

Understanding the Impact of Emerging Non-Volatile Memories on High-Performance, IO-Intensive Computing

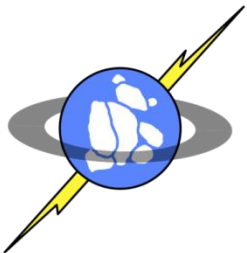
Adrian M. Caulfield

Joel Coburn, Todor I. Mollov, Arup De, Ameen Akel, Jiahua He[†], Arun Jagatheesan[†], Rajesh K. Gupta, Allan Snavely[†], Steven Swanson

Non-Volatile Systems Laboratory,
Department of Computer Science and Engineering

[†] San Diego Supercomputer Center

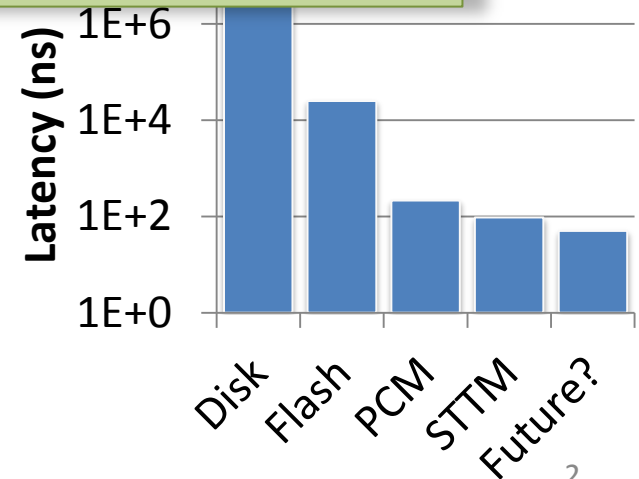
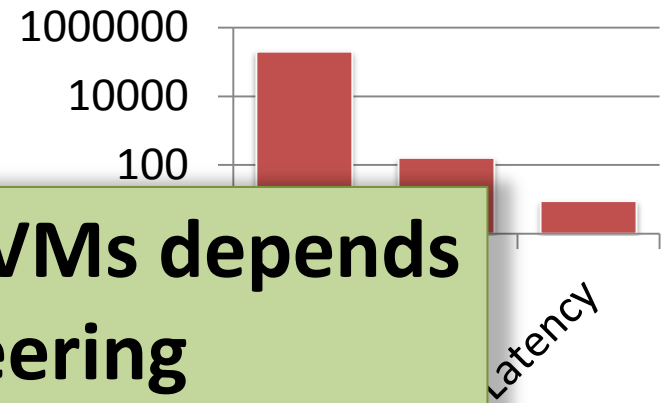
University of California, San Diego



Revolution in Storage Technology

- Slow disks drive IO system design
- Leveraging the benefits of NVMs depends critically on re-engineering hardware *and* software
- Emerging NVIMs – 10,000X faster
 - Revolutionary change

Performance Growth since 1970



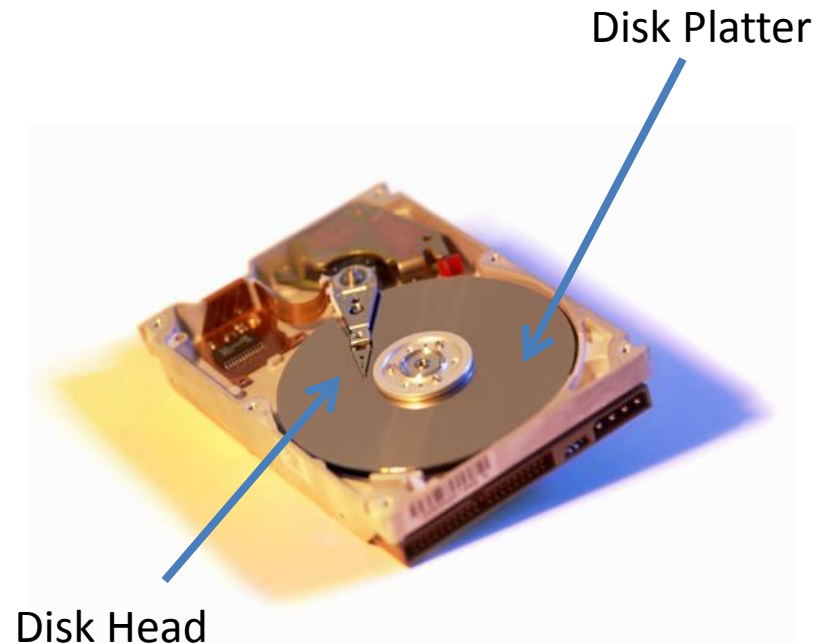
Overview

- Motivation
- **Storage Technologies**
- HASTE
- Performance Analysis

Hard Disks

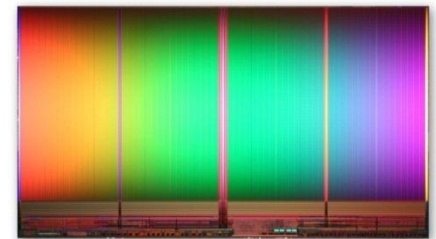
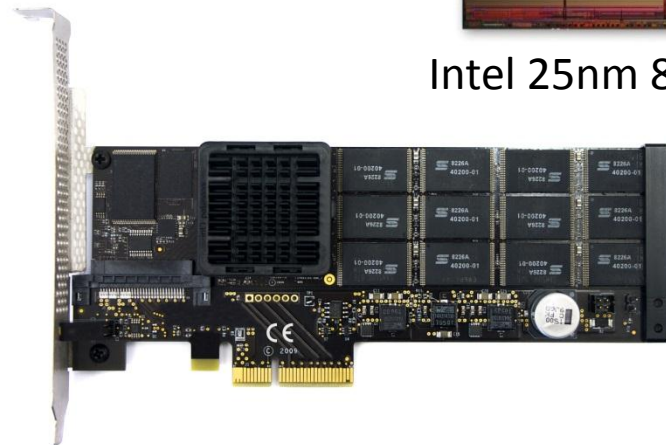
- “Standard” storage device of the last decades
- Characteristics
 - Slow random access latency (5-10 ms)
 - Sustained BW: 138 MB/s

