CRAY
THE SUPERCOMPUTER COMPANY

Overview of the Next Generation Cray XMT

Andy Kopser
Cray Inc

23 May 2010

Introduction

- Cray XMT
  - Scalable, multithreaded, shared memory machine
  - Designed for single-word random global access patterns
  - Very good at large graph problems
- Next Generation Cray XMT Goals
  - Memory System Improvements
    - Improve bandwidth for random access
    - Improve capacity for large graphs
  - Hot Spot Avoidance
    - Shared memory programming models generally susceptible to hot spotting
    - The current XMT is no exception
    - Add hot spot avoidance hardware to the CPU

Why Multithreading?

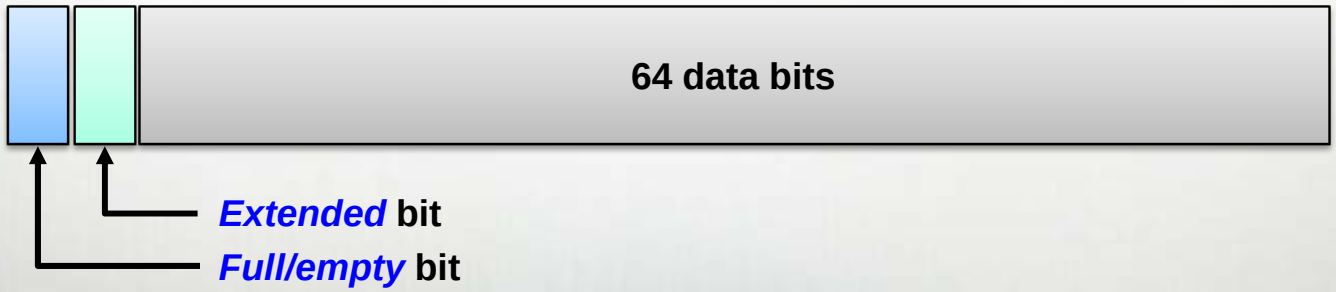
- Relative latency to memory continues to increase
  - Vector processors *amortize* memory latency
  - Cache-based microprocessors *reduce* memory latency
  - Multithreaded processors *tolerate* memory latency
- Multithreading is most effective when:
  - Parallelism is abundant
  - Data locality is scarce
- Large graph problems perform well on the Cray XMT
  - Semantic databases
  - Big data

Threads and Streams

- A thread is a software object
  - A program counter and a set of registers
  - Very lightweight
    - Not pthreads
    - No OS state
- A stream is a hardware object
  - Stores and manipulates a thread's state
  - Very lightweight stream creation
    - A single instruction executed from user space
- More threads than streams
- Threads multiplexed onto the processor's streams

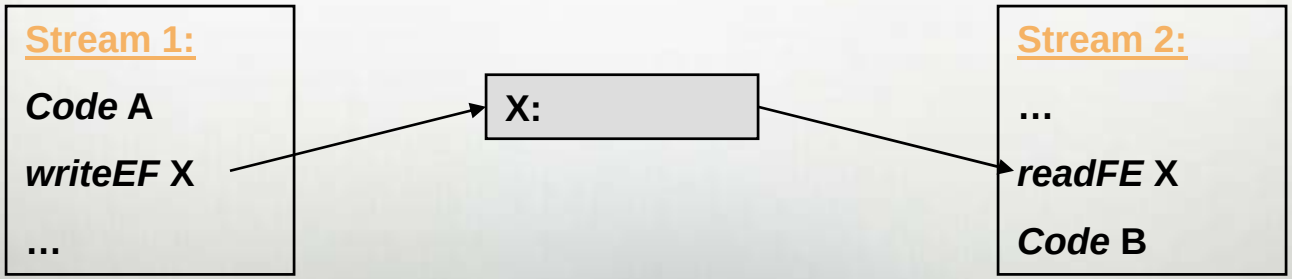
Extended Memory Semantics

- The XMT memory word has 66 bits
  - 64 bits of data, byte addressable
    - Data is stored big-endian
  - 2 tag bits
    - The *full/empty* bit
      - Used for synchronization
    - The *extended* bit
      - Set when the entry is forwarded or when a trap bit is set



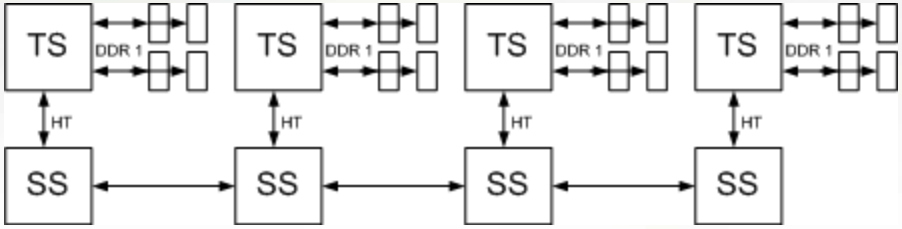
Full/Empty Access Control

- Specified by pointer or instruction
- Three access modes
  - FE_NORMAL
  - FE_FUTURE
  - FE_SYNC (*readFE*, *writeEF*)
- Provides efficient, abundant, fine-grained synchronization

