18-740: Computer Architecture
Recitation 5:
Main Memory Scaling Wrap-Up

Prof. Onur Mutlu
Carnegie Mellon University

Fall 2015

September 29, 2015

Review Assignments for Next Week

Required Reviews

- Due Tuesday Oct 6 @ 3pm
- Enter your reviews on the review website
- Please discuss ideas and thoughts on Piazza

Review Paper 1 (Required)

- Justin Meza, Qiang Wu, Sanjeev Kumar, and Onur Mutlu, **"A Large-Scale Study of Flash Memory Errors in the Field"** *Proceedings of the ACM International Conference on Measurement and Modeling of Computer Systems* (***SIGMETRICS***), Portland, OR, June 2015.

- Related paper
  - Justin Meza, Qiang Wu, Sanjeev Kumar, and Onur Mutlu, **"Revisiting Memory Errors in Large-Scale Production Data Centers: Analysis and Modeling of New Trends from the Field"** *Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks* (***DSN***), Rio de Janeiro, Brazil, June 2015.
    [Slides (pptx) (pdf)] [DRAM Error Model]

Review Paper 2 (Required)

- Yu Cai, Gulay Yalcin, Onur Mutlu, Erich F. Haratsch, Adrian Cristal, Osman Unsal, and Ken Mai, **"Error Analysis and Retention-Aware Error Management for NAND Flash Memory"** *Intel Technology Journal* (***ITJ***) *Special Issue on Memory Resiliency*, Vol. 17, No. 1, May 2013.

Review Paper 3 (Required)

- Edmund B. Nightingale, John R. Douceur, Vince Orgovan, "Cycles, cells and platters: an empirical analysisof hardware failures on a million consumer PCs," Eurosys 2011.
  - http://eurosys2011.cs.uni-salzburg.at/pdf/eurosys2011-nightingale.pdf

General Feedback on Project Proposals

Overall Feedback on Project Proposals

- Drive for becoming more concrete in the ideas
- And, be more ambitious
- And, be more thorough in related work (do not limit yourself only the papers we suggest)
- You will get more detailed, per-proposal feedback from us

Next Steps in Projects

- Next steps are:

  1. Do a thorough literature search in your area

  Go deeper. Find and cite the seminal works as well as recent works. Increase the scholarship.

  2. Make the ideas and mechanisms more concrete

  3. Develop and understand the means for testing your ideas

Next Week Recitation Plan

- Present your revised proposal to get more feedback
- 10 minutes per group max

General Feedback on Reviews

Feedback on Reviews

- Many reviews are very good quality

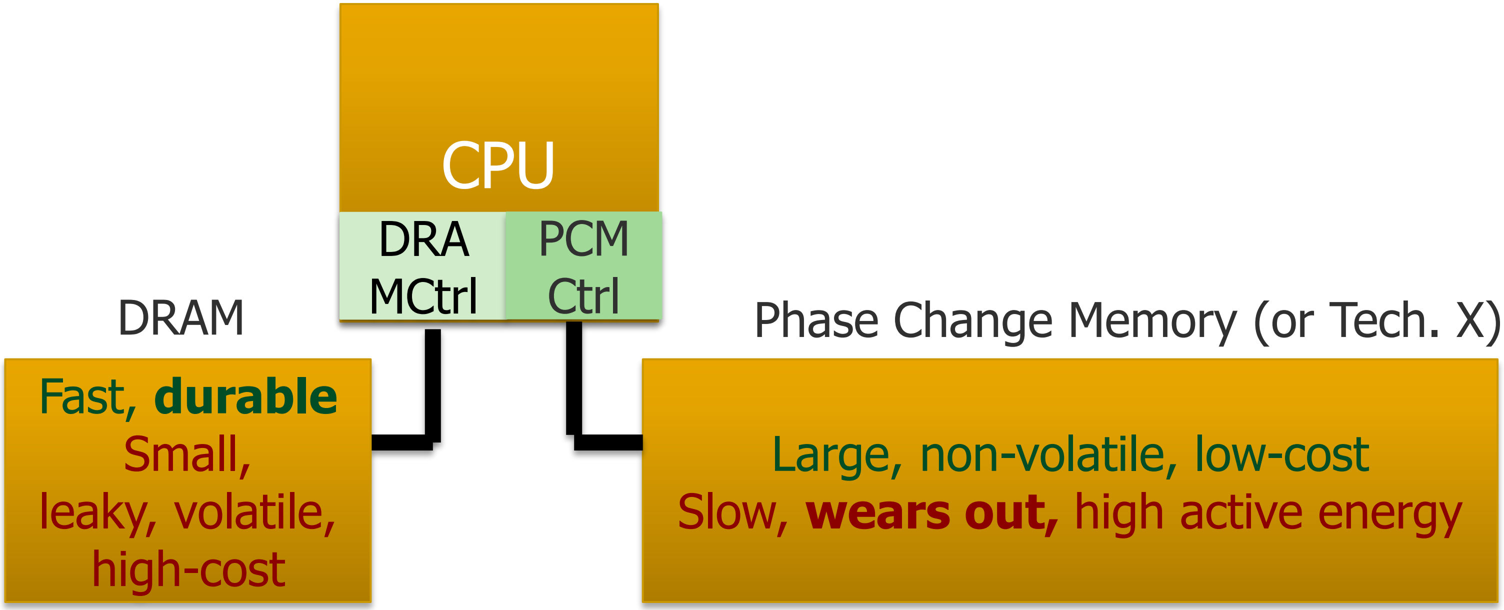
- Avoid being
  - Too terse: show the depth of your thinking
  - Too critical: evaluate the positive contribution of each paper

- Evaluate each paper considering when it was written
  - Example: The first cache paper did not consider alternative replacement policies
    - A paper cannot do everything: there is no perfect paper

- Develop more ideas after reading each paper

Rethinking Memory System Design (Continued)

Solution 3: Hybrid Memory Systems


CPU
DRA MCtrl
PCM Ctrl
DRAM
Fast, **durable**
Small,
leaky, volatile,
high-cost
Phase Change Memory (or Tech. X)
Large, non-volatile, low-cost
Slow, **wears out,** high active energy

Hardware/software manage data allocation and movement
to achieve the best of multiple technologies

Meza+, "Enabling Efficient and Scalable Hybrid Memories," IEEE Comp. Arch. Letters, 2012.

Yoon+, "Row Buffer Locality Aware Caching Policies for Hybrid Memories," ICCD 2012 Best Paper Award.

One Option: DRAM as a Cache for PCM

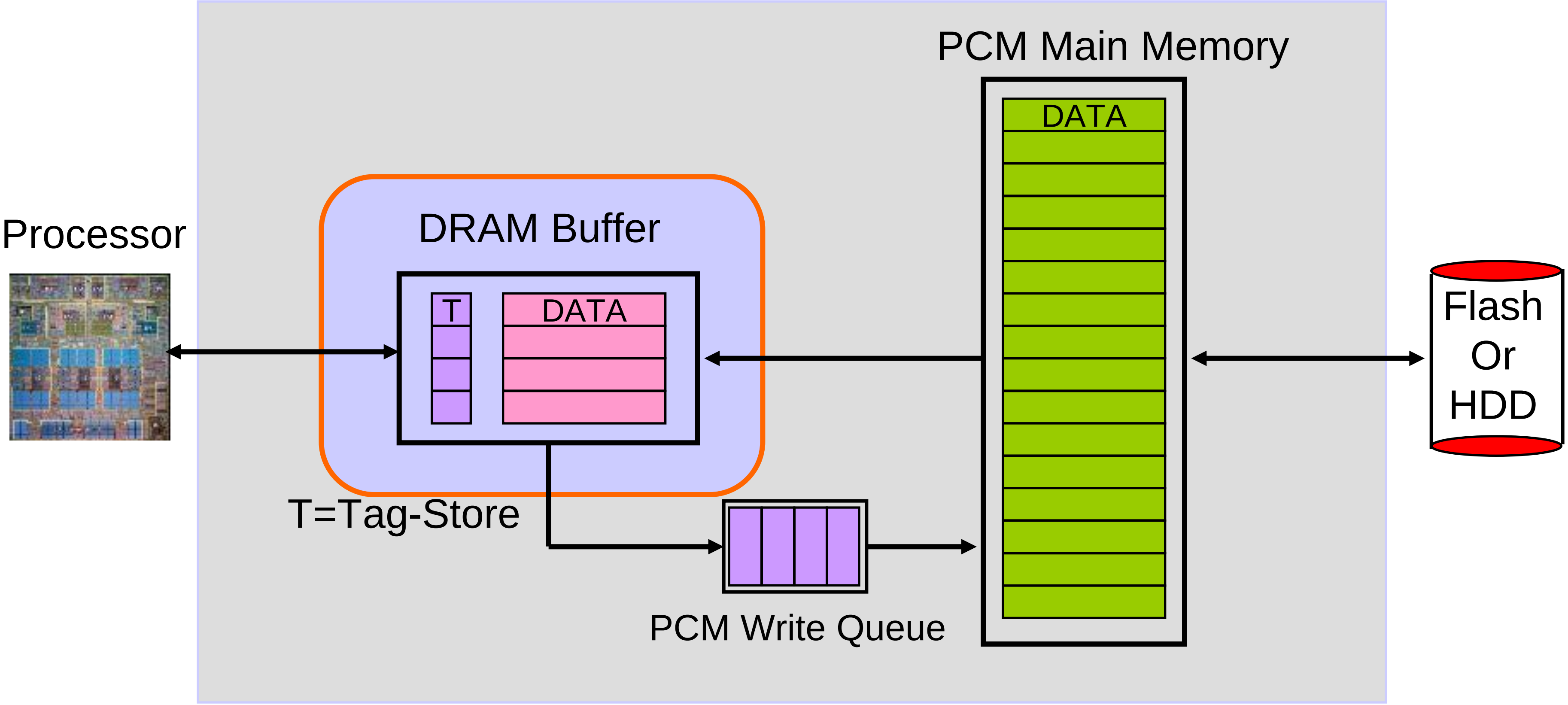
- PCM is main memory; DRAM caches memory rows/blocks
  - Benefits: Reduced latency on DRAM cache hit; write filtering
- Memory controller hardware manages the DRAM cache
  - Benefit: Eliminates system software overhead

- Three issues:
  - What data should be placed in DRAM versus kept in PCM?
  - What is the granularity of data movement?
  - How to design a low-cost hardware-managed DRAM cache?

- Two idea directions:
  - Locality-aware data placement **[Yoon+ , ICCD 2012]**
  - Cheap tag stores and dynamic granularity **[Meza+, IEEE CAL 2012]**

DRAM as a Cache for PCM

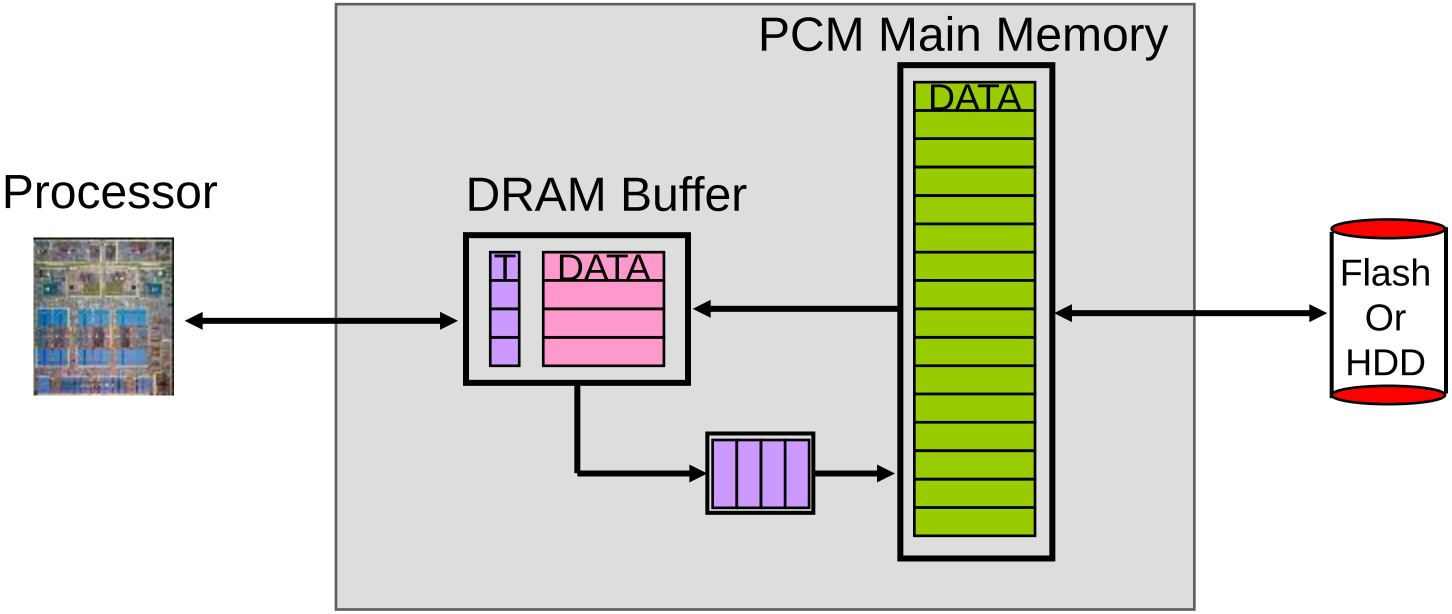
- Goal: Achieve the best of both DRAM and PCM/NVM
  - Minimize amount of DRAM w/o sacrificing performance, endurance
  - DRAM as cache to tolerate PCM latency and write bandwidth
  - PCM as main memory to provide large capacity at good cost and power



Qureshi+, "Scalable high performance main memory system using phase-change memory technology," ISCA 2009.

