



Chip Multi-threading and Sun's Niagara-series

Jinghe Zhang

Dept. of Computer Science

College of William and Mary

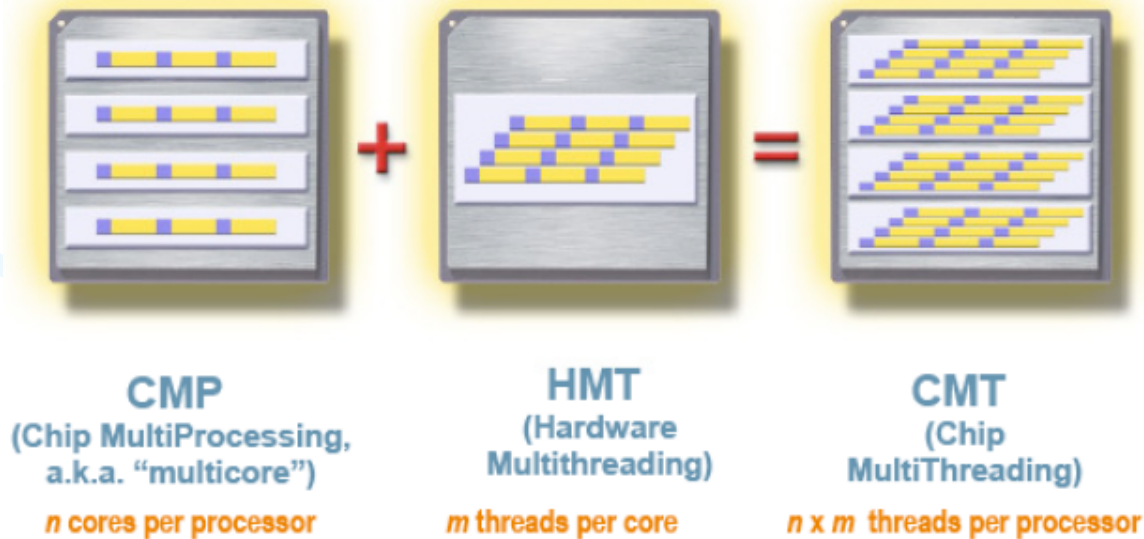


Outline

- **Why CMT processors?**
- **Sun's Microprocessor**
- **Conclusions**

What is Chip Multi-Threading?

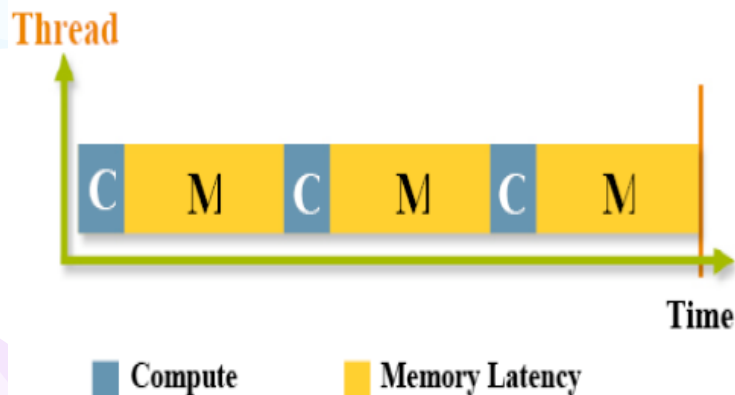
- CMP: Chip that provide multiprocessing, called Multi-core Architecture. There are many cores per processor.
- HMT: Hardware Multithreading. There are many threads per core.
- CMT: Chip Multithreading has support to CMP Technology and HMT Technology. There are many cores with multiple threads per processor.



Why CMT processors?

- Single Threading

Typical Utilization of Processor: 15-25%



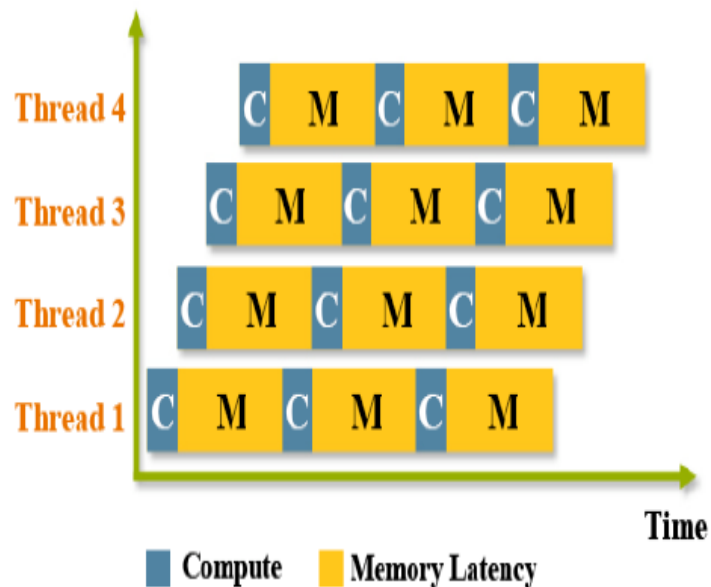
- **For a single thread**

- Memory is the bottleneck to improving performance
- Commercial server workloads exhibit poor memory locality
- Only a modest throughput speedup is possible by reducing compute time
- Conventional single-thread processors optimized for ILP have low utilizations

Why CMT processors?

