Implementation of an 8-Core, 64-Thread, Power-Efficient SPARC Server on a Chip

Umesh Gajanan Nawathe, Mahmudul Hassan, King C. Yen, Ashok Kumar, Aparna Ramachandran, and David Greenhill

***Abstract*—The second in the Niagara series of processors (Niagara2) from Sun Microsystems is based on the power-efficient chip multi-threading (CMT) architecture optimized for Space, Watts (Power), and Performance (SWaP) [SWap Rating $=$ Performance$/$(Space $*$ Power)]. It doubles the throughput performance and performance/watt, and provides $>10\times$ improvement in floating point throughput performance as compared to UltraSPARC T1 (Niagara1). There are two 10 Gb Ethernet ports on chip. Niagara2 has eight SPARC cores, each supporting concurrent execution of eight threads for 64 threads total. Each SPARC core has a Floating Point and Graphics unit and an advanced Cryptographic unit which provides high enough bandwidth to run the two 10 Gb Ethernet ports encrypted at wire speeds. There is a 4 MB Level2 cache on chip. Each of the four on-chip memory controllers controls two FBDIMM channels. Niagara2 has 503 million transistors on a 342 mm^2 die packaged in a flip-chip glass ceramic package with 1831 pins. The chip is built in Texas Instruments' 65 nm 11LM triple-Vt CMOS process. It operates at 1.4 GHz at 1.1 V and consumes 84 W.**

***Index Terms*—Chip multi-threading (CMT), clocking, computer architecture, cryptography, low power, microprocessor, multi-core, multi-threaded, Niagara series of processors, power efficient, power management, SerDes, SPARC architecture, synchronous and asynchronous clock domains, system on a chip (SoC), throughput computing, UltraSPARC T2.**

I. Introduction

TODAY'S datacenters face extreme throughput, space, and power challenges. Throughput demands continue increasing while space and power are fixed. The increase in power consumed by the servers and the cost of cooling has caused a rapid increase in the cost of operating a datacenter. The Niagara1 processor [5] (also known as the UltraSPARC T1) made a substantial attempt at solving this problem. This paper describes the implementation of the Niagara2 processor, designed with a wide range of applications in mind, including database, web-tear, floating-point, and secure applications. Niagara2, as the name suggests, is the follow-on to the Niagara1 processor based on the CMT architecture optimized for SWaP.

Fig. 1 illustrates the advantages of the CMT architecture. For a single thread, memory access is the single biggest bottleneck to improving performance. For workloads which exhibit poor memory locality, only a modest throughput speedup is possible by reducing compute time. As a result, conventional single-thread processors which are optimized for Instruction-Level-Parallelism have low utilization and wasted power. Having many threads makes it easier to find something useful to execute every cycle. As a result, processor utilization is higher and significant throughput speedups are achievable.

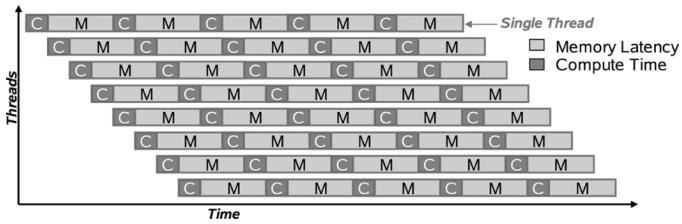


Fig. 1. Throughput computing using the CMT architecture.

The design of the Niagara2 processor started off with three primary goals in mind: 1) $>2\times$ throughput performance and performance/watt as compared to UltraSPARC T1; 2) $>10\times$ floating point throughput performance as compared to UltraSPARC T1; and 3) integration of the major system components on chip. Two options were considered to achieve these goals: 1) double the number of cores to 16 as compared to eight on UltraSPARC T1, with each core supporting four threads as in UltraSPARC T1; and 2) double the number of threads/core from four to eight and correspondingly double the number of execution units per core from one to two. Both options would have enabled us to achieve our first goal. The first option would have doubled the SPARC core area as compared to a lot smaller area increase with the second option. The second option was chosen as the area saved using this option allowed us to integrate a Floating Point and Graphics unit and a Cryptographic unit inside each SPARC core and also allowed integration of the critical SoC components on chip, thus enabling us to achieve our second and third goals as well.

II. Architecture and Key Statistical Highlights

A. Niagara2 Architecture

Fig. 2 shows the Niagara2 block diagram, and Fig. 3 shows the die micrograph. The chip has eight SPARC Cores, a 4 MB shared Level2 cache, and supports concurrent execution of 64 threads. The Level2 cache is divided into eight banks of 512 kB each. The SPARC Cores communicate with the Level2 cache banks through a high bandwidth crossbar. Niagara2 has a $\times 8$ PCI-Express channel, two 10 Gb Ethernet ports with XAUI interfaces and four memory controllers each controlling

Manuscript received April 17, 2007; revised September 27, 2007.

U. G. Nawathe is with Sun Microsystems, Santa Clara, CA 95054 USA (e-mail: umesh.nawathe@sun.com).

M. Hassan, K. C. Yen, A. Kumar, A. Ramachandran, and D. Greenhill are with Sun Microsystems, Sunnyvale, CA 94085 USA.

Digital Object Identifier 10.1109/JSSC.2007.910967

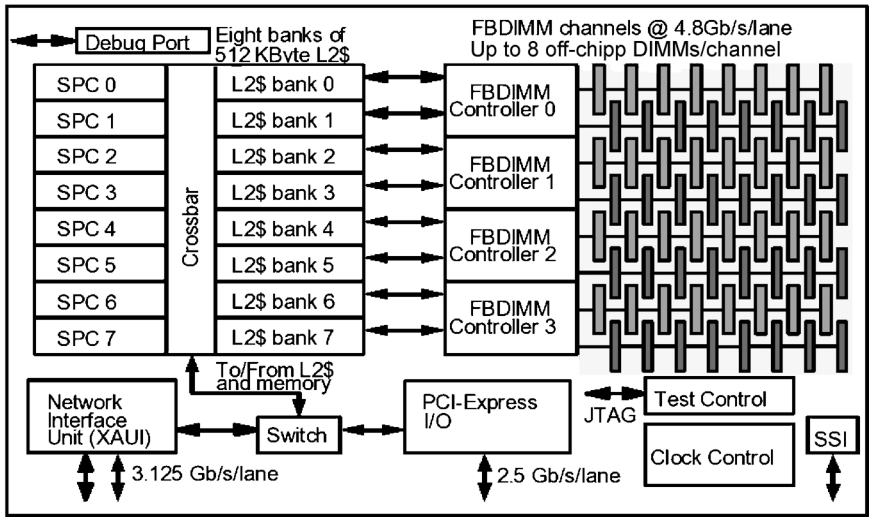


Fig. 2. Niagara2 block diagram.

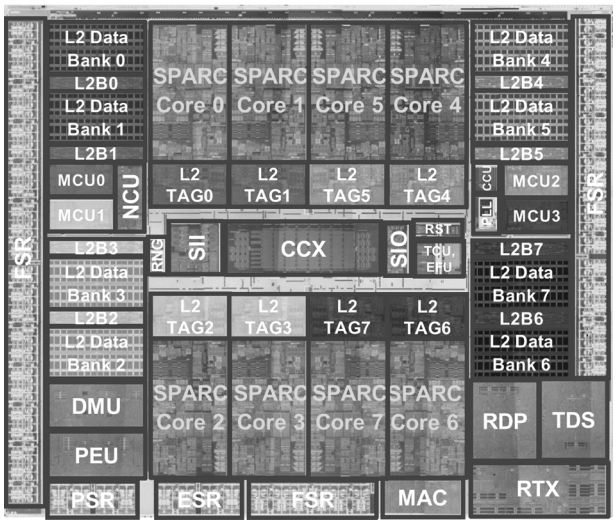


Fig. 3. Niagara2 die micrograph.

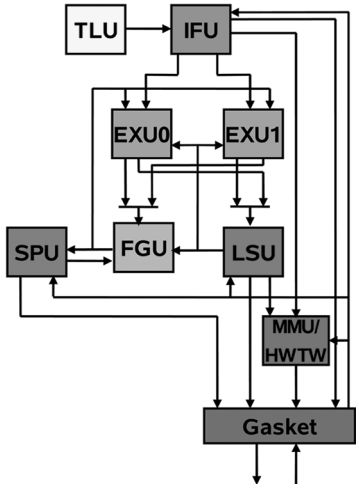


Fig. 4. SPC block diagram.

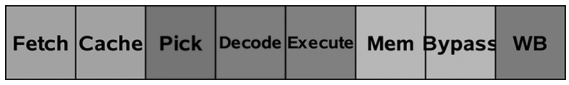


Fig. 5. Integer pipeline: eight stages.

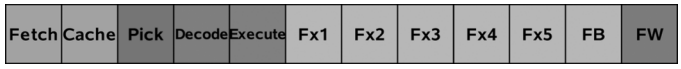


Fig. 6. Floating point pipeline: 12 stages.

two FBDIMM channels. These three major I/O interfaces are serializer/deserializer (SerDes) based and provide a total pin bandwidth in excess of 1 Tb/s. All the SerDes are on chip. The high levels of system integration truly makes Niagara2 a "server-on-a-chip", thus reducing system component count, complexity and power, and hence improving system reliability.

B. SPARC Core Architecture

Fig. 4 shows the block diagram of the SPARC Core. Each SPARC core (SPC) implements the 64-bit SPARC V9 instruction set while supporting concurrent execution of eight threads. Each SPC has one load/store unit (LSU), two Execution units (EXU0 and EXU1), and one Floating Point and Graphics Unit (FGU). The Instruction Fetch unit (IFU) and the LSU contain an 8-way 16 kB Instruction cache and a 4-way 8 kB Data cache respectively. Each SPC also contains a 64-entry Instruction-TLB (ITLB), and a 128-entry Data-TLB (DTLB). Both the TLBs are fully associative. The memory Management Unit (MMU) supports 8 K, 64 K, 4 M, and 256 M page sizes and has Hardware TableWalk to reduce TLB miss penalty. "TLU" in the block diagram is the Trap Logic Unit. The "Gasket" performs arbitration for access to the Crossbar. Each SPC also has an advanced Cryptographic/Stream Processing Unit (SPU). The combined bandwidth of the eight Cryptographic units from the eight SPCs is sufficient for running the two 10 Gb Ethernet ports encrypted. This enables Niagara2 to run secure applications at wire speed.

