13 SysCmd[2] from active low to high. Added industrial values to Recommended Operating Instructions Added industrial and commercial values to Absolute Maximum Ratings Changed Timer Interrupt Enable/Disable information in Boot Time Mode Stream table Added paragraph to Interrupt Handling section Clarification added to System Interface Parameters Additional information added to Clock Parameter table
1	January 2001	Applied PMC-Sierra template to existing MPD (QED) FrameMaker document. In the Pinout Table, changed all references from IP to INT Section 1, Features, changed High-performance system interface, 133 MHz maximum frequency, multiplexed address/data to 125 MHz. Changed QED references to PMC-Sierra or MIPS. Updated Section 7, Recommended Operating Conditions and Section 9 Power Consumption. Added System Interface Parameter values, Section 10.3, for 350 MHz and 400 MHz CPU speeds per data provided by Mark Scrivener.

Document Conventions

The following conventions are used in this datasheet:

- All signal, pin, and bus names described in the text, such as **ExtRqst*,** are in boldface typeface.
- All bit and field names described in the text, such as ***Interrupt Mask***, are in an italic-bold typeface.
- All instruction names, such as MFHI, are in san serif typeface.

Table of Contents

List of Figures

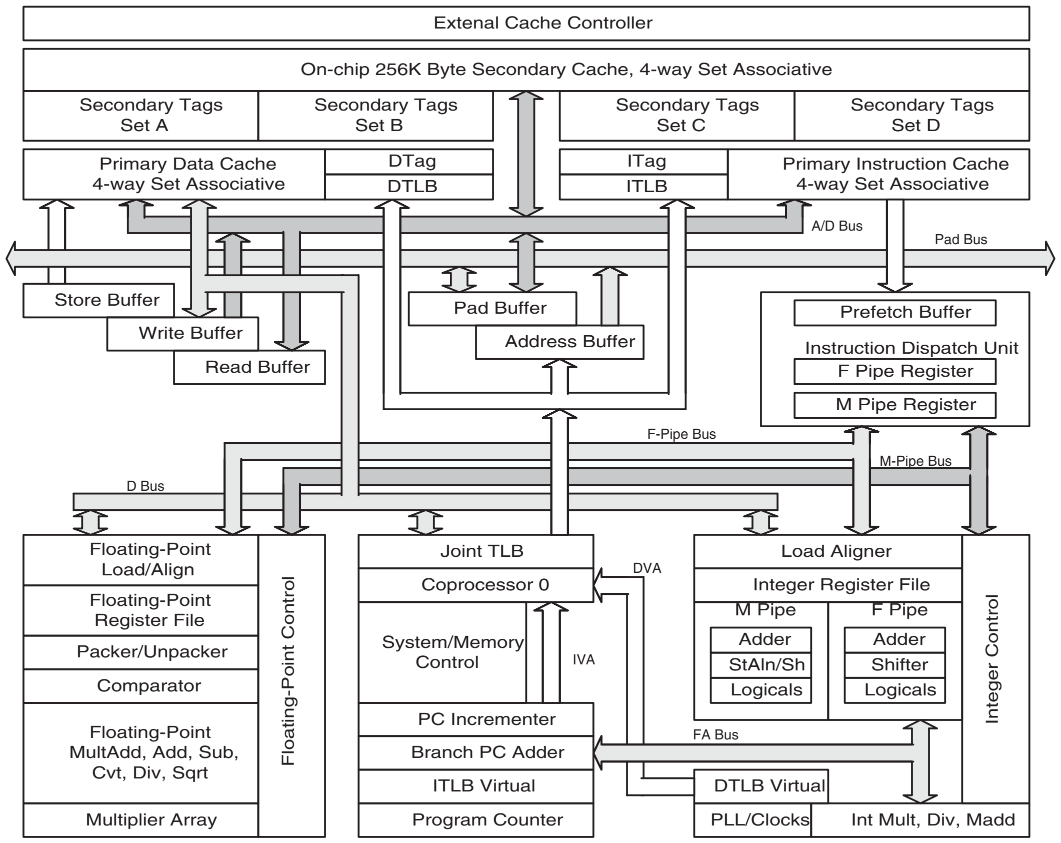
List of Tables

1 Features

- Dual issue symmetric superscalar microprocessor with instruction prefetch optimized for system level price/performance
  - 300, 350, 400 MHz operating frequency
  - >600 Dhrystone 2.1 MIPS @ 400 MHz
- High-performance system interface
  - 1000 MB per second peak throughput
  - 125 MHz max. freq., multiplexed address/data
  - Supports two outstanding reads with out-of-order return
  - Processor clock multipliers 2, 2.5, 3, 3.5, 4, 4.5, 5, 6, 7, 8, 9
- Integrated primary and secondary caches
  - All are 4-way set associative with 32 byte line size
  - 16 KB instruction, 16 KB data, 256 KB on-chip secondary
  - Per line cache locking in primaries and secondary
  - Fast Packet Cache™ increases system efficiency in networking applications
- Integrated external cache controller (up to 8 MB)
- High-performance floating-point unit — 800 MFLOPS maximum
  - Single cycle repeat rate for common single-precision operations and some double-precision operations
  - Single cycle repeat rate for single-precision combined multiply-add operations
  - Two cycle repeat rate for double-precision multiply and double-precision combined multiply-add operations
- MIPS IV superset instruction set architecture
  - Data PREFETCH instruction allows the processor to overlap cache miss latency and instruction execution
  - Single-cycle floating-point multiply-add
- Integrated memory management unit
  - Fully associative joint TLB (shared by I and D translations)
  - 64/48 dual entries map 128/96 pages
  - Variable page size
- Embedded application enhancements
  - Specialized DSP integer Multiply-Accumulate instructions, (MAD/MADU) and three-operand multiply instruction (MUL)
  - I&D Test/Break-point (Watch) registers for emulation & debug
  - Performance counter for system and software tuning & debug
  - Fourteen fully prioritized vectored interrupts — 10 external, 2 internal, 2 software
- Fully static CMOS design with dynamic power down logic
- RM5271 pin compatible, 304 pin TBGA package, 31x31 mm

2 Block Diagram

Figure 1 Block Diagram



3 Description

PMC-Sierra’s RM7000A is a highly integrated symmetric superscalar microprocessor capable of issuing two instructions each processor cycle. It has two high-performance 64-bit integer units as well as a high-throughput, fully pipelined 64-bit floating point unit.

The RM7000A integrates 16 KB 4-way set associative instruction and data caches along with an integrated 256 KB 4-way set associative secondary. The primary data and secondary caches are write-back and non-blocking. An optional external tertiary cache provides high-performance capability even in applications with very large data sets.

The memory management unit contains a 64/48-entry fully associative TLB and a 64-bit system interface supporting multiple outstanding reads with out-of-order return and hardware prioritized and vectored interrupts.

The RM7000A ideally suits high-end embedded control applications such as internetworking, high-performance image manipulation, high-speed printing, and 3-D visualization. The RM7000A is also applicable to the low end workstation market where its balanced integer and floating-point performance and direct support for a large tertiary cache (up to 8 MB) provide outstanding price/ performance.

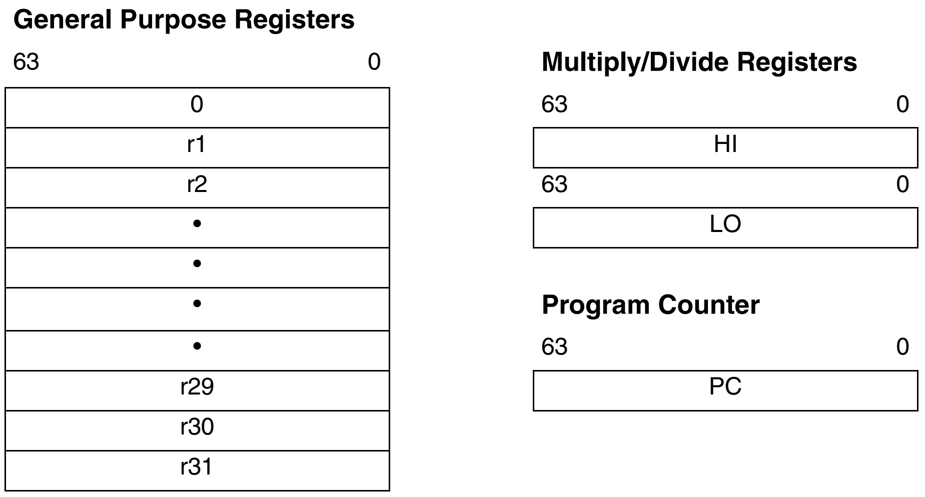
4 Hardware Overview

The RM7000A offers a high-level of integration targeted at high-performance embedded applications. The key elements of the RM7000A are described throughout this section.

4.1 CPU Registers

The RM7000A CPU contains 32 general purpose registers (GPR), two special purpose registers for integer multiplication and division, and a program counter; there are no condition code bits. Figure 2 shows the user visible state.

Figure 2 CP0 Registers



4.2 Superscalar Dispatch

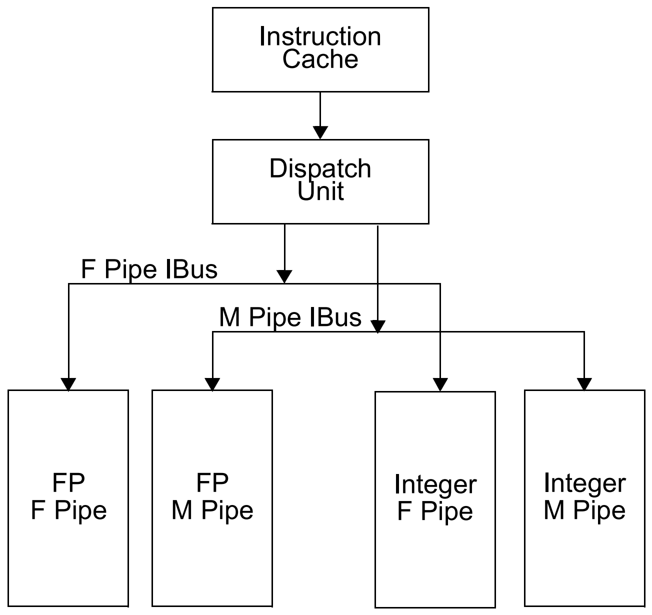
The RM7000A incorporates a superscalar dispatch unit that allows it to issue up to two instructions per cycle. For purposes of instruction issue, the RM7000A defines four classes of instructions: integer, load/store, branches, and floating-point. There are two logical pipelines, the *function*, or F, pipeline and the *memory*, or M, pipeline. Note however that the M pipe can execute integer as well as memory type instructions.

Table 1 Instruction Issue Rules

F Pipe	M Pipe
one of:	one of:
integer, branch, floating-point, integer mul, div	integer, load/store

Figure 2 is a simplification of the pipeline section and illustrates the basics of the instruction issue mechanism.

Figure 3 Instruction Issue Paradigm



The figure illustrates that one F pipe instruction and one M pipe instruction can be issued concurrently but that two M pipe or two F pipe instructions cannot be issued. Table 2 specifies more completely the instructions within each class.

Table 2 Dual Issue Instruction Classes

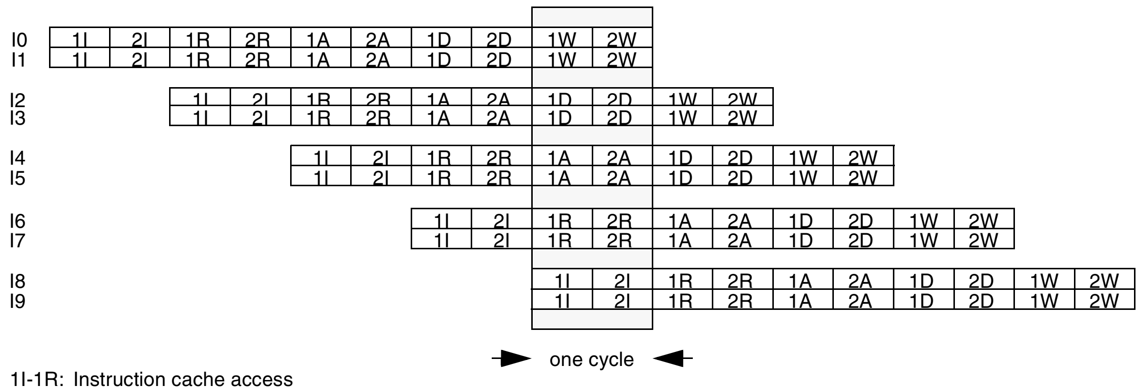
integer	load/store	floating-point	branch
add, sub, or, xor, shift, etc.	lw, sw, ld, sd, ldc1, sdc1, mov, movc, fmov, etc.	fadd, fsub, fmult, fmadd, fdiv, fcmp, fsqrt, etc.	beq, bne, bCzT, bCzF, j, etc.

