Piton: A 25-core Academic Manycore Research Processor

Michael McKeown, Yaosheng Fu, Tri Nguyen, Yanqi Zhou, Jonathan Balkind, Alexey Lavrov, Mohammad Shahrad, Samuel Payne*, and David Wentzlaff

Hot Chips 28

August 23, 2016

PRINCETON UNIVERSITY

*Work done at Princeton, now at NVIDIA

PRINCETON School of Engineering and Applied Science

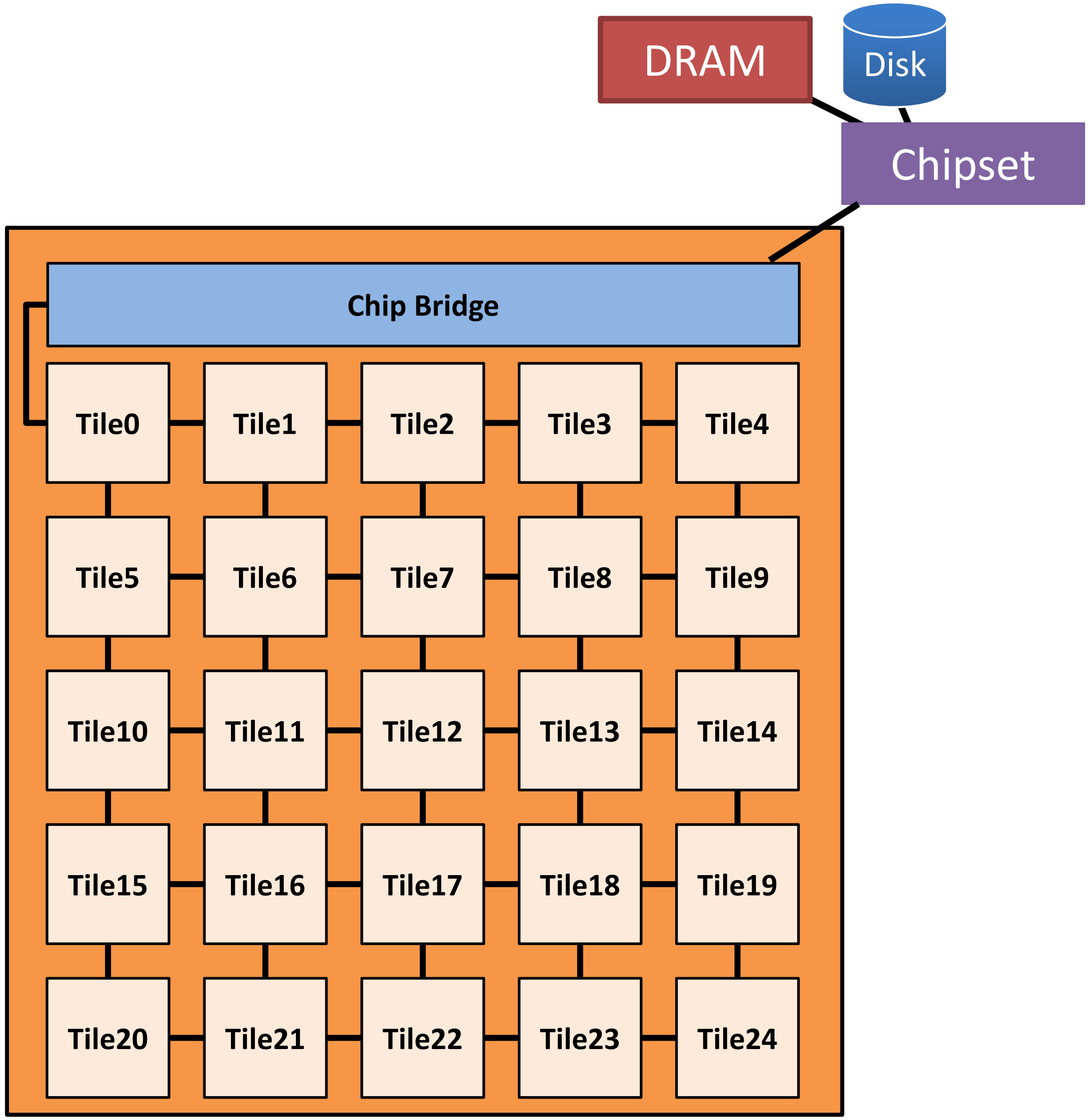
Cloud and Warehouse Scale Computing Growth

Cloud and Warehouse Scale Computing Growth

Cloud and Warehouse Scale Computing Growth

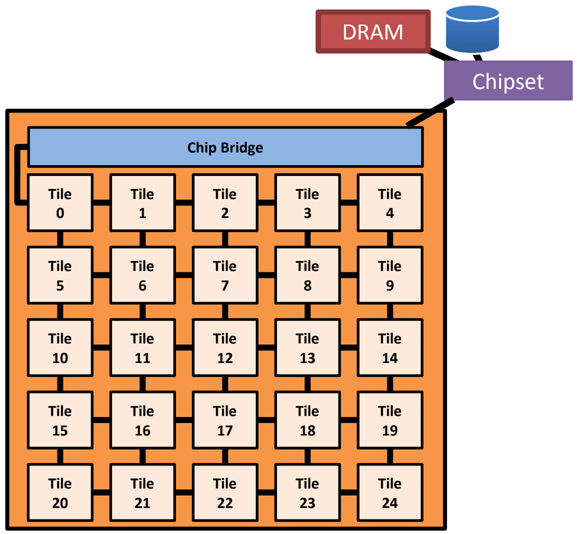
Cloud and Warehouse Scale Computing Growth

Piton: A Manycore for the Data Center and IaaS Clouds

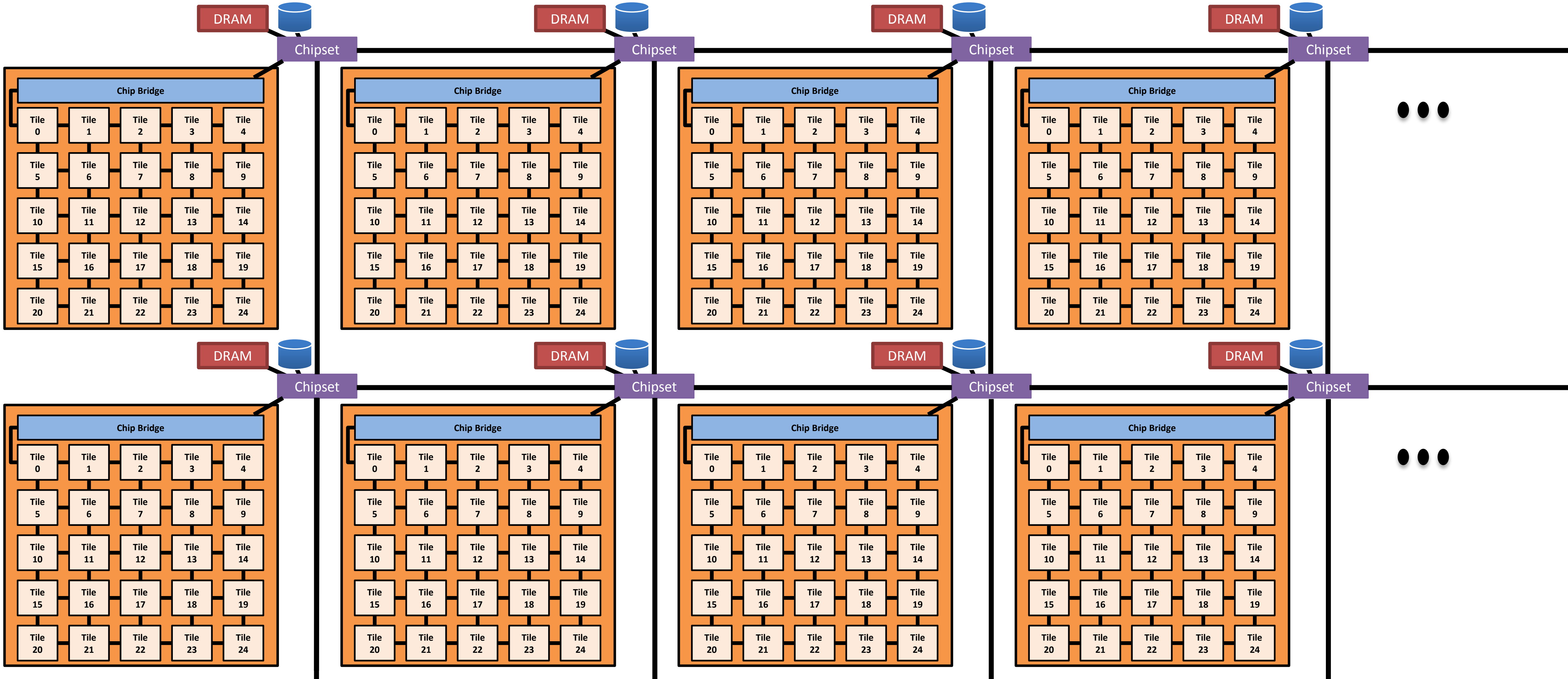


- Scalable architecture
- Exploit Commonality
- Enable novel Infrastructure as a Service (IaaS) economic models
- Break down boundaries between chips, boards, and racks

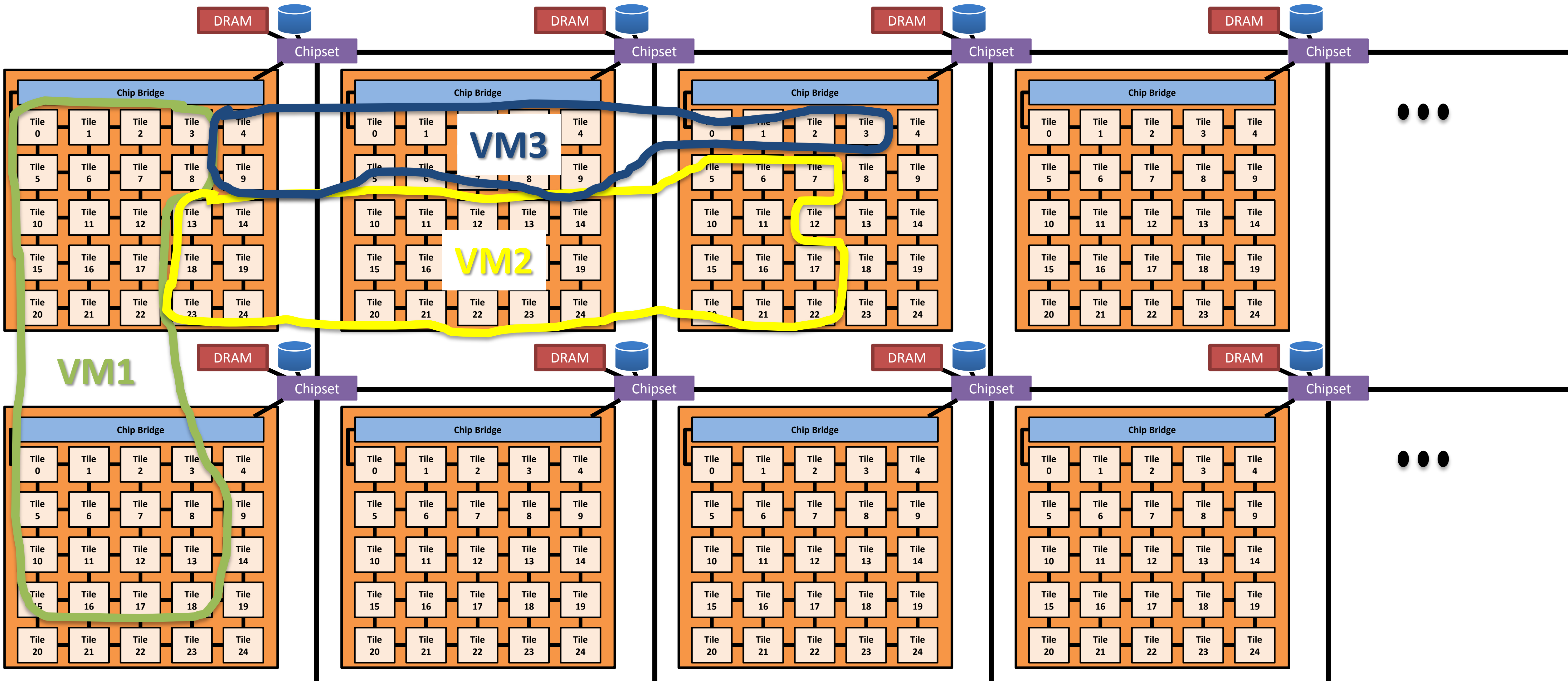
Piton: A Manycore for the Data Center and IaaS Clouds