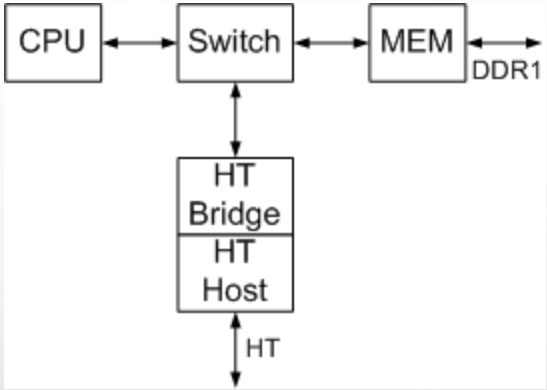


Cray XMT

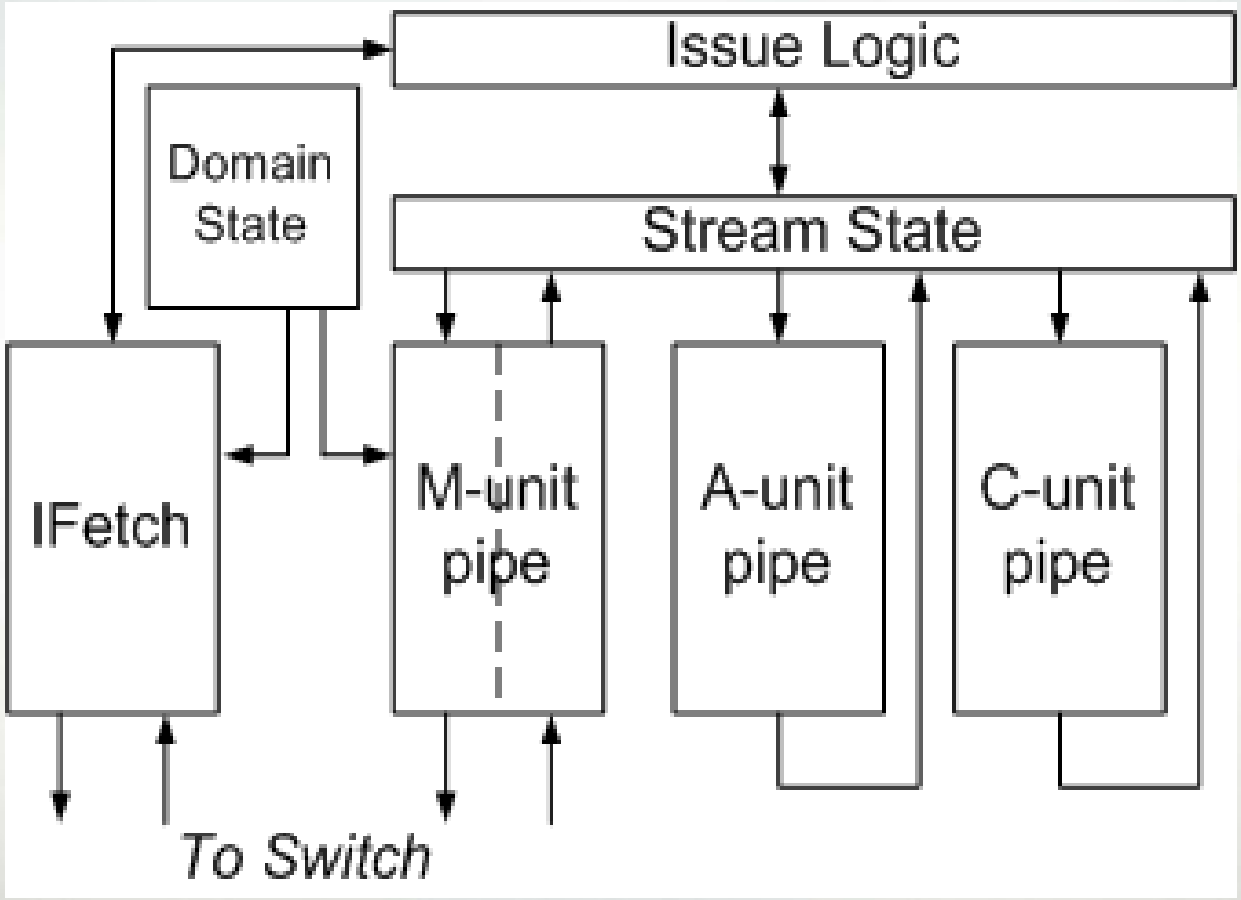
Cray XMT blade



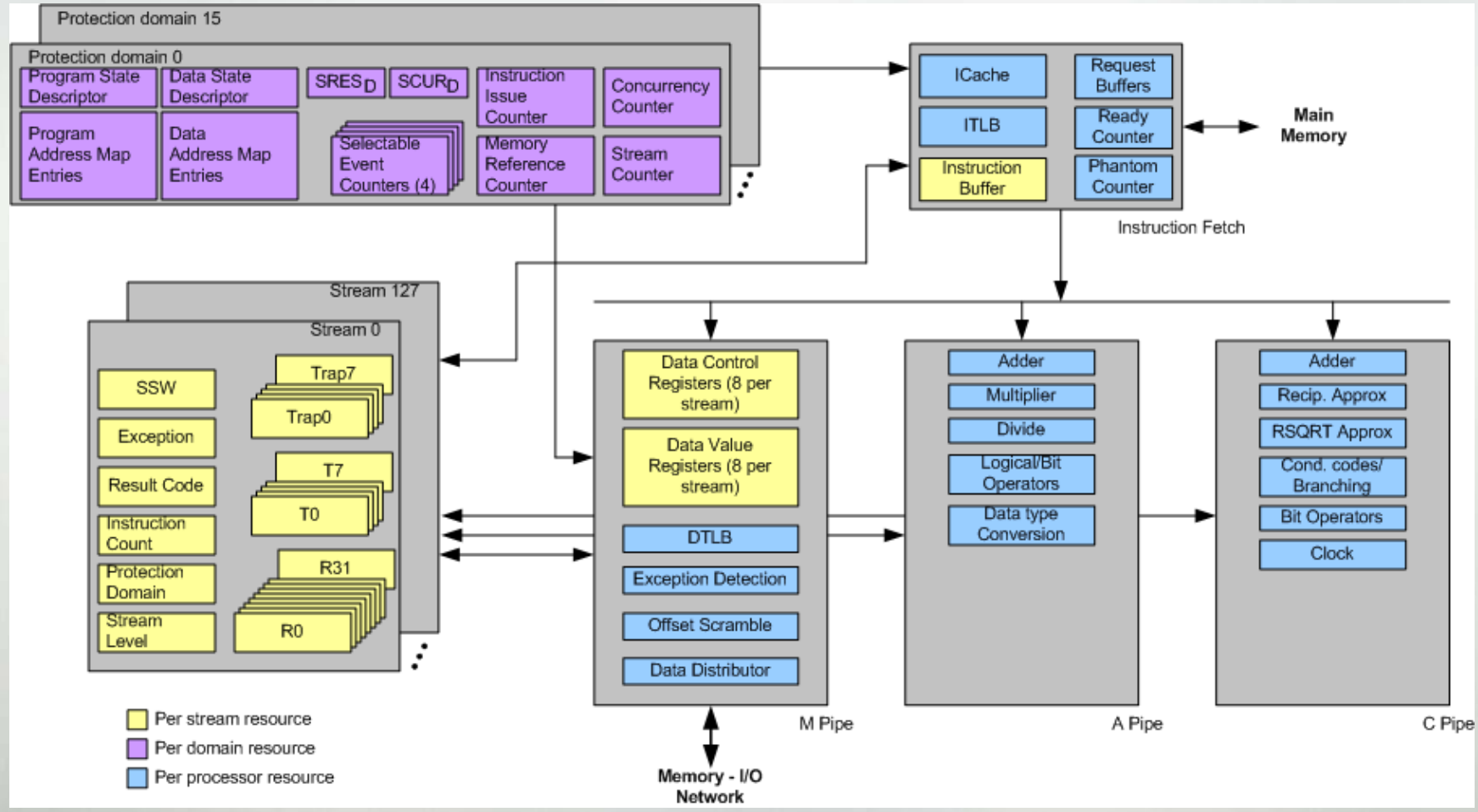
Threadstorm3



Cray Threadstorm CPU



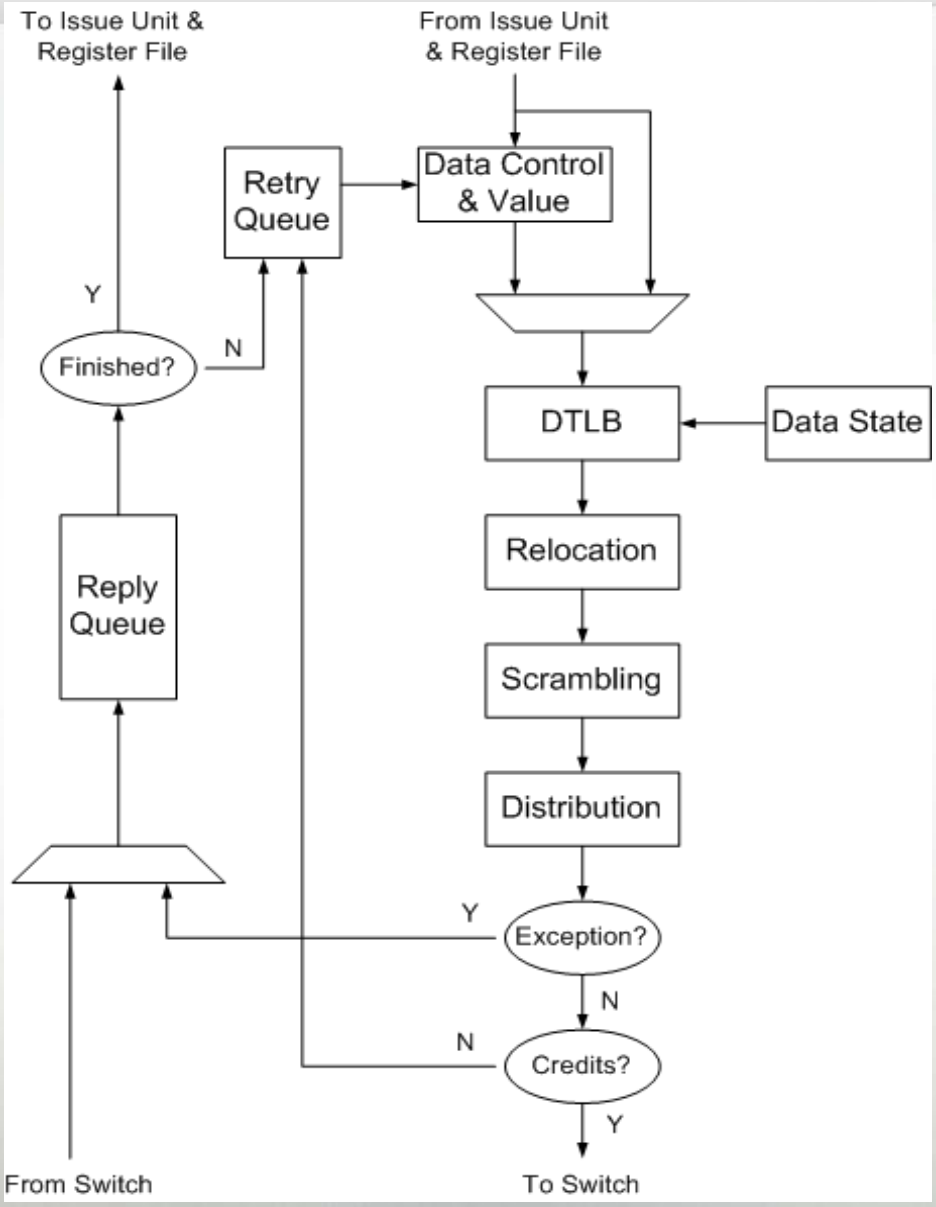
Central Processing Unit



CPU M-unit

- Storage to track up to 1024 memory references
- Performs data address translation
  - Relocate according to domain data state
  - Scrambling to hash address bits
