Redesigning Transaction Mechanisms for Fast, Solid-State Disks

Trevor Bunker, **Joel Coburn**

Rajesh K. Gupta, Steven Swanson

2012 Flash Memory Summit

NVSL
Non-volatile Systems Laboratory

UCSDCSE
Computer Science and Engineering

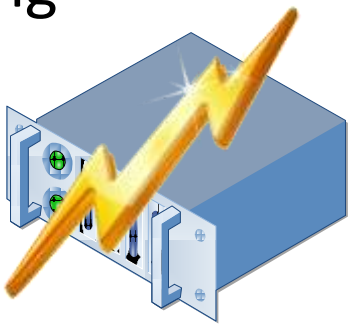
The Future of Storage

Hard Drives	PCIe-Flash 2007	PCIe-NVM 2013?
		
Lat.: 7.1ms	68us	8.2us
1x	104x	865x = 3.1x/yr
BW: 2.6MB/s	250MB/s	1.6GB/s
1x	96x	669x = 3.0x/yr

*Random 4KB reads from user space

Need for Consistency Guarantees

- Applications demand consistency
  - Transaction processing
  - File systems
  - Web services / Cloud computing
- Failures are a reality
  - Power loss
  - Application/OS crash
  - Hardware faults
- Maintaining consistency is expensive
  - Requires data versioning



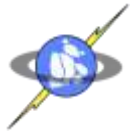
Designed with disk as the backing store!

Transaction Mechanisms

- Provide ACID semantics
  - Locking, logging, recovery
  - Requirements vary per application

- Key problem: maintaining versions of data
  - Databases: Write-ahead logging
  - File systems: Metadata journaling, shadow paging

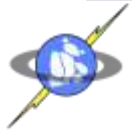
ARIES: A recovery algorithm for databases based on write-ahead logging



ARIES Features

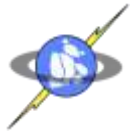
Designed to be fast, flexible, and scalable

Feature	Benefit(s)
Flexible storage management	Supports varying length data High concurrency
Fine-grained locking	High concurrency
Partial rollbacks via savepoints	Robust and efficient transactions
Recovery independence	Simple and robust recovery
Operation logging	High concurrency lock modes



ARIES Design Decisions

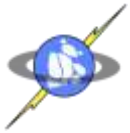
Design Decision	Advantages	How?
No-force	Eliminates synchronous random writes	Flush redo log entries to disk on commit
Steal	Reclaim buffer space More sequential writes Avoids false conflicts on pages	Write undo log entries before writing back dirty pages
Pages	Simplifies recovery	All updates are to pages Page writes are atomic
Log Sequence Numbers (LSNs)	Simplifies recovery Enables features like operation logging	LSNs provide an ordering on updates



Good for disk, not great for fast SSDs

Characteristics of Fast SSDs

- Random writes are just as cheap as sequential writes
- Flexible interface
  - No page/sector restriction on request size
- Lots of parallelism
  - Multiple memory controllers (MCs)
  - Sophisticated scheduler to handle many requests
- Bandwidth mismatch
  - Aggregate BW of MCs >> interconnect BW

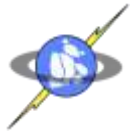


Moneta-TX: Support for Atomic Writes

- Combines multiple writes together in a group
  - Sequential or scattered
  - No fixed block size (byte-addressable)

