



OpenSPARC™

# OpenSPARC Slide-Cast

In 12 Chapters

Presented by OpenSPARC  
designers, developers, and  
programmers

- to guide users as they develop  
their own OpenSPARC designs  
and
- to assist professors as they  
teach the next generation

This material is made available under  
Creative Commons Attribution-Share 3.0 United States License





**OpenSPARC™**

## Chapter Four

# OPENSPARC T2 OVERVIEW

**Denis Sheahan**  
**Distinguished Engineer**  
**Niagara Architecture Group**  
**Sun Microsystems**



# Agenda

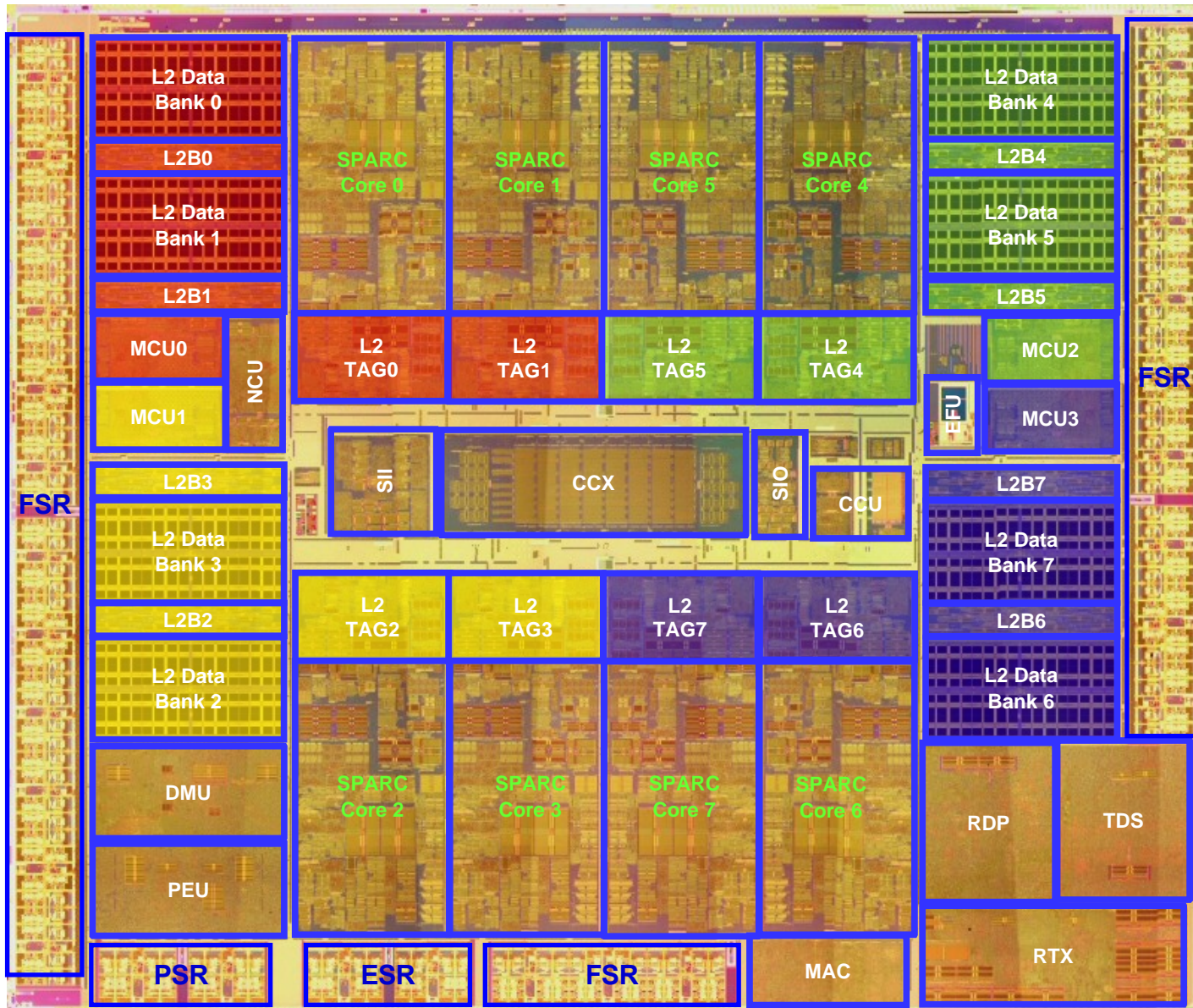
- Chip overview
- SPARC core
  - > Execution Units
  - > Power
  - > RAS
- Crossbar
- L2
- Summary

# OpenSPARC T2 Chip Goals

- Double throughput versus OpenSPARC T1
  - > Doubling cores versus increasing threads per core
  - > Utilization of execution units
- Improve throughput / watt
- Improve single-thread performance
- Improve floating-point performance
- Maintain SPARC binary compatibility

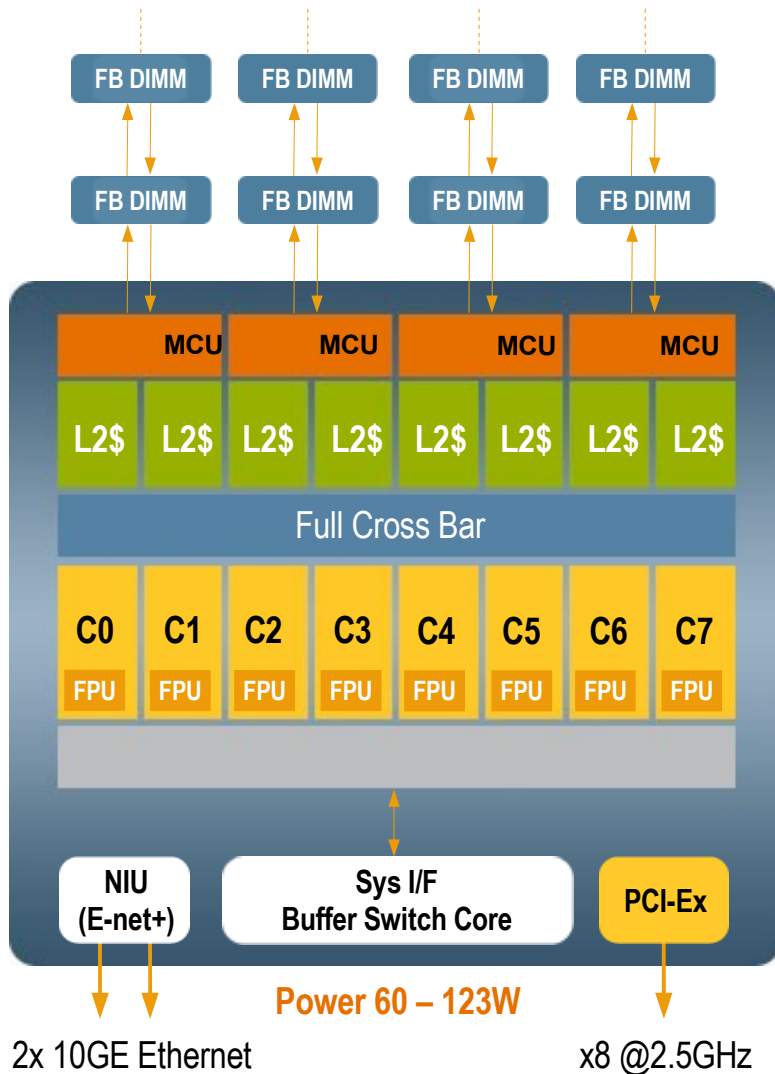
# UltraSPARC T2 Overview

- 8 SPARC cores, 8 threads each, 64 threads total
- Shared 4MB L2, 8 banks, 16 way associative
- Four dual-channel FBDIMM memory controllers
- Full 8x9 crossbar connects cores to L2 banks / SIU and vice versa
- SIU connects I/O to memory