  - Distribution to spread references across machine
- Issues requests to Switch
- Handles retries if necessary
- Updates stream state upon completion

M-unit

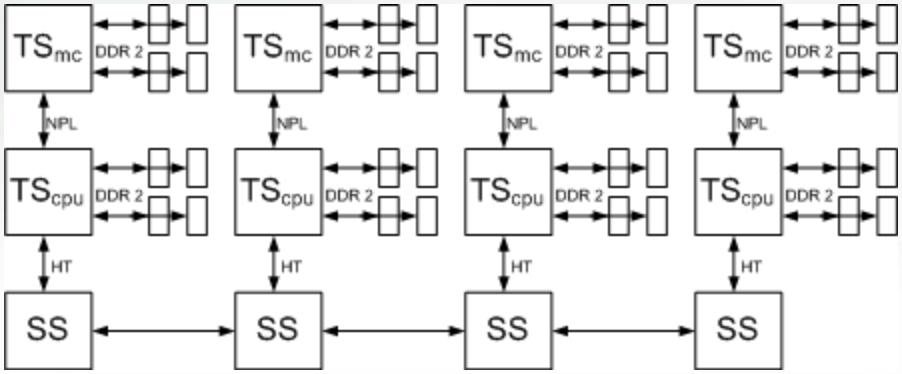

To Issue Unit &
Register File
From Issue Unit
& Register File
Retry
Queue
Data Control
& Value
Y
N
Finished?
DTLB
Data State
Relocation
Reply
Queue
Scrambling
Distribution
Y
Exception?
N
N
Credits?
Y
From Switch
To Switch