Write Filtering Techniques

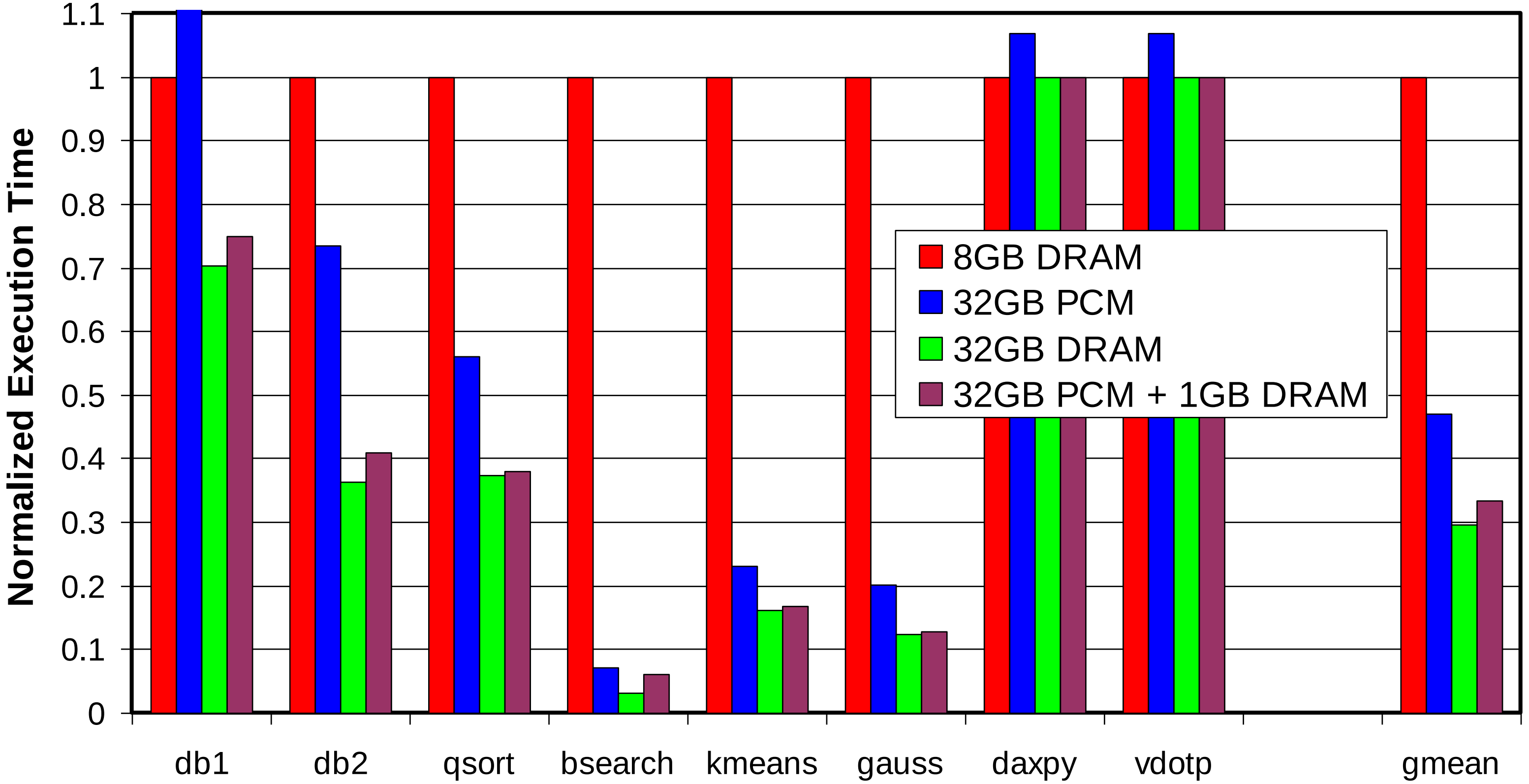
- Lazy Write: Pages from disk installed only in DRAM, not PCM
- Partial Writes: Only dirty lines from DRAM page written back
- Page Bypass: Discard pages with poor reuse on DRAM eviction



- Qureshi et al., "Scalable high performance main memory system using phase-change memory technology," ISCA 2009.

Results: DRAM as PCM Cache (I)

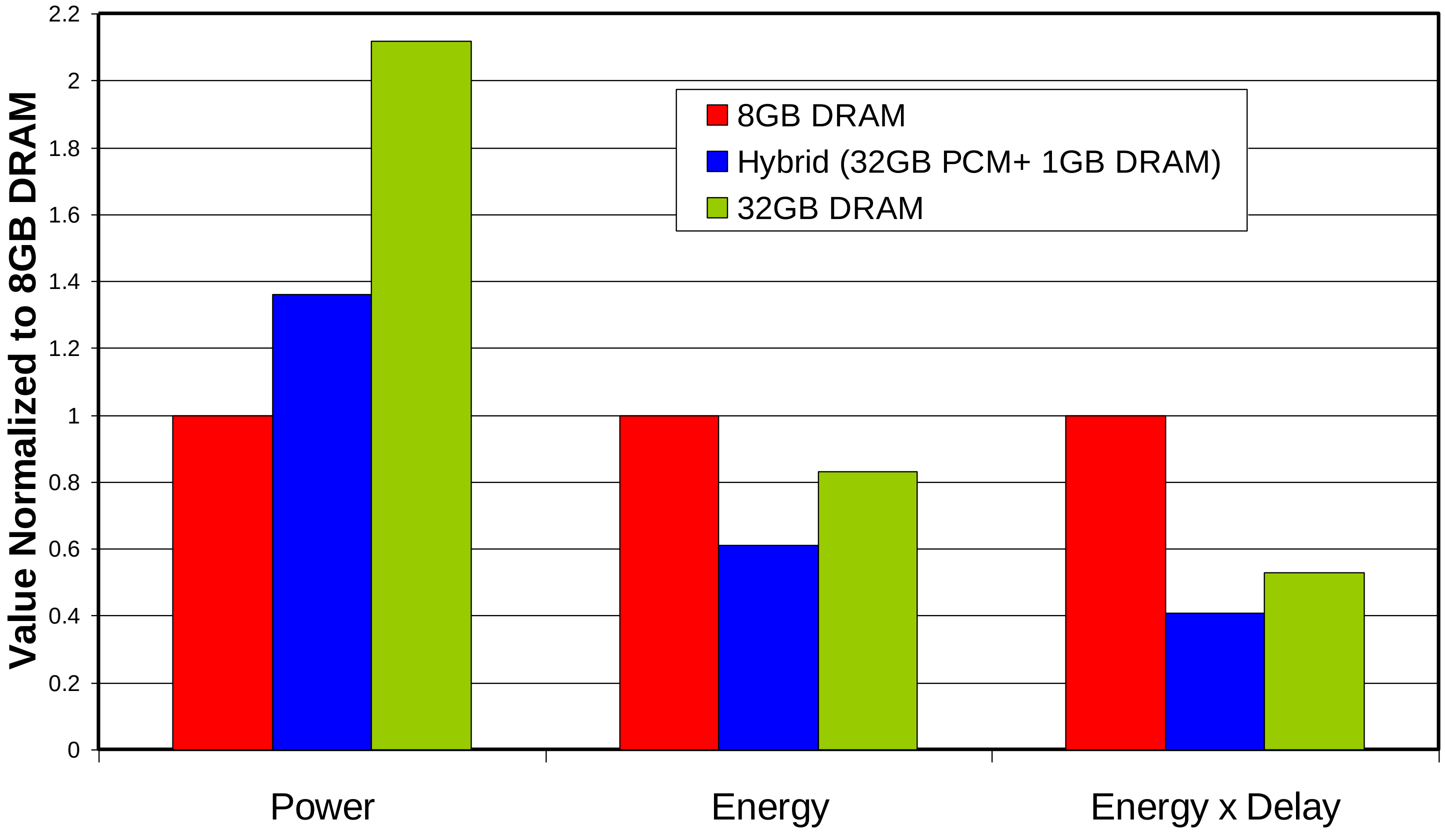
- Simulation of 16-core system, 8GB DRAM main-memory at 320 cycles, HDD (2 ms) with Flash (32 us) with Flash hit-rate of 99%
- Assumption: PCM 4x denser, 4x slower than DRAM
- DRAM block size = PCM page size (4kB)



Qureshi+, "Scalable high performance main memory system using phase-change memory technology," ISCA 2009.

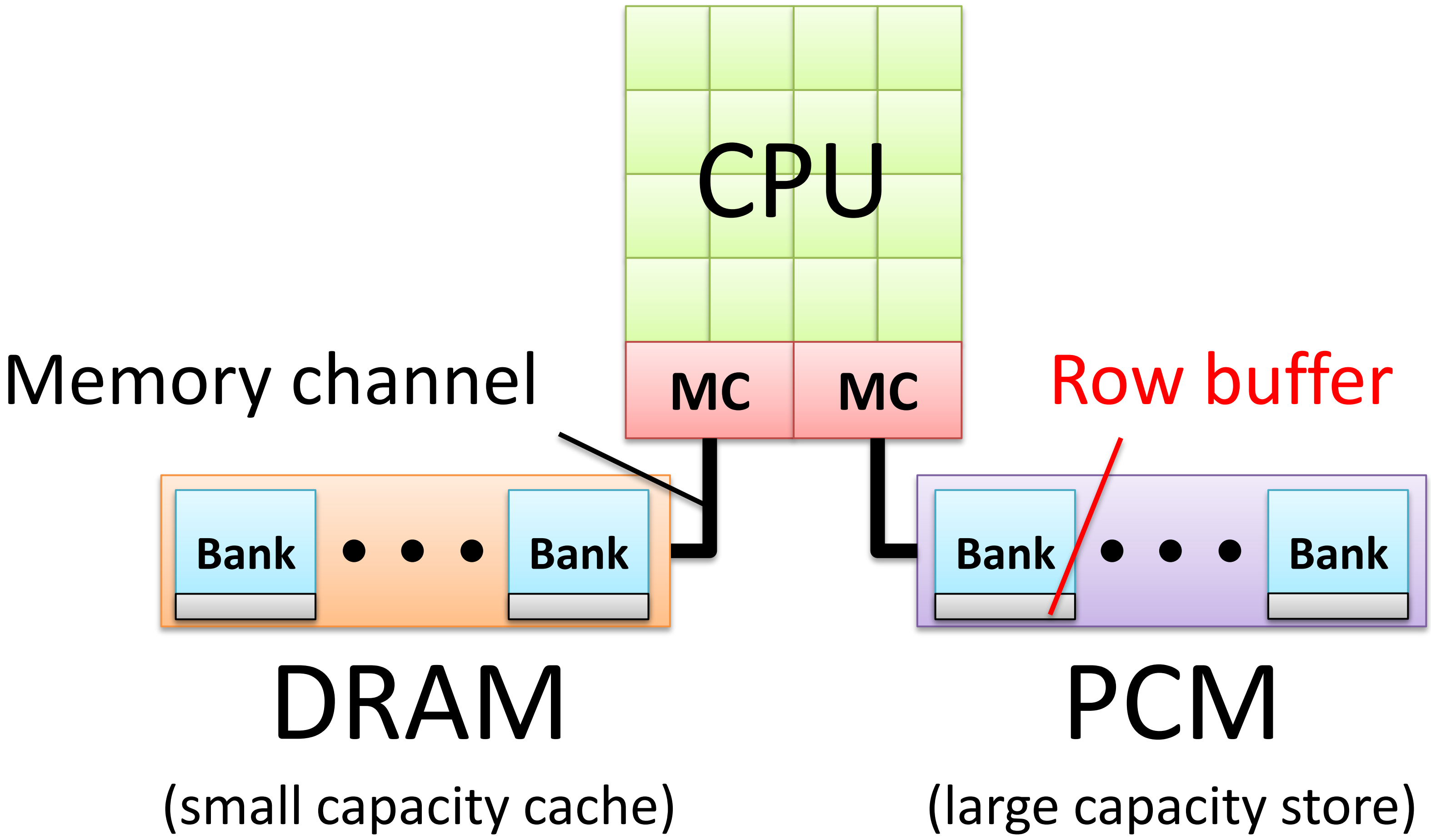
Results: DRAM as PCM Cache (II)

- PCM-DRAM Hybrid performs similarly to similar-size DRAM
- Significant power and energy savings with PCM-DRAM Hybrid
- Average lifetime: 9.7 years (no guarantees)



Qureshi+, "Scalable high performance main memory system using phase-change memory technology," ISCA 2009.

