A Chip Multithreaded Processor for Network-Facing Workloads

THE ARTICLE DESCRIBES AN EARLY IMPLEMENTATION OF A CHIP MULTITHREADED (CMT) PROCESSOR, SUN MICROSYSTEM'S TECHNOLOGY TO SUPPORT ITS THROUGHPUT COMPUTING STRATEGY. CMT PROCESSORS ARE DESIGNED TO HANDLE WORKLOADS CONSISTING OF MULTIPLE THREADS WITH LIGHT COMPUTATIONAL DEMANDS, TYPICAL OF WEB SERVERS AND OTHER NETWORK-FACING SYSTEMS.

Sanjiv Kapil
Harlan McGhan
Jesse Lawrendra
Sun Microsystems

With its throughput computing strategy, Sun Microsystems seeks to reverse long-standing trends towards increasingly elaborate processor designs by focusing instead on simple, replicated designs that effectively exploit threaded workloads and thus enable radically higher levels of performance scaling.[1]

Throughput computing is based on chip multithreading processor design technology. In CMT technology, maximizing the amount of work accomplished per unit of time or other relevant resource, rather than minimizing the time needed to complete a given task or set of tasks, defines performance. Similarly, a CMT processor might seek to use power more efficiently by clocking at a lower frequency—if this tradeoff lets it generate more work per watt of expended power. By CMT standards, the best processor accomplishes the most work per second of time, per watt of expended power, per square millimeter of die area, and so on (that is, it operates most efficiently).

The processor described in this article is a member of Sun's first generation of CMT processors designed to efficiently execute network-facing workloads. Network-facing systems primarily service network clients and are often grouped together under the label "Web servers." Systems of this type tend to favor compute dense form factors, such as racks and blades.[2] The processor's dual-thread execution capability, compact die size, and minimal power consumption combine to produce high throughput performance per watt, per transistor, and per square millimeter of die area. Given the short design cycle Sun needed to create the processor, the result is a compelling early proof of the value of throughput computing.

Design goals

To satisfy target market requirements, the design team set the following design goals:

- Focus on throughput (number of threads), not single-thread performance.
- Maximize performance-watt ratio.

- Minimize die area.
- Minimize design cost for one- to four-way servers.

Meeting these goals lets system designers pack processors tightly together to build inexpensive boxes or racks offering very high performance per cubic foot. Alternatively, they can use the higher packing densities and lower costs possible with the design to provide more memory or greater functional integration.

Three additional requirements constrained the four basic design goals: minimizing time to market and development cost while maintaining full scalable processor architecture (Sparc) V9 (64-bit) binary compatibility.[3] These constraints dictated our decision to leverage existing UltraSparc designs to the greatest extent possible, while the design goals motivated our unusual decision to combine elements from two different UltraSparc generations.

Overview

To maximize thread density while minimizing both die area and power consumption, we rejected the third-generation 29-million-transistor UltraSparc III processor design[4] in favor of the execution engine from the far smaller and less power-hungry first-generation 5.2-million-transistor UltraSparc I design.[5,6] This choice let us create a processor incorporating two complete four-way superscalar execution engines on a single die, conferring simple dual-thread capability (one thread per core), while keeping both die area and power consumption to a minimum. Achieving even higher thread counts, although suitable for the target market, was inconsistent with the processor's stringent schedule and die size goals.

The decision to use the UltraSparc I execution engine did introduce a complication, however: porting the older design, originally implemented in a mid-1990s 0.5-micron, 3.3-V processor, to contemporary deep-submicron technology. To avoid needless work, the design effort started from the most advanced version of the basic UltraSparc I design available: a 250-nanometer aluminum metal database created in 1998 for the last revision of the UltraSparc II processor (targeted at an upper frequency of 450 MHz). Even so, the problem of translating this design to current 130-nm copper metal technology, targeting a frequency of 1,200 MHz, proved formidable.

Minimizing power consumption was a concern throughout the design, affecting not only the choice of thread execution engine, but also the design of every functional block on the chip. The processor also provides low-power operating modes, allowing software to selectively reduce power consumption during periods of little or no activity by reducing the chip's operating frequency to 1/2 (idle mode) or 1/32 (sleep mode) of normal clock speed.

To further reduce costs (and reduce system-level power consumption by eliminating the need to drive external buses), we replaced the large (4- to 8-Mbyte) external L2 cache typical of UltraSparc II systems with 1 Mbyte of on-chip L2 cache. Moving the L2 cache on chip also substantially improved performance by reducing access latencies; moreover, cache frequency automatically scales up when the processor's clock rate increases.

To achieve compatibility with current-generation systems, we took the design's memory controller and JBus system interface from current UltraSparc IIIi processors, reworking them to interface with the dual UltraSparc II pipelines. Indeed, to maximize leverage and minimize costs, we designed the processor to be pin compatible with UltraSparc IIIi, fitting into the same 959-pin ceramic micro-PGA package used for the existing product. Finally, we implemented a robust set of reliability, availability, and serviceability (RAS) features to ensure high levels of data integrity, and we added new CMT features to control the processor's dual-thread operation. Figure 1 (next page) is a block diagram of the design, showing the major functional units and their interconnections.