Piton: A Manycore for the Data Center and IaaS Clouds

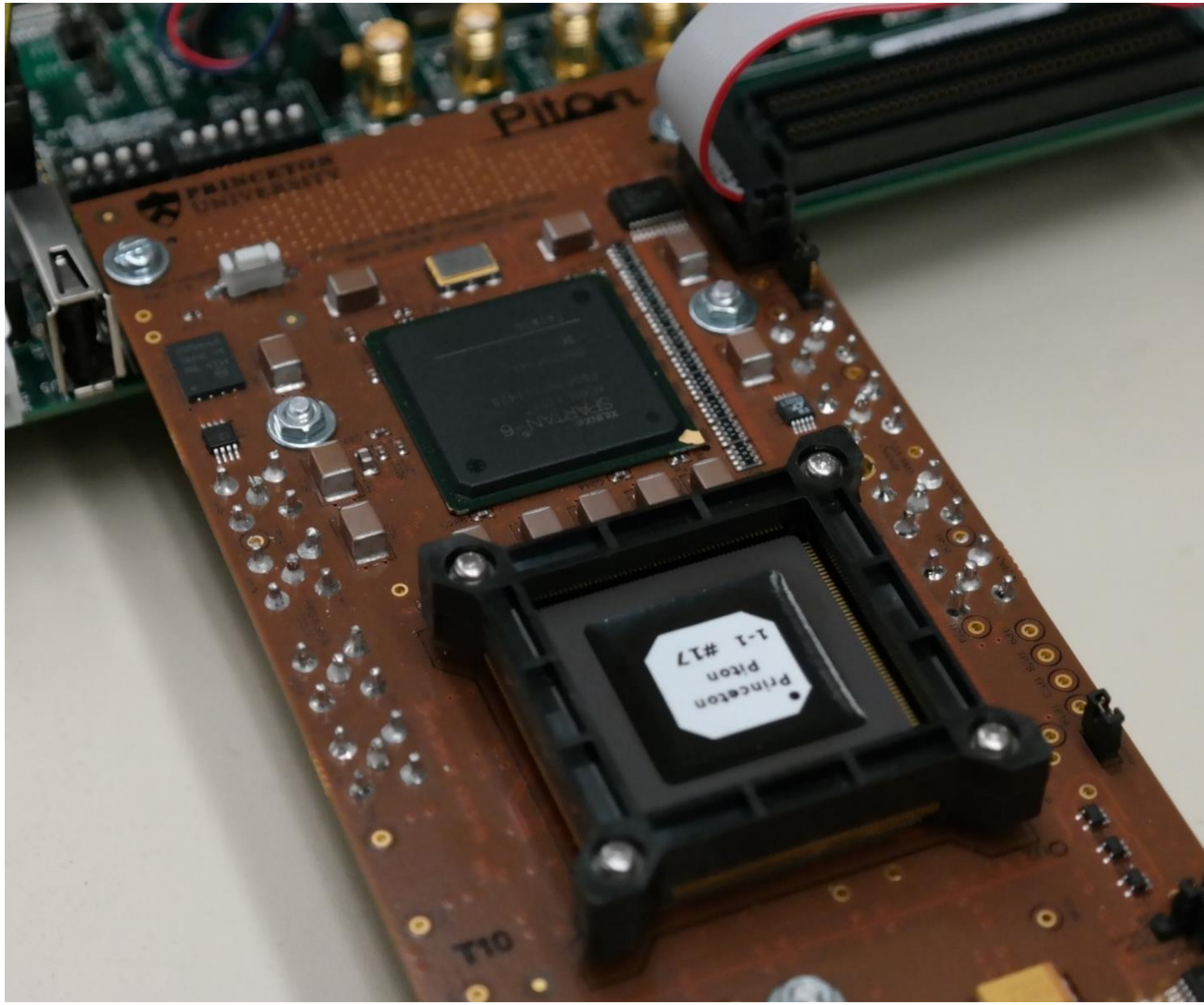


Piton: A Manycore for the Data Center and IaaS Clouds



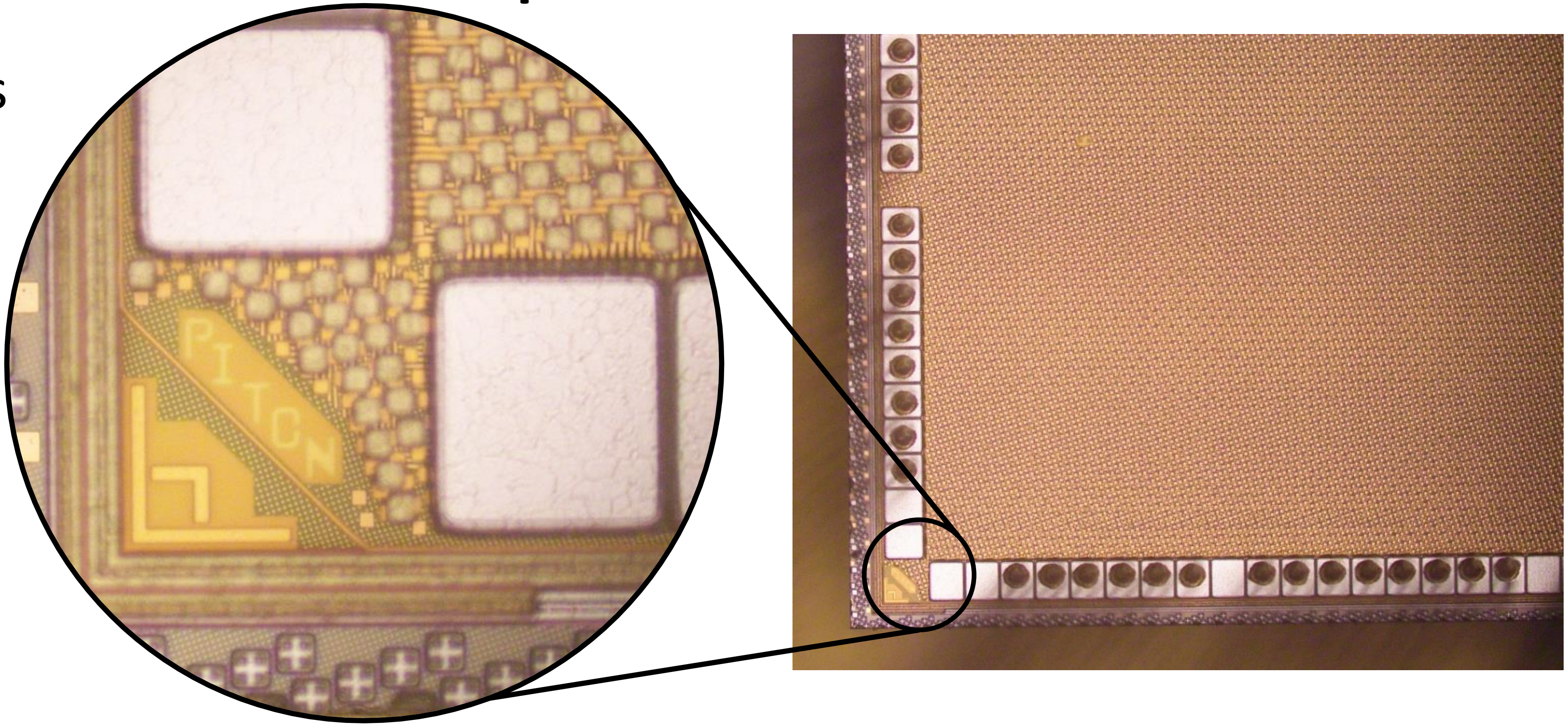
Piton Manycore Processor

- Manycore targeted at Cloud and WSCs
- SPARC V9 64-bit ISA. Boots standard OS
- Modern 64-bit NoC and tiled design for scalability
  - RTL scales to 65K cores intra-chip, 500 million cores per system
- Directory-based MESI cache coherence at distributed, shared L2 cache
  - NoCs and coherence protocol extend off-chip to support system wide shared memory
  - Coherence domains reduce directory storage and communication latency
- Multithreaded core for throughput and energy efficiency
- Energy efficient “drafting” mode
  - Reduce switching activity, I-cache accesses, and fetch bandwidth
- Memory bandwidth provisioning for IaaS
  - Charge commensurately in shared Cloud based systems



Piton Chip Stats

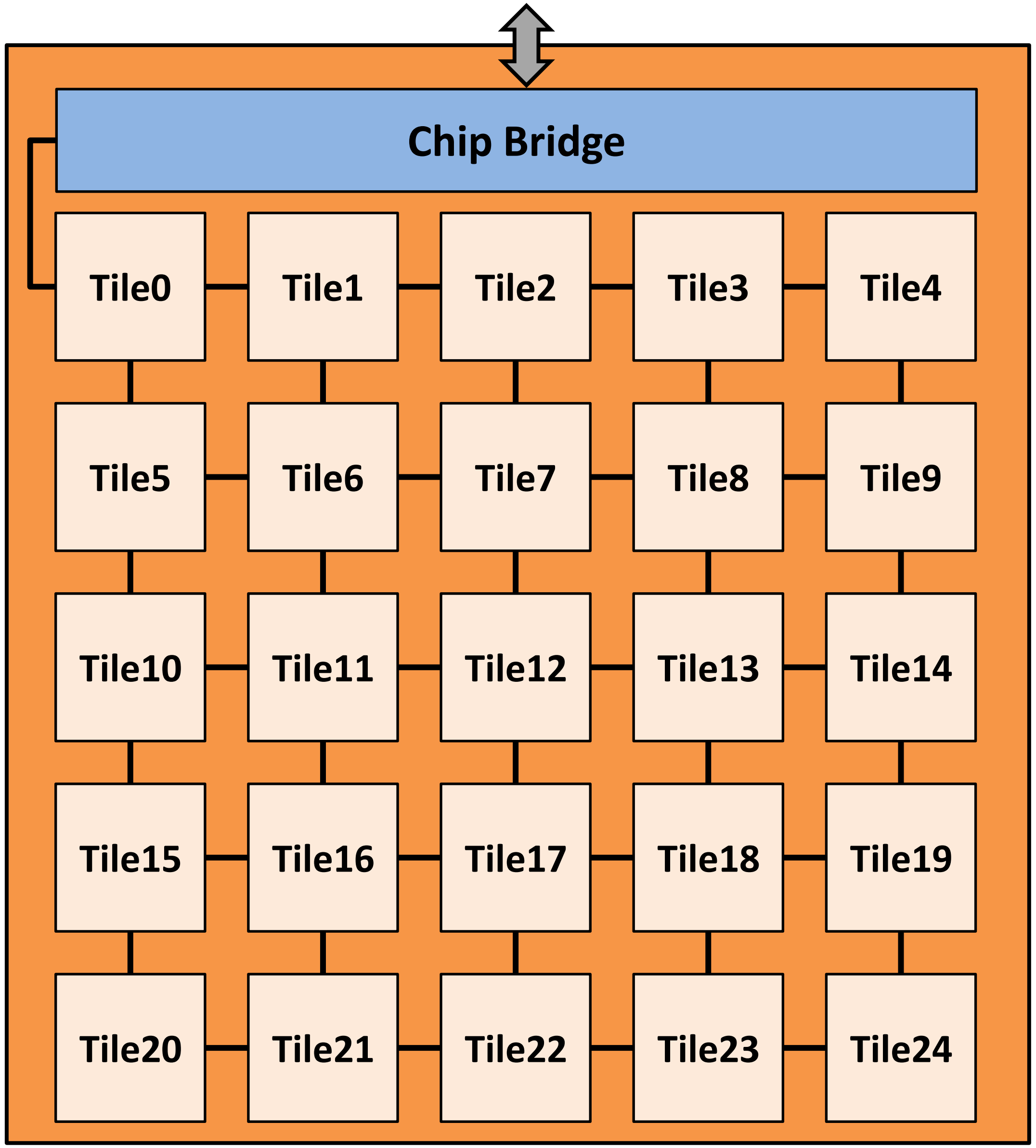
- IBM 32nm SOI process
  - Fabricated by IBM
- $36mm^2$ die (6mm x 6mm)
- 460 million transistors
- 1GHz target clock frequency @ 0.9V
- Among the largest chips built in academia



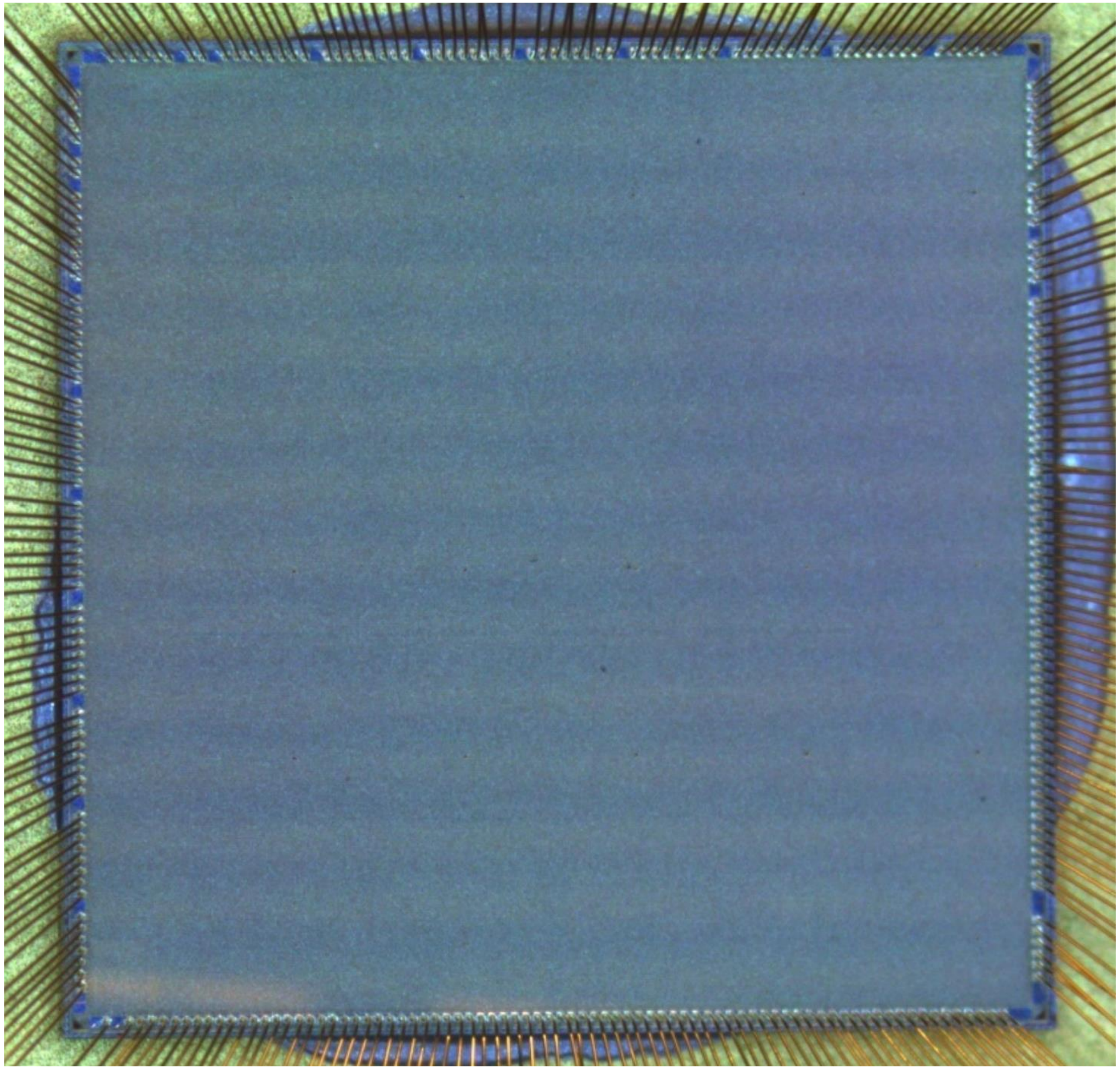
- Received silicon and has been tested working in lab
- 208-pin QFP wirebonded package with epoxy encapsulation
  - ~50% double wirebonds for power and ground

Piton Architecture Overview

- 25 tiles connected in 5x5 2D mesh topology
- 64-bit NoC interconnect
  - Dimension ordered routing for deadlock free network
  - 3 physical networks for protocol level deadlock avoidance
  - 1 cycle/hop latency
  - Credit-based flow control
- Cache coherence across all tiles maintained at shared, distributed L2 cache
- Chip bridge – off-chip interface
  - 2 32-bit unidirectional links
  - Multiplexes 3 physical NoCs into virtual channels for pin-limited communication
  - At 350MHz target frequency – 2.8GB/s
- Chip bridge extends NoCs and coherence protocol off-chip
  - Support for up 8k chips per system or 200k cores
  - All cores can share memory!