4.3 Pipeline

The logical length of both the F and M pipelines is five stages with state committing in the register write, or W, pipe stage. The physical length of the floating-point execution pipeline is actually seven stages but this is completely transparent to the user.

Figure 4 shows instruction execution within the RM7000A when instructions are issuing simultaneously down both pipelines. As illustrated in the figure, up to ten instructions can be executing simultaneously. This figure presents a somewhat simplistic view of the processors operation since the out-of-order completion of loads, stores, and long latency floating-point operations can result in there being even more instructions in process than what is shown.

Figure 4 Pipeline



1I-1R:	Instruction cache access
2I:	Instruction virtual to physical address translation
2R:	Register file read, Bypass calculation, Instruction decode, Branch address calculation
1A:	Issue or slip decision, Branch decision
1A:	Data virtual address calculation
1A-2A:	Integer add, logical, shift
2A:	Store Align
2A-2D:	Data cache access and load align
1D:	Data virtual to physical address translation
2W:	Register file write

Note that instruction dependencies, resource conflicts, and branches may result in some of the instruction slots being occupied by NOPs.

4.4 Integer Unit

The RM7000A implements the MIPS IV Instruction Set Architecture. Additionally, the RM7000A includes two implementation specific instructions not found in the baseline MIPS IV ISA, but that are useful in the embedded market place. These instructions are integer multiply-accumulate (MAD) and three-operand integer multiply (MUL).

The RM7000A integer unit includes thirty-two general purpose 64-bit registers, the HI/LO result registers for two-operand integer multiply/divide operations, and the program counter, or PC. There are two separate execution units, one of which can execute function (F) type instructions and one which can execute memory (M) type instructions. Refer to Table 1 for the instruction issue rules.

Note that integer multiply/divide instructions, as well as their corresponding MFHI and MFLO instructions, can only be executed in the F type execution unit. Within each execution unit the operational characteristics are the same as on previous MIPS designs with single cycle ALU operations (add, sub, logical, shift), one cycle load delay, and an autonomous multiply/divide unit.

Register File

The RM7000A has thirty-two general purpose registers with register location 0 (r0) hard wired to a zero value. These registers are used for scalar integer operations and address calculation. In order to service the two integer execution units, the register file has four read ports and two write ports and is fully bypassed both within and between the two execution units to minimize operation latency in the pipeline.

4.5 ALU

The RM7000A has two complete integer ALUs each consisting of an integer adder/subtractor, a logic unit, and a shifter. Table 3 shows the functions performed by the ALUs for each execution unit. Each of these units is optimized to perform all operations in a single processor cycle.

Table 3 ALU Operations

Unit	F Pipe	M Pipe
Adder	add, sub	add, sub, data address add
Logic	logic, moves, zero shifts (nop)	logic, moves, zero shifts (nop)
Shifter	non zero shift	non zero shift, store align

4.6 Integer Multiply/Divide

The RM7000A has a single dedicated integer multiply/divide unit optimized for high-speed multiply and multiply-accumulate operations. The multiply/divide unit resides in the F type execution unit. Table 4 shows the performance of the multiply/divide unit on each operation.

Table 4 Integer Multiply/Divide Operations

Opcode	Operand Size	Latency	Repeat Rate	Stall Cycles
MULT/U, MAD/U	16 bit	4	3	0
	32 bit	5	4	0
MUL	16 bit	4	3	2
	32 bit	5	4	3
DMULT, DMULTU	any	9	8	0
DIV, DIVD	any	36	36	0
DDIV, DDIVU	any	68	68	0

The baseline MIPS IV ISA specifies that the results of a multiply or divide operation be placed in the Hi and Lo registers. These values can then be transferred to the general purpose register file using the Move-from-Hi and Move-from-Lo (MFHI/MFLO) instructions.

In addition to the baseline MIPS IV integer multiply instructions, the RM7000A also implements the 3-operand multiply instruction, MUL. This instruction specifies that the multiply result go directly to the integer register file rather than the Lo register. The portion of the multiply that would have normally gone into the Hi register is discarded. For applications where it is known that the upper half of the multiply result is not required, using the MUL instruction eliminates the necessity of executing an explicit MFLO instruction.

The multiply-add instructions, MAD and MADU, multiply two operands and add the resulting product to the current contents of the Hi and Lo registers. The multiply-accumulate operation is the core primitive of almost all signal processing algorithms. Therefore, using the RM7000A eliminates the need for a separate DSP engine in many embedded applications.

4.7 Floating-Point Coprocessor

The RM7000A incorporates a high-performance fully pipelined floating-point coprocessor which includes a floating-point register file and autonomous execution units for multiply/add/convert and divide/square root. The floating-point coprocessor is a tightly coupled execution unit, decoding and executing instructions in parallel with, and in the case of floating-point loads and stores, in cooperation with the M pipe of the integer unit. The superscalar capabilities of the RM7000A allow floating-point computation instructions to issue concurrently with integer instructions.

4.8 Floating-Point Unit

The RM7000A floating-point execution unit supports single and double precision arithmetic, as specified in the IEEE Standard 754. The execution unit is broken into a separate divide/square root unit and a pipelined multiply/add unit. Overlap of divide/square root and multiply/add is supported.

The RM7000A maintains fully precise floating-point exceptions while allowing both overlapped and pipelined operations. Precise exceptions are extremely important in object-oriented programming environments and highly desirable for debugging in any environment.

Floating-point operations include:

- add
- subtract
- multiply
- divide
- square root
- reciprocal
- reciprocal square root
- conditional moves
- conversion between fixed-point and floating-point format
- conversion between floating-point formats
- floating-point compare

Table 5 gives the latencies of the floating-point instructions in internal processor cycles.

Table 5 Floating Point Latencies and Repeat Rates

Operation	Latency single/double	Repeat Rate single/double
fadd	4	1
fsub	4	1
fmult	4/5	1/2
fmadd	4/5	1/2
fmsub	4/5	1/2
fdiv	21/36	19/34
fsqrt	21/36	19/34
frecip	21/36	19/34
frsqrt	38/68	36/66
fcvt.s.d	4	1
fcvt.s.w	6	3
fcvt.s.l	6	3
fcvt.d.s	4	1
fcvt.d.w	4	1
fcvt.d.l	4	1
fcvt.w.s	4	1
fcvt.w.d	4	1
fcvt.l.s	4	1
fcvt.l.d	4	1
fcmp	1	1
fmov, fmovc	1	1
fabs, fneg	1	1

4.9 Floating-Point General Register File

The floating-point general register file (FGR) is made up of thirty-two 64-bit registers. With the floating-point load and store double instructions, LDC1 and SDC1, the floating-point unit can take advantage of the 64-bit wide data cache and issue a floating-point coprocessor load or store doubleword instruction in every cycle.

The floating-point control register file contains two registers; one for determining configuration and revision information for the coprocessor, and one for control and status information. These registers are primarily used for diagnostic software, exception handling, state saving and restoring, and control of rounding modes.

To support superscalar operations the FGR has four read ports and two write ports and is fully bypassed to minimize operation latency in the pipeline. Three of the read ports and one write port are used to support the combined multiply-add instruction while the fourth read and second write port allows for concurrent floating-point load or store and conditional move operations.

4.10 System Control Coprocessor (CP0)

The system control coprocessor (CP0) is responsible for the virtual memory sub-system, the exception control system, and the diagnostics capability of the processor.

For memory management support, the RM7000A CP0 is logically identical to the RM5200 Family. For interrupt exceptions and diagnostics, the RM7000A is a superset of the RM5200 Family, implementing additional features described in the following sections on Interrupts, Test/ Breakpoint registers, and Performance Counters.

The memory management unit controls the virtual memory system page mapping. It consists of an instruction address translation buffer (ITLB) a data address translation buffer (DTLB), a Joint TLB (JTLB), and coprocessor registers used by the virtual memory mapping sub-system.

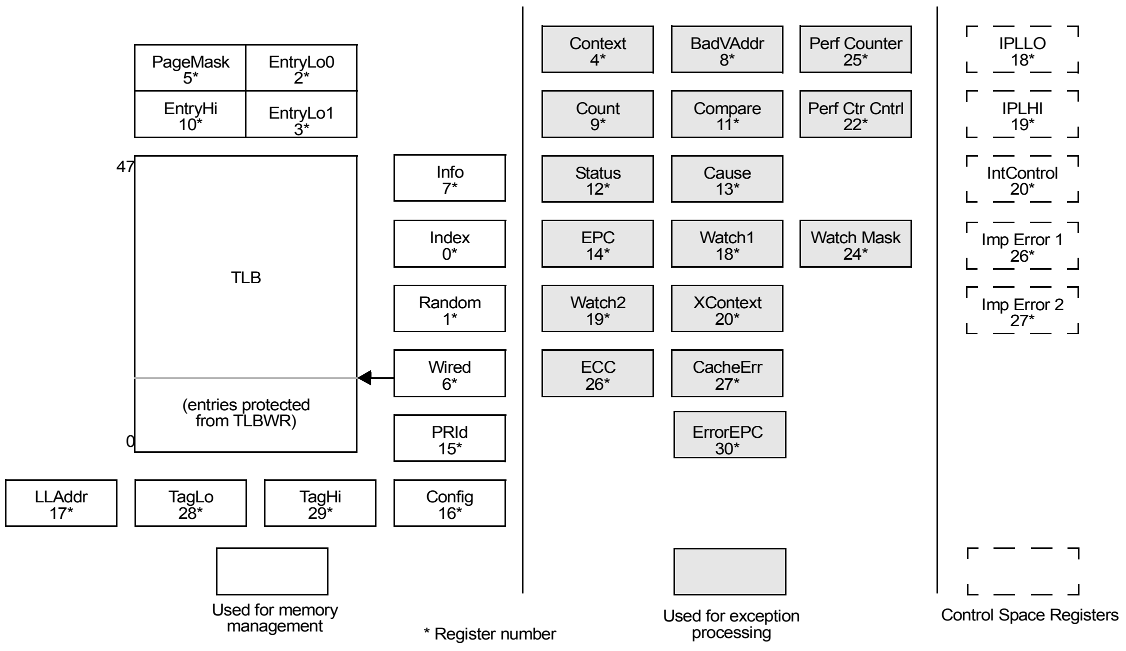
4.11 System Control Coprocessor Registers

The RM7000A incorporates all CP0 registers internally. These registers provide the path through which the virtual memory system's page mapping is examined and modified, exceptions are handled, and operating modes are controlled (kernel vs. user mode, interrupts enabled or disabled, cache features). In addition, the RM7000A includes registers to implement a real-time cycle counting facility, to aid in cache and system diagnostics, and to assist in data error detection.

To support the non-blocking caches and enhanced interrupt handling capabilities of the RM7000A, both the data and control register spaces of CP0 are supported. In the data register space, which is accessed using the MFC0 and MTC0 instructions, the RM7000A supports the same registers as found in the RM5200 Family. In the control space, which is accessed by the previously unused CTC0 and CFC0 instructions, the RM7000A supports five new registers. The first three of these new 32-bit registers support the enhanced interrupt handling capabilities; Interrupt Control, Interrupt Priority Level Lo (IPLLO), and Interrupt Priority Level Hi (IPLHI). These registers are described further in the section on interrupt handling. Two other registers, Imprecise Error 1 and Imprecise Error 2, have been added to help diagnose bus errors that occur on non-blocking memory references.

Figure 5 shows the CP0 registers.

Figure 5 CP0 Registers



4.12 Virtual to Physical Address Mapping

The RM7000A provides three modes of virtual addressing:

- user mode
- kernel mode
- supervisor mode

These modes allow system software to provide a secure environment for user processes. Bits in the CP0 Status register determine which virtual addressing mode is used. In user mode, the RM7000A provides a single, uniform virtual address space of 256 GB (2 GB in 32-bit mode).