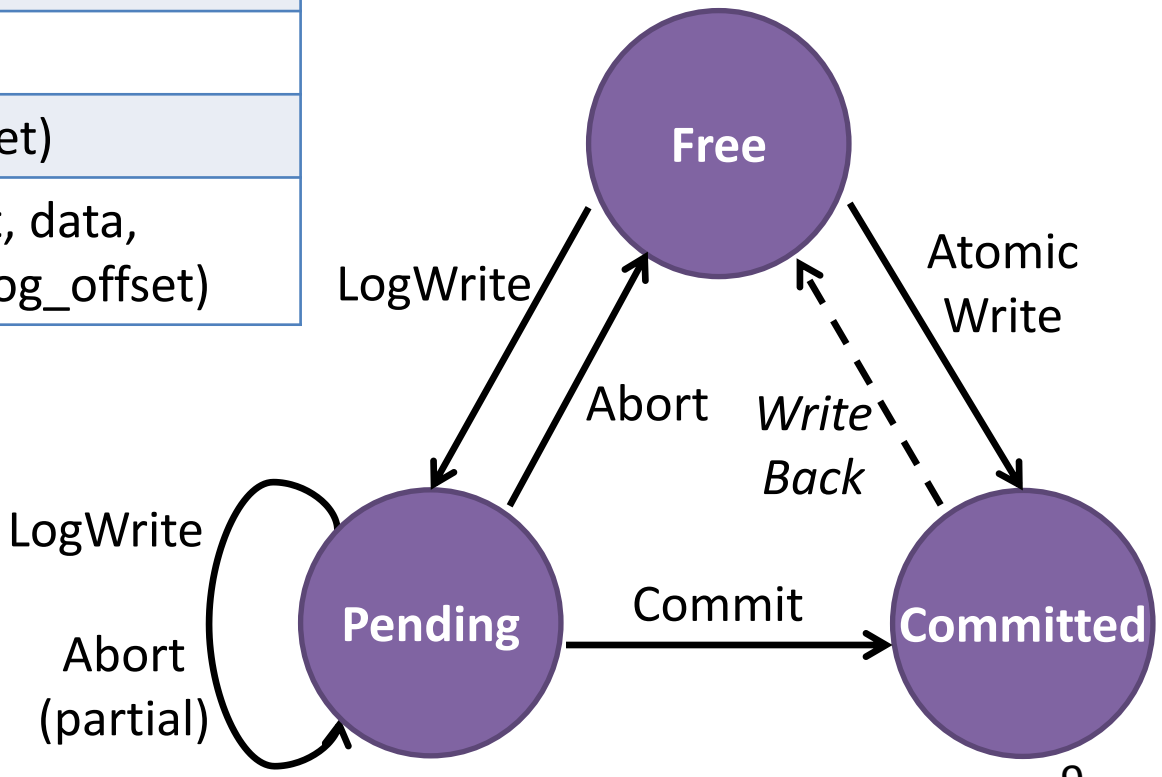
- Atomicity and durability
  - Building block for full ACID transactions
  - Consistency and isolation left to the programmer

- Logs are *visible* to and *managed* by the application
  - Transactions can see their own updates
  - Transactions scale with the amount of free space

