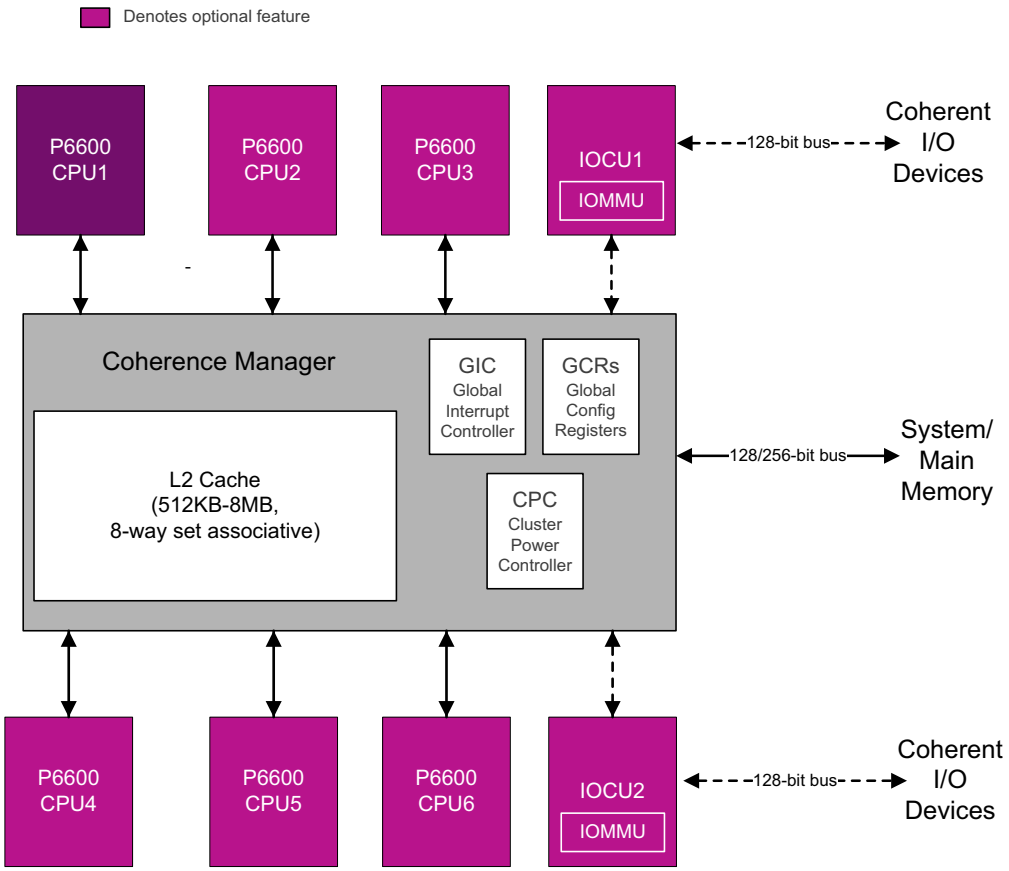
P6600 Multiprocessing System Datasheet

August 26, 2016

The P6600™ series of high performance multi-core microprocessor cores provides best in class power efficiency for use in system-on-chip (SoC) applications. The P6600 Multiprocessing System (MPS) combines a deep pipeline with multi-issue out-of order-execution to deliver outstanding computational throughput. The P6600 provides full virtualization support. The P6600 Multiprocessing System is fully configurable/synthesizable and contains up to six MIPS64® P6600 CPU cores, a system level Coherence Manager with integrated L2 cache, a coherent I/O port (IOCU), and optional floating point unit with SIMD functionality.

Figure 1 shows a block diagram of the P6600 Multiprocessing System (MPS). In the P6600 Multiprocessing System, the Coherence Manager (CM2) with the integrated L2 cache streamlines the dataflow. Multi-CPU coherence is handled in hardware by the Coherence Manager. The I/O Coherence Unit (IOCU) supports hardware I/O coherence by bridging a non-coherent OCP I/O interconnect to the Coherence Manager (CM2) 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 and/or voltage supply to idle cores.

Figure 1 P6600 Multiprocessing System Block Diagram



1. P6600 Features

P6600 Multiprocessor System is feature rich with the most current MIPS64 architecture, new CPU and system level features designed for the performance and features required for tomorrow's mainstream connected consumer electronics including smartphones, tablets, connected TVs and set-top boxes.

1.1 MIPS Architecture

P6600 Multiprocessing System has three key architecture features that sets the core's foundation.

1.1.1 MIPS64™ Release 6 Architecture

Imagination's MIPS64® architecture, an industry standard, is the foundation of the P6600 product offering. The MIPS64 architecture provides a solid high-performance foundation by incorporating powerful features, standardizing privileged mode instructions, supporting past ISAs, and provides a seamless upgrade path from the MIPS32 architecture. MIPS64 is based on a fixed-length, regularly encoded instruction set, and it uses a load/store data model. It is streamlined to support optimized execution of high-level languages. Arithmetic and logic operations use a three-operand format, allowing compilers to optimize complex expressions formulation. Availability of 32 general-purpose registers enables compilers to further optimize code generation by keeping frequently accessed data in registers.

MIPS64 provides backward compatibility, standardizing privileged mode, and memory management, and provides the information through the configuration registers. The MIPS64 architecture enables real-time operating systems and application code to be implemented once and reused.

1.1.2 MIPS® SIMD Architecture

SIMD (Single Instruction Multiple Data), important technology for modern CPU designs that improves performance by allowing efficient parallel processing of vector operations. A non-programmable hardware aids the CPU and GPU by handling heavy-duty multimedia codecs, the MIPS® SIMD Architecture (MSA) technology incorporates a software-programmable solution into the CPU to handle emerging codecs or a small number of functions not covered by dedicated hardware. This programmable solution allows for increased system flexibility. In addition, the MSA is designed to accelerate many compute-intensive applications by enabling generic compiler support.