Fig. 5 and Fig. 6 illustrate the Niagara2 integer and floating point pipelines, respectively. The integer pipeline is eight stages long. The floating point pipeline has 12 stages for most operations. Divide and Square-root operations have a longer pipeline.

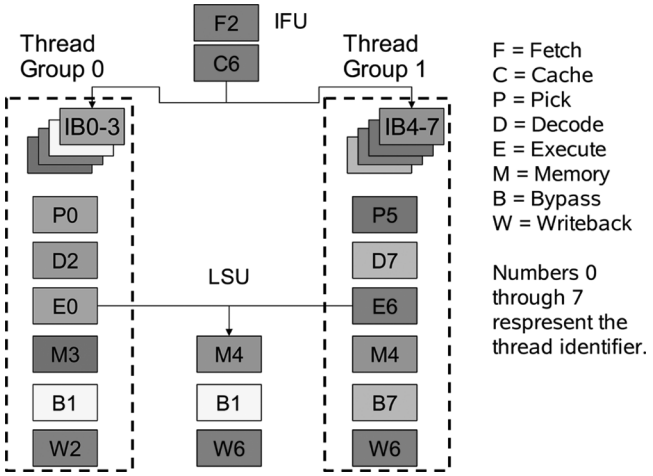


Fig. 7. Various threads occupying different pipeline stages.

Technology	65 nm CMOS (from Texas Instruments)
Nominal Voltages	1.1 V (Core), 1.5V (Analog)
# of Metal Layers	11
Transistor types	3 (SVT, HVT, LVT)
Frequency	1.4 Ghz @ 1.1V
Power	84 W @ 1.1V, 1.4 Ghz
Die Size	342 mm^2
Transistor Count	503 Million
Package	flip-chip glass ceramic
# of pins	1831 total; 711 Signal I/O

Fig. 8. Key statistical highlights.

The load-use latency is three cycles. There is a six-cycle latency for dependent FP operations. The ICACHE is shared between all eight threads. Each thread has its own instruction buffer. The Fetch stage/unit fetches up to four instructions per cycle and puts them into the thread's instruction buffer. Threads can be in "Wait" (as opposed to "Ready") state due to a ITLB miss, ICACHE miss, or their Instruction Buffer being full. The "Least-Recently-Fetched" algorithm is used to select one of "Ready" threads for which the next instruction will be fetched. Fig. 7 shows the Integer/Load/Store pipeline and illustrates how different threads can occupy different pipeline stages in a given cycle. In other words, threads are interleaved between pipeline stages with very few restrictions. The Load/Store and Floating Point units are shared between all eight threads. The eight threads within each SPARC core are divided into two thread groups (TGs) of four threads each. Once again, the threads could be in "Wait" states due to events such as a DCACHE miss, DTLB miss, or data dependency. The "Pick" stage tries to find one instruction from all the "Ready" threads (using the "Least-Recently-Picked" algorithm) from each of the two TGs to execute every cycle. Since each TG picks independently (w.r.t. the other TG), it can lead to hazards such as load instructions being picked from both TGs even though each SPC has only one load/store unit. These hazards are resolved in the "Decode" stage.

Niagara2's Primary and L2 cache sizes are relatively small compared to some other processors. Even though this may cause higher cache miss rates, the miss latency is well hidden by the presence of other threads whose operands/data is available and hence can make good use of the "compute" time slots, thus minimizing wastage of "compute" resources. This factor explains why the optimum design point moved towards having higher thread counts and lower cache sizes. In effect, this can be thought of as devoting more transistors on chip to the intelligent "processing" function as opposed to the nonintelligent "data-storing" function.

Performance measurements using several commercial applications and performance benchmarks (SpecJBB, SpecWeb, TPC-C, SpecIntRate, SpecFPRate, etc.) confirm that Niagara2 has achieved its goal of doubling the throughput performance and performance/watt as compared to UltraSPARC T1. Most of the gain comes from doubling the thread count and the number of execution units. Some of the gain comes from a higher operating frequency. Similarly, performance measurements using commonly used Floating Point benchmarks confirm that Niagara2's Floating Point throughput performance is more than an order of magnitude higher compared to UltraSPARC T1. Niagara2 has eight Floating Point units (FPUs), each occupying only 1.3 mm^2, against only one FPU for UltraSPARC T1. Also, the Niagara2 FPUs are within the SPCs as compared to UltraSPARC T1 where the SPCs had to access the FPU through the Crossbar. Another factor that helps performance is the higher memory bandwidth on Niagara2.

C. Key Statistical Highlights

The table in Fig. 8 lists some key statistical highlights of Niagara2's physical implementation. Niagara2 is built in Texas Instruments' 65 nm, 11LM, Triple-V_t CMOS process. The chip has $\sim$503 million transistors on a 342 mm^2 die packaged in a flip-chip glass ceramic package with 1831 pins. It operates at 1.4 GHz @ 1.1 V and consumes 84 W.

III. Memory Hierarchy, Caches, and Crossbar

Niagara2 has four memory controllers on chip, each controlling two FBDIMM channels. They are clocked by the DR (DRAM) clock, which nominally runs at 400 MHz corresponding to the FBDIMM SerDes link rate of 4.8 Gb/s. Up to eight DIMMs can be connected to each channel. Every transaction from each controller consists of 64 data bytes and ECC. Read transactions take two DR clock cycles, while Write transactions take four DR clock cycles. This yields a Read bandwidth of 51.2 GB/s and a Write bandwidth of 25.6 GB/s.

Niagara2 has two levels of caches on chip. Each SPC has a 16 kB Primary Instruction cache (ICACHE), and a 8 kB Primary Data cache (DCACHE). The ICACHE is 8-way set-associative with a 32 B line size. The DCACHE is 4-way set-associative with a 16 B line size. The 4 MB shared Level2 (L2) cache is divided into eight banks for 512 kB each. The number of banks are doubled to support the doubling of thread count as compared to UltraSPARC T1. The L2 cache is 16-way set associative with a 64 B line size. Each bank can read up to 16 B per cycle with a

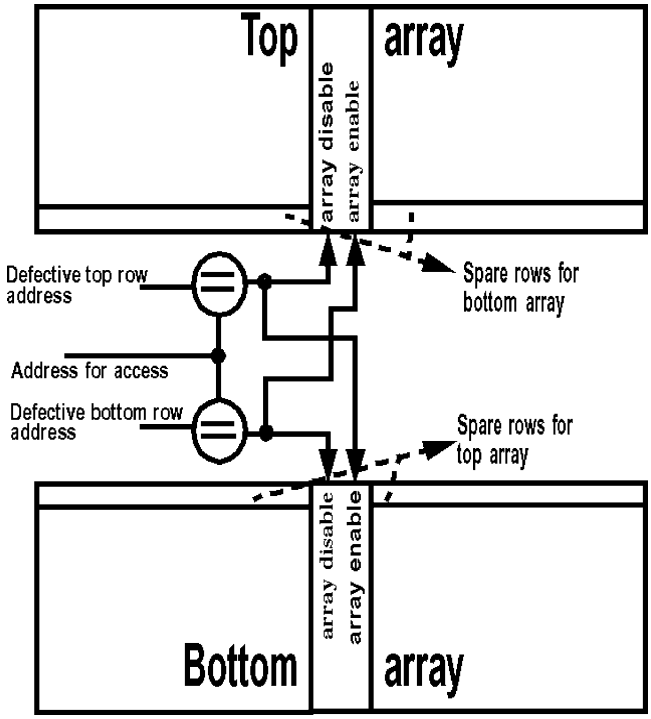


Fig. 9. L2 cache row redundancy scheme.

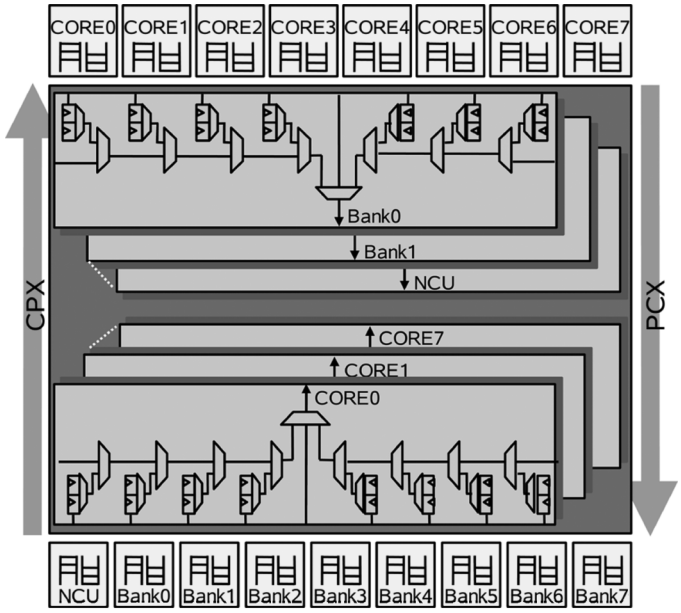


Fig. 10. Crossbar.

2-cycle latency. Addresses can be hashed to distribute accesses across different sets in case of hot cache sets caused by reference conflicts. All arrays are protected by single error correction, double error detection ECC, and parity. Data from different ways and different words is interleaved to improve soft error rates.

The L2 cache used a unique row-redundancy scheme. It is implemented at the 32 kB level and is illustrated in Fig. 9. Spare rows for one array are located in the adjacent array as opposed to the same array. In other words, spare rows for the top array are located in the bottom array and vice versa. When redundancy is enabled, the incoming address is compared with the address of the defective row and if it matches, the adjacent array (which is normally not enabled) is enabled to read from or write into the spare row. Using this kind of scheme enables a large (>30%) reduction in X-decoder area. The area reduction is achieved because the multiplexing required in the X-decoder to bypass the defective row/rows in the traditional row redundancy scheme is no longer needed in this scheme.

