Power Management in an Embedded Multiprocessor Cluster

Matthias Knoth,
MIPS Technologies, Inc.

Coherent microprocessor clusters have migrated into high-end embedded applications. Implemented within SoCs, designers must meet tight power supply and thermal budgets to fit operational requirements. A typical measure to achieve low-power footprints is performance scaling based on temporal application needs. Symmetrical multiprocessor clusters lend themselves particularly well to this technique. Recent advances in design methodology enable power gating and voltage/frequency scaling of SoC components. In this paper, power management techniques of embedded multiprocessor systems shall be illustrated using MIPS Technologies' MIPS32® 1004K™ Coherent Processing System (CPS)..

Document Number:MD00664
Revision 1.00
March 15, 2009

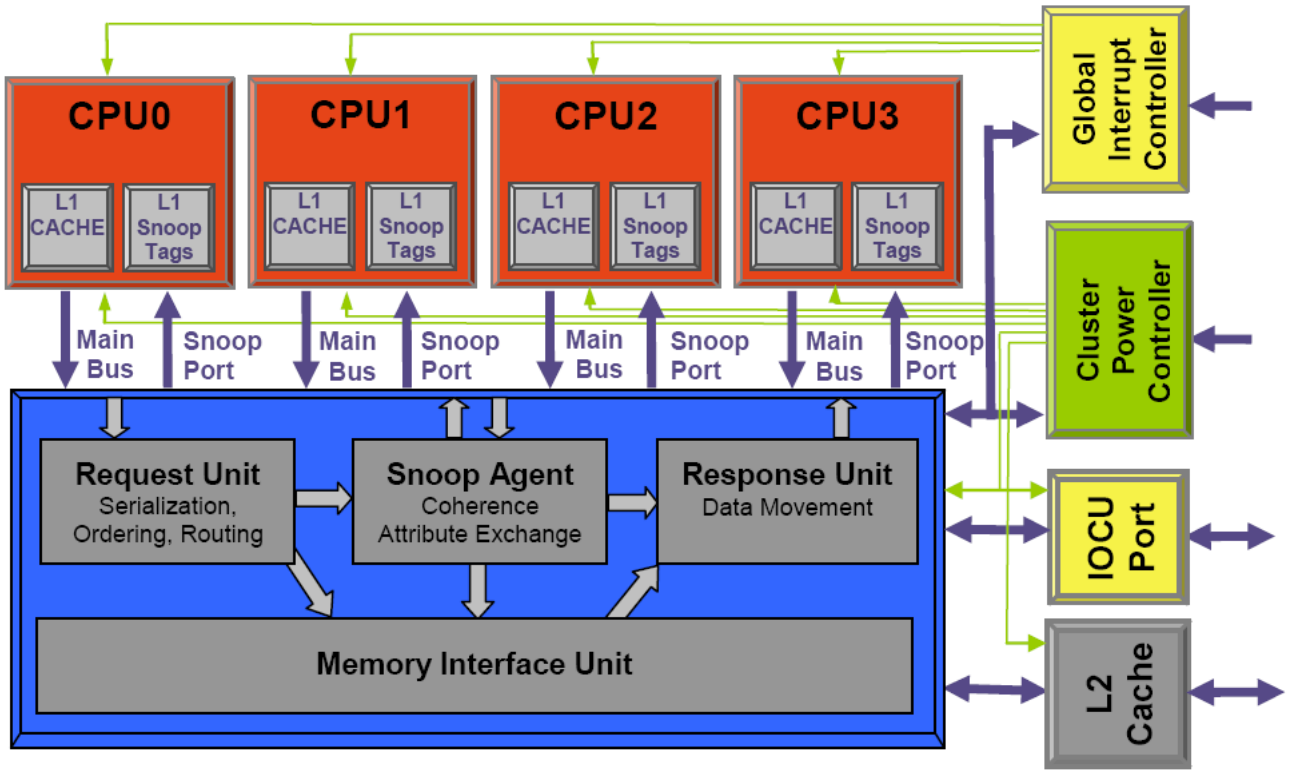
Introduction

Coherent microprocessor clusters, having localized instruction and data caches per CPU, require special techniques to maintain consistency between localized cache contents and their common address region. For embedded systems, designers typically apply snoop-based schemes to maintain memory coherence. This scheme introduces ownership attributes of local cache lines, which are posted throughout the cluster upon intent to use or change.