1.1.3 MIPS® Virtualization

To address security, privacy and reliability concerns in a wide range of devices, Imagination has added hardware supported virtualization technology into P6600 core. The hardware virtualization support enables Imagination's processors to be OmniShield-ready. OmniShield is Imagination's security technology which ensures that applications that need to be secure are effectively and reliably isolated from each other, as well as protected from non-secure applications.

Virtualization can be achieved with software only (para-virtualized) or with hardware assistance (fully virtualized). The core element of virtualization is the hypervisor, a small body of trusted and privileged code that sits above the hardware, managing and orchestrating all of the SoC resources. It manages the resources by defining access policies for each execution environment or "guest." Guests are isolated from each other, but can communicate with the hypervisor and with each other via secure APIs. This ensures the reliability of the system by allowing the rest of the guests to operate reliably even if one of the guests crashes. The hypervisor manages all memory I/O privileges of the subsystems.

1.2 System Level Features

- Up to six coherent MIPS64 P6600 CPU cores
- Superscalar, variable-length, out -of-order data return
- Support for power management with multiple power domains
- Cluster Power Controller (CPC) to shut down idle CPU cores to save power
- Hardware I/O coherence unit (IOCU)
- Hardware Virtualization Module Support
- Cache-to-cache data transfers
- Speculative memory reads to reduce latency
- Integrated 8-way set associative L2 cache controller supporting 512 KB to 8 MB cache sizes
- Shared L2 cache controller supporting 512 KB to 8 MB cache sizes
- Separate clock ratios on memory and IOCU OCP ports
- Clock ratio of 1:1 between Core, CM2, and L2 cache
- SOC system interface supports OCP version 2.1 protocol with 32- or 40-bit address and 128-bit or 256-bit data paths
- EJTAG Debug port supporting multi-processor debugging
- MIPS PDtrace
- Full scan design achieves test coverage in excess of 99% with memory BIST for internal SRAM arrays

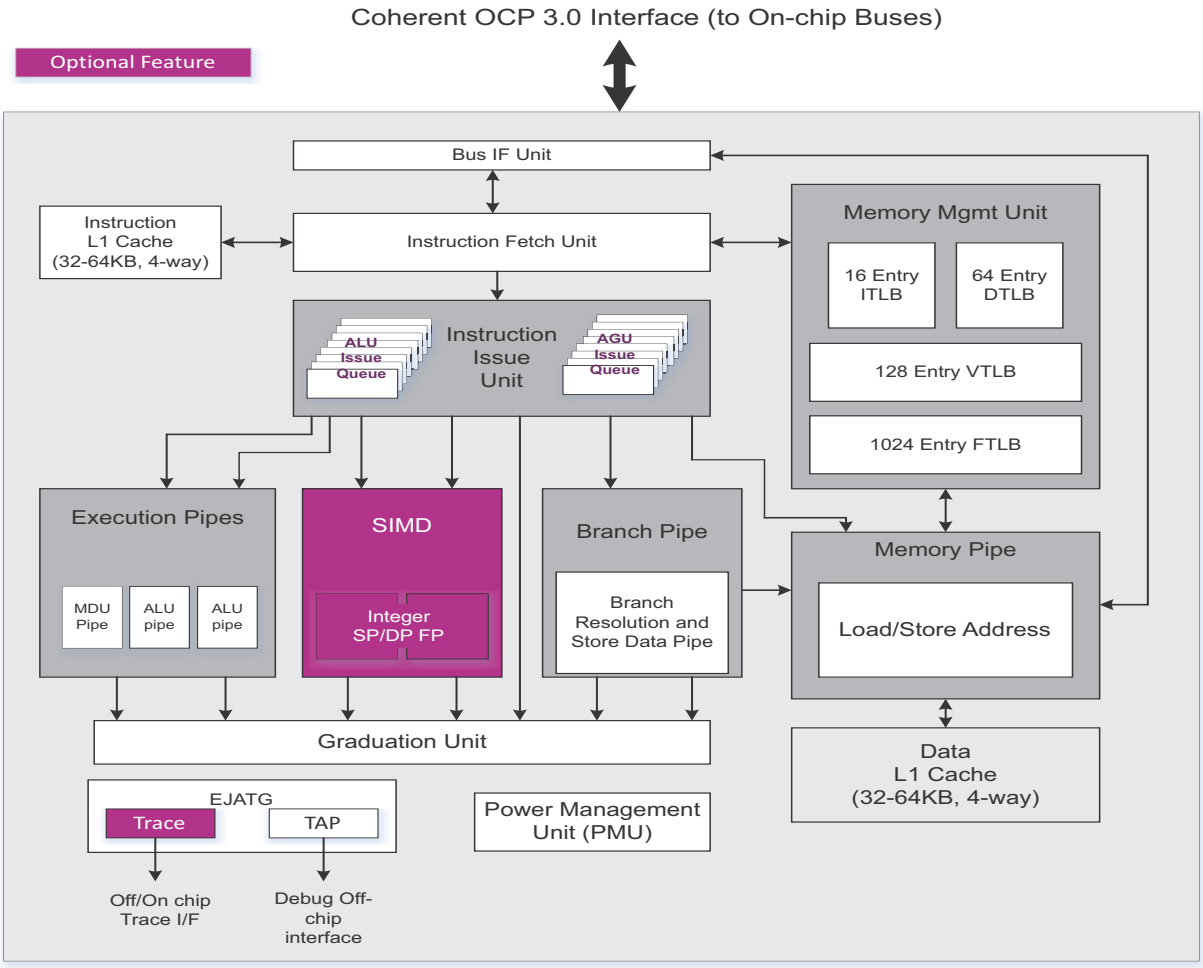
1.3 CPU Core Level Features

- 40-bit addressing
- Quad issue integer and dual issue 128-bit (integer/floating point) execution pipes
- Sophisticated branch prediction with fully associative Level 1 BTB
- Floating Point Unit with SIMD support and Out-Of-Order (OOO) execution
- Virtualization support
- Instruction Fetch Unit (IFU) with 4 instructions fetched per cycle
- Programmable Memory Management Unit with large first-level ITLB/DTLB backed by fast on-core second-level variable page size TLB (VTLB) and fixed page size TLB (FTLB):
- L1 Instruction and Data Caches can be configured as 32 or 64 KB per cache

