The University of Texas at Austin
Electrical and Computer Engineering
Cockrell School of Engineering

Accelerating Dependent Cache Misses with an Enhanced Memory Controller

Milad Hashemi, Khubaib, Eiman Ebrahimi, Onur Mutlu, Yale N. Patt

June 21, 2016

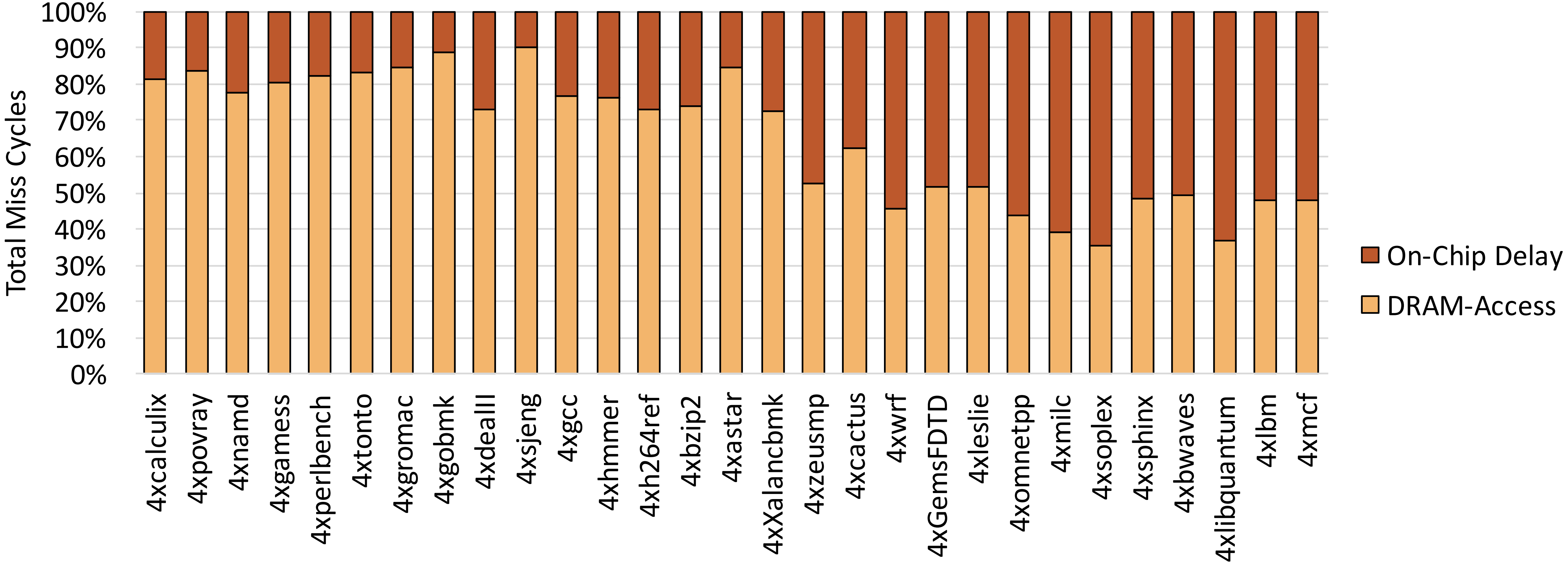
Overview

- Dependent Cache Misses
- Enhanced Memory Controller Microarchitecture
- Dependence Chain Generation
- EMC Performance Evaluation and Analysis
- Conclusions

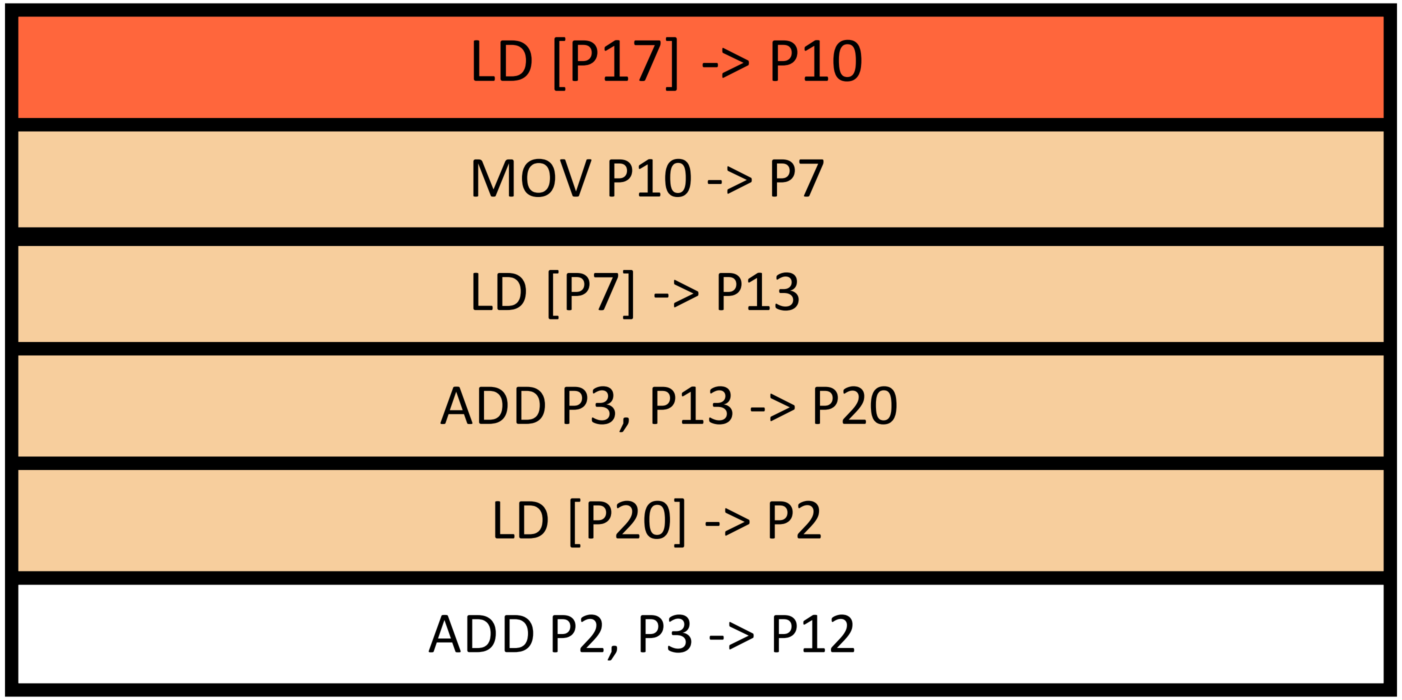
Effective Memory Access Latency

- The effective latency of accessing main memory is made up of two components:
  - DRAM access latency
  - On-chip latency

On-Chip Delay



Dependent Cache Misses



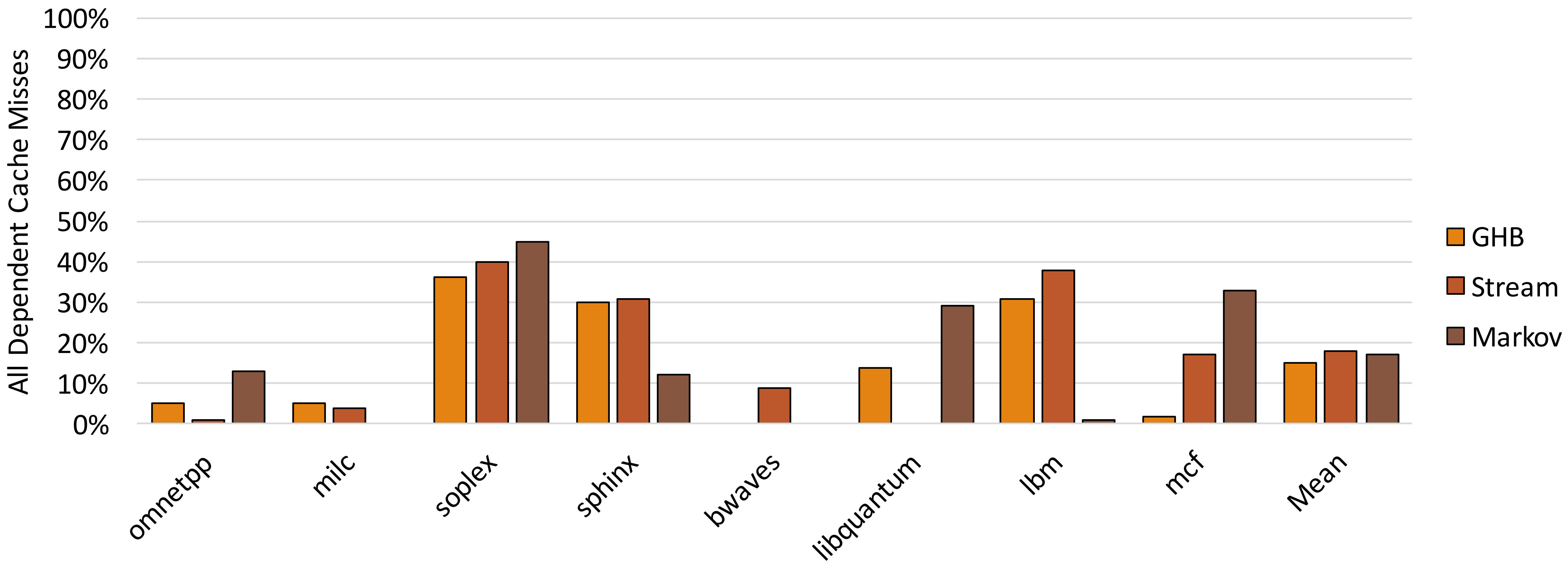
```
LD [P17] -> P10
MOV P10 -> P7
LD [P7] -> P13
ADD P3, P13 -> P20
LD [P20] -> P2
ADD P2, P3 -> P12
```