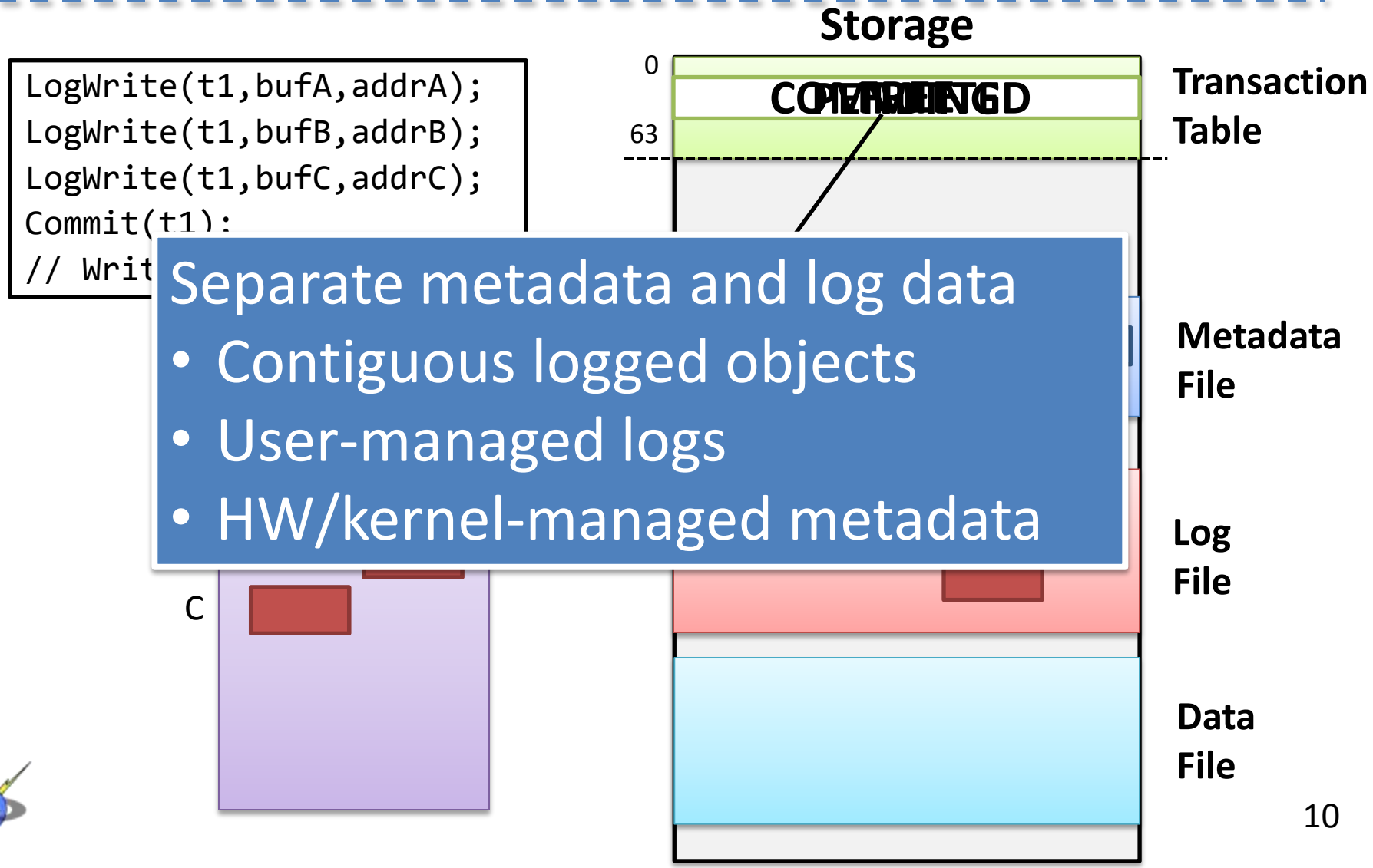


Software Interface

Commands
LogWrite(tid, file, offset, data, len, log_file, log_offset)
Commit(tid)
Abort(tid, log_file, log_offset)
AtomicWrite(tid, file, offset, data, len, log_file, log_offset)

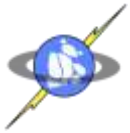


Atomic Write Execution

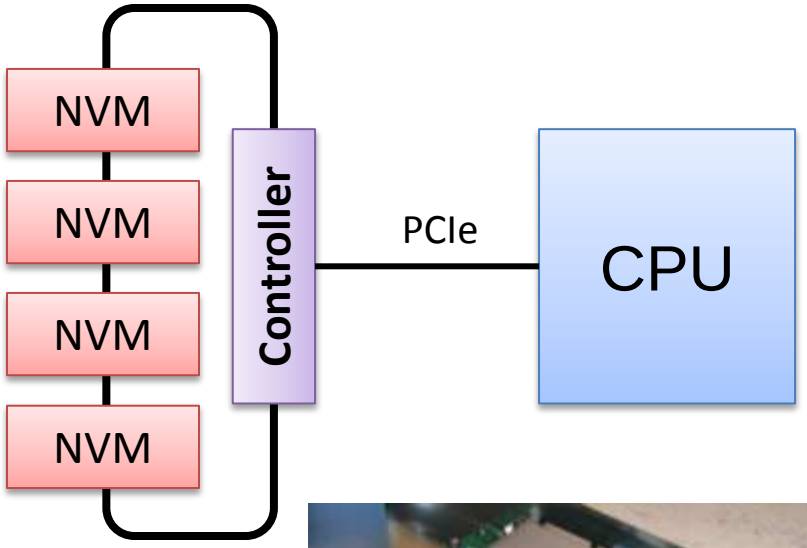


Rethinking ARIES for Moneta-TX

No-force →	Force policy in hardware at the memory controllers
Steal →	Hardware does in-place updates Eliminate undo logging Log always holds latest copy
Pages →	Hardware uses pages internally for the logged objects (striped) Software sees contiguous objects
LSNs →	Hardware maintains ordering with commit sequence numbers