2. P6600 CPU Core

Figure 2 shows a block diagram of a single P6600 core. The logic blocks in this diagram are described in the following sections.

Figure 2 P6600™ Core Block Diagram



For more information on the P6600 core in a multiprocessing environment, refer to Section 3. “Multiprocessing System”.

2.1 Instruction Fetch Unit

The Instruction Fetch Unit (IFU) fetches instructions from the instruction cache and supplies them to the Instruction Issue Unit (IIU). The IFU can fetch up to four MIPS64 instructions at a time from the 4-way associative instruction cache. Instructions can also be fetched immediately from refill buffers in the event of an instruction cache miss.

The IFU employs sophisticated branch prediction and instruction supply strategies. The main predictor consists of three 2048-entry global branch history tables (BHT) that are indexed by different combinations of instruction PC and global history. A proprietary scheme is used to combine information from the three arrays to make a branch direction prediction.

The IFU also has a hardware-based return prediction stack to predict subroutine return addresses.The main predictor corrects target mispredicts from lower-level predictors without paying a full branch resolution penalty. The IFU supports fully out-of-order branch resolution.

The IFU has a 16-entry micro-Instruction TLB (ITLB) used to translate the virtual address into a physical address and used to compare against tags in the instruction cache to determine a hit. Refer to Section 2.6 "Memory Management Unit (MMU)" for more information.

A 24-entry instruction buffer decouples the instruction fetch from the execution. To maximize performance, some 'bonding' (or concatenation) of instructions is done at this stage while other types of instruction 'bonding' are performed downstream.

The IFU can also be configured to allow for hardware prefetching of cache lines on a miss. This mechanism provides excellent performance without incurring the area, power and latency costs of more overly complicated branch or instruction prefetch strategies.

The Global History register is internal to the IFU block and supports a novel history computation scheme that factors different information into the history for different kinds of control transfer instructions.

The P6600 level 1 (L1) instruction cache incorporates 'next fetch way' hit prediction logic. This allows the IFU to power on only those cache tag and data arrays that will provide the final instruction bytes and contributes to low power consumption.

2.2 Instruction Issue Unit (IIU)

The Instruction Issue Unit (IIU) is responsible for receiving instructions from the IFU and dispatching them to the out-of-order instruction scheduling windows and global instruction tracking window at a rate of 4 instructions per cycle.

The IIU tracks dynamic data flow dependencies between operations and issues them to the various pipes as efficiently as possible. Two schedulers service the various integer pipes.

The schedulers employ multiple dependency wake-up and pick schemes to enable age-based scheduling at high frequency. These two schedulers provide superior performance and power characteristics.

The IIU helps to 'bond' load and store operations whereby two 32-bit loads or 64-bit or stores to adjacent locations are 'bonded' or concatenated into one 64-bit or 128-bit memory access. This allows a factor of two improvement in certain memory intensive codes.

The IIU also keeps track of the progress of each instruction through the pipeline, updating the availability of operands in the 'rename map' and in all dependent instructions. Renamed instructions are steered to the most appropriate schedulers, taking opcode and other information into account.

The IIU also keeps track of global pipeline flushes, adjusting the rename map and other control structures to deal with interrupts, exceptions and other unexpected changes of control.

2.3 Graduation Unit (GRU)

The Graduation Unit (GRU) is responsible for committing execution results and releasing buffers and resources used by these instructions. The GRU is also responsible for evaluating the exception conditions reported by execution units and taking the appropriate exception. Asynchronous interrupts are funneled into the GRU, which prioritizes those events with existing conditions and takes the appropriate interrupt.

After processing the exception conditions, the GRU performs the following functions:

- Destination register(s) are updated and the completion buffers are released.
- Graduation information is sent to the IIU so it can update the rename maps to reflect the state of execution results (such as GPRs).
- Resolved branch information is sent to the IFU so that branch history tables can be updated and if needed, a pipeline redirect can be initiated. If sequential control flow is aborted for any reason, the GRU signals all core units to flush and recover microarchitectural state. After recovery is complete, it allows the IIU to resume dispatching instructions.

2.4 Level 1 Instruction Cache

The Level-1 (L1) instruction cache is configurable at 32 or 64 KB in size and is organized as 4-way set associative. Up to four instruction cache misses can be outstanding. The instruction cache is virtually indexed and physically tagged to make the data access independent of virtual to physical address translation.

Each instruction cache entry contains a tag portion, a data portion, and a way select portion.

An instruction tag entry holds 21 - 29 bits of physical address, a valid bit, a lock bit, and a parity bit. The data entry consists of 256 bits (8 MIPS64 instructions) of data and 32 bits of parity for a total of 288 bits. The way-select entry contains a 6 bit least-recently-used (LRU) field.

The P6600 core supports instruction-cache locking. Cache locking allows critical code segments to be locked into the cache on a “per-line” basis, enabling the system programmer to maximize the performance of the system cache.

