MIPS32® I7200 Multiprocessor Core Family

Datasheet

Revision: 01.20
23/04/2018

Contents

List of Figures

List of Tables

1 Overview

The MIPS32® I7200 multiprocessing system (MPS) is a high performance multi-core cluster licensable IP solution. It is designed to deliver both high performance and low-latency responsiveness for system-on-chip (SoC) applications requiring rapid processing of real-time events.

Each core within the multi-core cluster is based on a 9-stage, dual-issue in-order pipeline with support for hardware multi-threading, designed to deliver high throughput and performance, and best-in-class efficiency per unit power and area. To complement the efficient pipeline design, the I7200 MPS utilizes the nanoMIPS instruction set architecture (ISA) to deliver this performance in smallest code size, providing for optimal use of the local memory resources to the CPU.

The cores of the I7200 MPS are coherently connected together via a Coherence Manager (CM, version 2.6) functional block, which includes a number of system level features and functional elements, including:

- Shared L2 cache
- Optional Global interrupt controller (GIC)
- Optional Cluster power controller (CPC)
- Accelerator/ IO coherency ports
- Global configuration registers (GCR)

The entire system offers many configurable options at the core and cluster level, and is available as fully synthesizable RTL for implementation in any semiconductor process technology.

1.1 CPU Core-Level Features

- 9-stage pipeline with dual-issue capability
- nanoMIPS32 Release 6 Instruction Set
  - 2B, 4B, 6B variable length instructions
  - Superior code density
- Optional nanoMIPS DSP ASE
  - 4 pairs of accumulator registers
  - Fractional data types, Saturating arithmetic
  - SIMD instructions operate on 1x32 or 2x16b or 4x8b simultaneously
- MIPS MT Application Specific Extension (ASE)
  - Support for 1 - 3 Virtual Processing Elements (VPEs)
  - Supports 1 - 9 thread contexts per core
  - Inter-Thread Communication (ITC) memory for efficient communication & data transfer
- Optional Programmable Memory Management Unit (MMU), TLB-based
  - 16/32/64 dual-entry JTLB per VPE
  - 8-entry DTLB
  - 4 - 12 entry ITLB
- Optional Memory Protection Unit (MPU)
  - Read/Write registers with reset values set at build time
  - Configurable number of overlay regions: 8 - 32 per core, in multiples of 4
  - Region size: 32B - 4GB
  - 4GB memory space divided into a series of 16 programmable 256 MB segments

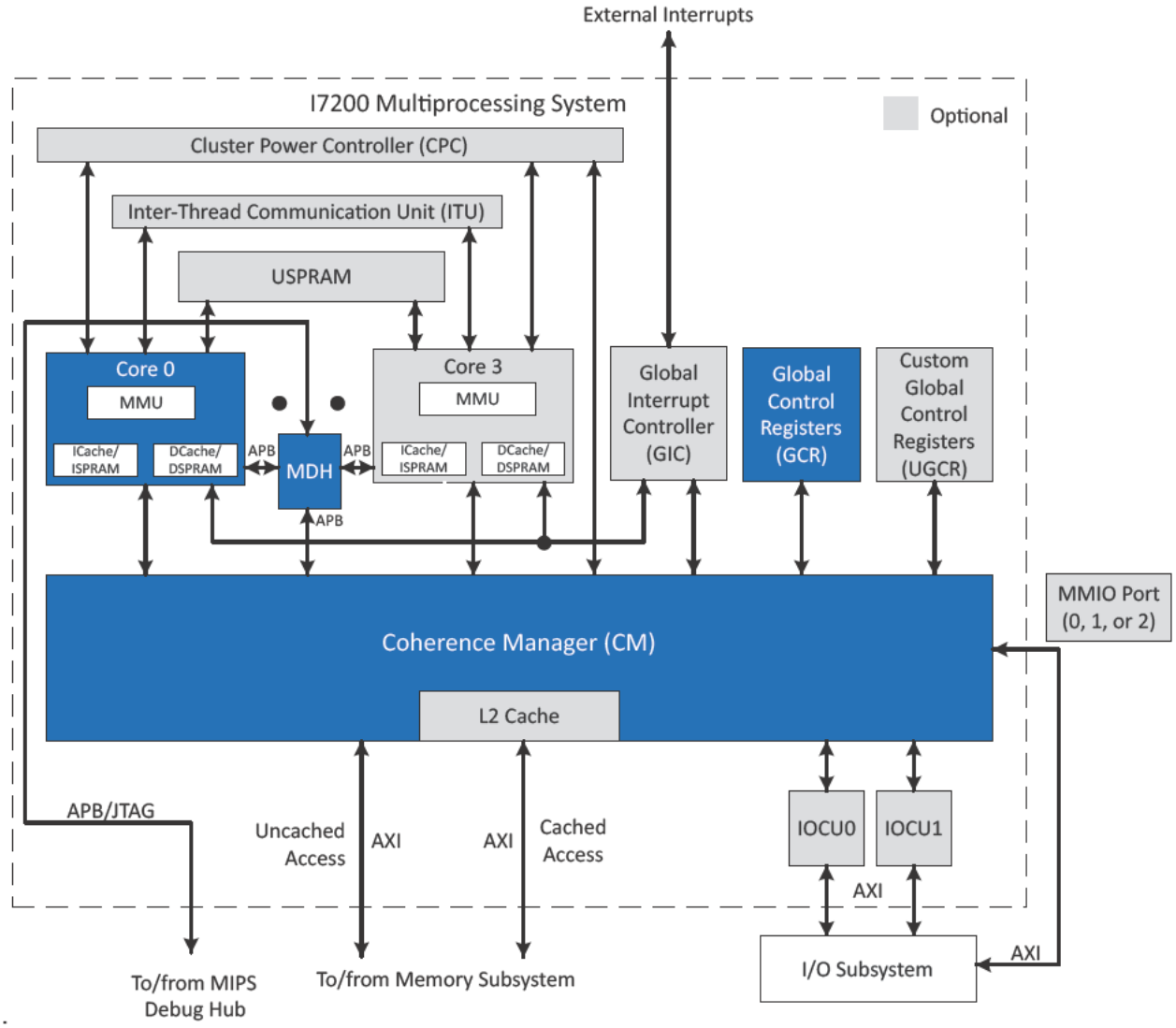
- Local Memory
  - L1 Cache sizes of 4/8/16/32/64/128 KB, 4-Way Set Associative
  - L1 MESI coherent cache states
  - 32-byte cache line size
  - 64-bit data and 32-bit address paths to caches
  - Virtually indexed, physically tagged
  - Supports up to 1MB of instruction and data Scratchpad RAM (optional)
  - Supports up to 1MB Unified SPRAM support (optional)
- Integrated integer Multiply/Divide Unit (MDU)
- Enhanced virtual addressing (EVA) mode allows for up to 3.0 GB of user or kernel virtual address space (optional)
- Privileged Resource Architecture

1.2 MPS Block Diagram

In the I7200 multiprocessing system, multi-CPU coherence is handled in hardware by the Coherence Manager (CM). The optional I/O Coherence Unit (IOCU) supports hardware I/O coherence by bridging a non-coherent I/O interconnect to the CM and handling ordering requirements. The Global Interrupt Controller (GIC) handles the distribution of interrupts between and among the CPUs. Under software

controlled power management, the Cluster Power Controller (CPC) can gate off the clocks, voltage supply, or both to idle cores. Figure 1 shows an I7200 MPS block diagram.