	Read	Write
Latency	5-10 ms	5-10 ms



NAND Flash Memory

- Heavy-duty firmware layer for management
 - Wear leveling
 - Hide other idiosyncrasies
- Two flash SSD interconnects
 - SATA
 - PCIe



Intel 25nm 8GB flash die

	Read	Write
Latency	25 us	200 us

Phase Change Memories and Spin-Torque Transfer Memories

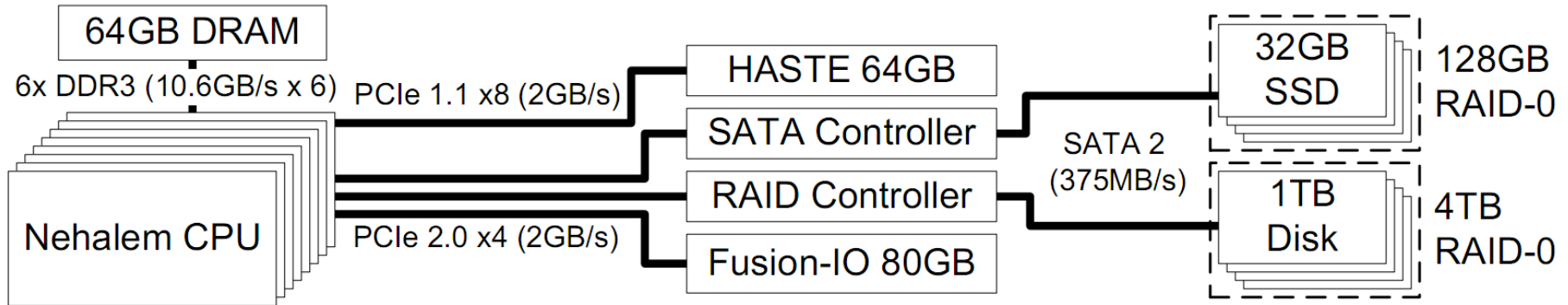
- Simpler wear leveling compared to Flash
 - Byte addressable
 - Higher endurance
- No firmware layer required
- Interface very similar to DRAM
- Possible to put PCM/STTM on DDR bus

Projected Latencies	Read	Write
PCM	65 ns	215 ns
STTM	29.5 ns	95 ns

Overview

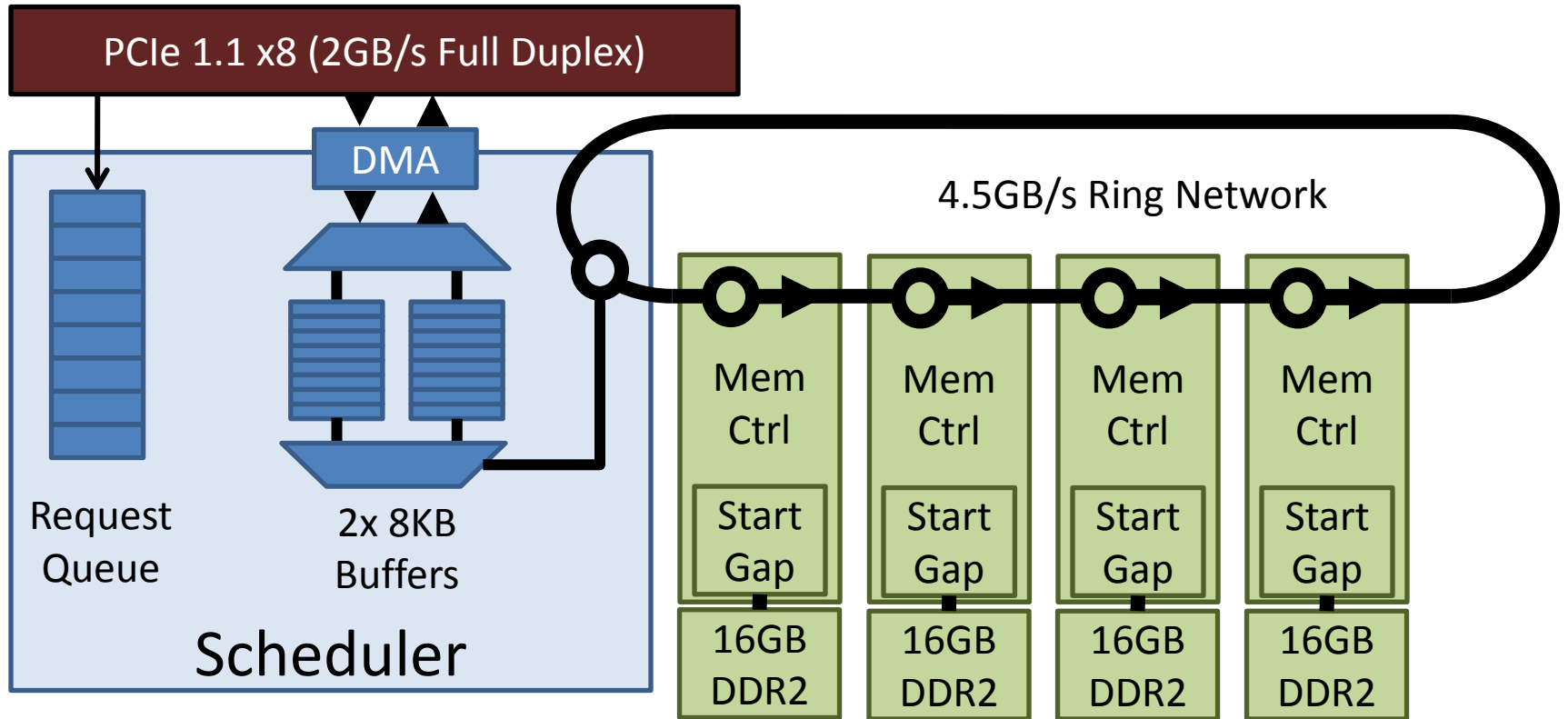
- Motivation
- Storage Technologies
- **HASTE**
- Performance Analysis