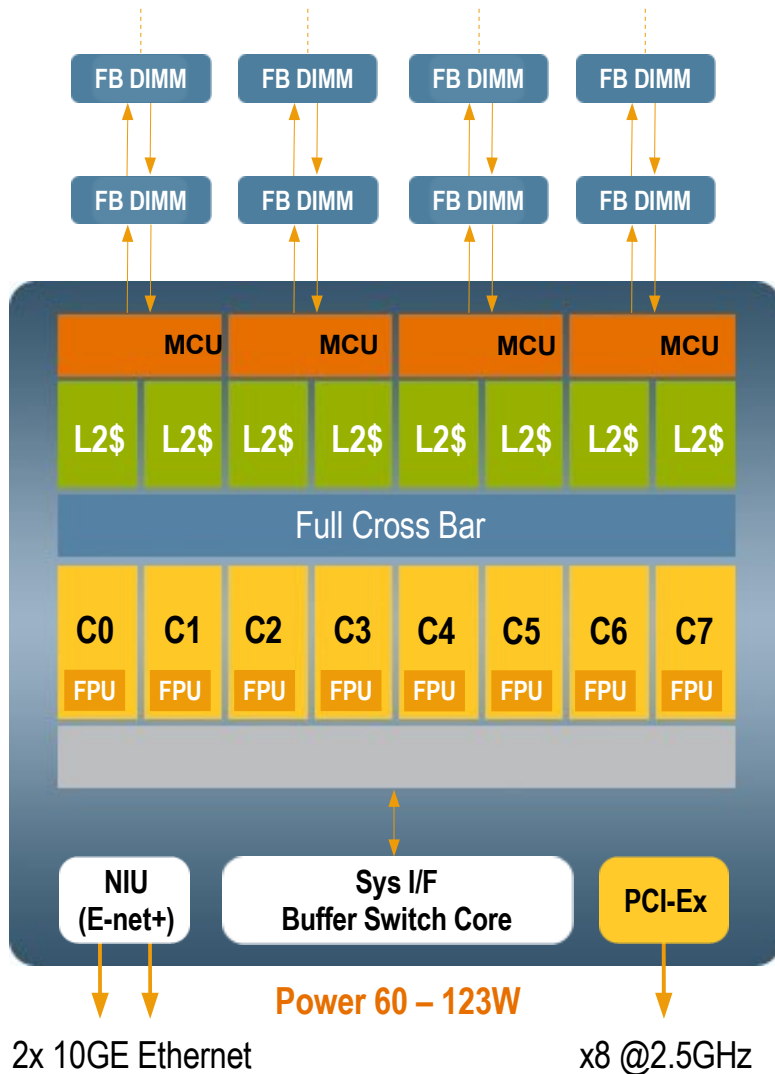
UltraSPARC T2 Die Photo

# UltraSPARC<sup>®</sup> T2 Processor: True System On a Chip



- Up to 8 cores @ 1.2 /1.4GHz
- Up to 64 threads per CPU
- Huge Memory Capacity
  - > Up to 512GB memory
  - > Up to 64 Fully Buffered Dimms
- High Memory Bandwidth
  - > 2.5x memory BW = 60+GB/S
- 8x FPUs, 1 fully pipelined floating point unit/core
- 4MB L2\$ (8 banks) 16 way
- Security co-processor / core
  - > DES, 3DES, AES, RC4, SHA1, SHA256, MD5, RSA to 2048 key, ECC, CRC32

# UltraSPARC<sup>®</sup> T2 Processor: True System On a Chip

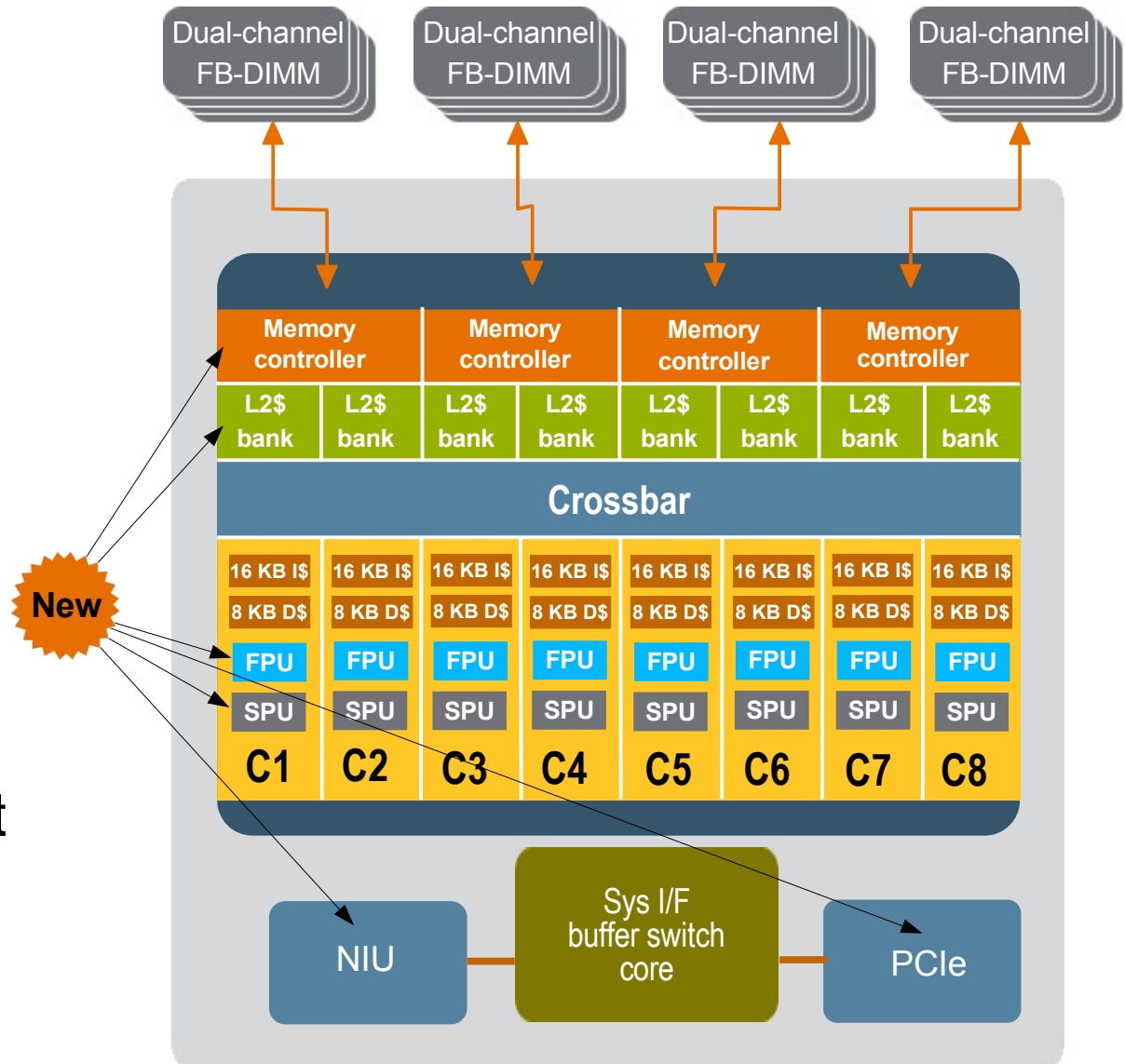


- Up to 8 cores @ 1.2 /1.4GHz
- Up to 64 threads per CPU
- Huge Memory Capacity
  - > Up to 512GB memory
  - > Up to 64 Fully Buffered Dimms
- High Memory Bandwidth
  - > 2.5x memory BW = 60+GB/S
- 8x FPUs, 1 fully pipelined floating point unit/core
- 4MB L2\$ (8 banks) 16 way
- Security co-processor / core
  - > DES, 3DES, AES, RC4, SHA1, SHA256, MD5, RSA to 2048 key, ECC, CRC32

