SPARC64 XII: Fujitsu’s Latest 12-Core Processor for Mission-Critical Servers

Takumi Maruyama, Yasunobu Akizuki, Takekazu Tabata, Kenichi Kitamura, Noriko Takagi, Hiroyuki Ishii, Shingo Watanabe, and Fumihiro Tawa
Fujitsu

The SPARC64 XII 12-core processor, developed for high-performance, mission-critical servers, runs at speeds of up to 4.35 GHz and achieves a peak performance of 417 GIPS and 835 Gflops. SPARC64 XII realizes a 2.3-2.9 times improvement in core performance over the previous-generation SPARC64 X+.

High performance and reliability are key requirements for processors used in UNIX servers. This article provides an overview of the SPARC64 XII chip and describes the processor pipeline. Performance of the chip is compared against the performance of the previous-generation SPARC64 X+. The article also highlights some of the processor’s extensive reliability, availability, and serviceability (RAS) features and server system features.

SPARC64 XII PROCESSOR CHIP OVERVIEW

The SPARC64 XII1 is targeted to achieve high speed and high throughput at the same time. High speed means high single thread performance. It has been achieved by high CPU frequency (up to 4.35 GHz), state-of-the-art out-of-order execution, and rich execution units. High throughput has been mainly achieved by many cores and threads, backed up by strong cache and memory. The SPARC64 XII chip comprises 12 identical cores with a Level-3 (L3) cache (see Figure 1).

The three CPU cores and an 8-Mbyte block of L3 cache are grouped as one last level cache and core unit (LCU). With four LCUs, the SPARC64 XII chip includes a total of 12 cores and 32 Mbytes of L3 cache. The SPARC64 XII chip also has various off-chip interfaces, such as four 16-byte double data rate fourth-generation (DDR4) memory interfaces, multi-port high-speed serial system interfaces for connection to other SPARC64 XII chips, and four PCI Express (PCIe), 8-Bytes/s ports.
The chip is fabricated using a 20-nm CMOS process, and the die size is about 795 mm² with around 5,450 million transistors. The chip runs as fast as 4.25 GHz or 4.35 GHz with high-speed mode enabled, and its peak performance is 417 GIPS and 835 Gflops.

Figure 1. SPARC64 XII die photo.

The major differences between SPARC64 X+ and SPARC64 XII are summarized in Table 1 and explained in the following sections.

Table 1. Differences between SPARC64 X+ and SPARC64 XII.

<table>
<thead>
<tr>
<th></th>
<th>SPARC64 X+</th>
<th>SPARC64 XII</th>
</tr>
</thead>
<tbody>
<tr>
<td>#cores, #threads</td>
<td>16core x 2threads</td>
<td>12core x 8threads</td>
</tr>
<tr>
<td>#instruction pipeline per core</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>Branch Prediction</td>
<td>G-share</td>
<td>Neural-net</td>
</tr>
<tr>
<td>$ structure</td>
<td>L1I: 64KB/core</td>
<td>L1I: 64KB/core</td>
</tr>
<tr>
<td></td>
<td>L1D: 64KB/core</td>
<td>L1D: 64KB/core</td>
</tr>
<tr>
<td></td>
<td>L2: 24MB</td>
<td>L2: 512KB/core</td>
</tr>
<tr>
<td></td>
<td></td>
<td>L3: 8MB/LCUx4LCU</td>
</tr>
<tr>
<td>L1$ coherency</td>
<td>MESI protocol</td>
<td>MEZASI protocol</td>
</tr>
<tr>
<td>RAS</td>
<td>ECC protected integer/floating point registers</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Parity protected internal registers</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ECC protected or duplicated cache</td>
<td></td>
</tr>
<tr>
<td></td>
<td>HW Instruction Retry</td>
<td></td>
</tr>
<tr>
<td>Memory I/F</td>
<td>DDR3-1600 x 64B</td>
<td>DDR4-2400 x 64B</td>
</tr>
<tr>
<td>IO I/F</td>
<td>PCI-GEN3 8lane x 2ch</td>
<td>PCI-GEN3 8lane x 4ch</td>
</tr>
<tr>
<td>Frequency</td>
<td>3.7GHz</td>
<td>4.25GHz (4.35GHz with High Speed mode enabled)</td>
</tr>
<tr>
<td>Process</td>
<td>28nm CMOS</td>
<td>20nm CMOS</td>
</tr>
<tr>
<td>Die size</td>
<td>587mm2</td>
<td>795mm2</td>
</tr>
<tr>
<td>System Cooling</td>
<td>LLC (Liquid Loop Cooling)</td>
<td>VLLC (Vapor Liquid Loop Cooling)</td>
</tr>
</tbody>
</table>
SPARC64 XII PROCESSOR CORE