- Multi-Threading

Utilization: Up to 85%*



* based on example of UltraSPARC T1

- **With many threads**

- It's possible to find something to execute every cycle
- Significant throughput speedups are possible
- Processor utilization is much higher

The slide features a decorative background on the left side with a light green balloon at the top, a light blue balloon in the middle, and a light purple balloon at the bottom. Yellow streamers and small triangles are scattered around the balloons. The main title is centered at the top in a large, bold, teal font.

Achievements and obstacles in improving single thread performance

- Dramatic gains have been achieved in single-threaded performance in recent years
- Used a variety of micro-architectural techniques
 - superscalar issue, out-of-order issue, on-chip caches & deeper pipelines
- Recent attempts to continue leveraging these techniques have not led to demonstrably better performance
- Power and memory latency considerations introduce increasingly insurmountable obstacles to improving single thread performance
 - Several high frequency designs have been abandoned

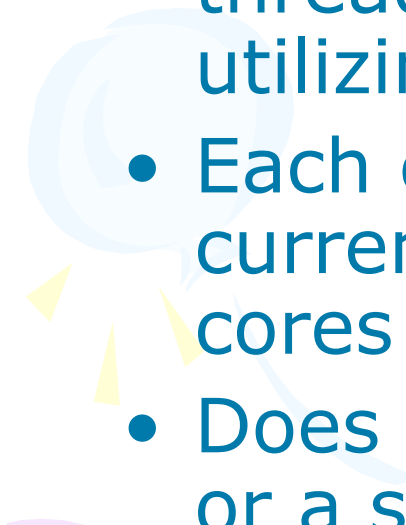



Target: Commercial Server Applications

- Server workloads are characterized by:
 - High thread-level parallelism (TLP)
 - Low instruction-level parallelism (ILP)
- Limited opportunity to boost single-thread performance
 - Majority of CPI is a result of off-chip misses
- Couple high TLP in the application domain with support for multiple threads of execution on a processor chip
- Chip Multi-threaded (CMT) processors come to rescue!



Engineering Solution

- Instead of optimizing each core, overall goal was running as many concurrent threads as possible maximizing and utilizing each core's pipeline
 - Each core is less complex than those of current high end processor, allowing 8 cores to fit on the same die.
 - Does not feature out-of-order execution, or a sizable amount of cache
 - Each core is a barrel processor
- 
- 

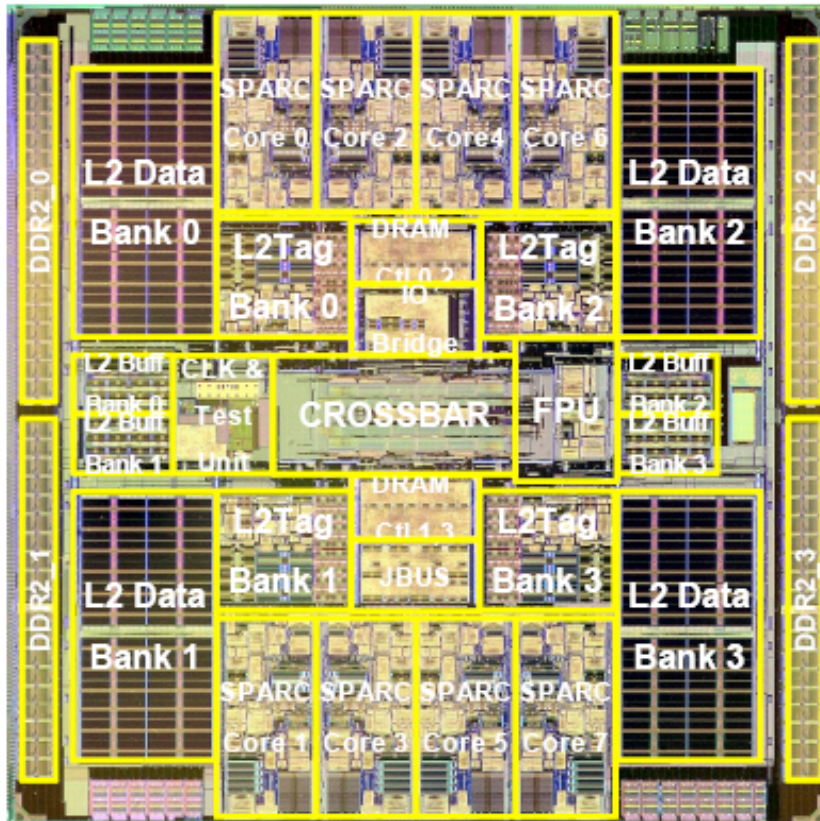
A decorative graphic on the left side of the slide features three balloons: a green one at the top, a light blue one in the middle, and a purple one at the bottom. Each balloon has a string and several small yellow triangular shapes radiating from it, resembling sunbeams or confetti. The balloons are arranged in a vertical line, with the green one at the top, the blue one in the middle, and the purple one at the bottom. The strings of the balloons are also visible, extending downwards.