Dependent Cache Misses

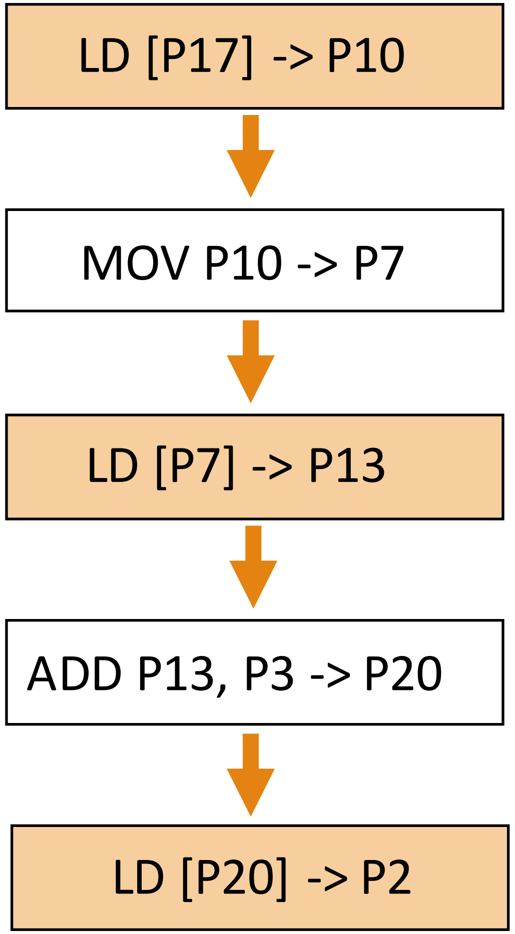
- The impact of effective memory latency on processor performance is magnified when a cache miss has dependent memory operations that will also result in a cache miss
  - Dependent cache misses form chains of long-latency operations that fill the reorder buffer and prevent the core from making forward progress
- Important in pointer-based workloads
- Dependent cache misses tend to have addresses that are difficult to predict with prefetching

The University of Texas at Austin
Electrical and Computer Engineering
Cockrell School of Engineering