When operating in the kernel mode, four distinct virtual address spaces, totalling 1024 GB (4 GB in 32-bit mode), are simultaneously available and are differentiated by the high-order bits of the virtual address.

The RM7000A processor also supports a supervisor mode in which the virtual address space is 256.5 GB (2.5 GB in 32-bit mode), divided into three regions based on the high-order bits of the virtual address. Figure 6 shows the address space layout for 32-bit operations.

Figure 6 Kernel Mode Virtual Addressing (32-bit)

0xFFFFFFFF	Kernel virtual address space
	(kseg3)
0xE0000000	Mapped, 0.5GB
0xDFFFFFFF	Supervisor virtual address space
	(ksseg)
0xC0000000	Mapped, 0.5GB
0xBFFFFFFF	Uncached kernel physical address space
	(kseg1)
0xA0000000	Unmapped, 0.5GB
0x9FFFFFFF	Cached kernel physical address space
	(kseg0)
0x80000000	Unmapped, 0.5GB
0x7FFFFFFF	User virtual address space
	(kuseg)
	Mapped, 2.0GB

When the RM7000A is configured for 64-bit addressing, the virtual address space layout is an upward compatible extension of the 32-bit virtual address space layout.

4.13 Joint TLB

For fast virtual-to-physical address translation, the RM7000A uses a large, fully associative TLB that maps virtual pages to their corresponding physical addresses. As indicated by its name, the JTLB is used for both instruction and data translations. The JTLB is organized as pairs of even/odd entries, and maps a virtual address and address space identifier (ASID) into the large, 64 GB physical address space. By default, the JTLB is configured as 48 pairs of even/odd entries. The optional 64 even/odd entry configuration is set at boot time.

Two mechanisms are provided to assist in controlling the amount of mapped space and the replacement characteristics of various memory regions. First, the page size can be configured, on a per-entry basis, to use page sizes in the range of 4 KB to 16 MB (in 4x multiples). The CP0 PageMask register is loaded with the desired page size of a mapping, and that size is stored into the TLB, along with the virtual address, when a new entry is written. Thus, operating systems can create special purpose maps; for example, an entire frame buffer can be memory mapped using only one TLB entry.

The second mechanism controls the replacement algorithm when a TLB miss occurs. The RM7000A provides a random replacement algorithm to select a TLB entry to be written with a new mapping. However, the processor also provides a mechanism whereby a system specific number of mappings can be locked into the TLB, thereby avoiding random replacement. This mechanism uses the CP0 Wired register and allows the operating system to guarantee that certain pages are always mapped for performance reasons and to avoid a deadlock condition. This mechanism also facilitates the design of real-time systems by allowing deterministic access to critical software.

The JTLB also contains information that controls the cache coherency protocol for each page. Specifically, each page has attribute bits to determine whether the coherency algorithm is:

- uncached
- write-back
- write-through with write-allocate
- write-through without write-allocate
- write-back with secondary and tertiary bypass

Note that both of the write-through protocols bypass both the secondary and the tertiary caches since neither of these caches support writes of less than a complete cache line.

These protocols are used for both code and data on the RM7000A with data using write-back or write-through depending on the application. The write-through modes support the same efficient frame buffer handling as the RM5200 Family.

4.14 Instruction TLB

The RM7000A uses a 4-entry instruction TLB (ITLB). The ITLB offers the following advantages;

- Minimizes contention for the JTLB
- Eliminates the critical path of translating through a large associative array
- Allows instruction address and data address translations to occur in parallel
- Saves power

Each ITLB entry maps a 4 KB page. The ITLB improves performance by allowing instruction address translation to occur in parallel with data address translation. When a miss occurs on an instruction address translation by the ITLB, the least-recently used ITLB entry is filled from the JTLB. The operation of the ITLB is completely transparent to the user.

4.15 Data TLB

The RM7000A uses a 4-entry data TLB (DTLB) for the same reasons cited above for the ITLB. Each DTLB entry maps a 4 KB page. The DTLB improves performance by allowing data address translation to occur in parallel with instruction address translation. When a miss occurs on a data address translation, the DTLB is filled from the JTLB. The DTLB refill is pseudo-LRU; the least recently used entry of the least recently used pair of entries is filled. The operation of the DTLB is completely transparent to the user.

4.16 Cache Memory

The RM7000A contains integrated primary instruction and data caches that support single cycle access, as well as a large unified secondary cache with a three cycle miss penalty from the primary caches. Each primary cache has a 64-bit read path and a 128-bit write path. Both caches can be accessed simultaneously. The primary caches provide the integer and floating-point units with an aggregate bandwidth of 6.4 GB per second at an internal clock frequency of 400 MHz. During an instruction or data primary cache refill, the secondary cache can provide a 64-bit datum every cycle following the initial three cycle latency for a peak bandwidth of 3.6 GB per second. For applications requiring even higher performance, the RM7000A also has a direct interface to a large external tertiary cache.

4.17 Instruction Cache

The RM7000A has an integrated 16 KB, four-way set associative instruction cache that is virtually indexed and physically tagged. The effective physical index eliminates the potential for virtual aliases in the cache.

The data array portion of the instruction cache is 64 bits wide and protected by word parity while the tag array holds a 24-bit physical address, 14 control bits, a valid bit, and a single parity bit.

By accessing 64 bits per cycle, the instruction cache is able to supply two instructions per cycle to the superscalar dispatch unit. For signal processing, graphics, and other numerical code sequences where a floating-point load or store and a floating-point computation instruction are being issued together in a loop, the entire bandwidth available from the instruction cache is consumed by instruction issue. For typical integer code mixes, where instruction dependencies and other resource constraints restrict the level of parallelism that can be achieved, the extra instruction cache bandwidth is used to fetch both the taken and non-taken branch paths to minimize the overall penalty for branches.

A 32-byte (eight instruction) line size is used to maximize the communication efficiency between the instruction cache and the secondary cache, tertiary cache, or memory system.

The RM7000A supports cache locking on a per line basis. The contents of each line of the cache can be *locked* by setting a bit in the Tag RAM. Locking the line prevents its contents from being overwritten by a subsequent cache miss. Refills occur only into unlocked cache lines. This mechanism allows the programmer to lock critical code into the cache, thereby guaranteeing deterministic behavior for the locked code sequence.

4.18 Data Cache

The RM7000A has an integrated 16 KB, four-way set associative data cache that is virtually indexed and physically tagged. Line size is 32 bytes (8 words). The effective physical index eliminates the potential for virtual aliases in the cache.

The data cache is non-blocking; that is, a miss in the data cache does not necessarily stall the processor pipeline. As long as no instruction is encountered which is dependent on the data reference which caused the miss, the pipeline continues to advance. Once there are two cache misses outstanding, the processor stalls if it encounters another load or store instruction.

The data array portion of the data cache is 64 bits wide and protected by byte parity while the tag array holds a 24-bit physical address, 3 control bits, a two-bit cache state field, and two parity bits.

The most commonly used write policy is write-back, which means that a store to a cache line does not immediately cause memory to be updated. This increases system performance by reducing bus traffic and eliminating the bottleneck of waiting for each store operation to finish before issuing a subsequent memory operation. Software can, however, select write-through on a per-page basis when appropriate, such as for frame buffers. Cache protocols supported for the data cache are as follows:

1. Uncached

   Reads to addresses in a memory area identified as uncached do not access the cache. Writes to such addresses are written directly to main memory without updating the cache.

2. Write-back

   Loads and instruction fetches first search the cache, reading the next memory hierarchy level only if the desired data is not cache resident. On data store operations, the cache is first searched to determine if the target address is cache resident. If it is resident, the cache contents are updated and the cache line is marked for later write-back. If the cache lookup misses, the target line is first brought into the cache, afterwhich the write is performed as above.

3. Write-through with write allocate

   Loads and instruction fetches first search the cache, reading from memory only if the desired data is not cache resident; write-through data is never cached in the secondary or tertiary caches. On data store operations, the cache is first searched to determine if the target address is cache resident. If it is resident, the primary cache contents are updated and main memory is written, leaving the *write-back* bit of the cache line unchanged; no writes occur to the secondary or tertiary caches. If the cache lookup misses, the target line is first brought into the cache, afterwhich the write is performed as above.

4. Write-through without write allocate

   Loads and instruction fetches first search the cache, reading from memory only if the desired data is not cache resident; write-through data is never cached in the secondary or tertiary caches. On data store operations, the cache is first searched to determine if the target address is cache resident. If it is resident, the cache contents are updated and main memory is written, leaving the *write-back* bit of the cache line unchanged; no writes occur to the secondary or tertiary caches. If the cache lookup misses, only main memory is written.

5. Fast Packet Cache™ (Write-back with secondary and tertiary bypass)

   Loads and instruction fetches first search the primary cache, reading from memory only if the desired data is not resident; the secondary and tertiary caches are not searched. On data store operations, the primary cache is first searched to determine if the target address is resident. If it is resident, the cache contents are updated, and the cache line marked for later write-back. If the cache lookup misses, the target line is first brought into the cache, afterwhich the write is performed as above.

Associated with the data cache is the *store buffer*. When the RM7000A executes a STORE instruction, this single-entry buffer is written with the store data while the tag comparison is performed. If the tag matches, then the data is written into the data cache in the next cycle that the data cache is not accessed (the next non-load cycle). The store buffer allows the RM7000A to execute a store every processor cycle and to perform back-to-back stores without penalty. In the event of a store immediately followed by a load to the same address, a combined merge and cache write occurs such that no penalty is incurred.

4.19 Secondary Cache

The RM7000A has an integrated 256 KB, four-way set associative, block write-back secondary cache. The secondary cache has a 32-byte line size, a 64-bit bus width to match the system interface and primary cache bus widths, and is protected with doubleword parity. The secondary cache tag array holds a 20-bit physical address, 2 control bits, a three bit cache state field, and two parity bits.

By integrating a secondary cache, the RM7000A is able to decrease the latency of a primary cache miss without significantly increasing the number of pins and the amount of power required by the processor. From a technology point of view, integrating a secondary cache leverages CMOS technology by using silicon to build the structures that are most amenable to silicon technology; building very dense, low power memory arrays rather than large power hungry I/O buffers.

Further benefits of an integrated secondary cache are flexibility in the cache organization and management policies that are not practical with an external cache. Two previously mentioned examples are the 4-way associativity and write-back cache protocol.

A third management policy for which integration affords flexibility is cache hierarchy management. With multiple levels of cache, it is necessary to specify a policy for dealing with cases where two cache lines at level n of the hierarchy could possibly be sharing an entry in level n+1 of the hierarchy.

The RM7000A allows entries to be stored in the primary caches that do not necessarily have a corresponding entry in the secondary; the RM7000A does not force the primaries to be a subset of the secondary. For example, if primary cache line A is being filled and a cache line already exists in the secondary for primary cache line B at the location where primary A's line would reside, then that secondary entry is replaced by an entry corresponding to primary cache line A and no action occurs in the primary for cache line B. This operation creates the aforementioned scenario where the primary cache line, which initially had a corresponding secondary entry, no longer has such an entry. Such a primary line is called an *orphan*. In general, cache lines at level n+1 of the hierarchy are called *parents* of level n's *children*.

Another RM7000A cache management optimization occurs for the case of a secondary cache line replacement where the secondary line is dirty and has a corresponding dirty line in the primary. In this case, since it is permissible to leave the dirty line in the primary, it is not necessary to write the secondary line back to main memory. Taking this scenario one step further, a final optimization occurs when the aforementioned dirty primary line is replaced by another line and must be written back. In this case it is written directly to memory, bypassing the secondary cache.

4.20 Secondary Caching Protocols

Unlike the primary data cache, the secondary cache supports only uncached and block write-back. As noted earlier, cache lines managed with either of the write-through protocols are not placed in the secondary cache. A new caching attribute, *write-back with secondary and tertiary bypass*, allows the secondary, and tertiary caches to be bypassed entirely. When this attribute is selected, the secondary and tertiary caches are not filled on load misses and are not written on dirty write-backs from the primary cache.