Sun's Microprocessors

- Ultrasparc T line

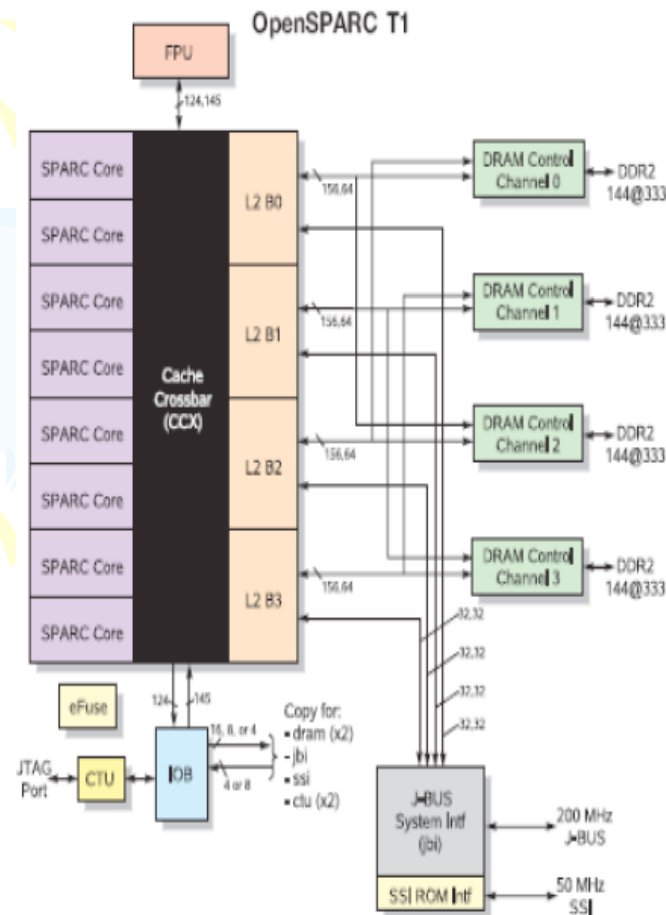
- Ultrasparc T1 Niagara 2005 90nm
- Ultrasparc T2 Niagara2 2007 65nm

UltraSPARC T1: Overview



- 90 nm technology, included 8 cores, 32 threads, only dissipate 70w.
- Maximum clock rate of 1.2 GHz.
- Cache, DRAM Channel shared across cores.
- Shared L2 Cache.
- On-die Memory controllers.
- Crossbar Switch

T1's architecture



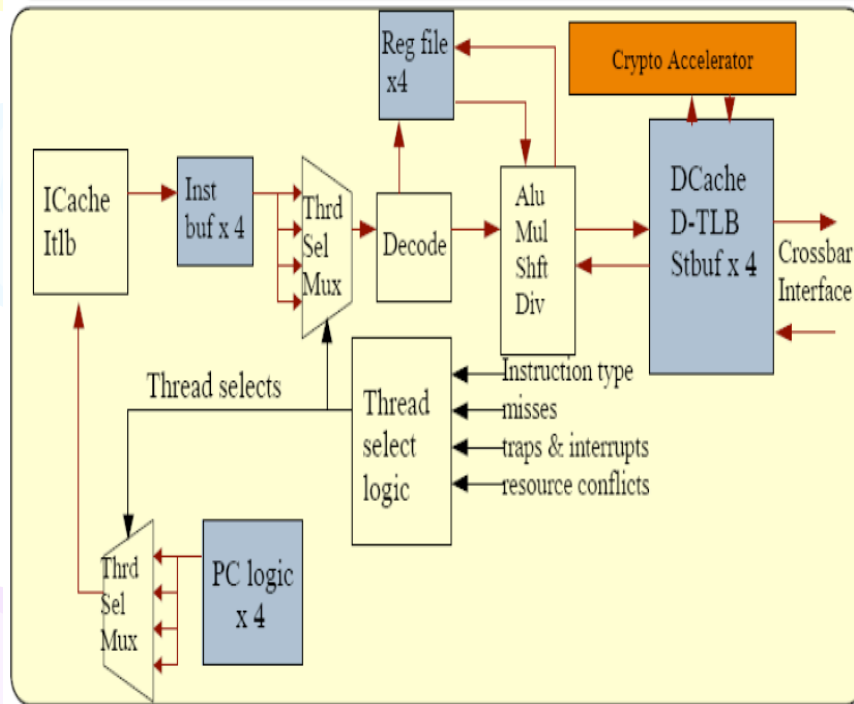
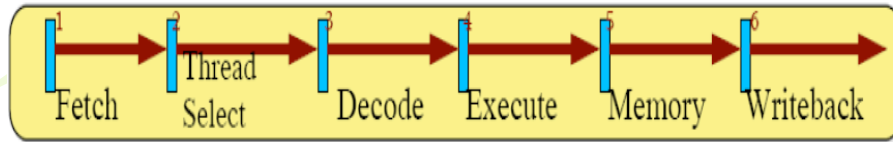
- 8 SPARC V9 CPU cores, with 4 threads per core, for a total of 32 threads
- Fine-grained thread scheduling
- One shared floating-point unit for eight cores
- 132 Gbytes/sec crossbar interconnect for on-chip communication
- 16 KB of L1 I-cache, 8 KB of L1 D-cache per CPU core
- 3 Mbytes of secondary (Level 2) cache – 4 way banked, 12 way associative shared by all CPU cores
- 4 DDR-II DRAM controllers – 144-bit interface per channel, 25 GBytes/sec peak total bandwidth