Each core consists of a Level-2 (L2) cache, one instruction fetch unit, and two instruction pipelines (see Figure 2). Each instruction pipeline executes up to four threads with simultaneous multithreading (SMT) technology. That is, SPARC64 XII executes up to eight threads per core and up to 96 threads per chip, which is three times more than the previous-generation SPARC64 X+.
mechanism, which keeps track of past branch target addresses, as well as branch direction (taken or not-taken) information. The SPARC64 XII adopts neural net branch prediction, unlike G-share branch prediction used in the SPARC64 X+, to achieve higher prediction accuracy.

Instruction Pipeline

Two instruction pipelines are included per core, and four instructions are supplied per cycle from the instruction buffer in the instruction fetch block to each instruction pipeline. These instructions are decoded, issued, executed, and committed independently in each instruction pipeline. Each instruction pipeline has its own resources, except L1 instruction cache and TLB. Because some applications such as databases run the same instruction sequence multiple times, sharing L1 instruction cache between the two instruction pipelines is effective. TLB is shared so that the operating system does not have to take care of the TLB of the other instruction pipeline when TLB invalidation is required. The resources are dynamically shared between threads to maximize performance, as shown in Table 2. When two threads or more are running in a core, most of the resources are fully utilized. Even when only one thread is running, L1 instruction cache is fully utilized.

Table 2. Core resource allocation.

<table>
<thead>
<tr>
<th>#Active threads</th>
<th>L1 $</th>
<th>IB</th>
<th>Rsv. Station</th>
<th>Rename Registers</th>
<th>Execution Units</th>
<th>FP/SP Port</th>
<th>L1 D$</th>
<th>CSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100%</td>
<td>25%</td>
<td>50%</td>
<td>50%</td>
<td>50%</td>
<td>50%</td>
<td>50%</td>
<td>50%</td>
</tr>
<tr>
<td>2</td>
<td>100%</td>
<td>25%</td>
<td>25%</td>
<td>50%</td>
<td>50%</td>
<td>50%</td>
<td>50%</td>
<td>50%</td>
</tr>
<tr>
<td>4</td>
<td>100%</td>
<td>25%</td>
<td>25%</td>
<td>25%</td>
<td>50%</td>
<td>50%</td>
<td>50%</td>
<td>50%</td>
</tr>
<tr>
<td>8</td>
<td>100%</td>
<td>12.5%</td>
<td>50%</td>
<td>12.5%</td>
<td>100%</td>
<td>12.5%</td>
<td>100%</td>
<td>12.5%</td>
</tr>
</tbody>
</table>

The subsequent description of instruction execution focuses on the operation of one instruction pipeline.

Instruction Decode

Four valid instructions can be decoded at the same time. When using SMT, instruction decode of a given thread is performed in the same cycle, and the thread is switched in each cycle. To improve performance, the resources required for execution, such as reservation stations, have been enhanced compared to the previous SPARC64 X+ processor. When performing instruction decode simultaneously, there are no restrictions on instruction type combinations. As long as there are free resources, instructions can be issued.

Instruction Issue and Execution

Decoded instructions are written into reservation stations. Among integer instructions, the addition and subtraction operations and the logical operations can be written into the reservation station for execution (RSE) or reservation station for address generation (RSA). Other integer instructions are written into the RSE only. In addition, the address calculations of load and store instructions are written into the RSA. The floating-point arithmetic instructions and the SIMD arithmetic instructions in High-Performance Arithmetic Computational extensions (HPC-ACE), which were originally developed for SPARC64 VIIIfx,4 are written into the reservation station for floating point (RSF), and the branch instructions are written into the reservation station for BRanch (RSBR). The RSE issues instructions to the two integer arithmetic execution units. The RSA issues instructions to the two address calculation units. The calculated addresses are sent to the TLB and L1 data cache.
The L1 data cache handles load and store instructions. The L1 data cache size is 32 Kbytes and four-way set-associative per instruction pipeline. The data cache can provide the data for the subsequent load instruction without waiting for the address calculation of the store instruction that comes later. Also, the L1 data cache is in a semi-three-port configuration that is simultaneously accessible by two load instructions and one store instruction. The L2 cache is shared between instruction and data. The L2 cache size is 512 Kbytes and eight-way set-associative per core.