Figure 1: MPS Block Diagram



1.3 System-Level Features

- 1 - 4 coherent MIPS32® I7200 CPU cores
- Second generation system-wide Coherence Manager (CM) providing L2 cache, I/O coherence, power management, and interrupt routing across all CPU cores
- Integrated 8-way set associative L2 cache controller supporting 0 KB to 8 MB cache sizes with wait state control for 1:1 clock and optimal SRAM speed
- Hardware assisted L2 Cache initialization and flushing
- Supports cache-to-cache data transfers
- Speculative memory reads and Out-of-order data return to reduce latency

- Up to two IO Coherence Units (IOCUs) enabling IO devices to be coherent with the cluster's L1 and L2 caches
- Ability to add User Defined Global Configuration Registers
- Multiple SoC AXI-4 interfaces:
  - Cacheable Memory interface supporting 64-bit, 128-bit or 256-bit data paths, and up to 32 outstanding reads and 32 outstanding writes.
  - Uncached Memory interface supporting 128-bit data path
  - Up to 2 Memory-Mapped IO (MMIO) interfaces support 64-bit data paths
  - Up to 2 IO transaction interfaces to connect with the IOCUs supporting 128-bit data paths.
  - DMA ports for Instruction, Data, and Unified Scratchpad RAMs supporting 64-bit data paths.
- Instruction, Data, Unified scratchpad RAMs supporting AXI-LITE DMA ports.
- Power Control
  - Minimum frequency: 0 MHz
  - Software controlled power-down mode
  - Cluster-level dynamic clocking
  - Cluster Power Controller (CPC) controlling shut down of idle CPU cores
- Debug architecture supporting multi-CPU debug and performance analysis
- MIPS On-Chip Instrumentation 32-Bit Debug System (OCI32)
  - APB debug slave port
  - Driven by MIPS Debug Hub (MDH)
  - Secure debug

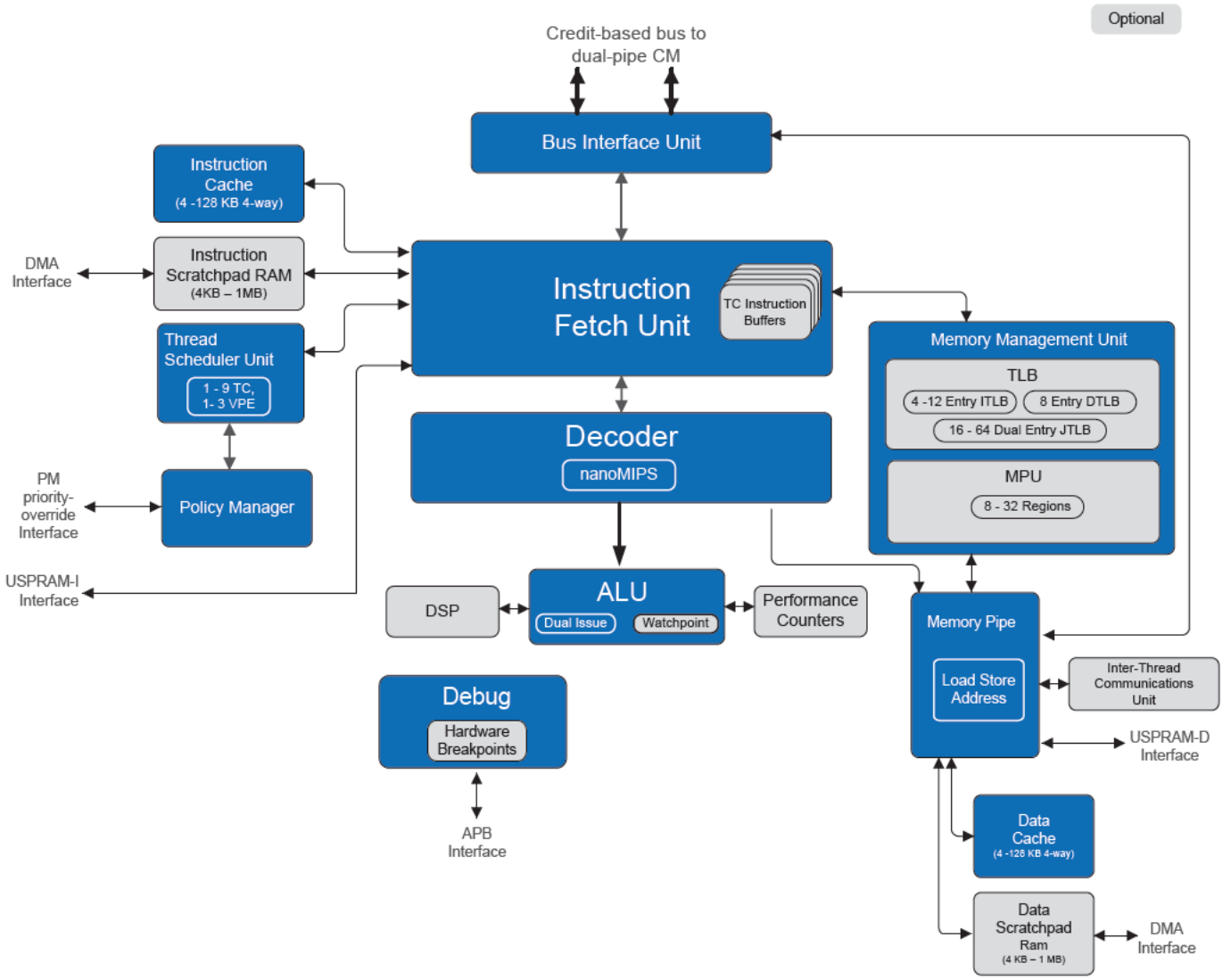
2 I7200 CPU Core

The following subsections describe the logic blocks in this diagram. For more information on the I7200 core in a multiprocessing environment, refer to the *Multiprocessing System* section.

2.1 CPU Core Block Diagram

Figure 2 shows a block diagram of a single I7200 core.

Figure 2: CPU Core Block Diagram



2.2 nanoMIPS Release 6 Instruction Set Architecture

nanoMIPS is a variable-length compressed instruction set that is completely standalone from the other MIPS ISAs. It is designed to compress the highest frequency instructions to 16-bit and use 48-bit instructions to efficiently encode 32-bit constants into the instruction stream. There are also a wider range of 32-bit instructions, adding carefully chosen high frequency instruction sequences into single operations; These include more flexible addressing modes, such as indexed and scaled indexed addressing, and branch compare with immediate and macro style instructions. The macro-like instructions compress prologue and epilogue sequences as well as a small number of high frequency instruction pairs, such as two move instructions or a move and function call. nanoMIPS also eliminates branch delay slots, following a precedent set by microMIPSR6.

To get the most from the new ISA, the ABI was redesigned to create a new symbiotic relationship between the two that pushes code density and performance further still. The ABI creates a fully link time relaxable model, which means that every last byte out of the code image is used even when deferring final addressing mode and layout decisions to link time. Mindful of the MIPS heritage, the redesign ensures that while open to any possible change, there is also minimal impact when porting code from MIPS to nanoMIPS and ample support to achieve source compatibility between the two.

The net effect of these changes leads to an average code size reduction of 20% smaller than microMIPSR6. Comparing the ISA in terms of number of instructions to issue versus microMIPS, a reduction of between 8% and 11% of dynamic instruction count is achieved using some of the more complex operations added to the ISA.

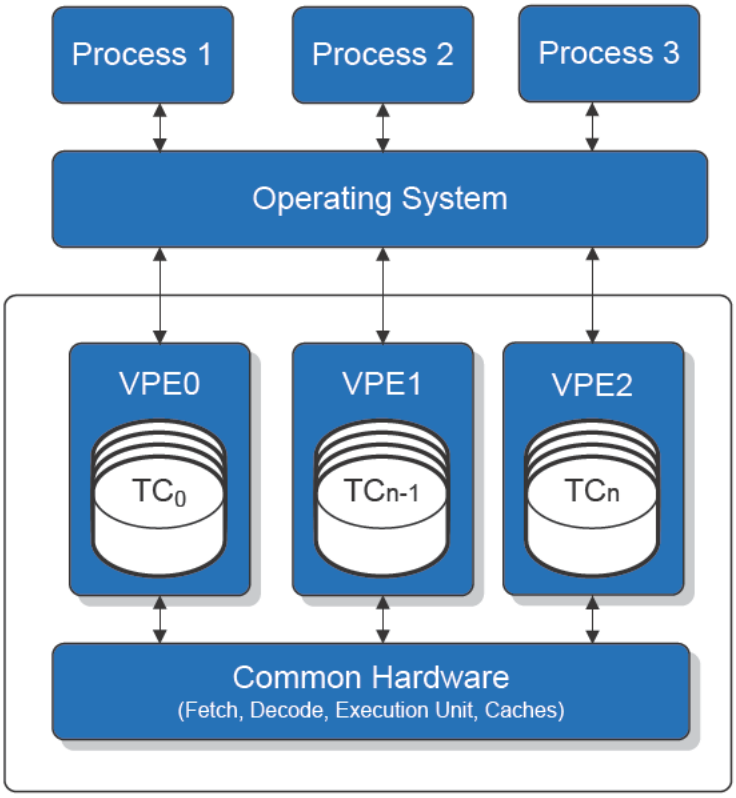
2.3 MIPS Multi-Thread Technology

The I7200 core implements multi-threaded (MT) architecture and supports the Application Specific Extensions (MT ASE).