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

The P6600 core implements virtual aliasing for the instruction cache, although this function can be disabled by the user.

2.5 Level 1 Data Cache

The Level 1 (L1) data cache is configurable at 32 or 64 KB in size. It is also organized as 4-way set-associative. Data cache misses are non-blocking and up to nine misses may be outstanding. The data cache is virtually indexed and physically tagged to make the data access independent of virtual-to-physical address translation. To achieve the highest possible frequencies using commercially available SRAM generators, cache access and hit determination are spread across three pipeline stages, dedicating an entire cycle for the SRAM access.

Each instruction cache entry contains a tag portion, a data portion, a way-select portion, and a dirty status portion.

- A data tag entry holds 21 bits of physical address in 32-bit addressing mode (29 bits in 40-bit addressing mode), a valid bit, a state bit, and a parity bit, making a total of 24 - 32 bits per tag entry.
- The data entry consists of 256 bits consisting of 32 bytes of data of data and 32 bits of parity for a total of 288 bits. The way-select entry contains a 6 bit least-recently-used (LRU) field, a 4-bit lock field, and a 4-bit lock parity field for a total of 14 bits.
- The Dirty state entry contains a 4-bit dirty field and a 4-bit dirty parity field.

The P6600 core supports a data-cache locking mechanism identical to that used in the instruction cache. Critical data segments are locked into the cache on a “per-line” basis. The locked contents can be updated on a store hit, but are not selected for replacement on a cache miss.

The P6600 core implements virtual aliasing for the data cache. This function is managed in hardware and is transparent to the user.

2.6 Memory Management Unit (MMU)

The P6600 core's Memory Management Unit (MMU) is primarily responsible for converting virtual addresses to physical addresses and providing attribute information for different segments of memory. The P6600 MMU contains the following Translation Lookaside Buffer (TLB) types:

- Instruction TLB (ITLB)
- Data TLB (DTLB)
- Variable Page Size Translation Lookaside Buffer (VTLB)
- Fixed Page Size Translation Lookaside Buffer (FTLB)

2.6.1 Instruction TLB (ITLB)

The ITLB is a 16-entry high speed TLB dedicated to performing translations for the instruction stream. The ITLB maps only 4 KB or 16 KB pages. Larger pages are split into smaller pages of one of these two sizes and installed in the ITLB.

The ITLB is managed by hardware and is transparent to software. The larger VTLB and FTLB structures are used as a backup structure for the ITLB. If a fetch address cannot be translated by the ITLB, the VTLB/FTLB attempts to translate it in the following clock cycle or when available. If successful, the translation information is copied into the ITLB for future use.

2.6.2 Data TLB (DTLB)

The DTLB is a 32-entry high speed TLB dedicated to performing translations for the data stream. The DTLB maps only 4 KB or 16 KB pages. Larger pages are split into one of these configured sizes and installed in the DTLB.

The DTLB is managed by hardware and is transparent to software. The larger VTLB and FTLB structures are used as a backup structure for the DTLB. If a fetch address cannot be translated by the DTLB, the VTLB/FTLB attempts to translate it in the following clock cycle or when available. If successful, the translation information is copied into the DTLB for future use.

2.6.3 Variable Page Size TLB (VTLB)

The VTLB is a fully associative variable translation lookaside buffer with 64 dual entries that can map variable size pages from 4KB to 256MB. When an instruction address is calculated, the virtual address is first compared to the contents of the ITLB and DTLB. If the address is not found in either the ITLB or DTLB, the VTLB/FTLB is accessed. If the entry is found in the VTLB, that entry is then written into the ITLB or DTLB. If the address is not found in the VTLB, a software TLB exception is taken. For data accesses, the virtual address is looked up in the VTLB only, and a miss causes a TLB exception.

2.6.4 Fixed Page Size TLB (FTLB)

The FTLB is 512 dual entries organized as 128 sets and 4 ways. Each set of each way contains dual data RAM entries and one tag RAM entry. If the tag RAM contents match the requested address, either the low or high RAM location of the dual data RAM is accessed depending on the state of the most-significant-bit (MSB) of the offset portion of the virtual address (VPN2). Each RAM location can only map a fixed page size, which is configurable to 4KB or 16KB.

2.6.5 Enhanced Virtual Address

The P6600 core supports a programmable memory segmentation scheme called Enhanced Virtual Address (EVA). EVA allows for more efficient use of 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)*.

2.6.6 Virtualization Support

Virtualization defines a set of extensions to the MIPS64 Architecture for efficient implementation of virtualized systems.

Virtualization is enabled by software. The key element is a control program known as a Virtual Machine Monitor (VMM) or hypervisor. The hypervisor is in full control of machine resources at all times.

The hypervisor is responsible for managing access to sensitive resources, maintaining the expected behavior for each VM, and sharing resources between multiple VMs.

In a traditional operating system, the kernel (or supervisor) typically runs at a higher level of privilege than user applications. The kernel provides a protected virtual-memory environment for each user application, inter-process communications, IO device sharing and transparent context switching. The hypervisor performs the same basic functions in a virtualized system, except that the hypervisor's clients are full operating systems rather than user applications.

The virtual machine execution environment created and managed by the hypervisor consists of the full Instruction Set Architecture (ISA), including all Privileged Resource Architecture (PRA) facilities, and any device-specific or board-specific peripherals and associated registers. It appears to each guest operating system as if it is running on a real machine with full and exclusive control.

The Virtualization Module enables full virtualization, and is intended to allow VM scheduling to take place while meeting real-time requirements, and to minimize costs of context switching between VMs.