Thread execution engine

We lifted the thread execution engine essentially intact from the UltraSparc II processor. The execution engine's design comprises a nine-stage execution pipeline, incorporating four-way superscalar instruction fetch and dispatch to a set of nine functional units: four integer, including load/store and branch; three floating-point (FP), including divide/square root; and two single-instruction, multiple-data (SIMD) operations. The pipeline also

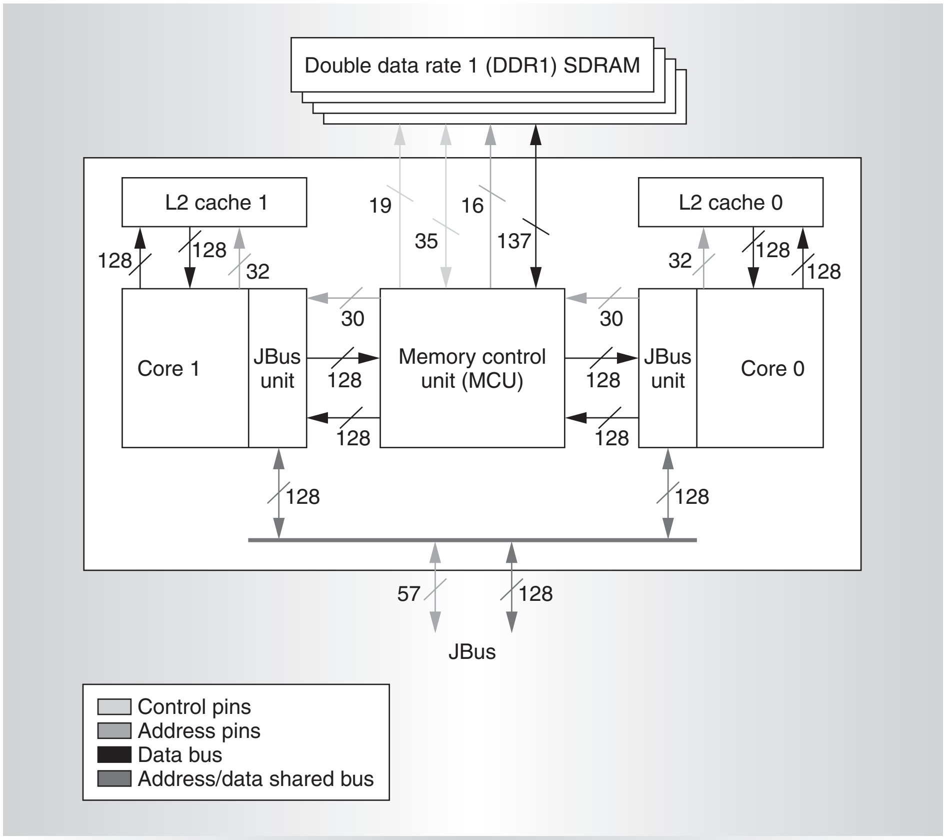


Figure 1. Processor block diagram. We reworked the memory controller and JBus system interface, originally from UltraSparc IIIi processors, to interface with UltraSparc II pipelines.

includes a set of eight overlapping general-purpose register windows, a set of 32 floating-point registers, and two separate 16-Kbyte L1 caches for instructions and data together with associated memory management units, each with a 64-entry translation look-aside buffer (TLB) for caching recently mapped virtual-to-physical address pairs.

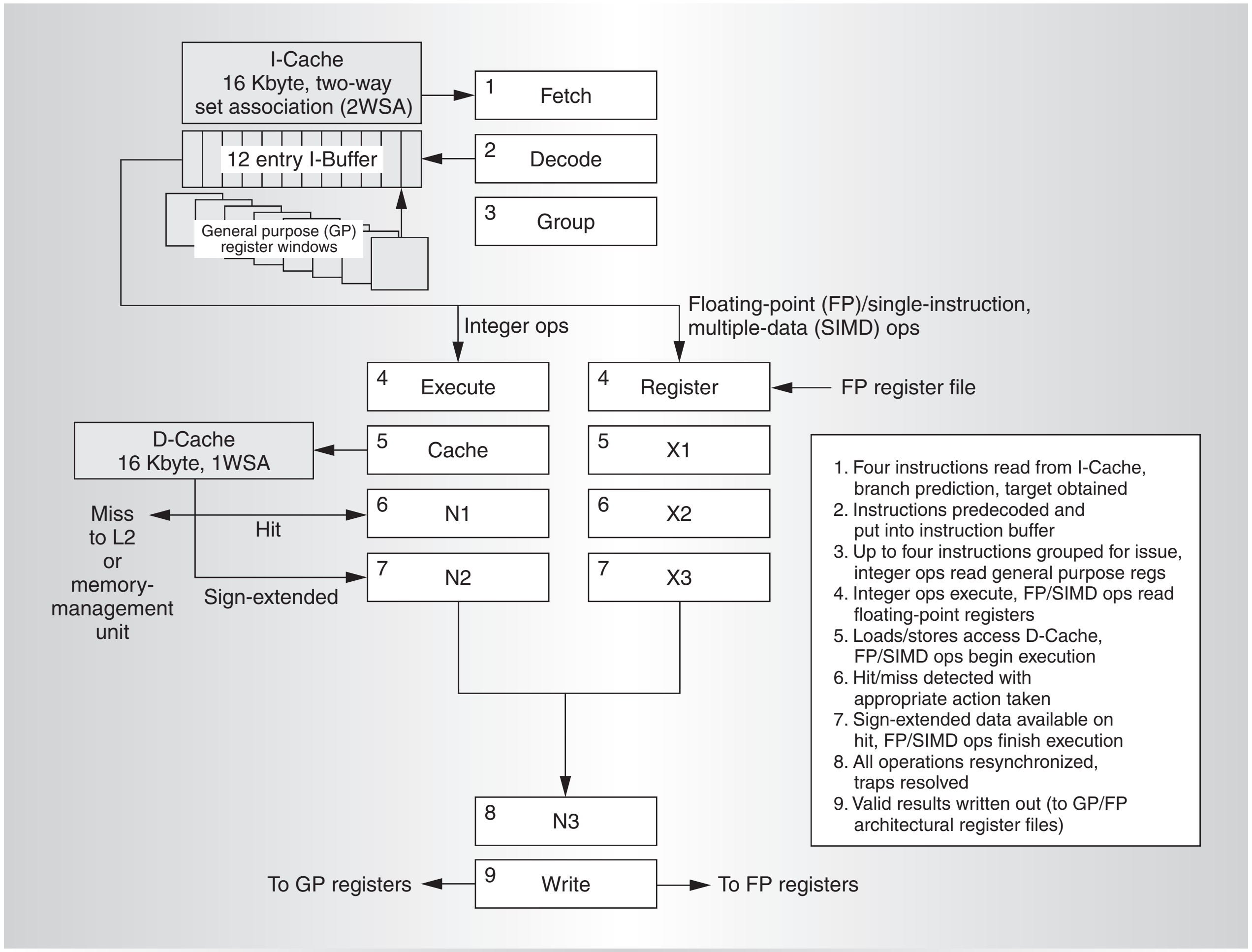
Between its two four-issue pipelines, the processor can fetch and issue eight instructions per clock cycle. A 12-entry instruction buffer decouples instruction fetch from instruction issue to buffer any discrepancies between these two rates. Figure 2 details the UltraSparc II pipeline.