Prefetching and Dependent Cache Misses



Dependence Chains



LD [P17] -> P10

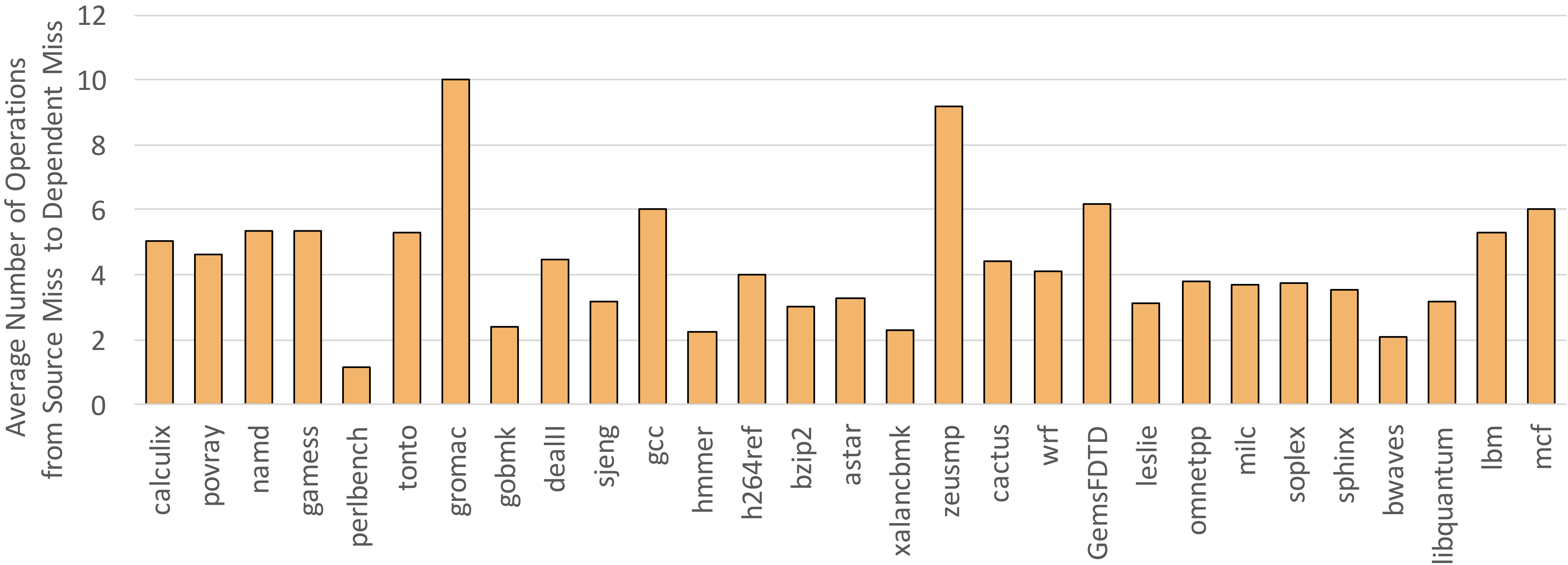
MOV P10 -> P7

LD [P7] -> P13

ADD P13, P3 -> P20

LD [P20] -> P2

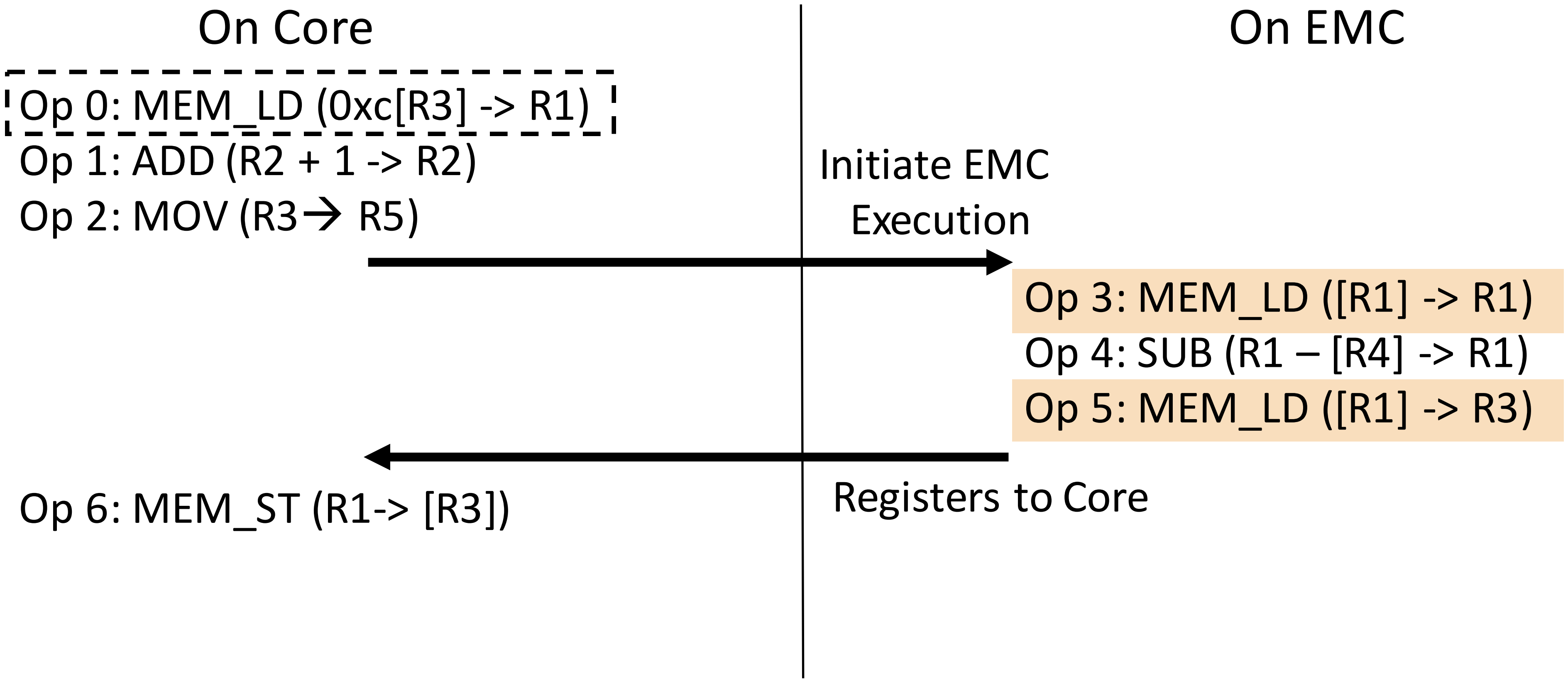
Dependence Chain Length



Reducing Effective Memory Access Latency

- Transparently reduce memory access latency for these dependent cache misses
    - Add compute capability to the memory controller
    - Modify the core to automatically identify the operations that are in the dependence chain of a cache miss
    - Migrate the dependence chain to the new enhanced memory controller (EMC) for execution when the source data arrives from main-memory
- Maintain traditional sequential execution model

Enhanced Memory Controller (EMC) Execution



On Core

Op 0: MEM_LD (0xc[R3] -> R1)

Op 1: ADD (R2 + 1 -> R2)

Op 2: MOV (R3→ R5)

Initiate EMC Execution

On EMC

Op 3: MEM_LD ([R1] -> R1)

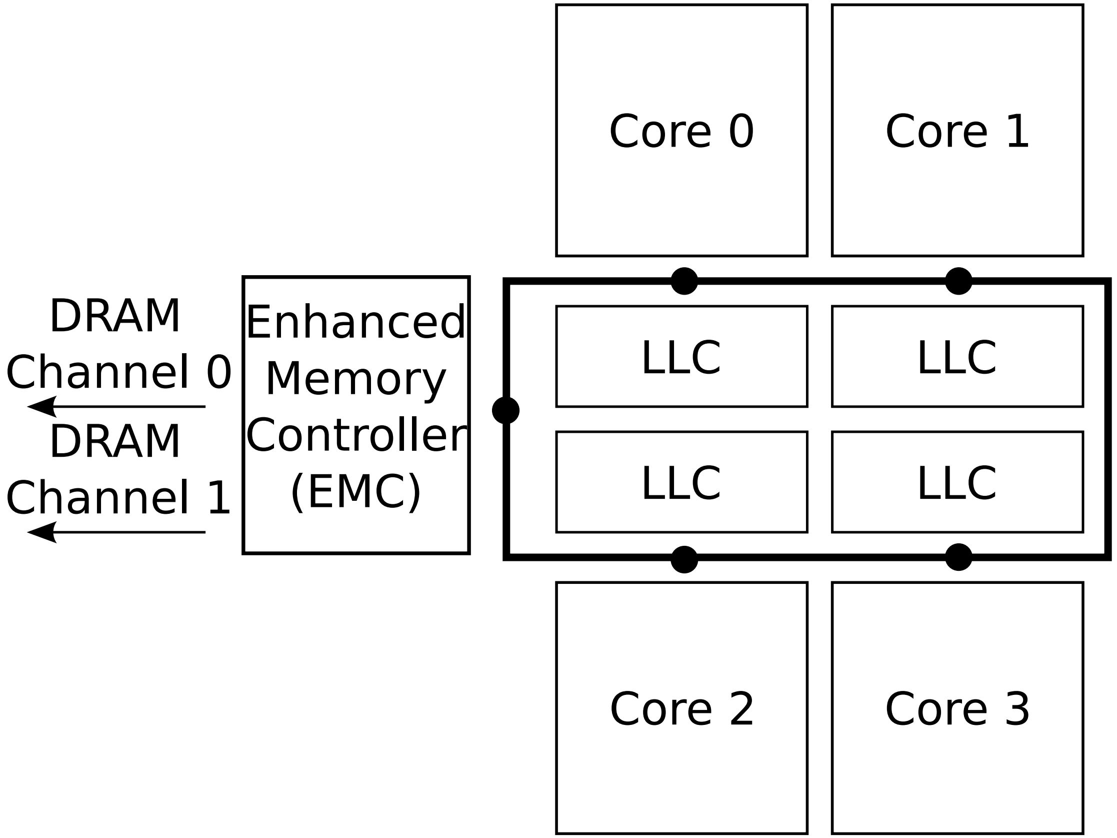
Op 4: SUB (R1 – [R4] -> R1)

Op 5: MEM_LD ([R1] -> R3)

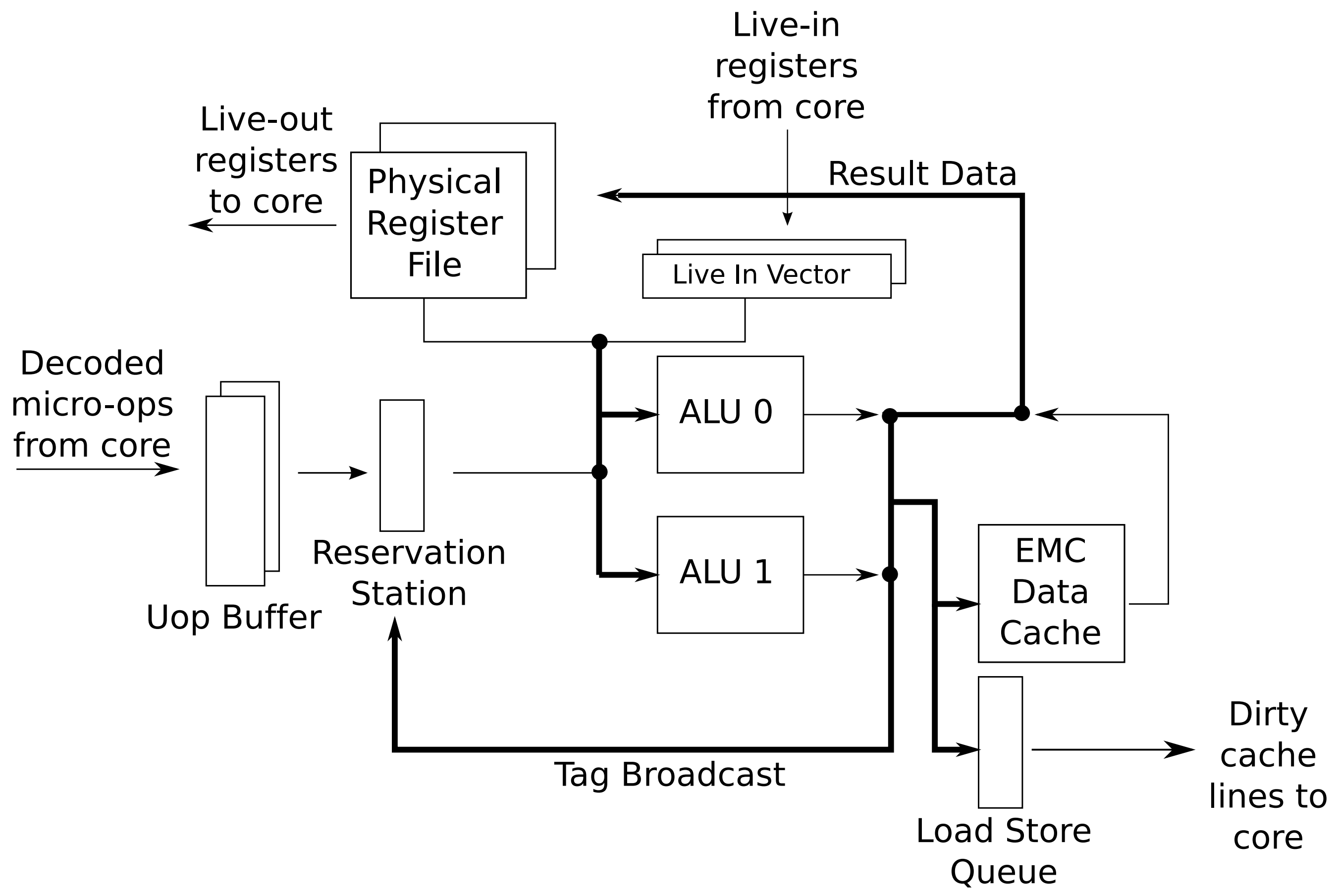
Registers to Core

Op 6: MEM_ST (R1-> [R3])

