Time to Rethink (Almost) Everything: Scalable Storage Systems for Fast Non-Volatile Memories

Steven Swanson

Non-Volatile Systems Laboratory

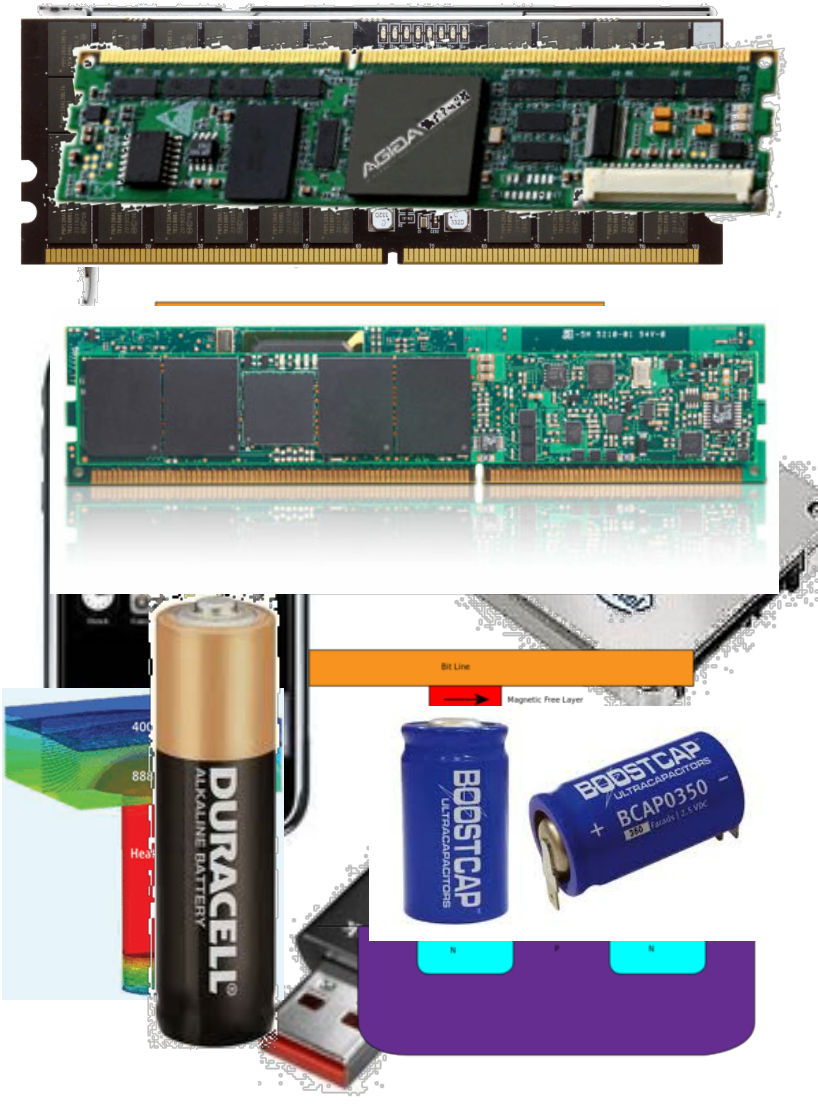
Computer Science and Engineering

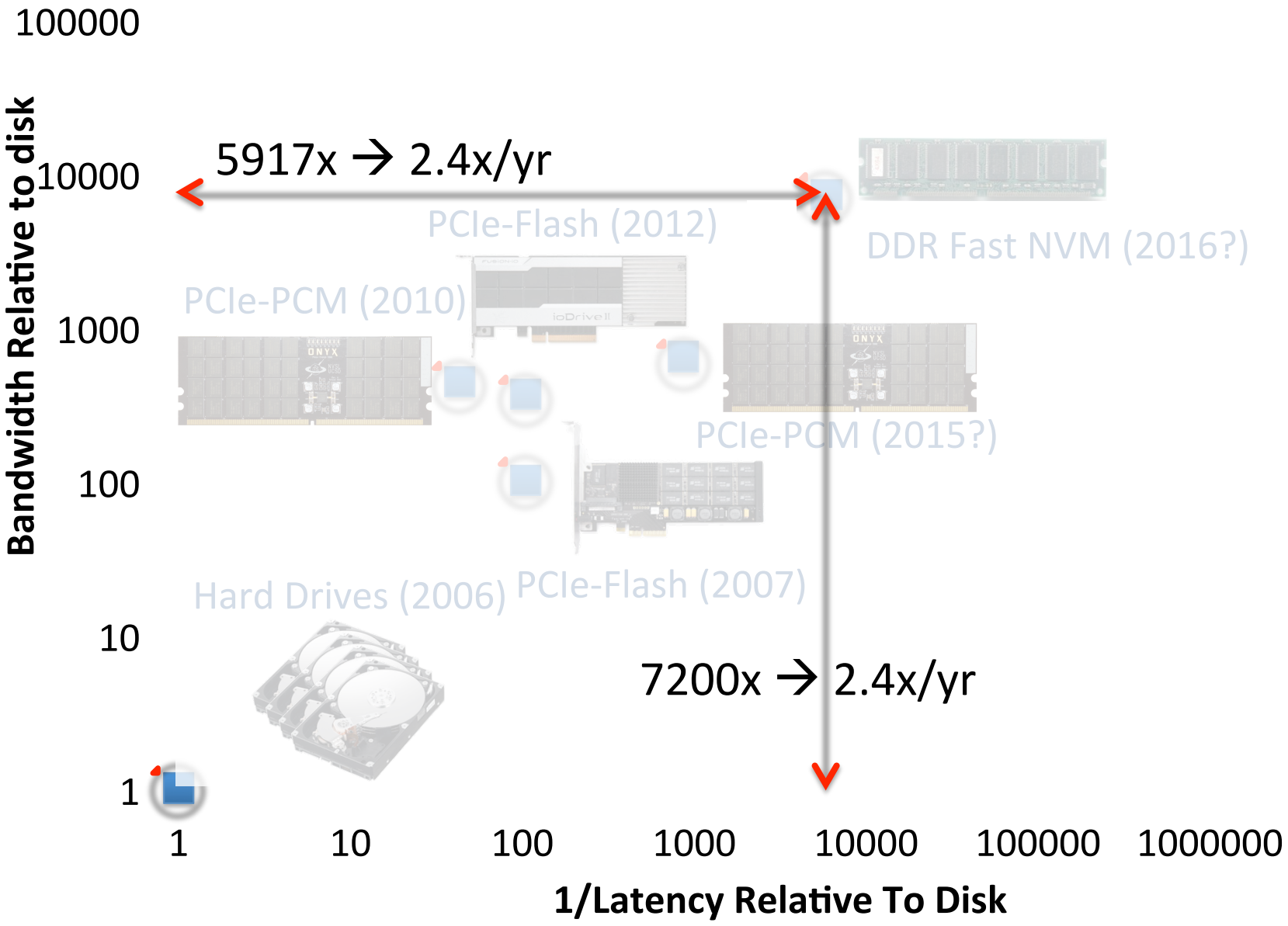
University of California, San Diego

UC San Diego

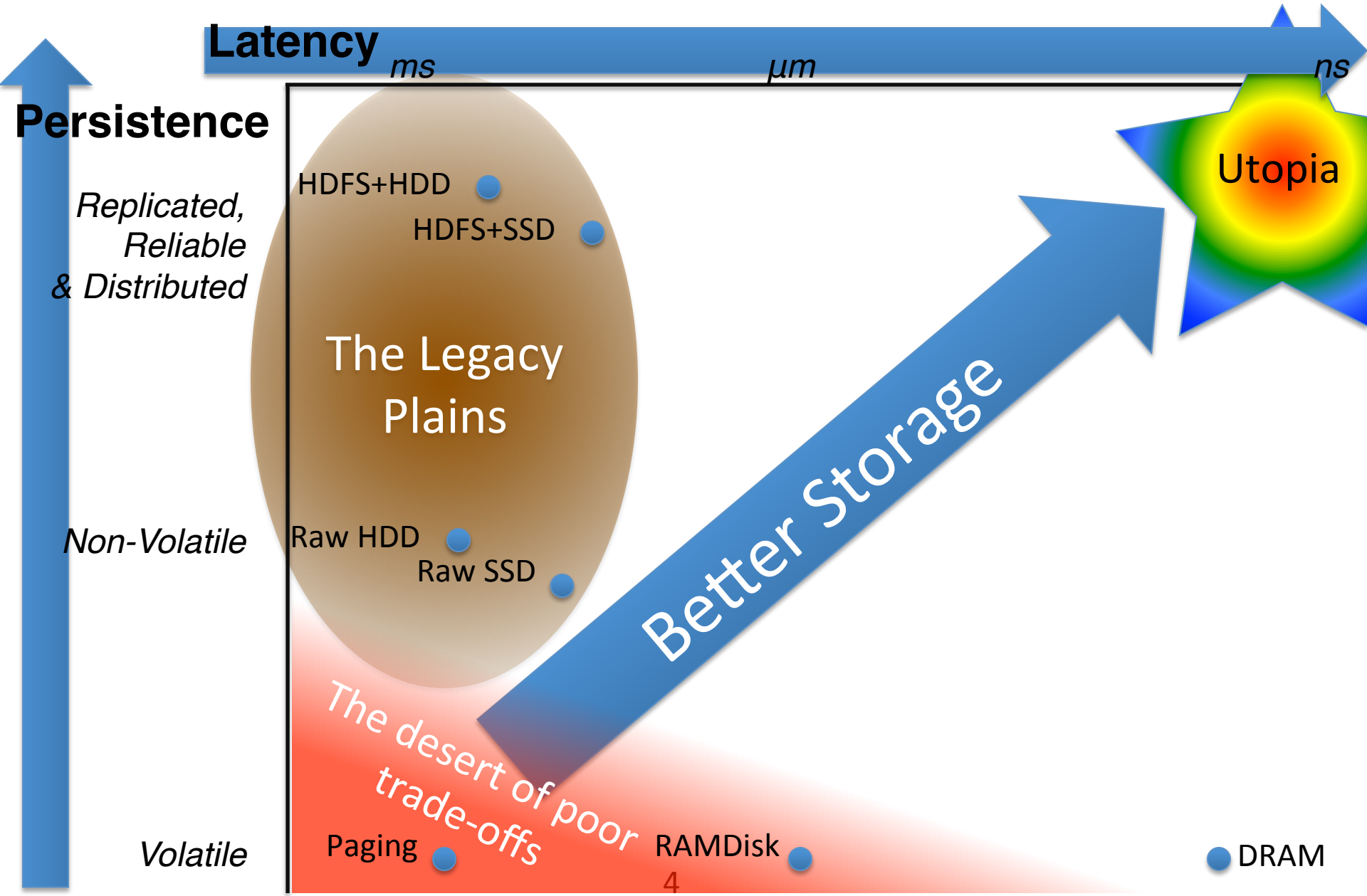
Solid State Memories

- NAND flash
  - Ubiquitous, cheap
  - Sort of slow, idiosyncratic
- Phase change, Spin torque MRAMs, etc.