# UltraSPARC T2 Architecture

A true system on a chip

- Up to 8 SPARC cores @ 1.0–1.4 GHz
  - > Up to 64 total threads
  - > 4-MB, 16-way, 8-bank L2\$
- 1 floating-point unit per core
- 1 SPU (crypto) per core
- FB-DIMM 1.0 support
- 8-lane PCI Express 1.0 bus interface
- 2 x 1/10 Gb on-chip Ethernet
- Power: < 95 W (nominal)



# UltraSPARC T2 “Zero Cost” Security



- One crypto unit integrated per core (eight total)
- Supports the ten most common ciphers and secure hashing functions
- Composed of two independent sub-units that operate in parallel
  - > Modular Arithmetic Unit
  - > Cipher/Hash Unit



# Integrated Multithreaded 10 GbE

- Dual, multithreaded, 10 GbE (XAUI)
  - > Up to 4X the performance of current network interface cards
  - > 16 Rx and Tx DMA channels for virtualization
- Limited classification
  - > Classified at layer 2 ,3 and 4 into Rx DMA buffer to match the flow
- Benefits
  - > Eliminates network I/O bottlenecks
  - > Enables faster network access



# Integrated Floating Point Unit

- Each UltraSPARC T2 core has its own Floating Point Unit
- Fully-pipelined (except divide/sqrt)
  - > Divide/sqrt in parallel with add or multiply operations of other threads
- Full VIS 2.0 implementation
- FPU performs integer multiply, divide, population count



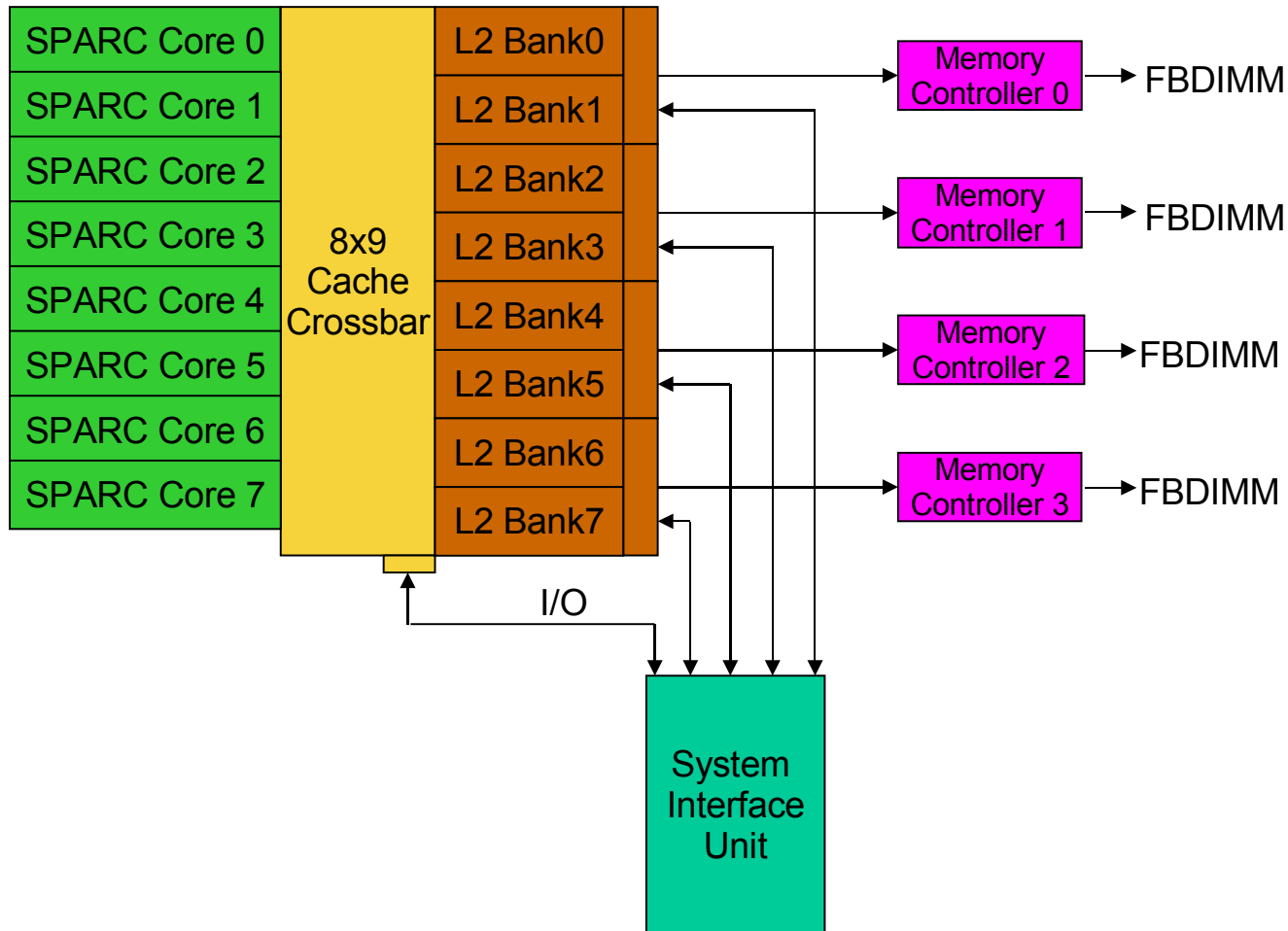
# UltraSPARC T2: 7 World Records

Built on a heritage of network throughput

- Standard performance benchmarks
  - > SPECint\_Rate2006 (single chip)
  - > SPECfp\_Rate2006 (single chip)
  - > Web Performance: SPECweb2005
  - > Unix Java VM (single socket): SPECjbb2005
  - > Java App Server: SPECjAppServer2004 (dual node)
  - > Unix ERP Platform: Single-socket  
SAP SD-2 Tier
  - > OLTP Platform: Database Tier  
SPECjAppServer2004 Dual Node Result