Hybrid Memory: A Closer Look



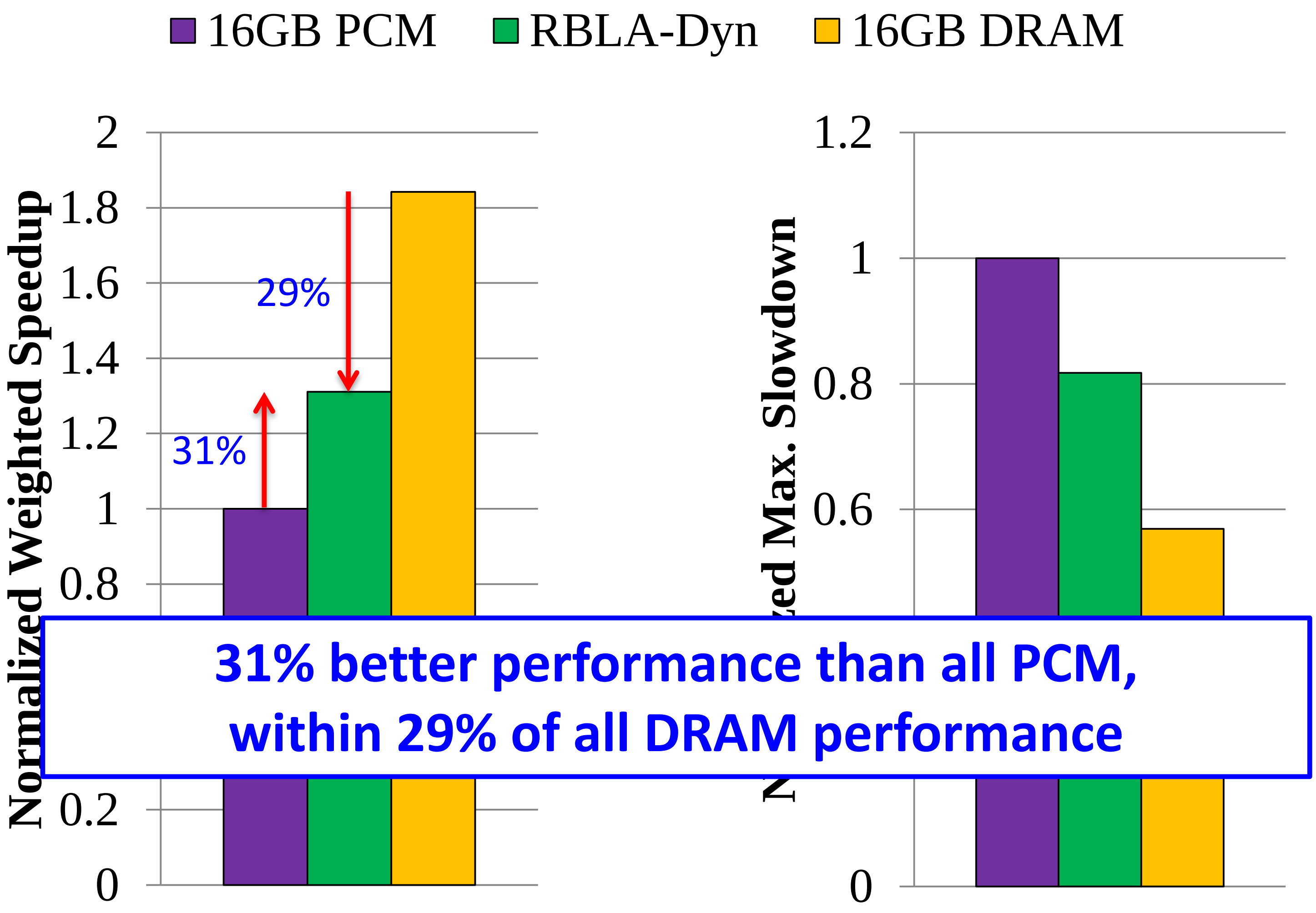
Key Observation

- Row buffers exist in both DRAM and PCM
  - Row hit latency **similar** in DRAM & PCM [Lee+ ISCA'09]
  - Row miss latency **small** in DRAM, **large** in PCM

- Place data in DRAM which
  - is likely to miss in the row buffer (low row buffer locality)→ miss penalty is smaller in DRAM

    AND
  - is reused many times → cache only the data worth the movement cost and DRAM space

Hybrid vs. All-PCM/DRAM [ICCD'12]

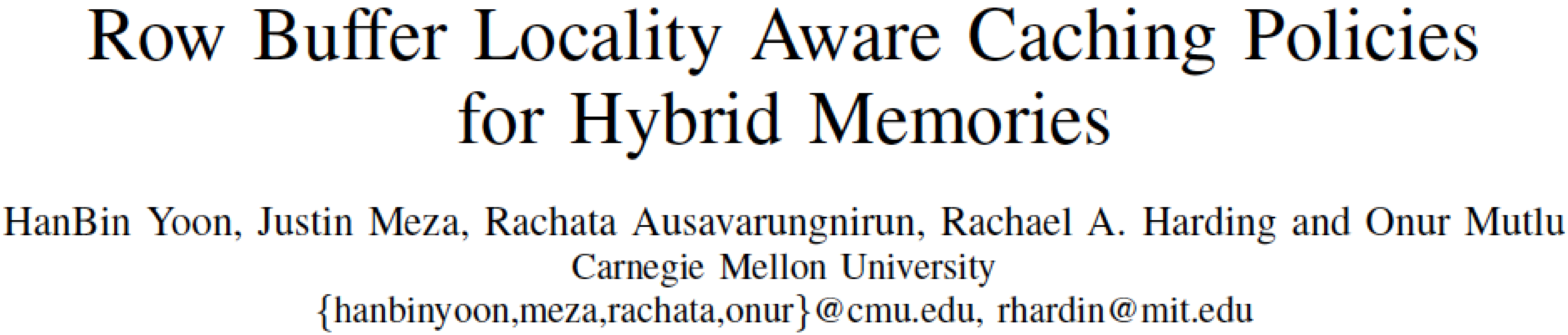


31% better performance than all PCM, within 29% of all DRAM performance

Yoon+, "Row Buffer Locality-Aware Data Placement in Hybrid Memories," ICCD 2012 Best Paper Award.

For More on Hybrid Memory Data Placement

- HanBin Yoon, Justin Meza, Rachata Ausavarungnirun, Rachael Harding, and Onur Mutlu, **"Row Buffer Locality Aware Caching Policies for Hybrid Memories"** *Proceedings of the 30th IEEE International Conference on Computer Design* (***ICCD***), Montreal, Quebec, Canada, September 2012. Slides (pptx) (pdf)

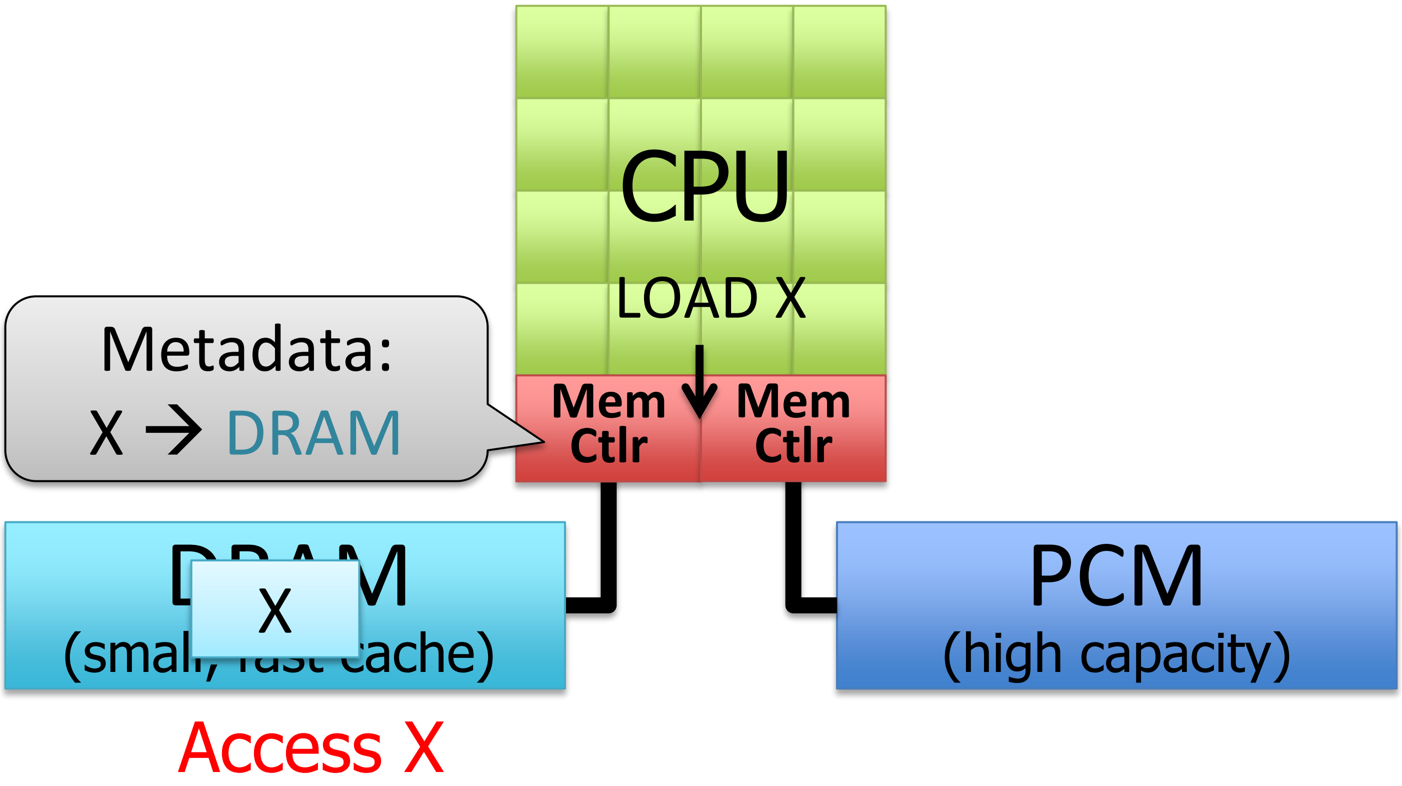


Row Buffer Locality Aware Caching Policies for Hybrid Memories

HanBin Yoon, Justin Meza, Rachata Ausavarungnirun, Rachael A. Harding and Onur Mutlu
Carnegie Mellon University
{hanbinyoon,meza,rachata,onur}@cmu.edu, rhardin@mit.edu

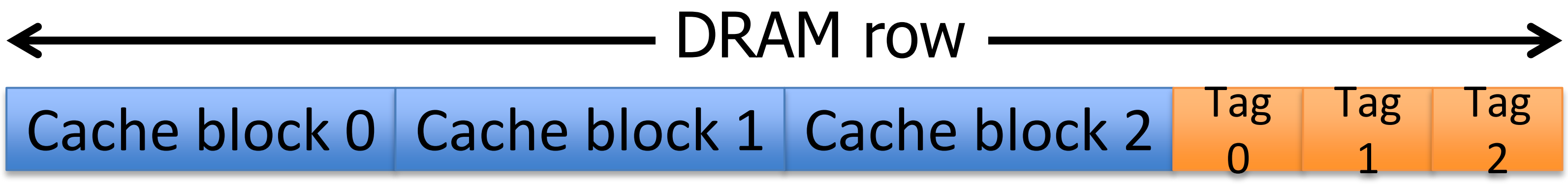
The Problem with Large DRAM Caches

- A large DRAM cache requires a large metadata (tag + block-based information) store
- How do we design an efficient DRAM cache?