Building on previous generation of MIPS multi-threaded (MT) processors, the I7200 core implements the same multi-threaded architecture and supports the MT ASE, which are based on a two-layered framework involving Virtual Processing Elements (VPEs) and Thread Contexts (TCs). Each I7200 core can support up to three VPEs, which share a dual-issue pipeline among other hardware resources. Because each VPE includes a complete copy of the processor state as seen by the software system, each VPE appears as a complete standalone processor to a multi-processor operating system, such as SMP Linux. The I7200 core

can support up to nine TCs allocated across three VPEs, optimized and partitioned at run time. Figure 3 shows the relationship of the OS, VPEs, TCs, and the common hardware in the I7200 core.

Figure 3: Single I7200 Core with Multiple VPEs



2.4 Operating Modes

The I7200 core supports four modes of operation:

- User mode: Most often used for application programs.
- Supervisor mode: Provides an intermediate privilege level with access to the ksseg (kernel supervisor segment) address space.
- Kernel mode: Used for handling exceptions and operating system kernel functions, including CP0 management and I/O device accesses.
- Debug mode: Used during system bring-up and software development. Refer to section "EJTAG Debug Support" for more information about debug mode.

2.5 CPU Module Blocks

The following sections describe the various CPU module blocks.

2.5.1 Instruction Fetch Unit

This block is responsible for fetching instructions for all Thread Contexts (TCs). Each TC has an instruction buffer (IBF) that decouples the fetch unit from the execution unit. When executing instructions from multiple TCs, a portion of the IBF is used as a skid buffer. Instructions are held in the IBF after being sent to the execution unit. This allows stalled instructions to be flushed from the execution pipeline without needing to be fetched again.

In order to fetch instructions without intervention from the execution unit, the fetch unit contains branch prediction logic. A voting selection between three Branch History Tables (BHT) and a Global History Register are used to predict the direction of branch instructions. A Return Prediction Stack (RPS) holds the return address from the most recent subroutine calls. The link address is pushed onto the stack whenever a BALC, BALRSC, JALRC, JALRC.hb, MOVE.BALC instruction is seen. The address is popped when a JRC or RESTORE.JRC instruction is executed.

2.5.2 Thread Schedule Unit (TSU)

This unit is responsible for dispatching instructions from different Thread Contexts (TCs), an internal policy manager assigns priorities for each TC. The TSU determines which TCs are available and selects the highest priority one available.

2.5.3 Policy Manager

The policy manager supports a software interface that controls the prioritization and relative allocation of hardware resources between each TC in the core. The I7200 core also offers external signals to control the TC priority, based on the requirements of a real-time system. Software can choose between two modes of operation: weighted-round-robin which maximizes general performance and throughput, or quality-of-service, which maximizes performance of a specific TC.

2.5.4 Execution Unit

The I7200 CPU execution unit implements a load/store architecture with single-cycle ALU operations (logical, shift, add, subtract) and an autonomous multiply/divide unit. Each TC on a I7200 CPU contains thirty-one 32-bit general-purpose registers used for integer operations and address calculations. Additional sets of shadow register files can be added to be dedicated for interrupt and exception processing. The register file is fully bypassed to minimize operation latency in the pipeline.

The execution unit includes:

- Dual Issue Pipeline to achieve very high performance with a balance of area and power:
  - One pipe dedicated to ALU, MDU, LSU, and CP0 instructions
  - One pipe dedicated to ALU and Control Transfer (CTI) instructions
  - Pipeline can issue up to two instructions and graduate two instructions per cycle
- 32-bit adder used for calculating the data address
- Logic for verifying branch prediction
- Load aligner
- Bypass multiplexers used to avoid stalls when executing instruciton streams where data producing instructions are followed closely by consumers of their results
- Leading Zero/One detect unit for implementing the CLZ and CLO instructions

- Arithmetic Logic Unit (ALU) for performing bit-wise logical operations
- Shifter and Store Aligner

2.5.5 Multiply/Divide Unit (MDU)

The multiply/divide unit (MDU) contains a separate pipeline for integer multiply and divide operations. This pipeline operates in parallel with one of the two ALUs in the integer unit pipeline.

The MDU consists of a pipelined 32 x 32 multiplier, result/accumulation registers (HI and LO) when a nanoMIPS DSP ASE is configured, a divide state machine, and the necessary multiplexers and control logic.

The MDU supports execution of one or accumulated operations every clock cycle. A divide operation can be executed as fast as one every six cycles.

2.5.6 Memory Management Options

The I7200 CPU uses either a TLB-based MMU or a simpler Memory Protection Unit (MPU).

When the TLB-based MMU is selected, each I7200 VPE contains a Memory Management Unit (MMU) that primarily converts virtual addresses to physical addresses and provides attribute information for different segments of memory. Each VPE contains a separate JTLB, so translations for each VPE are independent from the other VPEs.

The I7200 core alternatively can be configured with the optional direct mapped MPU. The MPU provides finer-grain protection of memory pages relative to an FMT MMU and also eliminates the address translation mechanism for space-sensitive applications. The MPU provides memory access control at both the segment and the region level.

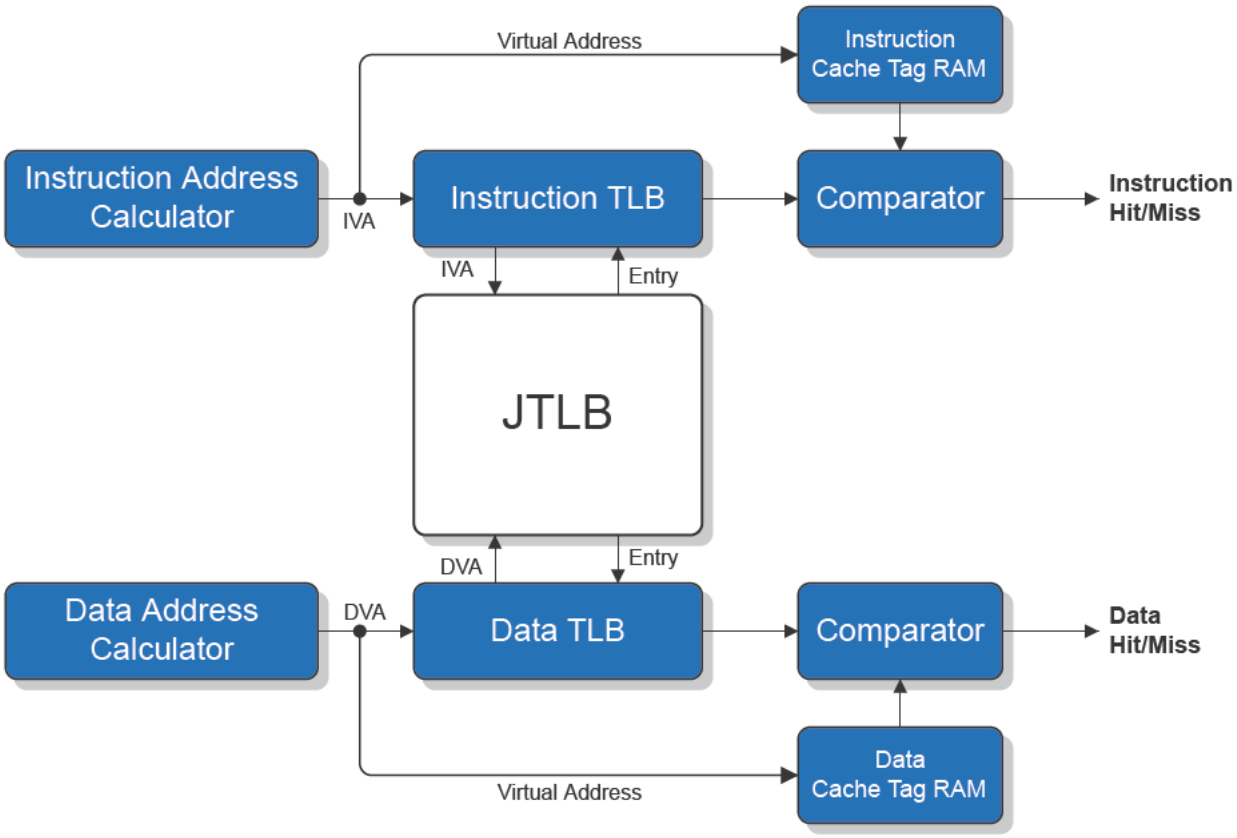
Translation Lookaside Buffer (TLB)

The basic TLB functionality is specified by the MIPS32 Privileged Resource Architecture. A TLB provides mapping and protection capability with per-page granularity. The I7200 implementation allows a wide range of page sizes to be simultaneously present.