Moneta: An SSD for Fast NVMs

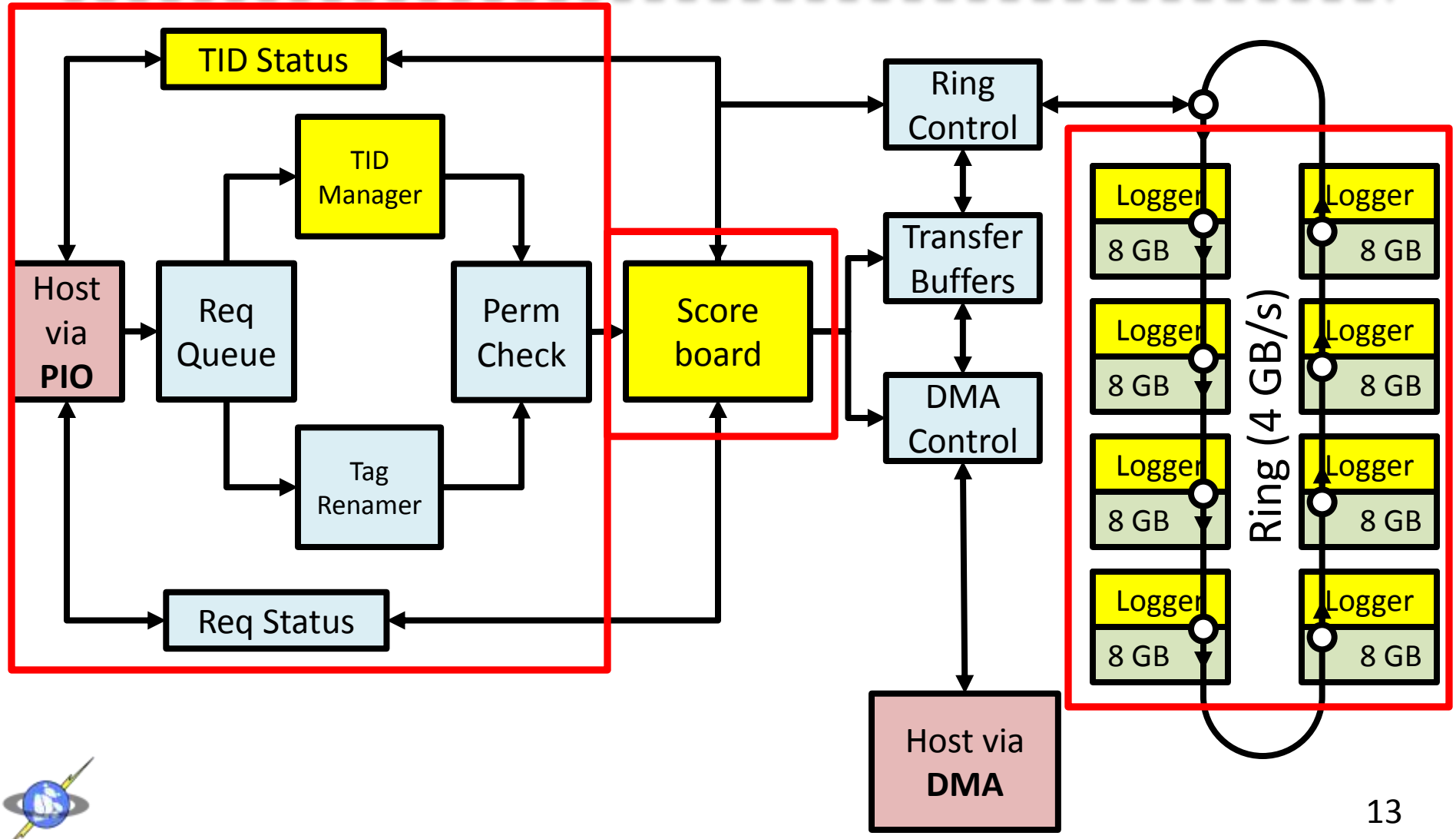


[SC 2010, MICRO 2010, ASPLOS 2012]