System Overview



Memory and Device	Interconnect	Capacity
Fusion-I/O IODrive	PCIe 2.0 4x	80GB
SLC NAND Flash SW RAID-0	PCIe 2.0 4x SATA 2 Controller	128GB
Disk HW RAID-0	PCIe 2.0 4x RAID Controller	4TB
DDR3-attached PCM or STTM	6x DDR3 Channels	64GB
PCIe-attached PCM or STTM	PCIe 1.1 8x	64GB

HASTE Architecture



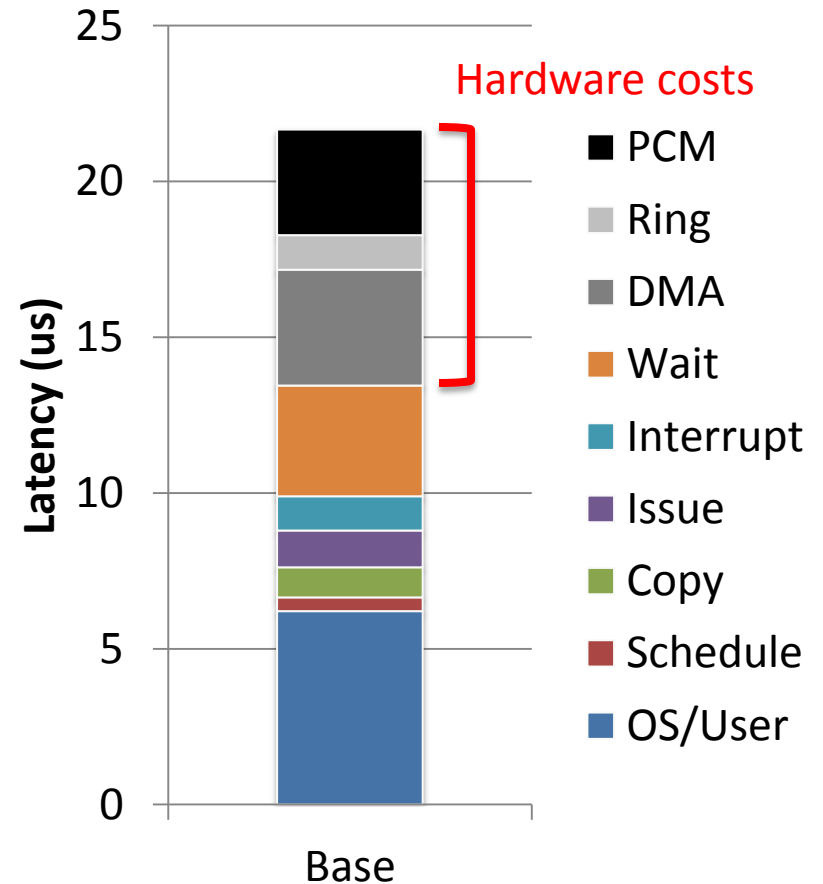
HASTE: Modeling Advanced NVMs

- FPGAs connected via PCIe
- DDR2 DRAM emulates NVMs
 - Adjust DDR timing
 - t_{rcd} : RAS-CAS Delay
 - t_{wrp} : Write/Read