2.7 Execution Pipelines

The P6600 core contains the following execution pipelines:

- Arithmetic Logic Pipeline
- Multiply-Divide Pipeline
- Memory Pipeline
- Branch Pipeline
- Two FPU3 Pipelines

Each of these execution units is described in the following subsections. Instructions intended for the arithmetic logic pipeline are driven by the out-of-order ALU Decode and Dispatch queue inside the Instruction Issue Unit (IIU) as shown in Figure 2. The other four pipelines are driven by the out-of-order Address Generation unit (AGU) Decode and Dispatch queue also located in the IIU.

2.7.1 Arithmetic Logic Pipeline

The arithmetic unit pipeline consists of one execution unit, called the ALU (Arithmetic Logic Unit), which performs integer instructions such as adds, shifts and bitwise logical operations with a single cycle latency. If the IIU decodes a single-cycle instruction, it is usually sent to the ALU dispatch queue that feeds the arithmetic unit pipeline. This pipeline also contributes to performing 'bonded' loads. Refer to Section 2.2 for a definition of instruction 'bonding'.

2.7.2 Multiply/Divide Pipeline

The multiply/divide pipeline executes integer multiplies, integer divides, and integer multiply-accumulate instructions. The multiply/divide pipeline incorporates a new very high-speed integer divider.

The MDU consists of a 64-bit multiplier, result/accumulation registers, a divide state machine, and all necessary multiplexers and control logic.

The MDU supports execution of one multiply or multiply-accumulate operation every clock cycle whereas divides can be executed as fast as one every four cycles.

2.7.3 Memory Pipeline

The memory pipeline primarily contains the LSU (Load Store Unit). The LSU is responsible for interfacing with the AGU dispatch queue (see Figure 2) and processing load/store instructions to read/write data from data caches and downstream memory.

It is capable of handling loads and stores issued out-of-order. The LSU has the ability to receive loads and stores in almost any order enables very high performance compared to an in-order machine. Such instruction-level parallelism allows maximum utilization of the memory pipe resources with minimal area and power.

The LSU can execute loads and stores at twice the rate of regular operations by concatenating data from two 32-bit or 64-bit memory to form a single 64-bit or 128-bit entity, respectively. This 'bonding' of instructions allows the LSU to provide almost all the benefits of dual memory access pipes without incurring the area and power costs of multiple tag, data and TLB structures.

The memory pipeline receives instructions from the Instruction Issue Unit (IIU) and interfaces to the L1 data cache. Loads are non-blocking in the P6600 core. Loads that miss in the data cache are allowed to proceed with their destination register marked unavailable. Consumers of this destination register are held back and replayed as needed after the cache miss has been serviced by the downstream memory subsystem, which includes the high performance L2 cache.

Graduated load misses and store hits and misses are sent in order to the Load/Store Graduation Buffer (LSGB). The LSGB has corresponding data and address buffers to hold all relevant attributes.

An 8-entry Fill Store Buffer (FSB) tracks outstanding fill or copy-back requests. It fills the data cache at the rate of 128-bits per cycle when an incoming line is completely received. Each FSB entry can hold an entire cache line.

The Load Data Queue (LDQ) keeps track of outstanding load misses and forwards the critical data to the main pipe as soon as it becomes available.

Hardware anti-aliasing allows using the core with operating systems that do not support software page coloring. The fully-associative DTLB operates a clock earlier in the LSU pipeline, making use of fast add-and-compare logic to enable virtual address to physical address translations that do not require the area and power expense of virtual tagging. All of this is done completely transparent to software.

2.7.4 Branch Pipeline

The Branch pipeline performs the following functions:

- Executes Branch and Jump instructions
- Performs Branch resolution
- Performs Jump resolution
- Sends the redirect to the Instruction Fetch Unit (IFU)
- Performs a write-back to the Link registers

2.7.5 Floating Point Pipelines

The optional Floating Point Unit with SIMD contains two execution pipelines. One pipeline executes SIMD logical ops, SIMD integer adds. The FP compares and stores. The other pipeline executes SIMD integer multiplies, SIMD vector shuffles, FP adds, FP multiplies, and FP divides.

For more information, refer to Section 2.12 "Floating Point Unit".

2.8 Bus Interface (BIU)

The BIU controls a 128-bit interface to the CM2. The interface implements the Open Core Protocol (OCP).

2.8.1 Write Buffer

The BIU contains a merging write buffer. This buffer stores and combines write transactions before issuing them to the external interface. The write buffer is organized as eight, 32-byte buffers. Each buffer can contain data from a single 32-byte aligned block of memory.

When using the write-through cache policy or performing uncached accelerated writes, the write buffer significantly reduces the number of write transactions on the external interface and reduces the amount of stalling in the core caused by the issuance of multiple writes in a short period of time.

The write buffer also holds eviction data for write-back lines. The load-store unit extracts dirty data from the cache and sends it to the BIU. In the BIU, the dirty data is gathered in the write buffer and sent out as a bursted write.

For uncached accelerated writes, the write buffer can gather multiple writes together and then perform a bursted write in order to increase the efficiency of the bus.

Gathering of uncached accelerated stores can start on any arbitrary address and can be combined in any order within a cache line. Uncached accelerated stores that do not meet the conditions required to start gathering are treated like regular uncached stores.

2.9 System Control Coprocessor (CP0)

In the 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, and whether interrupts are enabled or disabled. Configuration information, such as cache size and associativity, and the presence of features like a floating point unit, are also available by accessing the CP0 registers.

CP0 also contains the state used for identifying and managing exceptions. Exceptions can be caused by a variety of sources, including boundary cases in data, external events, or program errors.

2.10 Interrupt Handling

The P6600 core supports six hardware interrupts, two software interrupts, a timer interrupt, and a performance counter interrupt. These interrupts can be used in any of three interrupt modes, as defined in the MIPS64 Architecture:

- Interrupt compatibility mode.
- 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.