N-well power for the Primary and L2 cache memory cells is separated out as a test hook. This allows weakening of the pMOS loads of the SRAM bit cells by raising their threshold voltage, thus enabling screening cells with marginal static noise margin. This significantly reduces defective parts per million (DPPM) and improves reliability.

Fig. 10 shows the Niagara2 Crossbar (CCX). CCX serves as a high bandwidth interface between the eight SPARC Cores, shown on top, and the eight L2 cache banks, and the non-cacheable unit (NCU) shown at the bottom. CCX consists of two blocks: PCX and CPX. PCX ("Processor-to-Cache-Transfer") is a 8-input 9-output multiplexer (mux). It transfers data from the eight SPARC cores to the eight L2 cache banks and the NCU. Likewise, CPX ("Cache-to-Processor Transfer") is a 9-input 8-output mux, and it transfers data in the reverse direction. The PCX and CPX combined provide a Read/Write bandwidth of $\sim$270 GB/s. All crossbar data transfer requests are processed using a four-stage pipeline. The pipeline stages are: Request, Arbitration, Selection, and Transmission. As can be seen from the figure, there are $8 * 9 * 2 = 144$ possible source destination pairs for each data transfer request. There is a two-deep queue for each source–destination pair to hold data transfer requests for that pair.

IV. Clocking

Niagara2 contains a mix of many clocking styles—synchronous, mesochronous and asynchronous—and hence a large number of clock domains. Managing all these clock domains and domain crossings between them was one of the biggest challenges the design team faced. A subset of synchronous methodology, ratioed synchronous clocking (RSC) is used extensively. The concept works well for functional mode while being equally applicable to at-speed test of the core using the SerDes interfaces.

A. Clock Sources and Distribution

An on-chip phase-locked loop (PLL) uses a fractional divider [8], [9] to generate Ratioed Synchronous Clocks with support for a wide range of integer and fractional divide ratios. The distribution of these clocks uses a combination of H-trees and grids. This ensures they meet tight clock skew budgets while keeping power consumption under control. Clock Tree Synthesis is used for routing the asynchronous clocks. Asynchronous clock domain crossings are handled using FIFOs and meta-stability hardened flip-flops. All clock headers are designed to support clock gating to save clock power.

Fig. 11 shows the block diagram of the PLL. Its architecture is similar to the one described in [8]. It uses a loop filter capacitor referenced to a regulated 1.1 V supply (VREG). VREG is generated by a voltage regulator from the 1.5 V supply coming

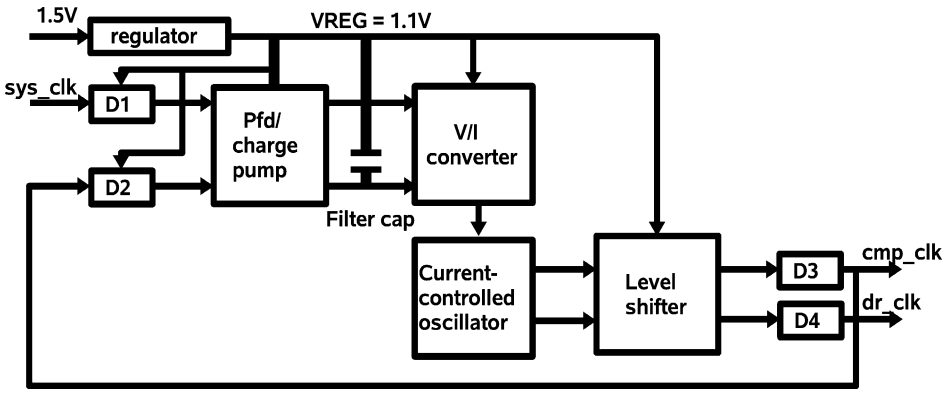


Fig. 11. PLL block diagram.

from the C4 bumps. The PLL uses a proportional and integral control compensation scheme. The current-controlled oscillator has relatively low swing, so it needs a level shifter to convert its swing to full rail. Apart from the two dividers D3 and D4, all components of PLL use the VREG supply to improve power supply rejection. D3 and D4 use the 1.1 V digital supply. The voltage-controlled oscillator (VCO) is composed of a V/I converter, a current-controlled oscillator, and a level shifter.

Fig. 12 provides an overview of chip level clocking and distribution, highlighting the major clock grids: l2clk, drl2clk, iol2clk and pcl2clk (nominal frequencies: 1.4 GHz, 400 MHz, 350 MHz, and 250 MHz, respectively). Another clock grid io2xl2clk (nominal frequency 700 MHz) exists in the MAC solely to double-pump single-port SRAMs that are used as dual-port memories at half-rate. Other smaller domains are concentrated mostly at the SerDes boundaries of the Memory Control Unit (MCU), PCI-Express Unit (PEU). and the Media Access Control (MAC). The CMP clock (the high-frequency Chip Multi Processing clock which clocks the SPCs, L2 cache, and Crossbar) and DR clock frequencies can vary in ratio from 2.00–5.25 in functional mode. The key relationships in functional mode are that the DR clock always runs at twice the sys_clk frequency, and the I/O clock runs at quarter rate of CMP. The system clock that drives the core PLL in Niagara2 is a copy of the reference clock (fbd_ref) driven by a clock buffer chip to the FBDIMM modules. As a result, there is no long-term drift between the DR and SerDes recovered clocks in the MCU, so the clock domains are mesochronous. However, the PCIe and XAUI interfaces get reference clocks from independent sources. Unlike the MCU, the boundaries in the PEU and MAC are asynchronous.

The I/O clock is distributed as a clock only within clusters by re-timing to l2clk a phase signal (io_phase) which toggles at a reduced CMP rate. Pipelining the common phase is cheaper and more efficient than a custom top level distribution. The cluster headers perform re-timing, clock gating, muxing, and related DFT functions before driving clock grids using pre-grid drivers. In short, iol2clk is derived from l2clk within any cluster and hence the iol2clk-l2clk skew is comparable to l2clk-l2clk skew across clusters. On the other hand, the latencies of CMP and DR clocks are loosely matched and therefore may exhibit large inter-domain skew in a single MCU. Large skew in this context is defined as skew approaching CMP cycle times.

Since the three gridded clocks, l2clk, drl2clk, and iol2clk, are ultimately derived from the same reference, they are ratioed synchronous. However, only l2clk–drl2clk and iol2clk-l2clk crossings need to be addressed. The known periodic relationships of the two interfaces are exploited to perform simplified domain crossing, described next. Before proceeding, it is necessary to make a distinction between clocks at the destination (suffixed by l2clk) and those at the source (prefixed by pll_) in Fig. 12 and Fig. 14; as shown in Fig. 12, the latency may vary between half to one and a half CMP cycles.

B. Domain Crossings for Ratioed Synchronous Clocks

One of the major design goals of Niagara2 was to support deterministic and repeatable behavior in functional mode for system debug and test. The serial I/O interfaces which account for 90% of the pins present challenges to that goal, but most of the logic directly supports it through ratioed synchronous clocking. In system debug mode, the repeatability issue is addressed by buffering in the memory pathway to account for maximum round-trip delays through the FBDIMM interfaces. Design for manufacturing tests achieves determinism and tester level cycle repeatability using a special mode described later.

Ratioed synchronous clocks have a number of other advantages: FIFO designs are simplified; timing analysis is more robust since there are no false paths in this scheme; and functional test vector generation becomes much easier. Sync-pulse-based domain crossing for the generic case in Niagara2 is illustrated in Fig. 13. All measurements are with respect to a starting point at time 0, or $k = 0$, where Fast Clock (FCLK), Slow Clock (SCLK), and the reference clock have their rising edges aligned. The *enable flop* always operates on the rising edge of the faster clock domain, and the pulse remains active for exactly one fast clock cycle. Also, data transfer is allowed every slow clock cycle thus achieving maximum communication throughput. To successfully use sync pulses, the following must be true:

a) There needs to be a co-incident or reference edge from which sync pulses can be generated. Thus, for a 2.75:1 ratio, the CMP clock rising edge would align with the DR clock edge every 11 CMP or 4 DR cycles.
b) Clock edges which coincide *nominally* at the source will have moved apart at the destination. Therefore, pll_cmp_clk–pll_dr_clk phase alignment at the PLL