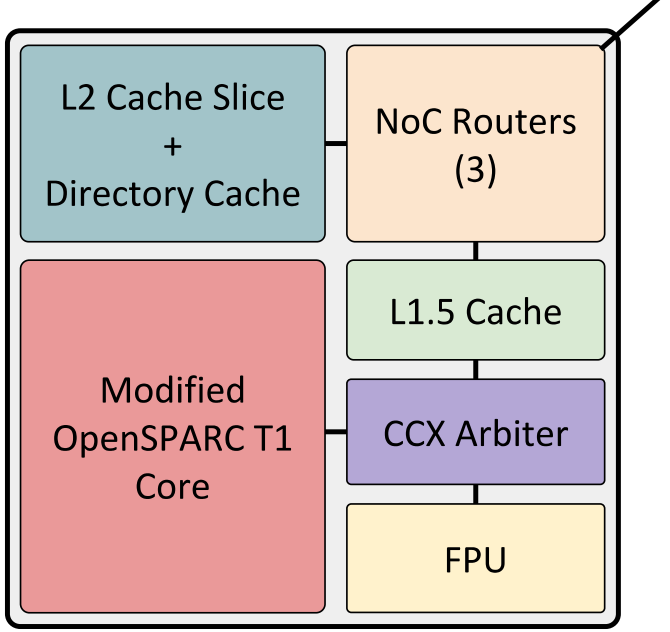


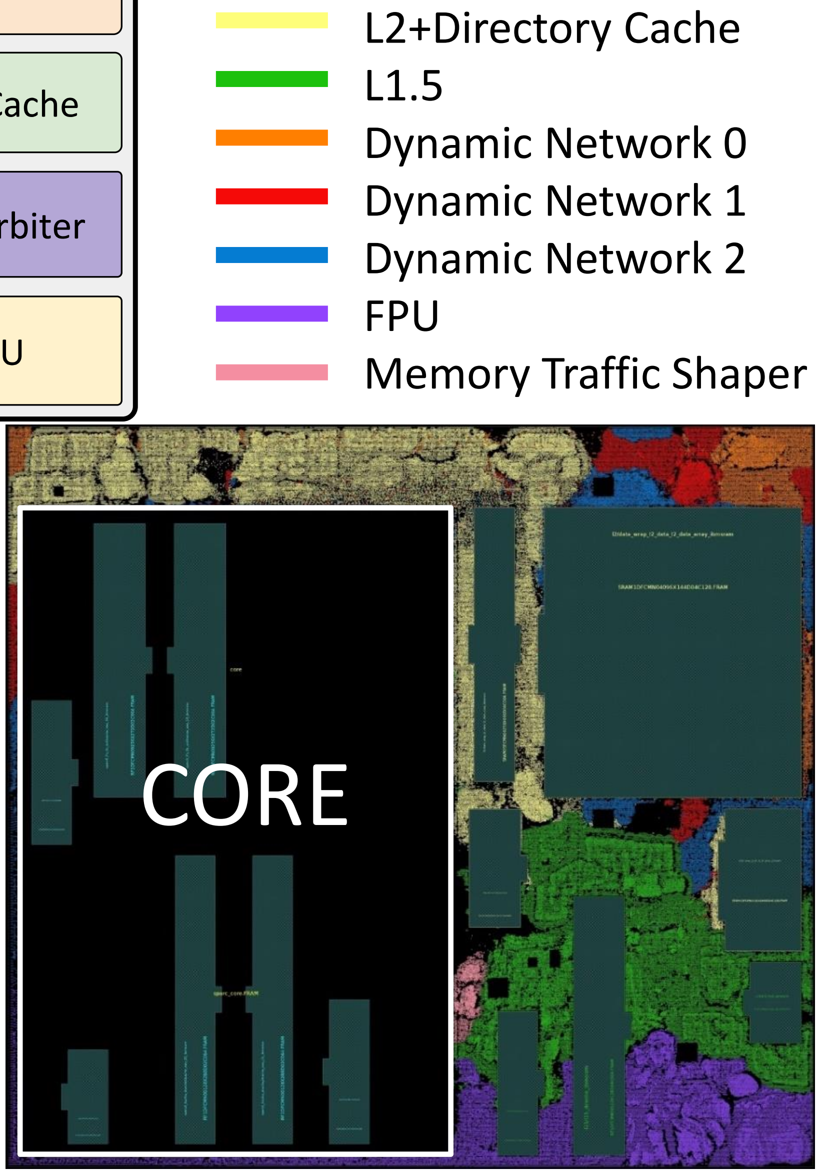
Piton Layout and Die



Piton Tile Architecture

- Modified OpenSPARC T1 Core
  - 2-way multithreaded (50 threads per chip)
  - Drafting mode for energy efficiency
- L1.5 Cache – 8KB private cache
  - Reduces bandwidth requirement to the shared distributed L2 cache
- L2 Cache – 64KB slice in each tile
  - Shared, distributed cache
  - Integrated directory cache for MESI coherence protocol
- 3 NoC routers
- Floating-point unit per tile from OpenSPARC T1
  - IEEE 754 compliant, fully pipelined except multiply and divide
  - Move-type FP instructions executed by core
- Memory traffic shaper
  - Provisions memory bandwidth on a per-core basis
  - Located between L1.5 and L2 in cache hierarchy





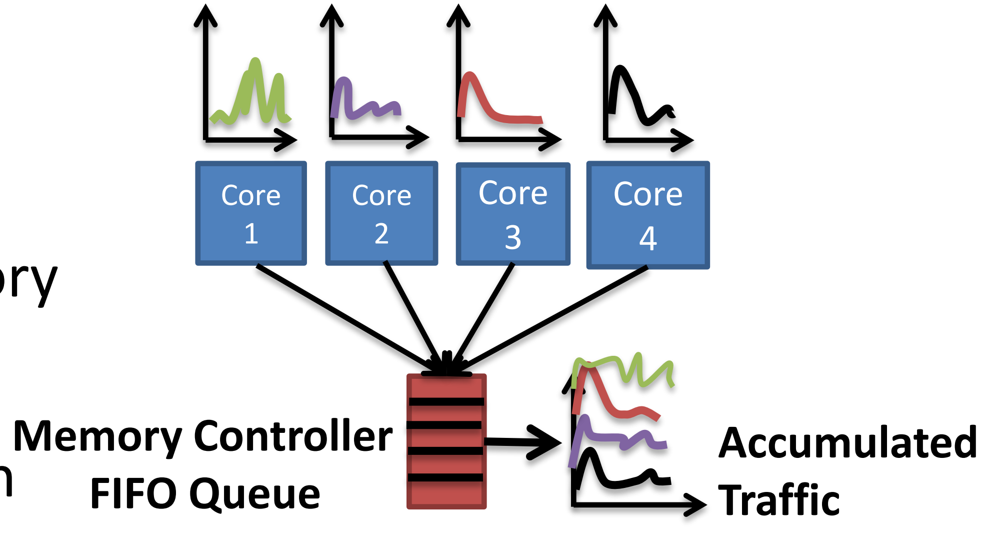
Memory Inter-arrival Time Traffic Shaper

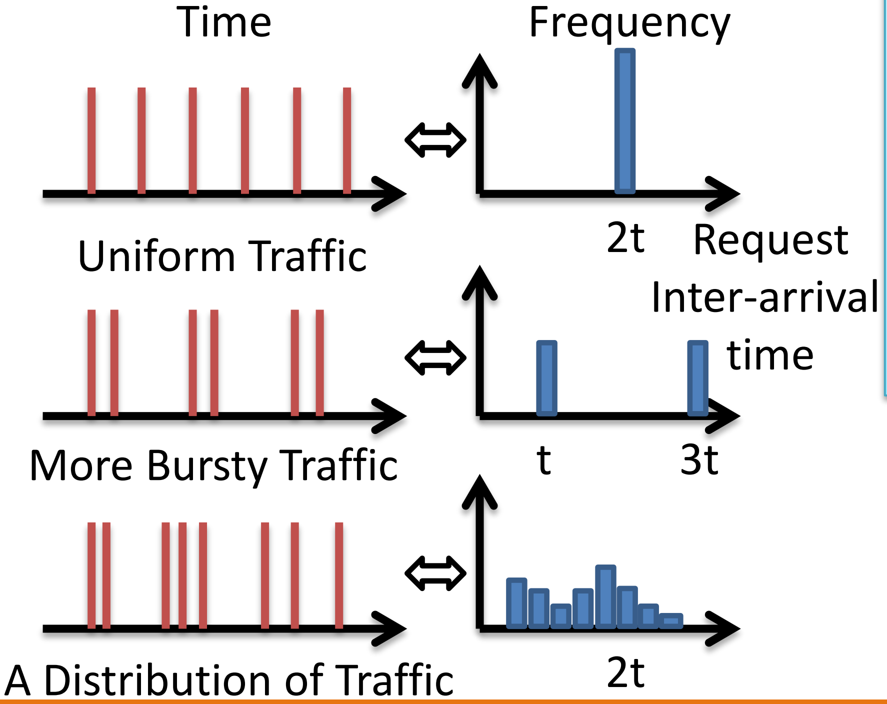
[Zhou, ISCA 2016]

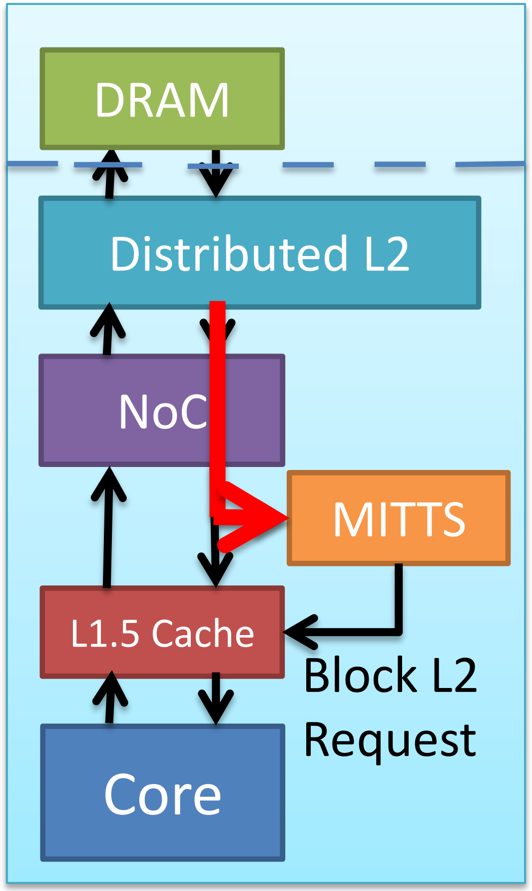
Problem: Off-chip memory bandwidth is key limited resource and applications do not share well

Solution: Restrict core/apps memory bandwidth to fit a particular inter-arrival distribution

- Shapes memory traffic based on temporal distance of requests (Inter-arrival time)
- Enables provisioning of memory bandwidth based on burstiness and bandwidth
- Shapes on per-core or per-application basis
- Distributed hardware
  - Located after L1.5
  - Shapes using hit/miss information from L2