2.11 Modes of Operation

The P6600 core supports four modes of operation:

- Two user modes (guest and root), most often used for application programs.
- Two supervisor modes (guest and root) provides an intermediate privilege level with access to the *ksseg* (kernel supervisor segment) address space.
- Two kernel modes (guest and root), typically used for handling exceptions and operating system kernel functions, including CP0 management and I/O device accesses.
- Debug mode is used during system bring-up and software development. Refer to Section 2.14 "EJTAG Debug Support" for more information on debug mode.

2.12 Floating Point Unit

The P6600 core features an optional IEEE 754 compliant 3rd generation Floating Point Unit with SIMD.[1]

The FPU3 contains thirty-two, 128-bit vector registers shared between SIMD and MIPS64 instructions.

SIMD instructions enable:

- Efficient vector parallel arithmetic operations on integer, fixed-point and floating-point data.
- Operations on absolute value operands.
- Rounding and saturation options available.
- Full precision multiply and multiply-add.
- Conversions between integer, floating-point, and fixed-point data.
- Complete set of vector-level compare and branch instructions with no condition flag.
- Vector (1D) and array (2D) shuffle operations.
- Typed load and store instructions for endian-independent operation.

The FPU3 with SIMD is fully synthesizable and operates at the same clock speed as the CPU. The IIU can issue up to two instructions per cycle to the FPU3.

The FPU3 contains two execution pipelines for floating point and SIMD instruction execution. These pipelines operate in parallel with the integer core and do not stall when the integer pipeline stalls. This allows long-running FPU3/SIMD operations such as divide or square root, to be partially masked by system stall and/or other integer unit instructions.

An out-of-order scheduler in the FPU3 issues instructions to the two execution units. The exception model is 'precise' at all times.

2.13 P6600 Core Power Management

The P6600 core offers several power management features, that support low-power designs, such as active power management and power-down modes of operation. The P6600 core is a static design that supports slowing or halting the clocks to reduce system power consumption during idle periods.

You can also use the Cluster Power Controller (CPC) to control your power management. Refer to "Cluster Power Controller (CPC)" on page 13 for more details.

1. Requires separate MIPS license.

2.13.1 Instruction-Controlled Power Management

The Instruction Controlled power-down mode is invoked through execution of an instruction. When the WAIT instruction is executed, the internal clock is suspended; however, the internal timer and some of the input pins continue to run. When the CPU is in this instruction-controlled power management mode, any interrupt, NMI, or reset condition causes the CPU to exit this mode and resume normal operation.

The P6600 core asserts a sleep signal whenever it has entered low-power mode (sleep mode). The core enters sleep mode when all bus transactions are complete and there are no running instructions.

The WAIT instruction can put the processor in a mode where no instructions are running. When the WAIT instruction is seen by the Instruction Fetch Unit (IFU), subsequent instruction fetches are stopped. The WAIT instruction is dispatched down the pipe and graduated. Upon graduation of the WAIT, the GRU waits for the processor to reach a quiescent state and allows the processor to enter sleep mode.

2.14 EJTAG Debug Support

The P6600 core includes an Enhanced JTAG (EJTAG) block for use in software debugging of application and kernel code. For this purpose, in addition to standard user/supervisor/kernel modes of operation, the P6600 core provides a Debug mode.

Debug mode is entered when a debug exception occurs and continues until a debug exception return instruction is executed. During this time, the processor executes the debug exception handler routine.

The EJTAG interface operates through the Test Access Port (TAP), a serial communication port used for transferring test data in and out of the P6600 core. In addition to the standard JTAG instructions, special instructions defined in the EJTAG specification define which registers are selected and how they are used.

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.

During synthesis, the P6600 core can be configured to support the following breakpoint options:

- Zero instruction, zero data breakpoints
- Four instruction, 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 address space identifier (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 the virtual address, ASID value, and the load/store value.

2.14.1 Fast Debug Channel

The P6600 CPU includes the EJTAG Fast Debug Channel (FDC) for efficient bi-directional data transfer between the CPU and the debug probe. Data is transferred serially via the TAP 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.

2.14.2 PDtrace

The P6600 core includes trace support for real-time tracing of instruction addresses, data addresses, performance counters, and processor pipeline inefficiencies. The trace information is collected in an on-chip or off-chip memory, for postcapture processing by trace regeneration software. Software-only control of trace is possible in addition to probe-based control.

An on-chip trace memory may be configured in size from 256B to 8 MB; it is accessed either through load instructions or the existing EJTAG TAP interface, which requires no additional chip pins. Off-chip trace is managed with the PIB2 (2nd-generation Probe Interface Block) hardware that ships with the product. It provides a selectable trace port width of 8, or 16 pins plus DDR clock. Trace data is streamed on these pins and captured using the MIPS Navigator™ Pro probe Other supported probes include DA-net and Joyner.

3. Multiprocessing System

The Multiprocessing System (MPS) consists of the logic modules: CPC, CM2, IOCU, GIC, and GCR (Figure 1). Each block is described throughout this section. In additional the clocking and debugging features are also described in this section.

3.1 Cluster Power Controller (CPC)

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 Cluster Power Controller (CPC). The CPC handles the power shutdown and ramp-up of all CPUs in the cluster. Any P6600 CPU that supports power-gating features is managed by the CPC.

The CPC also organizes power-cycling of the CM2 dependent on the individual core status and shutdown policy. Reset and root-level clock gating of individual CPUs are considered part of this sequencing.

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. Pin settings determine the course of action for each core after a CPC reset.

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

In case of a system cold start, after reset is released, the CPC powers up the P6600 CPUs as directed in the CPC cold start configuration pins. If at least one CPU has been chosen to be powered up on system cold start, the CM2 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. The CPC, after power up of any CPU, provides a test logic reset sequence per domain to initialize TAP logic.