Idea 1: Tags in Memory

- Store tags in the same row as data in DRAM
  - Store metadata in same row as their data
  - Data and metadata can be accessed together



- Benefit: No on-chip tag storage overhead
- Downsides:
  - Cache hit determined only after a DRAM access
  - Cache hit requires two DRAM accesses

Idea 2: Cache Tags in SRAM

- Recall Idea 1: Store all metadata in DRAM
  - To reduce metadata storage overhead

- Idea 2: Cache in on-chip SRAM frequently-accessed metadata
  - Cache only a small amount to keep SRAM size small

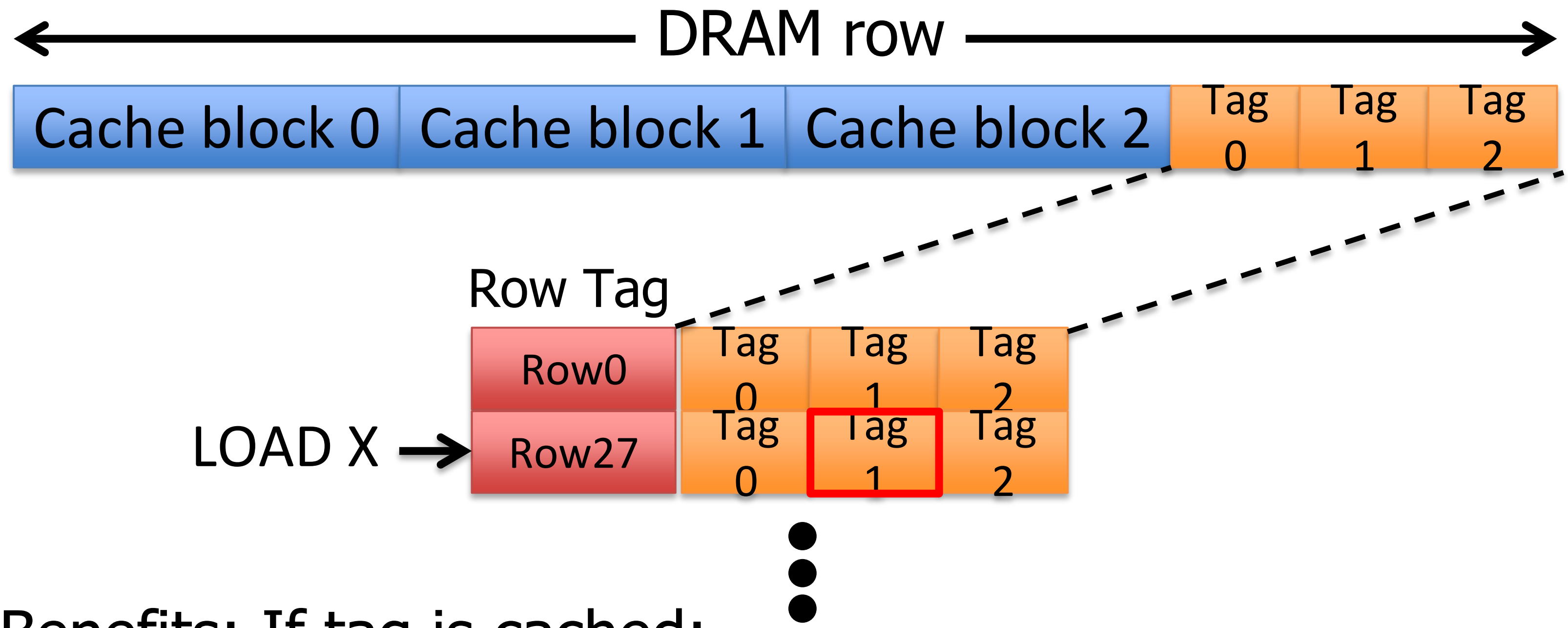
Idea 3: Dynamic Data Transfer Granularity

- Some applications benefit from caching more data
  - They have good spatial locality
- Others do not
  - Large granularity wastes bandwidth and reduces cache utilization

- Idea 3: Simple dynamic caching granularity policy
  - Cost-benefit analysis to determine best DRAM cache block size
  - Group main memory into sets of rows
  - Some row sets follow a fixed caching granularity
  - The rest of main memory follows the best granularity
    - Cost–benefit analysis: access latency versus number of cachings
    - Performed every quantum

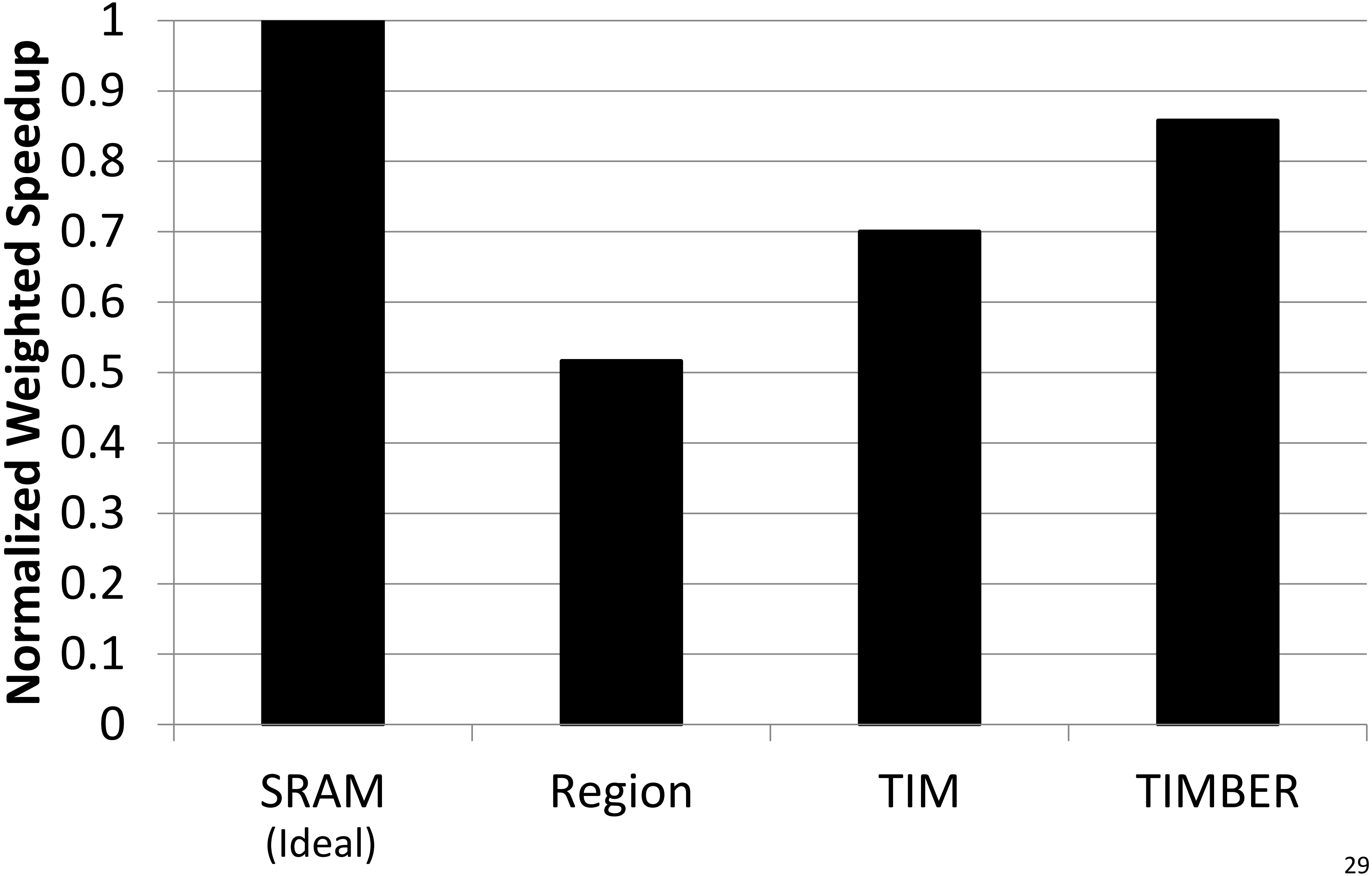
TIMBER Tag Management

- A Tag-In-Memory BuffER (TIMBER)
  - Stores recently-used tags in a small amount of SRAM

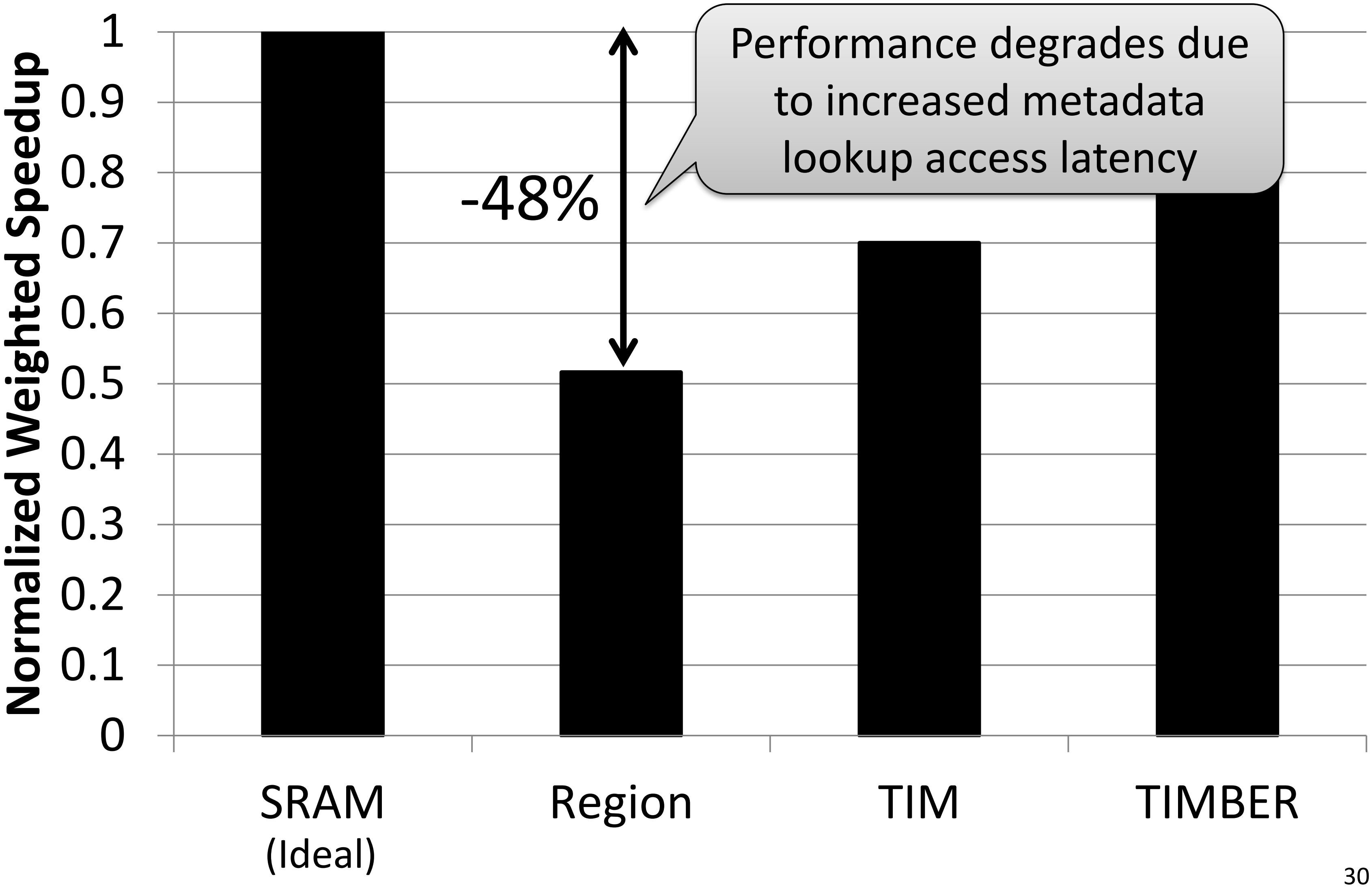


- Benefits: If tag is cached:
  - no need to access DRAM twice
  - cache hit determined quickly