  - Not (yet) ready for prime time
  - DRAM-like speed
  - DRAM or flash-like density
- DRAM + Battery + flash
  - Fast, available today
  - Reliability?

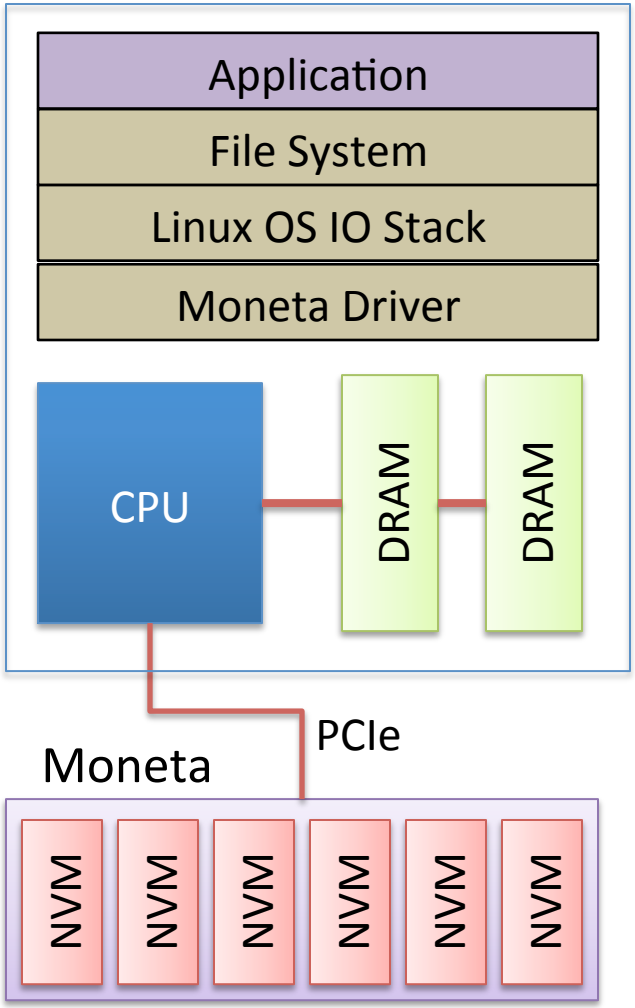



100000
10000
1000
100
10
1
Bandwidth Relative to disk
5917x → 2.4x/yr
PCIe-Flash (2012)
DDR Fast NVM (2016?)
PCIe-PCM (2010)
PCIe-PCM (2015?)
PCIe-Flash (2007)
Hard Drives (2006)
7200x → 2.4x/yr
1
10
100
1000
10000
100000
1000000
1/Latency Relative To Disk