Extra: Different types of Multi-threading

- Coarse-grained multithreading, or switch-on-event multithreading, is when a thread has full use of the CPU resource until a long-latency event such as miss to DRAM occurs, in which case the CPU switches to another thread.
- Fine-grained multithreading is sometimes called interleaved multithreading; in it, thread selection typically happens at a cycle boundary.
- An approach that schedules instructions from different threads on different functional units at the same time is called simultaneous multithreading (SMT) or, alternately, horizontal multithreading.

SPARC core

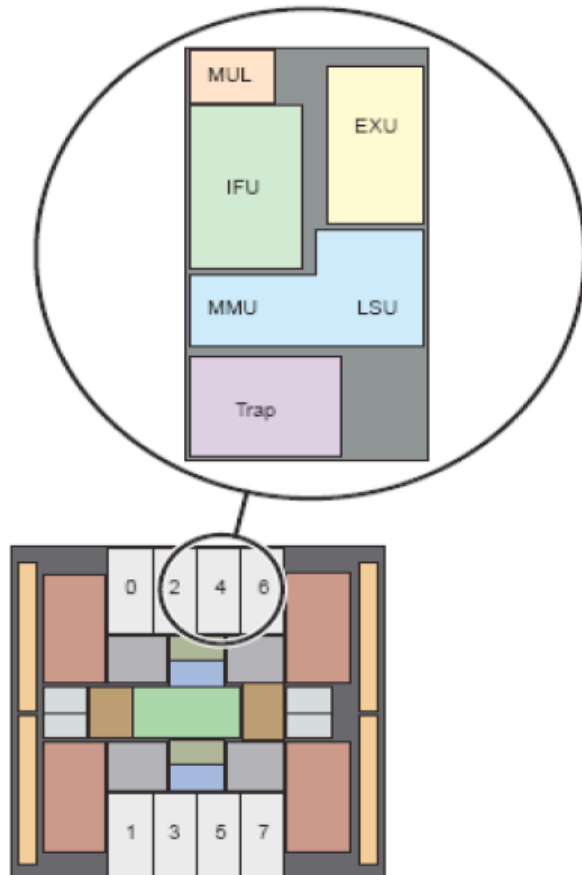


Each SPARC core has hardware support for four threads. The four threads share the instruction, the data caches, and the TLBs.

Each SPARC core has simple, in-order, single issue, six stage pipeline. These six stages are:

1. Fetch
2. Thread Selection
3. Decode
4. Execute
5. Memory
6. Write Back

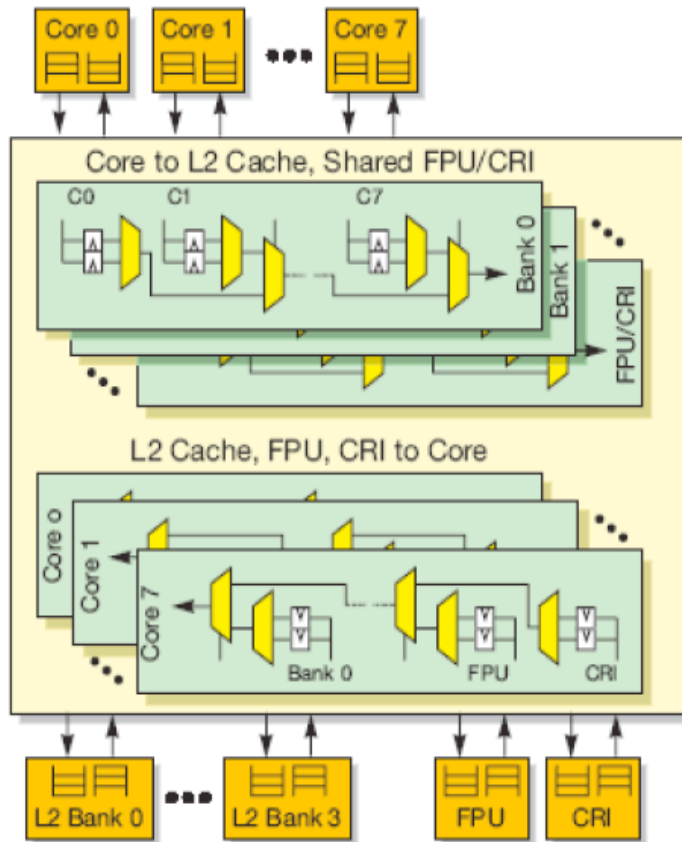
SPARC Core Units



Each SPARC core has the following units:

- Instruction fetch unit (IFU) includes the following pipeline stages – fetch, thread selection, and decode. The IFU also includes an instruction cache complex.
- Execution unit (EXU) includes the execute stage of the pipeline.
- Load/store unit (LSU) includes memory and writeback stages, and a data cache complex.
- Trap logic unit (TLU) includes trap logic and trap program counters.
- Stream processing unit (SPU) is used for modular arithmetic functions for crypto.
- Memory management unit (MMU).
- Floating-point frontend unit (FFU) interfaces to the FPU.

CPU-Cache Crossbar



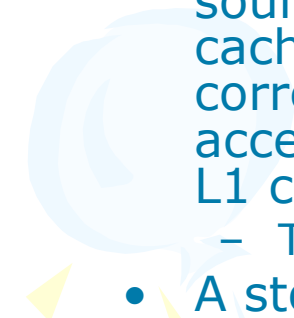
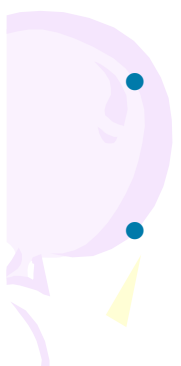
The eight SPARC cores, the four L2-cache banks, the I/O Bridge, and the FPU all interface with the crossbar.

The CPU cache crossbar (CCX) features include:

- Each requester queues up to two packets per destination.
- Three stage pipeline – request, arbitrate, and transmit.
- Centralized arbitration with oldest requester getting priority.
- Core-to-cache bus optimized for address plus double word store.
- Cache-to-core bus optimized for 16-byte line fill. 32-byte Icache line fill delivered in two back-to-back clocks.



Cache Coherence Protocol

- The L1 caches are write through, with allocate on load and no-allocate on stores. L1 lines are either in valid or invalid states. The L2 cache maintains a directory that shadows the L1 tags.
 - A load that missed in an L1 cache (load miss) is delivered to the source bank of the L2 cache w/ its replacement way from the L1 cache. There, the load miss address is entered in the corresponding L1 tag location of the directory, the L2 cache is accessed to get the missing line and data is then returned to the L1 cache.
 - The directory maintains a sharers list at L1-line granularity.
 - A store from a different or same L1 cache will look up the directory and queue up invalidates to the L1 caches that have the line. Stores do not update the local caches until they have updated the L2 cache. During this time, the store can pass data to the same thread but not to other threads;
 - A store attains global visibility in the L2 cache.
 - The crossbar establishes memory order between transactions from the same and different L2 banks, and guarantees delivery of transactions to L1 caches in the same order.
 - Direct memory access from I/O devices are ordered through the L2 cache.
- 
- 



What to expect next?