When implemented in 130-nm technology, the 64-bit, superscalar UltraSparc II pipeline design proved remarkably efficient in both area and power. The basic integer unit, including two full arithmetic logic units, required only about 8 mm^2 of area, with the FP and SIMD units, which share execution resources, adding less than 5 mm^2 more. The total core, including the execution units, both L1 caches, the L2 cache controller, the load/store unit, and associated miscellaneous logic, aggregated to less than 29 mm^2. Thus, together the two cores take up only about a quarter of the 206-mm^2 die.

The integer unit dissipates only about 3 watts, with the floating-point unit adding less than another 1.5 watts. Maximum power dissipation by the entire execution engine is less than 7 watts, so between them, the two engines are responsible for less than half the 32 watts maximum dissipated by the processor chip.

On-chip L2 cache

Supporting each of the two UltraSparc II execution engines in the processor is a high-performance, 512-Kbyte, four-way set-associative, on-chip L2 cache. A 128-bit data bus connects each core to the L2. The total load-to-use latency for the L2 is 9 to 10 cycles, with four cycles needed to transit in and out of the cache itself, including error-correcting code



I-Cache 16 Kbyte, two-way set association (2WSA)
12 entry I-Buffer
General purpose (GP) register windows
1 Fetch
2 Decode
3 Group
Integer ops
Floating-point (FP)/single-instruction, multiple-data (SIMD) ops
4 Execute
4 Register
FP register file
D-Cache 16 Kbyte, 1WSA
5 Cache
5 X1
6 N1
6 X2
7 N2
7 X3
Miss to L2 or memory-management unit
Hit
Sign-extended
8 N3
9 Write
To GP registers
To FP registers
1. Four instructions read from I-Cache, branch prediction, target obtained
2. Instructions predecoded and put into instruction buffer
3. Up to four instructions grouped for issue, integer ops read general purpose regs
4. Integer ops execute, FP/SIMD ops read floating-point registers
5. Loads/stores access D-Cache, FP/SIMD ops begin execution
6. Hit/miss detected with appropriate action taken
7. Sign-extended data available on hit, FP/SIMD ops finish execution
8. All operations resynchronized, traps resolved
9. Valid results written out (to GP/FP architectural register files)

Figure 2. UltraSparc II nine-stage execution pipeline. Numbers in the diagram correspond to stages listed.

(ECC) checking. Maximum throughput is one operation every two clock cycles, providing a peak bandwidth of 9.6 gigabits per second at 1.2 GHz.

In short, the L2 cache megacell has a two-cycle throughput with a four-cycle latency. The major implementation challenge for the L2 cache blocks was achieving the short latency and high frequency timing goal (with ECC and other features) within the constraints of a short design cycle. To meet this challenge, we designed a new physical L2 cache unit for the processor, using a semicustom approach from basic elements to top-level integration. We designed the SRAM arrays with a conventional custom design approach and implemented all other logic as standard control and data path flows. To reduce the blocks' design time and maximize array efficiency, we added minimal logic to the SRAM arrays (such as comparators, XOR stages for ECC, and parity). A semicustom routing methodology optimized timing and minimized coupling noise, while making it easier to iterate the design to accommodate changing functional feature sets.