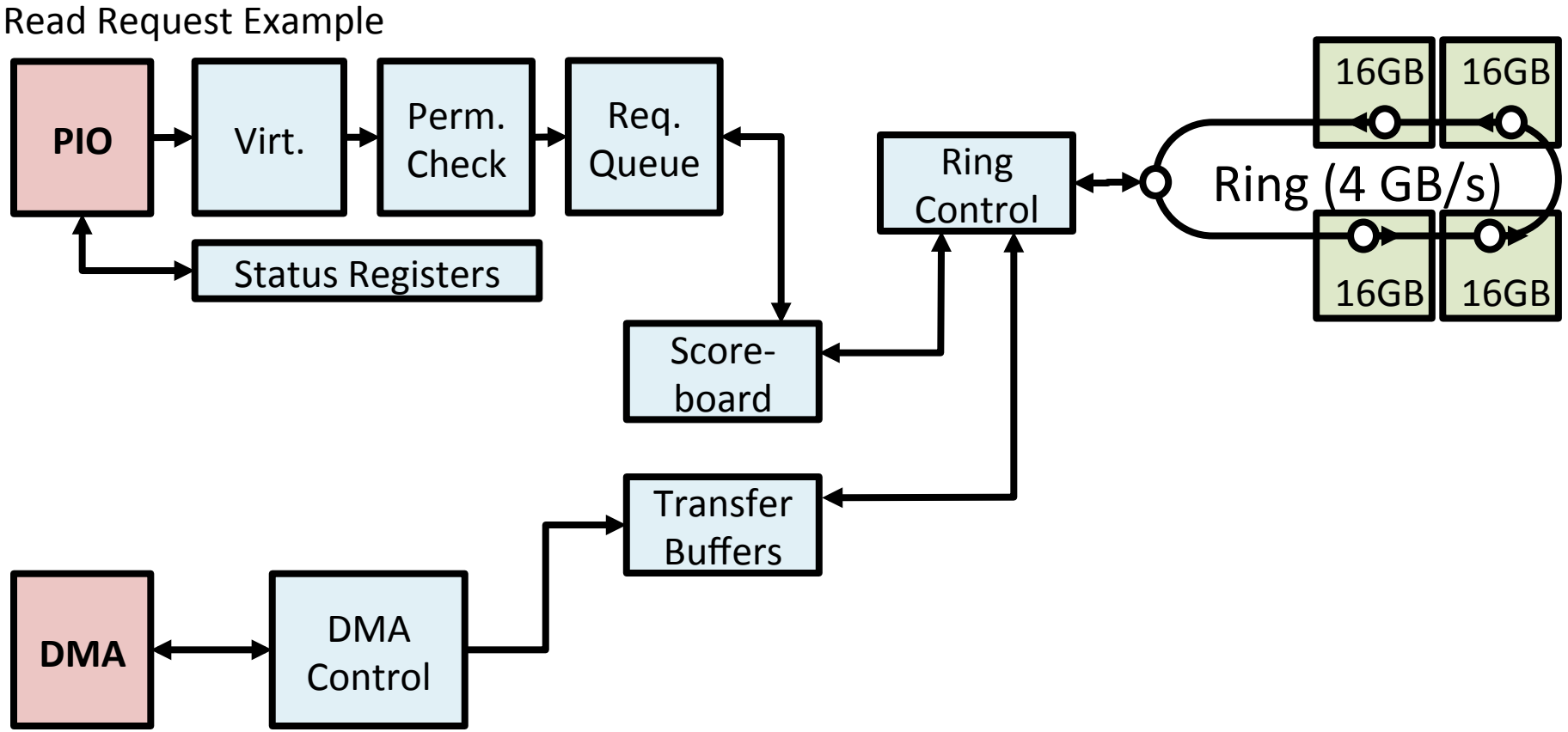
The Storage Landscape



Moneta:
SSD Architecture for Advanced NVMs

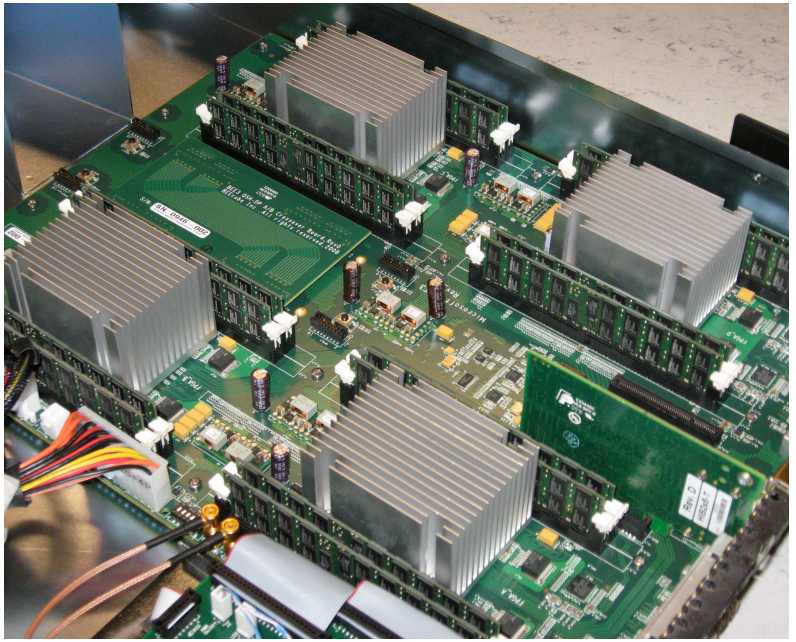


Moneta:
SSD Architecture for Advanced NVMs



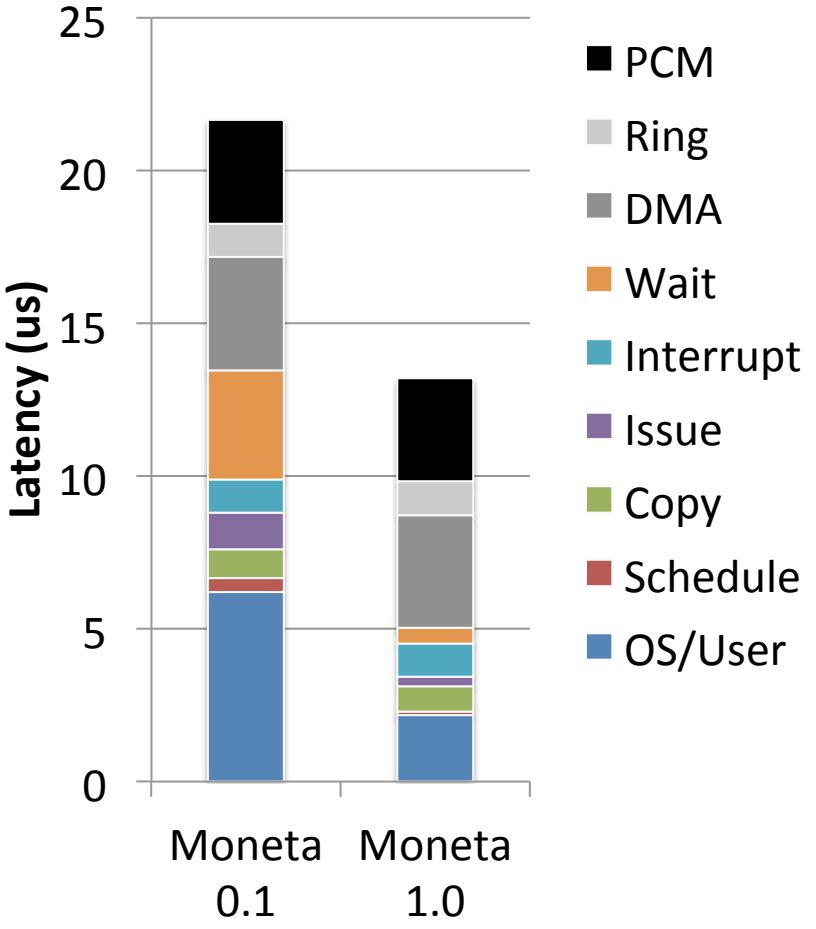
The Moneta Prototype

- FPGA-based implementation
- DDR2 DRAM emulates PCM
  - Configurable memory latency
  - 48 ns reads, 150 ns writes
  - 64GB across 8 controllers
- PCIe: 2 GB/s, full duplex