Metadata Storage Performance

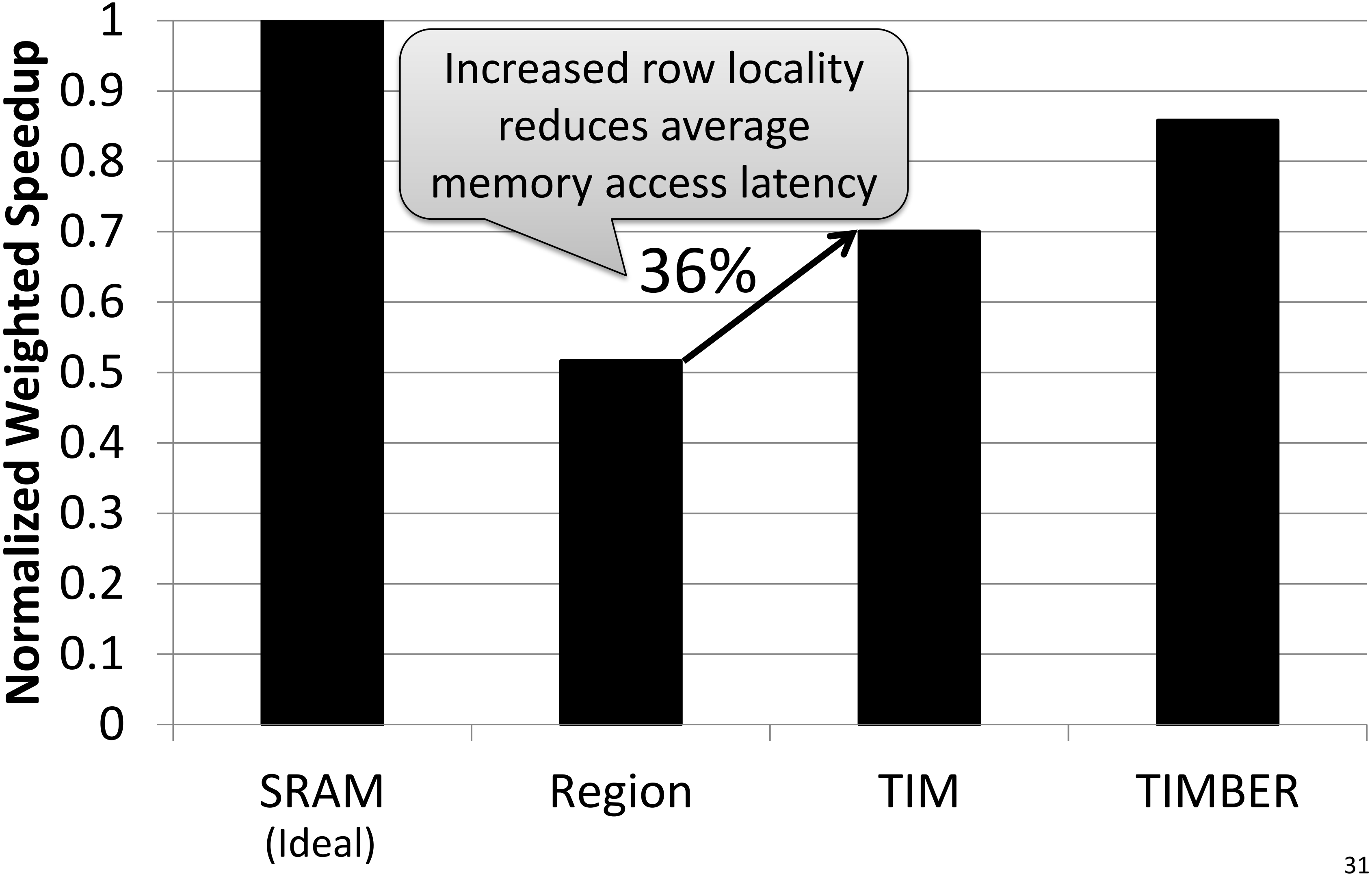


Metadata Storage Performance



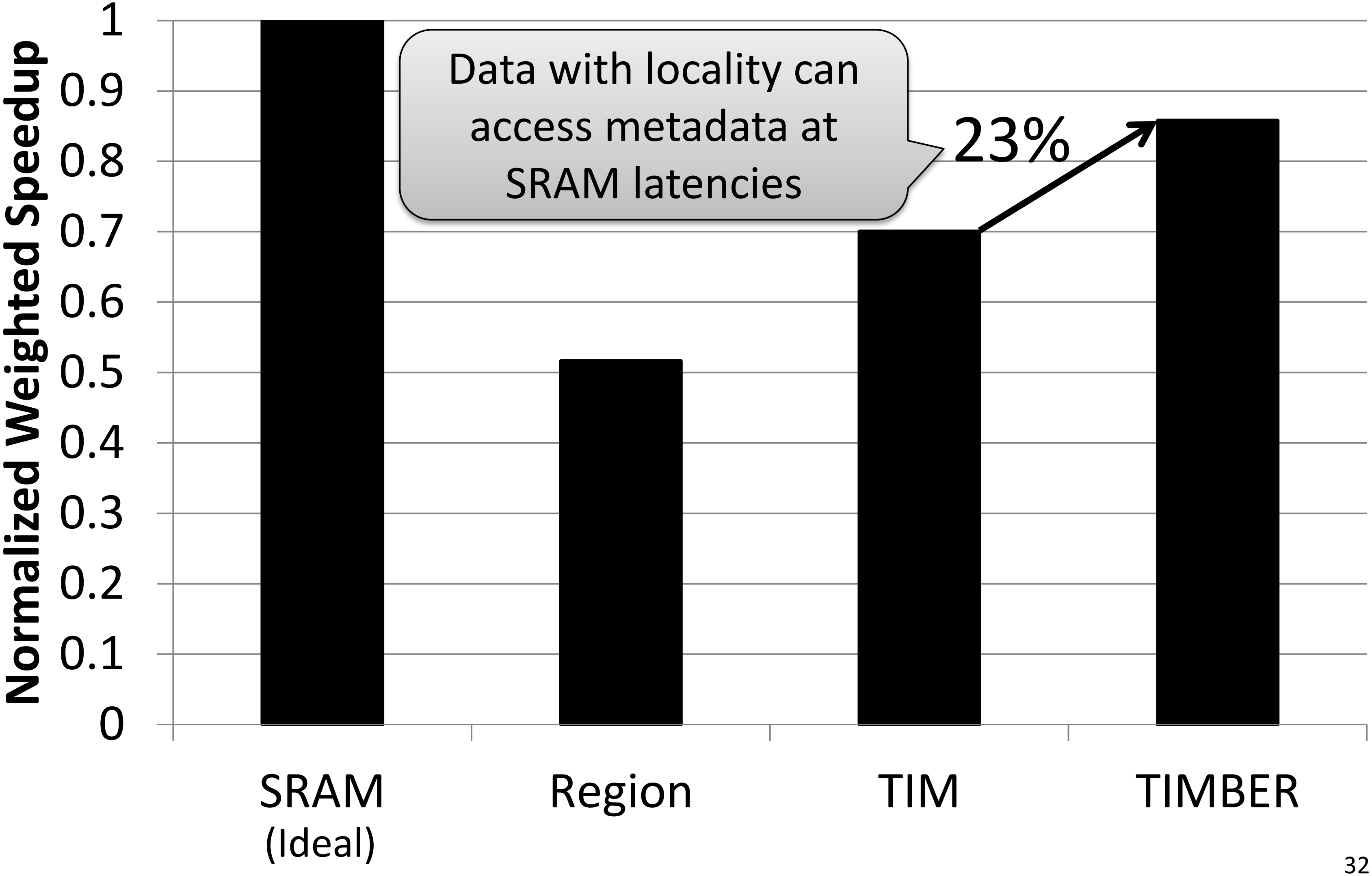
Performance degrades due to increased metadata lookup access latency

Metadata Storage Performance



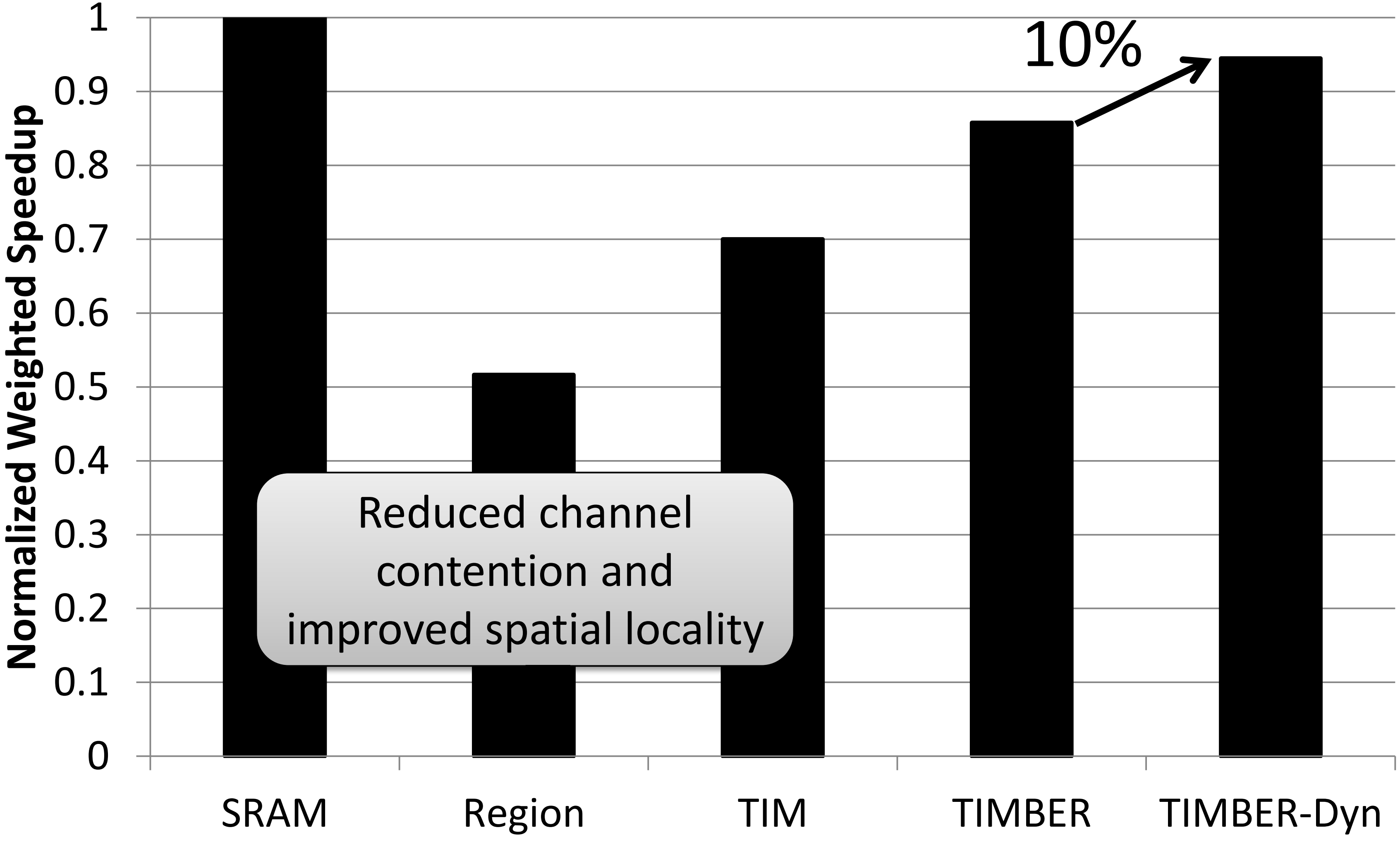
Increased row locality reduces average memory access latency

Metadata Storage Performance



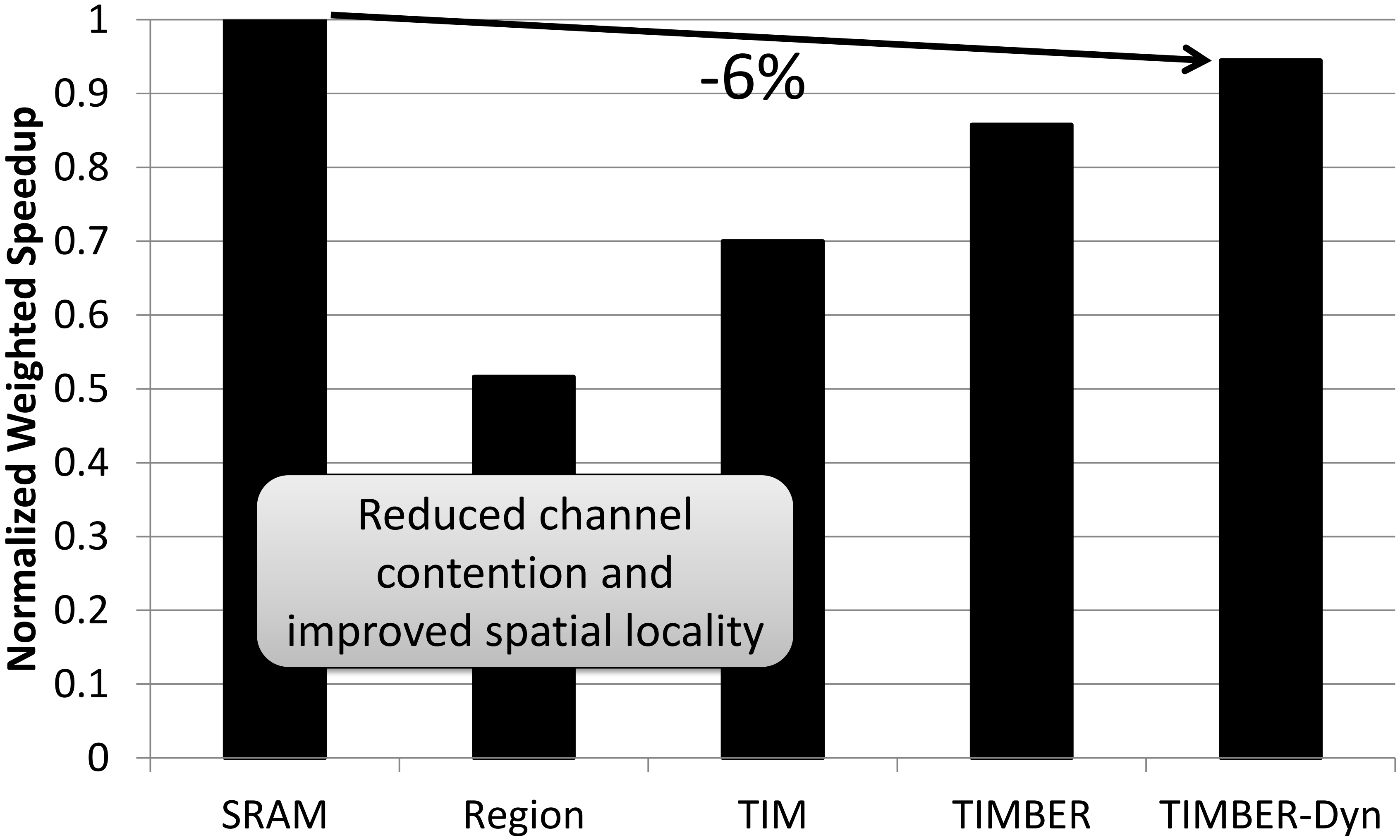
Data with locality can access metadata at SRAM latencies