Remote Memory Access

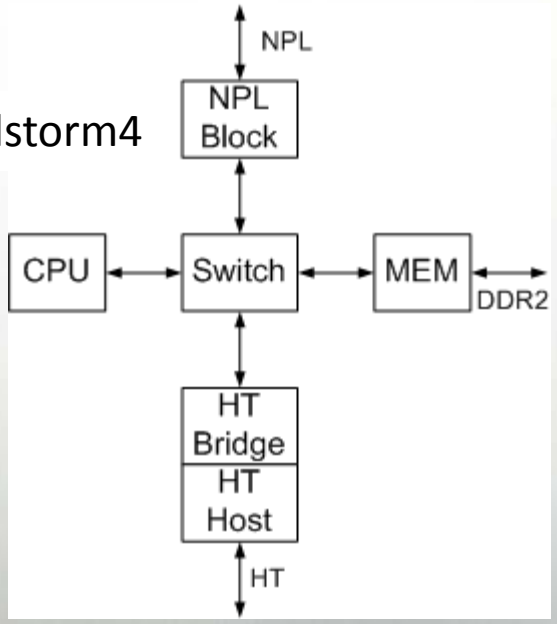
- All remote memory references go through the RMA block in the HyperTransport Bridge
- RMA block serves three purposes:
  - Bypass HT native addressing to allow up to 512TB of memory to be directly referenced
  - Support extended memory semantics
  - Encapsulate multiple references in each HT packet for efficient use of the link
- All RMA traffic packed into 64-byte payload of HT posted writes

Next Generation Cray XMT

Next Generation Cray XMT blade



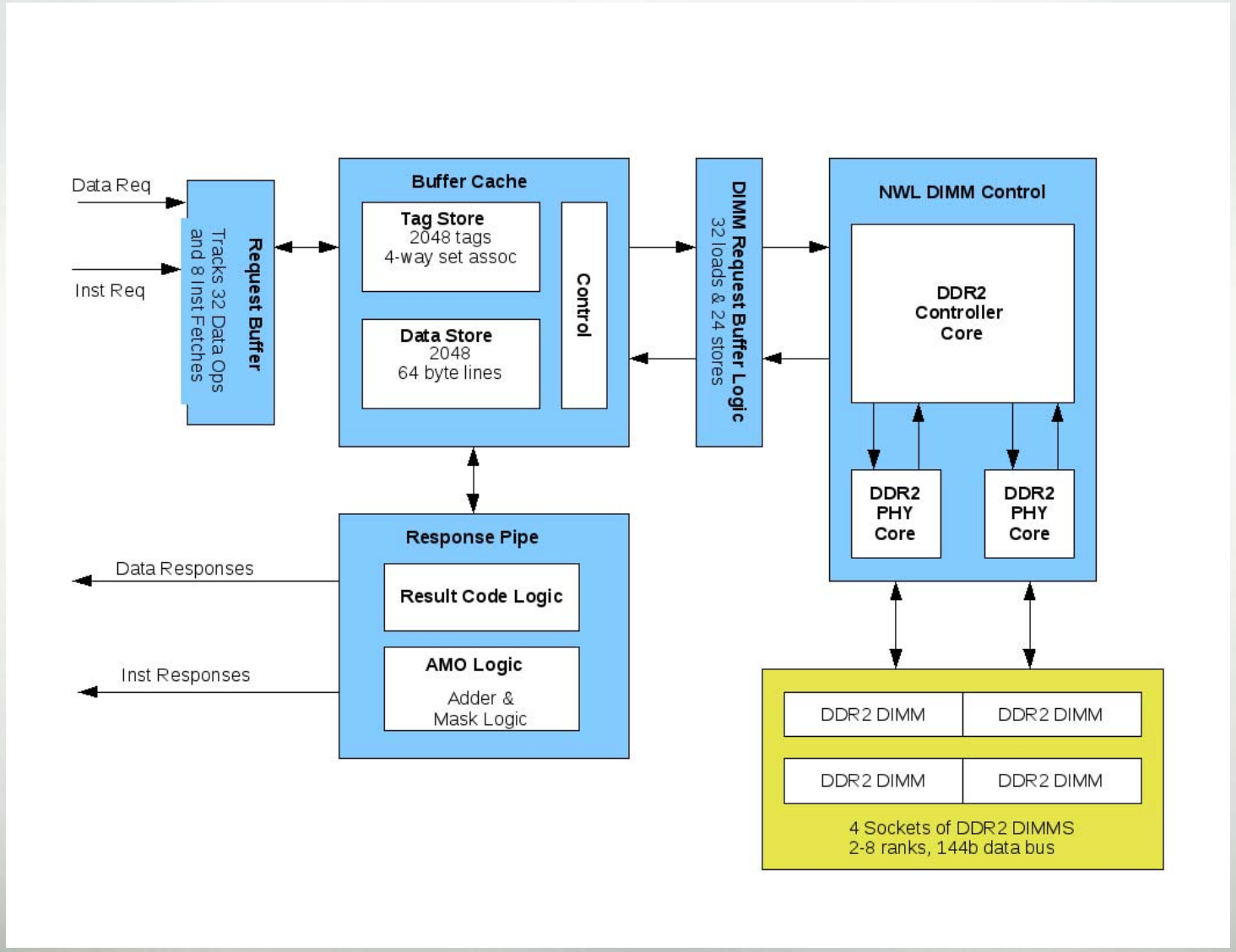
Threadstorm4



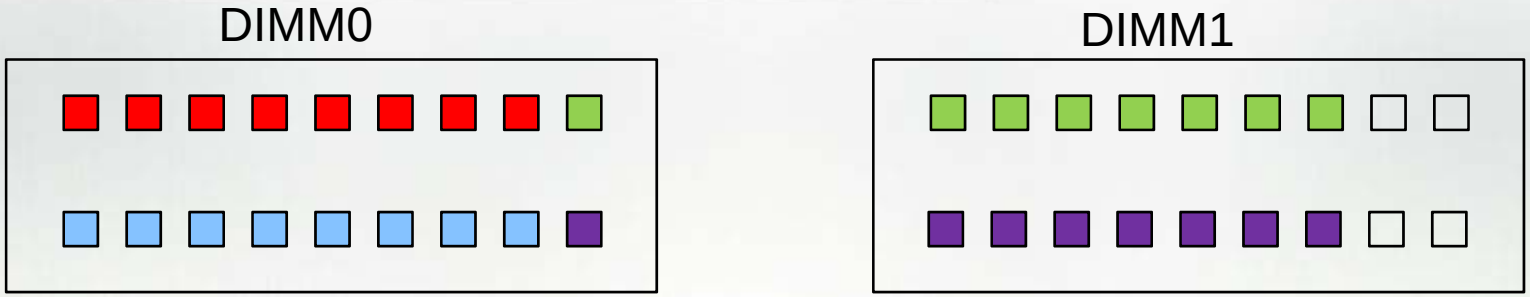
Memory Subsystem Organization

- Two memory controllers per node
  - Each 50% faster than the current implementation
  - 3x bandwidth improvement
  - 8x capacity improvement
- Optimized for single 8B word random address accesses
- 64b adder for atomic Fetch&Add
- 128kB buffer cache between Switch and DIMMs
- No coherency issues
  - All DIMM operations go through cache
  - This buffer is associated with the physical memory, not the processor
- 64B cache line