The TLB contains a fully associative Joint TLB (JTLB). To enable higher clock speeds, two smaller micro-TLBs are also implemented: the Instruction Micro TLB (ITLB) and the Data Micro TLB (DTLB). When an instruction or data address is calculated, the virtual address is compared to the contents of the appropriate micro TLB (uTLB). If the address is not found in the uTLB, the JTLB is accessed. If the entry is found in the JTLB, that entry is then written into the uTLB. If the address is not found in the JTLB, a TLB exception is taken.

Figure 4 shows how the ITLB, DTLB, and JTLB are implemented in the I7200 CPU.

Figure 4: I7200 Core Address Translation



Joint TLB (JTLB)

The JTLB is a fully associative TLB cache containing 16, 32, or 64-dual-entries per VPE mapping up to 128 virtual pages to their corresponding physical addresses. The address translation is performed by comparing the upper bits of the virtual address (along with the ASID) against each of the entries in the *tag* portion of the joint TLB structure.

The JTLB is organized as pairs of even and odd entries containing pages that range in size from 4 KB to 256 MB, in factors of four, into the 4 GB physical address space. The JTLB is organized in page pairs to minimize the overall size. Each *tag* entry corresponds to two data entries: an even page entry and an odd page entry. The highest order virtual address bit not participating in the tag comparison is used to determine which of the data entries is used. Because page sizes can vary on a page-pair basis, the determination of which address bits participate in the comparison and which bit is used to make the even-odd determination is decided dynamically during the TLB look-up.

Instruction TLB (ITLB)

The ITLB is managed by hardware and is transparent to software. The larger JTLB is used as a backing structure for the ITLB. If a fetch address cannot be translated by the ITLB, the JTLB is used to translate it.

The ITLB contains between 4 and 12 entries and is dedicated to performing translations for the instruction stream. The ITLB is a hybrid structure having 3 entries that are shared by all TCs plus an additional entry dedicated to each TC. Therefore, a core with one VPE and one TC would have a 4-entry TLB with all

entries dedicated to one TC. Conversely, a core with 1 VPE and 9 TC's would have a three shared entries, plus one entry per TC, for a total of 12 entries.

The ITLB maps 4 KB or 1 MB pages/subpages. For 4 KB or 1 MB pages, the entire page is mapped in the ITLB. If the main TLB page size is between 4 KB and 1 MB, only the current 4 KB subpage is mapped. Similarly, for page sizes larger than 1 MB, the current 1 MB subpage is mapped.

Data TLB (DTLB)

The DTLB is managed by hardware and is transparent to software. The larger JTLB is used as a backing structure for the DTLB. If a load/store address cannot be translated by the DTLB, a lookup is done in the JTLB. The JTLB translation information is copied into the DTLB for future use.

The DTLB is an 8-entry, fully associative TLB dedicated to performing translations for loads and stores. All entries are shared by all TCs. Similar to the ITLB, the DTLB maps either 4 KB or 1 MB pages/subpages.

Memory Protection Unit (Optional)

The MPU uses an underlying segmented memory scheme. The 4-Gigabyte address space is divided into 16 equal sized 256-Megabyte direct mapped segments. Each segment can have different attributes. The attributes are of two types. The CCA type determines if the segment will use L1 caches or be uncached. The remaining attributes are used to protect the use of the segment.

There are three different protection attributes that can be set in any combination:

- Read inhibit (RI) If set, will prevent data loads, resulting in an MPUL exception if a load is attempted.
- Write inhibit (WI) If set, will prevent data stores, resulting in an MPUS exception if a store is attempted.
- Execute inhibit (XI) If set, will prevent instruction fetches, resulting in an MPUL exception if a fetch is attempted.

In addition to the MPU segments the I7200 can be configured for up to 32 overlay regions that can change part of a segment's attributes or change multiple segment attributes at a time.

Enhanced Virtual Address

The I7200 core supports a programmable memory segmentation scheme called Enhanced Virtual Address (EVA), which allows for more efficient use of the 32-bit address space. Traditional MIPS virtual memory support divides up the virtual address space into fixed segments, each with fixed attributes and access privileges. Such a scheme limits the amount of physical memory available to 0.5GB, the size of kernel segment 0 (kseg0).

With EVA, the size of virtual address space segments can be programmed, as can their attributes and privilege access. With this ability to overlap access modes, kseg0 can now be extended up to 3.0GB, leaving at least one 1.0GB segment for mapped kernel accesses. This extended kseg0 is called xkseg0. This space overlaps with useg, because segments in xkseg0 are programmed to support mapped user accesses and unmapped kernel accesses. Consequently, the user space is equal to the size of xkseg0, which can be up to 3.0GB.

System Control Coprocessor (CP0)

In MIPS architecture, CP0 is responsible for the virtual-to-physical address translation and cache protocols, the exception control system, the processor's diagnostic capability, the operating modes (kernel, user, supervisor, and debug), and whether interrupts are enabled or disabled. Configuration information, such as cache size and associativity, presence of features like nanoMIPS, MIPS32, or DSP, is also available by accessing the CP0 registers.

Coprocessor 0 also contains the logic for identifying and managing exceptions. Exceptions can be caused by a variety of sources, including boundary cases in data, external events, or program errors.

Most of CP0 is replicated per VPE. A small amount of state is replicated per TC and some is shared between the VPEs.

2.5.7 Level 1 Data Cache

The Level 1 (L1) data cache is an on-chip memory block of 4/8/16/32/64 KB, with 4-way associativity. The 128 KB sized data cache is optionally supported when the MPU is configured. A tag entry holds 22 bits of physical address and two cache state bits. The data entry holds 64 bits of data per way. There are 4 data entries for each tag entry. The tag and data entries exist for each way of the cache. The way-select array holds the dirty and LRU replacement algorithm bits for all 4 ways (6-bit LRU, 4-bit dirty).

Virtual aliasing can occur when using 4 KB pages in the TLB and 32 or 64 KB cache sizes. Because it is quite challenging for software to manage virtual aliases across multiple devices, these larger cache arrays are banked on the aliased 1 or 2 physical address bits to eliminate the virtual aliases.

The I7200 CPU supports data-cache locking. Cache locking allows critical code or data segments to be locked into the cache on a “per-line” basis, enabling the system programmer to maximize the efficiency of the system cache. The locked contents can be updated on a store hit, but are not selected for replacement on a cache miss. Locked lines do not participate in the coherence scheme so processes which lock lines into a particular cache should be locked to that processor and prevented from migrating.

The cache-locking function is always available on all data-cache entries. Entries can then be marked as locked or unlocked on a per entry basis using the CACHE instruction.

2.5.8 Level 1 Instruction Cache

The Level 1 (L1) instruction cache is an on-chip memory block of 4/8/16/32/64 KB, with 4-way associativity. The 128 KB sized instruction cache is optionally supported when the MPU is configured. A tag entry holds 22 bits of physical address, a valid bit, and a lock bit. The instruction data entry holds two instructions (64 bits). There are four data entries for each tag entry. The tag and data entries exist for each way of the cache. The LRU replacement bits (6-bit) are shared among the 4 ways and are stored in a separate array.

The instruction cache block also contains and manages the instruction line fill buffer. Besides accumulating data to be written to the cache, instruction fetches that reference data in the line fill buffer are serviced either by a bypass of that data, or data coming from the external interface. The instruction cache control logic controls the bypass function.

The I7200 CPU supports instruction-cache locking. Cache locking allows critical code or data segments to be locked into the cache on a “per-line” basis, enabling the system programmer to maximize the efficiency of the system cache.