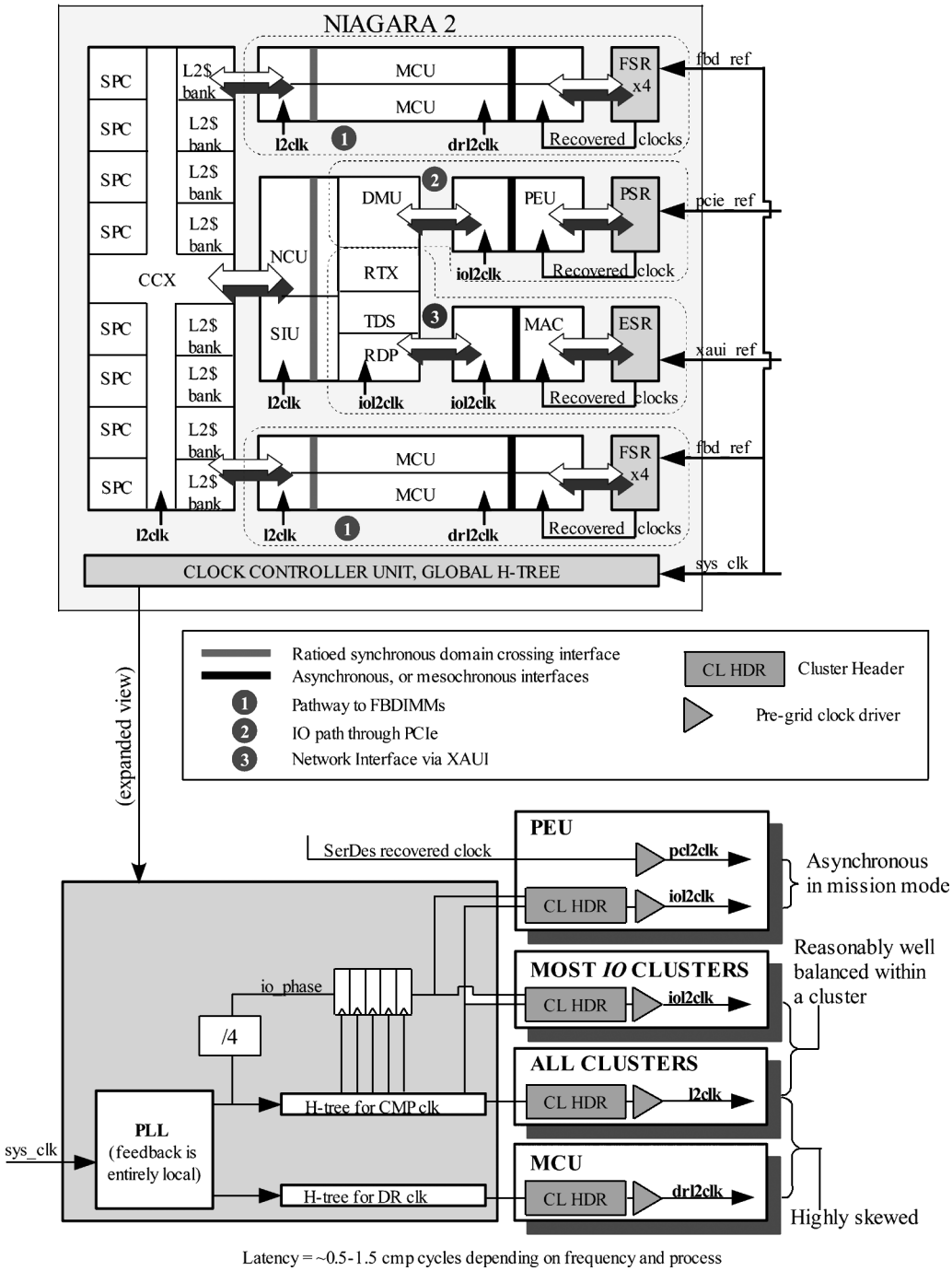
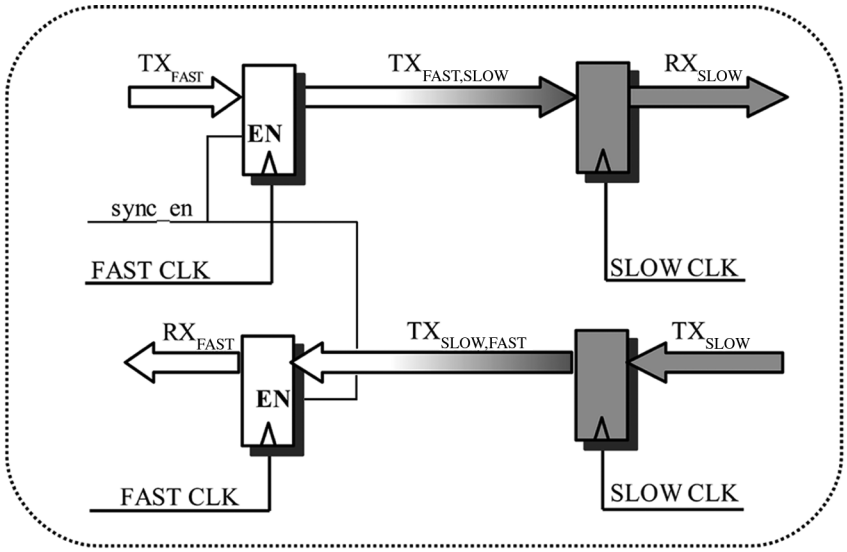


Fig. 12. Chip level clocking overview and major domains clock in Niagara2.

every 4 DR cycles is lost at the point of l2clk–drl2clk crossing after traversing 24 mm on chip.

c) Placement of the pulses to enable data transfers needs to meet setup and hold constraints for all conditions of PVT. This involves creating timing budgets that are used to drive the static timing analysis (STA) tools, as well as confirm design points.

In Niagara2, all conditions are satisfied by the generation and distribution of sync pulses as outlined in Fig. 14. The align detection block generates an *aligned* pulse deterministically using the fact that the reference and CMP clocks are phase aligned at the PLL due to local feedback. This *aligned_A* signal is effectively the output of a digital phase detector, its periodicity reflecting the common base period of the reference clock. *Aligned_A* is then used to track the CMP tree latency using delayed clocking to arrive at the Clock Control Unit (CCU) boundary as *Aligned_B*. Thus, *Aligned_A* is synchronous to cmp_pll_clk whereas *Algined_B* is synchronous to l2clk no matter what the balanced CMP latency. This is a cheaper alternative than having the PLL reference clock being sent out side by side along with the CMP clock. *Aligned_A* has one other function of generating a synchronous reset for the



f_{REF}	Frequency of base reference clock
f_{FCLK}	Frequency of fast clock derived from reference
f_{SCLK}	Frequency of slow clock derived from reference
T	Period of FCLK
k	Any slow clock (SCLK) cycle in the integer range 0,1,2, ... M-1

$$f_{FCLK} = Nf_{REF}, f_{SCLK} = Mf_{REF}, f_{FCLK}:f_{SCLK} = N:M$$

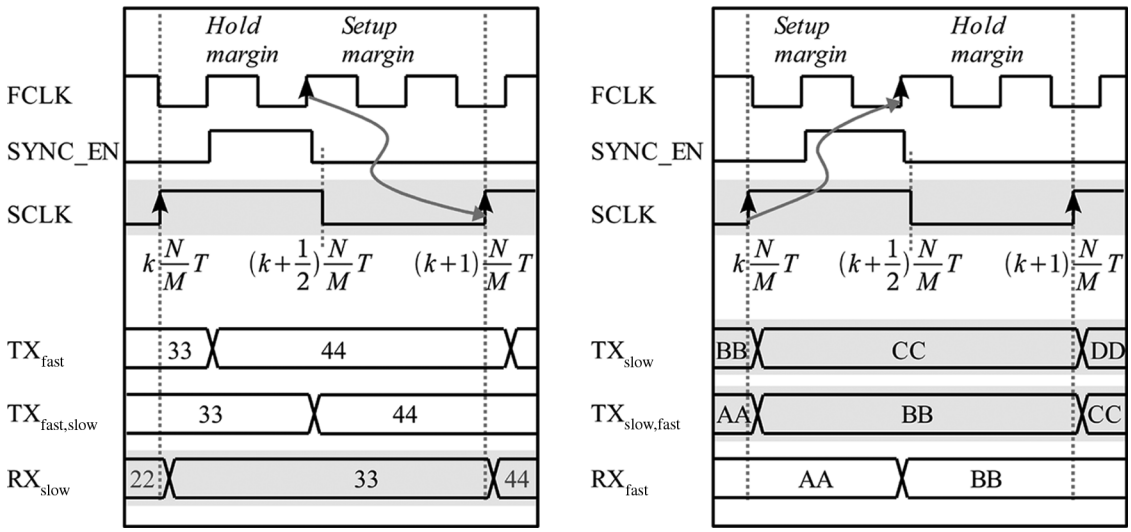


Fig. 13. Ratioed synchronous clock domain crossings.

fractional divider in the PLL such that dr_clk, cmp_clk, and ref_clk always have a coincident rising edge.

The only part that remains in the scheme is to generate the sync pulses. For maximum theoretical skew tolerance, in a CMP-DR crossing, the CMP rising edge that is closest to the center of the slow clock cycle at the domain crossing boundary is chosen. If two CMP edges are equidistant from the midpoint of the slow clock cycle, the earlier edge is chosen. This tends to roughly equalize setup and hold margins (with a slight bias for setup time) in any given slow clock cycle k. Referring back to Fig. 13, if the setup margin for any cycle k is $t_{k,\mathrm{setup-margin}}$, the hold margin for that cycle is given by $(N/M)T - t_{k,\mathrm{setup-margin}}$, where the slow clock period is $(N/M)T$. For CMP:DR ratios, k is in the range {0,3}, and hence there are four possible unique positions in each ratio. The margin (or equivalently, skew bounds) for ratioed synchronous crossings can further be expressed as shown in Table I.

From Table I, higher ratios of N/M are better, while lower CMP frequency gives more margin. Since DR frequency remains constant for a given DRAM choice, it is not clear whether the margin improves or reduces as one moves down the ratio table. So an entire table of sync pulses is built based on the al-