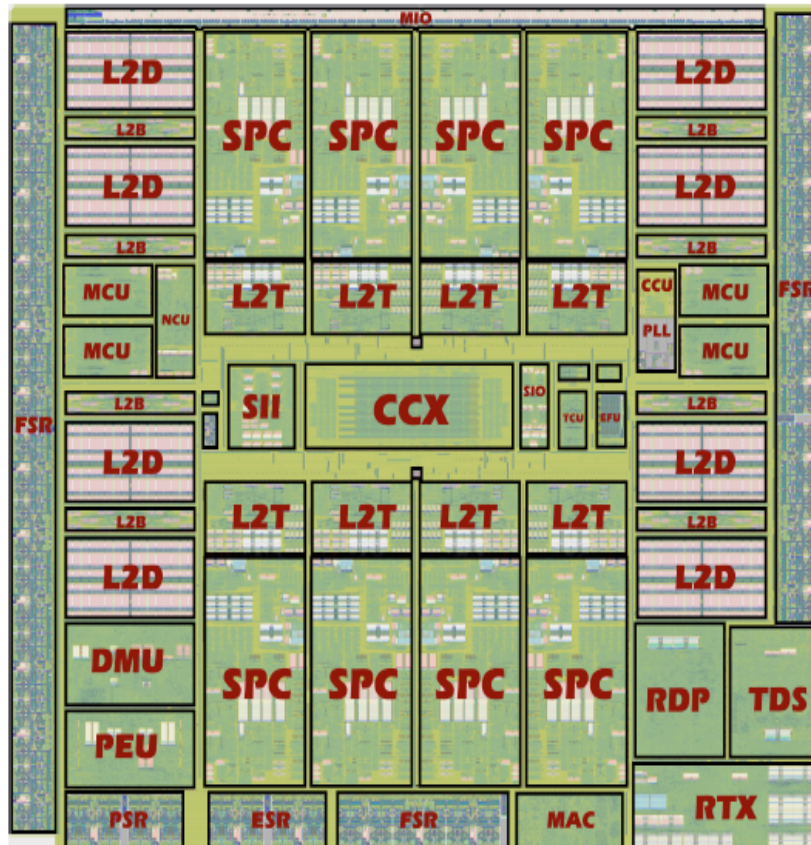




Niagara 2 Chip Goals

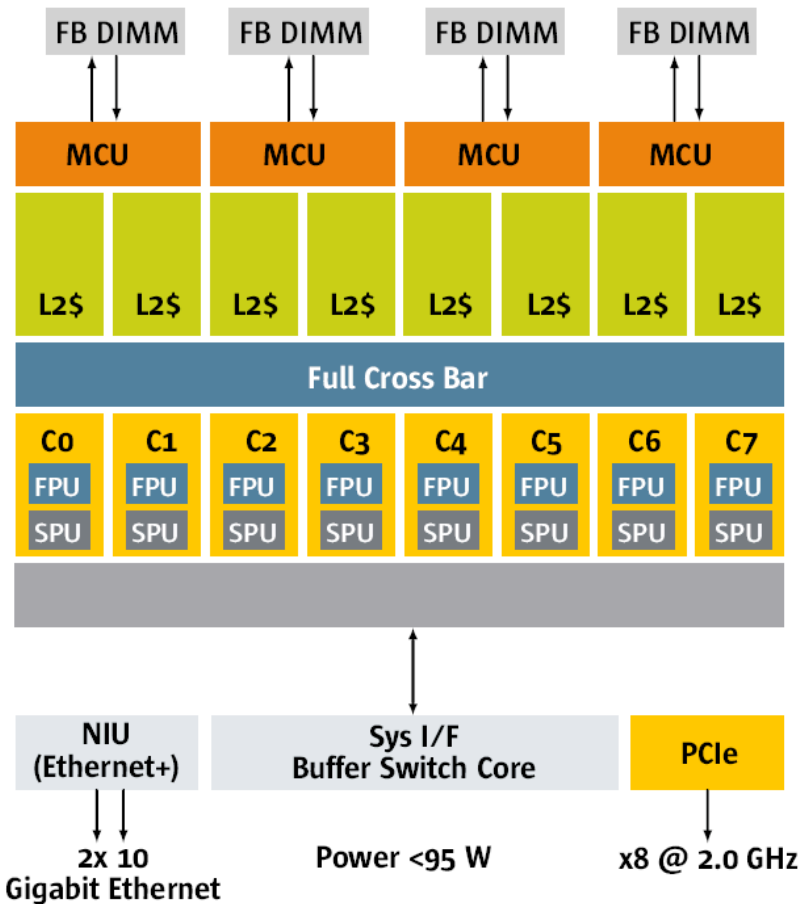
- **Goals of the T2 project were:**
 - Double UltraSparc T1's throughput and throughput/watt
 - Improve UltraSparc T1's FP single-thread (T1 was unable to handle workloads with more than 1-3% FP instructions) throughput performance
 - Minimize required area for these improvements
- **Considered doubling number of UltraSparc T1 cores**
 - 16 cores of 4 threads each
 - Takes too much die area
 - No area left for improving FP performance

“Niagara 2 Opens the Floodgates”



- 8 Sparc cores, 8 threads each
- Shared 4MB L2, 8-banks, 16-way associative
- Four dual-channel FBDIMM memory controllers
- Two 10/1 Gb Ethernet ports w/onboard packet classification and filtering
- One PCI-E x8 1.0 port
- 711 signal I/O, 1831 total

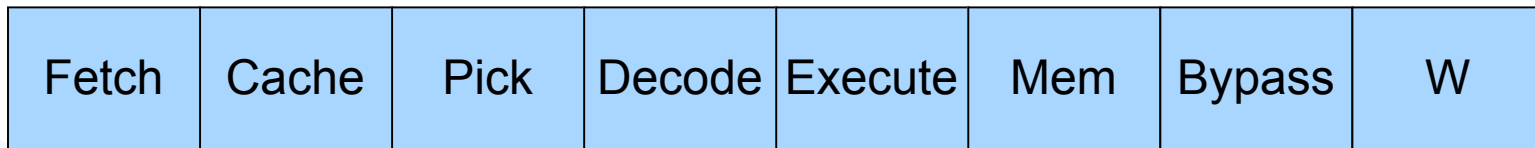
T2's architecture



- 8 Fully pipelined FPUs
- 8 SPUs
- 2 integer ALUs per core, each one shared by a group of four threads
- 4MB L2 Cache (8-banks, 16-way associative)
- 8 KB data cache and 16 KB instruction cache
- Two 10Gb Ethernet ports and one PCIe port

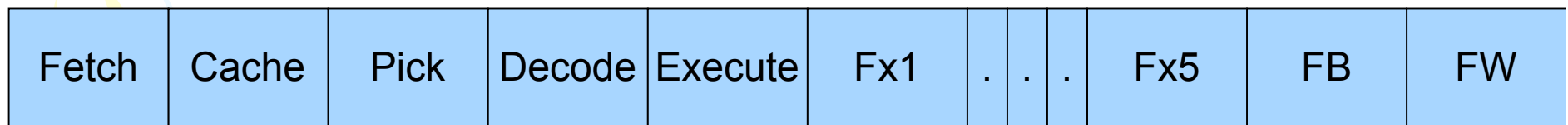
Efficient in-order single issue pipeline