The cache-locking function is always available on all instruction-cache entries. Entries can then be marked as locked or unlocked on a per entry basis using the CACHE instruction.

2.5.9 Level 1 Cache Memory Configuration

The I7200 CPU incorporates on-chip L1 instruction and data caches that are typically implemented from readily available single-port synchronous SRAMs and accessed in two cycles: one cycle for the actual SRAM read and another cycle for the tag comparison, hit determination, and way selection. The instruction and data caches each have their own 64-bit data paths and can be accessed simultaneously. The following table lists the I7200 CPU instruction and data cache attributes.

Table 1: CPU L1 Instruction and Data Cache Attributes

Parameter	Instruction	Data
Size (see Note)	4, 8, 16, 32 or 64 KB, or 128 KB when MPU is configured	4, 8, 16, 32 or 64 KB, or 128 KB when MPU is configured
Organization	4 way set associative	4 way set associative
Line Size	32 Bytes	32 Bytes
Read Unit	64 bits	64 bits
Write Policies	N/A	Coherent and non-coherent write-back and write allocate

Parameter	Instruction	Data
Miss restart after transfer of	Missed word	Missed word
Cache Locking	Per line	Per line

Note: For Linux based applications, MIPS recommends a 64 KB L1 cache size, with a minimum size of 32 KB.

2.5.10 Interrupt Handling

The I7200 core supports six hardware interrupts, two software interrupts, a timer interrupt, and a performance counter interrupt. These interrupts can be used in any of the following three interrupt modes:

- Interrupt compatibility mode: Acts identically to that of an implementation in Release 1 of the Architecture.
- Vectored Interrupt (VI) mode: Adds the ability to prioritize and vector interrupts to a handler dedicated to that interrupt.
- External Interrupt Controller (EIC) mode: Provides support for an external interrupt controller that handles prioritization and vectoring of interrupts. See the *Global Interrupt Controller* section for more information about interrupt handling functionality.

2.5.11 InterThread Communication Unit (ITU)

This block provides a mechanism for efficient communication between TCs. This block has a number of locations that can be used as mailboxes, FIFO mailboxes, mutexes, and semaphores.

2.5.12 Instruction, Data, and Unified Scratchpad RAM

The I7200 core allows blocks of scratchpad RAM to be attached to the load/store and/or instruction units. These allow low-latency access to a fixed block of memory, and unified scratch pad RAM (USPRAM). The size of both the instruction scratch pad RAM (ISPRAM), data scratch pad RAM (DSPRAM) can be configured from a range of 4 KB to 1 MB. These RAM's are used for the temporary storage of information and can be modified by the user at any time. At the cost of a higher access latency, the unified scratch pad RAM offers more flexibility by being accessible from both instruction and data ports of all cores.

3 Multiprocessing System

The I7200 Multiprocessing System (MPS) consists of the logic modules listed in the *Overview* section. Each of these blocks is described in the following sections.

3.1 Cluster Power Controller (CPC)

The Cluster Power Controller (CPC) has three versions:

- CPC Standard
- CPC Basic
- CPC External

CPC Standard — Individual CPUs within the cluster can have their clock and/or power gated off when they are not in use. This gating is managed by the CPC. The CPC handles the power shutdown and ramp-up of all CPUs in the cluster. Any I7200 CPU that supports power-gating features is managed by the CPC.

CPC Standard also organizes power-cycling of the CM, dependent on the individual core status and shut-down policy. Reset and root-level clock gating of individual CPUs are considered part of this sequencing.

The CPC Standard allows power-up to be controlled through hardware signals and software commands. After power-up the power state transition is controlled through software commands.

CPC Basic — The primary operating mode of CPC Basic allows powering up/down through hardware signals. However, the CPC can be placed in a special debug mode, which allows the power states to be controlled through software (similar to CPC Standard).The CPC Basic has less functionality, but instantiates all signals and each design only uses the signals that it requires.

The CPC Standard operates at the same clock rate as the Cores and the CM, but the CPC Basic uses its own clock input, allowing it to run at any frequency.

CPC External — the CPC External option offers only a hardware interface that allows an external power controller to control the power-up/down sequences of the cluster. CPC related registers are removed and unavailable in this option.

3.1.1 Reset Control

The reset input of the system resets the Cluster Power Controller (CPC). Reset sideband signals are required to qualify a reset as system cold or warm start. Register settings determine the course of action:

- Remain in powered-down
- Go into clock-off mode
- Power-up and start execution

This prevents random power up of power domains before the CPC is properly initialized. In case of a system cold start, after reset is released, the CPC powers up the I7200 CPUs as directed in the CPC cold start configuration. If at least one CPU has been chosen to be powered up on system cold start, the CM is also powered up.

When supply rail conditions of power gated CPUs have reached a nominal level, the CPC will enable clocks and schedule reset sequences for those CPUs and the coherence manager.

At a warm start reset, the CPC brings all power domains into their cold start configuration. However, to ensure power integrity for all domains, the CPC ensures that domain isolation is raised before power is gated off. Domains that were previously powered and are configured to power up at cold start remain powered and go through a reset sequence.

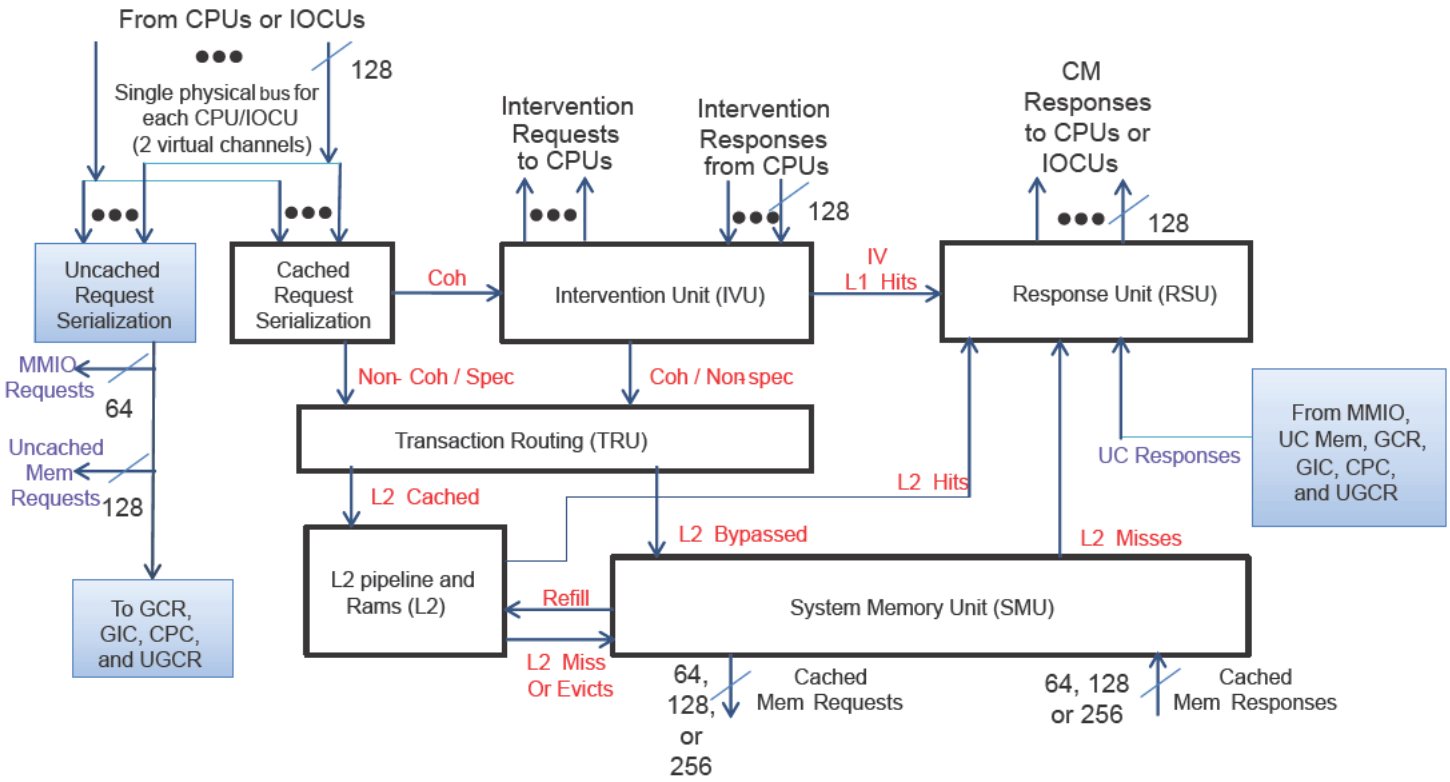
Within a warm start reset, sideband signals are also used to qualify if coherence manager status registers and GIC watch dog timers are to be reset or remain unchanged.