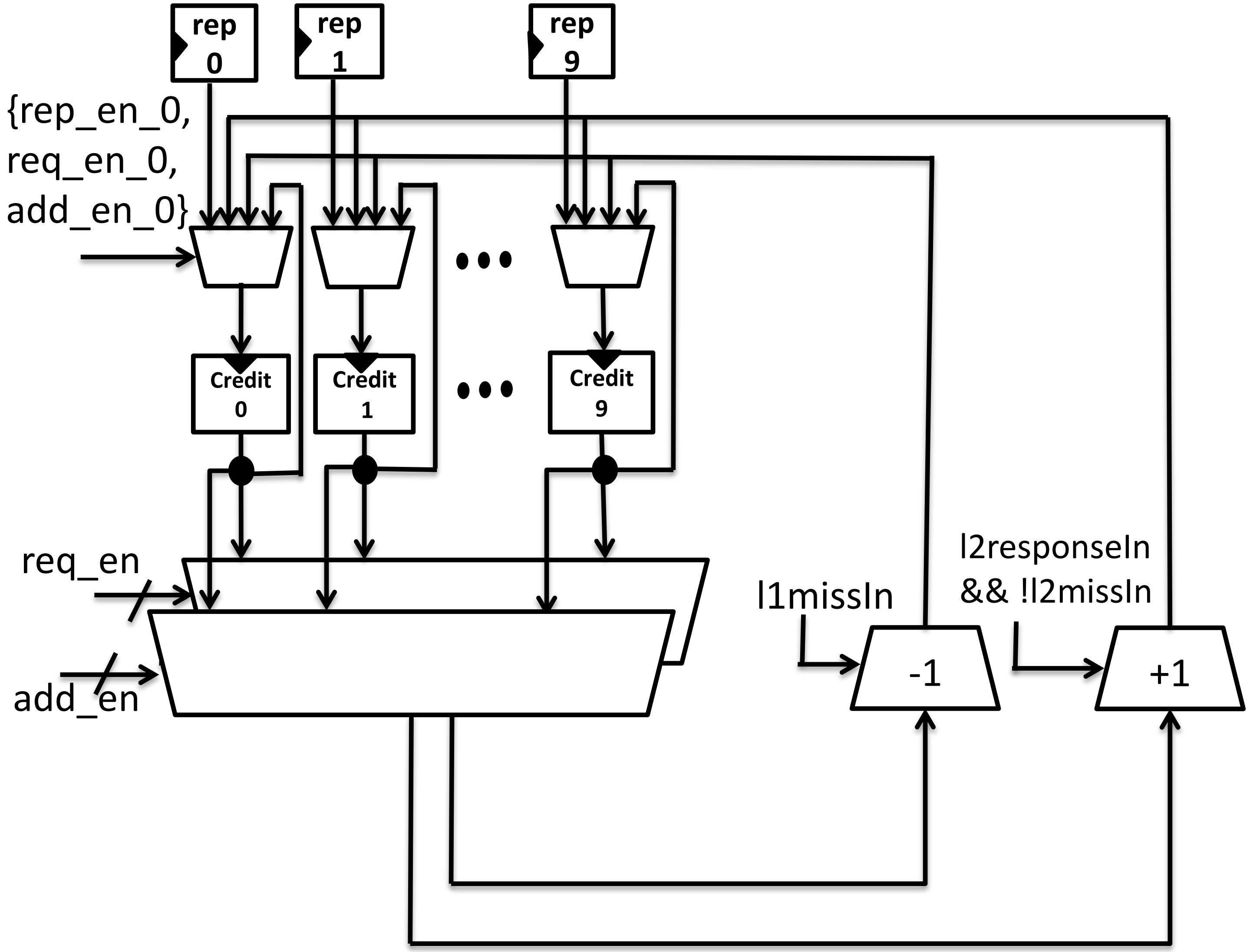




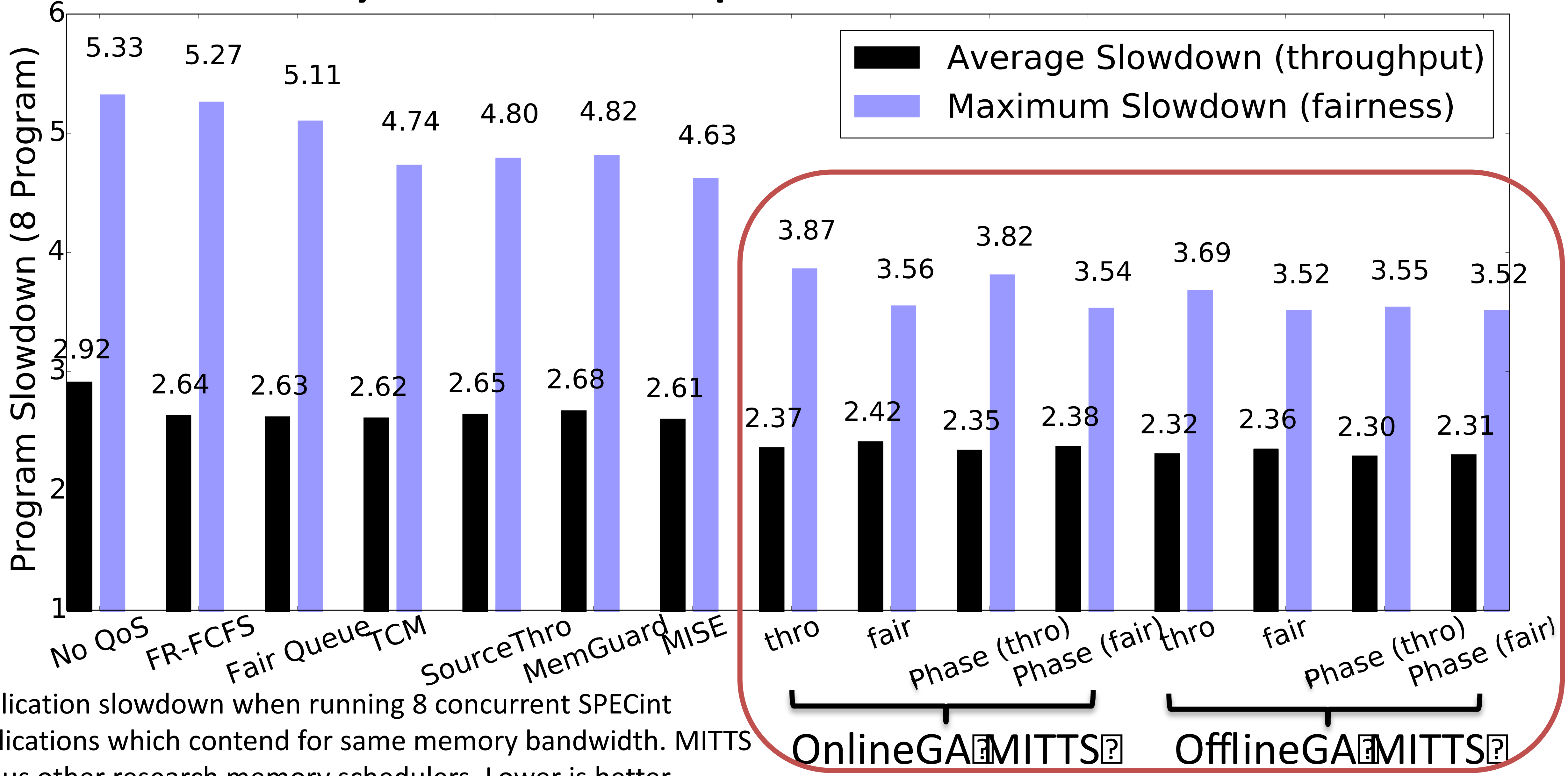


Memory Traffic Shaper Implementation

- Bin-based hardware
  - Array of bins contain credits for requests
  - Each bin represents an inter-arrival time
  - Stall memory transaction if not enough credits
  - Credits periodically replenished
- Speculates L2 miss, Rollback on Hit
  - Assume L1.5 miss is a L2 miss
  - Add back credits on L2 hit
  - Store the bin number per L1.5 miss
- Area
  - Less than 0.9% of tile area
  - 10 bins of 10 bits each



Memory Traffic Shaper Simulation Results



Application slowdown when running 8 concurrent SPECint applications which contend for same memory bandwidth. MITTS versus other research memory schedulers. Lower is better.

OpenSPARC T1 Core

- 6 Stage in-order pipeline
- 2-way multithreading to increase throughput and hide memory latency
- L1 instruction cache - 16KB, 4-way set associative, 32B line size
- L1 data cache - 8KB, 4-way set associative, 16B line size
- Implements SPARV9 ISA – Standard tool chain and OS compatible
- Energy efficient drafting mode

Execute
Regfile x 2
Floating-Point FE
Instruction Fetch
Load-Store
Multiplier
Trap Logic

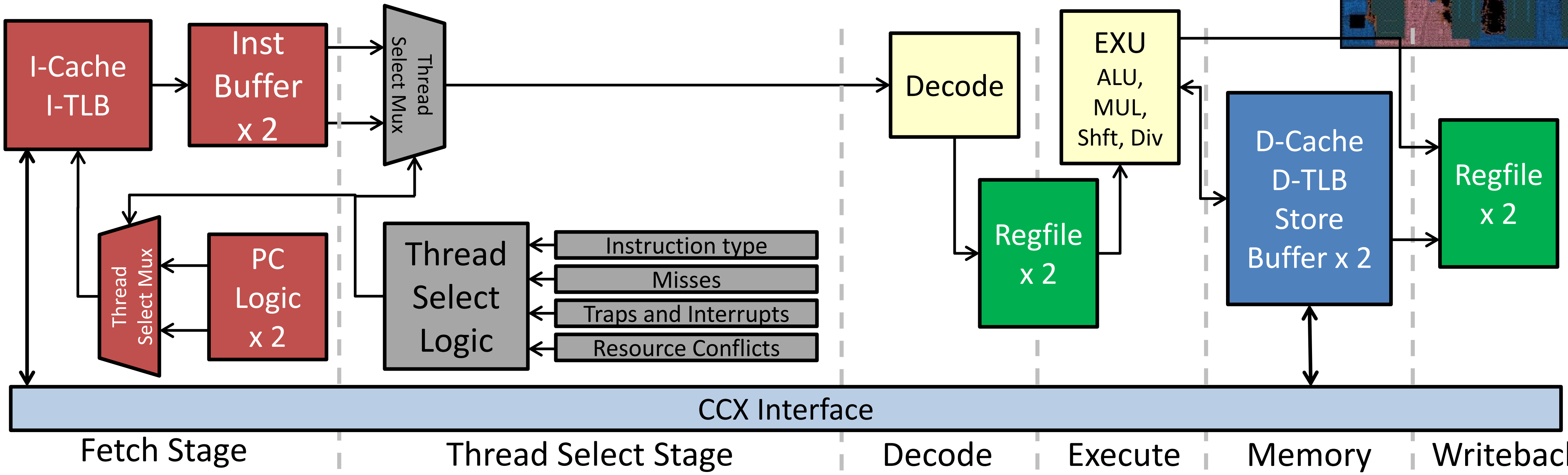
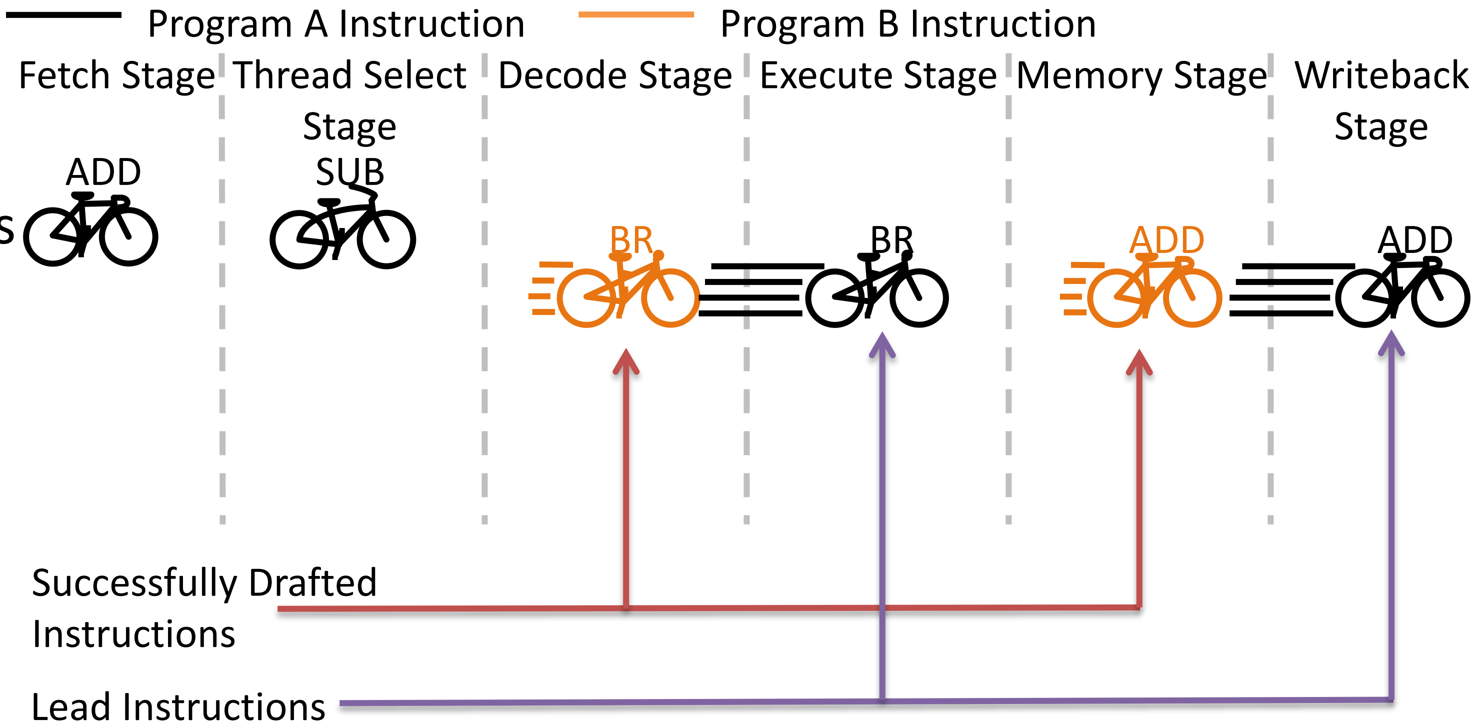


Diagram adapted from OpenSPARC T1 Microarchitecture Specification, Figure 1-2
http://www.oracle.com/technetwork/systems/opensparc/t1-01-opensparct1-micro-arch-1538959.html

Core Energy Efficient Drafting Mode

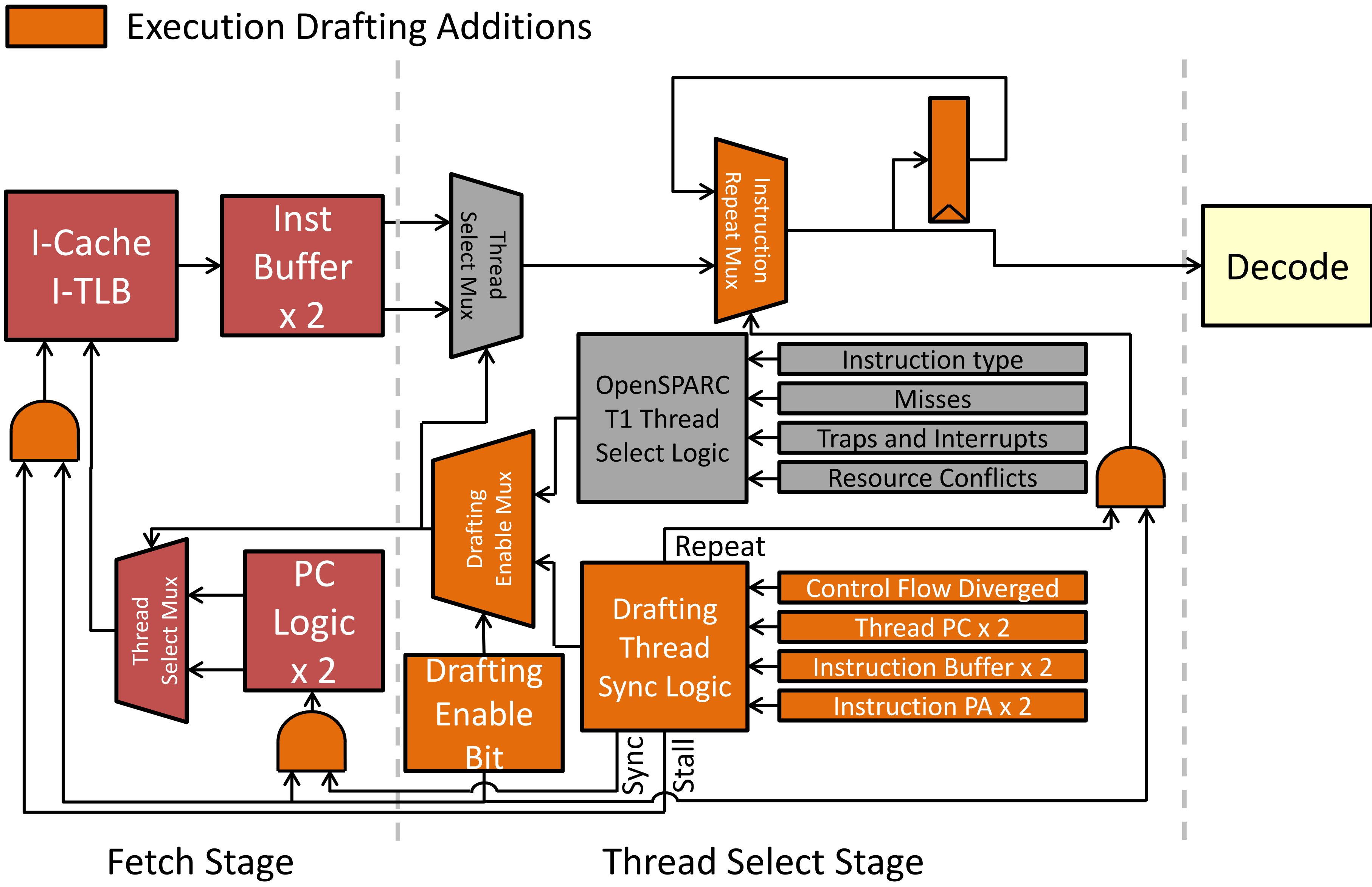
[McKeown, MICRO 2014]

- Aggregate similar or identical code to multithreaded core
- Align execution points of threads to identical instructions
  - Active synchronization
    - PC-based, random, hybrid
    - Can result in small performance overhead
- Two sources of energy savings
  - Drafting – issue identical instructions consecutively
    - Reduces activity factor on control and data signals
  - Disable fetch if instruction streams are the same
- **Goal: Maximize** $\frac{Performance}{Energy}$