Quad-Core System



EMC Microarchitecture



- No Front-End
- 16 Physical Registers NOT 256
- No Register Renaming
- 2-Wide NOT 4-Wide
- No Floating Point or Vector Pipeline
- 4kB Streaming Data Cache

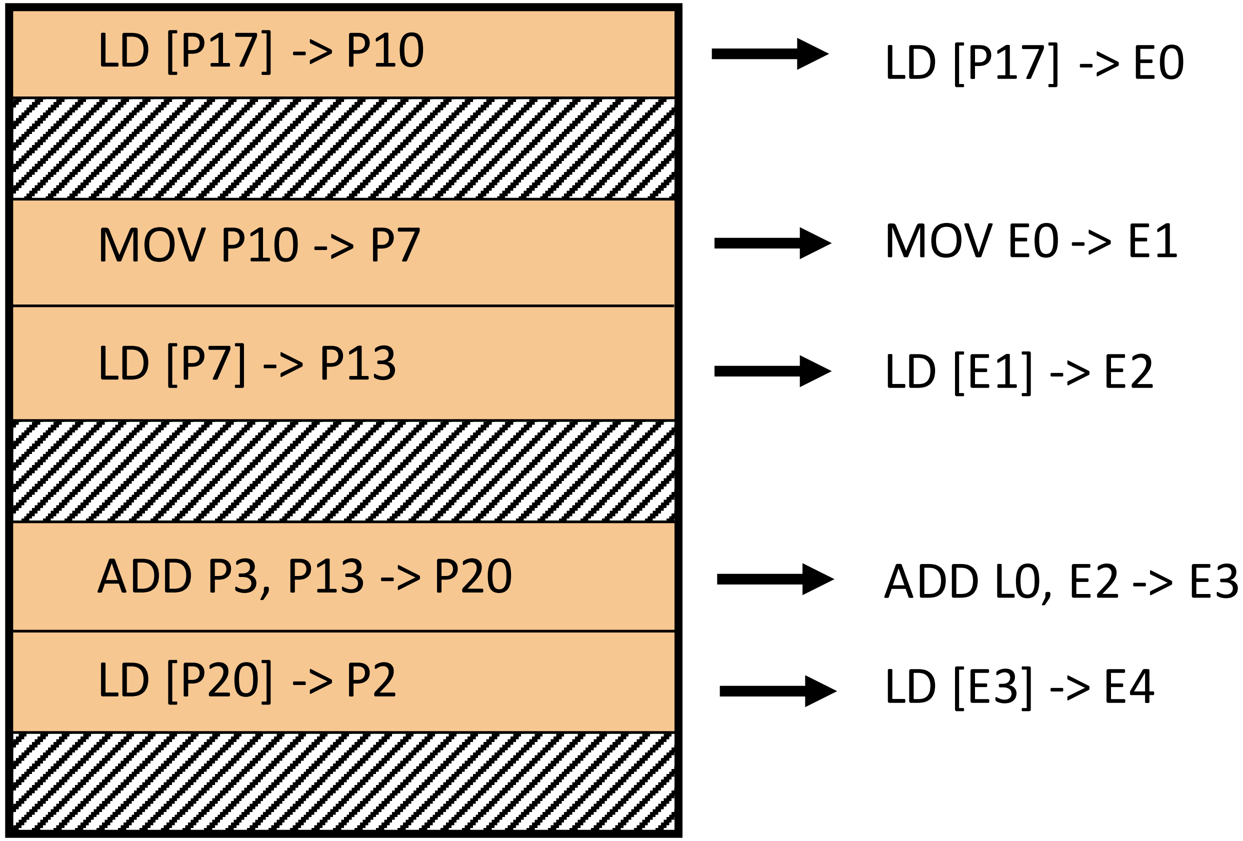
Dependence Chain Generation

Cycle: [illegible]

Live-In Vector: P3

Register Remapping Table:

Core Physical Register	EMC Physical Register



LD [P17] -> P10	LD [P17] -> E0
▨	
MOV P10 -> P7	MOV E0 -> E1
LD [P7] -> P13	LD [E1] -> E2
▨	
ADD P3, P13 -> P20	ADD L0, E2 -> E3
LD [P20] -> P2	LD [E3] -> E4
▨	

Memory Operations

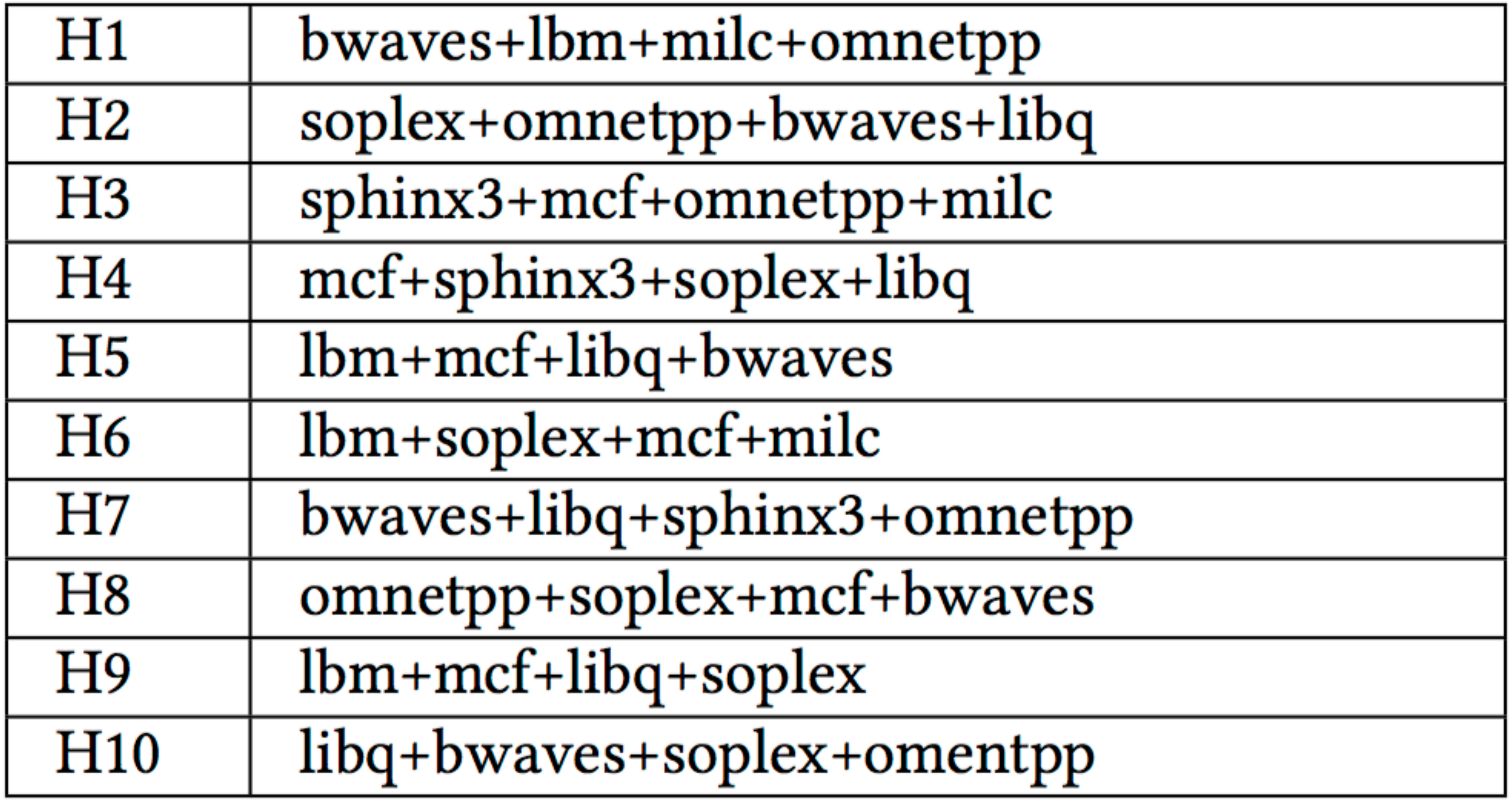
- Virtual Memory Support
  - Virtual address translation occurs through a 32-entry TLB per core
  - Execution at the EMC is cancelled at a page fault
- Loads first query the EMC data cache
  - Misses query the LLC in parallel with the memory access
  - A miss predictor [Qureshi and Loh: MICRO 2012] is maintained to reduce bandwidth cost
- Memory operations are retired in program order back at the core