In addition to controlling the deassertion of the CPC reset signal, there are memory-mapped registers that can set the reset exception address for each CPU. This allows different boot vectors to be specified for each of the cores so they can execute unique code if required. Each of the cores will have a unique CPU number, so it is also possible to use the same boot vector and branch based on that.

3.2 Coherence Manager (CM)

The Coherence Manager (version 2.6) with integrated L2 cache is responsible for establishing the global ordering of requests, collecting the intervention responses, and for sending the correct data back to the requester. A high-level view of the request/response flow through the CM is shown in Figure 5 "Coherence Manager with Integrated L2 Cache Block Diagram." As shown, the CM provides independent pipelines for cached and uncached transactions sent by the CPU cores or IOCUs. Each of the figure blocks is described in more detail in the following subsections.

Figure 5: Coherence Manager with Integrated L2 Cache Block Diagram



3.2.1 Request Unit (RQU)

The Request Unit (RQU) receives transactions from multiple CPU cores and/or the I/O ports and routes to either the cached or uncached pipeline. The cached pipeline serializes the transactions and routes them to the Intervention Unit (IVU) and/or the Transaction Routing Unit (TRU). The uncached pipeline independently serializes uncached requests and routes them to an uncached memory port, an MMIO port, or to configuration registers. The routing is based on the transaction type, the transaction address, and the CM's programmable address map.

Each incoming port of the RQU can be configured independently to support 6 - 16 outstanding cacheable reads.

Port Arbitration Priority

The CPU core and IOCU port requests to the CM contain a sideband bit to indicate the high vs. low priority of a transaction. For CPU-initiated requests, the priority sideband bit is determined by the priority of the TC that initiated the request. For IOCU requests, the sideband priority bit is set by the priority value driven with the corresponding request. The cached pipeline of the CM uses this high vs. low priority information when serializing the requests. High priority requests are generally serialized ahead of low-priority requests. However, a timeout value can be programmed to ensure that low-priority requests are provided some minimum bandwidth. The arbiter can also be programmed to limit the total number of low priority cacheable reads in-flight within the CM, thus reserving resources for high priority cacheable requests.

3.2.2 Intervention Unit (IVU)

The Intervention Unit (IVU) interrogates the L1 caches by placing requests on the intervention ports to CPU cores. Each processor responds with the state of the corresponding cache line. For most transactions, if a CPU core has the line in the MODIFIED or EXCLUSIVE state, it provides the data with its response. If the original request was a read, the IVU routes the data to the original requestor through the Response Unit (RSU). For the MESI protocol, intervention data may also be routed to the L2/Memory through the TRU (implicit writeback).

The IVU gathers the responses from each of the agents and manages the following actions:

- Speculative reads are resolved (confirmed or cancelled)
- Memory reads that are required because they were not speculative are issued to the Transaction Routing Unit (TRU).
- Modified data returned from the CPU is sent to the TRU to be written back to memory
- Data returned from the CPU is forwarded to the Response Unit (RSU) to be sent to the requester
- The MESI state in which the line is installed by the requesting CPU is determined (the “install state”). If there are no other CPUs with the data, a Shared request is upgraded to Exclusive.

Each device updates its cache state for the intervention and responds when the state transition has completed. The previous state of the line is indicated in the response. If a read type intervention hits on a line that the CPU has in a Modified or Exclusive state, the CPU returns the cache line with its response. A cacheless device, such as the IOCU, does not require an intervention port.

3.2.3 System Memory Unit (SMU)

The System Memory Unit (SMU) provides the interface to the cacheable memory port. For an L2 refill, the SMU reads the data from an internal buffer and issues the refill request to the L2 pipeline.

Note that the external interface may operate at a lower frequency than the Coherence Manager (CM), and the external block may not be able to accept as many requests as multiple CPUs can generate, so some buffering of the requests may be required.

3.2.4 Response Unit (RSU)

The RSU takes responses from the SMU, L2, IVU, uncached memory, MMIO, or register read response and places them on the appropriate interface for CPUs or IOCUs. Data from the L2 or SMU is buffered inside a buffer (RRB) associated with each RSU port, which is an enhancement over the previous generation Coherence Manager.

When a coherent read receives an intervention hit in the MODIFIED or EXCLUSIVE state, the Intervention Unit (IVU) provides the data to the RSU. The RSU then returns the data to the requesting core.

3.2.5 Transaction Routing Unit

The Transaction Routing Unit (TRU) arbitrates between requests from the RQU, IVU, and L2 CacheOp State Machine. The TRU routes requests to either the L2 or the SMU. The TRU also contains the request and intervention data buffers, which are written directly from the RQU and IVU, respectively. The TRU reads the appropriate write buffer when it processes the corresponding write request.

3.2.6 Level 2 Cache

The unified Level 2 (L2) cache holds both instruction and data references and contains a 7-stage pipeline to achieve high frequencies with low power while using commercially available SRAM generators.

Cache read misses are non-blocking; that is, the L2 can continue to process cache accesses while up to 32 misses are outstanding. The cache is physically indexed and physical tagged.

L2 Cache Configuration

The L2 cache in the CM can be configured as follows:

- 0, 32K, 64K, 128K, 256K, 512K, 1024K, 2048K, 4096K, or 8192K Bytes. The 32K and 64K L2 caches are only supported for configurations with 64B cache lines.
- 32 or 64-Byte line size
- 8 ways

L2 Pipeline Tasks

The L2 pipeline manges the flow of data to and from the L2 cache. The L2 pipeline performs the following tasks:

- Accesses the tags and data RAMs located in the memory block (MEM)
- Returns data to the RSU for cache hits
- Issues L2 miss requests
- Issues L2 write and eviction requests
- Returns L2 write data to the SMU. The SMU issues refill requests to the L2 for installation of data for L2 allocations

L2 Cache Features

- Supports write-back operation
- Pseudo-LRU replacement algorithm
- Programmable wait state generator to accommodate a wide variety of SRAMs
- Operates at same clock frequency as CPU
- Cache line locking support
- Bypass mode
- Fully static design: minimum frequency is 0 MHz
- Sleep mode
- Hardware can be programmed to initialize/flush all or part of the L2 cache.

Hardware Assisted L2 Cache Initialization and Flushing

The L2 CacheOp state machine is a hardware component that can be used to quickly initialize or flush the L2 cache.

- Software initiates the initalization or flushing of either the the entire L2 or a range of addresses by writing to the L2 State Machine CacheOp GCR.
- The L2 CacheOp state machine executes the corresponding series of L2 Cache operations.
- The software can detemine that the operations are complete by reading the L2 State Machine CacheOp GCR.

3.2.7 Coherence Manager Performance

The CM has a number of high performance features:

- Separate pipelines through the CM for cached and uncached transactions. This improves the performance of cacheable transactions, which could hit in L2, as they are not blocked by backed-up uncached transactions through the CM when the external system is slow to respond to uncached memory transactions.
- 256-bit wide internal data paths throughout the cached pipeline of the CM
- 64-, 128-, or 256-bit wide system AXI4.0 cacheable transaction interface
- 128-bit wide AXI4.0 uncacheable transaction interface
- Cache to Cache transfers: If a read request hits in another L1 cache in the EXCLUSIVE or MODIFIED state, it will return the data to the CM and forwarded to the requesting CPU, thus reducing latency on the miss.
- Speculative Reads: Coherent read requests are forwarded to the L2 cache before they are looked up in the other caches. This is speculating that the cache line will not be found in another CPU's L1 cache.

3.3 I/O Coherence Unit (Optional)

Optional support for hardware I/O coherence is provided by the I/O Coherence Unit (IOCU), which maintains I/O coherence of the caches in all coherent CPUs in the cluster. Up to 2 IOCUs are supported in the I7200 MPS.