4.21 Tertiary Cache

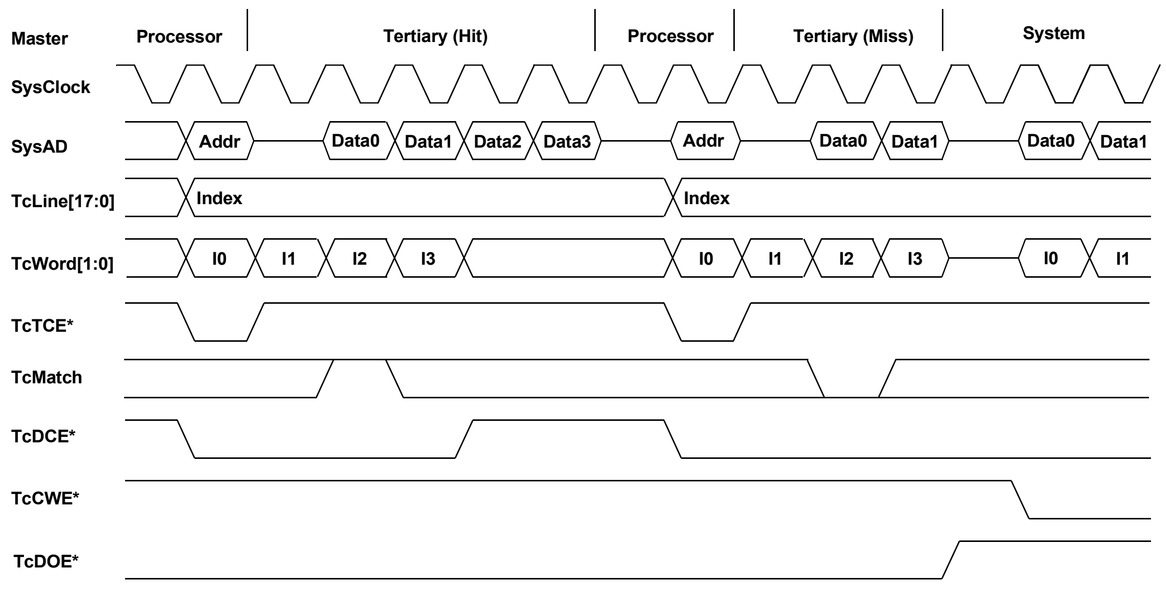
The RM7000A has direct support for an external tertiary cache. The tertiary cache is direct mapped and block write-through with byte parity protection for data. The RM7000A tertiary cache operates identical to the secondary cache of the RM527x while supporting additional size increments to support 4 MB and 8 MB caches.

The tertiary interface uses the **SysAD** bus for data and tags while providing a separate bus, **TcLine[17:0]**, for addresses, along with a number of tertiary cache specific control signals.

A tertiary read looks nearly identical to a standard processor read except that the tag chip enable signal, **TcTCE***, is asserted concurrently with **ValidOut*** and **Release***, initiating a tag probe and indicating to the external controller that a tertiary cache access is being performed. As a result, the external controller monitors the tertiary hit signal, **TcMatch**. If a hit is indicated the controller aborts the memory read and refrains from acquiring control of the system interface. Along with **TcTCE***, the processor also asserts the tag data enable signal, **TcTDE***, which causes the tag RAMs to latch the **SysAD** address internally for use as the replacement tag if a cache miss occurs.

On a tertiary miss, a refill is accomplished with a two signal handshake between the data output enable signal, **TcDOE***, which is deasserted by the controller, and the tag and data write enable signal, **TcCWE***, asserted by the processor. Figure 7 illustrates a tertiary cache hit followed by a miss.

Figure 7 Tertiary Cache Hit and Miss



Other capabilities of the tertiary interface include block write, tag invalidate, and tag probe. For details of these transactions as well as detailed timing waveforms for all the tertiary cache transactions, refer to the RM7000A Bus Interface Specification. The tertiary cache tag can easily be implemented with standard components such as the Motorola MCM69T618.

The RM7000A cache attributes for the instruction, data, internal secondary, and optional external tertiary caches are summarized in Table 6.

Table 6 Cache Attributes

Attribute	Instruction	Data	Secondary	Tertiary
Size	16KB	16KB	256KB	512K, 1M, 2M, 4M, or 8M
Associativity	4-way	4-way	4-way	direct mapped
Replacement Algorithm.	cyclic	cyclic	cyclic	direct replacement
Line size	32 byte	32 byte	32 byte	32 byte
Index	$vAddr_{11..0}$	$vAddr_{11..0}$	$pAddr_{15..0}$	$pAddr_{22..0}$
Tag	$pAddr_{35..12}$	$pAddr_{35..12}$	$pAddr_{35..16}$	$pAddr_{35..19}$
Write policy	n.a.	write-back, write-through	block write-back, bypass	block write-through, bypass
read policy	n.a.	non-blocking (2 outstanding)	non-blocking (data only, 2 outstanding)	non-blocking (data only, 2 outstanding)
read order	critical word first	critical word first	critical word first	critical word first
write order	NA	sequential	sequential	sequential
miss restart following:	complete line	first double (if waiting for data)	n.a.	n.a.
Parity	per word	per byte	per doubleword	per byte

4.22 Cache Locking

The RM7000A allows critical code or data fragments to be locked into the primary and secondary caches. The user has complete control over the locking function. For instruction and data fragments in the primary caches, locking is accomplished by setting either or both of the cache lock enable bits and specifying the set in the CP0 ECC register, then executing either a load instruction for data, or a Fill_I cache operation for instructions.

Only sets A and B within each cache can be locked. Locking within the secondary works identically to the primaries using a separate secondary lock enable bit and the same set selection field. As with the primaries, only sets A and B can be locked. Table 7 summarizes the cache locking capabilities.

Table 7 Cache Locking Control

Cache	Lock Enable	Set Select	Activate

Primary I	ECC[27]	ECC[28]=0→A
		ECC[28]=1→B
			Fill_I

Primary D	ECC[26]	ECC[28]=0→A
		ECC[28]=1→B
			Load/Store

Secondary	ECC[25]	ECC[28]=0→A
		ECC[28]=1→B
			Fill_I or
			Load/Store

4.23 Cache Management

To improve the performance of critical data movement operations in the embedded environment, the RM7000A significantly improves the speed of operation of certain critical cache management operations. In particular, the speed of the Hit-Writeback-Invalidate and Hit-Invalidate cache operations has been improved, in some cases by an order of magnitude, over that of other MIPS processors. For example, Table 8 compares the RM7000A with the R4000 processor.

Table 8 Penalty Cycles

Operation	Condition	Penalty	
		RM7000A	R4000
Hit-Writeback-Invalidate	Miss	0	7
	Hit-Clean	3	12
	Hit-Dirty	3+n	14+n
Hit-Invalidate	Miss	0	7
	Hit	2	9

For the Hit-Dirty case of Hit-Writeback-Invalidate in Table 8 above, if the writeback buffer is full from some previous cache eviction, then n is the number of cycles required to empty the writeback buffer. If the buffer is empty then n is zero.

The penalty value in Table 8 is the number of processor cycles beyond the one cycle required to issue the instruction that is required to implement the operation.

4.24 Primary Write Buffer

Writes to secondary cache or external memory, whether cache miss write-backs or stores to uncached or write-through addresses, use the integrated primary write buffer. The write buffer holds up to four 64-bit address and data pairs. The entire buffer is used for a data cache write-back and allows the processor to proceed in parallel with memory update. For uncached and write-through stores, the write buffer significantly increases performance by decoupling the **SysAD** bus transfers from the instruction execution stream.

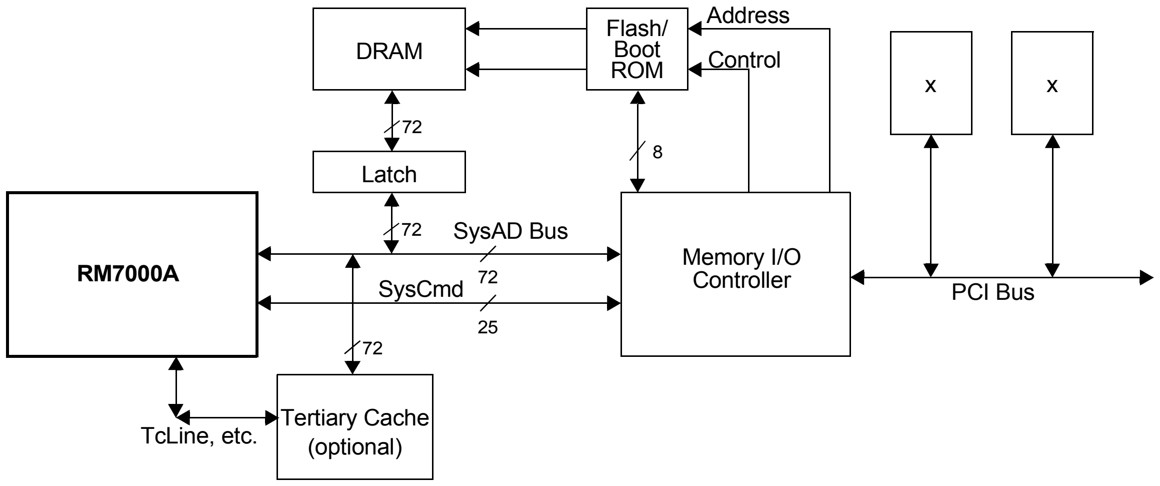
4.25 System Interface

The RM7000A provides a high-performance 64-bit system interface which is compatible with the RM5200 Family. As an enhancement to the **SysAD** bus interface, the RM7000A allows halfintegral clock multipliers, thereby providing greater granularity when selecting pipeline and system interface frequencies.

The **SysAD** interface consists of a 64-bit Address/Data bus with 8 check bits and a 9-bit command bus. In addition, there are ten handshake signals and ten interrupt inputs. The interface is capable of transferring data between the processor and memory at a peak rate of 1000 MB/sec with a 125 MHz SysClock.

Figure 8 shows a typical embedded system using the RM7000A. This example shows a system with a bank of DRAMs, an optional tertiary cache, and an interface ASIC which provides DRAM control as well as an I/O port.

Figure 8 Typical Embedded System Block Diagram



4.26 System Address/Data Bus

The 64-bit System Address Data (SysAD) bus is used to transfer addresses and data between the RM7000A and the rest of the system. It is protected with an 8-bit parity check bus, **SysADC[7:0]**.

The system interface is configurable to allow easy interfacing to memory and I/O systems of varying frequencies. The data rate and the bus frequency at which the RM7000A transmits data to the system interface are programmable at boot time via mode control bits. In addition, the rate at which the processor receives data is fully controlled by the external device. Therefore, either a low cost interface requiring no read or write buffering, or a faster, high-performance interface can be designed to communicate with the RM7000A.

4.27 System Command Bus

The RM7000A interface has a 9-bit System Command bus, **SysCmd[8:0]**. The command bus indicates whether the **SysAD** bus carries address or data information on a per-clock basis. If the **SysAD** bus carries address, the **SysCmd** bus indicates the transaction type (for example, a read or write). If the **SysAD** bus carries data, then the **SysCmd** bus contains information about the data (for example, this is the last data word transmitted, or the data contains an error). The **SysCmd** bus is bidirectional to support both processor requests and external requests to the RM7000A.

Processor requests are initiated by the RM7000A and responded to by an external device. External requests are issued by an external device and require the RM7000A to respond.

The RM7000A supports one- to eight-byte transfers as well as 32-byte block transfers on the **SysAD** bus. In the case of a sub-doubleword transfer, the 3 low-order address bits give the byte address of the transfer, and the **SysCmd** bus indicates the number of bytes being transferred.

4.28 Handshake Signals

There are ten handshake signals on the system interface. Two of these, **RdRdy*** and **WrRdy***, are driven by an external device to indicate to the RM7000A whether it can accept a new read or write transaction. The RM7000A samples these signals before deasserting the address on read and write requests.

ExtRqst* and **Release*** are used to transfer control of the **SysAD** and **SysCmd** buses from the processor to an external device. When an external device requires control of the bus, it asserts **ExtRqst***. The RM7000A responds by asserting **Release*** to release the system interface to slave state.