System Configuration

- Quad-Core
  - 4-wide Issue
  - 256 Entry Reorder Buffer
  - 92 Entry Reservation Station
- Caches
  - 32 KB 8-Way Set Associative L1 I/D-Cache
  - 1MB 8-Way Set Associative Shared Last Level Cache per Core
- Non-Uniform Memory Access Latency DDR3 System
  - 128-Entry Memory Queue
  - Batch Scheduling

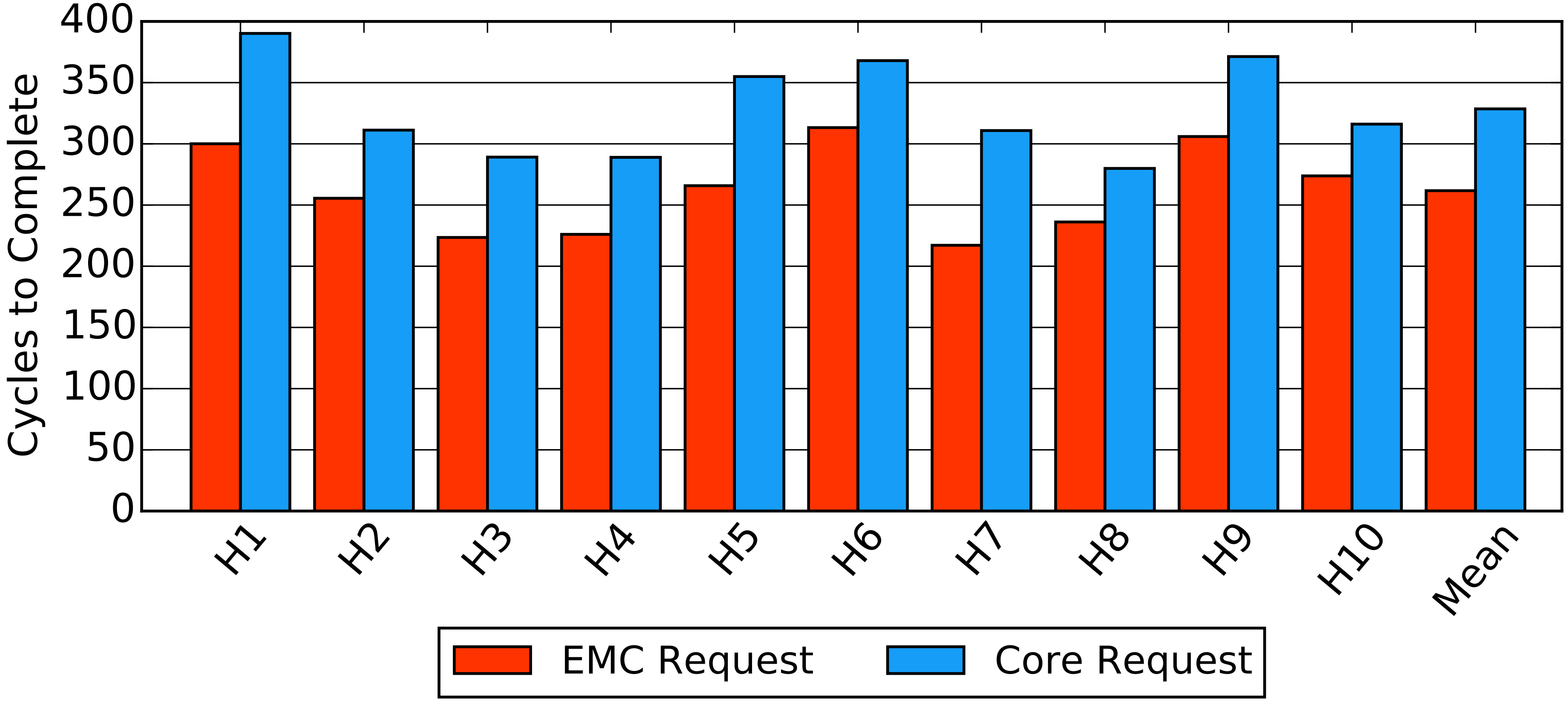
- Prefetchers
  - Stream, Markov, Global History Buffer
  - Feedback Directed Prefetching: Dynamic Degree 1-32
- EMC Compute
  - 2-wide issue
  - 2 issue contexts
  - Each Context Contains: 16 entry uop buffer, 16 entry physical register file
  - 4 kB Streaming Data Cache