# OpenSPARC T2 Block Diagram



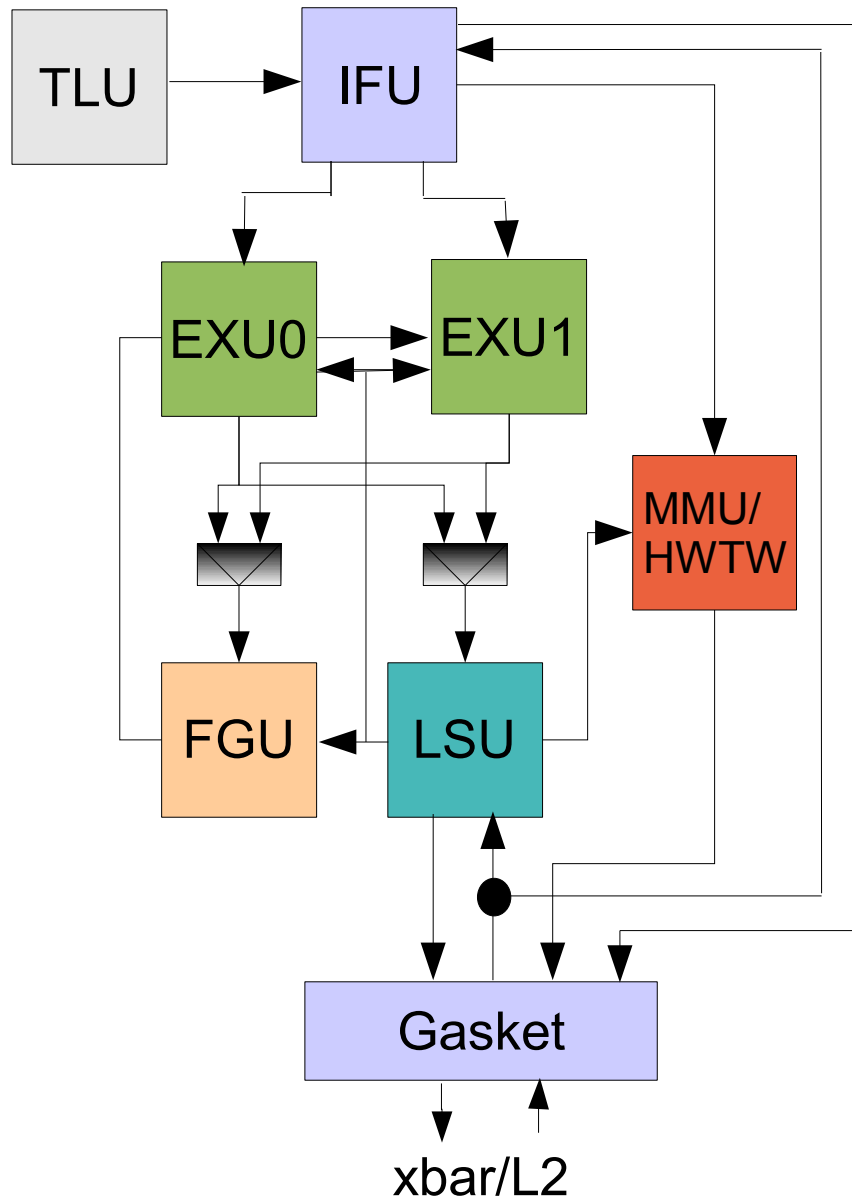
# OpenSPARC T1 to T2 Core Changes

- Increase threads from 4 to 8 in each core
- Increase execution units from 1 to 2 in each core
- Floating-point and Graphics Unit in each core
- New pipe stage: pick
  - > Choose 2 threads out of 8 to execute each cycle
- Instruction buffers after L1 instruction cache for each thread
- Increase set associativity of L1 instruction cache to 8
- Increase size of fully associative DTLB from 64 to 128 entries
- Hardware tablewalk for ITLB and DTLB misses
- Speculate branches not taken

# OpenSPARC T1 to T2 Chip Changes

- Increase L2 banks from 4 to 8
  - > 15 percent performance loss with only 4 banks and 64 threads
- FBDIMM memory interface replaces DDR2
  - > Saves pins
  - > Improved bandwidth
    - > 42 GB/sec read
    - > 21 GB/sec write
  - > Improved capacity (512 GB)
- RAS changes (to match T1 FIT rate)

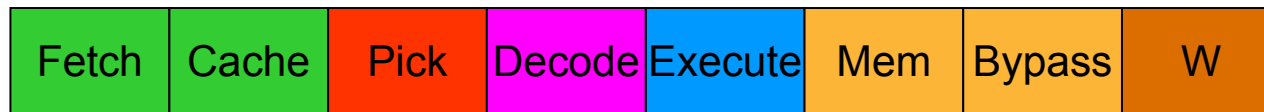
# SPARC Core Block Diagram



- IFU – Instruction Fetch Unit
  - > 16 KB I\$, 32B lines, 8-way SA
  - > 64-entry fully-associative ITLB
- EXU0/1 – Integer Execution Units
  - > 4 threads share each unit
  - > Executes one instruction/cycle
- LSU – Load/Store Unit
  - > 8KB D\$, 16B lines, 4-way SA
  - > 128-entry fully-associative DTLB
- FGU – Floating-Point and Graphics Unit
- TLU – Trap Logic Unit
  - > Updates machine state, handles exceptions and interrupts
- MMU – Memory Management Unit
  - > Hardware tablewalk (HWTW)
  - > 8KB, 64KB, 4MB, 256MB pages
- Gasket arbitrates between the core units for the crossbar interface