We used two main techniques to achieve the design's aggressive timing goals. First, to equalize pipeline stage length, we optimized the placement of each control and data path block to balance its wire and gate delays with the data and tag memories, according to the critical signals' timing requirements. Second, within the SRAM control and data path blocks, we used two pipelined cycles to implement access. The two-cycle pipelined design not only let the blocks operate at half the processor clock frequency, it also eliminated the typical clock skew overhead and setup time penalties and furthered the overall power

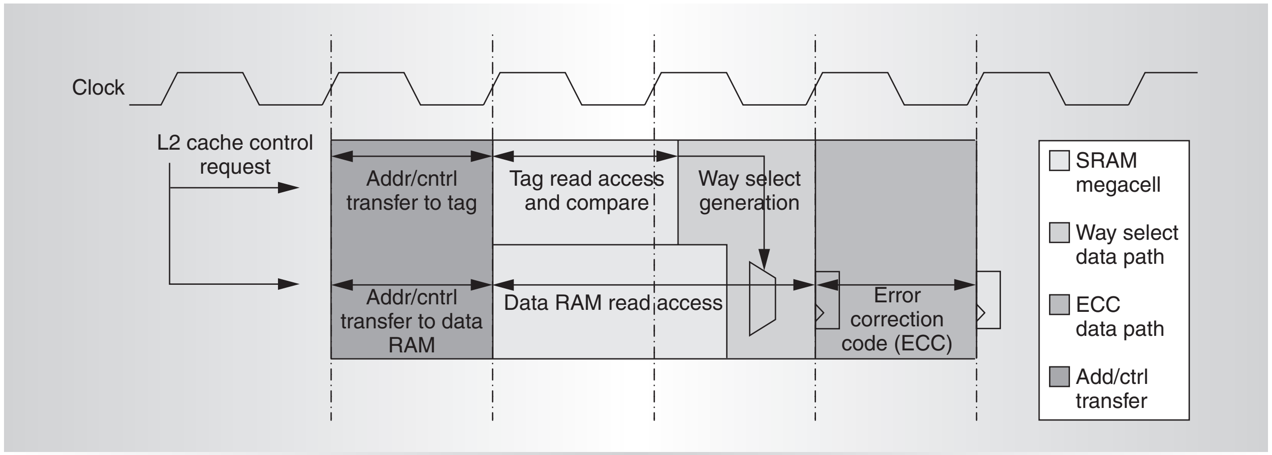


Figure 3. L2 cache pipeline design. Using the two-cycle design, the cache memory operates at half the processor clock frequency.

minimization goal. Figure 3 diagrams the L2 cache pipeline.

Memory interface

Two cores share the memory control unit integrated on the processor die. The MCU provides direct connectivity between processor chip and main memory through a 128-bit, two-channel, double data rate 1 (DDR1), 266-MHz SDRAM interface that delivers a peak bandwidth of 4.2 Gbps. The controller lets the processor access memories ranging in size from 256 Mbytes to 16 Gbytes. In normal operating modes, the controller can conduct up to three concurrent (pipelined) memory operations at the DDR interface and can queue 16 read and 10 write requests.

An early-access mechanism reduces memory latency by starting reads without waiting for them to become visible to other processors in multiprocessor configurations. If the read request is filled from a dirty line in another processor's L2 cache, then the requesting processor ignores the response from memory. To improve performance, pending reads are given priority over pending writes after read-after-write/write-after-read hazard detection. Selecting the bank select feature's XOR permutation along with supported memory interleave modes avoids potential bank conflicts between outstanding reads and associated writes (evictions) that might occur from the same index of the L2 cache, thus improving performance. The controller can handle transitions to 1/2 and 1/32 of normal operating frequency, tracking outstanding transactions to the DDR interface even when the processor operating frequency drops below the DRAM frequency.

A 266-MHz internal memory clock, provided by a separate on-chip phase-locked loop, controls memory interface clocking. A divide-by-two circuit generates the 133-MHz external memory clock and the 90-degree phase-shifted strobes as required by the DDR1 specification for memory writes. We synchronize the processor- and memory-clock domains with an edge-align detector circuit.[7]

During a memory read transaction, the MCU broadcasts read requests to the memory modules, which return 35 strobe signals. Strobe signals have the same frequency as the memory clock, but because of skew, the strobes have an unknown phase relationship both to each other and to the clock. Each strobe feeds a four-stage, asynchronous, flow-through, first-in, first-out (FIFO) buffer to absorb the skew and release the data synchronously to the processor clock domain. The interface stage at the FIFO's input side converts the strobe's rising and falling transitions into a pulse matching the FIFO's speed. A similar circuit at the output side removes a data item from the FIFO on each rising edge of the processor clock. Inside the FIFO, data shifts through the queue in a self-timed manner, independent of memory or processor clock. Because the FIFO's input and output interfaces are naturally decoupled, each can operate in a different clock domain. The FIFO circuit is simple, small, and has low fall-through latency.