Workloads

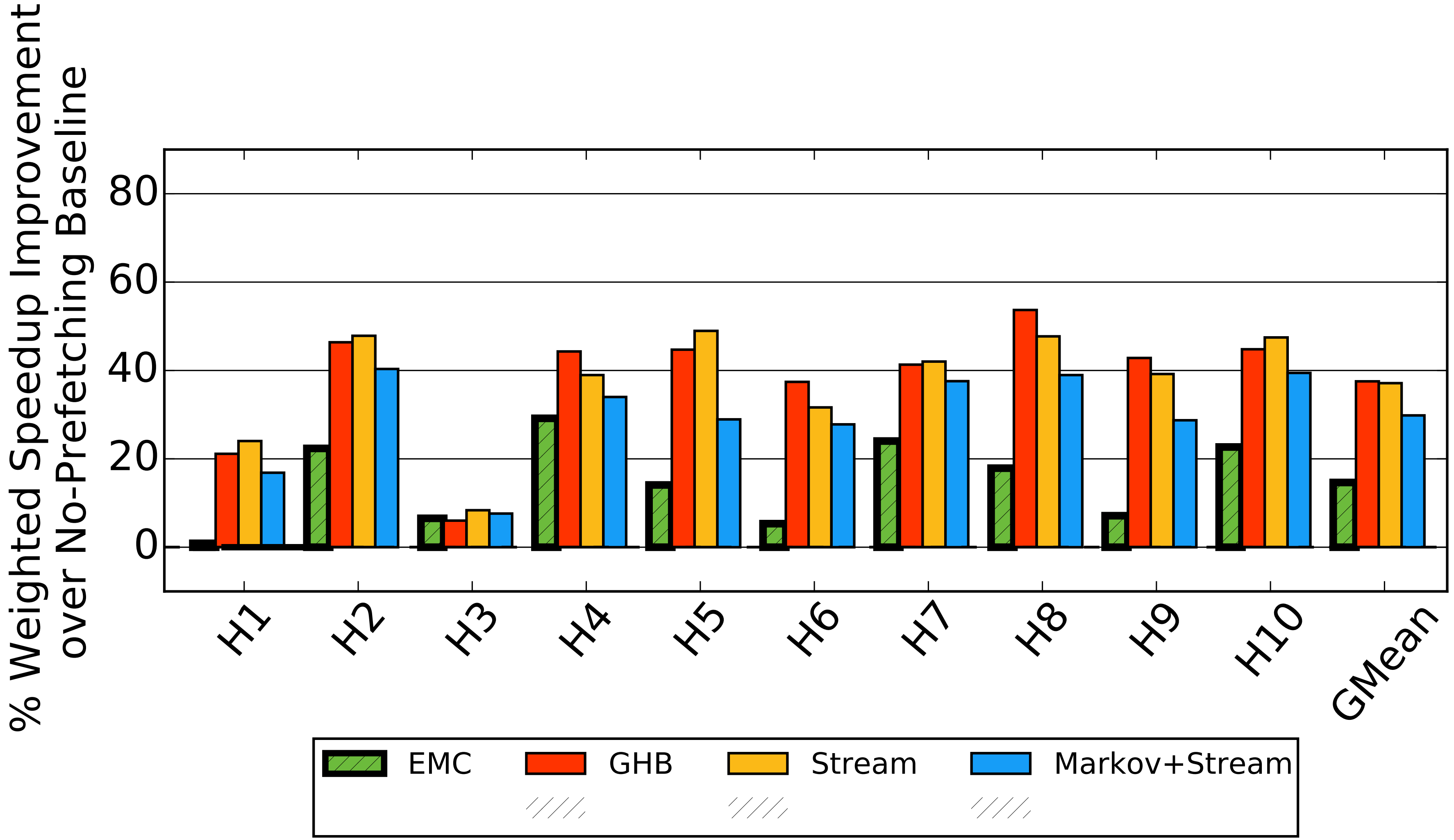


H1	bwaves+lbm+milc+omnetpp
H2	soplex+omnetpp+bwaves+libq
H3	sphinx3+mcf+omnetpp+milc
H4	mcf+sphinx3+soplex+libq
H5	lbm+mcf+libq+bwaves
H6	lbm+soplex+mcf+milc
H7	bwaves+libq+sphinx3+omnetpp
H8	omnetpp+soplex+mcf+bwaves
H9	lbm+mcf+libq+soplex
H10	libq+bwaves+soplex+omentpp

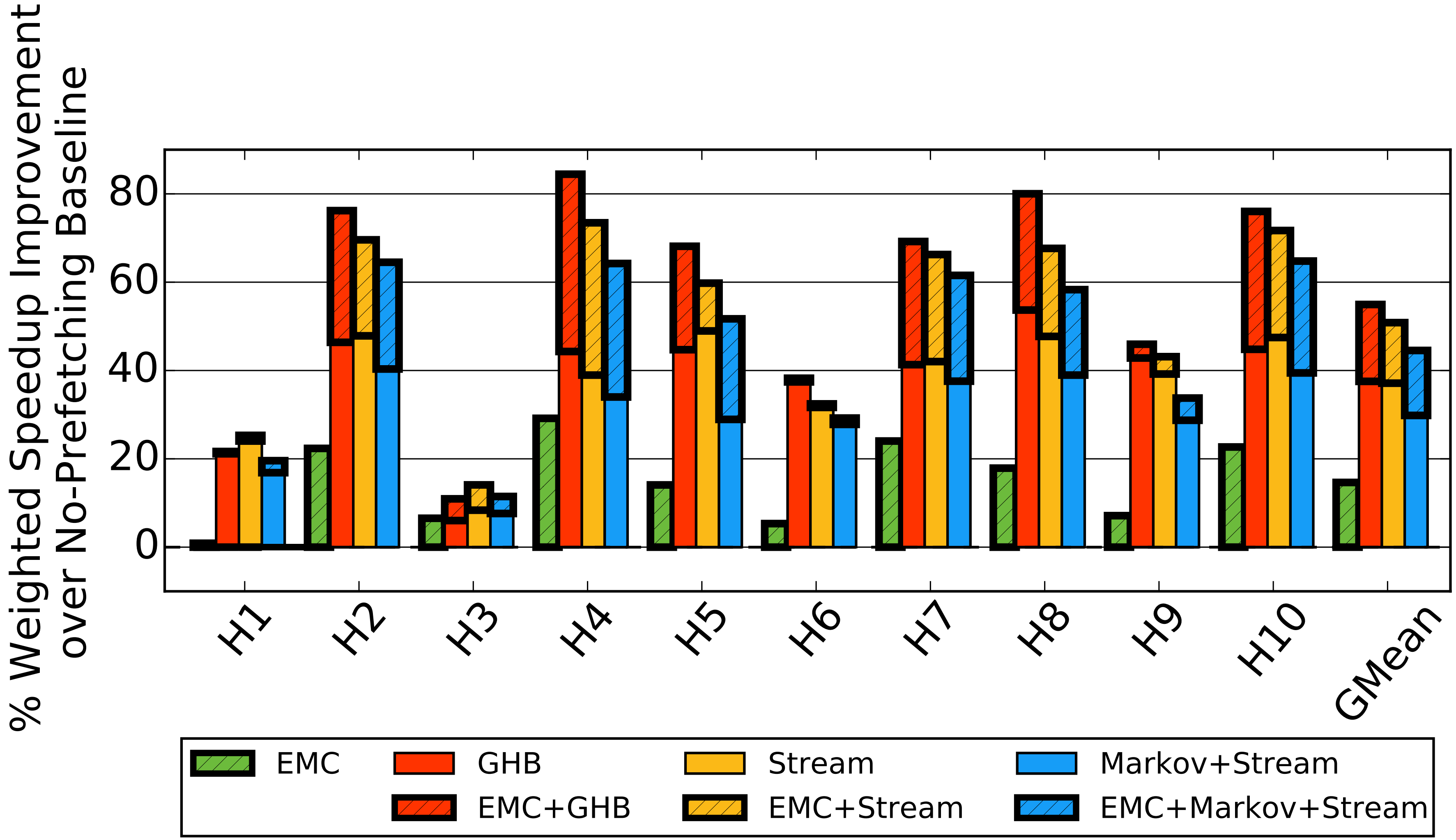
Effective Memory Access Latency Reduction