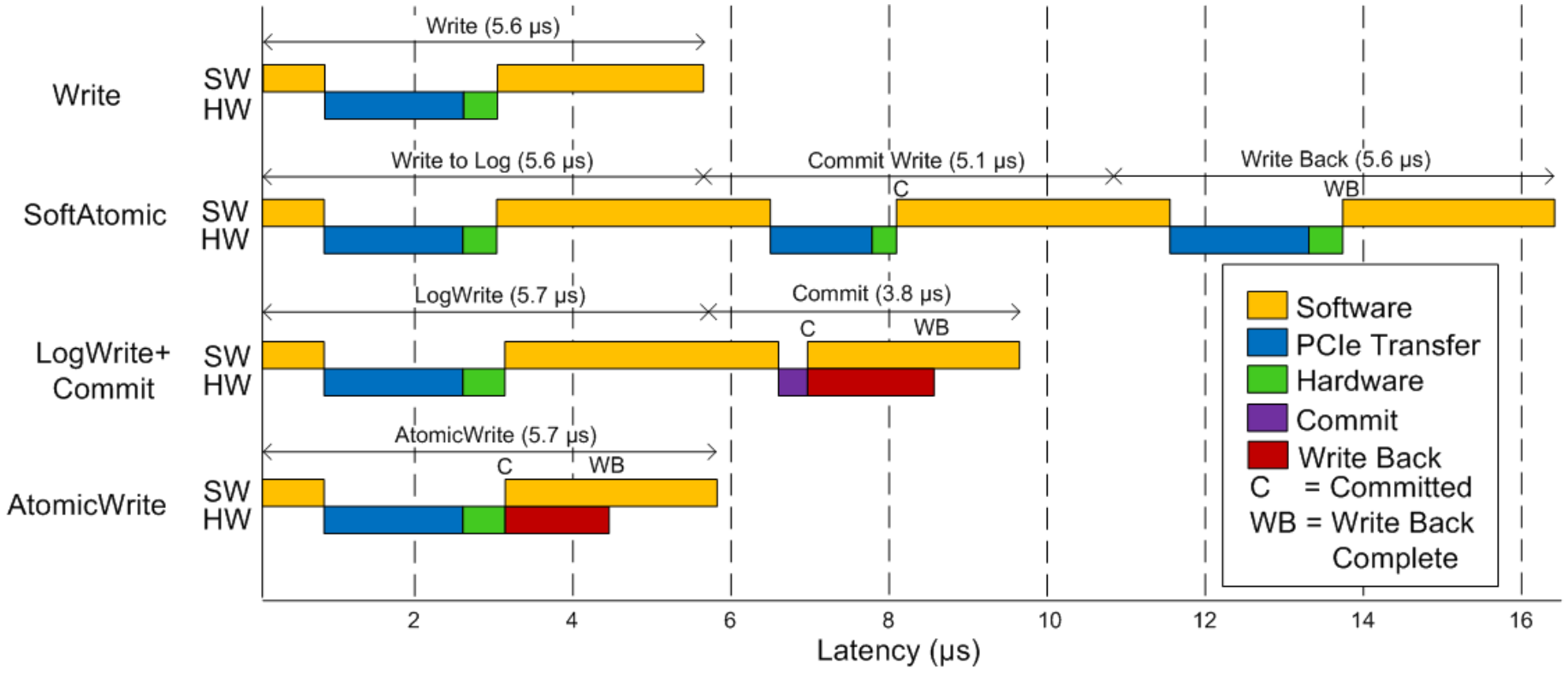
- FPGA-based prototype
  - DDR2 DRAM emulates PCM
  - PCIe: 2GB/s, full duplex
- Optimized driver and hw/sw interface
  - Eliminate disk-based bottlenecks in IO stack
- User-space driver
  - Eliminates OS and FS costs in common case