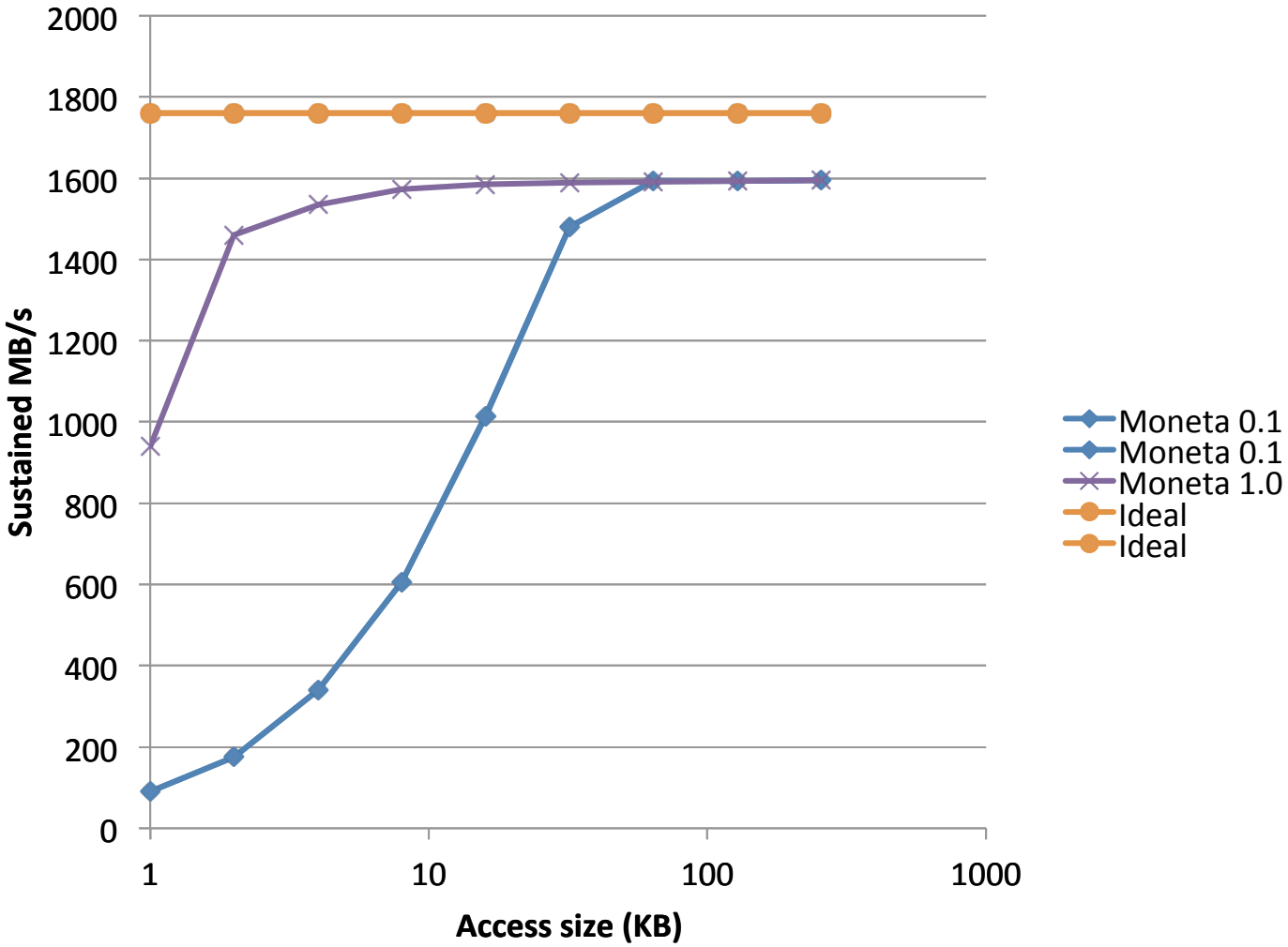


Reducing Software Overheads in Moneta

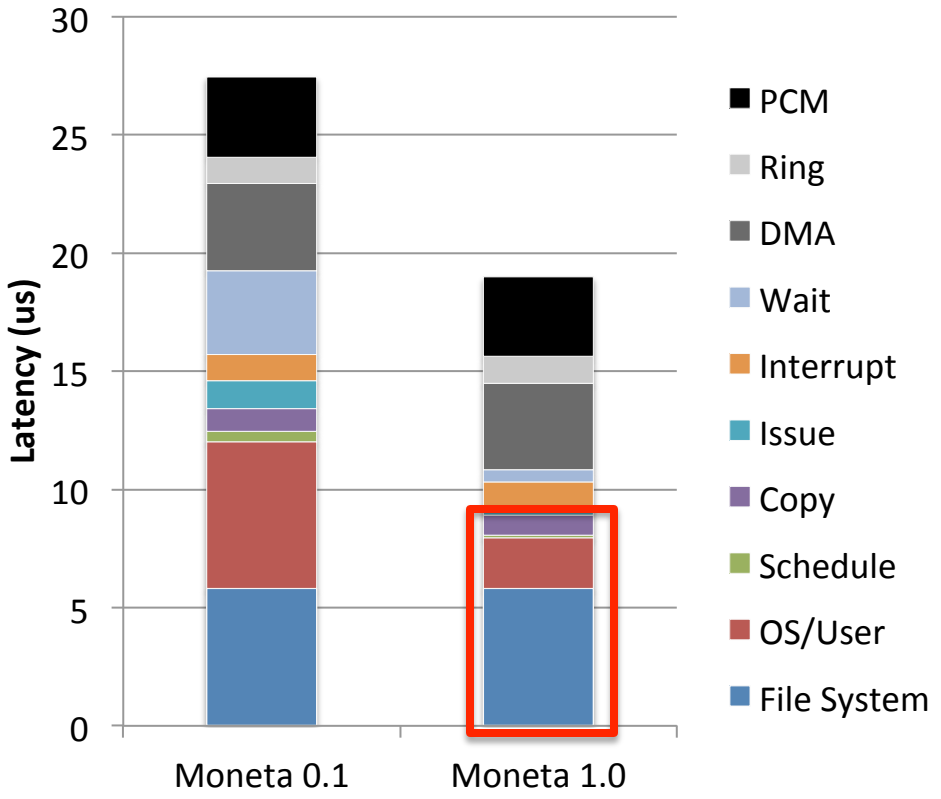
- Optimizations
  - Remove IO scheduler
  - Redesigned HW interface
  - Remove locks
- SW latency drops from 13.4 us to 5us
  - 62% reduction in latency
  - Increased concurrency
- Bandwidth increases by up to 10x