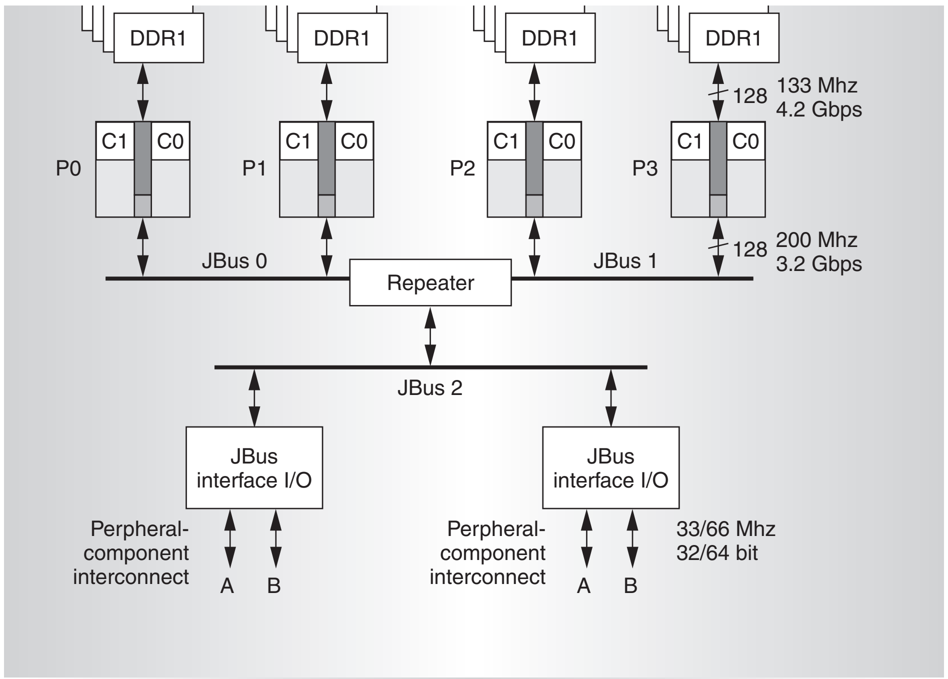


Figure 4. Four-way (eight-thread) multiprocessor system. The repeater divides the JBus into shorter segments to maximize bus operating frequency.

JBus system interface

The processor supports the JBus system interface, which combines high performance with the low cost demanded by Web and blade servers. JBus is a packet-switched, multiplexed data and address, 128-bit system interface with a low pin count (171 dynamic termination logic signals). It operates at a peak rate of 200 MHz, providing up to 3.2 Gbps of bandwidth to support cacheable and noncacheable intraprocessor, interprocessor, and I/O traffic. In addition to bus management, the JBus unit (JBU) helps enable key CMT management functions and is responsible for transitioning the processor into its low-power modes by dynamically reducing the processor's internal clock frequency.

The bus interface specifies a snoop-based, MOESI (modified owner-exclusive shared invalid),[8] write-invalidate, cache coherency protocol for a writeback L2 cache. It also specifies a distributed shared-memory protocol, allowing systems based on the design to scale from a single processor (two threads) up to a four-way (eight-thread) multiprocessor configuration. The bus supports seven ports that engage in distributed round-robin arbitration to determine which one gains access to the bus. Separate control signals broadcast bus requests and snoop results. A companion JBus interface I/O (JIO) chip supports one- to four-way multiprocessor configurations, as well as two 32/64-bit, 33/66-MHz peripheral-component interconnect (PCI) buses for I/O traffic. Optionally, we can configure one PCI interface as a 64-bit Sun UPA64s local graphics bus. Figure 4 shows a four-processor system configured with a repeater to electrically segment the JBus (limiting bus loading to achieve 200-MHz operation) and two JIO chips.

The processor automatically configures the dynamic termination logic (DTL) I/O at system boot time as either 50Ω or 25Ω, depending on system topology. In DTL receive mode, drivers have switchable, active 50Ω pull-up or high-impedance, tristate termination. With on-chip termination, the main requirement is to maintain nearly constant termination resistance over process, supply voltage, and temperature (PVT) variations, as well as output signal variations. To meet this requirement, each driver output stage consists of a set of various-size transistors connected in parallel. Special impedence-matching circuitry in the processor automatically seeks the appropriate dc output impedance for the prevailing PVT conditions by comparing a reference driver's output resistance with that of an external precision resistor, enabling and disabling reference driver transistors until the output resistance falls within a desired margin of the target impedance. Additional circuitry augments the primary pull-up and pull-down devices to improve the output switching current's linearity over the voltage range of operation.

Data reliability

To achieve high data reliability, the processor implements a more extensive set of error detection and correction mechanisms than UltraSparc II, which performed only parity error checking on the external L2 cache and limited error checking on the external system memory paths. The design expands error checking to all large RAM arrays within the chip itself, and additional features prevent detected errors from propagating silently. Specifically, whereas the UltraSparc II implemented 33 error-reporting bits and four error addresses, the design supports 59 error-reporting bits and 24 error addresses. The extra bits allow software to be aware of errors on internal RAM arrays and to detect possible side effects. Thus, the processor logs all corrected single-bit errors as well as the error mode (for example, whether an error in local DRAM resulted from a local access or a remote access by a foreign processor).

In general, the processor achieves reliability by detecting errors as early as possible and reporting errors as soon as it detects them. It provides software with enough information to identify the errors' causes and addresses so that error handlers can take the appropriate actions, and it clearly marks uncorrectable errors as bad data so that other threads don't unwittingly use them.

Parity checking protects both L1 caches as well as the L2 cache tags and JBus transactions. Full ECC checksums protect the data in L2 cache and memory, enabling correction of all single-bit errors and detection of any double-bit errors. On copy-out (cache-to-cache transfer) and write-back (cache-to-DRAM transfer), the design stores any data containing an uncorrectable multibit error with an invalid ECC, preventing it from propagating to other threads. Other errors the design reports include faulty system events, correctable single-bit and uncorrectable multibit memory errors, and attempts to read from or write to out-of-range memory addresses.