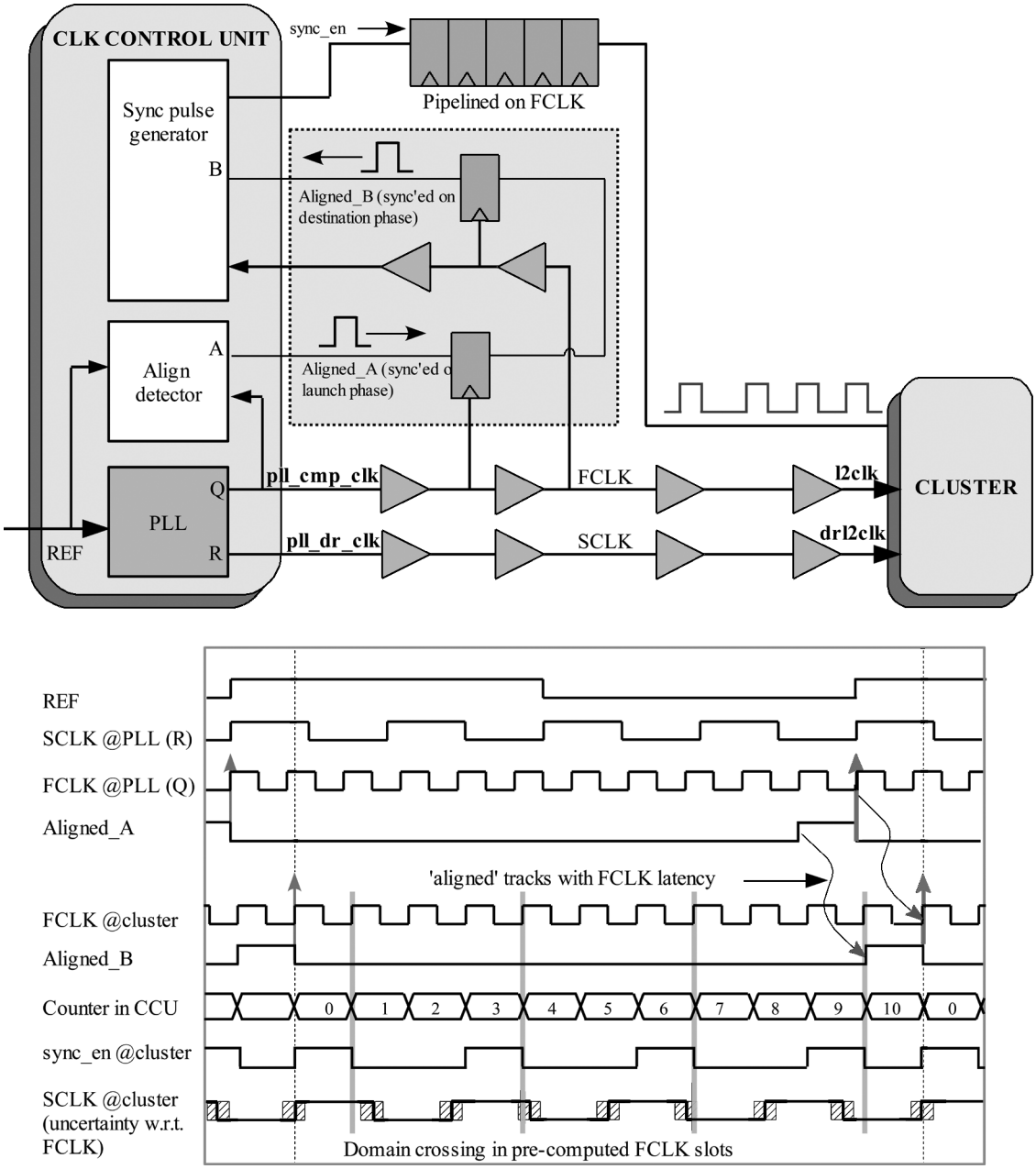


Fig. 14. Generation and distribution of sync pulses for 2.75:1 CMP:DR frequency ratio.

gorithm, where each row corresponds to a particular ratio, and each entry in the row describes the margin for that particular slow clock cycle k. Since the intent is to try to equalize setup and hold time margins, the obvious choice is to use the same sync pulse for data transfer in both directions. This has two effects: 1) the number of distributed pulses is halved, and 2) the analysis for determining setup/hold margin needs to be performed in only one direction. Because of the symmetry and periodicity, the worst case margin across both tables becomes *exactly* the same.

A fallout from this sync pulse scheme is that for static timing analysis (STA) performed at CMP frequency of 1.4 GHz, the domain crossing timing path is specified as a multi-cycle path with one CMP cycle ($=$ 714 ps @ 1.4 GHz) for setup/hold. The STA tools easily handle constraints specified in this manner. After accounting for clk-Q, propagation delay, flip-flop setup/hold times, clock skew and uncertainty, the tool dumps out a slack in the report, thus eliminating the need for post-processing thousands of inter-domain paths. Hence, the interfaces can be timed just as robustly to match the design.

V. On-Chip SerDes Interfaces

As stated earlier, Niagara2's three major I/O interfaces, namely the PCI-Express, XAUI-Ethernet, and FBDIMM memory interfaces are all SerDes based. All of them share a common microarchitecture. The major difference is that FBDIMM SerDes use VSS-referenced signaling as opposed to VDD-referenced signaling used by the PCI-Express and XAUI-Ethernet SerDes. In general, VDD-referenced signaling

TABLE I
MARGIN FOR RATIOED SYNCHRONOUS CROSSINGS

N/M = 2,4,6 ... (even)	$t_{k,\text{setup-margin}} = 0.5(N/M)T$	$t_{k,\text{hold-margin}} = 0.5(N/M)T$
N/M = 3,5,7 ... (odd)	$t_{k,\text{setup-margin}} = 0.5(N/M)T + 0.5T$	$t_{k,\text{hold-margin}} = 0.5(N/M)T - 0.5T$
N/M = arbitrary ratio	$0.5(N/M)T - T < t_{k,\text{setup-margin}} \leq 0.5(N/M)T + 0.5T$	$0.5(N/M)T - T < t_{k,\text{hold-margin}} \leq 0.5(N/M)T$

	FBDIMM	PCI-Express	Ethernet-XAUI
Signalling Reference	VSS	VDD	VDD
Link-rate (Gb/s)	4.8	2.5	3.125
# of North-bound (Rx) lanes	14 * 8	8	8 * 2
# of South-bound (Tx) lanes	10 * 8	8	8 * 2
Bandwidth (Gb/s)	921.6	40	50

Fig. 15. Niagara2's SerDes interfaces.

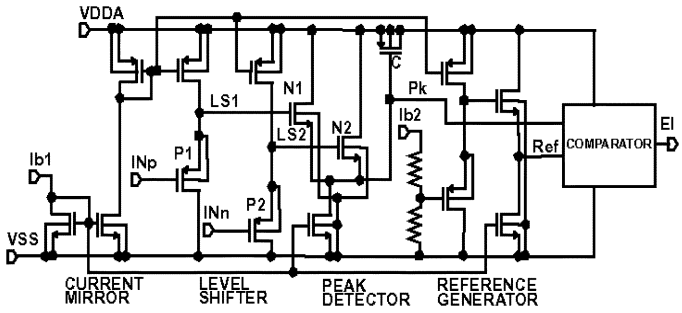


Fig. 16. Electrical idle detector (EID).

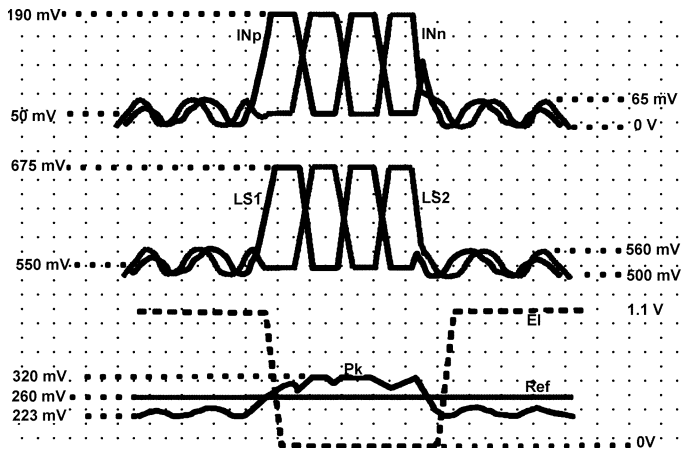


Fig. 17. EID waveforms.

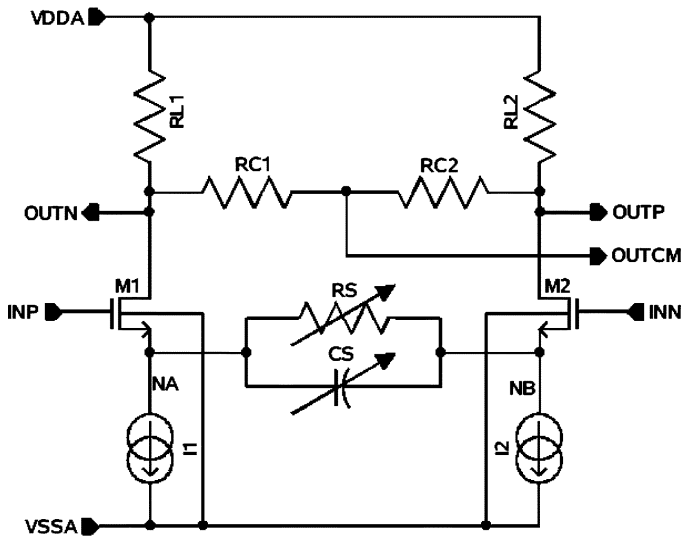


Fig. 18. Equalizer circuit used in SerDes macros.

schemes have a technology advantage. Analog circuits, especially differential style circuits favor NFETs, which have higher mobility resulting in better performance. Hence, level-shifters were employed to enable circuits from VDD-referenced PCI-Express and XAUI-Ethernet SerDes designs to be re-used for VSS-referenced FBDIMM SerDes. The table in Fig. 15 summarizes the number of lanes, link rates, and pin bandwidth of the Niagara2's three SerDes Interfaces. As can be seen from the last row in the table, the three SerDes interfaces combined provide a pin bandwidth in excess of 1 Tb/s.

A. Electrical Idle Detector

One of the interesting circuits in the SerDes is the Electrical Idle Detector (EID), illustrated in Figs. 16 and 17. Its primary role is to detect a quick reset which occurs when the remote transmitter (TX) is electrically idle (EI) and then becomes active again. EI is determined when the three lowest lanes simultaneously detect the loss of a signal with both the INn and INp below 65 mV. The level shifter comprised of source follower PFETs P1 and P2 level shift VINp and VINn to VLS1 and VLS2. NFETs N1 and N2 act like an analog OR gate, increasing Vpk if either VLS1 or VLS2 rises. Capacitor C helps retain Vpk over cycle transitions. Vpk is then compared with the Vref to detect EI. Bias currents Ib1 and Ib2 are generated using a circuit consisting of a bandgap reference, precision resistors, and current mirrors.