Dynamic Granularity Performance



Reduced channel contention and improved spatial locality

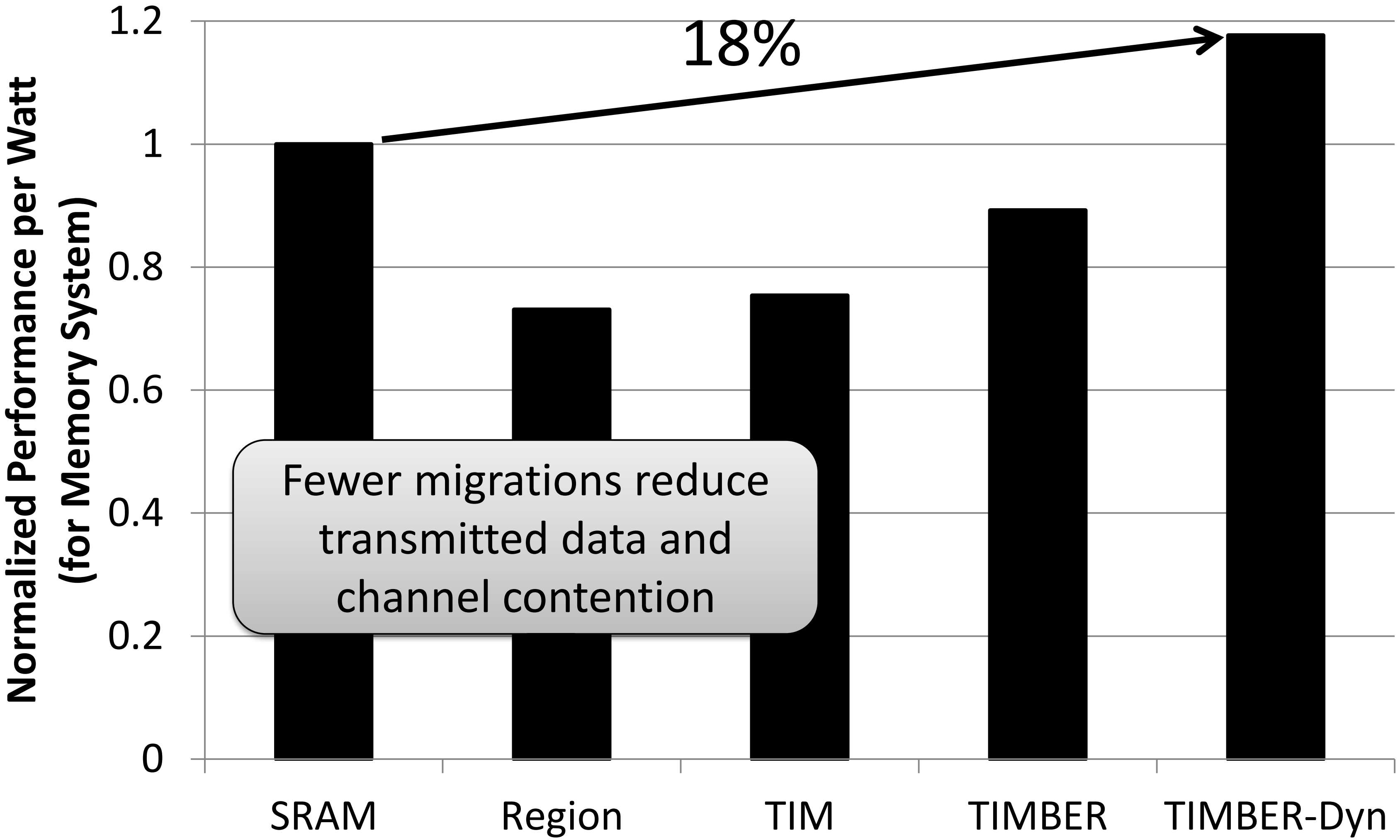
TIMBER Performance



Reduced channel contention and improved spatial locality

Meza, Chang, Yoon, Mutlu, Ranganathan, "Enabling Efficient and Scalable Hybrid Memories," IEEE Comp. Arch. Letters, 2012.

TIMBER Energy Efficiency

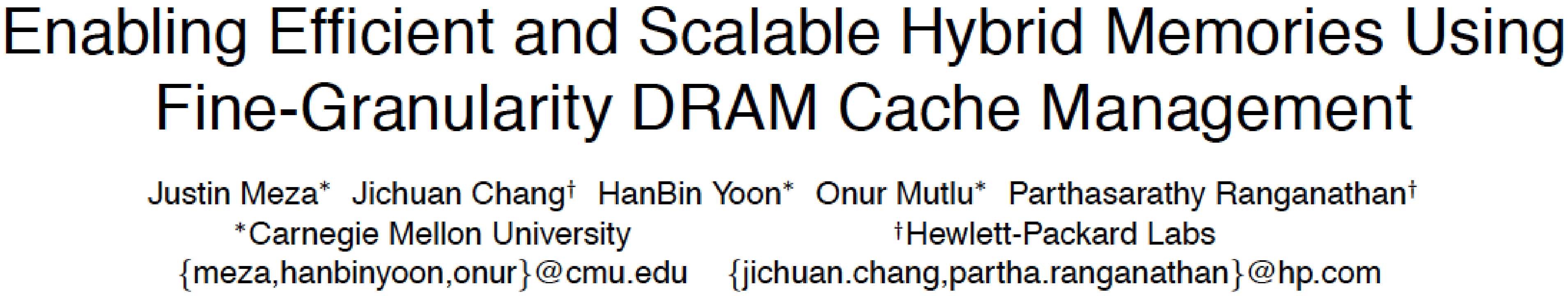


Fewer migrations reduce transmitted data and channel contention

Meza, Chang, Yoon, Mutlu, Ranganathan, “Enabling Efficient and Scalable Hybrid Memories,” IEEE Comp. Arch. Letters, 2012.

More on Large DRAM Cache Design

- Justin Meza, Jichuan Chang, HanBin Yoon, Onur Mutlu, and Parthasarathy Ranganathan, **"Enabling Efficient and Scalable Hybrid Memories Using Fine-Granularity DRAM Cache Management"** *IEEE Computer Architecture Letters* (***CAL***), February 2012.



Enabling Efficient and Scalable Hybrid Memories Using Fine-Granularity DRAM Cache Management

Justin Meza[*] Jichuan Chang[†] HanBin Yoon[*] Onur Mutlu[*] Parthasarathy Ranganathan[†]
[*]Carnegie Mellon University [†]Hewlett-Packard Labs
{meza,hanbinyoon,onur}@cmu.edu {jichuan.chang,partha.ranganathan}@hp.com

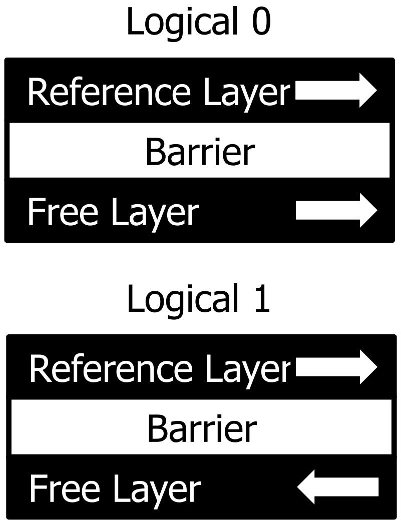
STT-MRAM as Main Memory

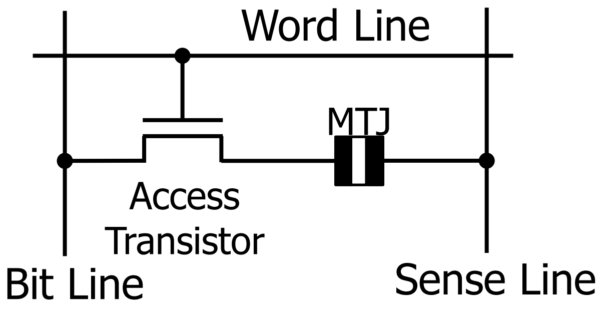
- Magnetic Tunnel Junction (MTJ) device
  - Reference layer: Fixed magnetic orientation
  - Free layer: Parallel or anti-parallel

- Magnetic orientation of the free layer determines logical state of device
  - High vs. low resistance

- Write: Push large current through MTJ to change orientation of free layer
- Read: Sense current flow

- Kultursay et al., "Evaluating STT-RAM as an Energy-Efficient Main Memory Alternative," ISPASS 2013.



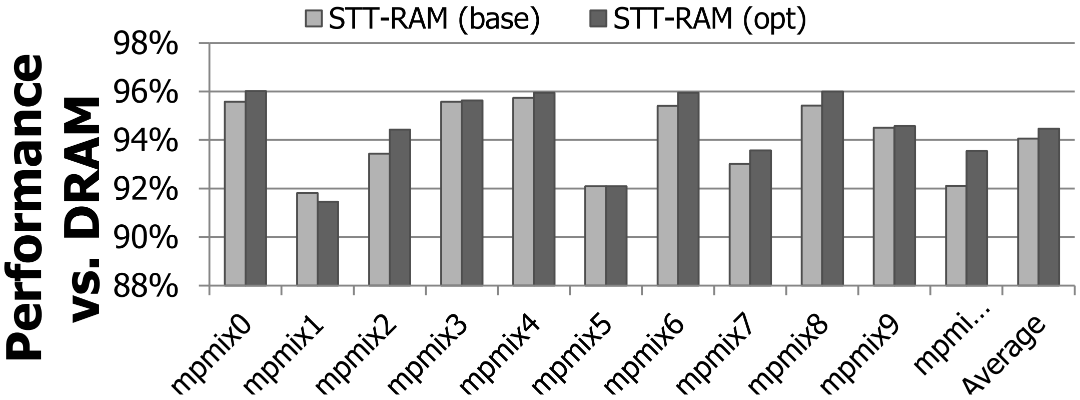


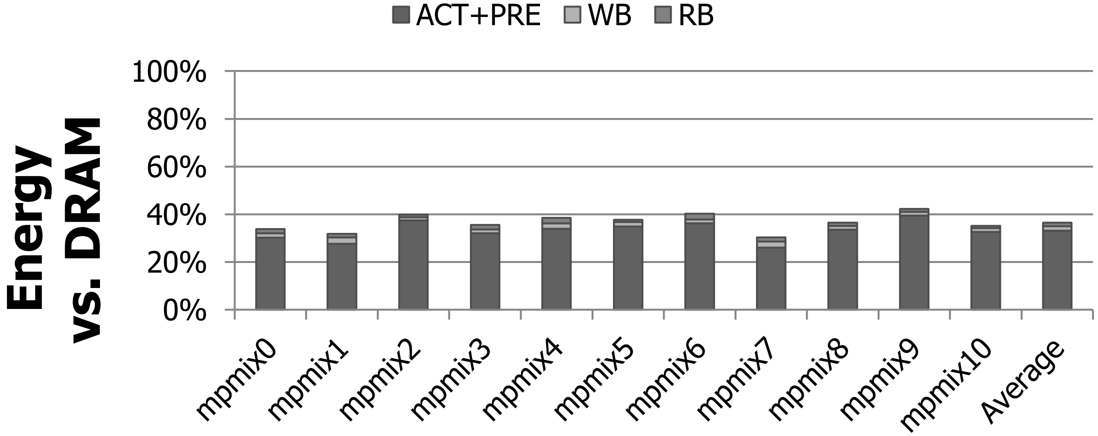
STT-MRAM: Pros and Cons

- Pros over DRAM
  - Better technology scaling
  - Non volatility
  - Low idle power (no refresh)
- Cons
  - Higher write latency
  - Higher write energy
  - Reliability?
- Another level of freedom
  - Can trade off non-volatility for lower write latency/energy (by reducing the size of the MTJ)

Architected STT-MRAM as Main Memory

- 4-core, 4GB main memory, multiprogrammed workloads
- ~6% performance loss, ~60% energy savings vs. DRAM