The IOCU acts as an interface block that manages coherent accesses initiated by IO devices to the Coherence Manager (CM). Coherent reads and writes of I/O devices generate interventions in other coherent CPUs that query the L1 cache. I/O reads access the latest data in caches or in memory, and I/O writes invalidate stale cache data and merge newer write data with existing data as required.

The IOCU also provides an AXI slave interface to the I/O interconnect for I/O devices to read and write system memory.

The IOCU design provides several features for easier integration:

- An AXI slave interface provides the capability to define cache attributes for each request—coherent or not, cacheable (in L2) or not, and L2 allocation policy
- Standard mapping units allow the cache attributes to be determined by signals driven with the request.
- Cacheable or Coherent Reads or Writes of any size may be allocated in the L2 cache.
- Bus width of 128 bits, supporting incremental bursts up to 256 beats on the I/O side. These requests are split into cache-line-sized requests on the CM side.
- Ensures proper ordering of responses for the split requests and tagged requests

.

3.3.1 Software I/O Coherence

For cases where a system redesign to accommodate hardware I/O coherence is not feasible, the CPUs and Coherence Manager provide support for an efficient software-managed I/O coherence. This support is through the globalization of hit-type CACHE instructions.

When a coherent address is used for the CACHE operations, the CPU makes a corresponding coherent request. The CM sends interventions for the request to all of the CPUs, allowing all of the L1 caches to be maintained together. The basic software coherence routines developed for single CPU systems can be reused with minimal modifications.

3.4 Memory Mapped IO (MMIO) Bus (Optional)

An MMIO request is initiated by a I7200 CPU access that targets an I/O device. In I7200, the MMIO address space is expected to be accessed by only using an Uncached or Uncached Accerlerated (UCA) Cache Coherency Attribute (CCA). Thus, the I7200 CPU requests targeting MMIO will be routed through the uncached pipeline of the CM and the CM sends those requests out on the external MMIO bus. The corresponding responses from the MMIO bus are captured by the CM, and the CM then routes those responses back to the requesting CPU core.

Features of the MMIO port are summarized below:

- AXI-4 compliant
- Configurable to be 0, 1 ,or 2 ports. This configuration is independent from the IOCUs.
- 64-bit data busses
- 32-bit address bus
- Interface is synchronous to I7200 cluster clock, but supports slower clock rates.
- Does not support cacheable reads on this port.
- Uncached accesses can only use up to 32-bit data and they will only be bursts of length 1.
- UCA writes can use up to 64-bit data in single data beat and can have bursts of multiple data beats.

3.5 Global Interrupt Controller

The optional Global Interrupt Controller (GIC) handles the distribution of interrupts between and among the CPUs in the cluster. This block has the following features:

- Software interface can be relocated throughout the memory-mapped address range
- Configurable number of system interrupts - from 8 to 256 in multiples of 8
- Support for different interrupt types:
  - Level-sensitive: active high or low
  - Edge-sensitive: positive, negative, or double-edge-sensitive
- Ability to mask and control routing of interrupts to a particular CPU
- Support for NMI routing
- Standardized mechanism for sending inter-processor interrupts

3.5.1 No-Global Interrupt Controller

At configuration time, you can also select the option to not instantiate the GIC within the cluster. When this option is selected, the interrupt-related signals for each CPU core appear as pins on the mips_soc module so you can use these pins to connect your own external Interrupt Controller.

3.6 Global Configuration Registers (GCR)

The Global Configuration Registers (GCR) are a set of memory-mapped registers that are used to configure and control various aspects of the Coherence Manager and the coherence scheme. Some of the GCR control options include:

- Address map: The base address for the GCR and GIC address ranges can be specified. An additional four address ranges can be defined as well. These control whether uncached requests go to memory or to memory-mapped I/O. A default can also be selected for addresses that do not fall within any range.
- Error reporting and control: Logs information about errors detected by the CM and controls how errors are handled (ignored, interrupt, and so on).
- Control Options: Various features of the CM can be disabled or configured. Examples of these are disabling speculative reads and preventing Read/Shared requests from being upgraded to Exclusive.

3.6.1 Custom GCRs

The CM provides the optional ability to implement a 64 KB block of custom registers that can be used to control system level functions. These registers are user defined and then instantiated into the design through a custom GCR block with a 64-bit wide interface.

The custom GCR registers are mapped into physical address space through the Global Custom Base register.

3.7 Debug Support

The I7200 core provides a Debug interface through the MIPS Debug Hub (MDH) and Advanced Peripheral Bus (APB) interface. The MDH provides the capability for connection to a JTAG or APB compatible debug system for improved debug performance and support for multi-core systems. For more information, refer to the MIPS Debug Hub Technical Reference Manual.

The APB accesses a number of registers used for determining core and debug status, changing debug modes, causing instructions to be executed, and providing other instrumentation functions.

3.7.1 Hardware Breakpoints

There are several types of simple hardware breakpoints defined in the EJTAG specification. These breakpoints stop the normal operation of the CPU and force the system into debug mode. There are two types of simple hardware breakpoints implemented in the I7200 CPU: Instruction breakpoints and Data breakpoints.

During build-time configuration, the I7200 CPU can be configured to support the following breakpoint options per VPE:

- Zero or four instruction breakpoints
- Zero or two data breakpoints

Instruction breaks occur on instruction fetch operations, and the break is set on the virtual address. Instruction breaks can also be made on the ASID value used by the MMU. A mask can be applied to the virtual address to set breakpoints on a range of instructions.

Data breakpoints occur on load and/or store transactions. Breakpoints are set on virtual address and ASID values, similar to the Instruction breakpoint. Data breakpoints can also be set based on the value of the load/store operation. Finally, masks can be applied to both the virtual address and the load/store value.

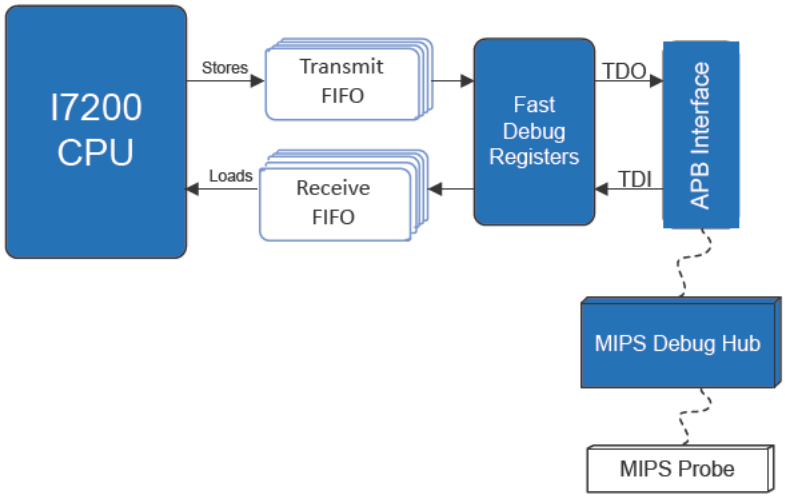
3.7.2 Inter-CPU Debug Breaks

The CPS includes registers that enable cooperative debugging across all CPUs. Each core features an output that indicates it has entered debug mode (possibly through a debug breakpoint). Registers are defined that allow CPUs to be placed into debug groups such that whenever one CPU within the group enters debug mode, a debug interrupt is sent to all CPUs within the group, causing them to also enter debug mode and stop executing non-debug mode instructions.

3.7.3 Fast Debug Channel

The I7200 CPU includes the EJTAG Fast Debug Channel (FDC) as a mechanism for efficient bidirectional data transfer between the CPU and the debug probe. Data is transferred via the APB interface. A pair of memory- mapped FIFOs buffer the data, isolating software running on the CPU from the actual data transfer. Software can configure the FDC block to generate an interrupt based on the FIFO occupancy or can poll the status.