As a core acquires ownership of a particular cache line with intent to modify - e.g. due to a pending store operation, peer CPUs are notified and will update their cache line on demand. Similarly, localized access to cache lines due to pending load instructions can stay localized if permitted by ownership attributes. Competing access to a particular cache line must be ordered to maintain consistency. The implemented snoop protocol supports “MESI” style cache line attributes to mark lines as modified, exclusive, shared or invalid. Duplication of L1 tags per CPU promotes parallel operation of execution pipeline and snoop request processing.

The MIPS Technologies 1004K™ Coherent Processing System (CPS, Figure 1) consists of up to four microprocessor cores, each providing two independent virtual processor environments and threads.

Figure 1 MIPS32® 1004K™ Coherent Processing System (CPS)



The CPS employs a centralized coherence manager unit (CM) in order to facilitate snoop traffic between cores, routing of requests, ordering, and serialization,. The CM also manages data movement to and from higher levels of memory hierarchy, such as the L2 cache.

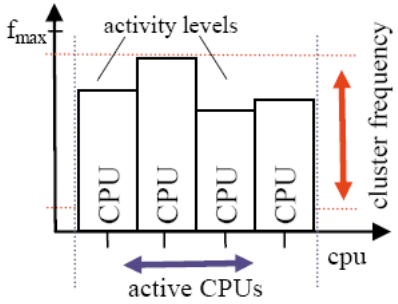
To enhance processing throughput, designers can employ a specialized I/O coherence unit to stream data in and out of the coherent domain. A programmable interrupt controller manages external interrupt distribution into the CPS. The controller is capable of routing up to 256 interrupt sources to CPUs. Designers can program cluster internal events as well as external sources as edge- or level-sensitive with selectable activity level, masked and distributed to cores in

variant severity levels such as masked or non-masked interrupt, or to be used as thread yield qualifier to support thread communication.

Operating systems such as SMP Linux directly support the symmetrical coherent cluster by allocating virtual processors or threads on-demand and migrating processes between cluster resources. Integral parts of those software systems are load balancing and power management features, which communicate to hardware resources via dedicated device drivers. Typically, application inherent performance requirements - e.g. a standardized bit rate; and explicit requirements, such as user choices of a display quality or format, determine an application's overall performance granted by the OS. The OS will adjust hardware thread allocation and processing frequency accordingly.

Adapting performance levels based on temporal demand largely affects energy consumption. Deactivation of individual CPUs scales cluster power linearly. But frequency reduction can impact power exponentially if supply voltage is lowered according to the lower silicon carrier mobility (speed) requirements. In addition to the reduction in dynamic power, which is rooted in CMOS technologies' recharge of parasitic capacitances (P ~ CfV2), lower voltage also reduces the amount of leakage current drawn by nanometer CMOS technologies.

Figure 2 Performance Scaling to Optimize Power



Thread allocation and processing frequency provide two independent variables to adjust performance levels within the cluster, and lay the foundation for power management in the CPS. Therefore, a programmable power controller resides within the CPS to provide operating system access and manage power features of the individual CPS components.