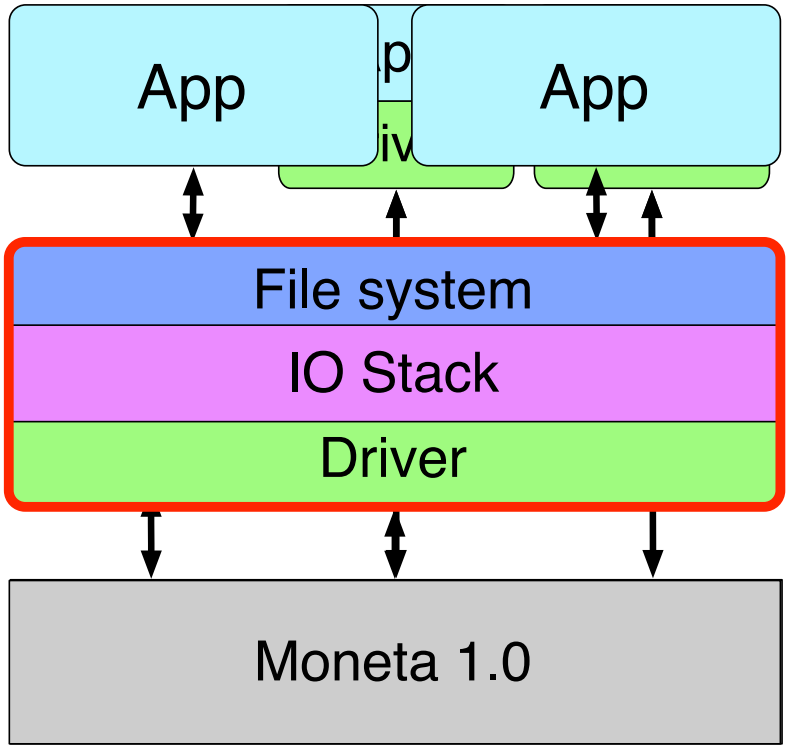


Bandwidth Impact of SW Reduction



Eliminating FS and OS overheads



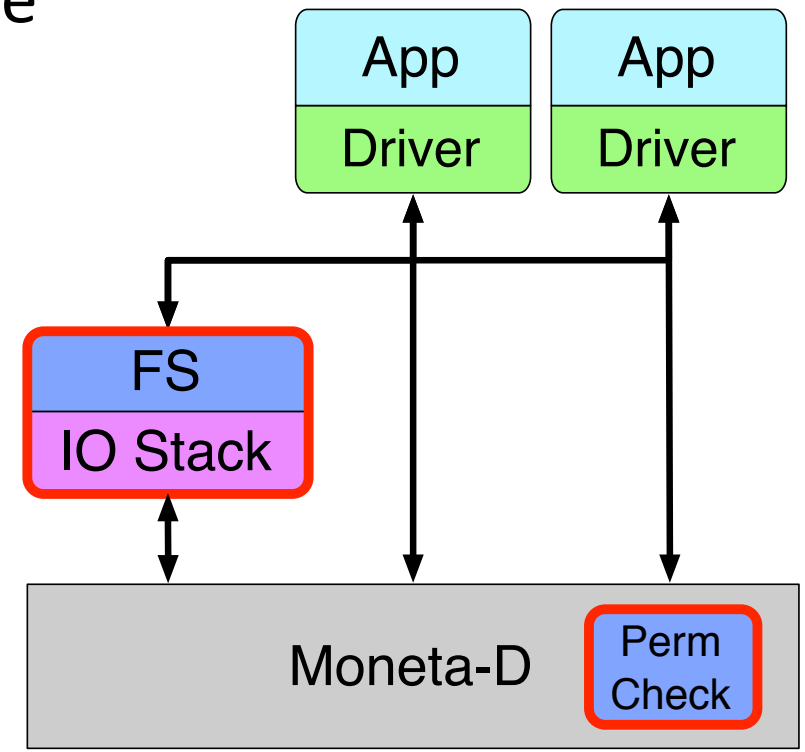


- 4.1% is to support protection and permissions
  - Move protection checks to hardware
  - Allow applications to access Moneta directly

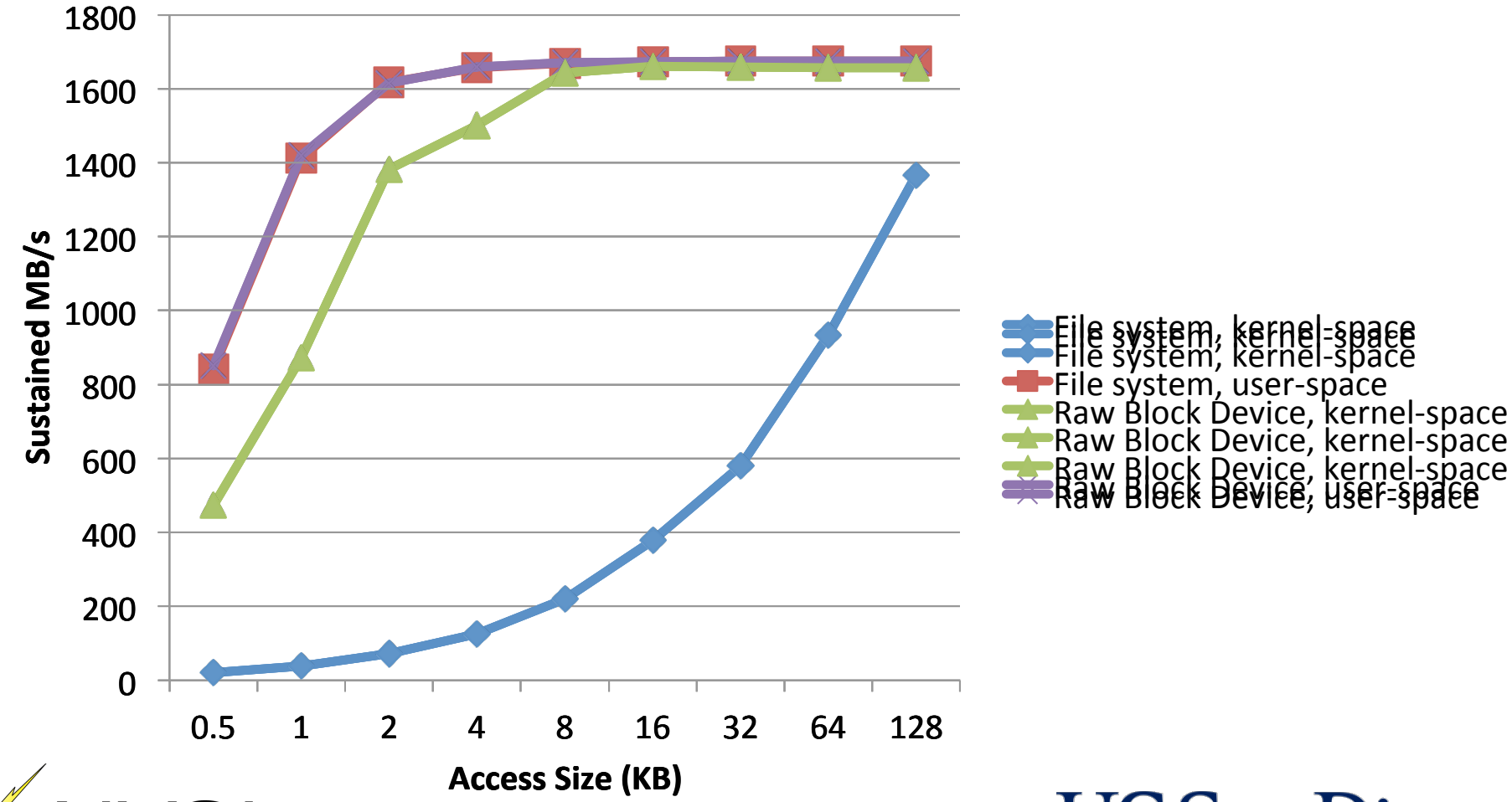
4.1% is to support protection and permissions