CMT features

Sun's specification for CMT processors provides these designs with a standard interface that lets software manage multiple threads on a die consistently and uniformly. Because the processor's dual-thread design uses independent dual cores, each supporting a single execution thread, with cache coherency enforced by the JBU, its key CMT feature is the software's ability to dynamically park or unpark and disable or re-enable either core.

The core-parking feature stops instruction execution on the core but retains cached data and instructions, leaving the core's L2 cache fully responsive to external snoop requests. In normal operation, this feature provides the opportunities to reduce bus contention at boot time—only one core is enabled, speeding the process by eliminating the need to share access to the slower-booting ROM bus—and to conserve power under light processing loads.

The core-disable feature not only stops a core from executing instructions, but also eliminates its caches from the cache coherency domain. Core disable is primarily of value as a design-for-test feature during chip debugging, but it can also potentially salvage some dies with one defective core as single-thread processors. Other CMT features include the ability to steer errors and to perform externally initiated resets to either or both cores as appropriate.

To implement the new CMT features, we defined several new CMT registers, some specific to the individual cores and some shared by all cores. The shared registers manage the global functions needed by the processor, such as core disable, core parking, and error steering. Hardware imposes no restrictions on individual cores' read or write accesses to these common registers, so any core can fully manage the processor's thread configuration. It is, therefore, the software's responsibility to arbitrate access to these registers, avoiding potential write or read conflicts.

The CMT registers included in each core provide necessary core-specific functionality—for example, the core's ID, interrupt IDs, and error reports. We also added eight scratchpad registers to each core, giving software a place to hold commonly used variables for the thread executing on that core (although, strictly speaking, the scratch-pad registers are not required for CMT).

Most of these new registers can be written and read through the Joint Test Action Group (JTAG) interface, allowing service processors to configure each system processor during reset.

Physical implementation challenges

Texas Instruments manufactures the processor in TI's 130-nm, seven-metal-layer copper CMOS process technology. Figure 5 is a photograph of the die and package. To achieve the goal of porting the legacy UltraSparc II blocks, originally laid out in 1994 to run at 167 MHz in a 0.5-micron process, to a deep-submicron technology with a target frequency of 1.2 GHz, we addressed a variety of issues: new layout design rules, coupling noise, electromigration, poly resistance, negative bias temperature instability (NBTI), channel leakage, and intradie process variation.

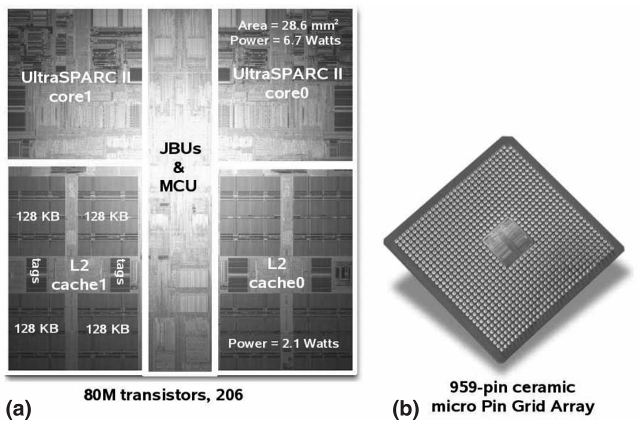


Figure 5. The processor includes an 80-million transistor, 206 mm^2 die (a) and a 956-pin ceramic micro pin grid array (b).

Design rule changes

Foundries set design rules to address manufacturability and yield issues. When a foundry moves to a new generation of process technology, its design rules must also change to reflect changes in the characteristic electrical properties and behaviors of circuits implemented in the smaller geometries. The design team's task was to quickly and effectively convert circuits created in compliance with design rules valid for an earlier generation of process technology into circuits compliant with rules valid for a later generation. In all, we had to contend with 36 new layout design rules related to isolated vias, minimum areas and holes, wide metals, and substrate taps.

Manually fixing the large number of violations while meeting the aggressive schedule goals was impossible. So we developed automated flows to accelerate generation of the modifications needed to eliminate violations of the new layout rules.

Increased coupling noise

Another early concern about the aggressive design shrink was increased coupling noise—the tighter pitches required by smaller geometries typically result in increased capacitance between parallel wires. However, the use of new low-*k* dielectric materials in and between wire layers greatly reduced capacitance. To address top-level route noise issues, we rerouted the core using a minimum of 1.5 times wire pitch at double spacing. This half shielded all critical signals and fully shielded clock signals. We used the same approach in routing the top level for both the previously designed UltraSparc II core and the newly designed L2 cache.

Electromigration

Electromigration in vias presented a triple threat, given the smaller geometries, the replacement of the more EM-resistant tungsten with copper, and the design's higher target frequency. As a result of these changes, the vias in the design are smaller, more prone to erosion, and under greater stress than the vias in the original UltraSparc II design. To meet the threat, we created new footprint-compatible EM-resistant cells that could reliably sustain the highest current loads in the processor. We addressed EM for the top-level route by using double vias to minimize the need for fixes at the last stage.

Poly resistance

A problem not anticipated at the project's outset was the impact of poly resistance on the chip's frequency. The 0.13 micron process technology applies a much higher poly gate bias to the mask than did earlier generations of technology, while the poly sheet resistance also increases. This affected gate delay, with an overall impact on the top-level timing paths of roughly 10 percent. To strap the poly gates in question, we established a binning process to identify the worst cases among the most commonly used cells.