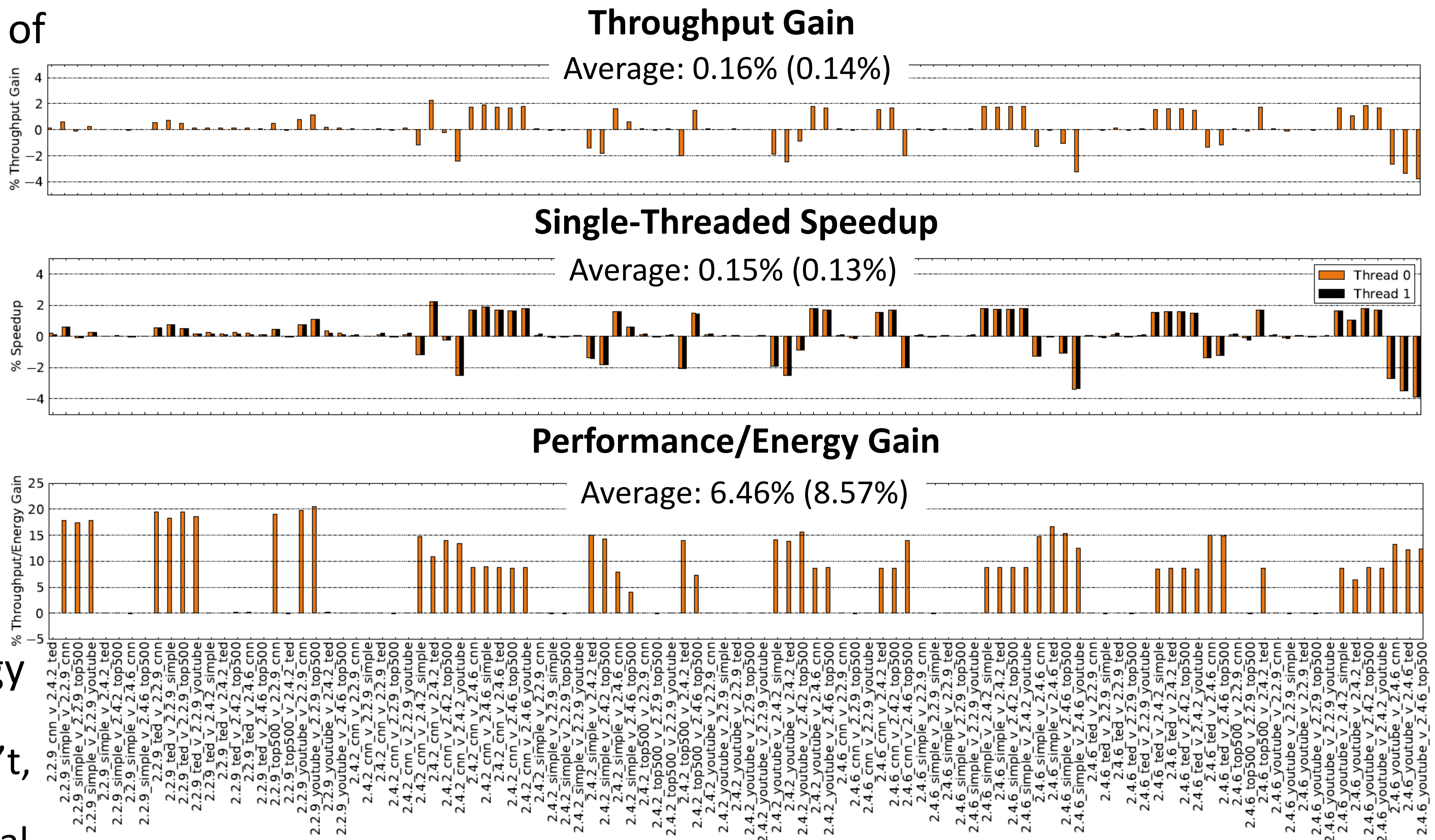
Drafting Mode Implementation

- Addition of thread synchronization logic
- Enable bit multiplexes standard thread select with drafting thread select
- Additional logic to disable fetch and repeat an instruction for another thread



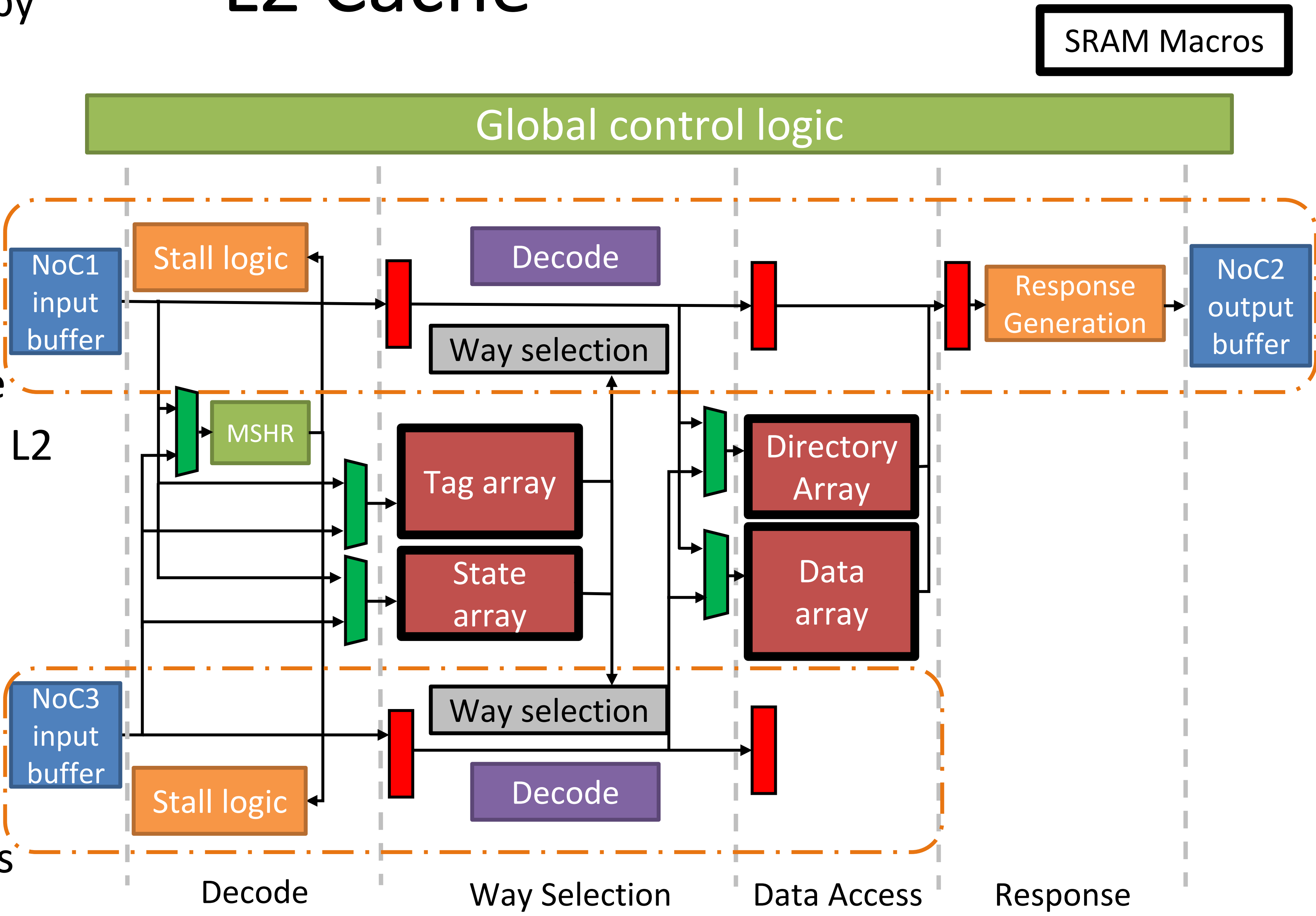
Drafting Mode Simulation Results

- Different versions of Apache hosting different sites
- Two averages:
  - Excluding same program same input (Including same program same input)
- Small impact on throughput and single threaded performance
- Large benefit to throughput/energy
  - Even in cases where there isn't, performance impact is minimal



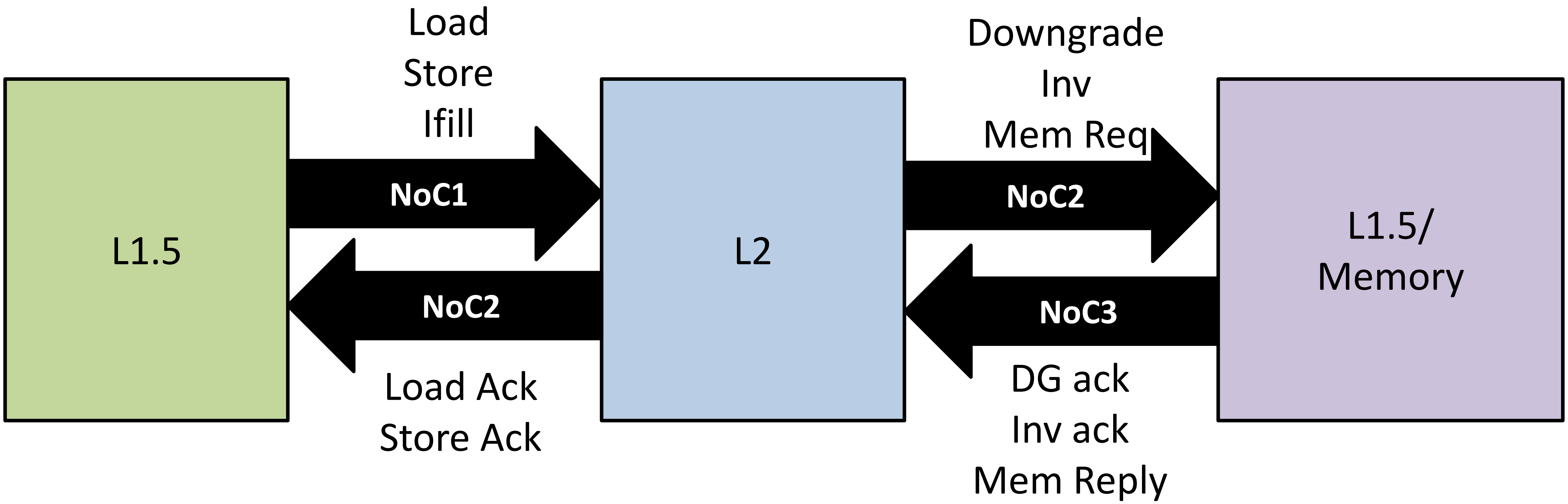
L2 Cache

- Distributed cache shared by all tiles
  - 64KB slice per tile
  - 1.6MB aggregate cache per chip
- 4-way set associative
- 64-byte line size
- Integrated directory cache
- Configurable cache line to L2 slice mapping:
  - Low, middle, or high-order address bit interleaving
  - Bitwise AND of low and middle-order address bits
- 4-stage dual pipelines
- SRAM macros are shared between the two pipelines
- 5-cycle hit latency



Directory-based MESI Coherence Protocol

- ~35 different message types
- 3 physical NoCs with point-to-point ordering to avoid deadlock
- Notable features:
  - Silent eviction in exclusive (E) and shared (S) states
  - No acknowledgment on L1.5 writeback

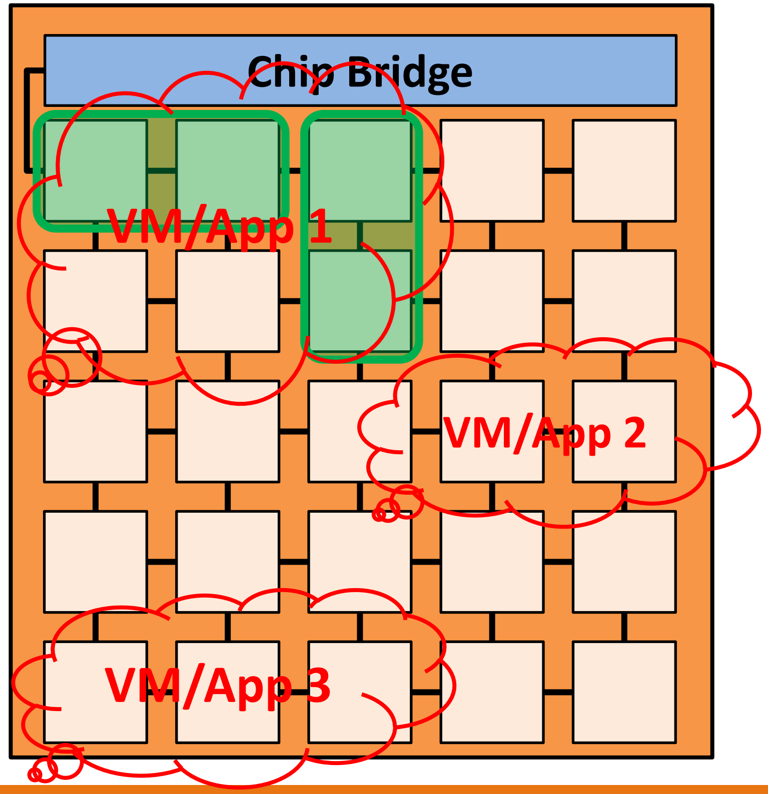


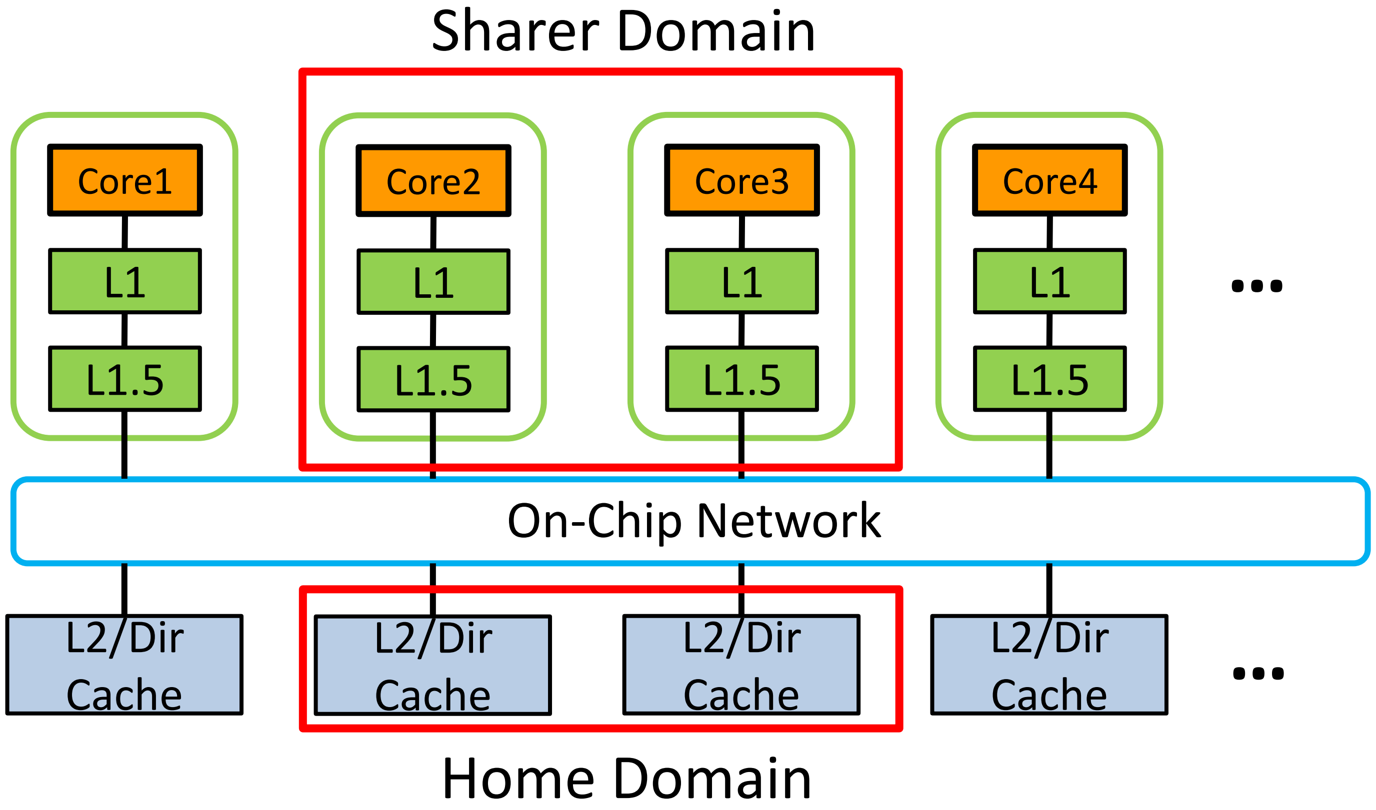
Coherence Domains

[Fu, MICRO 2015]