Memory Subsystem



Memory Error Correction

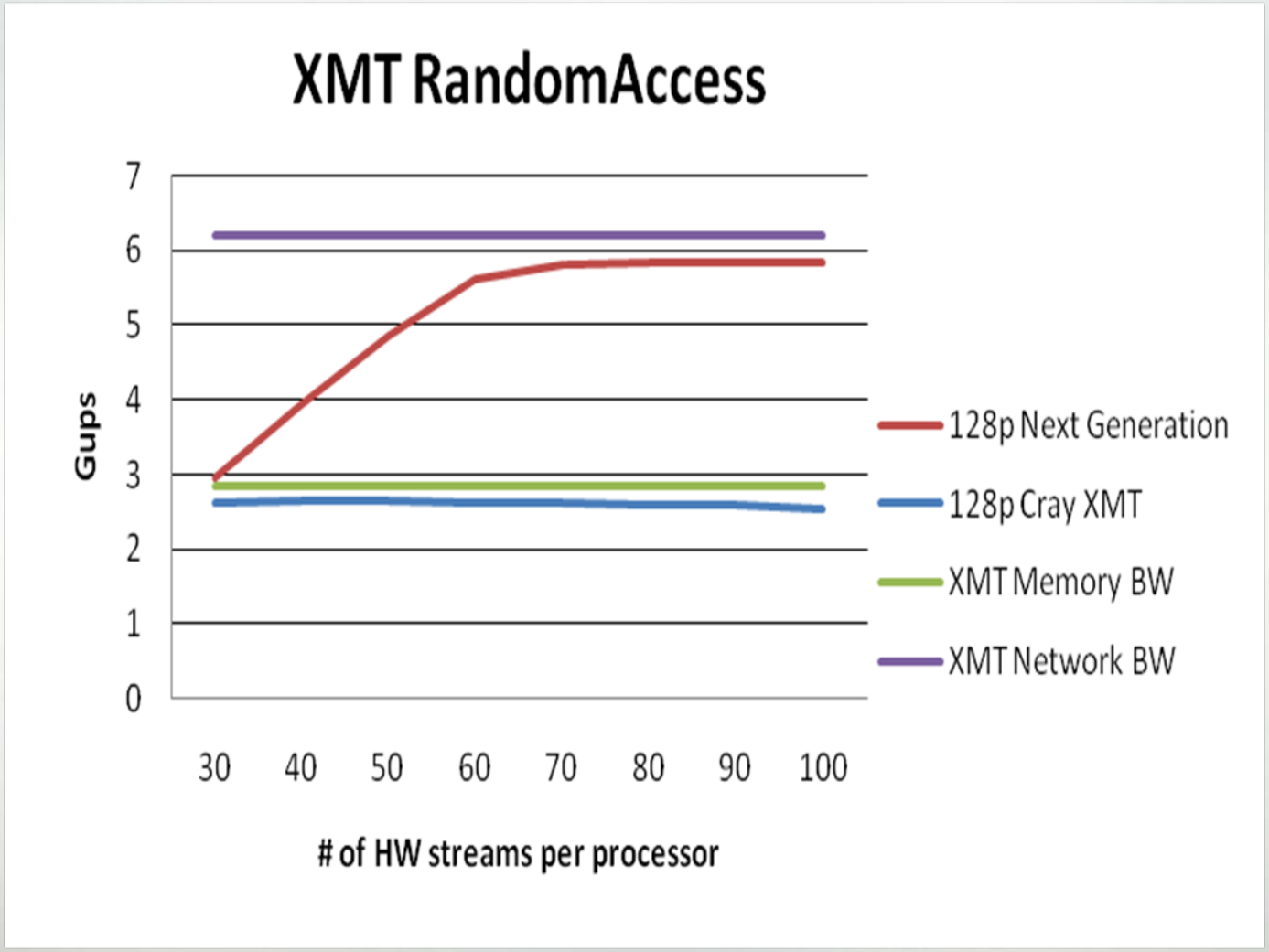


- Standard DIMMs store 9 bytes per address
  - 8 bytes for data
  - 1 byte for check bits
- Each DIMM rank implemented with 18 4-bit memory parts
- Correct any number of errors in a single part
  - Gang two DIMMS together
  - Reed-Solomon code implemented over two flit times
    - 288 bits total
    - 32 parts for data
    - 1 part for state
    - 3 parts for check bits

Next Generation XMT DIMMs

- DDR2 registered DIMMs at 300MHz
  - Supports Burst=4
    - Allows 64B cache line in ganged mode
    - Better for single word random accesses
  - DDR3 only supports Burst=8, doubling cache line size
  - Better timing windows
- DIMMs supported by hardware:
  - 4GB Dual Rank
  - 8GB Dual Rank
  - 8GB Quad Rank
- 8 DIMM slots per node
  - 32GB per node using 4GB DIMMs
  - 64GB per node using 8GB DIMMs