5µs latency, 1.8M IOPS for 512 B requests

Moneta-TX Hardware Architecture

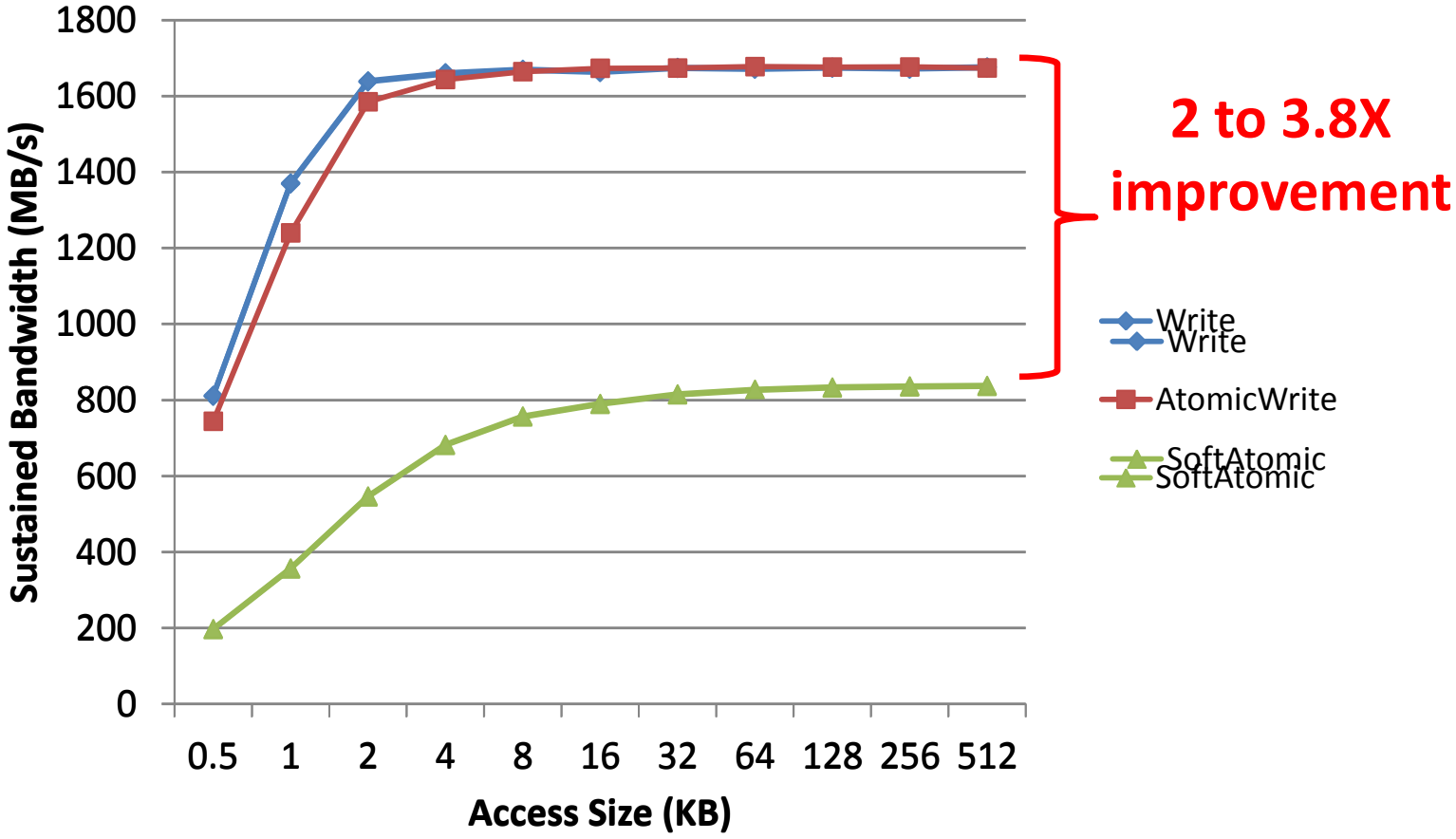


Latency Breakdown

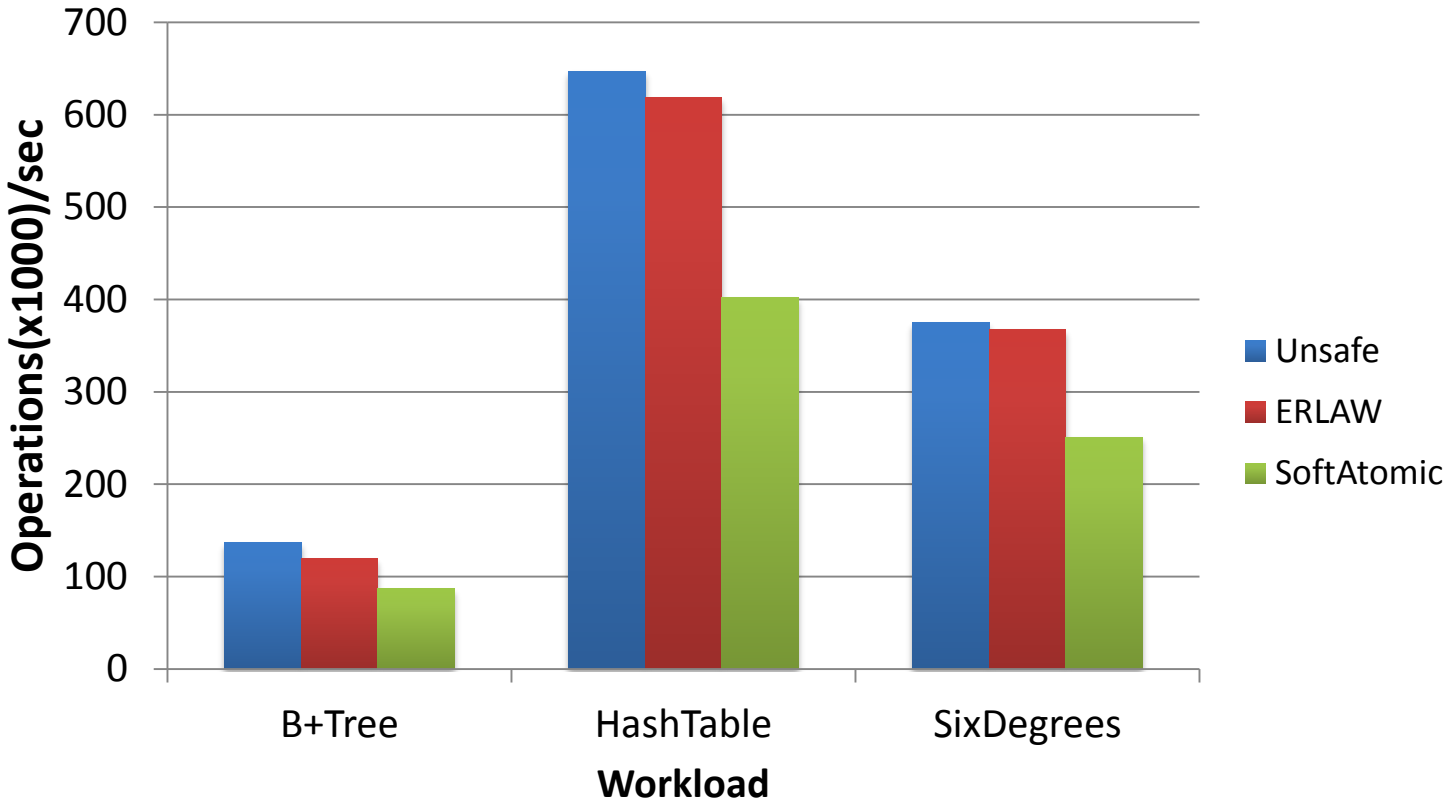


2X faster than SoftAtomic

Bandwidth Comparison

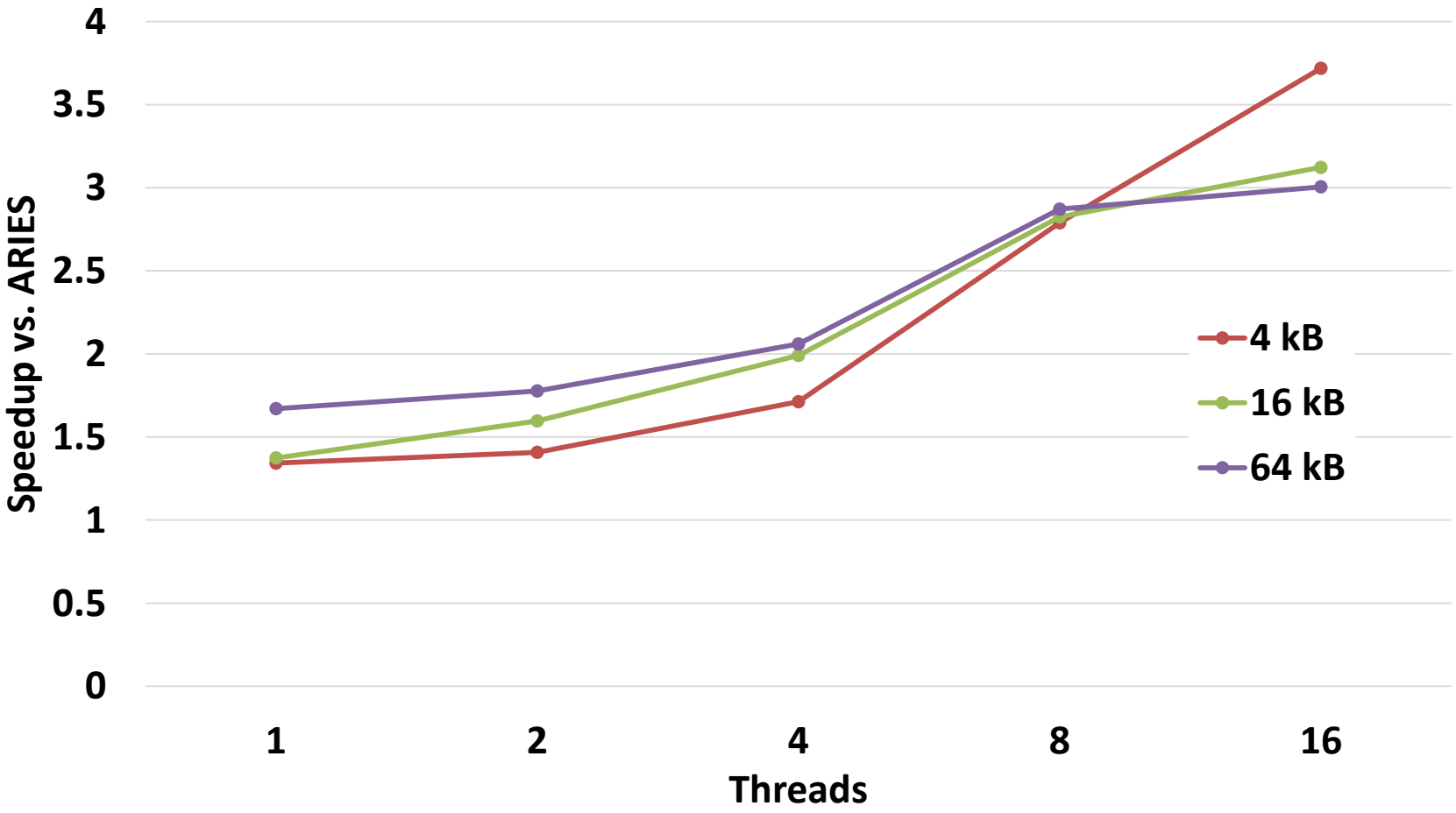


Persistent Data Structures



Within 2 to 16% of the performance of Unsafe

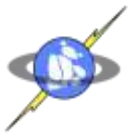
Replacing ARIES No-Force/Steal with Moneta-TX Atomic Writes



4X throughput improvement

Conclusion

- Hardware support makes atomicity and durability almost free
  - Exploits parallelism, low-latency, bandwidth
- Exposing the logs makes the system flexible
  - Possible to implement ARIES high-level features
- Big gains in redesigning our applications for a transactional storage interface



Thank You!

NVSL
Non-volatile Systems Laboratory

UCSDCSE
Computer Science and Engineering

Thanks to everyone who worked on Moneta:
Adrian Caulfield, Todor Mollov, Arup De, Alex Eisner, Ameen Akel