- VM/Application or page level coherence domains
  - Coherence only needs to be maintained within a domain
- Domains are created and modified at runtime

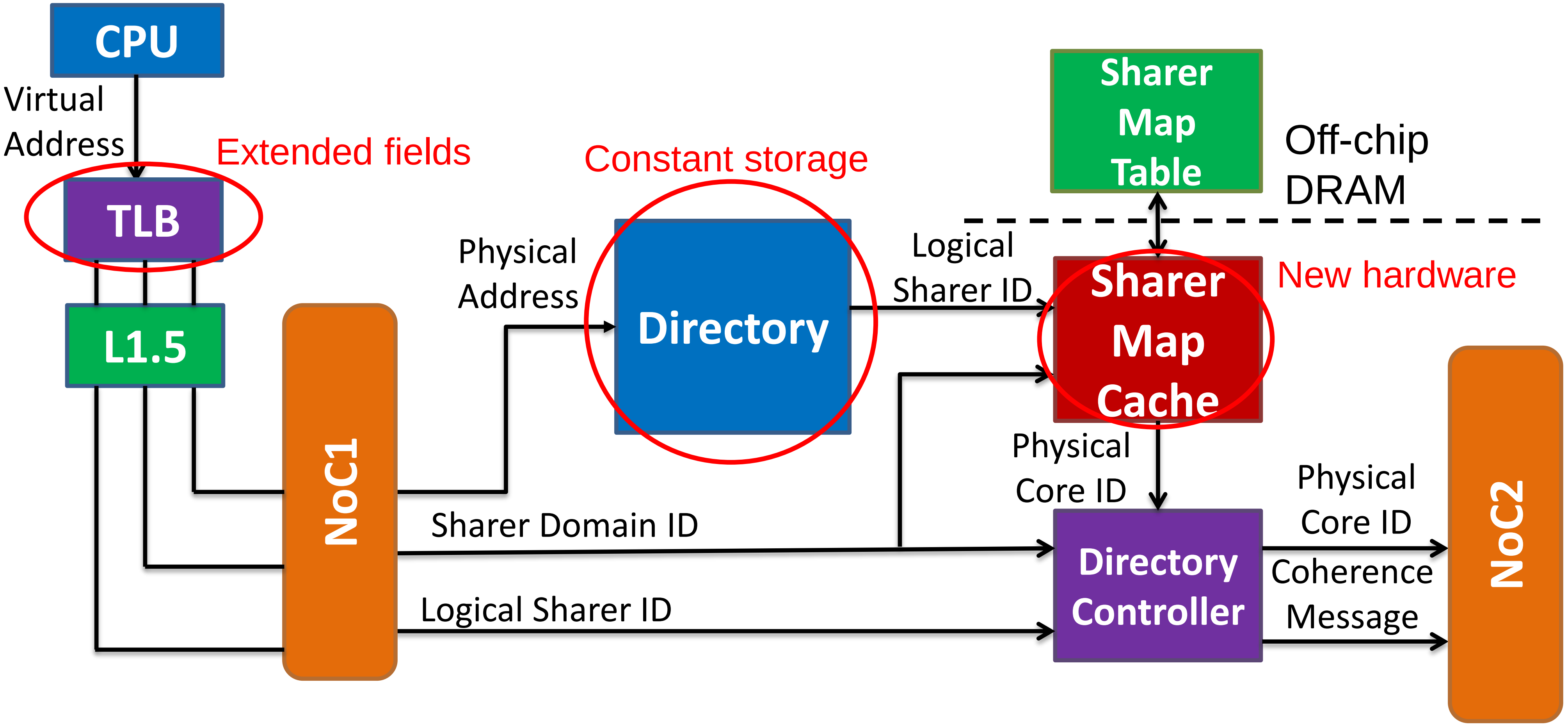
- Limit the maximum coherence domain size to achieve constant storage overhead (64 max sharers in Piton)
- Restrict home domain placement to reduce communication latency





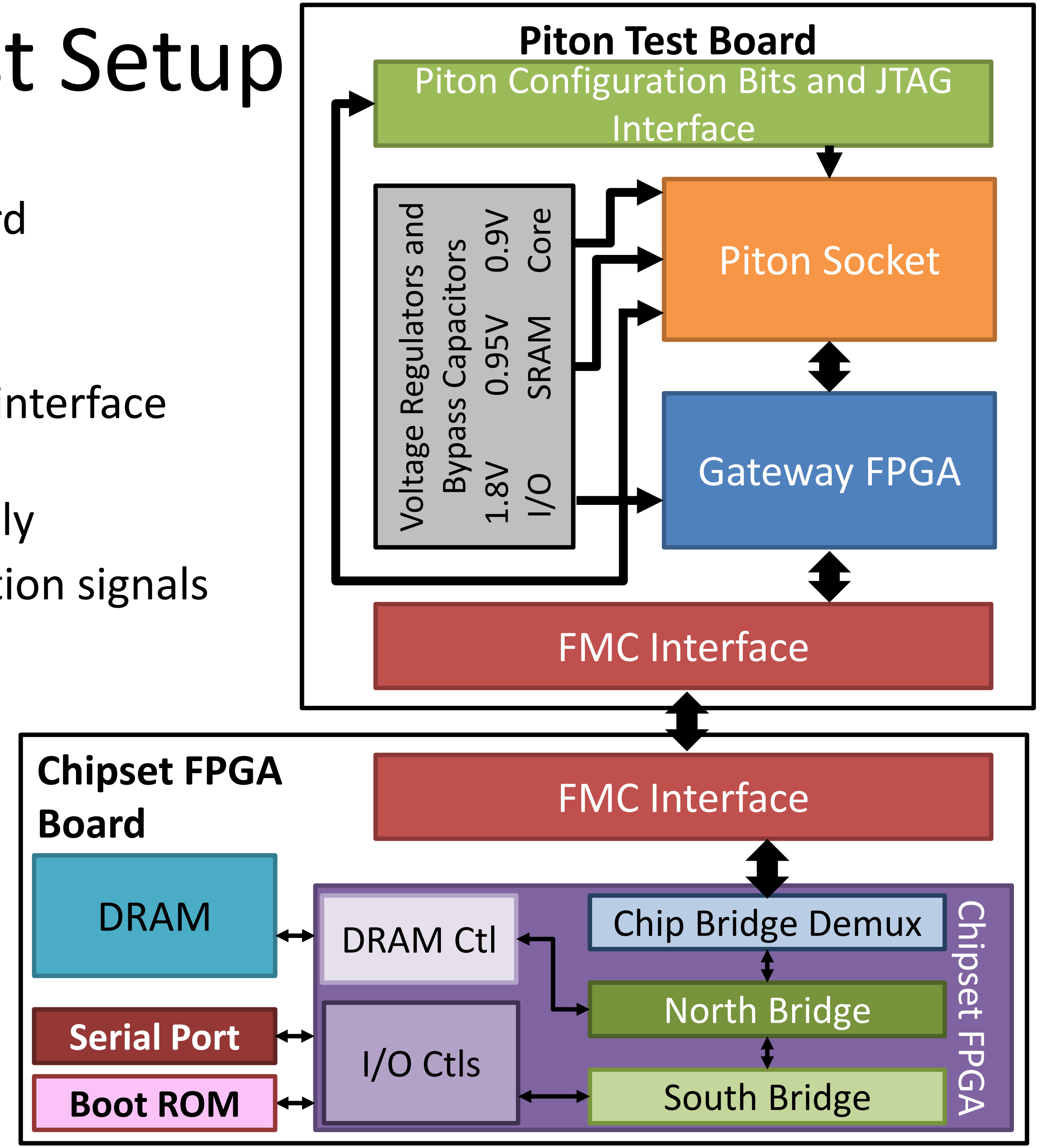
Coherence Domains Implementation

- Additional indirection layer called Sharer Map Cache (SMC)
  - Located after coherence directory
- TLB entries extended with coherence domain IDs
  - Coherence domain IDs managed by SW
- Sharer vector and domain ID index into SMC
  - Cache is backed by full mapping table in DRAM
- SMC outputs physical core IDs of sharers
  - Used in coherence messages