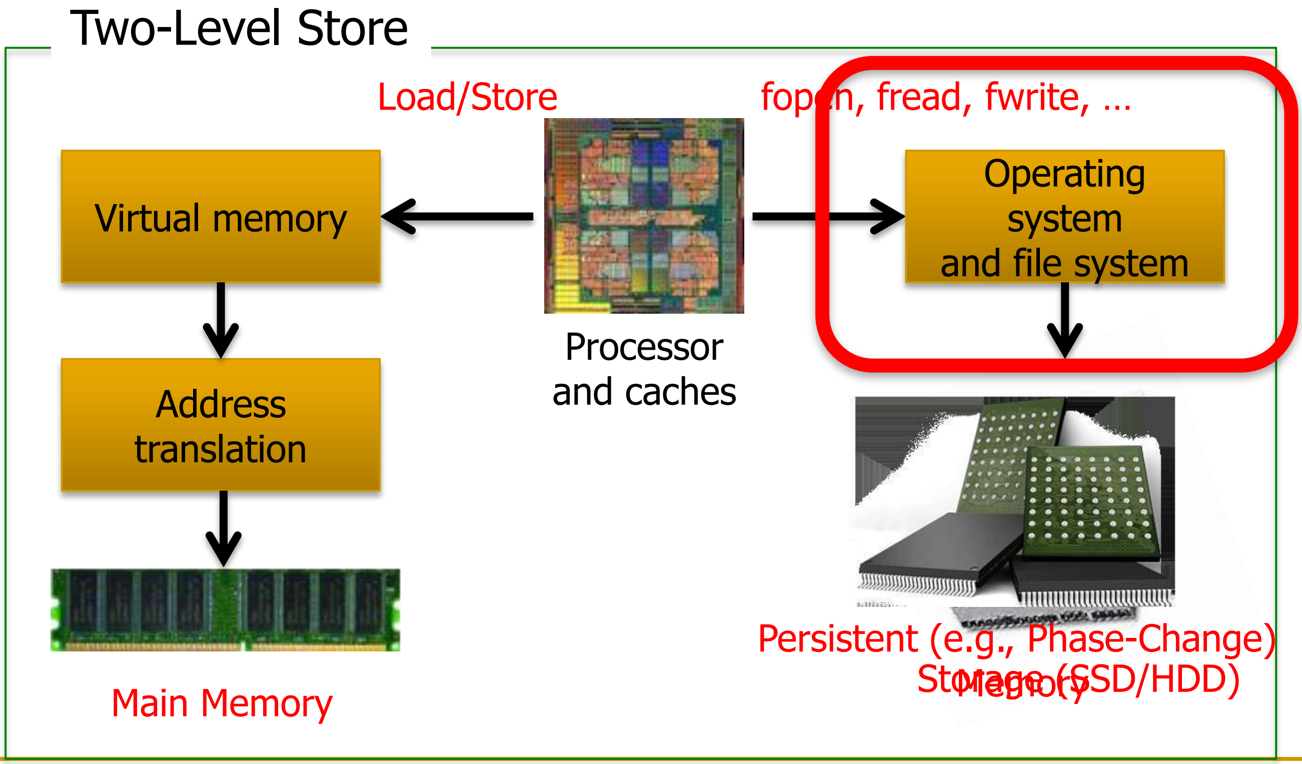
Kultursay+, "Evaluating STT-RAM as an Energy-Efficient Main Memory Alternative," ISPASS 2013.

Other Opportunities with Emerging Technologies

- Merging of memory and storage
  - e.g., a single interface to manage all data
- New applications
  - e.g., ultra-fast checkpoint and restore
- More robust system design
  - e.g., reducing data loss
- Processing tightly-coupled with memory
  - e.g., enabling efficient search and filtering

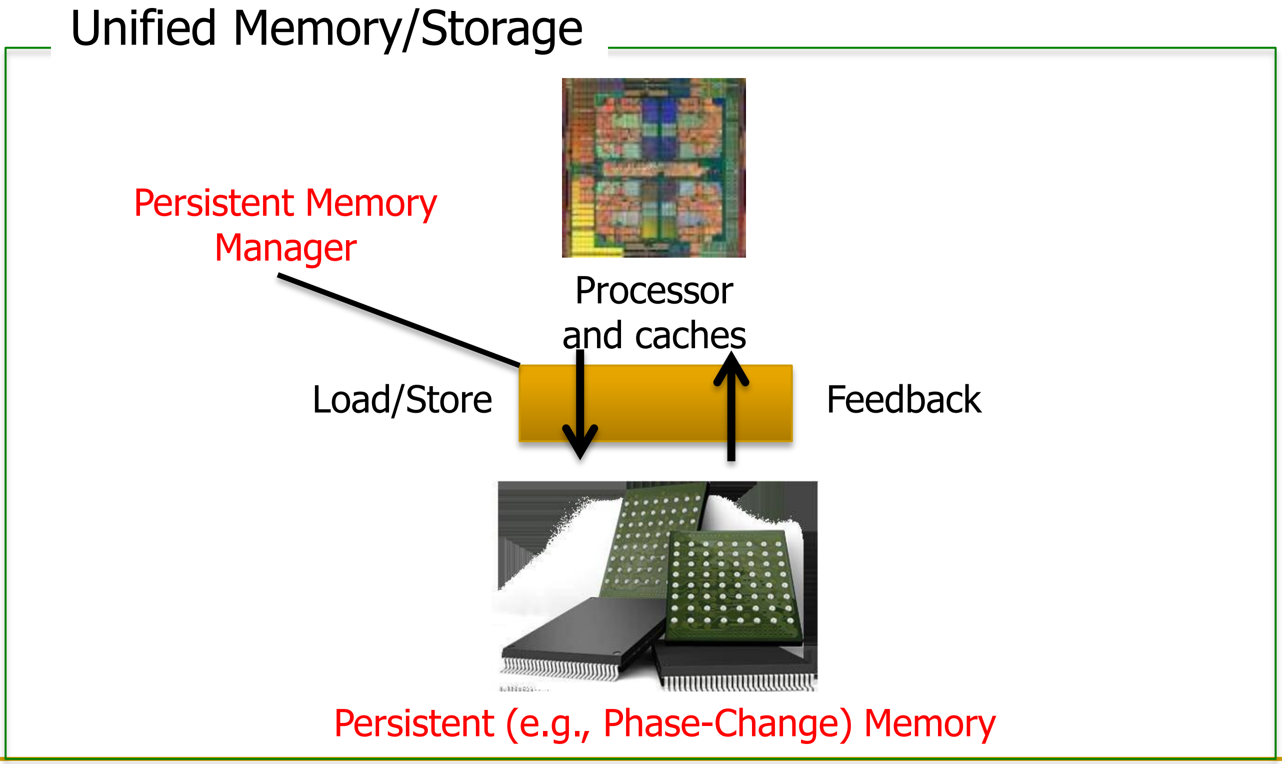
Coordinated Memory and Storage with NVM (I)

- The traditional two-level storage model is a bottleneck with NVM
  - **Volatile** data in memory → a **load/store** interface
  - **Persistent** data in storage → a **file system** interface
  - Problem: Operating system (OS) and file system (FS) code to locate, translate, buffer data become performance and energy bottlenecks with fast NVM stores



Coordinated Memory and Storage with NVM (II)

- Goal: Unify memory and storage management in a single unit to eliminate wasted work to locate, transfer, and translate data
  - Improves both energy and performance
  - Simplifies programming model as well


Unified Memory/Storage
Persistent Memory Manager
Processor and caches
Load/Store
Feedback
Persistent (e.g., Phase-Change) Memory

Meza+, "A Case for Efficient Hardware-Software Cooperative Management of Storage and Memory," WEED 2013.

The Persistent Memory Manager (PMM)

- Exposes a load/store interface to access persistent data
  - Applications can directly access persistent memory → no conversion, translation, location overhead for persistent data

- Manages data placement, location, persistence, security
  - To get the best of multiple forms of storage

- Manages metadata storage and retrieval
  - This can lead to overheads that need to be managed

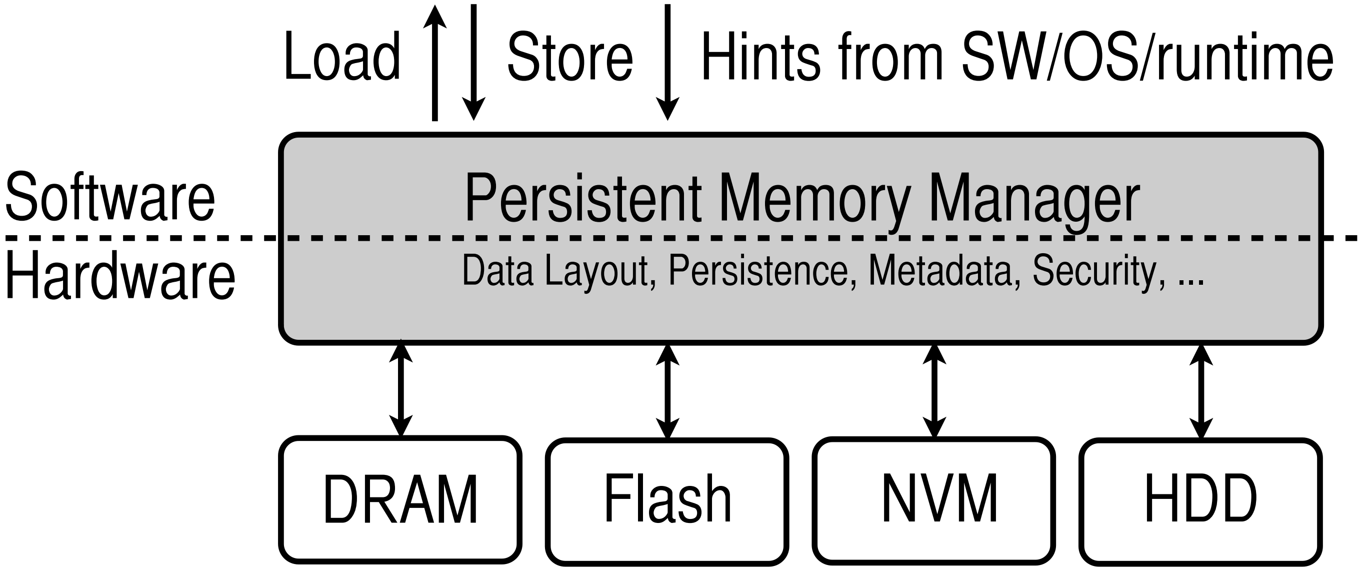
- Exposes hooks and interfaces for system software
  - To enable better data placement and management decisions

- Meza+, "A Case for Efficient Hardware-Software Cooperative Management of Storage and Memory," WEED 2013.

The Persistent Memory Manager (PMM)

```
int main(void) {
  // data in file.dat is persistent
  FILE myData = "file.dat";
  myData = new int[64];
}
void updateValue(int n, int value) {
  FILE myData = "file.dat";
  myData[n] = value; // value is persistent
}
```