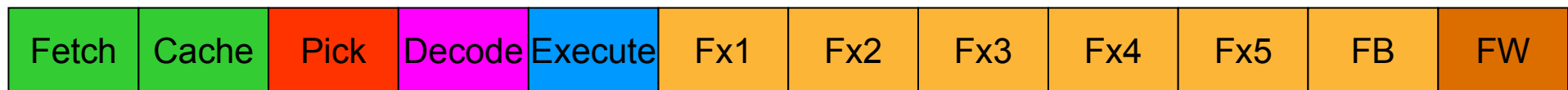
# SPARC Core Pipeline

- 8 stage integer pipeline



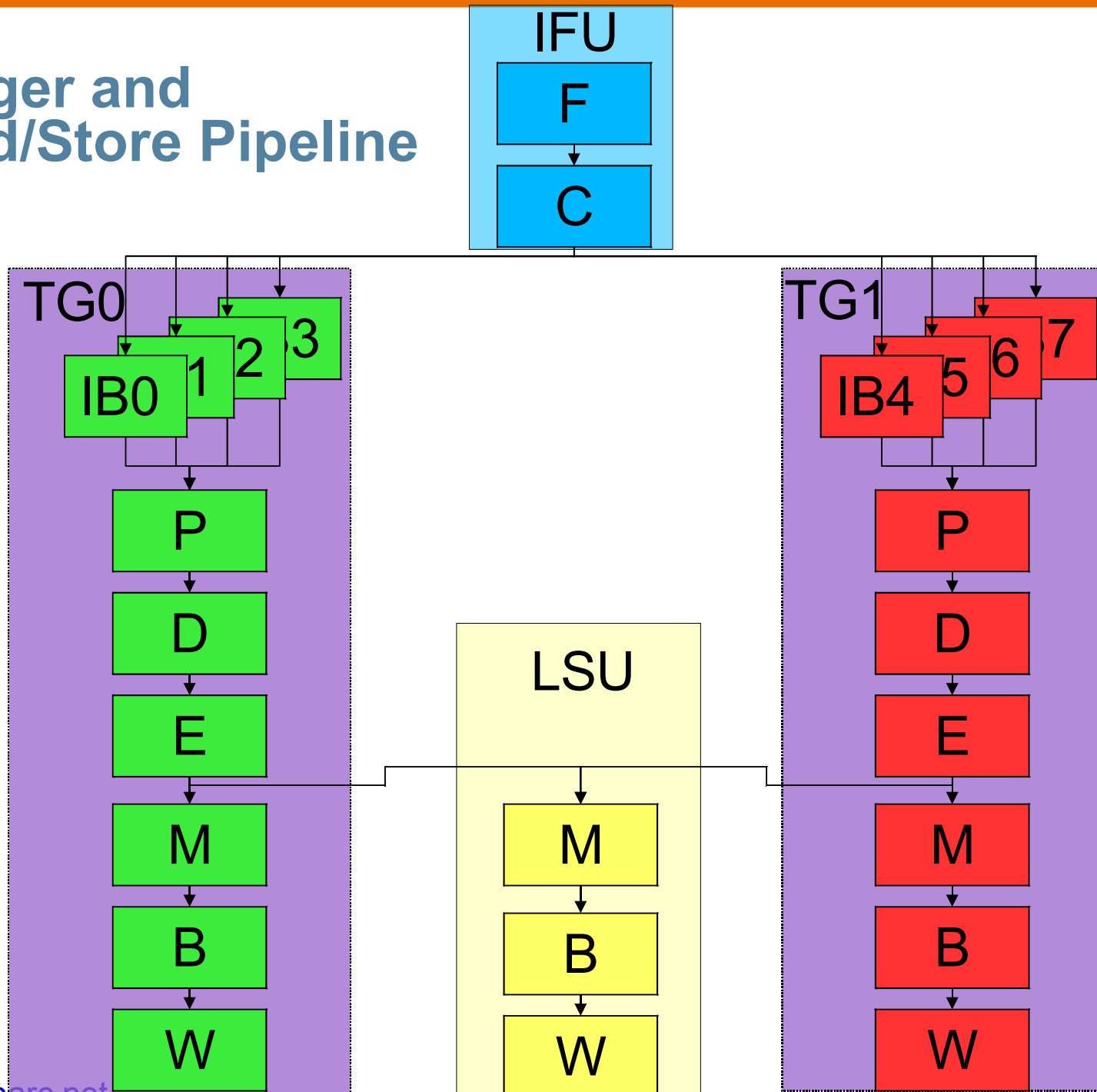
- > 3 cycle load-use penalty
  - > Memory (data address translation, access tag/data array)
  - > Bypass (late way select, data formatting, data forwarding)

- 12 stage floating-point pipeline

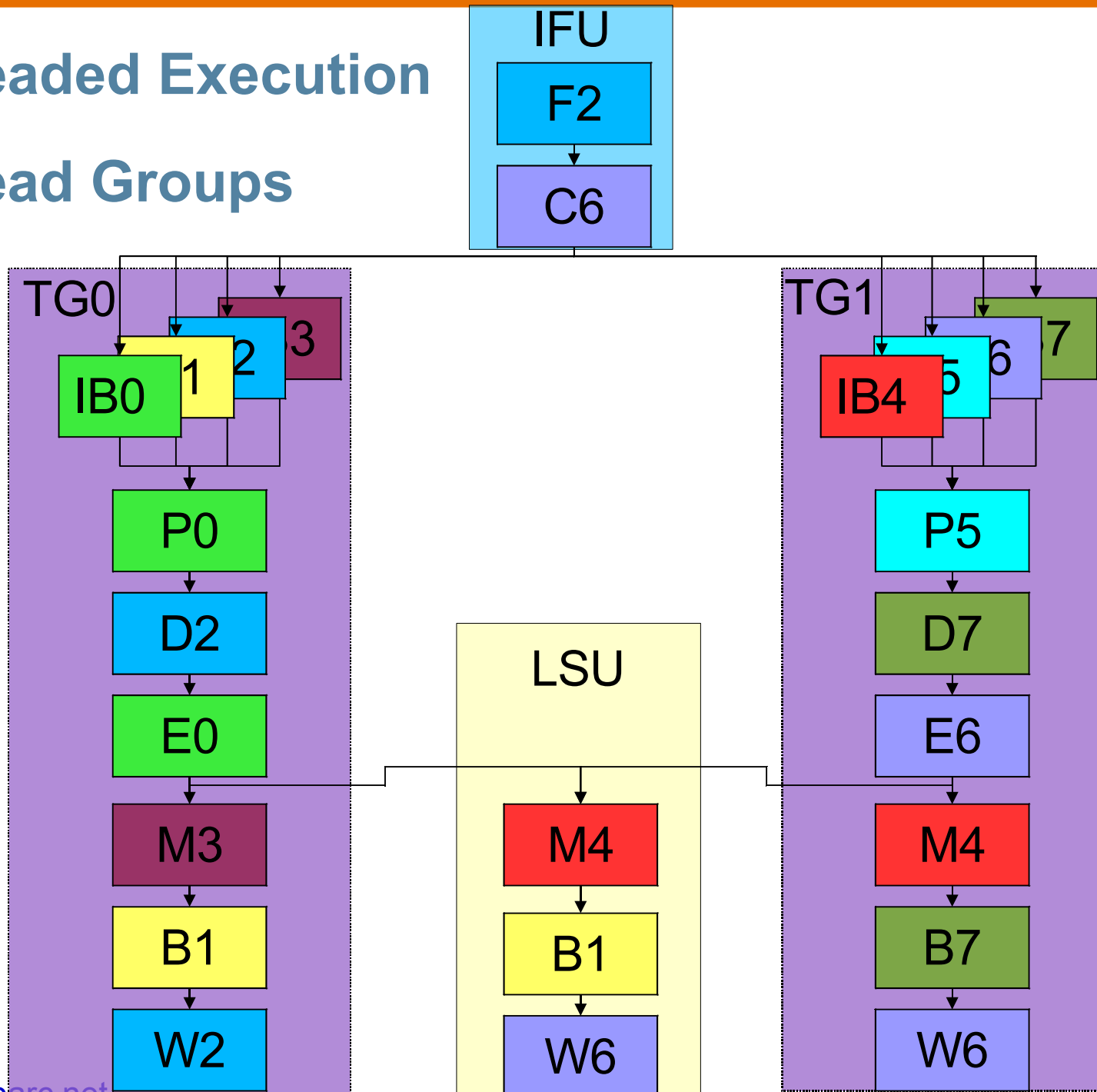


- > 6 cycle latency for dependent FP instructions
- > Longer pipeline for divide/sqrt