RandomAccess Performance



Hot Spot Avoidance Techniques

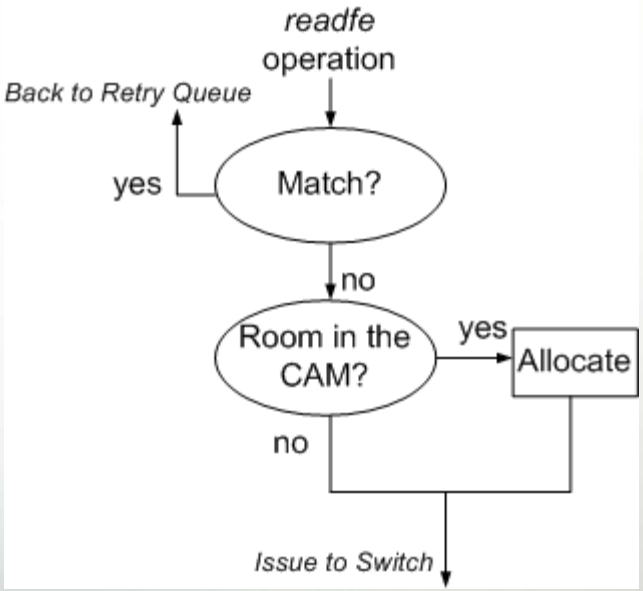
- Many streams may access the same memory location simultaneously
- Threadstorm4 solves the problem in the M-unit
  - Allow only one outstanding reference of a given type for each address
  - Use the network more efficiently
- Synchronized Reference CAM for *readFE* (or *writeEF*)
  - Only one operation can find the location full (or empty)
  - Others are deferred and tried later
- Fetch&Add Combining CAM
  - Fetch&Add operands to same address combined in M-unit
  - One network request satisfies multiple Fetch&Add requests

Synchronized Reference Hot Spots

- *readFE* waits for full, then loads and sets empty
- *writeEF* waits for empty, then stores and sets full
- Critical code segment may be protected by readFE/writeEF
  - If frequently executed, *readFE* may cause hot spot
- Retries handled by M-unit—one round-trip to memory and back for each retry
- Each processor may issue about 100 *readFE* operations at once
  - At most one will be successful
  - Others just consume network and memory bandwidth

SynchRef CAM

- SynchRef CAM in next generation Cray XMT avoids hot spots
- Only one *readFE* to a given address can succeed
  - Don't allow more than one on the network
- When *readFE* would be injected, check in the CAM

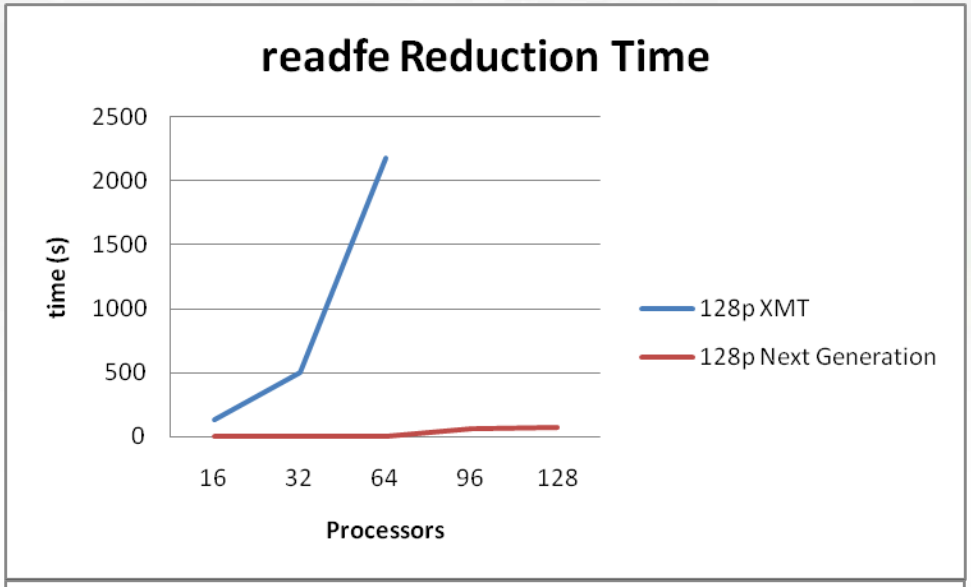


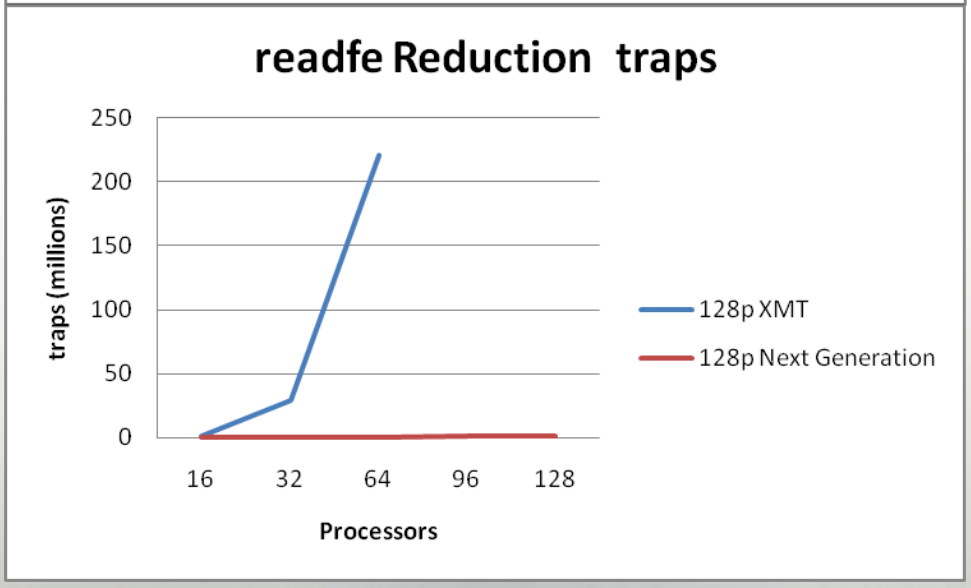
CAM entry deallocated when response is received

readFE Reduction

- Test SynchRef CAM with worst possible program
- Large reduction protected by *readFE/writeEF* pair
- Only one stream at a time does work
- Run on 100 streams per processor
- For N processors, 100*N streams compete to read location