- Eight-stage integer pipeline



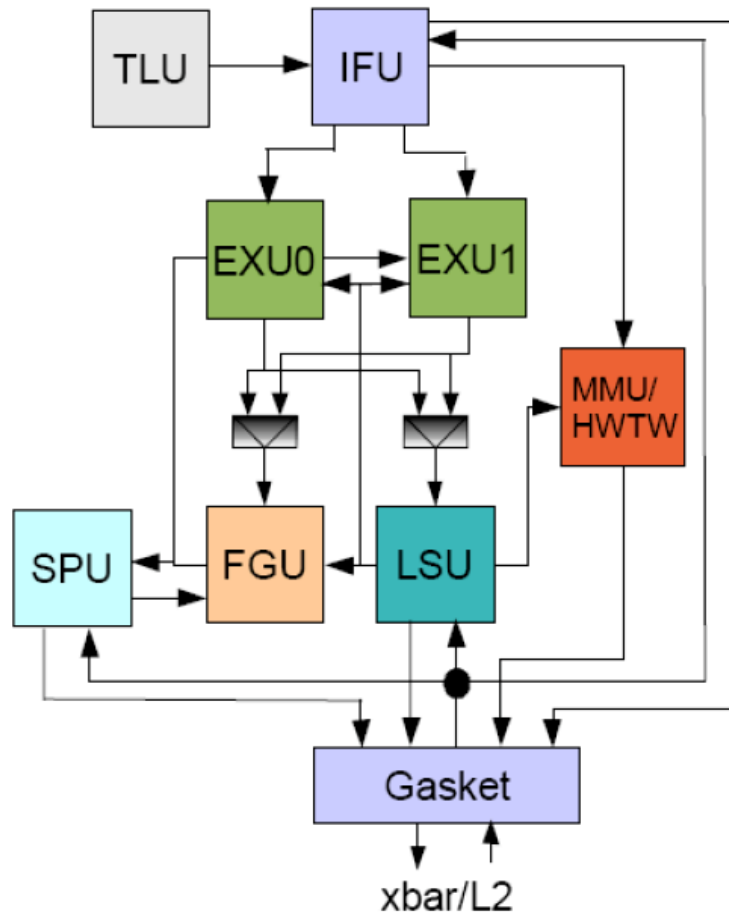
- Pick is for selecting 2 threads for execution (Added this stage for T2)
- In the bypass stage, the load/store unit (LSU) forwards data to the integer register files (IRFs) with sufficient write timing margin. All integer operations pass through the bypass stage