Removing Protection Overheads

1. Virtualized Moneta interface
2. User space library
3. Protection enforcement
4. OS/FS/App Implications



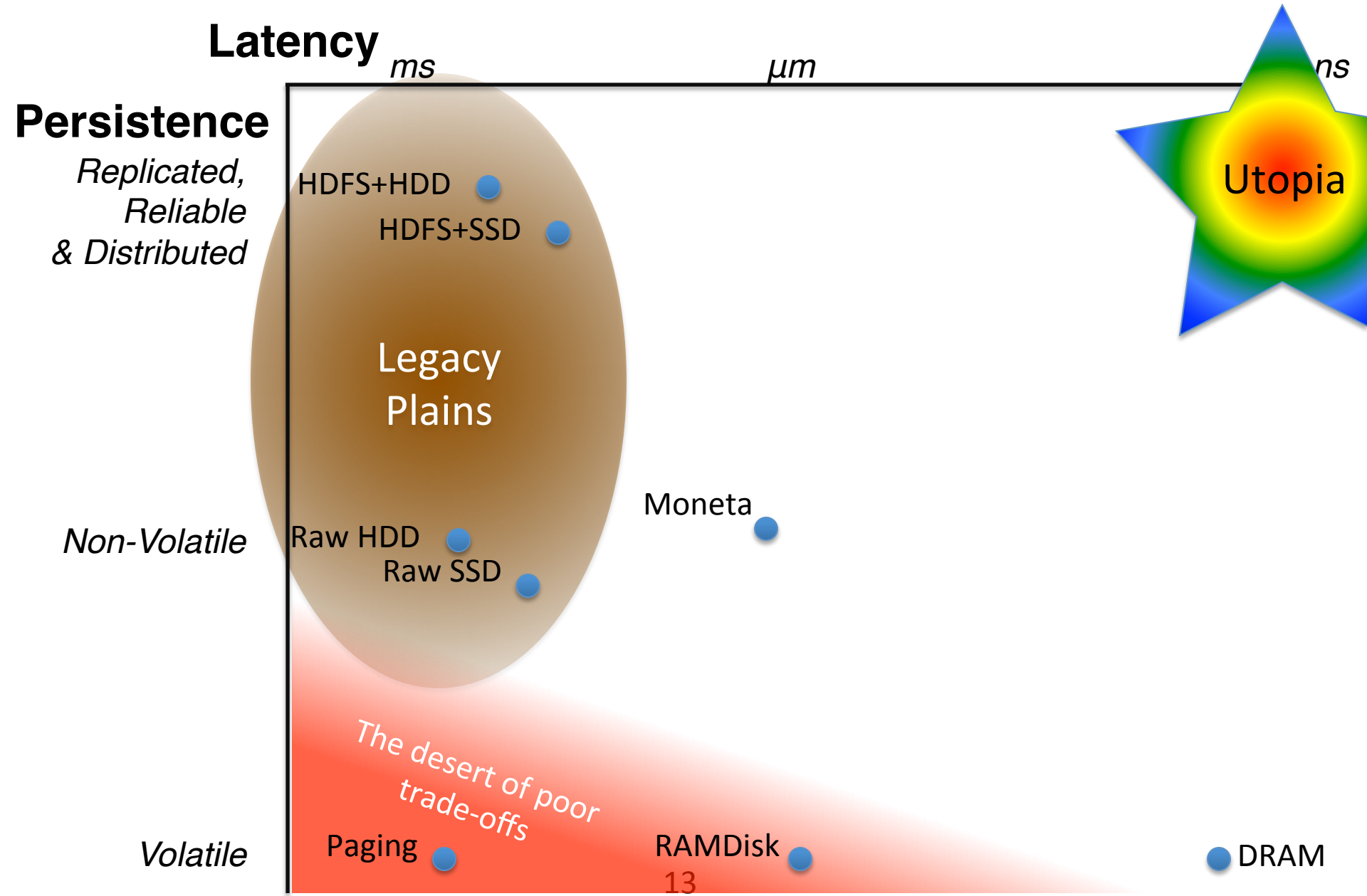
Direct-access Bandwidth Improvements



NVSL
Non-volatile Systems Laboratory

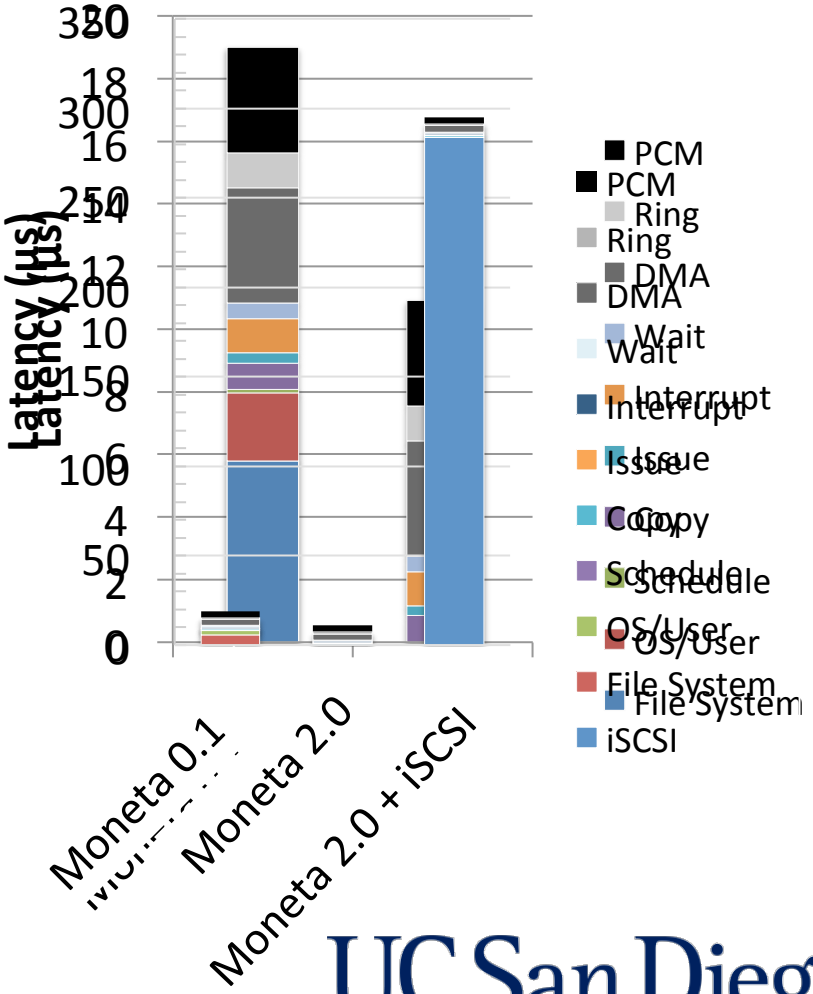
UC San Diego

The Storage Landscape



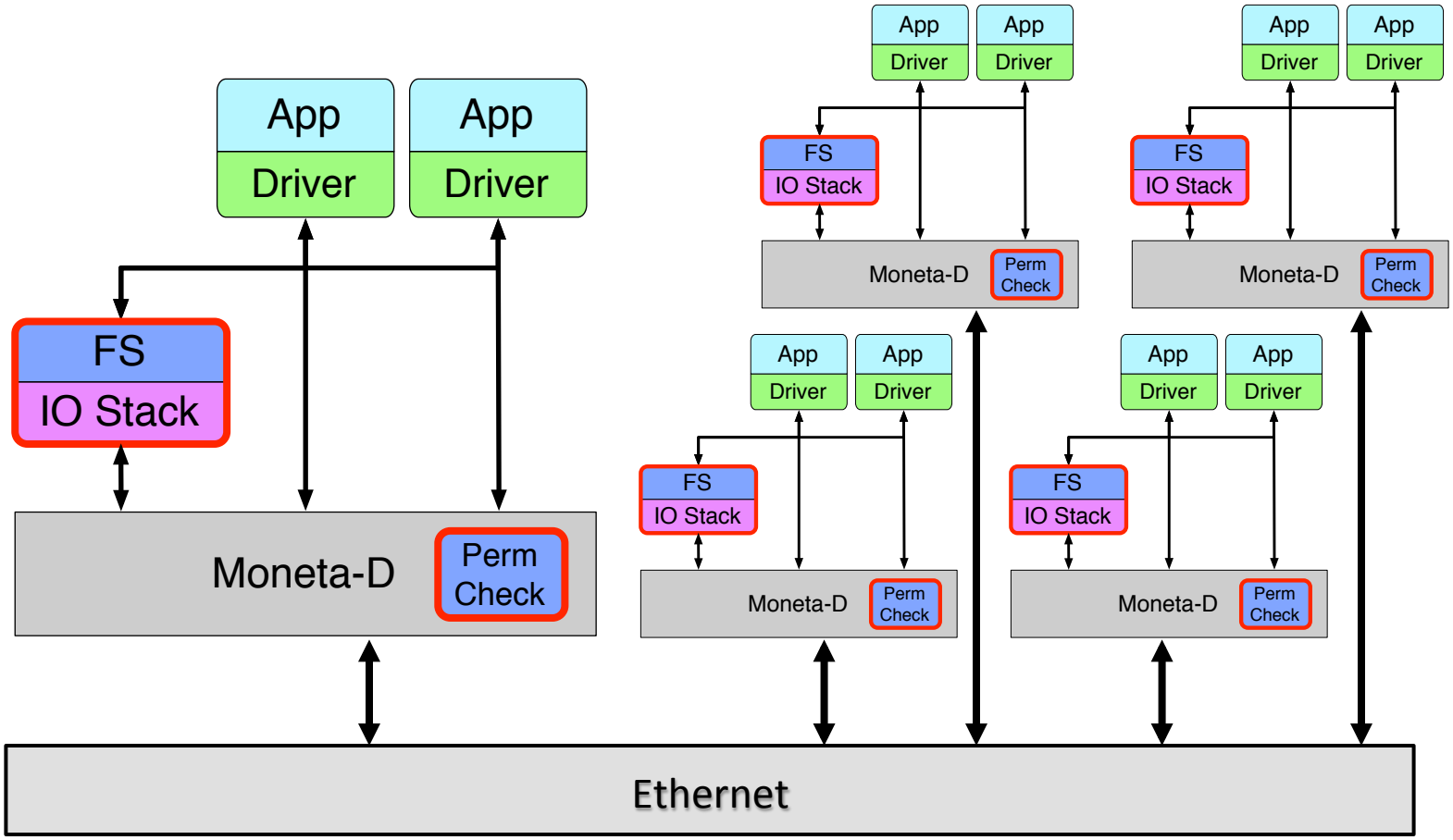
QuickSAN: Moneta + Network

- To scale Moneta, we must address network costs
- Block transport costs are enormous