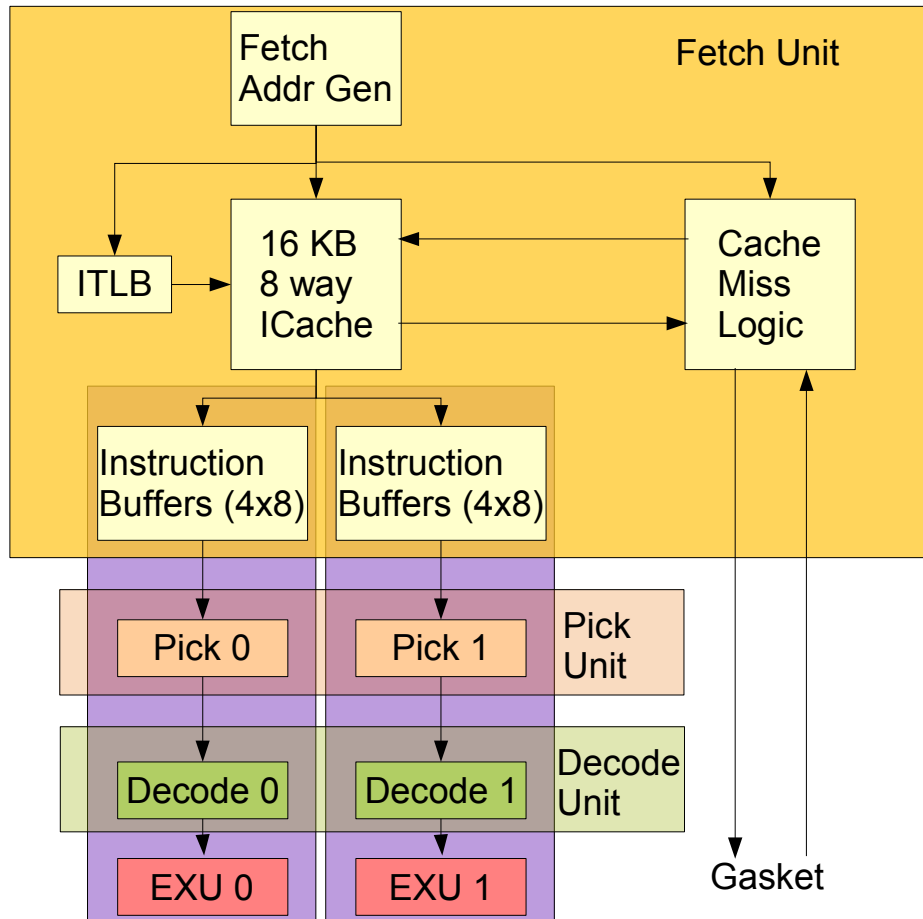
# Integer and Load/Store Pipeline



# Threaded Execution and Thread Groups

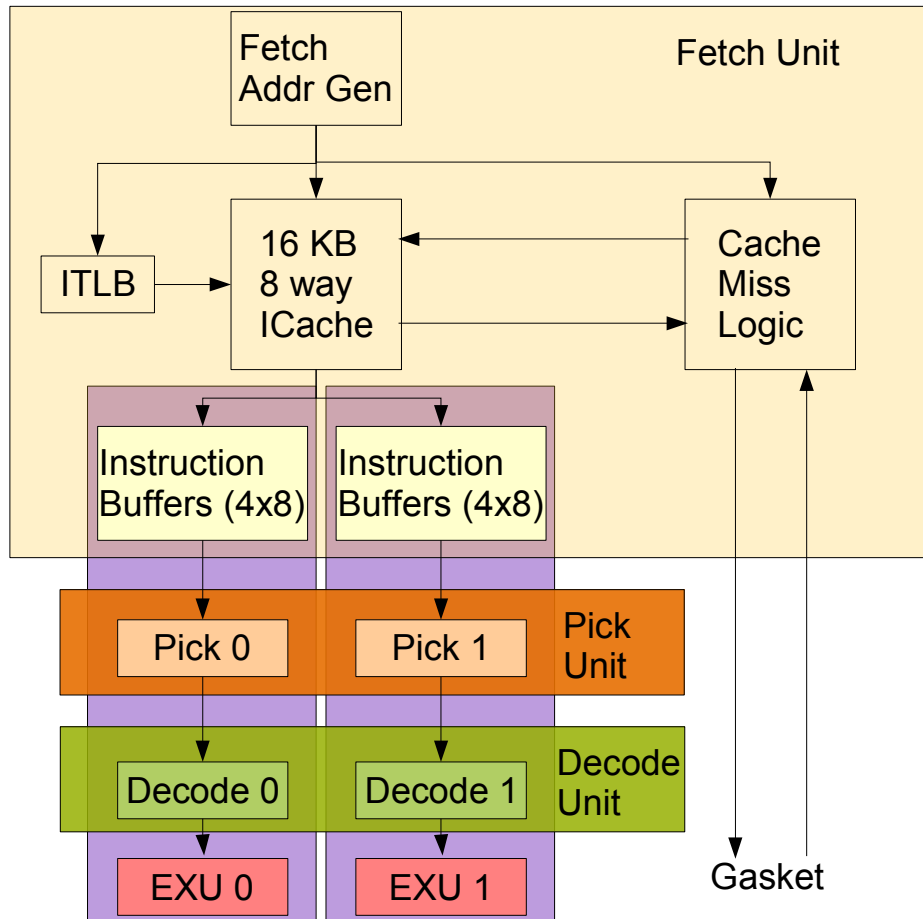


# Instruction Fetch



- Instruction cache and fetch shared between the eight threads
- Fetch up to four instructions per cycle
  - > Each thread in ready or wait state
  - > Wait state caused by:
    - > TLB miss
    - > cache miss
    - > instruction buffer full
  - > Least-recently fetched among ready threads
  - > One instruction buffer/thread
- Branches assumed to be not-taken; 5-cycle penalty if taken
  - > T1 switched threads if branch or load fetched
- Limited I\$ miss prefetching
- Pick and Decode decoupled from Fetch by the instruction buffer

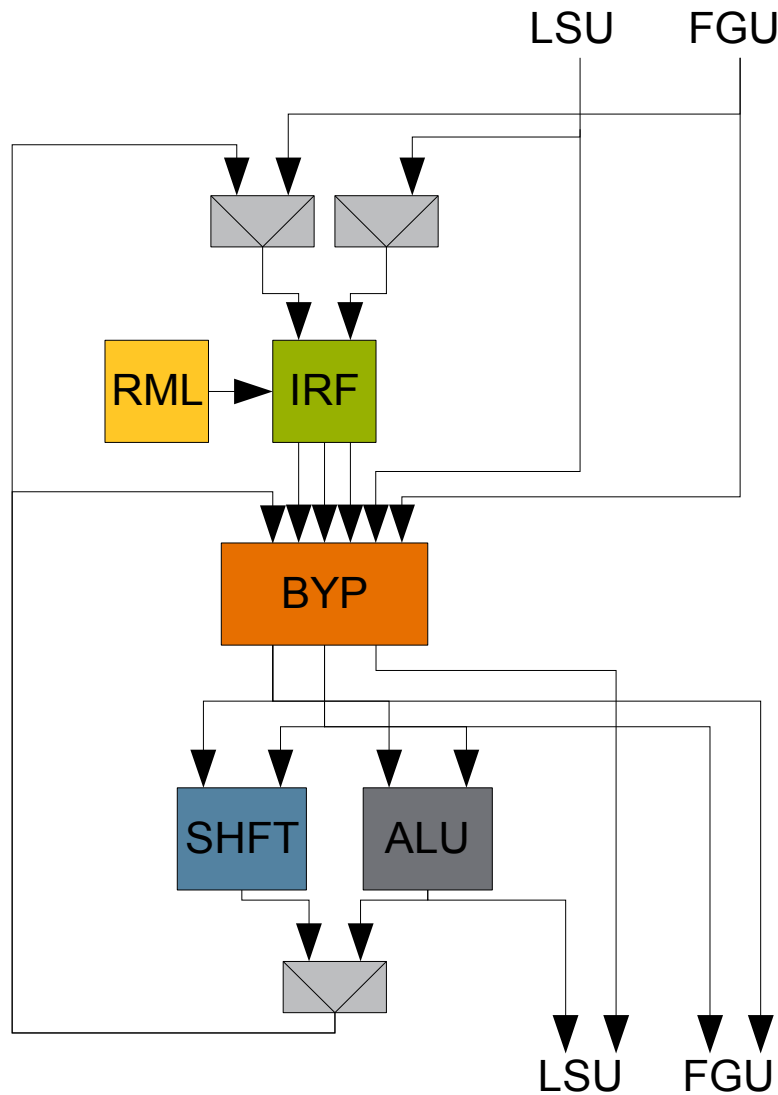
# Instruction Pick and Decode



- Threads divided into two groups of four threads each
- One instruction from each thread group picked each cycle
  - > Least-recently picked within a thread group among ready threads
  - > Wait states: dependency, D\$ miss, DTLB miss, divide/sqrt, ...
  - > Gives priority to nonspeculative threads (e.g. no load)
- Decode resolves conflicts
  - > Each thread group picks independently of the other
  - > Both thread groups pick load/store or FGU instructions
- Independent instructions after loads