CPU Power Management Modes

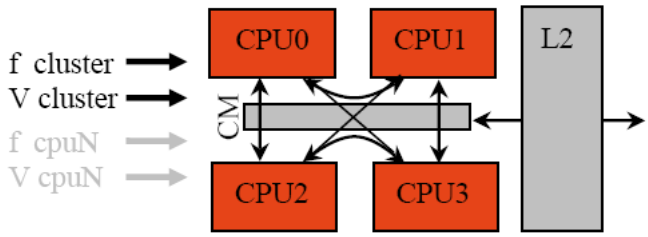
From an operating system perspective, cluster CPUs are characterized by their performance level and processor state. Both are interlinked, but it is advantageous to handle them separately for control purposes. Performance levels can be used to adjust frequency and voltage, whereas the processor state controls CPU activity levels, such as power states and coherence behavior.

Within the coherent cluster, each individual CPU can assume a range of power states which are managed by the cluster power controller (CPC) hardware and proposed by the operating system. These power states reflect different levels of CPU activity and environmental interactions. Designers can directly model operating system processor states towards these hardware-based power modes. A cluster CPU provides four modes of operation – coherent, non-coherent, clock-off and power-down.

- Coherent Mode (Figure 3): This CPU operates as member of an L1 cache coherent domain and exchanges coherence messages with processor peers to maintain cache coherence. The coherence manager posts snoop requests of peer cores towards a particular cache line at the snoop port of the CPU. The global interrupt controller can route interrupts towards this CPU. To sustain optimal performance, communication between this CPU and the CM, and other members of the coherent domain, is maintained at equal speeds. Within coherent mode, designers

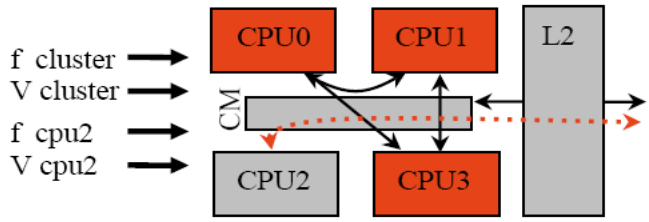
can adjust performance levels through frequency changes to all members of the coherent domain. Cluster-wide voltage adjustments are permitted according to a given cluster frequency. The power controller prevents power-down, and software attempts to deactivate this CPU because of its remaining snoop response duties. Coherent mode can be exited via software, leading to a non-coherent operation mode.

Figure 3 Coherent Mode



- Non-Coherent Mode (Figure 4): In this mode, the CPU operates outside the coherent cluster and does not emit snoop messages towards peer cores. Also, the coherence manager will not forward snoopy messages generated by other coherent CPUs, but continues to process memory requests. Operating as a standalone CPU, the snoop port is deactivated.

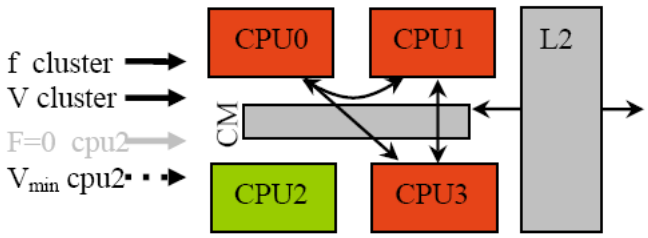
Figure 4 Non-Coherent Mode



Typically, applications running on this CPU using the coherent domain will be brought down or migrate to other CPUs before non-coherent mode is invoked. The global interrupt controller continues to perform interrupt routing towards this core. Since coupling to other CPUs is not required, the operating frequency of a CPU in non-coherent mode is adjusted individually, based on application needs. Supply voltage can accompany frequency changes. Software, interacting with the cluster power controller, can send this CPU into a clock-off or power-down state on operating system demand. Non-coherent mode is particularly well-suited for control plane operations and housekeeping tasks surrounding the activities of a coherent data processing system. Such a CPU can also assume lower power states or even power down. As processing demand rises within the coherent domain, a CPU in non-coherent mode can join the cluster again at any time.

- Clock-Off Mode (Figure 5): A CPU operating outside of the coherent domain can assume clock-off mode. Clock distribution towards this CPU is cut at the clock generator level and sets the CPU and its complete clock tree as inactive. The CPU becomes non-responsive to external events, but maintains its internal state. Therefore, the cluster power controller can perform fast wakeup of the CPU based on system events. After a wake-up, the CPU resumes a non-coherent mode of operation.