  - Think software layers
  - Complex file systems

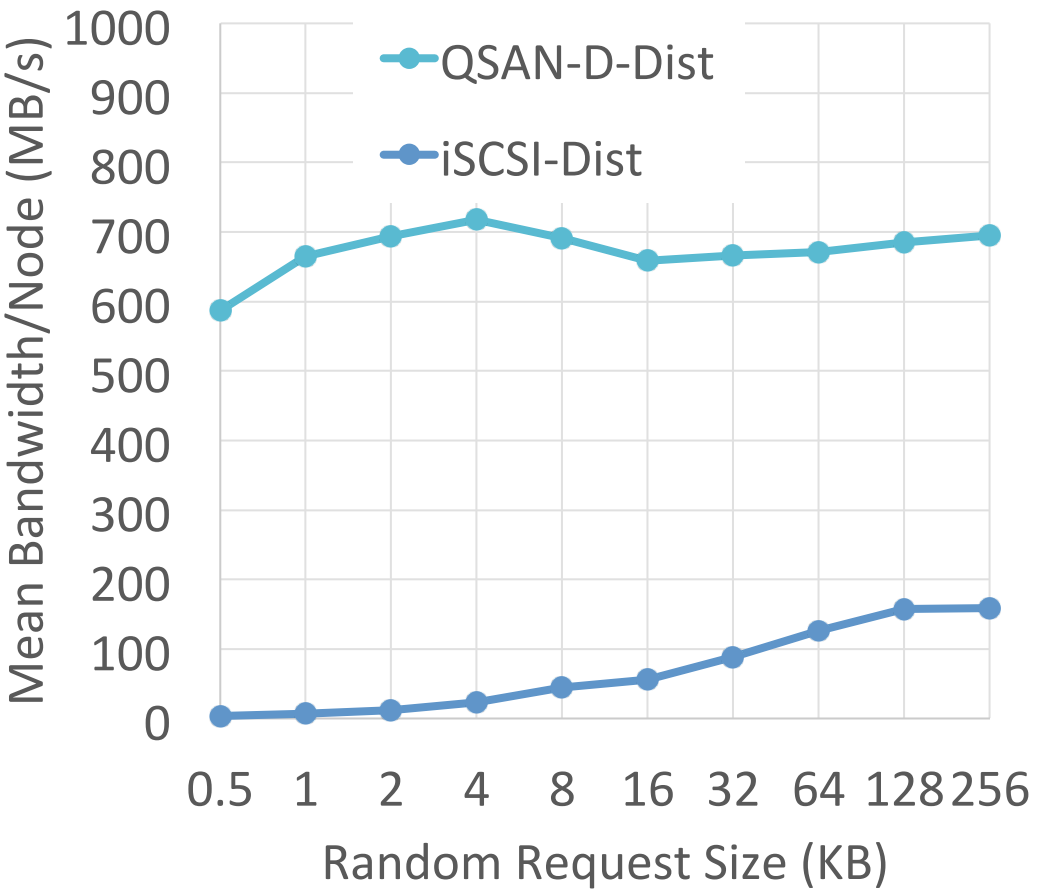


Distributed, Direct-Access Storage

- Storage Distributed throughout SAN on each client
- Allows applications to exploit locality of data



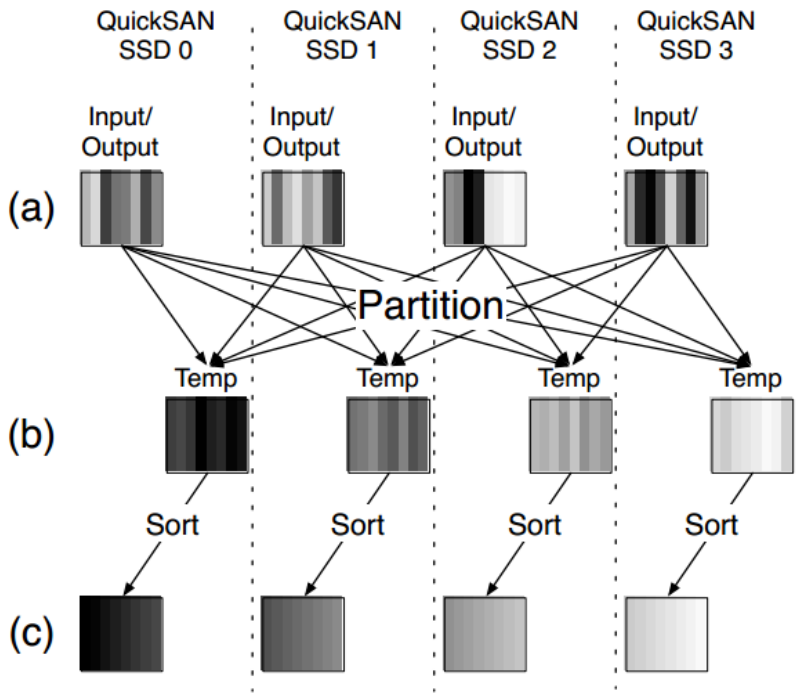
QuickSAN Performance: Writes



- vs. iSCSI
  - 195x increase in bandwidth
  - 16x reduction in latency
- vs. FibreChannel
  - Est. 5x reduction in latency
- Still ~17us longer than Moneta-D