PRqst* and **PAck*** are used to transfer control of the **SysAD** and **SysCmd** buses from the external agent to the processor. These two pins have been added to the **SysAD** interface to support multiple outstanding reads and facilitate non-blocking caches. When the processor needs to reacquire control of the interface, it asserts **PRqst***. The external device responds by asserting **PAck*** to return control of the interface to the processor.

RspSwap* is also a new pin and is used by the external agent to indicate to the processor when it is returning data out of order. For example, when there are two outstanding reads, the external agent asserts **RspSwap*** when it is going to return the data for the second read before it returns the data for the first read. **RspSwap*** must be asserted by the external agent two cycles ahead of when it presents data so that the processor has time to switch to the correct address for writes into the tertiary cache.

RdType is another new pin on the interface that indicates whether a read is an instruction read or a data read. When asserted, the reference is an instruction read. When deasserted it is a data read. **RdType** is only valid during valid address cycles.

ValidOut* and **ValidIn*** are used by the RM7000A and the external device respectively to indicate that there is a valid command or data on the **SysAD** and **SysCmd** buses. The RM7000A asserts **ValidOut*** when it is driving these buses with a valid command or data, and the external device drives **ValidIn*** when it has control of the buses and is driving a valid command or data.

4.29 System Interface Operation

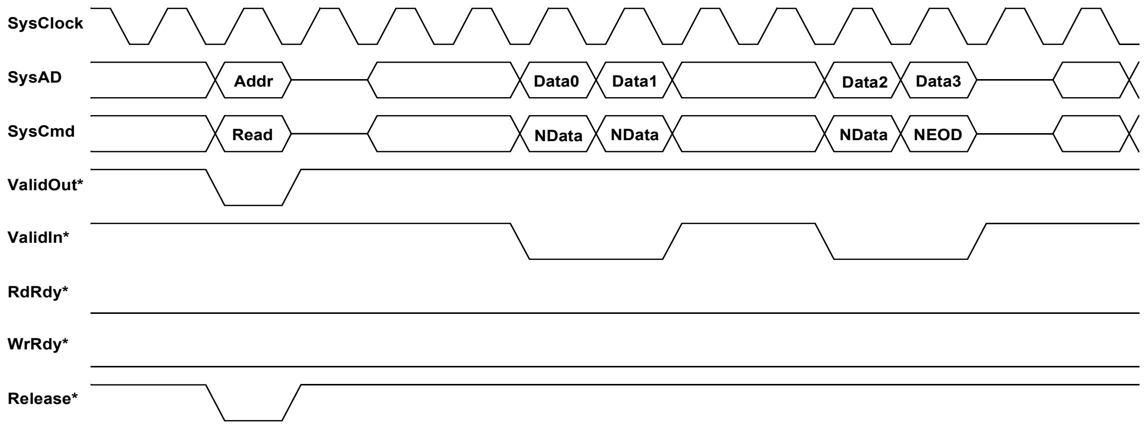
To support non-blocking caches and data prefetch instructions, the RM7000A allows two outstanding reads. An external device may respond to read requests in whatever order it chooses by using the response order indicator pin **RspSwap***. No more than two read requests are submitted to the external device. Support for multiple outstanding reads can be enabled or disabled via a boot-time mode bit. Refer to Table 16 for a complete list of mode bits.

The RM7000A can issue read and write requests to an external device, while an external device can issue null and write requests to the RM7000A.

For processor reads, the RM7000A asserts **ValidOut*** and simultaneously drives the address and read command on the **SysAD** and **SysCmd** buses. If the system interface has **RdRdy*** asserted, then the processor tristates its drivers and releases the system interface to slave state by asserting **Release***. The external device can then begin sending data to the RM7000A.

Figure 9 shows a processor block read request and the external agent read response for a system with either no tertiary cache or a transaction where the tertiary is being bypassed.

Figure 9 Processor Block Read



In Figure 9 the read latency is 4 cycles (**ValidOut*** to **ValidIn***), and the response data pattern is DDxxDD. Figure 10 shows a processor block write where the processor was programmed with write-back data rate boot code 2, or DDxxDDxx.

Finally, Figure 11 shows a typical sequence resulting in two outstanding reads both with initial tertiary cache accesses, as explained in the following sequence.

1. The processor issues a read which misses in the tertiary cache.
2. The external agent takes control of the bus in preparation for returning data to the processor.
3. The processor encounters another internal cache miss and therefore asserts **PRqst*** in order to regain control of the bus.
4. The external agent pulses **PAck***, returning control of the bus to the processor.
5. The processor issues a read for the second miss.
6. The second cycle also misses in the tertiary.
7. The **RspSwap*** pin is asserted to denote the out of order response. Not shown in the figure is the completion of the data transfer for the second miss, or any of the data transfer for the first miss.
8. The external agent retakes control of the bus and begins returning data (out of order) for the second miss to the processor

Figure 10 Processor Block Write

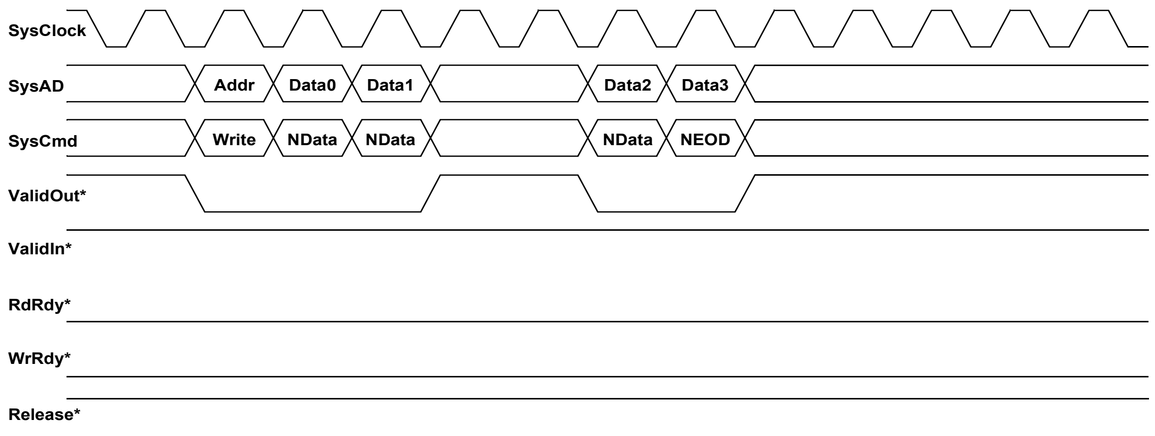
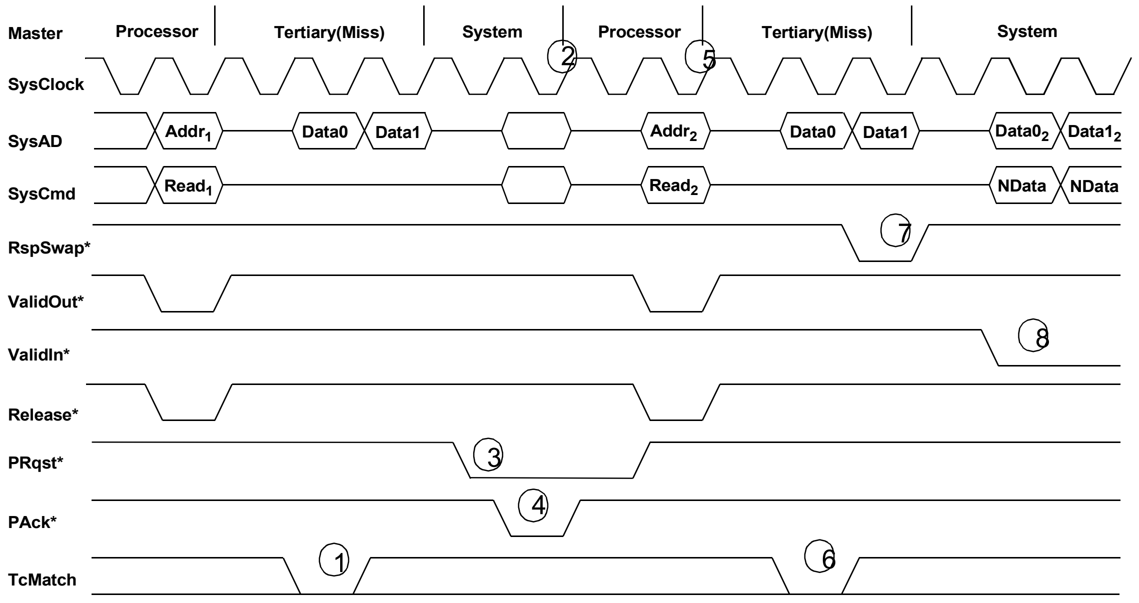


Figure 11 Multiple Outstanding Reads



4.30 Data Prefetch

The RM7000A is the first PMC-Sierra design to support the MIPS IV integer data prefetch (PREF) and floating-point data prefetch (PREFX) instructions. These instructions are used by the compiler or by an assembly language programmer when it is known or suspected that an upcoming data reference is going to miss in the cache. By appropriately placing a prefetch instruction, the memory latency can be hidden under the execution of other instructions. In cases where the execution of a prefetch instruction would cause a memory management or address error exception the prefetch is treated as a NOP.

The “Hint” field of the data prefetch instruction is used to specify the action taken by the instruction. The instruction can operate normally (that is, fetching data as if for a load operation) or it can allocate and fill a cache line with zeroes on a primary data cache miss.

4.31 Enhanced Write Modes

The RM7000A implements two enhancements to the original R4000 write mechanism: Write Reissue and Pipeline Writes. The original R4000 allowed a write on the **SysAD** bus every four **SysClock** cycles. Hence for a non-block write, this meant that two out of every four cycles were wait states.

Pipelined write mode eliminates these two wait states by allowing the processor to drive a new write address onto the bus immediately after the previous data cycle. This allows for higher **SysAD** bus utilization. However, at high frequencies the processor may drive a subsequent write onto the bus prior to the time the external agent deasserts **WrRdy***, indicating that it can not accept another write cycle. This can cause the cycle to be aborted.

Write reissue mode is an enhancement to pipelined write mode and allows the processor to reissue aborted write cycles. If **WrRdy*** is deasserted during the issue phase of a write operation, the cycle is aborted by the processor and reissued at a later time.

In write reissue mode, a rate of one write every two bus cycles can be achieved. Pipelined writes have the same two bus cycle write repeat rate, but can issue one additional write following the deassertion of **WrRdy***.

4.32 External Requests

The RM7000A can respond to certain requests issued by an external device. These requests take one of two forms: *Write* requests and *Null* requests. An external device executes a *write* request when it wishes to update one of the processors writable resources such as the internal interrupt register. A *null* request is executed when the external device wishes the processor to reassert ownership of the processor external interface. Once the external device has acquired control of the processor interface via **ExtRqst***, it can execute a null request after completing an independent transaction between itself and system memory in a system where memory is connected directly to the **SysAD** bus. Normally this transaction would be a DMA read or write from the I/O system.

4.33 Test/Breakpoint Registers

To facilitate hardware and software debugging, the RM7000A incorporates a pair of Test/Breakpoint, or Watch registers, called Watch1 and Watch2, Each Watch register can be separately enabled to watch for a load address, a store address, or an instruction address. All address comparisons are done on physical addresses. An associated register, Watch Mask, has also been added so that either or both of the Watch registers can compare against an address range rather than a specific address. The range granularity is limited to a power of two.

When enabled, a match of either Watch register results in an exception. If the Watch is enabled for a load or store address then the exception is the Watch exception as defined for the R4000 by Cause exception code twenty-three. If the Watch is enabled for instruction addresses then a newly defined *Instruction Watch* exception is taken and the Cause code is sixteen. The Watch register which caused the exception is indicated by Cause bits 25:24. Table 9 summarizes a Watch operation.

Table 9 Watch Control Register

Register	Bit Field/Function					
	63	62	61	60:36	35:2	1:0
Watch1, 2	Store	Load	Instr	0	Addr	0
	31:2				1	0
Watch Mask	Mask				Mask Watch 2	Mask Watch 1

Note that the *W1* and *W2* bits of the Cause register indicate which Watch register caused a particular Watch exception.