There are memory-mapped registers that can set the value for each CPU's *SI_ExceptionBase* pins. 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 2 (CM2)

The Coherence Manager with integrated L2 cache (CM2) is responsible for establishing the global ordering of requests and for collecting the intervention responses and sending the correct data back to the requester. A high-level view of the request/response flow through the CM2 is shown in Figure 3. Each of the blocks is described in more detail in the following subsections.

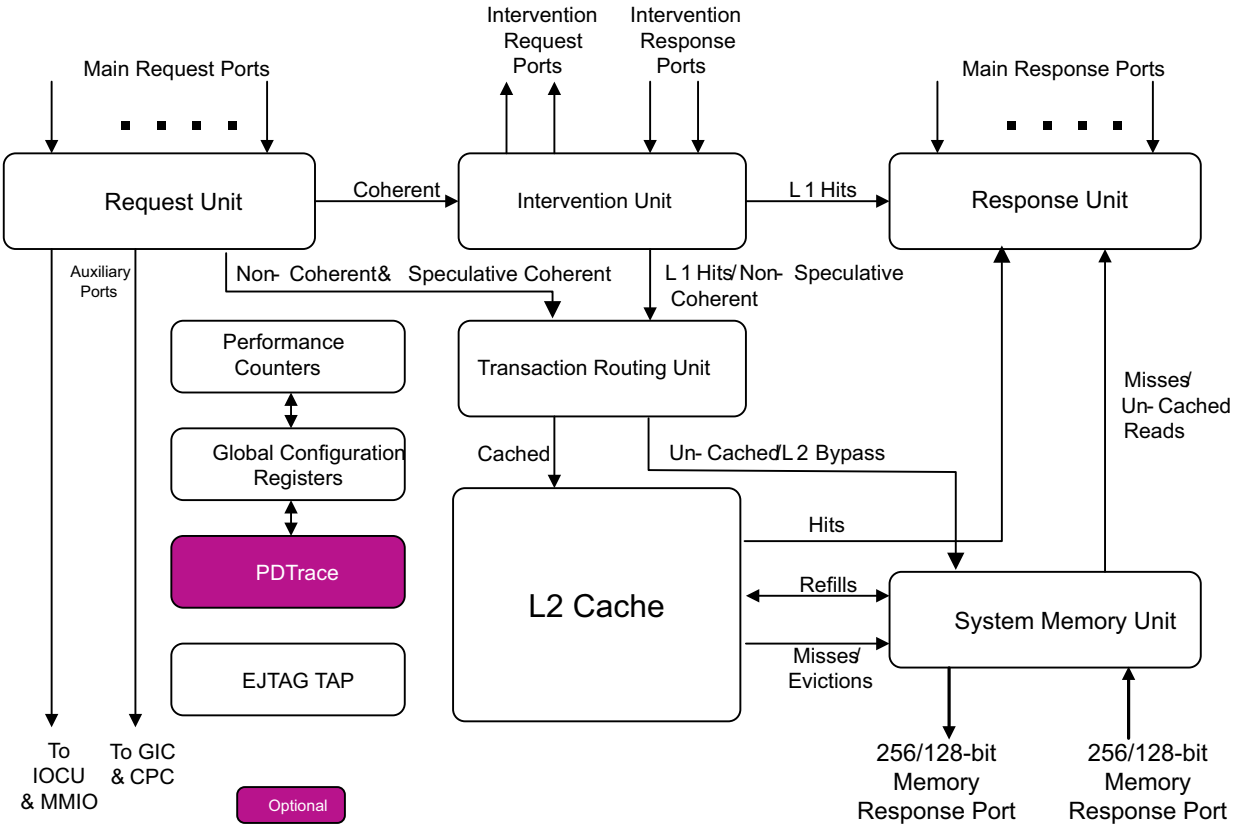
3.2.1 Request Unit (RQU)

The Request Unit (RQU) receives OCP bus transactions from multiple CPU cores and/or I/O ports, serializes the transactions and routes them to the Intervention Unit (IVU), Transaction Routing Unit (TRU), or an auxiliary port used to access a configuration registers or memory-mapped IO. The routing is based on the transaction type, the transaction address, and the CM2's programmable address map.

3.2.2 Intervention Unit (IVU)

The Intervention Unit (IVU) interrogates the L1 data caches by placing requests on the intervention OCP interfaces. Each processor responds with the state of the corresponding cache line. If the processor has the corresponding data in its L1 data cache, it provides the data with its response. If the original request was a read, the IVU routes the data to the original requestor via the Response Unit (RSU).

Figure 3. Coherence Manager 2 (CM2) with Integrated L2 Cache Block Diagram



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 the L2 cache or memory.
- Data returned from the CPU is forwarded to the Response Unit (RSU) to be returned 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 memory OCP port. For an L2 refill, the SMU reads the data from an internal buffer and issues the refill request to the L2 pipeline.

3.2.4 Response Unit (RSU)

The RSU takes responses from the SMU, L2, IVU, or auxiliary port and places them on the appropriate OCP interface. Data from the L2 or SMU is buffered inside a buffer associated with each RSU port.

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 and IVU, and 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 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 15 misses are outstanding. The cache is physically indexed and physical tagged.

- *L2 Cache Configuration* provides the following L2 cache configuration options: 512KB, 1MB, 2MB, 4MB, and 8MB
- *L2 Pipeline Tasks* manages 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 are
  - Supports write-back operation.
  - Pseudo-LRU replacement algorithm
  - Programmable wait state generator to accommodate a wide variety of SRAMs.
  - L2 prefetcher. Hardware recognizes streams of sequential accesses and prefetches memory data into the L2 cache.
  - Operates at same clock frequency as CPU.
  - Cache line locking support
  - ECC support for resilience to soft errors
  - Single-bit error correction and 2-bit error detection support for Tag and Data arrays
  - Single bit detection only for WS array
  - Bypass mode

- Fully static design: minimum frequency is 0 MHz
- Sleep mode
- Memory BIST for internal SRAM arrays, with support for integrated (March C+, IFA-13) or custom BIST controller.