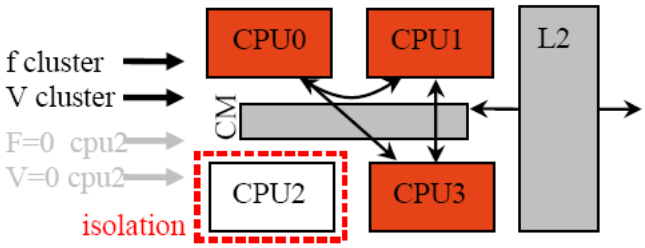
Figure 5 Clock-Off Mode



In clock-off mode, power consumption is reduced to leakage currents and can be further reduced by adjusting the supply voltage. Voltage reduction levels are determined by state retention requirements of standard cell registers and cache RAMs, but also by desired power-up times upon wakeup requests. If no retention or fast wakeup is required, a CPU in clock-off mode can be powered down.

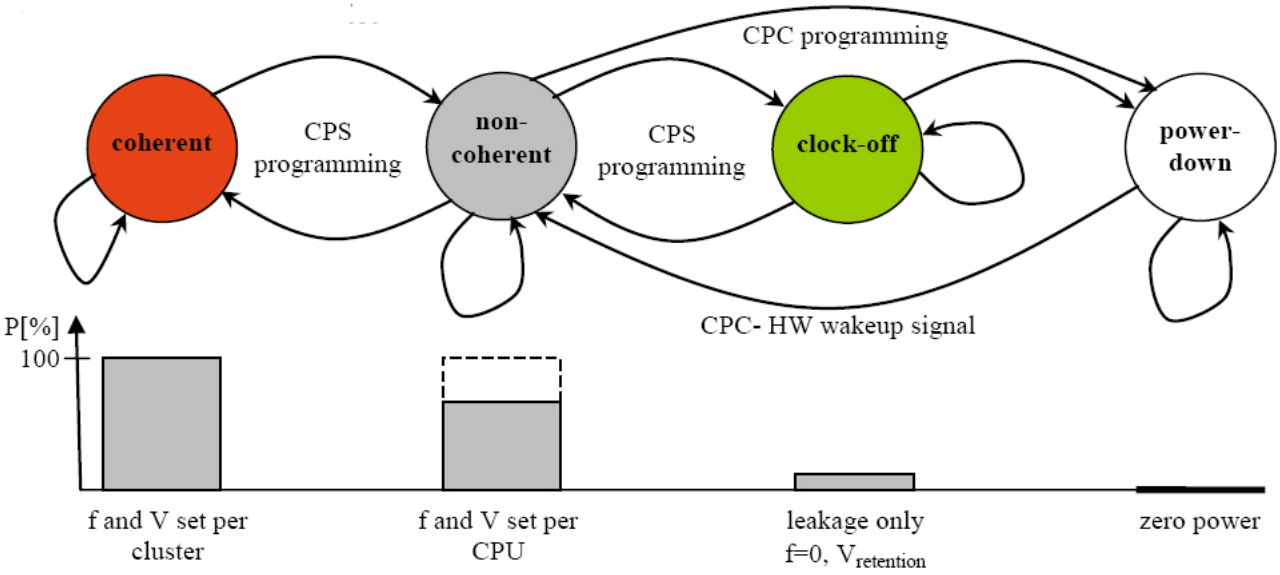
- Power-Down Mode (Figure 6): The CPU is electrically isolated from its surroundings, while other cores in non-power-down mode continue operation. On-chip power gating cells disconnect the supply grid, so it is not necessary to switch external supply. This CPU will lose its internal state and is required to observe a reset and initialization sequence upon power-up. Neither dynamic nor leakage power is consumed. Designers can program the cluster power controller, which schedules a power-down sequence to maintain logical and electrical consistency throughout the system. This includes preventing CPUs in coherent mode from powering down before leaving the coherent domain, and initiating a power-up sequence based on a peer core request or system-level hardware intervention. The cluster power controller also recognizes when all CPUs have reached power-down mode, and initiates a power-down for the now unused coherence manager. Such a situation also enables a shutdown or clock-off mode for higher level memory, such as the L2 cache.