NBTI

NBTI is the aging effect that causes *p*-channel metal-oxide-semiconductor (PMOS) drive current to decrease over the silicon's lifetime. To better withstand NBTI, we reduced the role of PMOS transistors by enhancing many circuits' margins and redesigning one of the circuits most affected by NBTI—the current-mode latch sense amps used for caches and TLBs. We also increased self-time margins on memories, analyzed flip-flops for writeability, and set up new guidelines to account specifically for NBTI effects.

Channel leakage

Transistor channel leakage worsens as geometries shrink, requiring modification of many circuits to enhance leakage tolerance. In the original UltraSparc II library, designers implemented the scan slave latch as a dynamic latch without a keeper device. Although this was a feasible way to reduce flip-flop size in previous generations, it's not acceptable for deep-submicron processes. For this design, we made all flip-flops fully static. We rigorously checked keeper size ratio for dynamic domino gates and skewed CMOS gates, using keeper ratio guidelines that we defined to guarantee proper functionality even at the worst-case burn-in corner with noise. The improved transistor design rules and intense layout efforts let us do this without changing the original footprints.

Intradie process variation

Intradie process variation created challenges for chip clocking. Because one of the design's primary goals was low power, we selected a balanced-tree clock distribution. However, process variation across the die area impacted clock skew, which, in turn, made meeting hold time requirements a major challenge. Applying a single worst-case skew to the full timing run resulted in an unreasonably high number of hold time violations. The design challenge here was twofold:

- filtering out false violations by applying the real clock skew for each path, and
- minimizing the engineering effort needed to fix the real violations.

To accomplish the first task, we performed hold time analysis using the worst-case skew. We then postprocessed the results, analyzing source and destination flip-flops on each violating path and adjusting clock skew to use actual rather than worst-case skew. Applying real clock data reduced the number of hold time violations by about 80 percent.

For the second goal, we developed a flow that swaps flip-flops and buffers with corresponding versions that have extra delay or a relaxed hold time constraint, allowing for setup-time slack. We developed another flow that identifies the right place to insert buffers to solve hold time violations without creating a critical path.

Design validation

In developing a comprehensive test suite for the processor, we achieved significant leverage from a large body of existing tests spanning three UltraSparc generations. Because the design is based on the UltraSparc II pipeline, we started with existing regression tests, modifying them as needed to account for new blocks. Thus, for example, because the execution pipeline was largely unchanged, tests developed for the UltraSparc II pipeline required virtually no modification.

Another productive leverage point was the UltraSparc IIIi test suite. Not only were the JBU and MCU derived from the UltraSparc IIIi design, but the extensive compatibility between the UltraSparc II and UltraSparc III instruction sets made it relatively simple to port many of the UltraSparc IIIi tests for these blocks. The primary changes needed were for different privileged registers, timing differences, and the use of bus agent numbers.

We also ported CMT tests developed for the data-facing UltraSparc IV processor, another dual-thread CMT design. Because all Sun CMT processors adhere to a common specification that defines CMT functions, registers, and interfaces, CMT tests developed for UltraSparc IV were directly applicable to the design's CMT functions and registers.

In theory, handwritten tests can completely exercise a design. In practice, however, creating a truly comprehensive set of directed tests becomes more difficult as the corner cases become deeper and subtler with increasing design complexity. To reduce the time needed to generate tests that hit these corner cases, we used random test generators. Once again, we were able to leverage prior work. With minimal changes, we adapted five in-house random code generators to the task of exercising the processor. Using automated scripts, we simulated tens of billions of cycles prior to tape-out, catching many bugs overlooked by the directed test suites.

Other testing techniques included the development of a stand-alone test bench to exercise the memory subsystem, as well as formal verification techniques. To make sure all configuration options (such as clock ratios, dual inline memory module types, and disabled caches) worked correctly, we ran a subset of the directed tests with different combinations of options. We also ran the random tests with many different options. This approach uncovered bugs that had escaped other detection methods.

Microprocessor designers must provide test vectors to the fab for factory testing of manufactured dies to ensure their basic functionality before they are packaged. For a dual-thread design, the simplistic approach to testing both cores (namely, running each test on each core in sequence), would require about twice as much tester time as a single-thread design. To alleviate this problem, we wrote a tool that automatically converts single-thread tests into tests that run on both cores in parallel. Because most single-thread tests do not stress bus bandwidth, the parallelized test runs in much less than twice the time of the single-thread test.

Because the processor is pin-compatible with the UltraSparc IIIi processor, it was easy to boot the Solaris operating system within a few hours of receiving the first processors, using an existing UltraSparc IIIi system (changing only the power supply voltages and system software). We then applied a suite of test applications to stress the processor, taking advantage of the chip's faster cycle time (compared to simulation). Four of the five random test generators could also be used on a real system, and we ran them extensively. When we encountered a failure, we isolated it and then reproduced it in simulation for designers to analyze. In many cases, we took advantage of logic analyzers already customized for the JBus protocol, which greatly accelerated debugging.

Table 1. Comparison of processors based on UltraSparc II design.