B. Equalizer

The Equalizer circuit is illustrated in Fig. 18. This circuit consists of two matched inverting amplifiers for each of the inputs INN and INP. The resistor RS placed between the source terminals of the two NFETs help program the gain of the circuit. At lower frequencies, the capacitor CS acts as an open circuit, and hence RS helps set the DC gain of the circuit. The lower the value of RS, the higher is the DC gain. Capacitor CS is, in effect, a programmable zero that allows the gain to rise as frequency rises because the impedance of the capacitor reduces as the frequency rises. Thus, this network can be tuned to provide a complementary frequency response characteristic to that of the wiring channel. As a result, the net system frequency response is flattened and the range of the system frequency response is

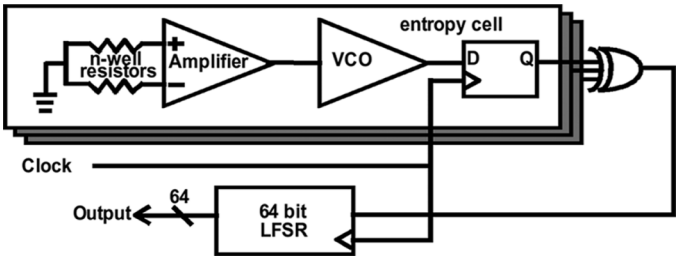


Fig. 19. Niagara2's true random number generator.

extended. At high enough frequencies, the amplifiers roll off and the system becomes unstable. Therefore, NFETs provide the best choice for this type of equalization circuit. The sole purpose of resistors RC1 and RC2 is to generate a common mode signal.

VI. Cryptography Support

An extremely important feature of Niagara2 is its extensive support for cryptography and the fact that the cryptographic engines provide sufficient bandwidth to enable running secure applications at wire speeds. There is a Cryptographic unit inside each SPC. It implements various encryption and decryption algorithms, hash functions, checksum generation, and modular arithmetic. A true random number generator inside Niagara2 provides support for these functions. All the commonly used cipher algorithms are supported, including RC4, AES, DES, Triple-DES, SHA-1, SHA-256, MD5, and CRC32c.

Fig. 19 illustrates the on-chip true random number generator. It consists of three entropy cells. Each cell by itself provides sufficient entropy. Having three cells provides redundancy and also enables reduction of entropy accumulation time to get the desired amount of entropy. The source of entropy is thermal noise from n-well resistors. This noise, after amplification by a differential amplifier, modulates the VCO frequency. On-chip clock samples the VCO output and sends it to a linear feedback shift register (LFSR) which accumulates entropy over a pre-set accumulation time. Privileged software can program a timer with the desired entropy accumulation time. This timer blocks loads from the LFSR before the pre-set entropy accumulation time has elapsed to make sure the random number in the LFSR has enough entropy before it is used.

VII. Physical/Global Design and Methodology

Clusters are composed and routed flat (i.e., not hierarchically) for better route optimization. Over-the-block routing is extensively used. Repeaters and wire-classes were designed to enable crossing the widest clusters without the need for repeaters. In exceptional cases where wires needed repeaters within clusters, area-pins were used to drop down into the cluster, get repeated, and come back out again. Custom clock insertion is used to meet tight clock skew budgets. Most of the design uses a static cell-based design methodology. Low-Vt gates are used to selectively speed up critical paths; however, their use was tightly controlled to have minimal impact on leakage power.

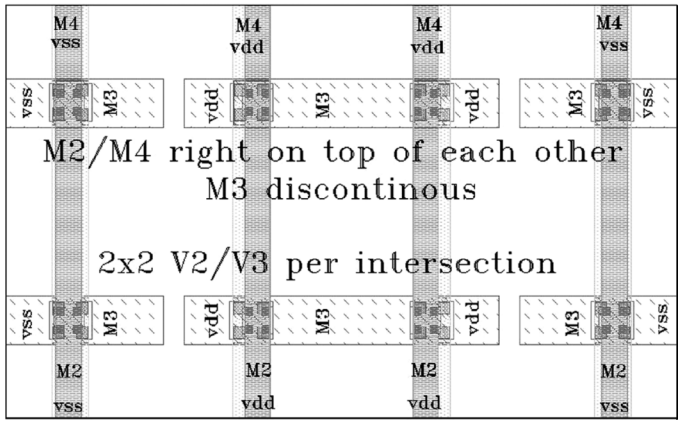


Fig. 20. M3 post power grid.

A. DFM Methodology

Niagara2 employs an extensive Design for Manufacturing (DFM) methodology. Single poly-orientation was used everywhere except I/O cells which had to be rotated by 90 degrees between perpendicular sides of the die. Larger-than-minimum design rules were used in certain cases (e.g., to reduce the effects of poly/diffusion flaring, near stress-prone topologies to reduce chances of dislocations in Si-lattice, etc). A limited number of gate-poly pitches were used so that the Optical Proximity Correction (OPC) algorithm could be optimized for them to yield better gate-CD control. Dummy polys were used extensively to shield gates to reduce gate-CD variation. OPC simulations of critical cell layouts were performed to ensure sufficient manufacturing process margin. Statistical simulations were extensively used to reduce unnecessary design margin that could result from designing to the FAB-supplied corner models. Redundant vias were placed and metal overlap of contacts/vias was increased wherever possible. Most of the Niagara2 design uses a static cell-based design methodology. All custom designs, which also used nonstatic circuit design styles, were proven on test chips prior to first silicon.

B. Power Grid and Decoupling Capacitors (DECAPs)

Current is supplied to the chip through >5600 bumps, which are about equally divided between VDD and VSS bumps. A lot of emphasis was placed on maintaining continuity of the power grid wherever possible. All the global metals from M11–M7 are continuous in most parts of the chip and the lower level grids are stitched together wherever possible. A minimum of 2×2 vias were required at every power grid intersection. The power grid in each metal layer was made symmetric; this allowed block layouts to be easily reused even if blocks are flipped.

A novel method called "M3 post" (illustrated in Fig. 20) was used for M4 to M2 power hook-up. This method involved having discontinuous VDD and VSS as opposed to the conventional method of having a continuous but alternate VDD and VSS on M3. Direct connections from M2 to M4 due to stacked vias (V2/V3's) helped reduce localized (M2–M4) power supply variations. IR drop was also reduced due to having double the V2/V3's as compared to the conventional method. The reduced current flow in M3 due to this method made it easier to meet

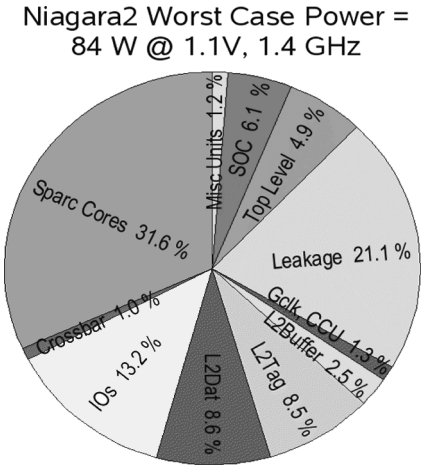


Fig. 21. Power consumption.

Power EM spec. For example, for a typical layout of big buffers, the worst power EM violation reduced from $\mathrm{J/Jmax}$ of 1.57 with a conventional M3 power grid to $\mathrm{J/Jmax}$ of 0.93 with M3 post grid. Similarly, IR drop from M4 to devices improved from 89.1 mV to 55.8 mV.

An interactive script was used for insertion of DECAPs based on available empty space, and the placement was always assured to be DRC/LVS clean. The width of DECAPs used in standard cell blocks matched the width of the standard cells. The channel DECAPs were similar to DECAPs used in standard cell blocks, except that they had M4 and below embedded into the leaf DECAP cell to reduce data base size. About 700 nF of explicit DECAP was added on chip (this does not include implicit DECAP due to the metal power grid and the quiet cap).

VIII. Power and Power Management

A. Power

Niagara2's SOC design is optimized for performance/watt and enables reduction of total power consumption and power density at the chip and system level. Niagara2's simplified pipeline and reduced speculation in instruction execution reduces wasted power. A Niagara2-based system is a lot more power efficient as compared to, for example, a system with eight single-core processors (on separate chips) each having their own I/O (DRAM, networking, and PCI-Express) interfaces. Such a system will have 8 times the I/O interfaces and hence will consume a lot more power in those interfaces. Also, extra power will be consumed in driving the off-chip multi-processor coherency fabric. In comparison, for Niagara2 there are only one set of I/O interfaces and the coherency between the eight processor cores is handled on chip by the crossbar, which consumes less than 1 W of power.

Niagara2's total power consumption is 84 W @ 1.1 V and 1.4 GHz operation. The pie-chart in Fig. 21 shows the power consumed by the various blocks inside Niagara2. Almost a third of the total power is consumed by the eight SPARC cores. L2 cache Data, Tag and Buffer blocks together account for $\sim$20% of the total. SOC logic consumes 6% while I/Os consume 13%. Leakage is about 21% of the total power. Clocks to unused clusters are gated off to save dynamic power. Within units, clocks to groups of flops are separated into independent domains depending upon the functionality of the corresponding logic. Clocks to each domain can be turned off independently of one another when the related logic is not processing valid instructions. This saves dynamic power.

B. Technique to Reduce Leakage Power

Niagara2 uses gate-bias (GBIAS) cells to reduce leakage power. GBIAS cells are footprint-compatible with the corresponding standard-Vt non-GBIAS versions. The only layout difference is that the GBIAS cell has an additional identification layer (GBIAS). All the transistors of any cell having this layer are fabricated with 10% longer channel length.

The table in Fig. 22 illustrates the reduction in leakage and corresponding increase in delay as the channel length was increased above minimum for three different gates. 10% larger channel length was chosen, resulting in, on an average, about 50% reduced leakage on a per-cell basis with about 14% impact on the cell delay. High-Vt (HVT) cells were considered as well for leakage reduction. We did not have an unconstrained choice of Vt for the HVT cells. For HVT cells using the available HVT transistors, the delay impact was much larger. As a result, approximate calculations lead to the conclusion that using HVT cells would have enabled substitution of only about one-third of the number of gates as compared to using GBIAS gates with 10% larger channel length. Hence, the GBIAS option was chosen. This enabled about 77% additional leakage saving as compared to using HVT cells. Cells in non-timing-critical paths could be replaced by their GBIAS versions as long as this did not result in any new noise, slew, or timing violations. Because of footprint compatibility, the swapping was easily done at the end of the design cycle without any timing, noise, or area impact. This reduced leakage power by 10%–15%.