3.2.7 CM2 Configuration Registers

The Registers block (GCR) contains the control and status registers for the CM2. It also contains registers that control the Trace Funnel, EJTAG TAP state machine, and other multi-core features.

3.2.8 Performance Counter Unit

The CM2 implements a Performance Counter Unit (PERF) that contains the performance counters and associated logic.

3.2.9 Coherence Manager Performance

The CM2 has a number of high performance features:

- 256-bit wide internal data paths throughout the CM2
- 128-bit or 256-bit wide system OCP interface
- Integrated L2 cache provides low latency for L2 cache hits
- CM2 and L2 can process up to 1 request per cycle
- *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 CM2 and it will be 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 (IOCU)

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.

The IOCU acts as an interface block between the Coherence Manager (CM2) and I/O devices. Reads and writes from I/O devices may access the L1 and L2 caches by passing through the IOCU and the CM2. Each request from an I/O device may be marked as coherent, non-coherent cached, or uncached. Coherent requests access the L1 and L2 caches. Non-coherent cached requests access only the L2 cache. Uncached requests bypass both the L1 and L2 caches and are routed to main memory.

The IOCU also provides a legacy (without coherent extensions) OCP slave interface to the I/O interconnect for I/O devices to read and write system memory. The design also includes an OCP Master port to the I/O interconnect that allows the CPUs to access registers and memory on the I/O devices.

The IOCU design provides several features for easier integration:

- Supports incremental bursts up to 256 bytes (16 beats of 128b data) on I/O side. These requests are split into cache-line-sized requests on the CM side
- Read responses with different TagIDs may be returned out-of-order
- Integrated I/O Memory Management Unit (IOMMU)

In addition, the IOCU contains the following features used to enforce transaction ordering.

- Writes are issued to the CM in the order they were received.
- The CM provides an acknowledge (ACK) signal to the IOCU when writes are "visible" (guaranteed that a subsequent CPU read will receive that data).

- Non-coherent write is acknowledged after serialization
- Coherent write is acknowledged after intervention complete on all CPUs

3.3.1 Software I/O Coherence

For cases where 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 L1 CACHE operations, the CPU makes a corresponding coherent request. The CM2 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 Global Interrupt Controller

The 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 through relocatable memory-mapped address range.
- Configurable number of system interrupts - from 128 to 1256.
- Support for different interrupt types:
  - Level-sensitive: active high or low.
  - Edge-sensitive: positive-, negative-, or double-edge sensitive.
- Virtualization support allows each interrupt to be mapped to Guest or Root.
- 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 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.

3.5.1 Inter-CPU Debug Breaks

The MPS includes registers that enable cooperative debugging across all CPUs. Each core features a debug output signal 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.

3.5.2 CM2 Control Registers

Control registers in the CM2 allow software to configure and control various aspects of the operation of the CM2. Some of the 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 non-coherent 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 CM2 and controls how errors are handled (ignored, interrupt, etc.).
- *Control Options*: Various features of the CM2 can be disabled or configured. Examples of this are disabling speculative reads and preventing ReadShared requests from being upgraded to Exclusive.

4. Clocking Options

The P6600 core has the following clock domains:

- Cluster domain — This is the main clock domain, and includes all P6600 cores (including optional FPU3) and the CM2 (including Coherence Manager, Global Interrupt Controller, Cluster Power Controller, trace funnel, IOCU, and L2 cache).
- System Domain - The OCP port connecting to the SOC and the rest of the memory subsystem may operate at a ratio of the cluster domain. Supported ratios are 1:1, 1:2, 1:3, 1:4, 1:5, and 1:10.
- TAP domain - This is a low-speed clock domain for the EJTAG TAP controller
- IO Domain - This is the OCP port connecting the IOCU to the I/O Subsystem. This clock may operate at a ratio of the CM2 domain. Supported ratios are the same as the system domain.

5. Design For Test (DFT) Features

The P6600 core provides the following test for determining the integrity of the core.

- Internal Scan: The P6600 core supports 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: The P6600 core provides an integrated memory BIST solution for testing of all internal SRAMs.

Memory BIST can also 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.

6. Configuration Options

The P6600 provides a number of configuration options as shown in Table 1. These are options available to you to select for your P6600 configuration.

Table 1. P6600 Multiprocessing System Configuration Options

Parameter	Configurable Options
Number of Cores	1, 2, 3, 4, 5, or 6
L1 Instruction Cache	32 or 64 KB
L1 Data Cache	32 or 64 KB
MIPS64 + SIMD	None or MIPS64 + SIMD
System Interrupts	128 or 256

Table 1. P6600 Multiprocessing System Configuration Options

Parameter	Configurable Options
L2 Cache	512 KB, 1 MB, 2 MB, 4 MB, or 8 MB
Physical Address Bits	40
Location of Boot Exception Vector	Configurable
External Interface Type	OCP or AXI
External Interface Width	128- or 256-bit
PDtrace	None, on-chip only, off-chip only, both on- and off-chip

7. Revision History

The following table shows the revision history for the P6600 Multiprocessing System data sheet.

Revision	Date	Description
01.00	October 21, 2015	Initial release of the P6600 data sheet.
01.01	December 1, 2015	Fixed error in Figure 2 — P6600 Core Block Diagram.
01.02	August 26, 2016	Add PDtrace to feature list