Instruction Commit

Results from out-of-order executions are written into different locations: general update buffer (GUB) for integer data, floating-point update buffer (FUB) for floating-point data, and store port for store data. Instruction commit is executed to update architecture registers and memory. The maximum number of instructions to be committed at one time is four. When using SMT, an instruction commit by one thread is performed in one cycle, and threads switch on alternating cycles.

SPARC64 XII PROCESSOR STRUCTURE OUTSIDE THE CORE

The three CPU cores and an 8-Mbyte block of L3 cache are grouped as one LCU, as shown in Figure 3. The LCU provides high L3 cache throughput (96 bytes per cycle) and short latency. The coherency of the L3 cache blocks between LCUs is maintained by hardware, with the new cache protocol called MEZASI, which is enhanced from the standard modified, exclusive, shared, and invalid (MESI) protocol. The additional two cache states (Z and A) are used to represent how a given cache line is shared between LCUs. In addition, the L3 cache block of a given LCU is used as a victim cache of the other LCUs when no thread is running on the LCU. It also supports direct memory access (DMA) write to L3 cache from I/O device, as well as speculative memory access without waiting for other L3 cache access.

![Figure 3. SPARC64 XII structure outside the core.](image)

SYSTEM STRUCTURE

SPARC64 XII has four 16-byte interfaces directly connected to DDR4-2400 DIMMs. Thanks to DDR4 and memory access efficiency improvements, this provides 153 Gbytes/s peak and 128...
Gbytes/s sustained memory throughput, almost twice that of the previous SPARC64 X+ processor’s 65 Gbytes/s with DDR3-1600 DIMMs. The chip also includes four PCIe GEN3 ports, which is double the number of ports in SPARC64 X+.

SPARC64 XII has four chip interconnect ports. Two ports are used to connect two SPARC64 XII processors with peak throughput of 50 Gbytes/s. The remaining two ports are used to connect to crossbar (XB) chips for communication with other SPARC64 XII processors. Up to 32 SPARC64 XII CPUs (3,072 threads) can be connected together with directory-based cache coherency to reduce latency and minimize the frequency of snoop transactions in large configurations.

The processor chips are used in Fujitsu’s latest UNIX server, called SPARC M12. The SPARC M12 system has a vapor and liquid loop cooling (VLLC) unit to cool down SPARC 64 XII processors. VLLC is evaporative cooling, taking heat away when liquid changes to vapor. The VLLC structure is shown in Figure 4. The pumps circulate coolant in the VLLC system, and the radiator dissipates the heat absorbed by the cooling plate into the air. VLLC has achieved twice the cooling performance compared with the liquid loop cooling unit used in the previous M10 systems.

![Figure 4. The VLLC unit.](image)

**SPARC64 XII PROCESSOR RAS**

Similar to SPARC64 X+, SPARC64 XII has robust RAS capabilities. Both the tags and data of all caches are either error-correcting code (ECC) protected or duplicated with parity protection. Both integer and floating-point architecture registers are ECC protected, with 1-bit error correction. Other internal registers and data paths are parity protected, and ALUs are either parity or residue protected. Figure 5 graphically shows RAS coverage of the SPARC64 XII chip. Most of the chip area is gray, which indicates 1-bit error is correctable. Dark gray indicates 1-bit error is detectable to prevent data corruption. White indicates 1-bit error harmless. SPARC64 XII guarantees data integrity through its RAS features.

**Hardware Instruction Retry**

SPARC64 XII implements an instruction retry mechanism for correcting single-bit errors that occur in the registers and ALUs. When an error is detected, all instructions that are currently being executed are cancelled. Because all internal states before the commit stages can be discarded, the programmable resources will see the results of only those instructions that have completed execution without causing errors. The instruction retry hardware re-executes the instruction that caused the error alone, to maximize the possibility of successful execution. If the instruction commits successfully, the processor automatically resumes normal execution. During the process, software intervention is not needed.
PERFORMANCE

SPARC64 XII has achieved much higher performance than SPARC64 X+, as shown in Figure 6. The comparison is done with industry-standard CPU benchmarks and Fujitsu’s internal OLTP benchmark. The vertical axis shows performance relative to SPARC64 X+ running at 3.7 GHz. The single thread performance is equal to or greater than that of SPARC64 X+; the core performance is 2.3-2.9 times, and the chip performance is 1.8-2.2 times that of SPARC64 X+.