The swapping algorithm works as follows. The project STA tool and a set of scripts determine which cells can be swapped without creating new timing, noise, and slew violations. First, all cells that have timing margins larger than a set threshold are swapped to their GBIAS equivalent. The STA tool then computes the timing graph and swaps back all GBIAS cells that are on paths with less than a predefined positive slack. Then, a script evaluates all the receivers connected to the outputs of the GBIAS cells that went through timing qualification, and determines if sufficient noise margin exists. The script calculates the new noise values by increasing the latest noise snapshot by a percentage that was derived from simulations and analysis. Once a list of cells that can be safely swapped is built, a custom program performs the swaps in the actual layout database. A footprint-compatibility check for identical pins and metal shapes is built into the program to maintain LVS and DRC cleanliness.

C. Power Management

Niagara2 has several power management features to manage power consumption. Software can dynamically turn threads on or off as required. Niagara2 has a Power Throttling mode which provides the ability to control instruction issue rates to manage power. The graph in Fig. 23 shows that this can reduce dynamic power by up to 30% depending upon the level of workload.

Channel L	INV		NAND2		NOR2	
	Ioff (avg N, P)	Delay	Ioff (avg N, P)	Delay	Ioff (avg N, P)	Delay
Lmin	1.000	1.000	1.000	1.000	1.000	1.000
1.025*Lmin	0.800	1.035	0.810	1.035	0.810	1.035
1.050*Lmin	0.660	1.070	0.725	1.069	0.675	1.069
1.075*Lmin	0.555	1.106	0.570	1.103	0.580	1.103
1.10*Lmin	0.475	1.140	0.495	1.137	0.500	1.138
1.125*Lmin	0.415	1.175	0.430	1.170	0.445	1.172
HVT Lmin	0.085	1.427	0.095	1.439	0.090	1.448

Average Leakage saving with 1.10*Lmin GBIAS swap ->	0.51
HVT swap equivalent to how many GBIAS swap ->	3.17
Leakage saving with one HVT swap ->	0.91
Leakage saving with 3.17 GBIAS swap ->	1.61
Additional leakage savings by using GBIAS instead of HVT ->	77.44%

Fig. 22. Leakage/delay of HVT and longer channel length (GBIAS) cells.

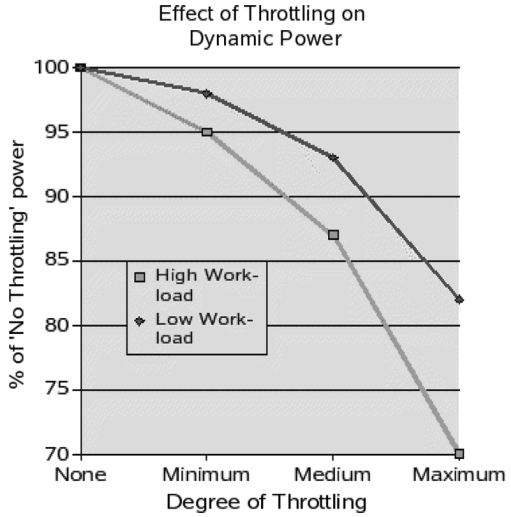


Fig. 23. Power throttling.

On-chip thermal diodes monitor the die temperature and help ensure reliable operation in case of exceptional circumstances such as failure of the cooling system. On-chip memory controllers enable DRAM power-down modes and control DRAM access rates to control memory power.

IX. DFT Features

Niagara2 has several DFT features to aid testing and debug. Since almost all the physical I/Os on Niagara2 consist of SerDes, functional testing of the Core presents several challenges. For one, SerDes link speeds are higher than what most standard testers can handle. Also, standard testers lack the intelligence required for SerDes protocol-level testing. Regular shmoos fail due to the inherent in-determinism caused by unknown phase alignment of the SerDes. A deterministic test mode (DTM) was added in Niagara2 specifically to solve this problem. In addition, Niagara2 has a dedicated debug port (168-pin non-SerDes-based I/Os) which has the ability to monitor various on-chip signals to help testing and debug. All except a few flops in Niagara2 are connected in 1 of 32 scan chains to enable ATPG and SCAN testing. All RAM and CAM arrays in Niagara2 are testable using MBIST and Direct Memory Observe (DMO) using Macrotest. DMO reduces test time by enabling fast bit-mapping required for array repair. The Transition Test technique is used for speed testing of targeted critical paths. SERDES designs incorporate external, internal, and pad loopback capabilities to enable their testing. Architecture design enables use of < 8 SPCs and/or L2 cache banks. This has proved to be a valuable feature for Niagara2 because it has shortened our debug cycle by making partially functional die usable. It will also provide an additional advantage of increasing overall yield by enabling partial core products.

A. Functional Testing Using DTM

Testers are synchronous devices, and need cycle level accuracy. As shown in the conceptual model in Fig. 24, serial links protocols for FBDIMM 1.0 and PCIe 1.0 use embedded clock within data that are skewed independently and arbitrarily. Therefore, there is indeterminism on the recovered clock-data bundles with respect to received data; in addition, the recovered *byte clocks* for the bundles are mesochronous. The transmit side is similar and no better.

However, without using the SerDes interfaces, it would be impossible to get large amounts of data on chip to perform functional testing of the cores in a cost effective way. Niagara2 addresses the problem by introducing a special reduced rate DTM where the recovered data from the serial links are driven in a controlled manner to transfer data deterministically and repeatably. The PCI-Express SerDes block (PSR) gets the same reference clock as sys_clk and fbd_ref. The CMP domains continue to operate at full speed, and their outputs are observed at reduced rates through CMOS drivers in the debug port. Outbound data through the FBDIMM SerDes (FSR) and PSR links unfortunately cannot be forced to be deterministic; instead, they are compressed, and routed to the debug port for strobing on the tester. The Network Interface path does not participate in DTM; however, the lack of one high-speed interface does not hamper DTM effectiveness.

Although the baud rate for SerDes are lowered to accommodate tester capability, the macros including the clock-data recovery (CDR) sub-blocks fully operate in mission mode and make no distinction between functional and test mode. DTM uses two concepts; continuing with the FBDIMM1.0 example:

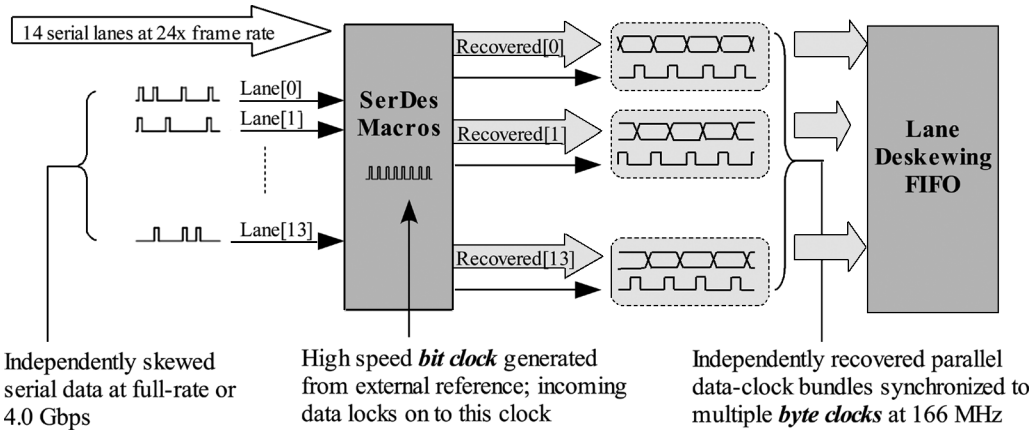


Fig. 24. Clock–data recovery of a single FBDIMM 1.0 receive channel within an FSR cluster.

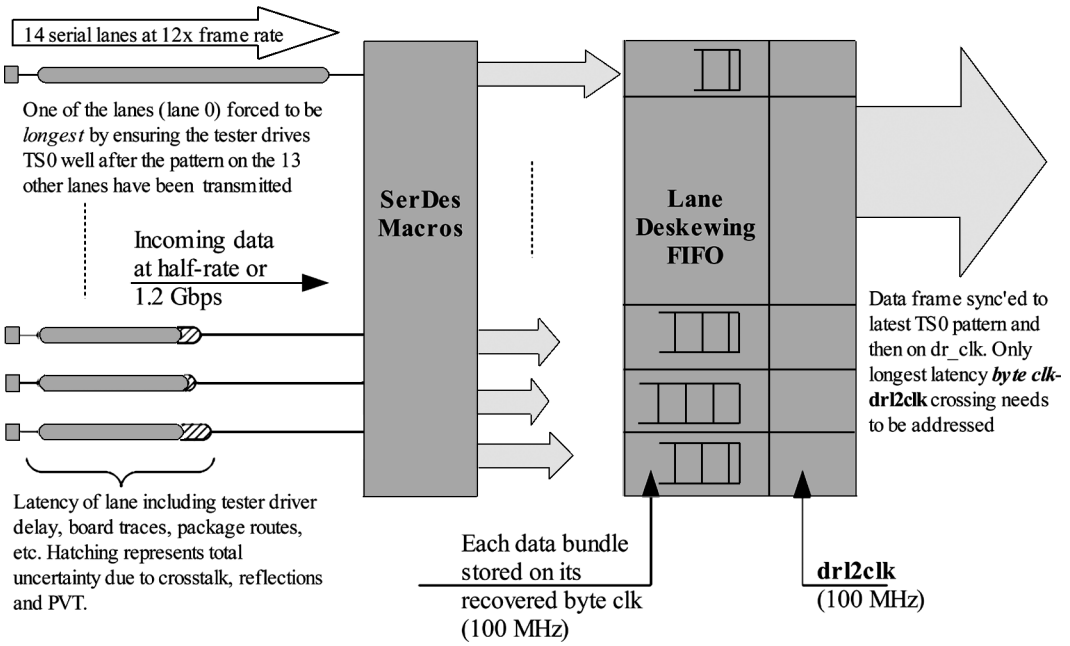


Fig. 25. Clock-data recovery of a single FBDIMM 1.0 receive channel (expanded) for DTM.

i) the link training pattern (TS0) can be used to identify the start of a clock-bundle pair, and hence control the generation of the slow byte clock with respect to TS0 start times;
ii) the queue in the MCU which performs aggregation of the data to construct frames need to wait for the slowest bundle, and therefore the slowest link.