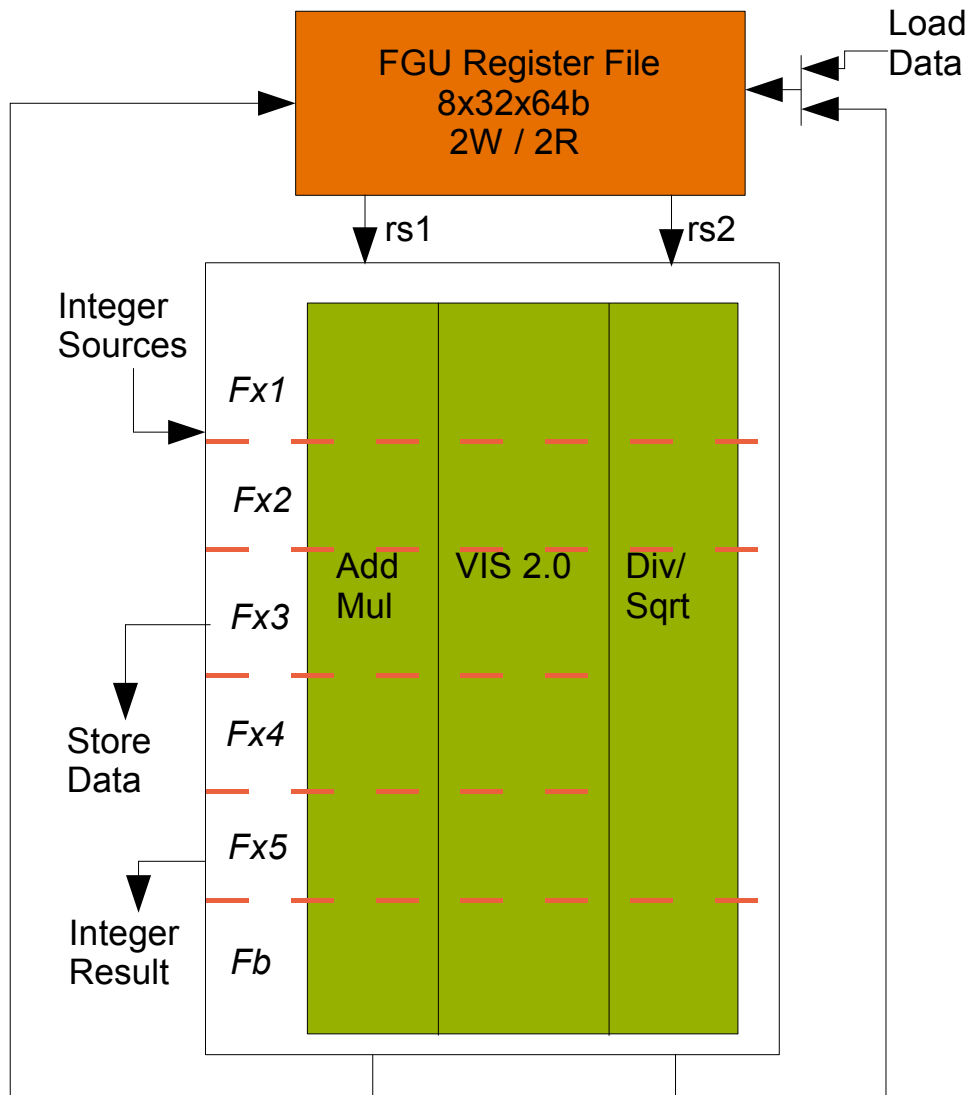
# Execution Unit

- Executes integer operations and some graphics operations
- Generates addresses for loads and stores
- Adder / logic unit, shifter
- Each EXU contains state for four threads
  - > Integer register file (IRF)
    - > 8 register windows per thread
    - > 4 global levels per thread
    - > Window or global level change requires multiple cycles (but pipelined)
  - > Register window management logic (RML)





# Floating-point and Graphics Unit



- Fully pipelined (except divide/sqrt)
  - > Divide/sqrt in parallel with add or multiply operations of other threads
- FGU performs integer multiply, divide, population count
- FGU predicts exceptions in Fx1 stage

# Memory Management Unit

- Hardware tablewalk of up to 4 translation storage buffers (TSBs) (a.k.a page tables)
  - > Each TSB supports one page size
- Three search modes:
  - > Sequential – search TSBs in order
  - > Burst – search TSBs in parallel
  - > Prediction – use VA to predict TSB to search
    - > Two-bit predictor orders first two TSB searches
- Up to 8 pending misses
  - > ITLB or DTLB miss per thread