Figure 6 Power-Down Mode



Transitions between CPU modes (Figure 7) are initiated through the operating system talking to coherence manager registers, migrating threads between CPUs and by placing commands into the cluster power controller (CPC) queue. The CPC provides feedback about power states for each component and can notify threads about completion of an initiated sequence. The CPC also provides hardware interlock over-command sequences to ensure power and CPU response integrity at all times. If all CPUs have reached power-down mode, the coherence manager and selectively the level 2 cache are powered down. The CPC however, remains alive to initiate a power-up and reset sequence.

Figure 7 Cluster CPU States



Powering up the CPS consists of several phases. In addition to required initialization and bootstrap operation of an SMP operating system, a staged ramp-up of the system also provides the lowest power footprint and limits inrush current at power-on-reset.

First, CPU0 is powered and a reset sequence is applied. Thereafter, CPU0 executes boot code and initializes the coherence manager. After the operating system reaches multi-tasking run levels, the OS can choose to power up more CPUs. Those CPUs might remain in non-coherent mode to perform tasks within independent address spaces, or they can combine address spaces and join the coherent domain. Other cores might remain isolated and powered-down.

CPU and CPS Performance Levels

As described earlier, an operating system uses CPU performance levels and processor states to meet processing demand and manage power. Each processor state gives rise to performance levels obtainable for this CPU. Obviously, power-down and clock-off modes represent the lowest levels. Operating system performance level requests are directly linked to CPU operating frequencies.

Members of a coherent domain are required to operate at an equal frequency level to optimize snoop latency. However, for certain applications, system architects might trade off the number of coherent threads versus operating frequency to optimize overall energy consumption. The exponential dependency of power to voltage, and therefore operating frequency (P~fV2; f=F(V) => P~V(2+k)), helps determine those tradeoffs. Further, the thread granularity of coherent cluster CPUs (the CPS provides two threads per physical core) can weigh-in towards deciding for more threads at reduced voltage to optimize energy.