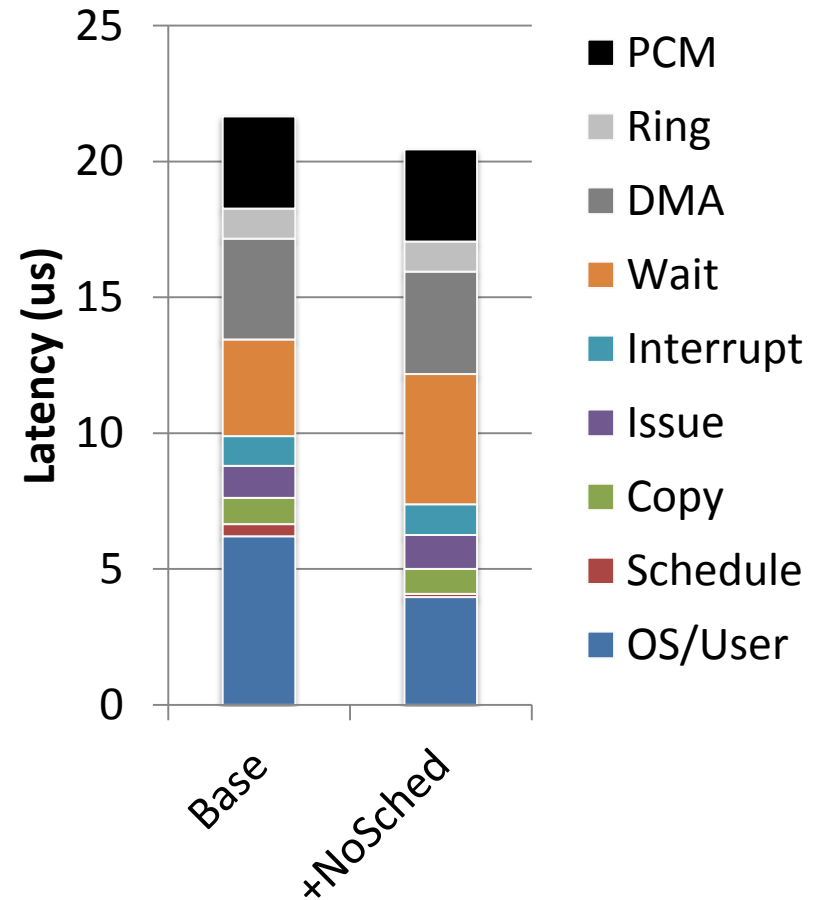
Software is Critical

- Baseline Latencies:
 - Hardware: 8.2 us
 - Software: 13.4 us



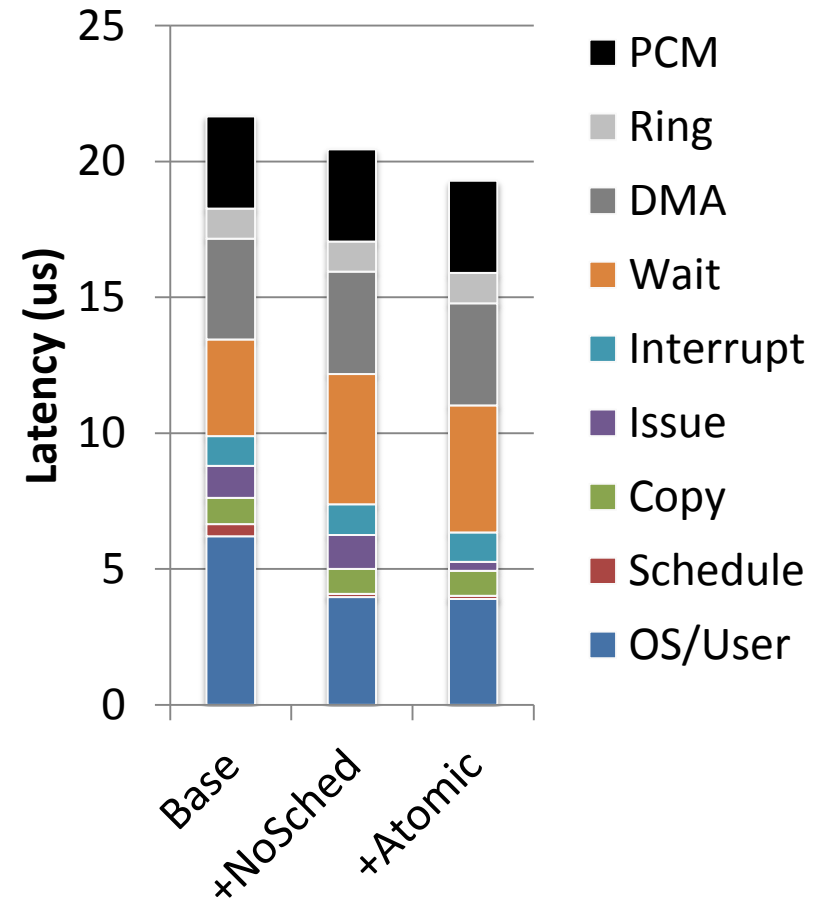
Removing the IO Scheduler

- Reduces both IO sched. and OS time
- 10% software latency savings



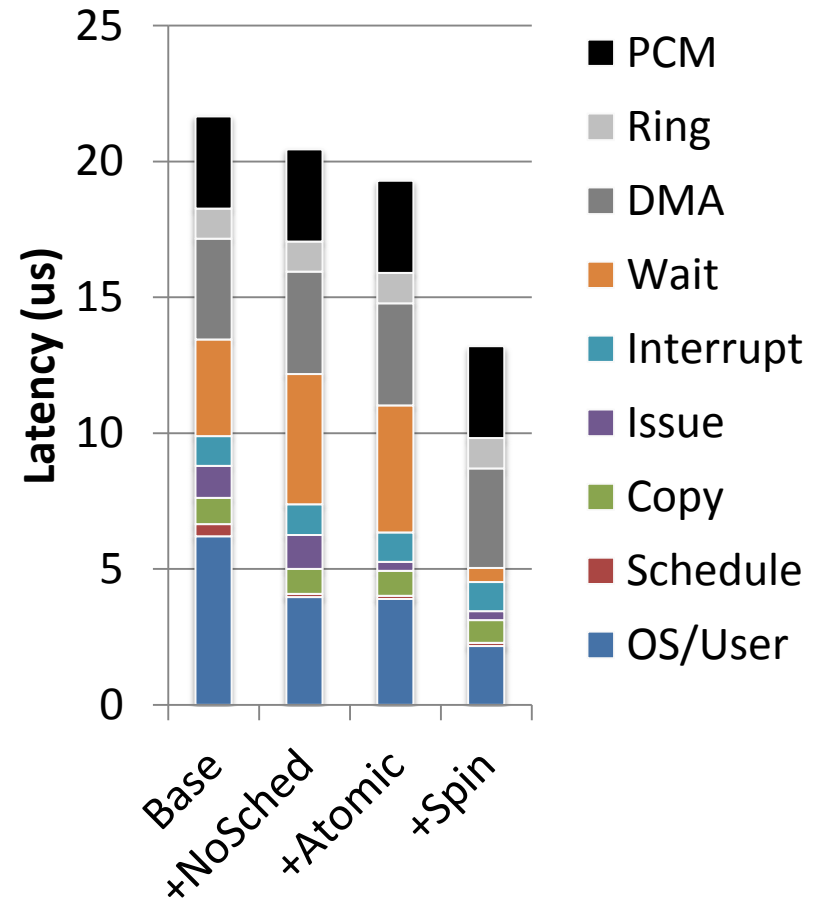
Atomic Operations

- Co-designed HW interface and kernel
 - Eliminated locks
 - Increased concurrency
- 10% savings vs NoSched