Figure 6: Fast Debug Channel



3.8 Clocking Options

The I7200 core has the following clock domains:

- Cluster Domain: This is the main clock domain; it includes all I7200 MPS cores and the CM (including CM, GIC, IOCU, and L2 cache).
- Main Memory Domain: The AXI port(s) connecting to the SoC and the rest of the memory subsystem can operate at the following ratios of the cluster domain: 1:1, 1:2, 1:3, 1:4, 1:5, 1:6, 1:7 and 1:8. Because I7200 has two separate AXI ports for cached and uncached memory transactions, the clock ratios for each of these ports can be set independently. These AXI clocks are synchronous to the cluster clock.
- IO Domain(s): These are the AXI port(s) connecting to and from the I/O subsystem. These clocks can operate at the following ratios of the cluster domain: 1:1, 1:2, 1:3, 1:4, 1:5, 1:6, 1:7 and 1:8. Clock ratios for each of the I/O and MMIO AXI ports can be set up independently. These AXI clocks are synchronous to the cluster clock.
- MDH Domain: This is a low-speed clock domain for the EJTAG MDH controller. This clock domain is asynchronous to the cluster clock domain.
- CPC Domain: This is a low-speed clock domain for the Cluster Power Controller (CPC). This clock domain is asynchronous to the cluster clock domain in the CPC Basic configuration.

3.9 Design for Test (DFT) Features

The I7200 core provides the following test for determining the integrity of the core.

- Internal Scan: The I7200 core supports a full mux-based scan for maximum test coverage, with a configurable number of scan chains. ATPG test coverage can exceed 99%, depending on standard cell libraries and configuration options.

Memory BIST can be inserted with a CAD tool or other user-specified method. Wrapper modules and signal buses of configurable width are provided within the core to facilitate this approach.

4 Build-Time Configuration Options

The I7200 Coherent Processing System allows some features to be customized based on the intended application.

For a core that has already been built, software can determine the value of many of these options by querying an appropriate register field. Refer to the *MIPS32® I7200 Processor Family Software User's Manual* for a more complete description of these fields. The value of some options that do not have a functional effect on the core are not visible to software.

4.1 Configuration Options

The key configuration options that can be selected when the core is synthesized and implemented are listed below.

Table 2: I7200 Build-Time Configuration Options

Option	Choices	Software Visibility
System Options		
Number of CPUs	I7200 Dual: 2-core (configured as 1 or 2) I7200 Quad: 4-core (configured as 1, 2, 3, or 4)	GCR_CONFIG.PCORES
CPU Options		
Number of VPEs per CPU core	1 to 3	MVPConf0.PVPE
Number of threads per VPE	1 - 9	MVPConf0.PTC
TLB size (per VPE)	16, 32, or 64 dual entries	Config1MMU.Size
MMU type	TLB / MPU / FMT	Config4.MT
Number of MPU regions	8 - 32, by 4	MPU_Config.NumEntries
Number of outstanding instruciton fetches	2, 4, or 8	N/A
Number of outstanding data cache misses	4 or 8	N/A
Number of outstanding loads	4 or 9	N/A
Number of fill-store buffers	4 or 8	N/A
Number of RPS	4, 8, or 12	N/A
Power gating	Enabled or not	N/A
Clock gating	Enabled or not	N/A
PrID company option	0x0 - 0x7F	PrID.CompanyOption
I-cache size	4, 8, 16, 32, 64, 128 KB (128 KB available only if MPU is configured)	Config1.IL, Config1.IS, Config1.IA
Instruction ScratchPad RAM interface	Present or not	Config.ISP
Instruction ScratchPad RAM size	4 - 1024 KB in powers of 2	N/A
D-cache size	4, 8, 16, 32, 64, 128 KB (128 KB available only if MPU is configured)	Config1.DL, Config1.DS, Config1.DA

Option	Choices	Software Visibility
Data Scratch Pad RAM interface	Present or not	Config.DSP
Data Scratch Pad RAM size	4 - 1024 KB in powers of 2	N/A
Unified Scratch Pad RAM interface	Present or not	Config7.USP
Unified Scratch Pad RAM size	4 - 1024 KB in powers of 2	N/A
Number of watchpoint registers (per VPE)	• None • 1 instr, 1 data • 2 instr, 2 data • 3 instr, 3 data • 4 instr, 4 data	Config7.WR
Number of breakpoints (per VPE)	• None • 4 instr, 2 data	DCR.IB, DCR.DB, DBS.BCN, IBS.BCN
Performance counters	• None • 2 counter pairs • 4 counter pairs	Config1.PC
Yield Manager	Standard or custom	N/A
Coherene Manager Options		
Number of Address Regions	• 4 standard only • 4 standard + 2 attribute • 4 standard + 4 attribute	GCR_CONFIGNUM_ADDR _REGIONS
Default GCR base address and writeability	Any 32KB-aligned physical address Hardwired or programmable	GCR_BASE
Number of MMIO ports	0, 1, 2	GCR_CONFIG.NUMMMIO
Base GCR address set by software	Yes/No	N/A
Boot in EVA mode	Yes/No	GCR_Cx_RESET_EXT _BASE.EVAReset
Use legacy exception vector boot	Yes/No	GCR_Cx_RESET_EXT_ BASE.Legacy UseExceptionBase
Default Exception Base for each CPU	Any 4KB-aligned physical address	GCR_Cx_RESET_BASE
Boot exception vector overlay region size	1 MB to 256 MB	GCR_Cx_RESET_EXT _BASE.BEV ExceptionBaseMask
Custom GCR register block	Enabled or not	GCR_CUSTOM_STATUSG.GU _EX
L2 Hit Latency Reduction Bypass Enables	TRU bypass, RHR bypass, IVU bypass	N/A
CM Internal Buffer Sizes		
RBB Depth	6,8,10,12,16 per port	N/A
IWDB Depth	4,8,12,16	N/A
PIQ Depth	8, 12, 16	N/A
RNCQ Depth	4, 8	N/A
AAT Depth	8,12,16	N/A
IMQ Depth	4, 8	N/A
RWDQ Depth	0, 4, 8	N/A
IO Read Buffer Depth	8, 12, 16	N/A
IO Write Buffer Depth	8, 12, 16	N/A

Option	Choices	Software Visibility
Global Interrupt Controller Options		
Include Internal GIC	Yes (default) / No (GIC not included)	CM_GCR_GIC_STATUS.GIC _EX
Number of system interrupts	8 - 256 in multiples of 8	GIC_SH_ CONFIGNUMINTERRUPTS
Local routing of CPU sourced interrupts (per VPE)	Present or not	N/A
External interrupt controller (EIC)	Present or not	CONFIG3.VEIC
Boot in EIC mode	Yes/No	CONFIG3.VEIC
ITU Options		
Select ITU	Yes/No	N/A
Number of single entry mailboxes	0, 1, 2, 4, 8, or 16	ITCAddressMap1.NumEntries
Number of 4 entry FIFOs	0, 1, 2, 4, 8, or 16	
Cluster Power Controller Options		
CPC Version	Standard (default) / CPC Basic / CPC-External	CPC_REVISION _REG.MAJOR _ REV[7:6]
Microstep delay in cycles	1 - 1024	CPC_SEQDEL _REG.MICROSTEP
RailEnable delay	1 - 1024	CPC_RAIL_REG.RAILDELAY
Reset Width	5 - 1024	CPC_RESETLEN _REG.RESETLEN
Power Gating Enabled	Enabled or not	N/A
Clock Tree Root Gating Enabled for low power use	Enabled or not	N/A
IOCU Options		
Number of IOCU's	0, 1, or 2	GCR_CONFIGNUMIOCU
IODB implementation style	Flops or generator	N/A
MConnID mask	0 - 8 bit	N/A
L2 Cache Options		
Cache Size	0, 32, 64,128, 256, 512, 1024, 2048, 4096, or 8192 KBytes	
Cache Line Size	32 bytes or 64 bytes	
Cached Memory port data width	64-bit, 128-bit, or 256-bit	N/A
Number of outstanding cached memory reads	8,12,16,32	N/A
Number of outstanding cached memory writes	8,12,16,32	N/A
Global Options (applicable to multiple blocks)		
Clock gating	Top-level, or none	N/A

5 Revision History

Document Revision	Date	Description
1.0	January 31, 2018	Initial release RC1.0
1.1	March 31, 2018	Release 2.0
1.2	April 30, 2018	Release 2.1 Updated sections: • 1.1 - MPU features • Figure 2 - CPU Core Block Diagram • 2.2 - nano MIPS Release 6 ISA • 2.5.6 - Memory Management Options: added MPU section • 3.2.1.a - Port Arbitration Priority