4.34 Performance Counters

To facilitate system tuning, the RM7000A implements a performance counter using two new CP0 registers, PerfCount and PerfControl. The PerfCount register is a 32-bit writable counter which causes an interrupt when bit 31 is set. The PerfControl register is a 32-bit register containing a 5-bit field which selects one of twenty-two event types as well as a handful of bits which control the overall counting function. Note that only one event type can be counted at a time and that counting can occur for user code, kernel code, or both. The event types and control bits are listed in Table 10.

Table 10 Performance Counter Control

PerfControl Field	Description
4:0	Event Type 00: Clock cycles 01: Total instructions issued 02: Floating-point instructions issued 03: Integer instructions issued 04: Load instructions issued 05: Store instructions issued 06: Dual issued pairs 07: Branch prefetches 08: External Cache Misses 09: Stall cycles 0A: Secondary cache misses 0B: Instruction cache misses 0C: Data cache misses 0D: Data TLB misses 0E: Instruction TLB misses 0F: Joint TLB instruction misses 10: Joint TLB data misses 11: Branches taken 12: Branches issued 13: Secondary cache writebacks 14: Primary cache writebacks 15: Dcache miss stall cycles (cycles where both cache miss tokens taken and a third address is requested) 16: Cache misses 17: FP possible exception cycles 18: Slip Cycles due to multiplier busy 19: Coprocessor 0 slip cycles 1A: Slip cycles doe to pending non-blocking loads 1B: Write buffer full stall cycles 1C: Cache instruction stall cycles 1D: Multiplier stall cycles 1E: Stall cycles due to pending non-blocking loads - stall start of exception
7:5	Reserved (must be zero)
8	Count in Kernel Mode 0: Disable 1: Enable
9	Count in User Mode 0: Disable 1: Enable
10	Count Enable 0: Disable 1: Enable
31:11	Reserved (must be zero)

The performance counter interrupt only occurs when interrupts are enabled in the Status register, IE=1, and the ***Interrupt Mask*** bit 13 (***IM[13]***) of the coprocessor 0 interrupt control register is set.

Since the performance counter can be set up to count clock cycles, it can be used as either a second timer, or a watchdog interrupt. A watchdog interrupt can be used as an aid in debugging system or software "hangs." Typically the software is setup to periodically update the count so that no interrupt occurs. When a hang occurs the interrupt ultimately triggers, thereby breaking free from the hang-up.

4.35 Interrupt Handling

In order to provide better real time interrupt handling, the RM7000A provides an extended set of hardware interrupts, each of which can be separately prioritized and separately vectored.

In addition to the standard six external interrupt pins, the RM7000A provides four more interrupt pins for a total of ten external interrupts.

As described above, the performance counter is also a hardware interrupt source using **INT[13]**. Historically in the MIPS architecture, interrupt 7 (**INT[7]**) was used as the timer interrupt. The RM7000A provides a separate interrupt, **INT[12]**, for this purpose, thereby releasing **INT[7]** for use as a pure external interrupt.

All interrupts (**INT[13:0]**), the Performance Counter, and the Timer, have corresponding interrupt mask bits, ***IM[13..0]***, and interrupt pending bits, ***IP[13..0]***, in the Status, Interrupt Control, and Cause registers. The bit assignments for the Interrupt Control and Cause registers are shown in Table 11 and Table 12. The Status register has not changed from the RM5200 Family and is not shown.

The *IV* bit in the Cause register is the global enable bit for the enhanced interrupt features. If this bit is clear then interrupt operation is compatible with the RM5200 Family.

In the Interrupt Control register, the interrupt vector spacing is controlled by the Spacing field as described below. The ***Interrupt Mask*** field (***IM[15:8])*** contains the interrupt mask for interrupts eight through thirteen. ***IM[15:14]*** are reserved for future use.

The Timer Enable (TE) bit is used to gate the Timer Interrupt to the Cause Register. If TE is set to 0, the Timer Interrupt is not gated to IP[12]. If TE is set to 1, the Timer Interrupt is gated to IP[12].

The setting for Mode Bit 11 is used to determine if the Timer Interrupt replaces the external Interrupt (Int[5]*) as an input to IP[7] in the Cause Register. If Mode Bit 11 is set to 1, the Timer Interrupt is gated to IP[7].

In order to utilize both the external Interrupt (Int[5]*) and the internal Timer Interrupt, Mode Bit 11 must be set to 0, and TE must be set to 1. In this case, the Timer Interrupt will utilize IP[12], and Int[5]* will utilize IP[7]. Please also reference the logic diagram for interrupt signals in the RM7000 User Manual.

The Interrupt Control register uses ***IM13*** to enable the Performance Counter Control.

Priority of the interrupts is set via two new coprocessor 0 registers called Interrupt Priority Level Lo (IPLLO) and Interrupt Priority Level Hi (IPLHI).

Table 11 Cause Register

31	30	29,28	27	26	25	24	23..8	7	6..2	0,1
BD	0	CE	0	W2	W1	IV	IP[15..0]	0	EXC	0

Table 12 Interrupt Control Register

31..16	15..8	7	6..5	4..0
0	IM[15..8]	TE	0	Spacing

Table 13 IPLLO Register

31..28	27..24	23..20	19..16	15..12	11..8	7..4	3..0
IPL7	IPL6	IPL5	IPL4	IPL3	IPL2	IPL1	IPL0

Table 14 IPLHI Register

31..28	27..24	23..20	19..16	15..12	11..8	7..4	3..0
0	0	IPL13	IPL12	IPL11	IPL10	IPL9	IPL8

In the IPLLO and IPLHI registers, each interrupt is represented by a four-bit field, thereby allowing each interrupt to be programmed with a priority level from 0 to 13 inclusive. The priorities can be set in any manner, including having all the priorities set exactly the same. Priority 0 is the highest level and priority 15 the lowest. The format of the priority level registers is shown in Table 13 and Table 14 above. The priority level registers are located in the coprocessor 0 control register space.

In addition to programmable priority levels, the RM7000A also permits the spacing between interrupt vectors to be programmed. For example, the minimum spacing between two adjacent vectors is 0x20 while the maximum is 0x200. This programmability allows the user to either set up the vectors as jumps to the actual interrupt routines or, if interrupt latency is paramount, to include the entire interrupt routine at one vector. Table 15 illustrates the complete set of vector spacing selections along with the coding as required in the ***Interrupt Control register*** bits 4:0.

In general, the active interrupt priority, combined with the spacing setting, generates a vector offset which is then added to the interrupt base address of 0x200 to generate the interrupt exception offset. This offset is then added to the exception base to produce the final interrupt vector address.

Table 15 Interrupt Vector Spacing

ICR[4..0]	Spacing
0x0	0x000
0x1	0x020
0x2	0x040
0x4	0x080
0x8	0x100
0x10	0x200
others	reserved

4.36 Standby Mode

The RM7000A provides a means to reduce the amount of power consumed by the internal core when the CPU is not performing any useful operations. This state is known as Standby Mode.

Executing the WAIT instruction enables interrupts and causes the processor to enter Standby Mode. If the **SysAD** bus is currently idle when the WAIT instruction completes the W pipe stage, the internal processor clock stops, thereby freezing the pipeline. The phase lock loop, or PLL, internal timer/counter, and the "wake up" input pins: **INT[9.0]***, **NMI***, **ExtReq***, **Reset***, and **ColdReset*** continue to operate in their normal fashion.

If the **SysAD** bus is not idle when the WAIT instruction completes the W pipe stage, then the WAIT is treated as a NOP until the bus operation is completed. Once the processor is in Standby, any interrupt, including the internally generated timer interrupt, causes the processor to exit Standby and resume operation where it left off. The WAIT instruction is typically inserted in the idle loop of the operating system or real time executive.

4.37 JTAG Interface

The RM7000A interface supports JTAG boundary scan in conformance with IEEE 1149.1. The JTAG interface is useful for checking the integrity of the processor's pin connections.

4.38 Boot-Time Options

The RM7000A operating modes are initialized at power-up by the boot-time mode control interface. The serial boot-time mode control interface operates at a very low frequency (**SysClock** divided by 256), allowing the initialization information to be kept in a low cost EPROM or system interface ASIC.

4.39 Boot-Time Modes

The boot-time serial mode stream is defined in Table 16. Bit 0 is presented to the processor as the first bit in the stream when **VccOK** is de-asserted. Bit 255 is the last bit transferred.

Table 16 Boot Time Mode Stream

Mode bit	Description	Mode bit	Description
0	reserved (must be zero)	17:16	System configuration identifiers - software visible in processor Config[21..20] register
4:1	Write-back data rate 0: DDDD 1: DDxDDx 2: DDxxDDxx 3: DxDxDxDx 4: DDxxxDDxxx 5: DDxxxxDDxxxx 6: DxxDxxDxxDxx 7: DDxxxxxxDDxxxxxx 8: DxxxDxxxDxxxDxxx 9-15: reserved	19:18	Reserved: Must be zero
7:5	SysClock to Pclock Multiplier Mode bit 20 = 0 / Mode bit 20 = 1 0: Multiply by 2/x 1: Multiply by 3/x 2: Multiply by 4/x 3: Multiply by 5/2.5 4: Multiply by 6/x 5: Multiply by 7/3.5 6: Multiply by 8/x 7: Multiply by 9/4.5	20	Pclock to SysClock multipliers. 0: Integer multipliers (2,3,4,5,6,7,8,9) 1: Half integer multipliers (2.5,3.5,4.5)
8	Specifies byte ordering. Logically ORed with BigEndian input signal. 0: Little endian 1: Big endian	23:21	Reserved: Must be zero
10:9	Non-Block Write Control 00: R4000 compatible non-block writes 01: reserved 10: pipelined non-block writes 11: non-block write re-issue	24	JTLB Size. 0: 48 dual-entry 1: 64 dual-entry
11	Timer Interrupt Enable/Disable 0: External Int[5]* gated to IP[7] 1: Internal timer Interrupt gated to IP[7]	25	On-chip secondary cache control. 0: Disable 1: Enable
12	Enable the external tertiary cache 0: Disable 1: Enable	26	Enable two outstanding reads with out-of-order return 0: Disable 1: Enable
14:13	Output driver strength - 100% = fastest 00: 67% strength 01: 50% strength 10: 100% strength 11: 83% strength	255:27	Reserved: Must be zero
15	External Tertiary cache RAM type: 0: Dual-cycle deselect (DCD) 1: Single-cycle deselect (SCD)		

5 Pin Descriptions

The following is a list of control, data, clock, tertiary cache, interrupt, and miscellaneous pins of the RM7000A.

Table 17 System Interface

Pin Name	Type	Description
ExtRqst*	Input	External request Signals that the system interface is submitting an external request.
Release*	Output	Release interface Signals that the processor is releasing the system interface to slave state
RdRdy*	Input	Read Ready Signals that an external agent can now accept a processor read.
WrRdy*	Input	Write Ready Signals that an external agent can now accept a processor write request.
ValidIn*	Input	Valid Input Signals that an external agent is now driving a valid address or data on the SysAD bus and a valid command or data identifier on the SysCmd bus.
ValidOut*	Output	Valid output Signals that the processor is now driving a valid address or data on the SysAD bus and a valid command or data identifier on the SysCmd bus.
PRqst*	Output	Processor Request When asserted this signal requests that control of the system interface be returned to the processor. This is enabled by Mode Bit 26.
PAck*	Input	Processor Acknowledge When asserted, in response to PRqst*, this signal indicates to the processor that it has been granted control of the system interface.
RspSwap*	Input	Response Swap RspSwap* is used by the external agent to signal the processor when it is about to return a memory reference out of order; i.e., of two outstanding memory references, the data for the second reference is being returned ahead of the data for the first reference. In order that the processor will have time to switch the address to the tertiary cache, this signal must be asserted a minimum of two cycles prior to the data itself being presented. Note that this signal works as a toggle; i.e., for each cycle that it is held asserted the order of return is reversed. By default, anytime the processor issues a second read it is assumed that the reads will be returned in order; i.e., no action is required if the reads are indeed returned in order. This is enabled by Mode Bit 26.
RdType	Output	Read Type During the address cycle of a read request, RdType indicates whether the read request is an instruction read or a data read.