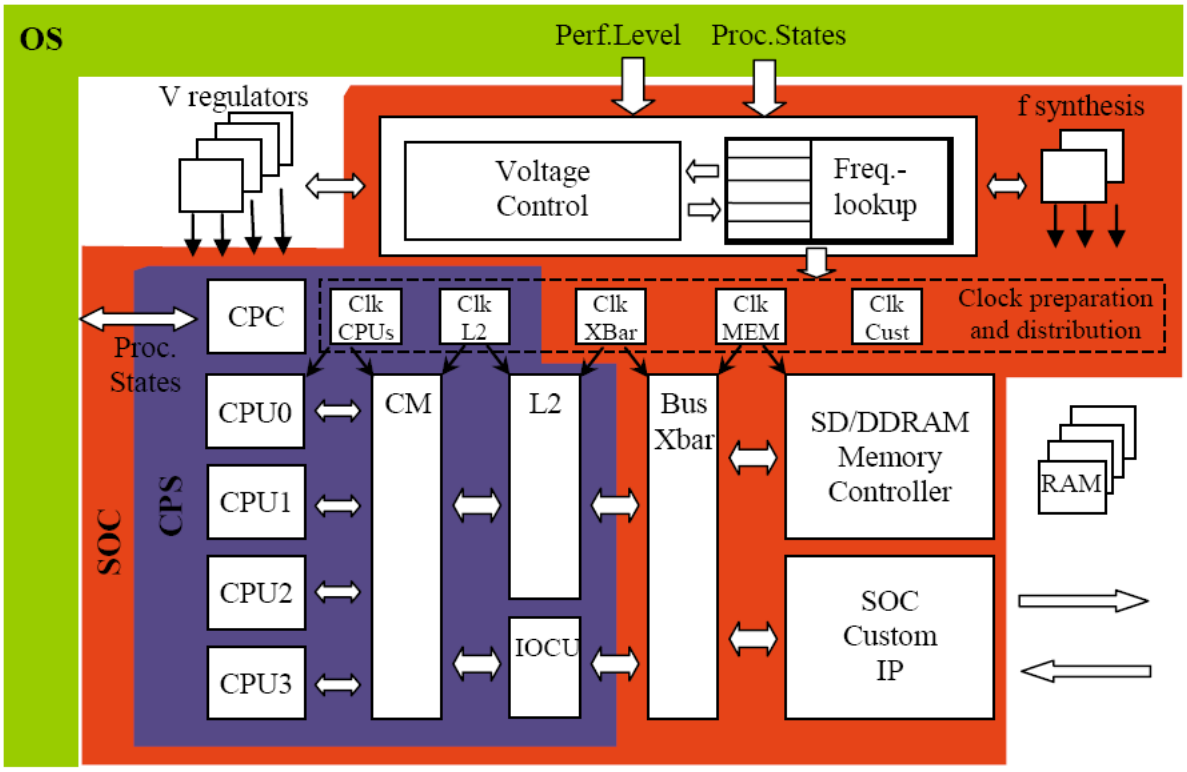
Other aspects of this tradeoff are dynamic changes in processing loads that can be met through spawning of additional hardware threads, or voltage/frequency ramp-up on demand. Designers must consider time constants for both choices for each particular application. To meet performance demands rapidly, one of the two virtual CPUs within a physical CPU core can be kept asleep and prepared in advance to quickly join the coherent domain. This can trigger the powering-up of more CPUs from their power-down state, or raising voltage for a CPU in clock-off state that was held before at a retention supply level.

CPUs in non-coherent mode can be individually controlled in operating frequency relative to the coherence manager and coherent domain CPUs. Designers can adjust operating voltages accordingly.

Reduction in CPU operating frequencies also makes room for power reductions in higher memory hierarchies. Response time requirements for a level two cache towards L1 refill requests become more relaxed. Similarly, bus requests to the SoC could become less demanding. The operating system can evaluate the overall demand on L2 and bus bridge traffic for a given performance level, and reprogram clock generation for those components based on demand.

Figure 8 outlines the propagation of OS performance level requests towards frequency and voltage changes to optimize energy consumption of a SoC. An OS decision for a cluster or single core performance level is translated to frequencies and expected processor states. CPC, voltage control, frequency synthesis and clock distribution are reprogrammed to reach the new performance level. Programmable clock preparation circuits route and divide PLL synthesized frequencies and determine CPU and interface clock speeds. Operating voltages are derived from frequency requests and desired speed of recovery. Therefore, any short-term drops in performance levels do not influence voltage supplies.