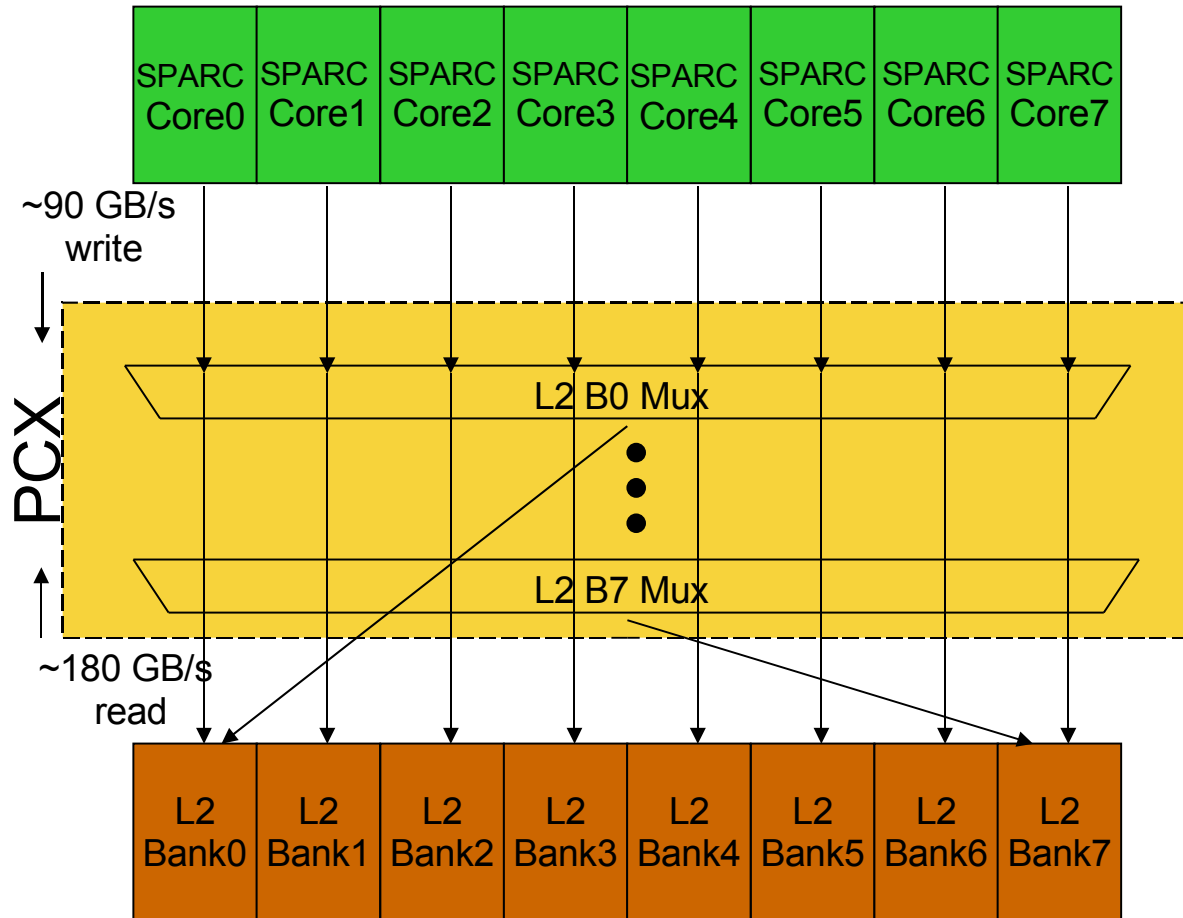
# Core Power Management

- Minimal speculation
  - > Next sequential I\$ line prefetch
  - > Predict branches not-taken
  - > Predict loads hit in D\$
  - > Pick independent instructions after loads
  - > Hardware tablewalk search control
- Extensive clock gating
  - > Datapath
  - > Control blocks
  - > Arrays
- External power throttling
  - > Add stall cycles at decode stage

# Core Reliability and Serviceability

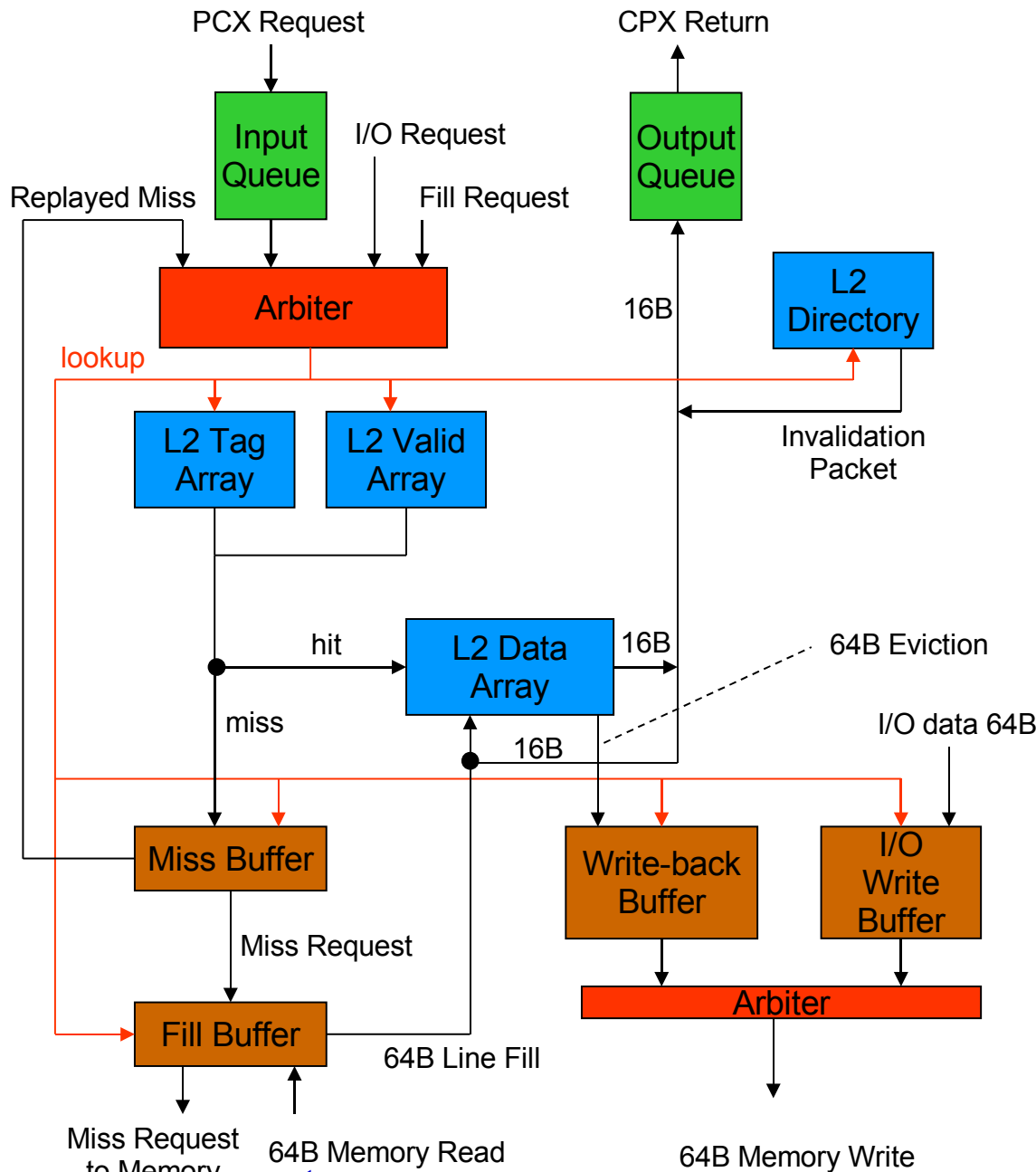
- Extensive RAS features
  - > Parity-protection on I\$, D\$ tags and data, ITLB, DTLB CAM and data, store buffer address
  - > ECC on integer RF, floating-point RF, store buffer data, trap stack, other internal arrays
- Combination of hardware and software correction flows
  - > Hardware re-fetch for I\$, D\$
  - > ECC inside the core is corrected by software

# Crossbar



- Two complementary, non-blocking, pipelined switches
  - > PCX – processor to cache
  - > CPX – cache to processor
- 8 load/store requests and 8 data returns can be done at the same time
- Arbitration for a target is required
- Priority given to oldest requestor to maintain fairness and order
- Three cycle arbitration protocol
  - > Request, arbitrate, and grant
- Supports 8 byte writes from a core to a bank
- Supports 16 byte reads from a bank to core

# L2 Cache



- 4 MB L2 cache
  - > 16 way set associative
  - > 8 L2 banks
  - > 64 byte line size
    - > T1: 3 MB, 12 ways, 4 banks
- L2 cache is write-back, write-allocate
  - > L1 data cache is write-thru
- Support for partial stores
- L2 cache manages coherency
  - > Maintains directories for all 16 L1 caches
- 16 byte data transfers to the cores

# Summary

- >2x throughput and throughput/watt vs. OpenSPARC T1
- Greatly improved floating-point performance
- Significantly improved integer performance



OpenSPARC™

# OpenSPARC Slide-Cast

In 12 Chapters

Presented by OpenSPARC  
designers, developers, and  
programmers

- to guide users as they develop  
their own OpenSPARC designs  
and
- to assist professors as they  
teach the next generation

This material is made available under  
Creative Commons Attribution-Share 3.0 United States License