Pin Name	Type	Description
SysAD(63:0)	Input/Output	System address/data bus A 64-bit address and data bus for communication between the processor and an external agent.
SysADC(7:0)	Input/Output	System address/data check bus An 8-bit bus containing parity check bits for the SysAD bus during data cycles.
SysCmd(8:0)	Input/Output	System command/data identifier bus A 9-bit bus for command and data identifier transmission between the processor and an external agent.
SysCmdP	Input/Output	System Command/Data Identifier Bus Parity For the RM7000A, unused on input and zero on output.

Table 18 Clock/Control Interface

Pin Name	Type	Description
SysClock	Input	System clock Master clock input used as the system interface reference clock. All output timings are relative to this input clock. Pipeline operation frequency is derived by multiplying this clock up by the factor selected during boot initialization
VccP	Input	Vcc for PLL Quiet VccInt for the internal phase locked loop. Must be connected to VccInt through a filter circuit.
VssP	Input	Vss for PLL Quiet Vss for the internal phase locked loop. Must be connected to VssInt through a filter circuit.

Table 19 Tertiary Cache Interface

Pin Name	Type	Description
TcCLR*	Output	Tertiary Cache Block Clear Requests that all valid bits be cleared in the Tag RAMs. Many RAMs may not support a block clear therefore the block clear capability is not required for the cache to operate.
TcCWE*(1:0)	Output	Tertiary Cache Write Enable Asserted to cause a write to the cache. Two identical signals are provided to balance the capacitive load relative to the remaining cache interface signals.
TcDCE*(1:0)	Output	Tertiary Cache Data RAM Chip Enable When asserted this signal causes the data RAMs to read out their contents. Two identical signals are provided to balance the capacitive load relative to the remaining cache interface signals
TcDOE*	Input	Tertiary Cache Data RAM Output Enable When asserted this signal causes the data RAMs to drive data onto their I/O pins. This signal is monitored by the processor to determine when to drive the data RAM write enable in a tertiary cache miss refill sequence.
TcLine(17:0)	Output	Tertiary Cache Line Index
TcMatch	Input	Tertiary Cache Tag Match This signal is asserted by the cache Tag RAMs when a match occurs between the value on its data inputs and the contents of the addressed location in the RAM.
TcTCE*	Output	Tertiary Cache Tag RAM Chip Enable When asserted this signal will cause either a probe or a write of the Tag RAMs depending on the state of the Tag RAMs write enable signal. This signal is monitored by the external agent and indicates to it that a tertiary cache access is occurring.
TcTDE*	Output	Tertiary Cache Tag RAM Data Enable When asserted this signal causes the value on the data inputs of the Tag RAM to be latched into the RAM. If a refill of the RAM is necessary, this latched value will be written into the Tag RAM array. Latching the Tag allows a shared address/data bus to be used without incurring a penalty to re-present the Tag during the refill sequence.
TcTOE*	Output	Tertiary Cache Tag RAM Output Enable When asserted this signal causes the Tag RAMs to drive data onto their I/O pins.
TcWord(1:0)	Input/Output	Tertiary Cache Double Word Index Driven by the processor on cache hits and by the external agent on cache miss refills.
TcValid	Input/Output	Tertiary Cache Valid This signal is driven by the processor as appropriate to make a cache line valid or invalid. On Tag read operations the Tag RAM will drive this signal to indicate the line state.

Table 20 Interrupt Interface

Pin Name	Type	Description
Int*(9:0)	Input	Interrupt Ten general processor interrupts, bit-wise ORed with bits 9:0 of the interrupt register.
NMI*	Input	Non-maskable interrupt Non-maskable interrupt, ORed with bit 15 of the interrupt register (bit 6 in R5000 compatibility mode).

Table 21 JTAG Interface

Pin Name	Type	Description
JTDI	Input	JTAG data in JTAG serial data in.
JTCK	Input	JTAG clock input JTAG serial clock input.
JTDO	Output	JTAG data out JTAG serial data out.
JTMS	Input	JTAG command JTAG command signal, signals that the incoming serial data is command data.

Table 22 Initialization Interface

Pin Name	Type	Description
BigEndian	Input	Big Endian / Little Endian Control Allows the system to change the processor addressing mode without rewriting the mode ROM.
VccOK	Input	Vcc is OK When asserted, this signal indicates to the RM7000A that the VccInt power supply has been above the recommended value for more than 100 milliseconds and will remain stable. The assertion of VccOK initiates the reading of the boot-time mode control serial stream.
ColdReset*	Input	Cold Reset This signal must be asserted for a power on reset or a cold reset. ColdReset must be de-asserted synchronously with SysClock.
Reset*	Input	Reset This signal must be asserted for any reset sequence. It may be asserted synchronously or asynchronously for a cold reset, or synchronously to initiate a warm reset. Reset must be de-asserted synchronously with SysClock.
ModeClock	Output	Boot Mode Clock Serial boot-mode data clock output at the system clock frequency divided by two hundred and fifty six.
ModeIn	Input	Boot Mode Data In Serial boot-mode data input.

6 Absolute Maximum Ratings[1]

Symbol	Rating	Limits	Unit
V_{TERM}	Terminal Voltage with respect to VSS	–0.5[2] to +3.9	V
T_{CASE}	Operating Temperature		
	Commercial	0 to +85	°C
	Industrial	–40 to +85	°C
T_{STG}	Storage Temperature	–55 to +125	°C
I_{IN}	DC Input Current[3]	±20	mA
I_{OUT}	DC Output Current[4]	±20	mA

Notes

1. Stresses greater than those listed under ABSOLUTE MAXIMUM RATINGS may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect reliability.
2. V_{IN} minimum = -2.0 V for pulse width less than 15 ns. V_{IN} should not exceed 3.9 Volts.
3. When $V_{IN} < 0V$ or $V_{IN} >$ VccIO
4. Not more than one output should be shorted at a time. Duration of the short should not exceed 30 seconds.

7 Recommended Operating Conditions

Grade	CPU Speed	Temperature	Vss	VccInt	VccIO	VccP
Commercial	300 - 350 MHz	0°C to +85°C (Case)	0V	1.65V ± 50 mV	3.3 V ± 150 mV or 2.5 V ± 200 mV	1.65V ± 50 mV
	400 MHz	0°C to +70°C (Case)	0V	1.8V ± 50 mV	3.3 V ± 150 mV or 2.5 V ± 200 mV	1.8V ± 50 mV
Industrial	350MHz	-40°C to +85°C (Case)	0V	1.65 ± 50 mV	3.3 V ± 150 mV or 2.5 V ± 200 mV	1.65V ± 50 mV

Notes

1. VccIO should not exceed VccInt by greater than 2.0 V during the power-up sequence.
2. Applying a logic high state to any I/O pin before VccInt becomes stable is not recommended.
3. As specified in IEEE 1149.1 (JTAG), the JTMS pin must be held high during reset to avoid entering JTAG test mode. Refer to the RM7000A Family Users Manual, Appendix E.
4. VccP must be connected to VccInt through a passive filter circuit. See RM7000 Family User's Manual for recommended circuit.

8 DC Electrical Characteristics

(V_{cc}IO = 3.15V - 3.45V)

Parameter	Minimum	Maximum	Conditions
V_{OL}		0.2V	$\|I_{OUT}\|$= 100 μA
V_{OH}	VccIO - 0.2V		
V_{OL}		0.4V	$\|I_{OUT}\|$ = 2 mA
V_{OH}	2.4V		
V_{IL}	-0.3V	0.8V	
V_{IH}	2.0V	VccIO + 0.3V	
I_{IN}		±15 μA ±15 μA	V_{IN} = 0 V_{IN} = VccIO

(V_{cc}IO = 2.3V - 2.7V)

Parameter	Minimum	Maximum	Conditions
V_{OL}		0.2V	$\|I_{OUT}\|$= 100 μA
V_{OH}	2.1V		
V_{OL}		0.4V	$\|I_{OUT}\|$ = 1 mA
V_{OH}	2.0		
V_{OL}		0.7V	$\|I_{OUT}\|$ = 2 mA
V_{OH}	1.7		
V_{IL}	-0.3V	0.7V	
V_{IH}	1.7V	VccIO + 0.3V	
I_{IN}		±15 μA ±15 μA	V_{IN} = 0 V_{IN} = VccIO

9 Power Consumption

Parameter		Conditions	CPU Speed		
			300 MHz	350 MHz	400 MHz
			Max[1]	Max[1]	Max[1]
VccInt Power (mWatts)	standby		865	865	925
	active	Maximum with no FPU operation[2]	2350	2750	3550
		Maximum worst case instruction mix	2500	3000	4000

Notes

1. Worst case supply voltage (maximum VccInt) with worst case temperature (maximum TCase).
2. Dhrystone 2.1 instruction mix.
3. I/O supply power is application dependant, but typically <20% of VccInt.

10 AC Electrical Characteristics

10.1 Capacitive Load Deration

Parameter	Symbol	Min	Max	Units
Load Derate	C_{LD}		2	ns/25pF

10.2 Clock Parameters

			CPU Speed						
		Test	300 MHz		350 MHz		400 MHz		
Parameter	Symbol	Conditions	Min	Max	Min	Max	Min	Max	Units
SysClock High	t_{SCHigh}	Transition ≤ 5ns	3		3		3		ns
SysClock Low	t_{SCLow}	Transition ≤ 5ns	3		3		3		ns
SysClock Frequency			33.3	100	33.3	117	33.13	125	MHz
SysClock Period	t_{SCP}		10	30	8.5	30	8	30	ns
Clock Jitter for SysClock	$t_{JitterIn}$			±150		±150		±150	ps
SysClock Rise Time	t_{SCRise}			2		2		2	ns
SysClock Fall Time	t_{SCFall}			2		2		2	ns
ModeClock Period	$t_{ModeCKP}$			256		256		256	t_{SCP}
JTAG Clock Period	$t_{JTAGCKP}$		4		4		4		t_{SCP}

Note: Operation of the RM7000A is only guaranteed with the Phase Lock Loop Enabled.

10.3 System Interface Parameters

Parameter[1]	Symbol	Test Conditions	CPU Speed						Units
			300 MHz		350 MHz		400 MHz		
			Min	Max	Min	Max	Min	Max	
Data Output[2,3]	t_{DO}	mode14..13 = 10[5,6] (fastest)	1.0	4.5	1.0	4.5	1.0	4.5	ns
		mode14..13 = 01[5,6] (slowest)	1.0	5.5	1.0	5.5	1.0	5.5	ns
Data Setup[4]	t_{DS}[6]	t_{rise} = see above table t_{fall} = see above table	2.5		2.5		2.5		ns
Data Hold[4]	t_{DH}		1.0		1.0		1.0		ns

Notes

1. Timings are measured from 0.425 x VccIO of clock to 0.425 x VccIO of signal for 3.3V I/O. Timings are measured from 0.48 x VccIO of clock to 0.48 *x* VccIO of signal for 2.5V I/O.
2. Capacitive load for all maximum output timings is 50 pF. Minimum output timings are for theoretical no load conditions - untested.
3. Data Output timing applies to all signal pins whether tristate I/O or output only.
4. Setup and Hold parameters apply to all signal pins whether tristate I/O or input only.
5. Only mode 14:13 = 10 is tested and guaranteed.
6. Data shown is for 3.3 V I/O. For 2.5 V I/O derate all times by .5 nS.

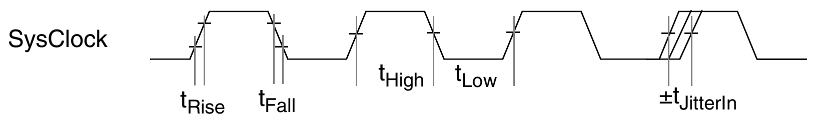
10.4 Boot-Time Interface Parameters

Parameter	Symbol	Min	Max	Units
Mode Data Setup	t_{DS}	4		SysClock cycles
Mode Data Hold	t_{DH}	0		SysClock cycles

11 Timing Diagrams

11.1 Clock Timing

Figure 12 Clock Timing



System Interface Timing (SysAD, SysCmd, ValidIn*, ValidOut*, etc.)