- 12-stage floating point pipeline



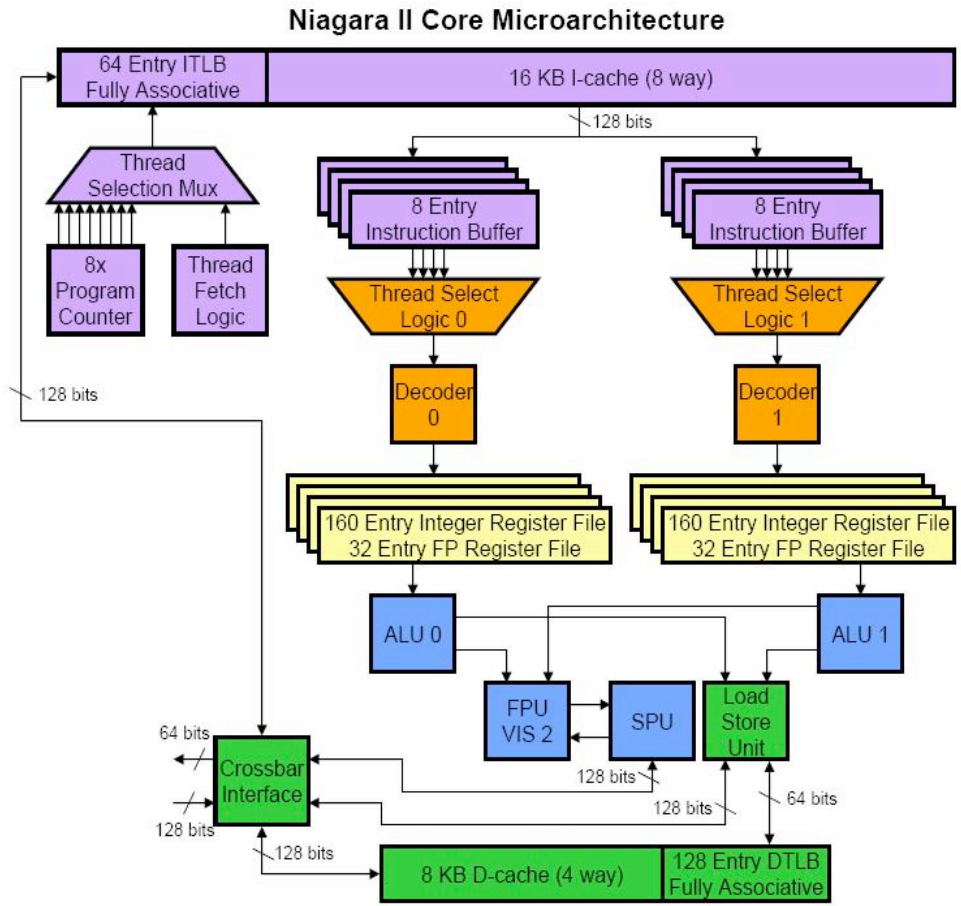
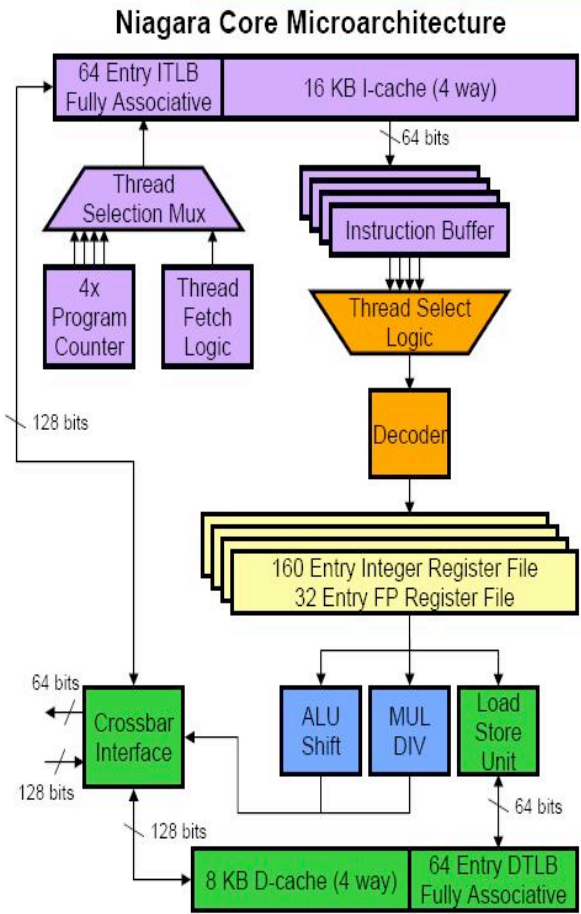
- › 6-cycle latency for dependent FP ops
- › Integer multiplies are pipelined between different threads. Integer multiplies block within the same thread.
- › Integer divide is a long latency operation. Integer divides are not pipelined between different threads.

Core Units



- EXU0/1 – Integer Execution Units
 - 4 threads share each unit
 - Executes one integer instruction/cycle
- FGU – Floating/Graphics Unit
- **The most visible changes: the dual execution pipelines and the new FGU**

Comparison



Comparison: Main Features

Series	UltraSPARC T1/T2		
Models	UltraSPARC T1	UltraSPARC T2	
Nr. of cores	8 cores	8 cores	
Impl. of the cores	Monolithic	Monolithic	
Architecture	SPARC V9	SPARC V9	
Cores	Scalar integer FX cores	Dual-issue FX/FP cores	
Introduction	11/2005	2007	
Technology	90 nm	65 nm	
Die size	379 mm ²	342 mm ²	
Nr. of transistors	279 mtrs.	n.a.	
f_c [GHz]	1.2	1.4	
L2	Size/allocation	3 MB/shared	4 MB/shared
	Implementation	On-die	On-die
L3	—	—	
I/O-bus	JBus (3.2 GB/s)	JBus (3.2 GB/s)	
Interconnection NW	Bandwidth: >200 GB/s	Full 8*9 crossbar switch	
Memory controller	4-channels, on-die, 400 MT/s	4-channels, on-die, 400 MT/s	
Memory bandwidth	25.6 GB/s	42.7 GB/s	
TDP [W]	63	72 (est.)	
Multithreading	4-way/core	8-way/core	



Conclusions

- CMT designs support many HW threads via efficient sharing of on-chip resources
- CMT designs are rapidly evolving
 - There will be more and more generations in the future
- CMT processors are a perfect match for thread-rich commercial applications
- Need to begin exploiting Speculative parallelism approaches for applications with limited TLP
- Resource sharing and resultant inter-thread interactions gives rise to many design challenges
 - Speculative prefetching, request prioritization, hot sets, hot banks, BW limitations to name but a few...



Conclusions

- Niagara 1: a revolutionary 32-way multithreaded processor with breakthrough performance levels, dramatically reduces power, cooling and space consumption
- Niagara 2: based on Niagara 1, is the industry's first "system on a chip," packing the most cores and threads of any general-purpose processor available, and integrating all the key functions of a server on a single chip

CMT processors present an exciting range of opportunities and challenges



- Questions?