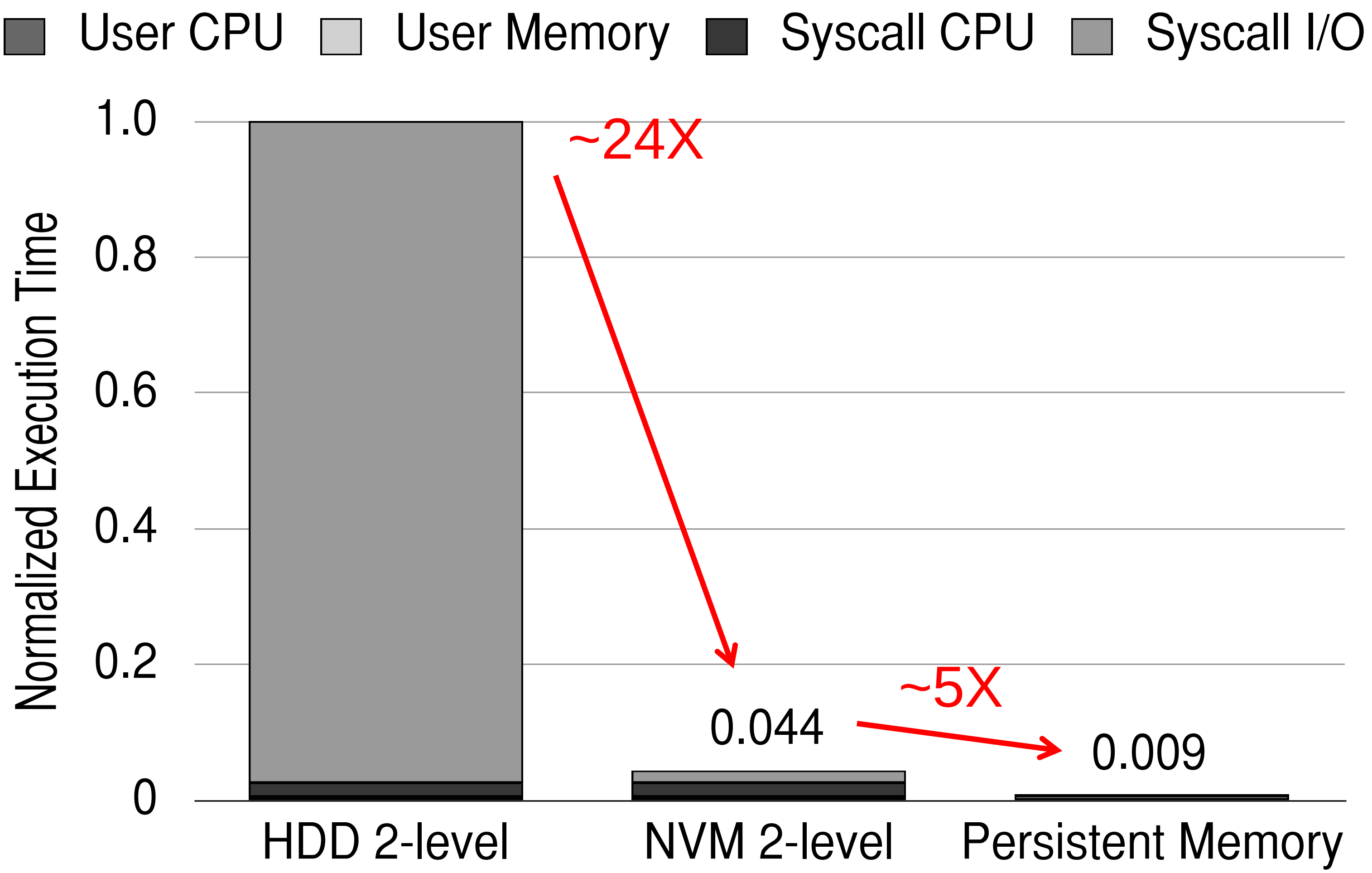
Persistent objects



Load
Store
Hints from SW/OS/runtime
Software
Hardware
Persistent Memory Manager
Data Layout, Persistence, Metadata, Security, ...
DRAM
Flash
NVM
HDD

PMM uses access and hint information to allocate, locate, migrate and access data in the heterogeneous array of devices

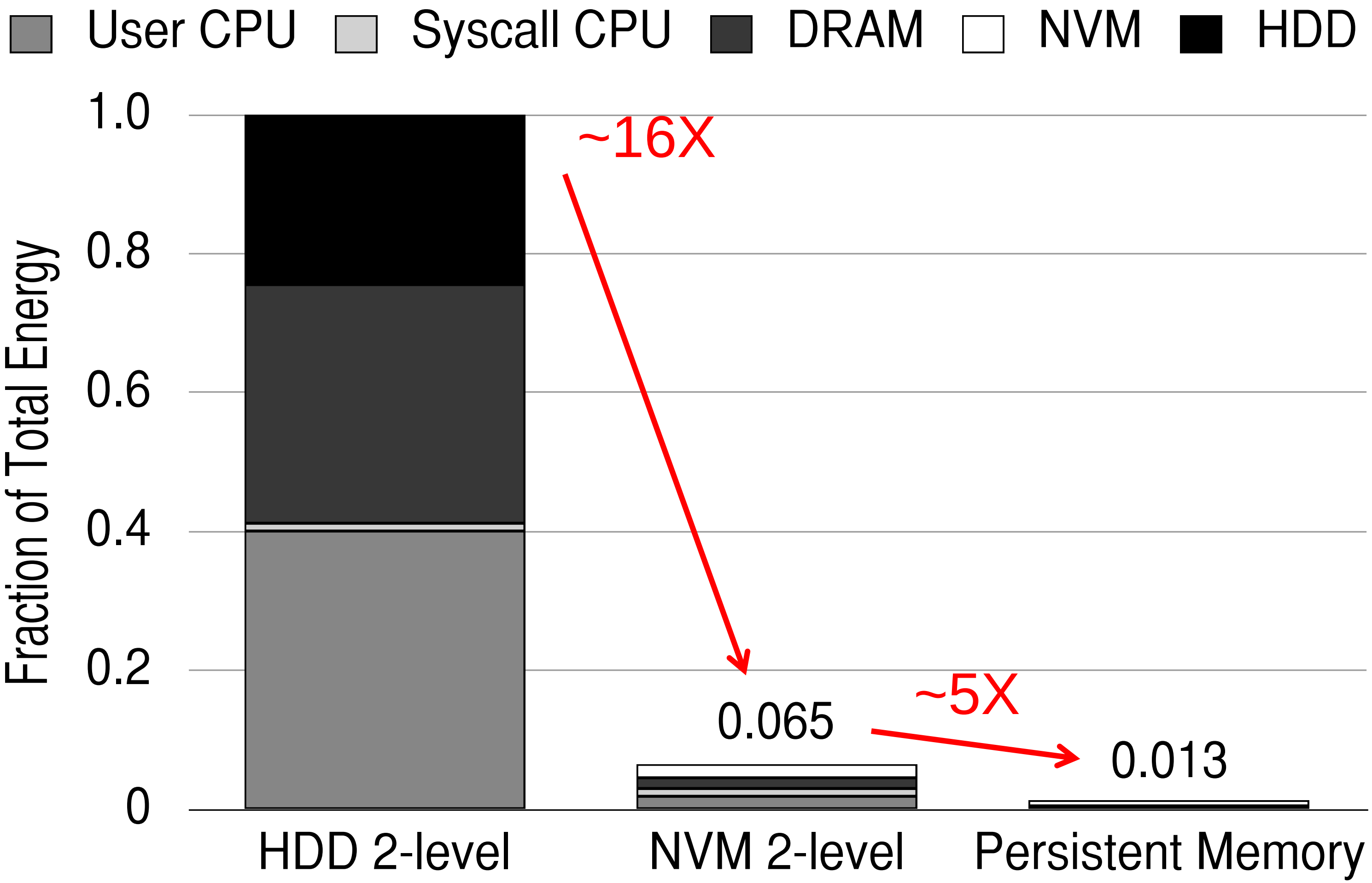
Performance Benefits of a Single-Level Store



SAFARI

Meza+, "A Case for Efficient Hardware-Software Cooperative Management of Storage and Memory," WEED 2013.

Energy Benefits of a Single-Level Store



Meza+, "A Case for Efficient Hardware-Software Cooperative Management of Storage and Memory," WEED 2013.

Some Principles for Memory Scaling

Principles for Memory Scaling (So Far)

- Better cooperation between devices and the system
  - Expose more information about devices to upper layers
  - More flexible interfaces

- Better-than-worst-case design
  - Do not optimize for the worst case
  - Worst case should not determine the common case

- Heterogeneity in design (specialization, asymmetry)
  - Enables a more efficient design (No one size fits all)

- These principles are coupled

Summary: Memory Scaling

- Memory scaling problems are a critical bottleneck for system performance, efficiency, and usability

- New memory architectures
  - **A lot of hope in fixing DRAM**

- Enabling emerging NVM technologies
  - **A lot of hope in hybrid memory systems and single-level stores**

- System-level memory/storage QoS
  - **A lot of hope in designing a predictable system**

- Three principles are essential for scaling
  - Software/hardware/device cooperation
  - Better-than-worst-case design
  - Heterogeneity (specialization, asymmetry)

18-740: Computer Architecture
Recitation 5:
Main Memory Scaling Wrap-Up

Prof. Onur Mutlu
Carnegie Mellon University

Fall 2015

September 29, 2015

Another Discussion: NAND Flash Scaling

- Onur Mutlu,
  "Error Analysis and Management for MLC NAND Flash Memory"
  *Technical talk at Flash Memory Summit 2014 (**FMS**)*, Santa Clara, CA, August 2014. Slides (ppt) (pdf)

Cai+, "Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis," DATE 2012.

Cai+, "Flash Correct-and-Refresh: Retention-Aware Error Management for Increased Flash Memory Lifetime," ICCD 2012.

Cai+, "Threshold Voltage Distribution in MLC NAND Flash Memory: Characterization, Analysis and Modeling," DATE 2013.

Cai+, "Error Analysis and Retention-Aware Error Management for NAND Flash Memory," Intel Technology Journal 2013.

Cai+, "Program Interference in MLC NAND Flash Memory: Characterization, Modeling, and Mitigation," ICCD 2013.

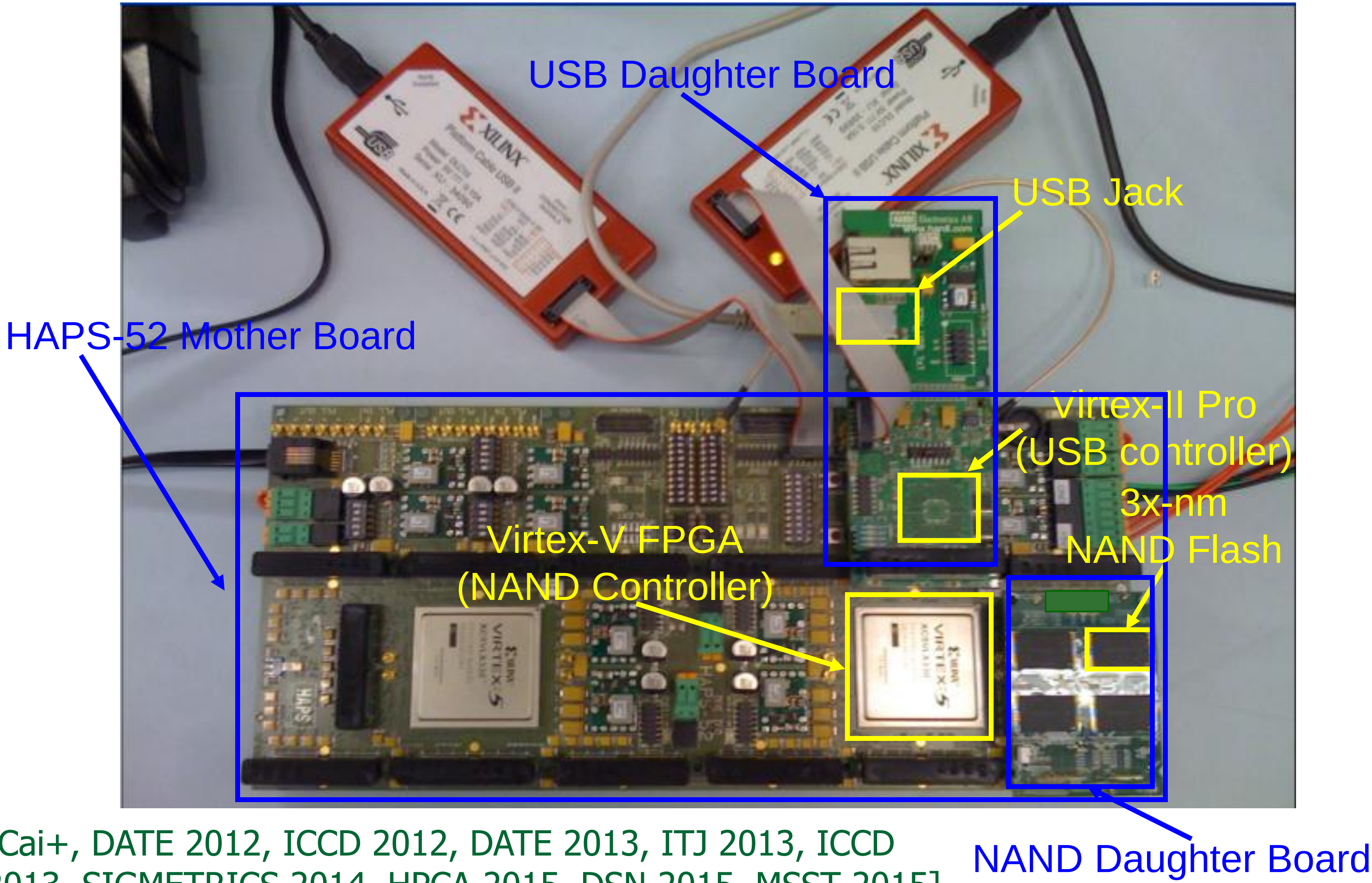
Cai+, "Neighbor-Cell Assisted Error Correction for MLC NAND Flash Memories," SIGMETRICS 2014.

Cai+,"Data Retention in MLC NAND Flash Memory: Characterization, Optimization and Recovery," HPCA 2015.

Cai+, "Read Disturb Errors in MLC NAND Flash Memory: Characterization and Mitigation," DSN 2015.

Luo+, "WARM: Improving NAND Flash Memory Lifetime with Write-hotness Aware Retention Management," MSST 2015.

Experimental Infrastructure (Flash)



[Cai+, DATE 2012, ICCD 2012, DATE 2013, ITJ 2013, ICCD 2013, SIGMETRICS 2014, HPCA 2015, DSN 2015, MSST 2015]