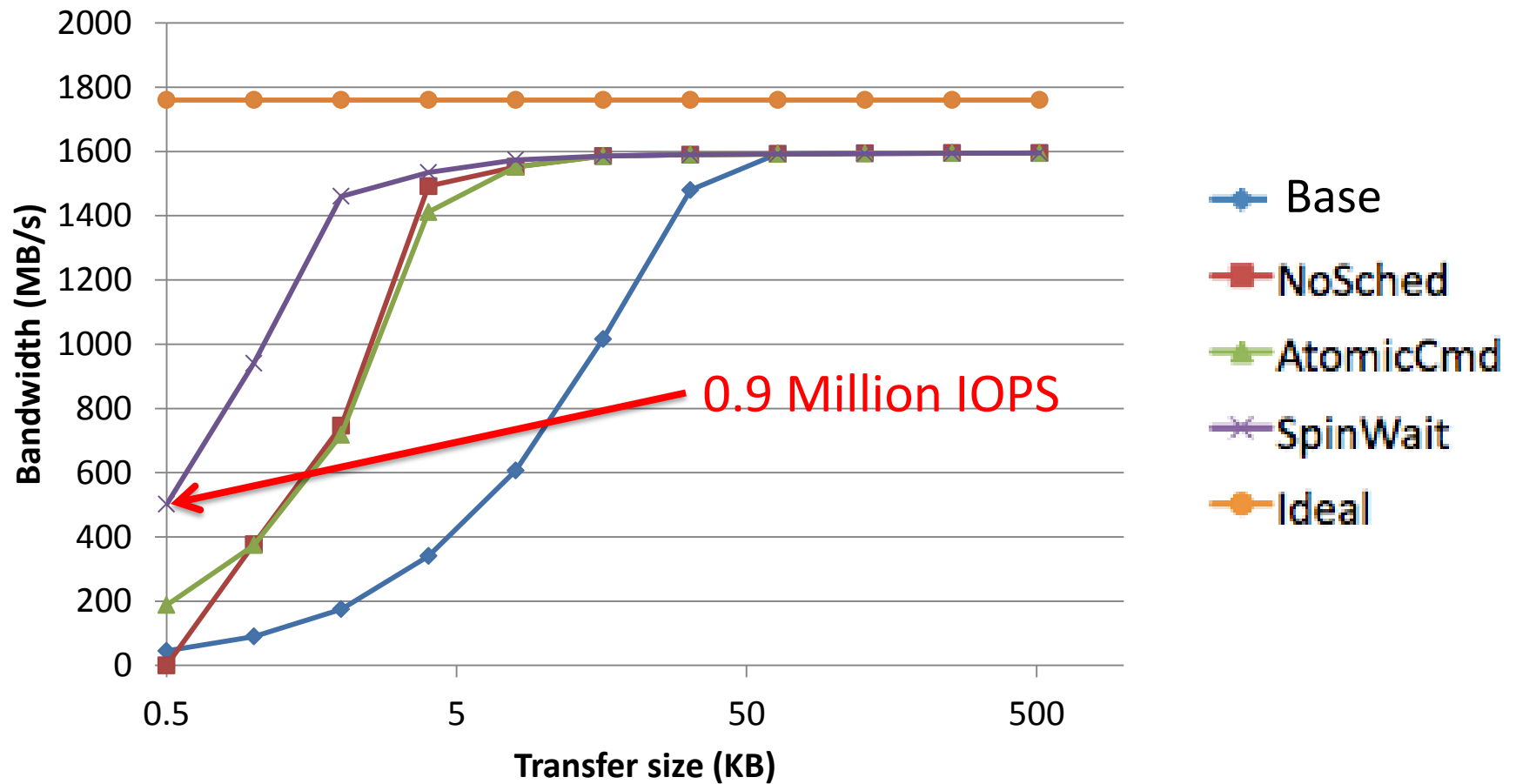
Add Spin-Waits

- Spin-waits are faster than sleeping
 - Helps for < 4KB requests
 - Sleep for larger requests.
- 5 us of software
 - 54% less SW vs. Atomic
 - 62% vs. Base



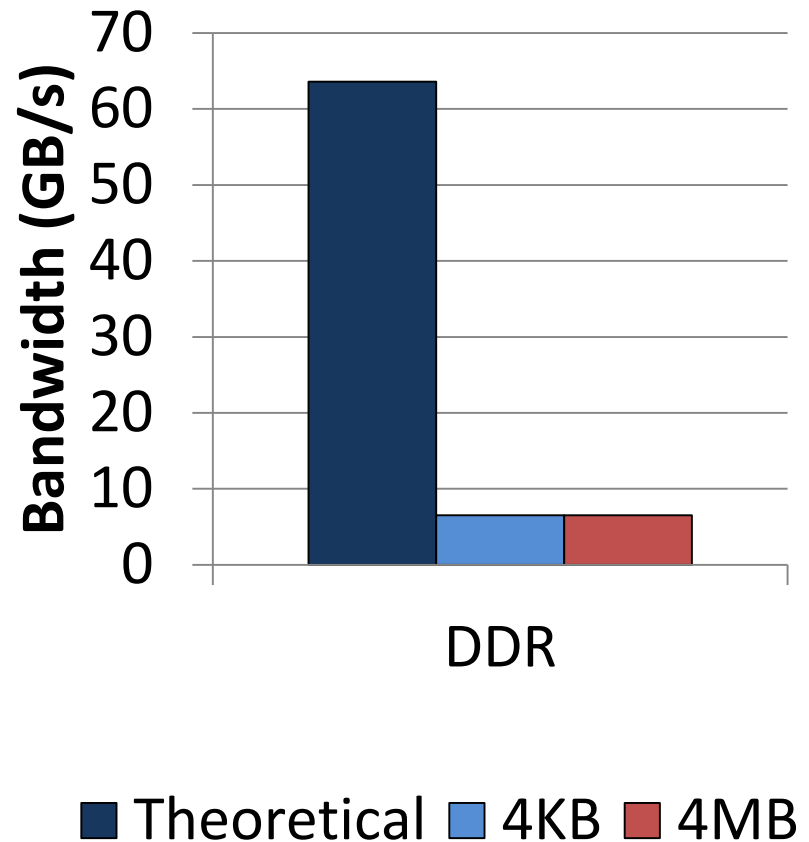
HASTE Bandwidth

Random Writes



DDR Efficiency

- DDR attached SSD gets only 11% BW utilization
- No improvement for large requests
- Possible limitations:
 - Driver and OS overhead
 - CPU throughput