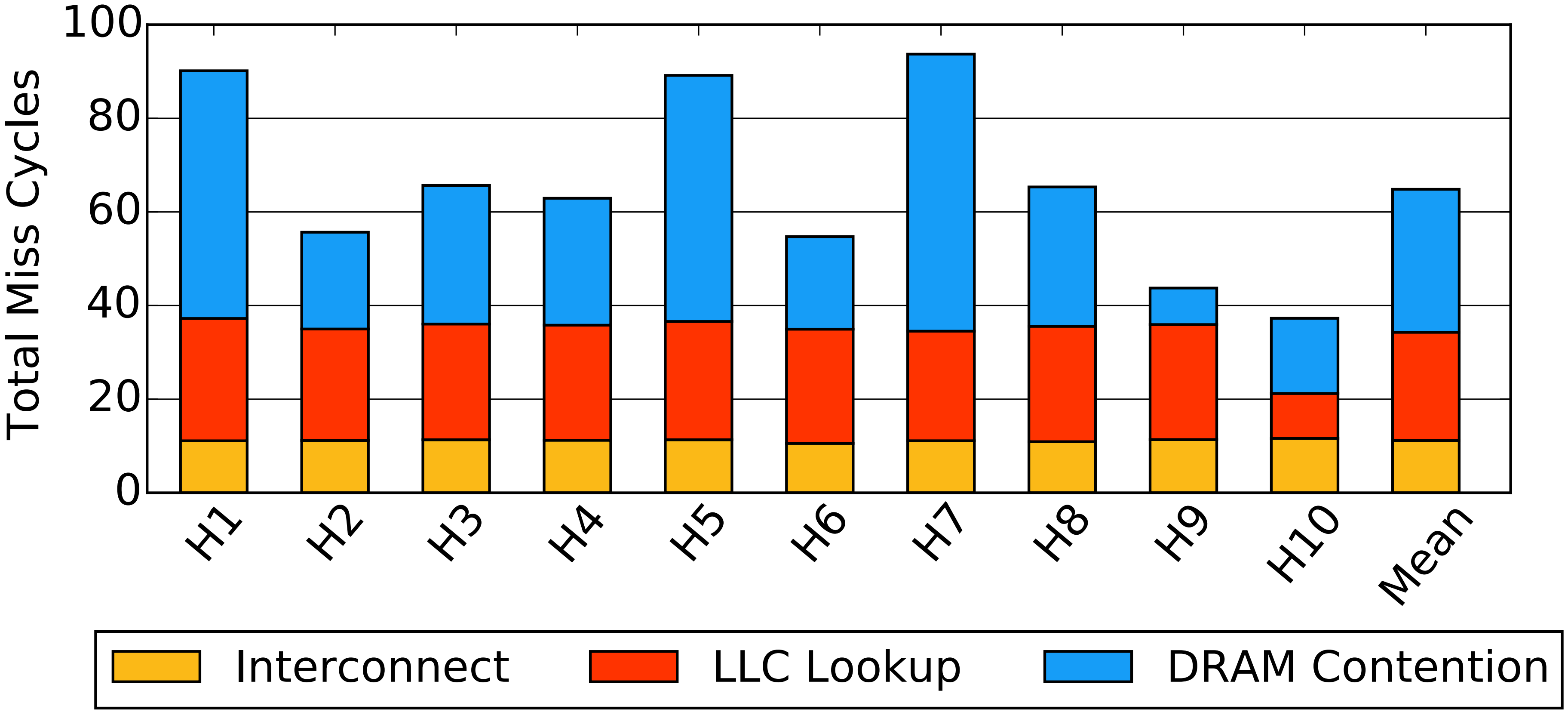
Quad-Core Performance



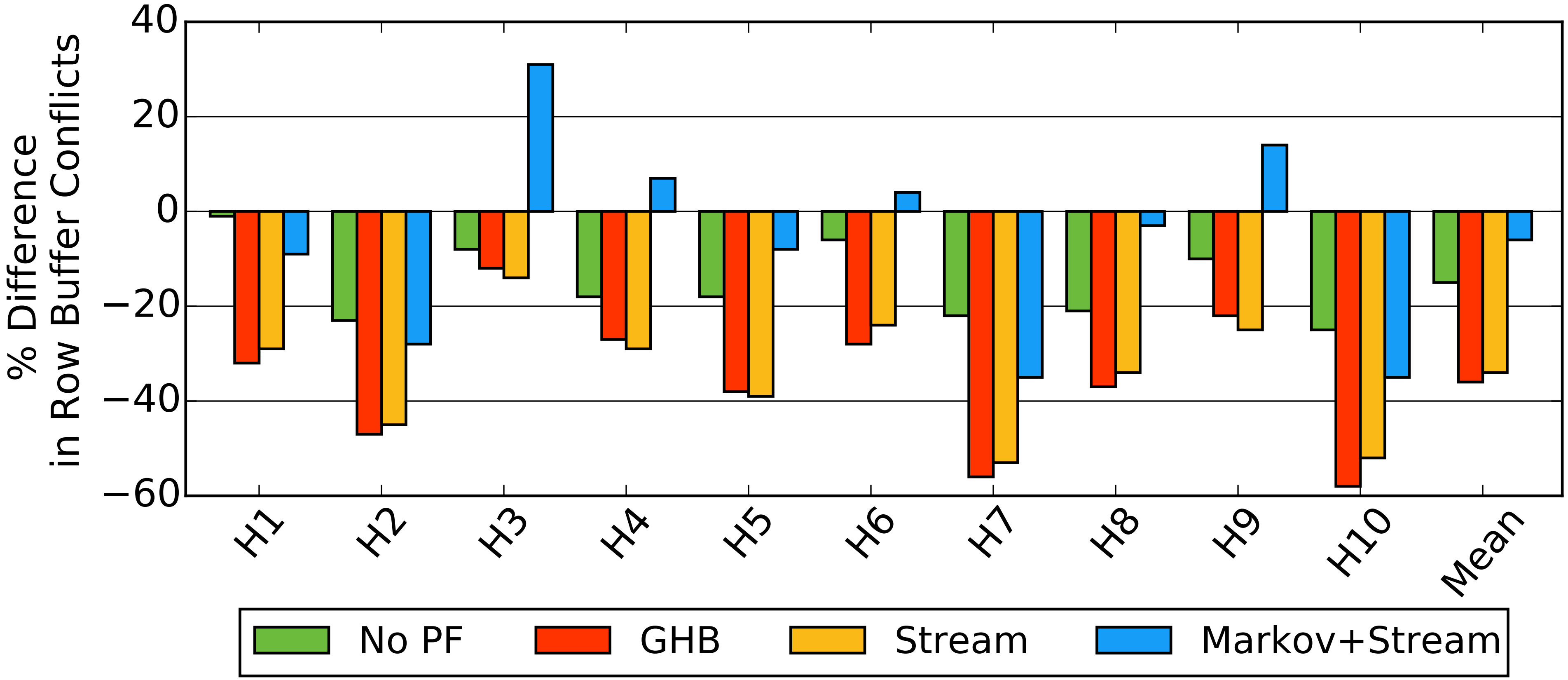
Quad-Core Performance with Prefetching



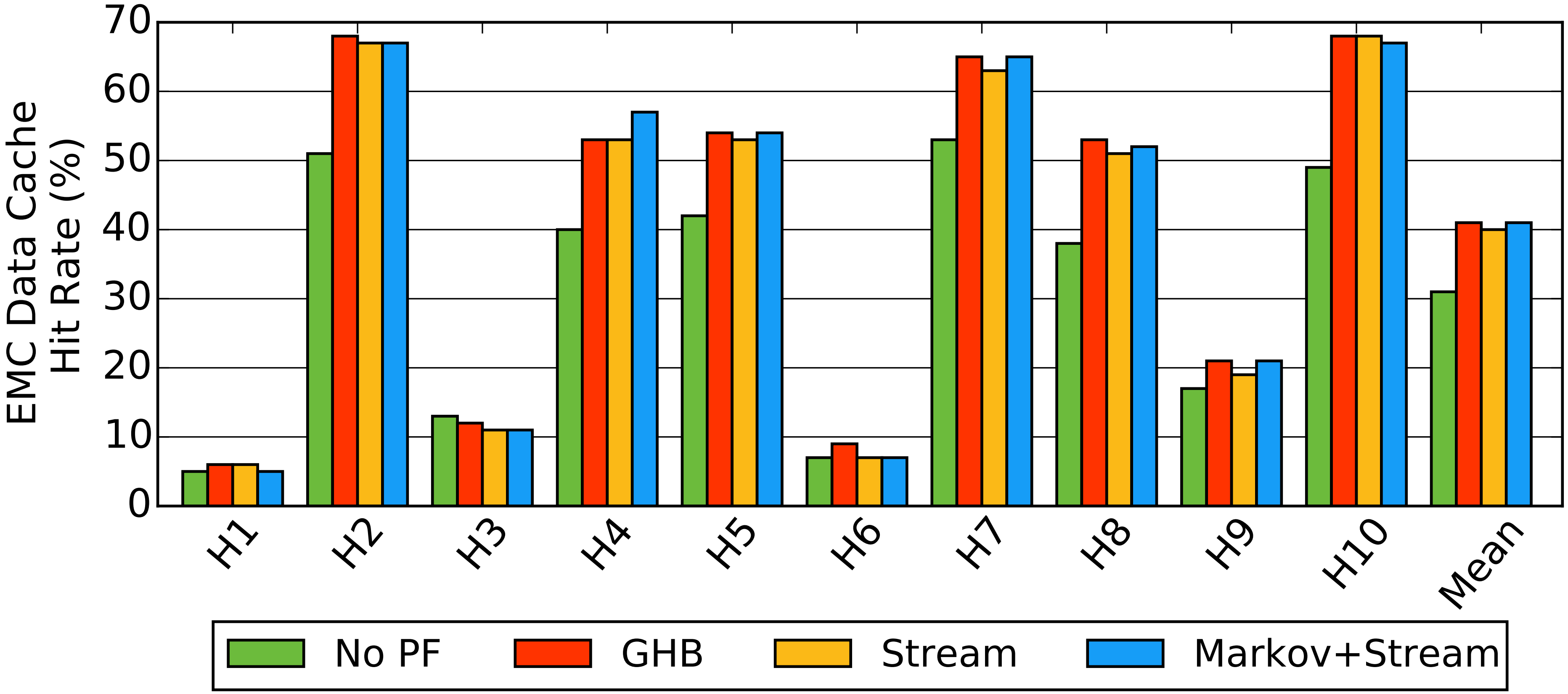
Latency Reduction Factors



Row-Buffer Conflict Reduction



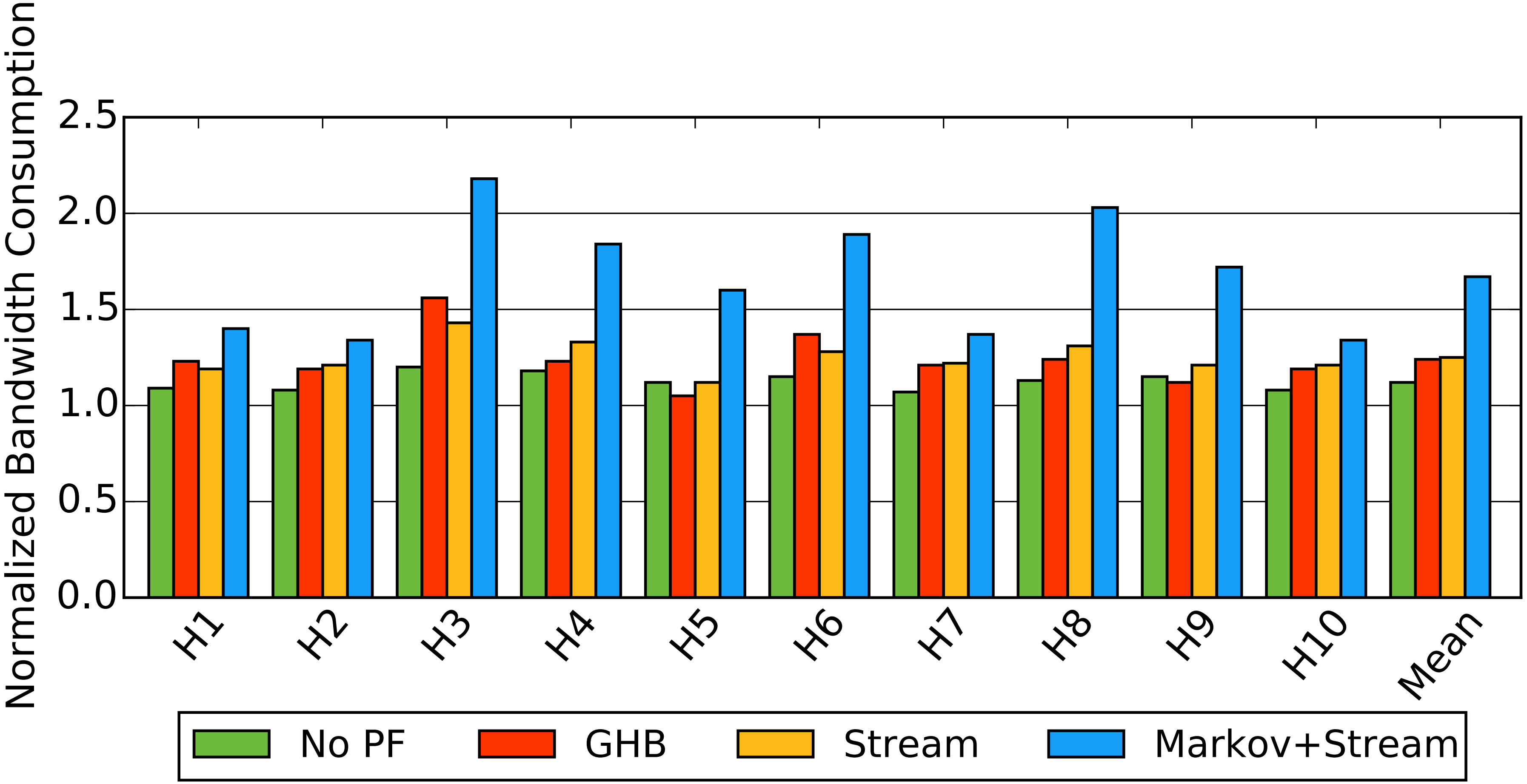
EMC Data Cache Hit Rate



Overhead: On-Chip Bandwidth

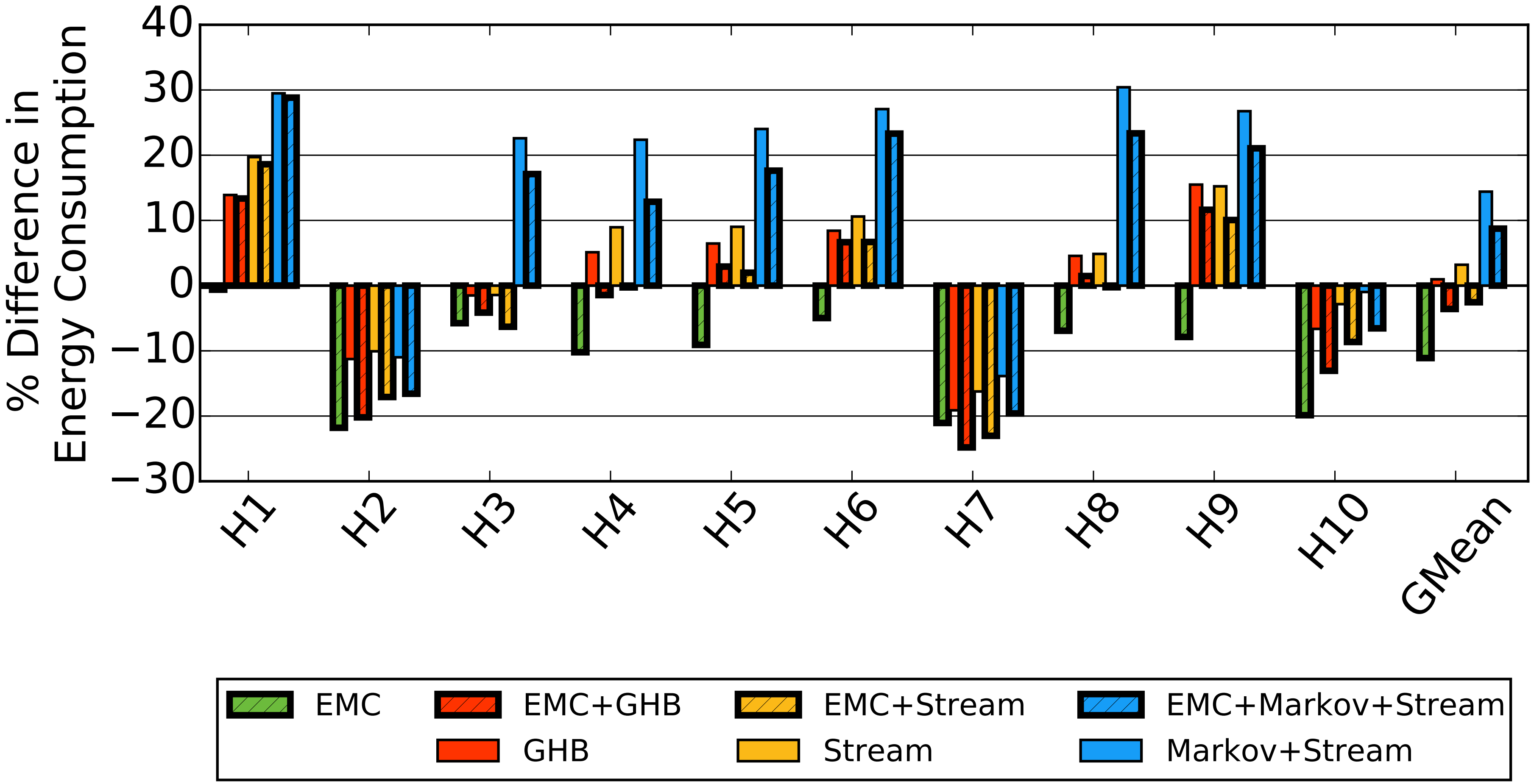
- H1-H10 observe a 33% average increase in data ring messages and a 7% increase in control ring messages
  - Sending dependence chains and live-ins to the EMC
  - Sending live-outs back to the core

Overhead: Memory Bandwidth



The University of Texas at Austin
Electrical and Computer Engineering
Cockrell School of Engineering

Energy Consumption



Other Information in the Paper

- 8-core evaluation with multiple distributed memory controllers
- Analysis of how the EMC and prefetching interact
- EMC sensitivity to increasing memory bandwidth
- More details of EMC execution: memory and control operations
- 5% EMC Area Overhead

Conclusions

- Adding an enhanced, compute capable, memory controller to the system results in two benefits:
  - EMC generates cache misses faster than the core by bypassing on-chip contention
  - EMC increases the likelihood of a memory access hitting an open row buffer
- 15% average gain in weighted speedup and a 11% reduction in energy consumption over a quad-core baseline with no prefetching
- Memory requests issued from the EMC observe a 20% lower latency on average than requests that are issued from the core

The University of Texas at Austin
Electrical and Computer Engineering
Cockrell School of Engineering

Accelerating Dependent Cache Misses with an Enhanced Memory Controller

Milad Hashemi, Khubaib, Eiman Ebrahimi, Onur Mutlu, Yale N. Patt

June 21, 2016