SynchRef CAM Performance





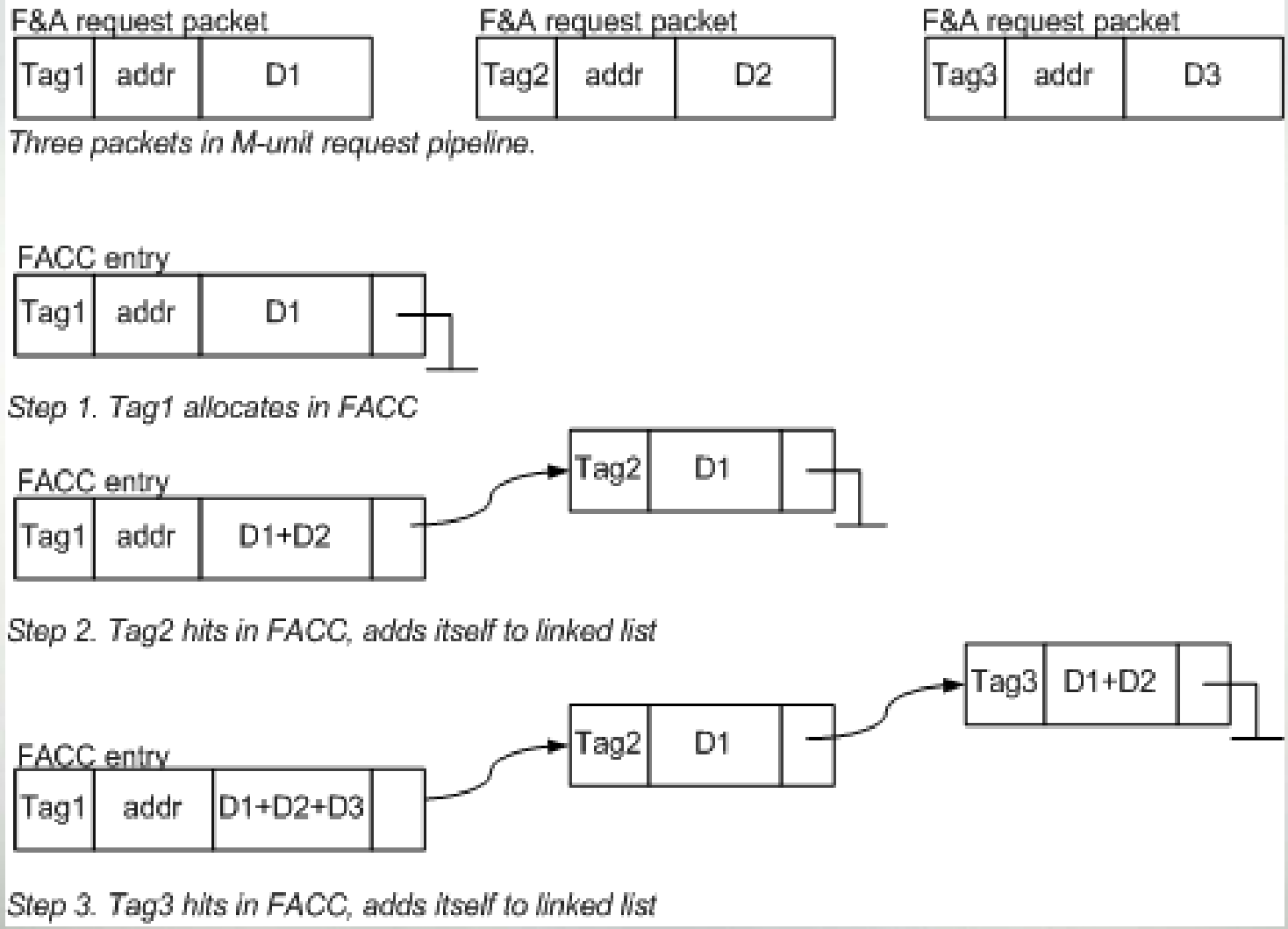
Fetch & Add Hot Spots

- Cray XMT supports fetching and non-fetching atomic add operations
- A single memory location may be accessed by all streams
  - Queue pointer or global reduction
- Each processor generates about 100 Fetch&Add requests
  - Oversubscribes memory node