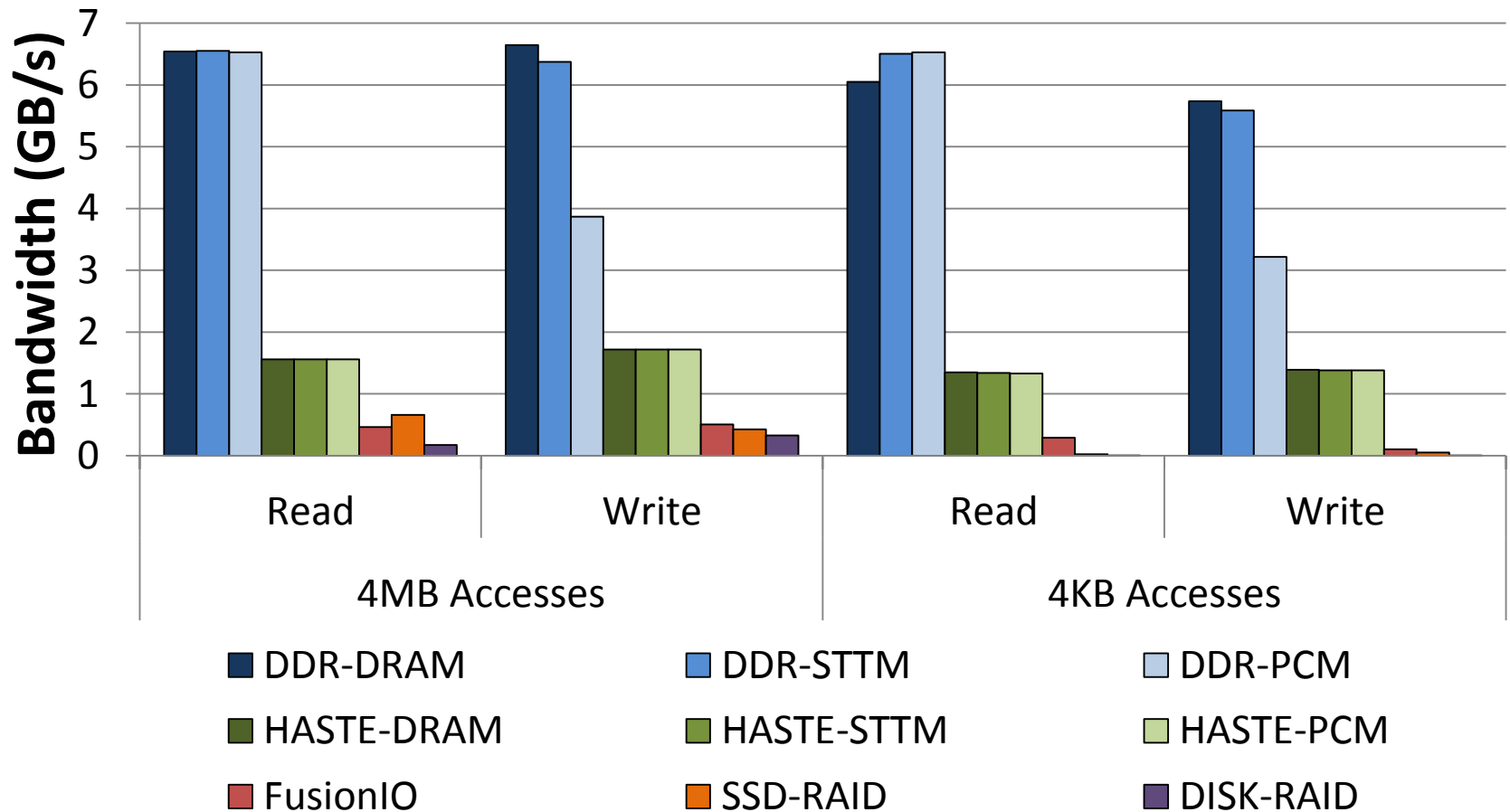
Overview

- Motivation
- Storage Technologies
- HASTE
- **Performance Analysis**

Workloads

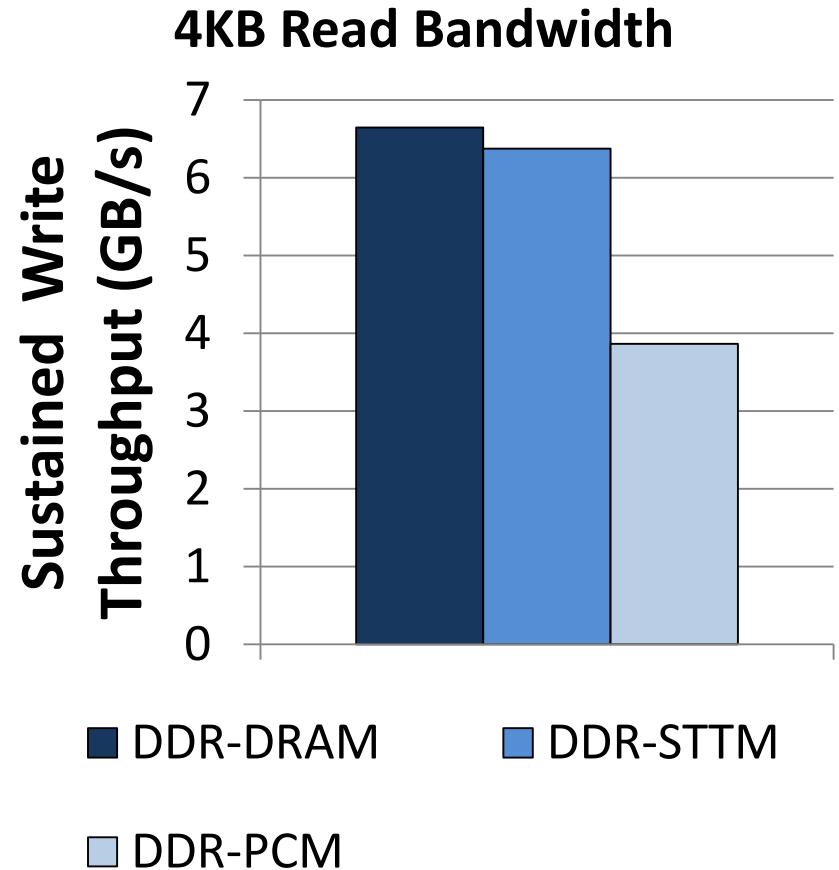
Name	Footprint	Description
Basic IO Benchmarks		
XDD NoFS	64 GB	Low-level IO performance without file system
XDD XFS	64 GB	XFS file system performance
Database Applications		
Berkeley-DB Btree	16 GB	Transactional updates to btree key/value store
Berkeley-DB HashTable	16 GB	Transactional updates to hash table key/value store
BiologicalNetworks	35 GB	Biological database queried for properties of genes and biological-networks
PTF	50 GB	Palomar Transient Factory sky survey queries
Memory-hungry Applications		
DGEMM	21 GB	Matrix multiply with 30,000 x 30,000 matrices
NAS Parallel Benchmarks	8-35 GB	7 apps from NPB suite modeling scientific workloads