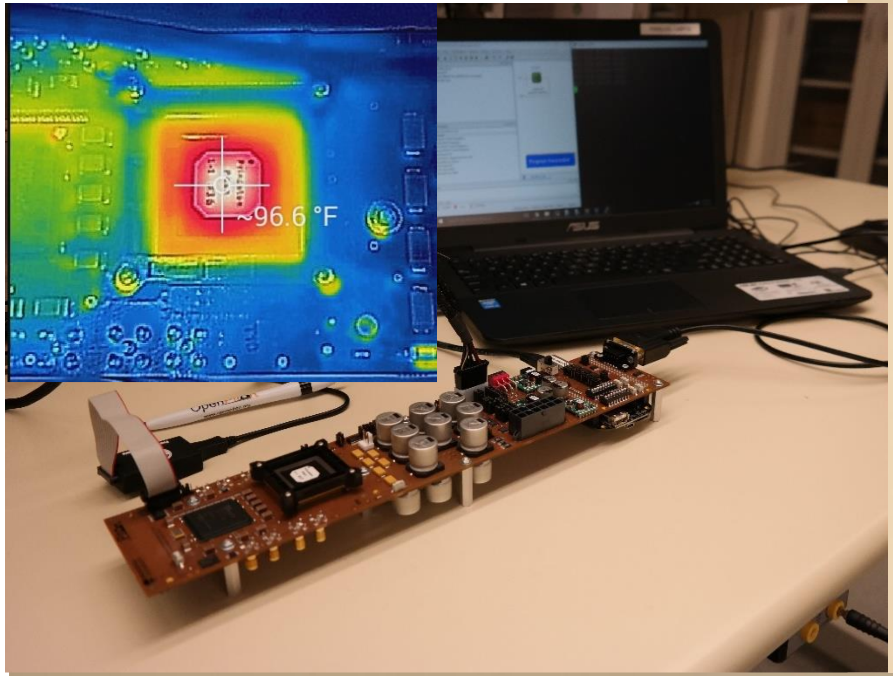


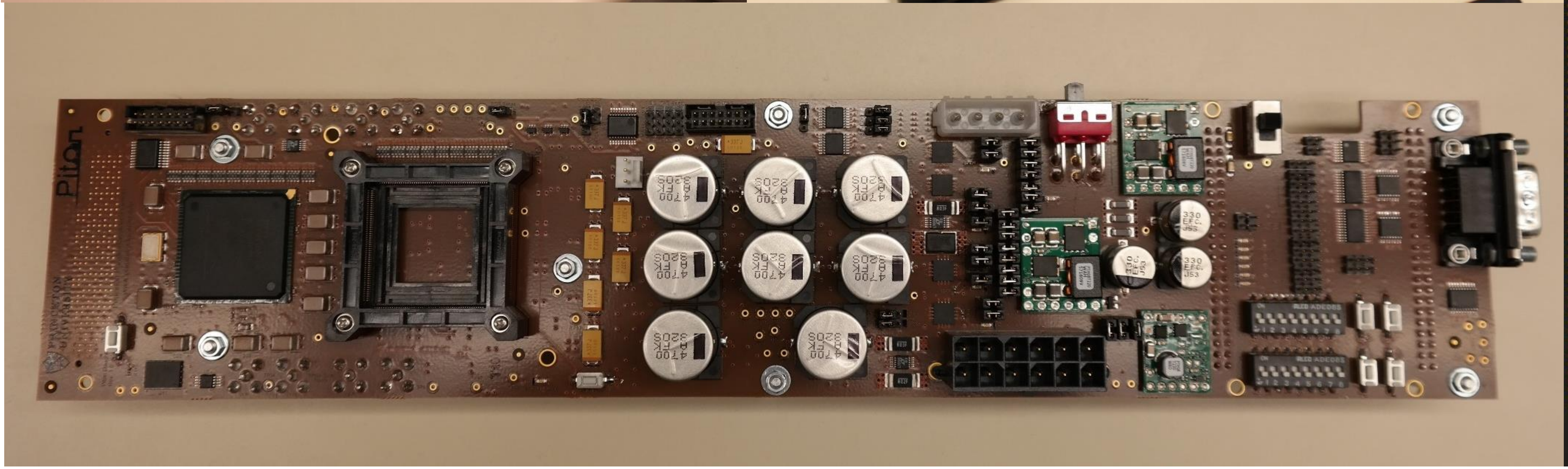
Piton Test Setup

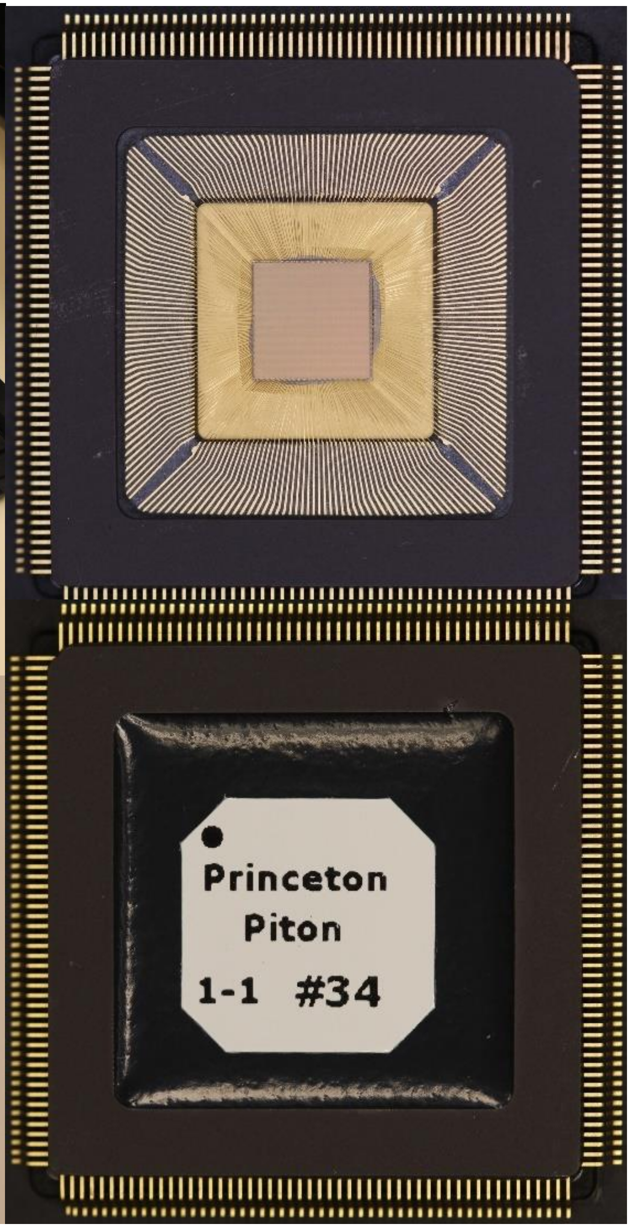
- Custom Piton Test Board
  - Modified from Double Trouble Daughterboard
    - Prof. Michael Taylor @ UCSD – www.bjump.org
  - QFP 208 socket for Piton
  - Gateway FPGA to transmit Piton chip bridge interface over FMC connector to chipset FPGA board
  - Voltage regulation from 12V ATX power supply
  - Access to Piton JTAG interface and configuration signals
  - Debugging interfaces (UART)
- Chipset in Host FPGA board
  - Most any FPGA board with a FMC connector
    - Genesys2, ML605, VC707
  - Includes:
    - Chip bridge demux
    - North and south bridges
    - DRAM and I/O controllers



Piton Test Setup







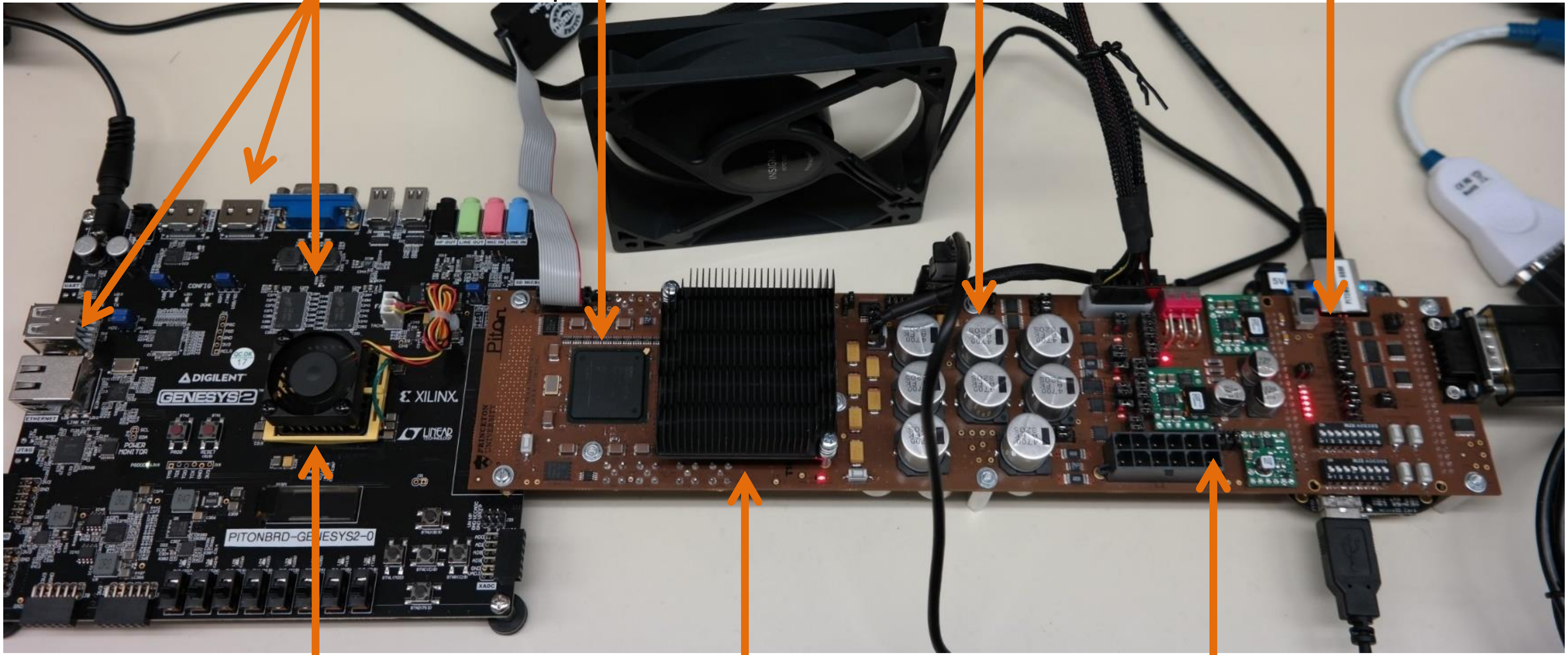
Piton Test Setup

DRAM + I/O

Gateway FPGA
Spartan 6

Bulk Decoupling

Misc. Configuration



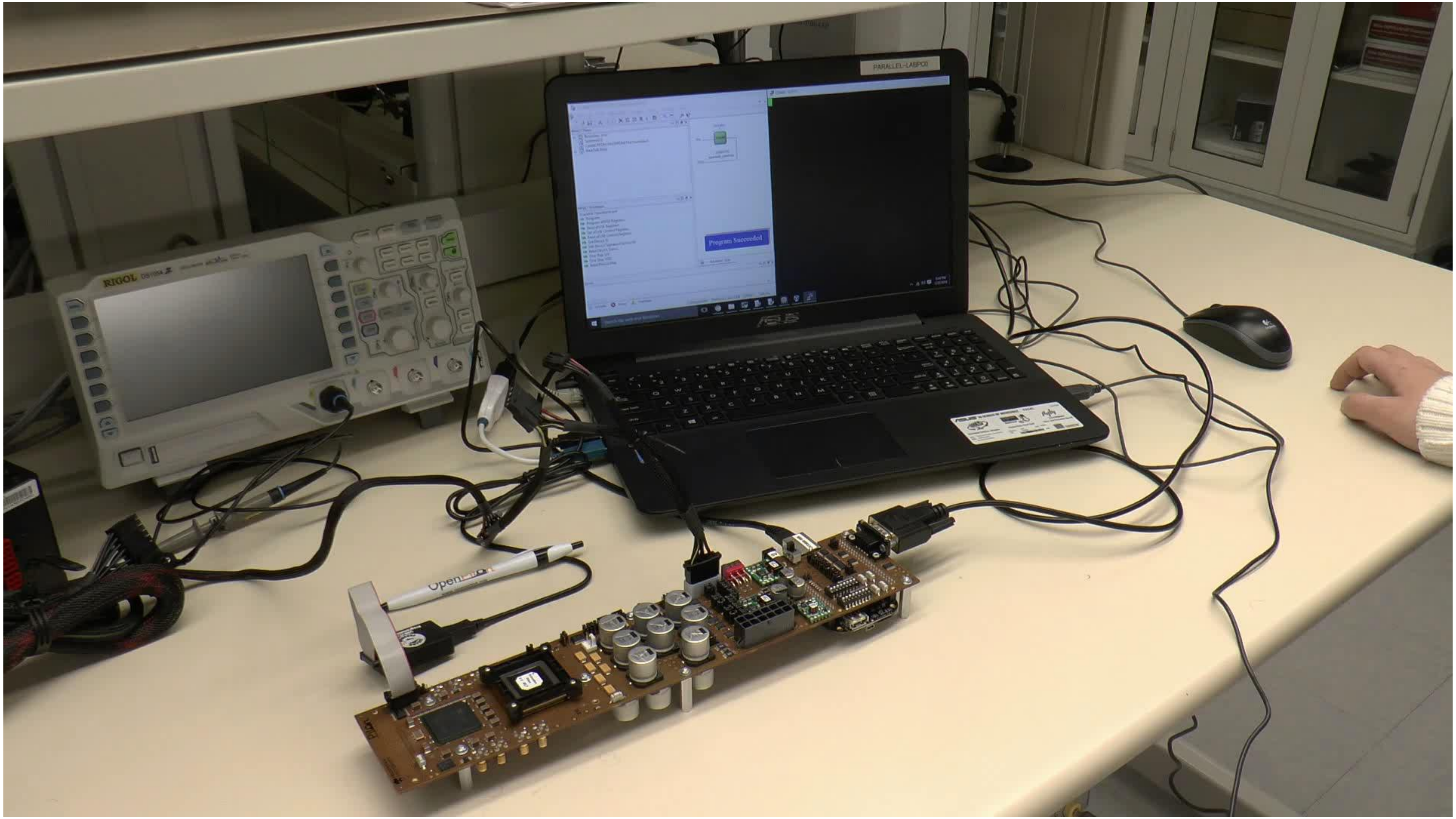
Chipset FPGA
Kintex 7

Piton + Heat Sink

Power Supply

Piton Demo

Extended Demo: http://parallel.princeton.edu/piton/helloworld_demo.html

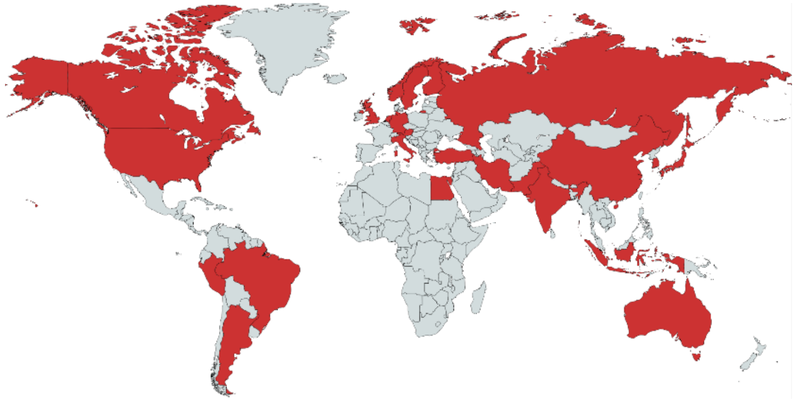


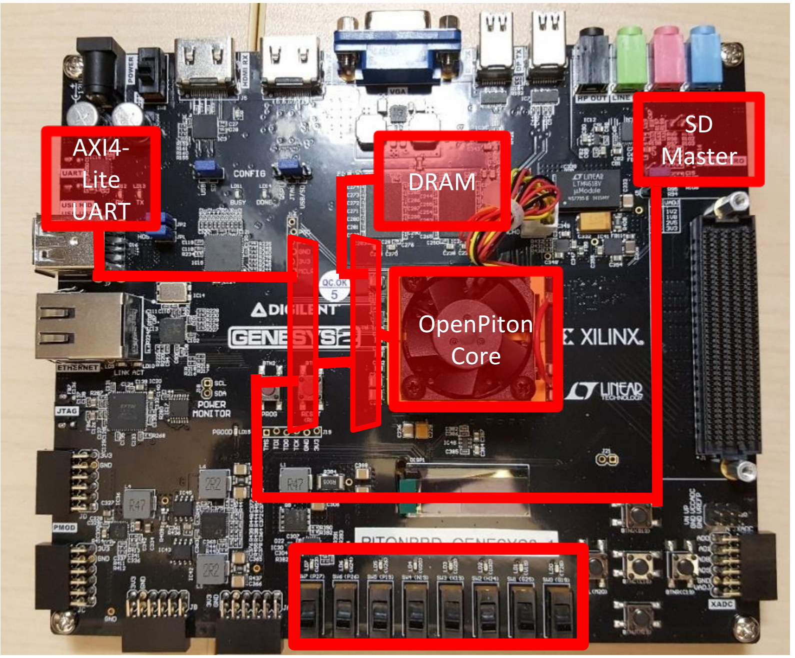
OpenPiton

- Open source release
  - RTL
  - Simulation infrastructure
  - Test/validation suite
  - FPGA synthesis
  - ASIC synthesis and backend
- Highly configurable
  - Scales to ½ billion cores
  - Configurable cache sizes
  - Configurable NoC topology

- Targets multiple FPGAs at different price points
- Great for:
  - Research in many domains
  - ASIC Tapeouts
  - Education
- BSD uncore and GPL core

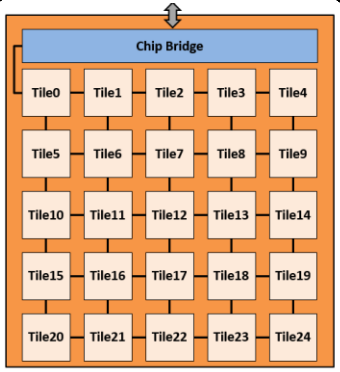
http://www.openpiton.org

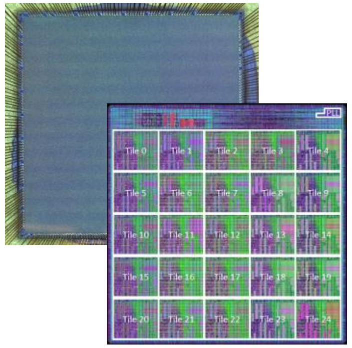


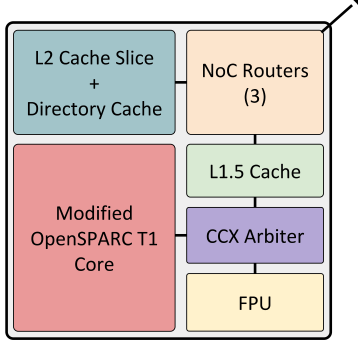


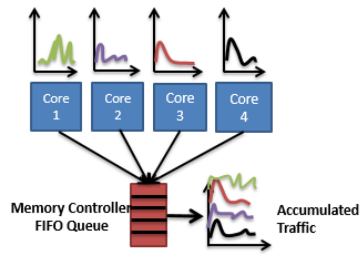
Piton: A 25-core Academic Manycore Research Processor

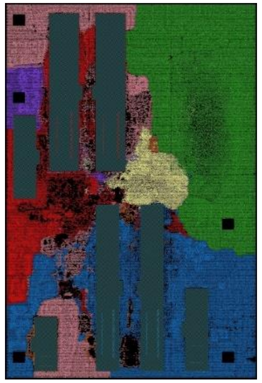
Michael McKeown, Yaosheng Fu, Tri Nguyen, Yanqi Zhou, Jonathan Balkind, Alexey Lavrov, Mohammad Shahrad, Samuel Payne, and David Wentzlaff

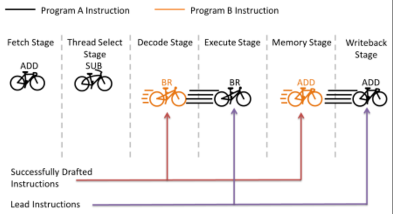


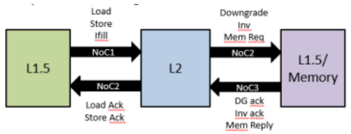


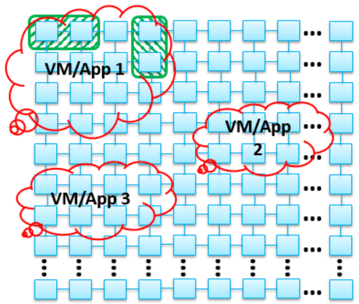


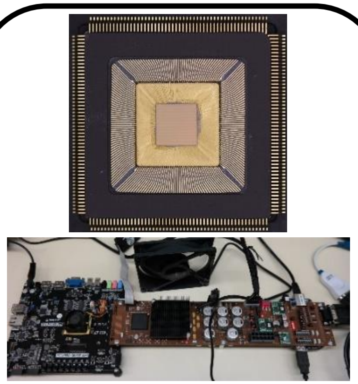












OpenPiton

http://www.openpiton.org

PRINCETON UNIVERSITY

PRINCETON School of Engineering and Applied Science

SUPPLEMENTAL SLIDES

Support

This work was partially supported by the NSF under Grants No. CCF-1217553, CCF-1453112, and CCF-1438980, AFOSR under Grant No. FA9550-14-1-0148, and DARPA under Grants No. N66001-14-1-4040 and HR0011-13-2-0005. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of our sponsors.