Fig. 25 illustrates how the FBDIMM (and similarly PCI-Express) channels may be manipulated to generate clocks within an acceptable tolerance for synchronous crossing into the controller. The core clock speed is allowed to vary independently such that the CMP:DR and CMP:I/O clock ratios are always integers in DTM. In this manner, the processor core clock may be swept from 8× sys_clk to 15× sys_clk for functional at-speed testing.

Fig. 26 shows the frequency versus VDD shmoo plot at 95 °C from first silicon. As can be seen, the chip operates at 1.4 GHz at 1.1 V with sufficient margin.

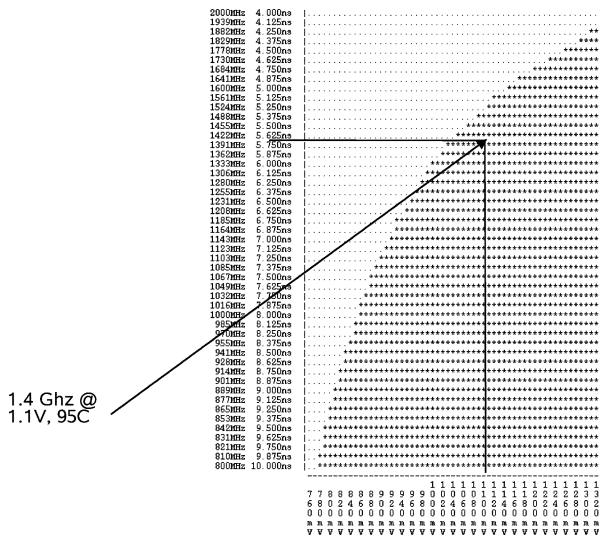


Fig. 26. VDD versus frequency shmoo plot.

X. Conclusion

Niagara2 is the second-generation 8-core 64-thread SPARC processor from Sun Microsystems based on the power-efficient CMT architecture optimized for Space, Watts (power), and Performance (SWaP). It integrates the major server functions on chip, thus reducing system component count, complexity, and power and hence significantly improving reliability. Niagara2 breaks new ground by doubling the throughput performance and performance/watt and providing $> 10\times$ improvement in floating point throughput performance as compared to its predecessor, the UltraSPARC T1. The advanced cryptographic support that it provides at high bandwidth enables running secure applications at wire speeds. All these features help Niagara2 enable a new generation of power-efficient fully secure datacenters.

Acknowledgment

The authors acknowledge the contributions of the Niagara2 Design team and other teams inside Sun Microsystems who contributed to the development of Niagara2, and Texas Instruments for fabricating Niagara2 and co-developing the SerDes Macros in collaboration with Sun Microsystems. The authors also acknowledge M. Young, G. Peterson, and F. Desantis for their management support.

References

[1] U. G. Nawathe, M. Hassan, L. Warriner, K. Yen, B. Upputuri, D. Greenhill, A. Kumar, and H. Park, "An 8-core 64-thread 64b power-efficient SPARC SoC," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2007, p. 108.

[2] G. Grohoski, "Niagara2: A highly threaded server-on-a-chip," presented at the 18th Hot Chips Symp., Palo Alto, CA, Aug. 2006.

[3] R. Golla, "Niagara2: A highly threaded server-on-a-chip," presented at the Microprocessor Forum, San Jose, CA, Oct. 2006.

[4] S. Shastry, A. Bhatia, and S. Reddy, "A single-cycle-access 128-entry fully associative TLB for multi-core multi-threaded server-on-a-chip," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2007, p. 410.

[5] A. S. Leon, K. W. Tam, J. L. Shin, D. Weisner, and F. Schumacher, "A power-efficient high-throughput 32-thread SPARC processor," *J. Solid-State Circuits*, vol. 42, no. 1, pp. 7–16, Jan. 2007.

[6] A. S. Leon, J. L. Shin, K. W. Tam, W. Bryg, F. Schumacher, P. Kongetira, D. Weisner, and A. Strong, "A power-efficient high-throughput 32-thread SPARC processor," in *IEEE ISSCC Dig. Tech. Papers*, Feb. 2006, pp. 295–304.

[7] J. M. Hart, K. T. Lee, and D. ChenL. Cheng, C. Chou, A. Dixit, D. Greenley, G. Gruber, K. Ho, J. Hsu, N. G. Malur, and J. Wu, "Implementation of a fourth-generation 1.8-GHz dual-core SPARC V9 microprocessor," *J. Solid-State Circuits*, vol. 41, no. 1, pp. 210–217, Jan. 2006.

[8] H.-T. Ahn and D. J. Allstot, "A low-jitter 1.9 V CMOS PLL for UltraSPARC microprocessor applications," *J. Solid-State Circuits*, vol. 35, no. 3, pp. 450–454, Mar. 2000.

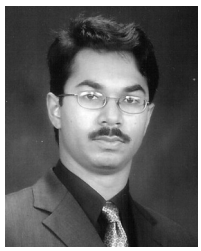
[9] B. Chi and B. Shi, "2/3 divider cell using phase switching technique," *Electron. Lett.*, vol. 37, no. 14, pp. 875–877, Jul. 2001.

[10] M. A. El Sheikh and A. Hafez, "Phase mismatch in phase switching frequency dividers," in *Proc. 15th Int. Conf. Microelectronics*, Dec. 2003, pp. 106–109.

Umesh Gajanan Nawathe received the B.S. degree in electronics and telecommunication from the College of Engineering, University of Poona, India, and the M.S. degree in electrical engineering from the University of Michigan, Ann Arbor.

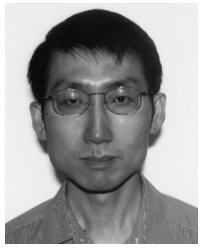
He is a Senior Manager with Sun Microsystems, Santa Clara, CA. Since joining Sun in 2003, he has worked on the Niagara series of microprocessors. His areas of interest include low-power and high-performance circuit design, clocking, design methodology, and process technology. Before joining Sun, he held senior technical and management positions working at MIPS/Silicon Graphics designing microprocessors based on the MIPS architecture, and at Intel prior to that. He has authored or co-authored and presented papers and seminars at the IEEE International Solid-State Circuits Conference (ISSCC), International Symposium of Physical Design (ISPD), and IEEE Santa Clara Valley Solid States Circuits Chapter.

Mr. Nawathe was the recipient of the Sun 2007 Chairman's Award for Innovation.



Mahmudul Hassan received the B.S. degree in electrical engineering with Highest Honors from the Georgia Institute of Technology, Atlanta, GA, in 1997, and the M.S. degree in electrical engineering from Stanford University, Stanford, CA, in 2003.

He joined Texas Instruments, Dallas, TX, in 1997 as an ASIC Designer with technical focus on physical design, implementation and delivery of large high-speed ASICs in the networking and telecom space. In 2003, he moved to the Broadband Communications Group working on SoC designs specializing in clocking, reset, and processor-assisted memory BIST techniques. He joined Sun Microsystems, Sunnyvale, CA, in 2005 to lead efforts on Niagara2 high-speed clocking, domain crossing, global signal distribution and deterministic test methodologies. His interests are in high-speed system design and in solving communication bottlenecks in data processing architectures.



King C. Yen received the B.Eng. degree in electrical engineering from Imperial College, London, U.K., and the M.S.E.E. degree from Stanford University, Stanford, CA, in 1990 and 1994, respectively.

He joined Sun Microsystems, Sunnyvale, CA, in 1998, where he has since worked on floating-point, I/O and analog designs on microprocessors. He is currently analog circuit design lead. Prior to joining Sun, he worked on analog circuit design at Marvell Technology Group Ltd., Chrontel Inc., and Motorola.

Ashok Kumar received the B.S. degree in electrical and electronics engineering in 1990 and the M.S. degree in microelectronics in 1991, both from Birla Institute of Technology and Science, Pilani, India.

He then joined the same institute and taught various electrical, electronics, and microelectronics courses. In 1994, he joined Temic Usha, India, where he performed circuit design and chip integrations for microcontroller chips. In 1998, he joined Motorola and designed microcontrollers for automobile applications, working on various phases of chip design from libraries to full chip integration. He joined Sun Microsystems, Sunnyvale, CA, in 2002 and is currently working as Senior Staff Engineer defining global circuit design methodologies for current CMT microprocessors. His interests are custom circuit design, chip integration and design methodologies.

Aparna Ramachandran received the B.E. degree in electronics and communication engineering from the University of Madras, India, in 1998, and the M.S. degree in electrical engineering from Arizona State University, Tempe, in 2000.

Since joining Sun Microsystems, Sunnyvale, CA, in January 2001, she has been working as a circuit designer for current CMT microprocessors. Her interests are custom circuit design and design methodologies.

David Greenhill received the B.Sc. degree in physics from Imperial College, London, U.K.

He is a Distinguished Engineer with Sun Microsystems, Sunnyvale, CA. He is the Chief Engineer for the Sparc Volume Processors team. This group develops Niagara CPUs and server systems for highly threaded applications. He joined Sun in 1992 and has worked for 15 years on the UltraSparc series of microprocessors. His areas of interest are in circuit design, system design, technology development and design methodology. Before joining Sun, he worked at Inmos on Transputer microprocessors and graphics chip designs. He has authored and co-authored many papers and presentations for the IEEE International Solid-State Circuits Conference (ISSCC) and IEEE JOURNAL OF SOLID-STATE CIRCUITS (JSSC).

Mr. Greenhill has been an active member of the IEEE, serving on the ISSCC program committee, as JSSC Guest Editor, and on conference panels. He was co-chair of the DAC/ISSCC student design contest. He was the recipient of the Sun 2002 Chairman's Award for Innovation.