Features	IIe	IIi	Dual thread
First silicon (year)	1999	2001	2002
Technology	180 nm, Al	180 nm, Cu	130 nm, Cu
Frequency (MHz)	500	650	1,200
Watts (maximum)	13	17.6	32*
Transistors (millions)	23	38	80
Area (sq. mm.)	118	159	206
Memory bandwidth (peak Mbps)	800	1,040	4,200
SPECint 2000	174	246	493 est.
SPECfp 2000	182	276	600 est.
SPECint_rate2000	2.02	2.85	11.00 est.
SPECfp_rate2000	2.11	3.2	12.20 est.
SPECint_rate2000 per watt	0.16	0.16	0.34 est.
SPECfp_rate2000 per watt	0.16	0.18	0.38 est.

*Power consumption under maximum stress on a tester. In a system running application code, typical power consumption rarely exceeds 24 watts and can be reduced in sleep mode to less than 5 watts. Clocking the chip at 900 MHz also dramatically affects power consumption, reducing it to 14 watts, less than half the 32 watts consumed at 1,200 MHz (because the chip can use a lower supply voltage).

Performance

Table 1 compares the overall performance scores of the new dual-thread design with previous derivatives of the UltraSparc II design. Scores for the dual-thread processor are preliminary estimates based on measured results with 16 Gbytes of memory and both cores active. Scores for the UltraSparc IIe and IIi processors are official and are posted on the SPEC Web site (http://www.spec.org).

The fact that the dual-thread processor delivers roughly four to six times more throughput per unit of time than the earlier UltraSparc II designs is primarily due to its higher frequency, dual-thread design, and greater bandwidth. But not only the absolute throughput scores are higher. This design delivers throughput performance more efficiently than earlier designs. Specifically, it offers about twice as much performance per watt of expended power, 60 to 80 percent more performance per transistor, and about three times the performance per square millimeter of silicon. It accomplishes this efficiency despite the fact that it must run at higher power levels to reach its higher operating frequency. It also incorporates many more transistors and requires a larger die. With the dual-thread design, however, as with all CMT designs, better performance is not simply a matter of having more resources to apply, but rather of applying available resources more efficiently than earlier (non-CMT) designs.

Overall, the project results proved gratifying. By leveraging existing designs, tests, and infrastructure, we went from project inception to working silicon for a highly integrated 80-million-transistor processor in about two years. More important, this dual-thread design represents a first step on Sun's throughput computing roadmap to the more highly threaded CMT processors possible in the coming 90- and 65-nm generations of process technology. By demonstrating significant gains in the efficiency with which transistors, chip area, and expended power can be converted into throughput performance, the processor also validates the basic notions behind Sun's CMT initiative. An up-to-date list of articles on Sun's throughput computing initiative is available at http://www.sun.com/processors/pressreleases. MICRO

Acknowledgments

This article, like the project it describes, was a collaborative effort. Other members of the project team who made significant contributions to its writing are Dina Bistry, Gaurav Garg, Mohammed A. Jolarpet, Ana Sonia Leon, Avi Liebermensch, Subir Mukherjee, Bruce Petrick, Jin-uk Luke Shin, Jae C. Son, Jeffrey Su, Ken Swanson, Toshinari Takayanagi, Muthukumar Vairavan, and Dan Vo.

References

1. G. Haff, "Sun: Better Computing Through Threads," *Illuminata*, 9 July 2003; also available at http://www.sun.com/processors/pressreleases.
2. S. Kapil, "Gemini: A Power-Efficient Chip-MultiThreaded (CMT) UltraSparc Processor," *Proc. 15th Hot Chips Symp.*, IEEE CS, 2003; http://www.hotchips.org/archive/index.html.
3. D. Weaver and T. Germond, *The Sparc Architecture Manual*, version 9, Prentice Hall, 1994.
4. T. Horel and G. Lauterbach, "UltraSparc-III: Designing Third-Generation 64-Bit Performance," *IEEE Micro*, vol. 19, no. 3, May-June 1999, pp. 73-85.
5. M. Tremblay and J.M. O'Connor, "UltraSparc I: A Four-Issue Processor Supporting Multimedia," *IEEE Micro*, vol. 16, no. 2, Mar.-Apr. 1996, pp. 42-50.
6. L.A. Lev et al., "A 64b Microprocessor with Multimedia Support," *IEEE J. Solid State Circuits*, vol. 30, no. 11, Nov. 1995, pp. 1227-1238.
7. K. Georgios et al., "Implementation of a Third Generation 1.1 GHz 64-bit Microprocessor," *IEEE J. Solid-State Circuits*, vol. 37, no.11, Nov. 2002, pp. 1464-1466.
8. D.A. Patterson and J.L. Hennessy, *Computer Architecture: A Quantitative Approach*, 3rd ed., Morgan Kaufmann, 2002.

Sanjiv Kapil is the lead architect for the processor design. His research interests include modeling microprocessor performance and high-speed interconnect architectures. He has an MTech in computer science from the Indian Institute of Technology, New Delhi.

Harlan McGhan is the strategic marketing manager for UltraSparc processors. His research interests include the history of computers and trends in microprocessor design. He has a BA in philosophy from Michigan State University and is pursuing a PhD in the history and logic of science from Princeton University.

Jesse Lawrendra is the senior engineering manager for the processor's front-end design. His research interests include design reusability and effective approaches to building a system on a chip integrating multiple complex IPs. He has an MS in electrical engineering from the University of California, Davis, and an MBA from UC Berkeley.

Direct questions and comments about this article to Sanjiv Kapil, Sun Microsystems, 430 N. Mary Ave., Sunnyvale, CA 94085-4199; Sanjiv.Kapil@Sun.com.