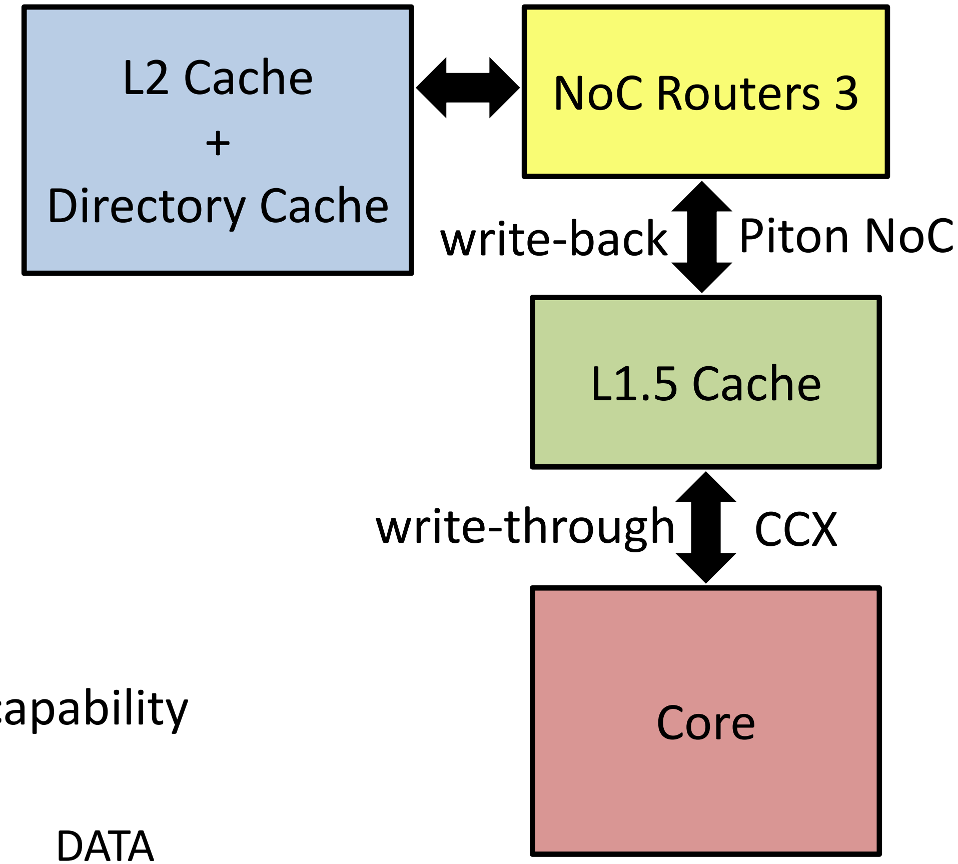
Team and Timeline

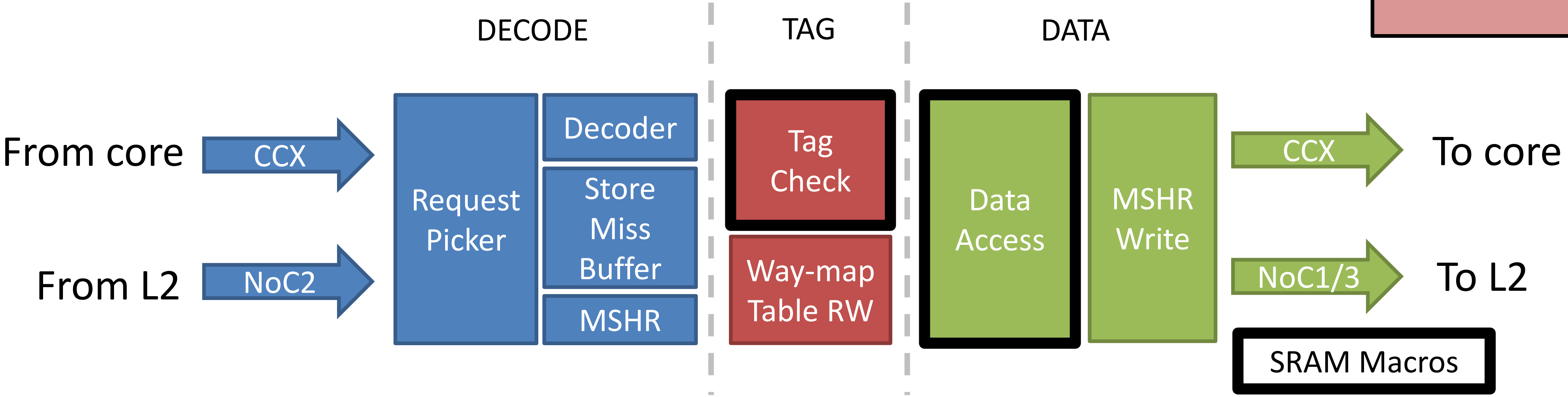
- Team
  - Started with 5 PhD students
  - Now 11 PhD students
  - ~3 undergraduates per year

- Timeline
  - Design, RTL, and Verification – 8 months
  - Physical design – 6 months
  - Test setup and Bring-up – 6 months, ongoing

L1.5 Cache

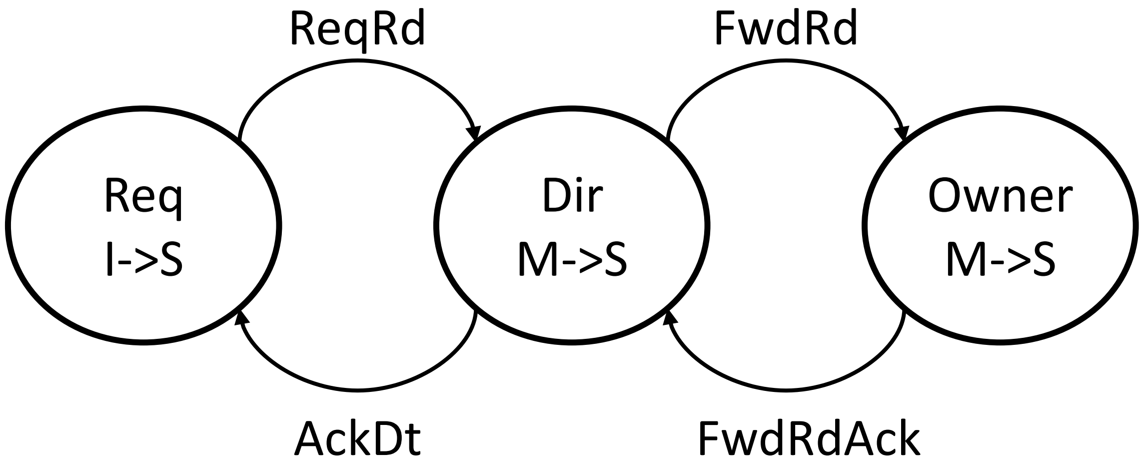
- 8KB, 4-way set associative, 16B line size
  - Same size as core's L1 data cache
  - Does not cache instructions
- 3-stage pipeline
  - Request decode, tag check, data access
  - 4 cycle hit latency
- Transducer and write-back layer
  - Transduces core's CCX interface to Piton NoC protocol
  - Encapsulates write-through L1 data cache with write-back capability
    - Reduces bandwidth usage to distributed L2 cache





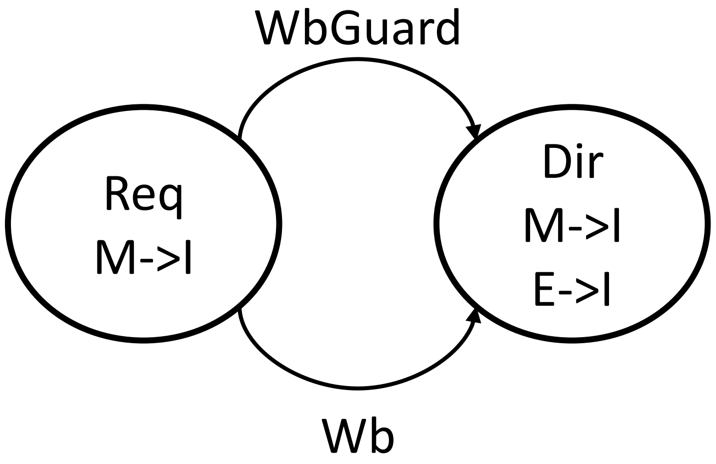
Directory-based MESI Coherence Protocol

- Four-hop message communication (no direct communication between private L1.5 caches)



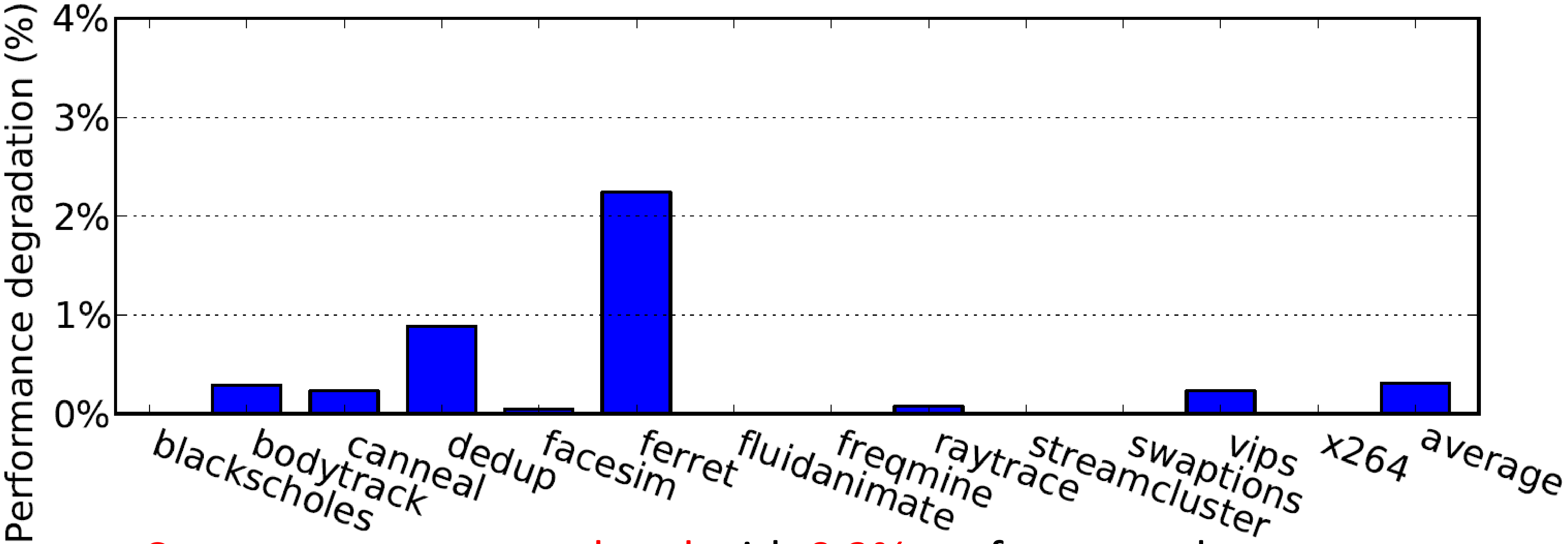
- Silent eviction in E and S states

- No need for acknowledgement upon write-back of dirty lines from L1.5 to L2



Coherence Domains Simulation Results

- Multi-program workload - PARSEC benchmarks with 16 threads
- 1024-core chip simulation – compared to baseline with 1024-bit sharer vectors and no coherence domains (same thread count and thread to core mapping)



- Constant storage overhead with 0.3% performance loss on average
- Performance overhead due to sharer map cache

Glossary of Terms

- Ack - Acknowledgment
- AckDt – Acknowledgment with Data
- ASIC – Application Specific Integrated Circuit
- CCX – CPU-Cache Crossbar
- DG Ack – Downgrade Acknowledgment
- Dir - Directory
- FMC – FPGA Mezzanine Card
- FP – Floating Point
- FPGA – Field Programmable Gate Array
- FwdRd – Forward Read Request
- FwdRdAck – Forward Read Acknowledgment
- GA – Genetic Algorithm
- IaaS – Infrastructure as a Service
- Ifill – Instruction Fill
- Inv – Invalidate
- Inv Ack – Invalidate Acknowledgment
- ISA – Instruction Set Architecture
- LLC – Last level cache
- Mem Req – Memory Request
- Mem Reply – Memory Reply
- MESI – Modified, Exclusive, Shared, Invalid
- MITTS – Memory Inter-arrival time traffic shaper
- NoC – Network on Chip
- OS – Operating system
- PC – Program Counter
- QFP – Quad Flat Pack
- Req - Requestor
- ReqRd – Read Request
- SMC – Sharer Map Cache
- SOI – Silicon on Insulator
- TLB – Translation Lookaside Buffer
- Wb – Writeback
- WbGuard – Writeback Guard
- WSC – Warehouse Scale Computer