Figure 13 Input Timing

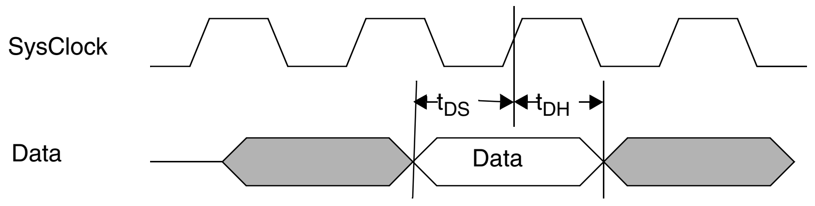
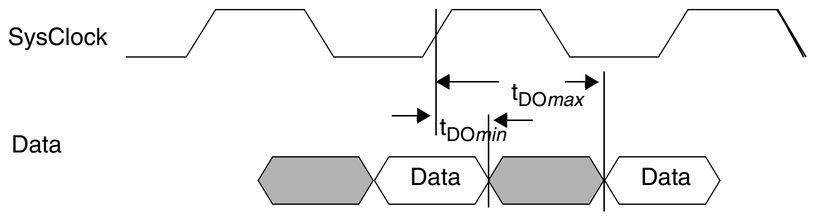
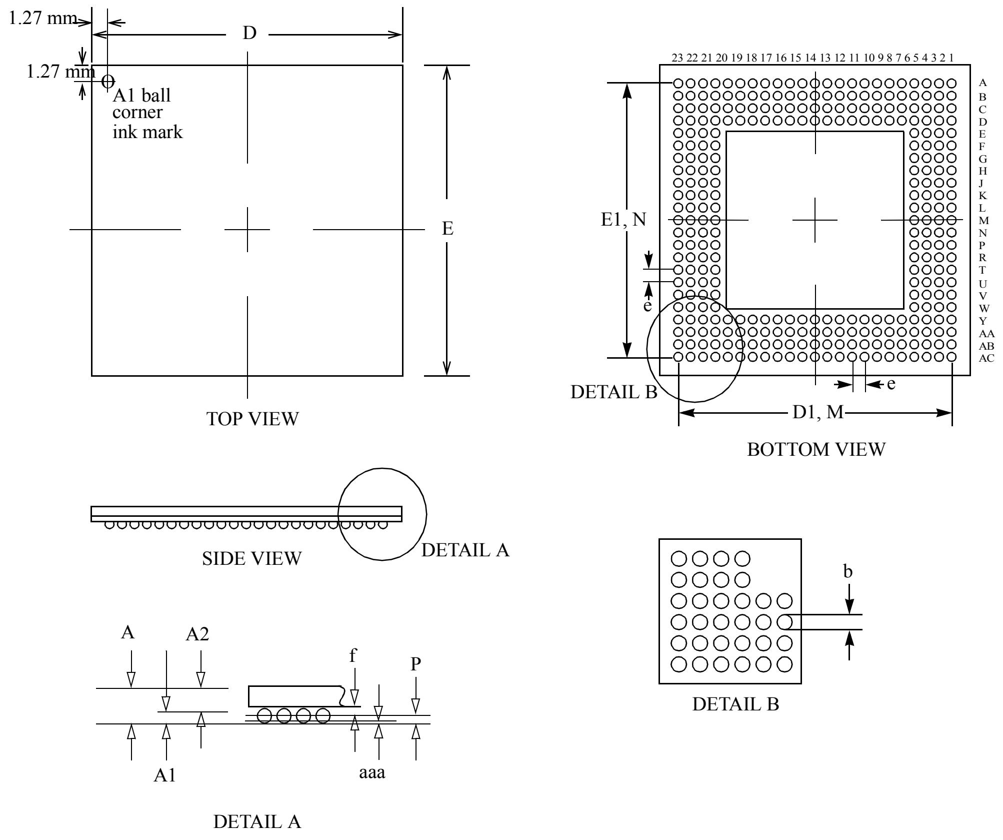


Figure 14 Output Timing



12 Packaging Information

Figure 15 304 TBGA Drawing



Body Size: 31.0 x 31.0 mm Package

Symbol	Min	Nominal	Max	Note
A	1.45	1.55	1.65	Overall Thickness
A1	0.60	0.65	0.70	Ball Height
A2	0.85	0.90	0.95	Body Thickness
D, E	30.80	31.00	31.20	Body Size
D1, E1		27.94		Ball Footprint
M,N		23 x 23		Ball Matrix
M1		4		Number of Rows Deep
b	0.65	0.75	0.85	Ball Diameter
e		1.27		Ball Pitch
aaa		0.15		Coplanarity
bbb		0.15		Parallel
f	0.30	0.35	0.40	Seating Plan Clearance
P		0.25		Encapsulation Height
Theta JC		0.3		Deg. C/Watt
Theta JA		13		Deg. C/Watt @ 0 cfm air flow.

Note: All dimensions in millimeters unless otherwise indicated.

13 RM7000A Pinout

Pin	Function	Pin	Function	Pin	Function	Pin	Function
A1	VccIO	A2	VssIO	A3	VssIO	A4	TcLine[11]
A5	Do not connect	A6	VssIO	A7	Do Not Connect	A8	VssIO
A9	SysAD[32]	A10	SysADC[1]	A11	Do Not Connect	A12	VssIO
A13	VccInt	A14	VccInt	A15	SysAD[63]	A16	VssIO
A17	SysAD[61]	A18	VssIO	A19	Do Not Connect	A20	TcLine[4]
A21	VssIO	A22	VssIO	A23	VccIO	B1	VssInt
B2	VccIO	B3	VssInt	B4	VssIO	B5	TcLine[10]
B6	SysAD[35}	B7	SysAD[34]	B8	VccInt	B9	SysAD[33]
B10	SysADC[5]	B11	SysADC[0]	B12	Do Not Connect	B13	SysADC[7]
B14	SysADC[6]	B15	Do Not Connect	B16	SysAD[30]	B17	SysAD[29]
B18	SysAD[28]	B19	TcLine[5]	B20	VssIO	B21	VssInt
B22	VccIO	B23	VssIO	C1	VssIO	C2	VssInt
C3	VccIO	C4	VccIO	C5	Do Not Connect	C6	TcLine[9]
C7	SysAD[3]	C8	SysAD[2]	C9	VccInt	C10	SysAD[0]
C11	SysADC[4]	C12	VccInt	C13	SysADC[3]	C14	SysADC[2]
C15	SysAD[62]	C16	VccInt	C17	SysAD[60]	C18	TcLine[6]
C19	Do Not Connect	C20	VccIO	C21	VccIO	C22	VssInt
C23	VssIO	D1	TcLine[13]	D2	VssIO	D3	VccIO
D4	VccIO	D5	VccIO	D6	VccIO	D7	TcLine[8]
D8	VccInt	D9	VccIO	D10	SysAD[1]	D11	VccInt
D12	VccIO	D13	VccInt	D14	SysAD[31]	D15	VccIO
D16	VccInt	D17	TcLine[7]	D18	VccIO	D19	VccIO
D20	VccIO	D21	VccIO	D22	VssIO	D23	Do Not Connect
E1	VccInt	E2	TcLine[14]	E3	TcLine[12]	E4	VccIO
E20	VccIO	E21	Do Not Connect	E22	Do Not Connect	E23	TcLine[1]
F1	VssIO	F2	TcLine[16]	F3	TcLine[15]	F4	VccIO
F20	VccIO	F21	TcLine[3]	F22	TcLine[0]	F23	VssIO
G1	SysAD[36]	G2	SysAD[4]	G3	TcLine[17]	G4	VccInt
G20	TcLine[2]	G21	VccInt	G22	SysAD[59]	G23	SysAD[58]
H1	VssIO	H2	SysAD[37]	H3	SysAD[5]	H4	Do Not Connect
H20	VccInt	H21	SysAD[27]	H22	SysAD[26]	H23	VssIO
J1	SysAD[7]	J2	SysAD[6]	J3	VccInt	J4	VccIO
J20	VccIO	J21	Vccint	J22	SysAD[57]	J23	SysAD[56]
K1	SysAD[40]	K2	SysAD[8]	K3	SysAD[39]	K4	SysAD[38]
K20	SysAD[25]	K21	SysAD[24]	K22	SysAD[55]	K23	SysAD[23]
L1	SysAD[10]	L2	SysAD[41]	L3	SysAD[9]	L4	VccInt
L20	VccInt	L21	SysAD[54]	L22	SysAD[22]	L23	SysAD[53]
M1	VssIO	M2	SysAD[11]	M3	SysAD[42]	M4	VccIO
M20	VccIO	M21	SysAD[52]	M22	SysAD[21]	M23	VssIO

Pin	Function	Pin	Function	Pin	Function	Pin	Function
N1	SysAD[43]	N2	VccInt	N3	SysAD[12]	N4	SysAD[44]
N20	SysAD[19]	N21	SysAD[51]	N22	VccInt	N23	SysAD[20]
P1	SysAD[13]	P2	SysAD[45]	P3	SysAD[14]	P4	VccInt
P20	VccInt	P21	SysAD[49]	P22	SysAD[18]	P23	SysAD[50]
R1	SysAD[46]	R2	SysAD[15]	R3	SysAD[47]	R4	VccIO
R20	VccIO	R21	SysAD[16]	R22	SysAD[48]	R23	SysAD[17]
T1	VssIO	T2	RspSwap*	T3	PRqst*	T4	VccInt
T20	ExtRqst*	T21	VccOK	T22	BigEndlan	T23	VssIO
U1	PAck*	U2	VccInt	U3	ModeClock	U4	JTCK
U20	VccInt	U21	NMI*	U22	Reset*	U23	ColdReset*
V1	VssIO	V2	JTDO	V3	JTMS	V4	VccIO
V20	VccIO	V21	INT[9]*	V22	VccInt	V23	VssIO
W1	JTDI	W2	VccIO	W3	Do Not Connect	W4	VccIO
W20	VccIO	W21	INT[6]*	W22	INT[8]*	W23	VccInt
Y1	Do Not Connect	Y2	VssIO	Y3	VccIO	Y4	VccIO
Y5	VccIO	Y6	VccIO	Y7	RdRdy*	Y8	Release*
Y9	VccIO	Y10	TcWord[0]	Y11	VccInt	Y12	VccIO
Y13	SysCmd[5]	Y14	VccInt	Y15	VccIO	Y16	VccInt
Y17	INT[2]*	Y18	VccIO	Y19	VccIO	Y20	VccIO
Y21	VccIO	Y22	VssIO	Y23	INT[7]*	AA1	VssIO
AA2	VssInt	AA3	VccIO	AA4	VccIO	AA5	Do Not Connect
AA6	TcMatch	AA7	ValidOut*	AA8	SysClock	AA9	VccInt
AA10	Do Not Connect	AA11	Do Not Connect	AA12	SysCmd[0]	AA13	SysCmd[4]
AA14	SysCmd[8]	AA15	TcTCE*	AA16	TcValid	AA17	VccInt
AA18	INT[3]*	AA19	Do Not Connect	AA20	VccIO	AA21	VccIO
AA22	VssInt	AA23	VssIO	AB1	VssIO	AB2	VccIO
AB3	VssInt	AB4	VssIO	AB5	ModeIn	AB6	ValidIn*
AB7	VccP	AB8	VccInt	AB9	VccInt	AB10	TcCWE[0]*
AB11	TcDCE[0]*	AB12	SysCmd[1]	AB13	SysCmd[3]	AB14	SysCmd[7]
AB15	TcClr*	AB16	TcTDE*	AB17	TcDOE*	AB18	INT[0]*
AB19	INT[4]*	AB20	VssIO	AB21	VssInt	AB22	VccIO
AB23	VssInt	AC1	VccIO	AC2	VssInt	AC3	VssIO
AC4	RdType	AC5	WrRdy*	AC6	VssIO	AC7	VssP
AC8	VssIO	AC9	TcWord[1]	AC10	TcCWE[1]*	AC11	TcDCE[1]*
AC12	VssIO	AC13	SysCmd[2]	AC14	SysCmd[6]	AC15	SysCmdP
AC16	VssIO	AC17	TcTOE*	AC18	VssIO	AC19	INT[1]*
AC20	INT[5]*	AC21	VssIO	AC22	VssIO	AC23	VccIO

14 Ordering Information

RM7000A -123 T I

Temperature Grade:
(blank) = commercial
I = Industrial

Package Type:
T = TBGA

Device Maximum Speed

Device Type
A = 0.18 micron process geometry

Valid Combinations

RM7000A-300T

RM7000A-350T

RM7000A-400T

RM7000A-350TI