Bandwidth w/o File System

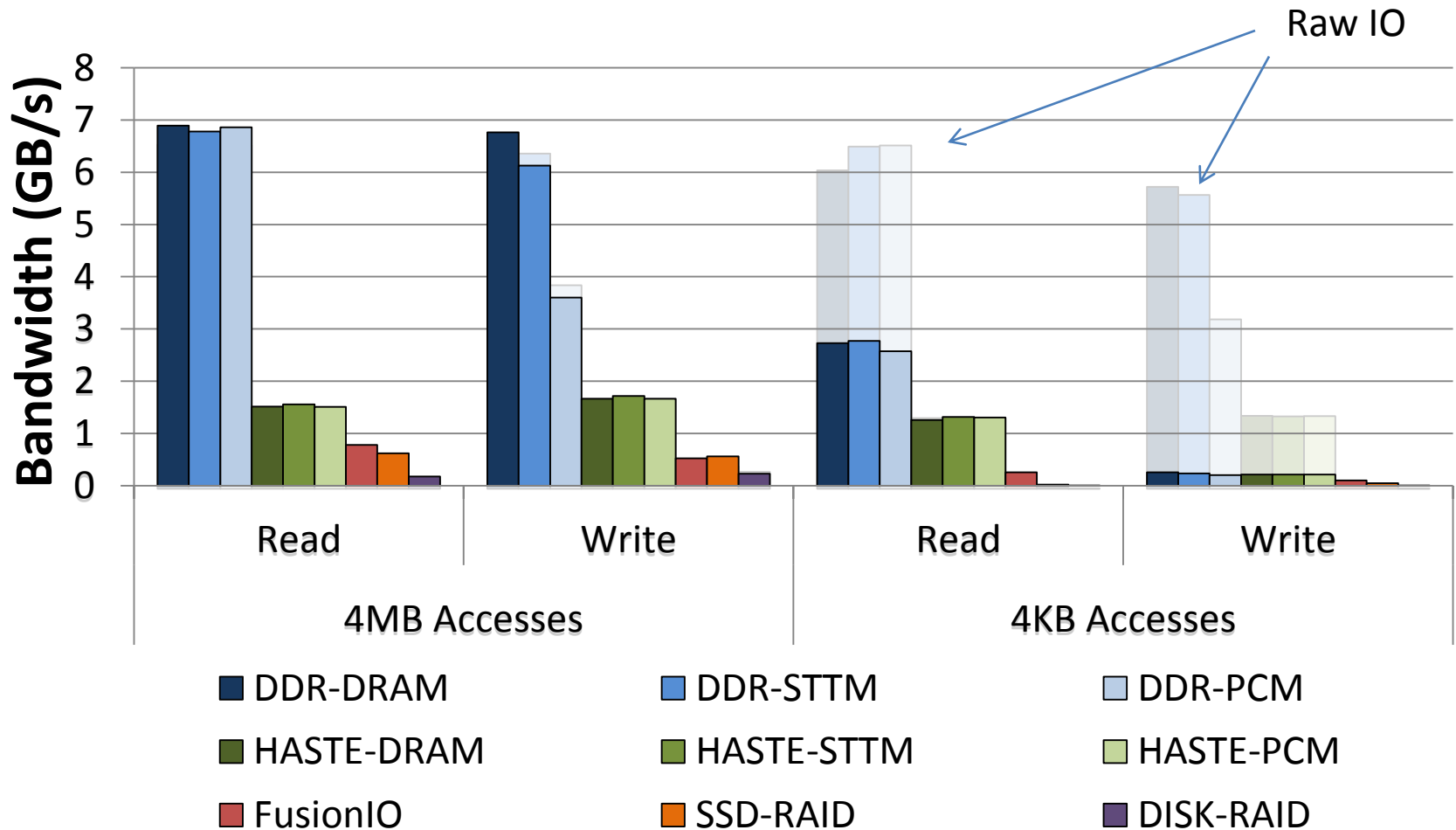


DDR Performance Trend

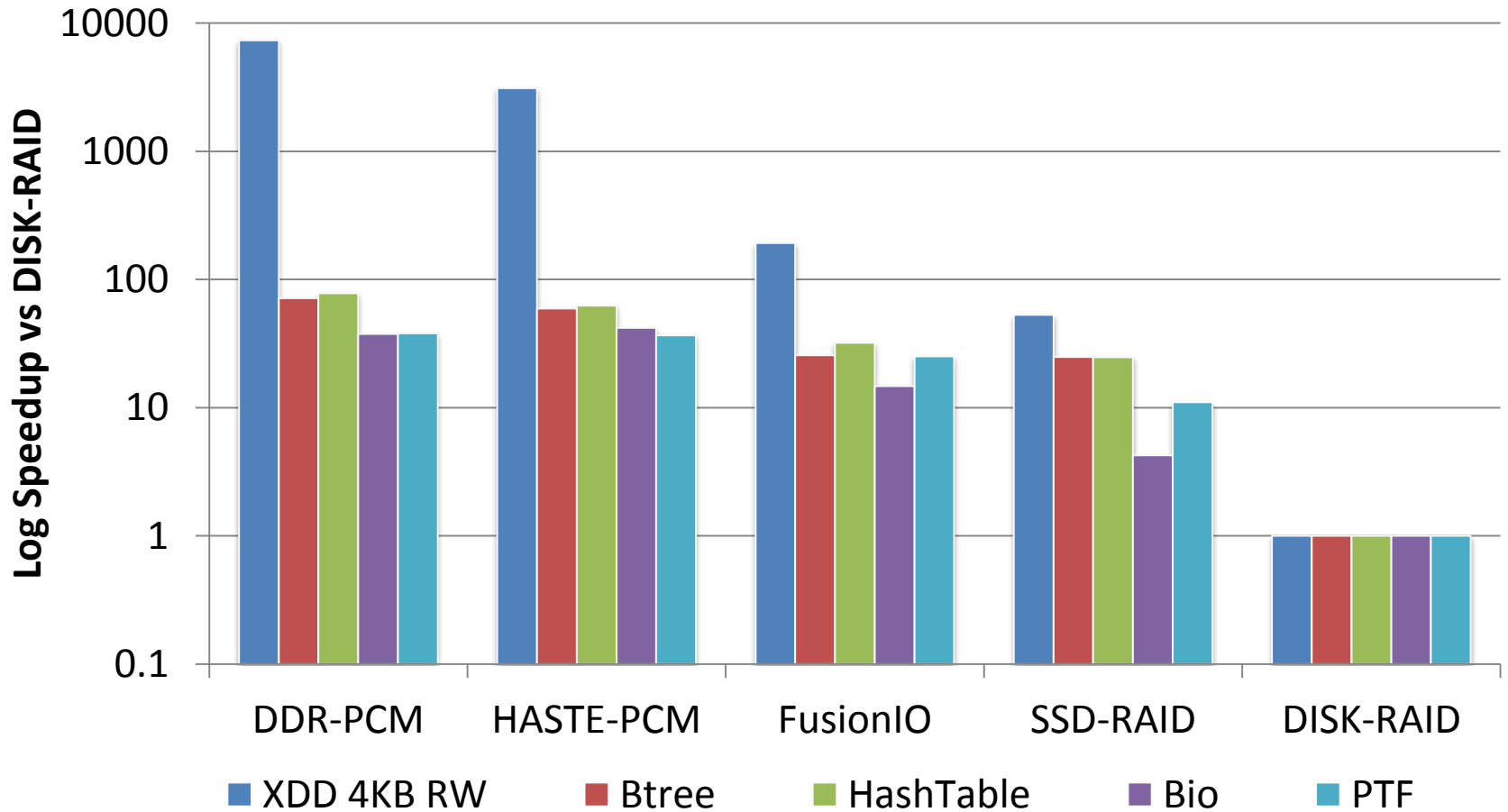
- DDR bus exposes latency
- Request per cache-lines
 - Memory latency on each line
 - 128 row access latencies/8KB