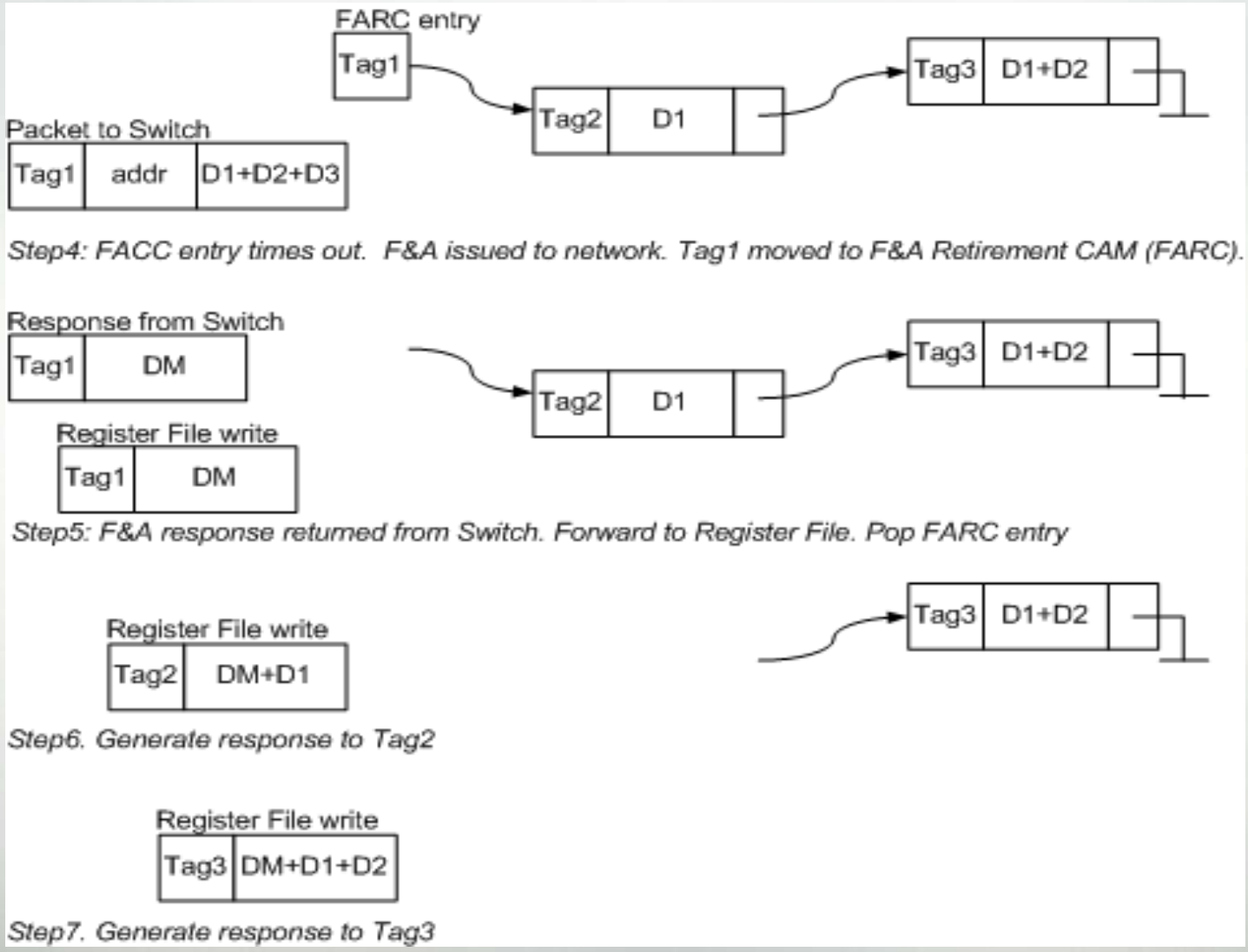
Fetch & Add Combining

- Fetch & Add Combining in next generation Cray XMT eliminates hot spots
- Fetch & Add operation checks in F&A Combining CAM (FACC)
  - If a match is not found, allocate in the FACC
  - If a match is found, attach itself to a linked list of dependents
- FACC entry
  - Accumulates data
  - Generate network request after specified wait time
- F&A Retirement CAM entry
  - Allocated when network request is made
  - Pointer to linked list of dependents
  - When response is received, multiple register file writes generated

Fetch & Add Combining Example



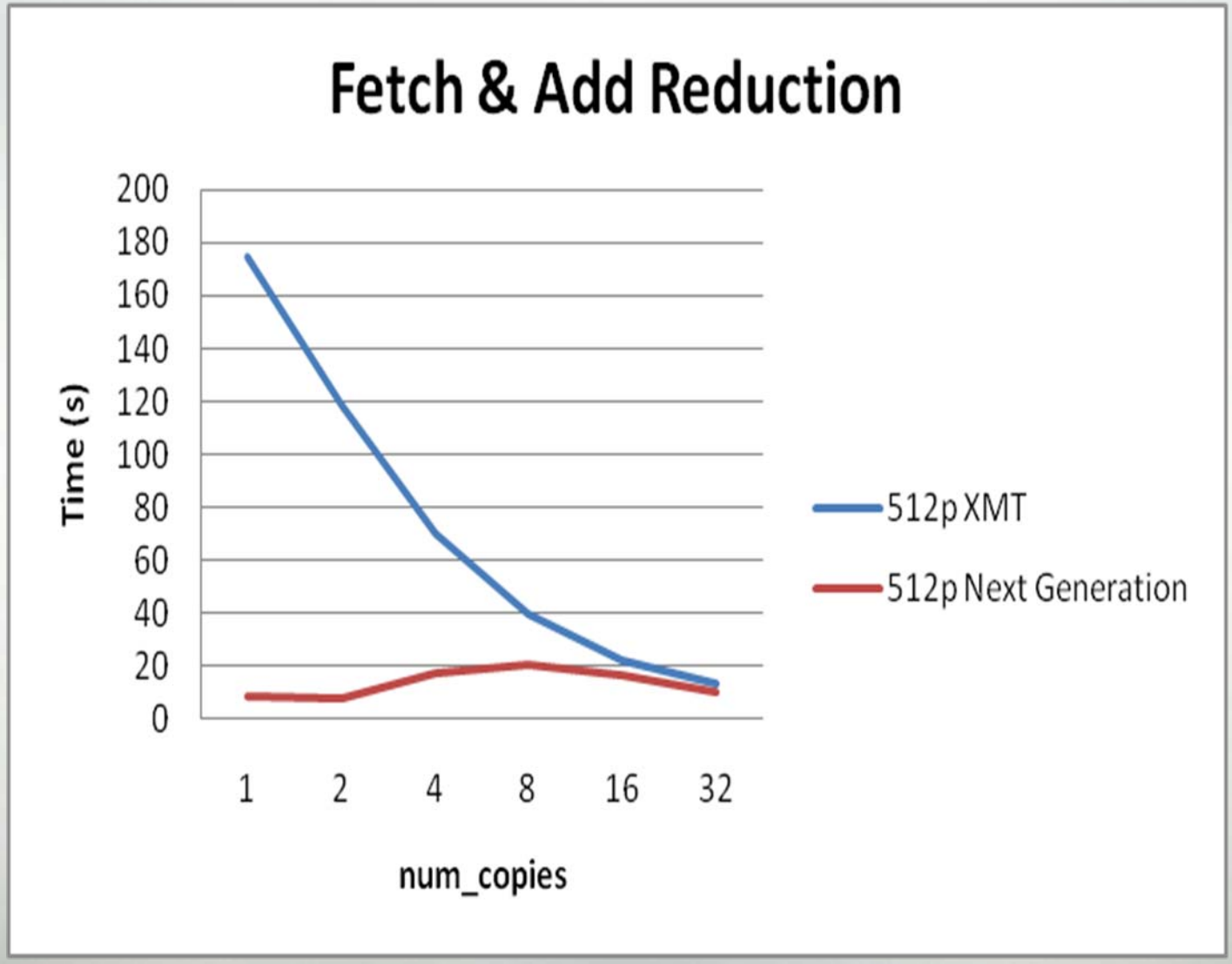
Fetch & Add Combining Example (continued)



Fetch & Add Reduction

- Current Cray XMT trick when updating a global accumulator
  - Make several copies of the accumulator
  - Randomly select one to update
  - Requires an additional computation at the end
- Test F&A Combining Logic using this trick
  - Perform global additive reduction
  - Vary the number of copies: 1, 2, 4, 8, 16, 32
- Current Cray XMT
  - Hot spot created with small numbers of copies
  - Performance improves as copies are added
- Next generation Cray XMT
  - Performs best with a single copy

Fetch & Add Combining Performance



Conclusion

- Next Generation builds on successful Cray XMT
- Memory system improved significantly
  - 3x improvement in bandwidth
  - 8x improvement in capacity
- Hot Spot Avoidance
  - Productivity—simple implementation performs best
  - Reliability—difficult programs cannot interrupt system services
  - Performance—use network more efficiently


CRAY
THE SUPERCOMPUTER COMPANY