Performance Instrumentation and Cycle Accounting

It is important for a processor to have a mechanism to gather information to identify performance bottlenecks of a given program. SPARC64 XII has special registers called performance control registers (PCRs) and performance instrumentation counter registers (PICs). Performance information, such as instruction statistics, cache events, and bus transaction events, can be measured through PCRs and PICs.
Cycle accounting is a method used to analyze performance bottlenecks. The number of CPU cycles to execute an instruction is divided into time spent in various execution states, such as executing instructions, waiting for a memory access, and waiting for execution to complete. SPARC64 XII executes instructions out of order, and instructions are thoroughly mixed together (one instruction may be waiting for data from memory, another executing a floating-point division, for example). Cycle accounting classifies cycles by the number of instructions committed. If no instruction is committed at a given cycle, the reason of “no commit” is measured.

Processor Core Performance

SPARC64 XII’s core performance of integer benchmarks is analyzed with cycle accounting, as shown in Figure 7. The vertical bars show average instruction execution time per core of SPARC64 X+ and SPARC64 XII. The average performance improvement is 2.33 times (which is more than 133 percent), while degree of performance improvement varies depending on each benchmark. Overall execution time of SPARC64 XII is less than half that of SPARC64 X+ thanks to its dual instruction pipeline structure. Also, waiting time for the L3/L2 cache access is reduced significantly due to the three-level cache hierarchy of SPARC64 XII, compared to the two levels of SPARC64 X+.

![Figure 7. Core instruction execution time.](image)

Further detailed analysis results identifying the cause of core performance improvement from SPARC64 X+ are shown in Figure 8. The eight-way SMT (dual pipeline plus four-way SMT pipeline) design of SPARC64 XII improves performance by 111 percent, compared to the two-way SMT of SPARC64 X+. The frequency increase from 3.7 GHz to 4.35 GHz improves performance by 10 percent. The three-level cache hierarchy structure increases the performance by 7 percent.
SUMMARY

SPARC64 XII is Fujitsu’s 12th SPARC processor and was designed for Fujitsu’s latest UNIX server, M12. The processor enhancements are to increase throughput, while keeping its high single thread performance and its robust RAS features. SPARC64 XII measured results have shown 2.3-2.9 times the core performance of SPARC64 X+, thanks to a dual pipeline structure, increased multithread capability, a processor frequency increase, and other enhancements.

Figure 8. SPARC64 XII core performance improvement breakdown.

ACKNOWLEDGMENTS

It is an honor to present the SPARC64 XII processor on behalf of the SPARC64 XII processor design team.

REFERENCES

ABOUT THE AUTHORS

Takumi Maruyama is a senior director at Fujitsu. He started working on the first SPARC64 processor design in 1993 and has been involved in the development of the SPARC64 GP, V, VI, VII, VIIIIfx, IXfx, X, X+, and XII processors. He has a bachelor’s degree in Mathematical Engineering and Instrumentation Physics from the University of Tokyo. Contact him at takumi_maruyama@jp.fujitsu.com.

Yasunobu Akizuki is an engineer in the Processor Development Division in the Advanced System Research & Development Unit at Fujitsu. His technical interests include instruction control units. He has a bachelor’s degree in computer science from the University of Electro-Communications. Contact him at akizuki@jp.fujitsu.com.

Takekazu Tabata is an engineer at Fujitsu. His research interests include microprocessor architecture. He has a master’s degree in information engineering from the Hiroshima City University. Contact him at tabata.takekazu@jp.fujitsu.com.

Kenichi Kitamura is an engineer at Fujitsu. He started working on the SPARC64 processor design in 2006 and has been involved in the development of many SPARC64 processors. He has a Master of Informatics and Network from the University of Electro-Communications. Contact him at kitamura_ken1@jp.fujitsu.com.

Noriko Takagi is an engineer in the Processor Development Division in the Advanced System Research & Development unit at Fujitsu. Her technical interests include cache architecture for processors. She has a master’s degree in information physics and computing from the University of Tokyo. Contact her at takagi.noriko@jp.fujitsu.com.

Hiroyuki Ishii is an engineer in the Processor Development Division in the Advanced System Research & Development unit at Fujitsu. His research interests focus on cache architecture for processors. Ishii has a master’s degree in electrical and electronic engineering from Muroran Institute of Technology. Contact him at hiroyuki.ishii@jp.fujitsu.com.

Shingo Watanabe is an engineer at Fujitsu. His research interests include analysis of processor performance and architecture. Watanabe has a master’s degree in information engineering from the Kyushu Institute of Technology. Contact him at snabe@jp.fujitsu.com.

Fumihiro Tawa is an engineer in the Server Engineering Development Division at Fujitsu and has been involved in liquid cooling unit development of LLC and VLLC. Contact him at tawa.fumihiro@jp.fujitsu.com.