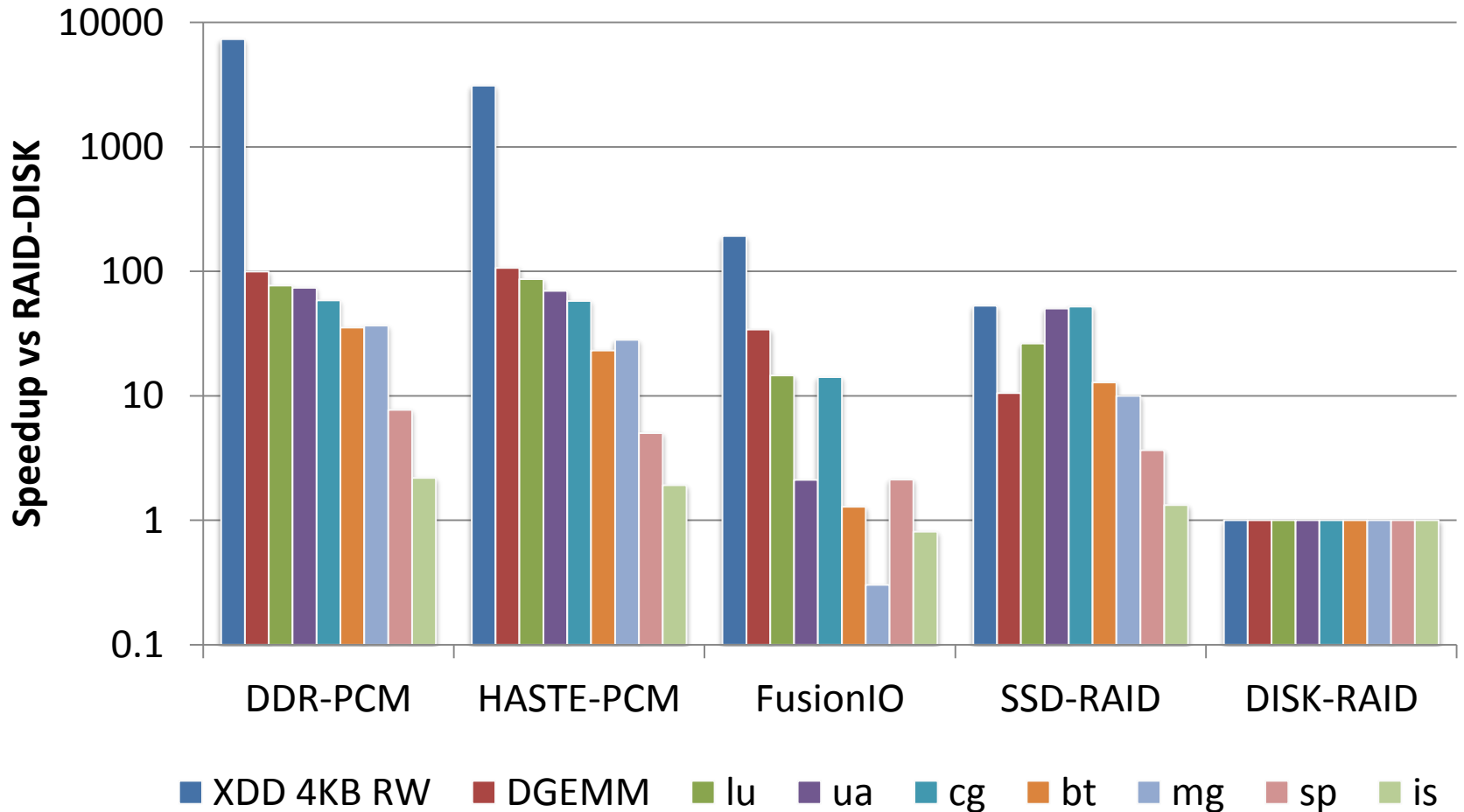
Bandwidth with XFS



Database Performance



Memory-Hungry App Performance



Power and Cost Analysis

NVM		
Configuration	Applications	Costs
Primary storage	Intensive random read/writes Small (< 1 TB) per-node data sets	Extremely low cost/IOPS Power similar to current SSDs
Cache for disk array	Checkpointing, web services, large data sets, locality, bursty writes	Reduced cost/IOPS Power negligible vs. disk array
Hybrid	Database logging, swap space, fast access to fixed fraction of data	Reduced cost/IOPS Power negligible vs. disk array

The cost effectiveness of fast NVMs depends whether we can optimize these applications to exploit them

Conclusion

- Enormous memory device latency reduction
- 7,500x Raw IO and 100x Application gains
- Software is not ready to take advantage of fast NVMs
- PCM, STTM, others will cause even larger changes
- We optimized the kernel driver from 13us to 5us
- Applications need to be optimized also

Thank You!

Any Questions?