Figure 8 OS Performance Level Driven Frequency and Voltage Control in an SOC



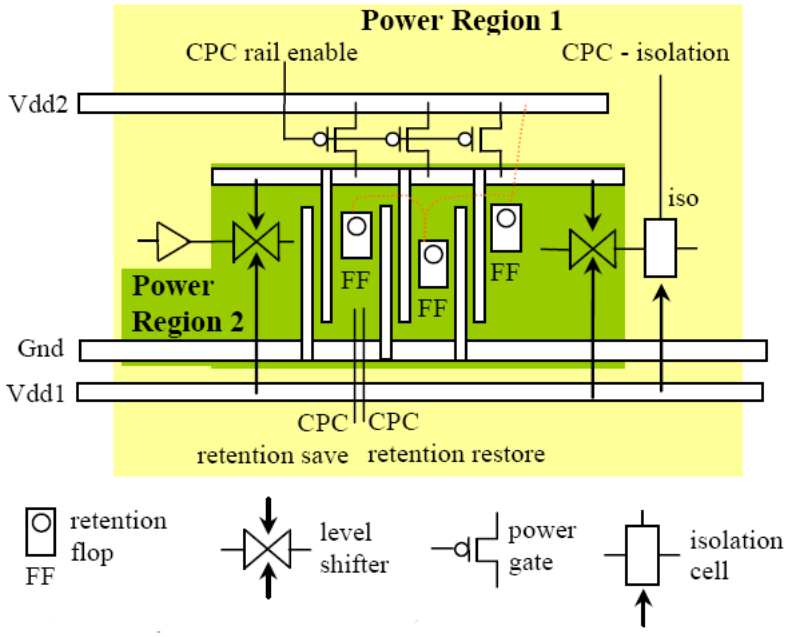
Aspects of Physical Implementation

Changing clocks and the power supply dynamically between SoC subcomponents creates new challenges for SoC integrators. State-of-the-art EDA flows and enhanced standard cell and memory libraries enable the previously discussed power management techniques. At its core are four types of standard cells – power gates, isolation cells, level-shifting cells and retention flops.

Power gate cells are switching CMOS transistors, inserted in supply or ground rails to gate supply voltages for a design region. These are controlled by the cluster power manager (CPC). To guarantee proper logic levels driven into active chip regions, powered-down region outputs connect to isolation cells with predefined isolation values. The CPC invokes isolation of a region before it is powered-down. It remains isolated until stable power supply is established again and the CPC has ascertained proper function for that region.

Regions of different supply voltages need logic level translation to communicate. Level-shifting cells are required. Further, it is possible to maintain the state of flops through retention flops that contain holding circuits of small drive strength, which are supplied separately with relatively high impedance while main supply regions are disconnected.

Figure 9 Power Management Cell Application



Retention techniques can eliminate leakage currents of standard cell regions. However, the application of retention flops increases silicon area slightly. Fast wakeup also requires retention of instruction and data cache contents, which contribute to leakage currents.

EDA flows support the application of these power management techniques through all phases of the design process. Through these flows, power intent for each region is described separately from logic design using standardized formats. Logic simulation drives un-powered regions to high impedance, and mimics the effects of isolation cells towards powered regions. Similarly, logic simulation supports flop state retention behavior. Design synthesis and physical implementation reuse the descriptions of power intent, and insert power related standard cells without user intervention. Formal comparison tools ensure consistency between physical implementation, original power intent and design languages. The CPS describes its power intent, conforming to those power intent formats, and is consistent with higher-level controls such as the CPC to support SoC integrators.

SoC implementers must pay close attention to supply rail dimensioning and power integrity. On-chip switching of rails that supply large design regions—without influencing neighboring active or retention sleep regions—requires voltage drop analysis and must include pad, bonding and package characteristics. Further, application of voltage scaling techniques must reflect board-level supply networks including programmable regulators to properly dimension a system.

Conclusion

Within an embedded multiprocessor cluster like MIPS Technologies' 1004K CPS, power consumption is managed through performance scaling based on temporal performance demand. Power reductions are achieved through dynamic voltage and frequency changes as well as power gating of CPU power islands. Symmetrical processor clusters allow the migration of tasks and threads between physical processor cores. Temporarily unused cores can leave the coherent domain and power down or reduce their voltage to a retention level. Alternatively, cores can operate outside the coherent domain to fulfill operations in non-coherent address space. The operating system manages power through performance level and processor state. These variables are used to drive the physical power management features of the system. A power controller enables the interaction between physical cluster implementation and the operating system.

About the Author

Matthias Knoth is a Design Engineer for MIPS Technologies, Inc., responsible for low-power micro-architecture and 1004K processor implementation. Knoth has more than 13 years' experience in the semiconductor industry with companies including Siemens Central Research, Siemens Microelectronics, Infineon Technologies and Quicksilver Technology. Knoth holds a Masters Degree in Electronics from the University of Technology, Chemnitz, Germany.