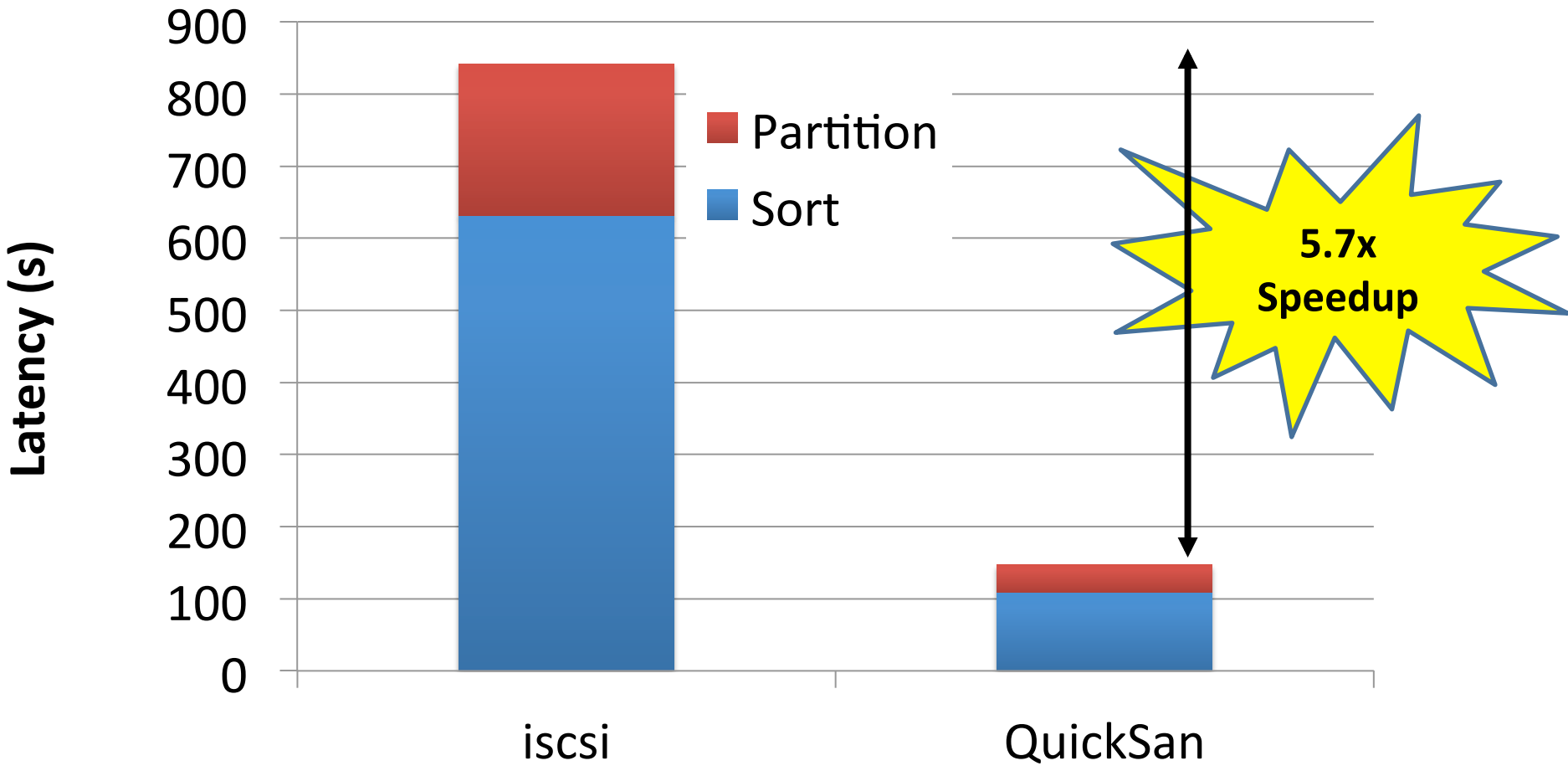
Workload: Distributed Sort

- Dist. external sort
  - Used by MapReduce, others
  - Based on TritonSort
- Two phases:
  - Partition: sort keys into groups on all nodes
  - Sort: sort groups locally on each node
- Leverage non-uniform access in QuickSAN

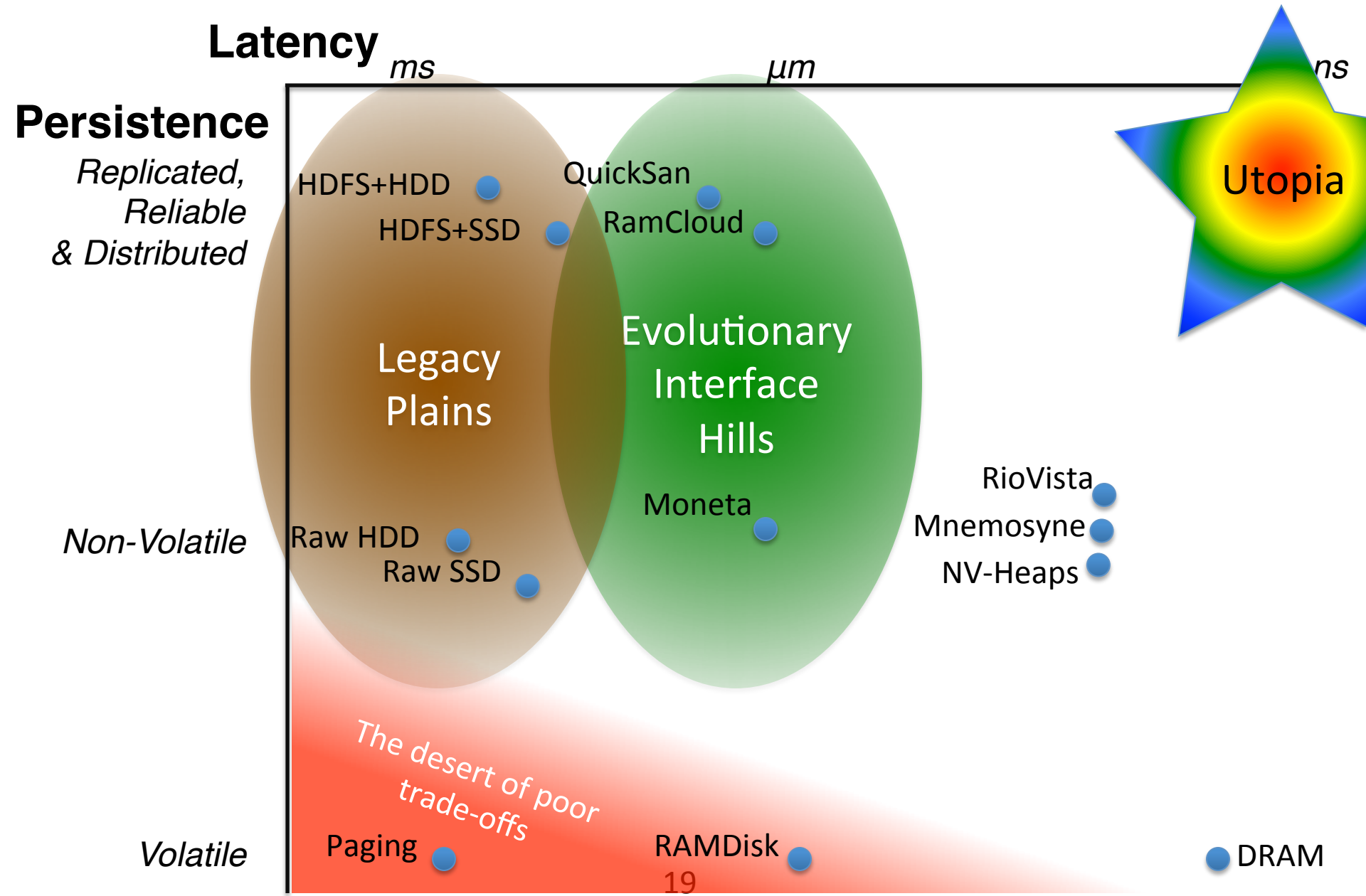


Sort Performance

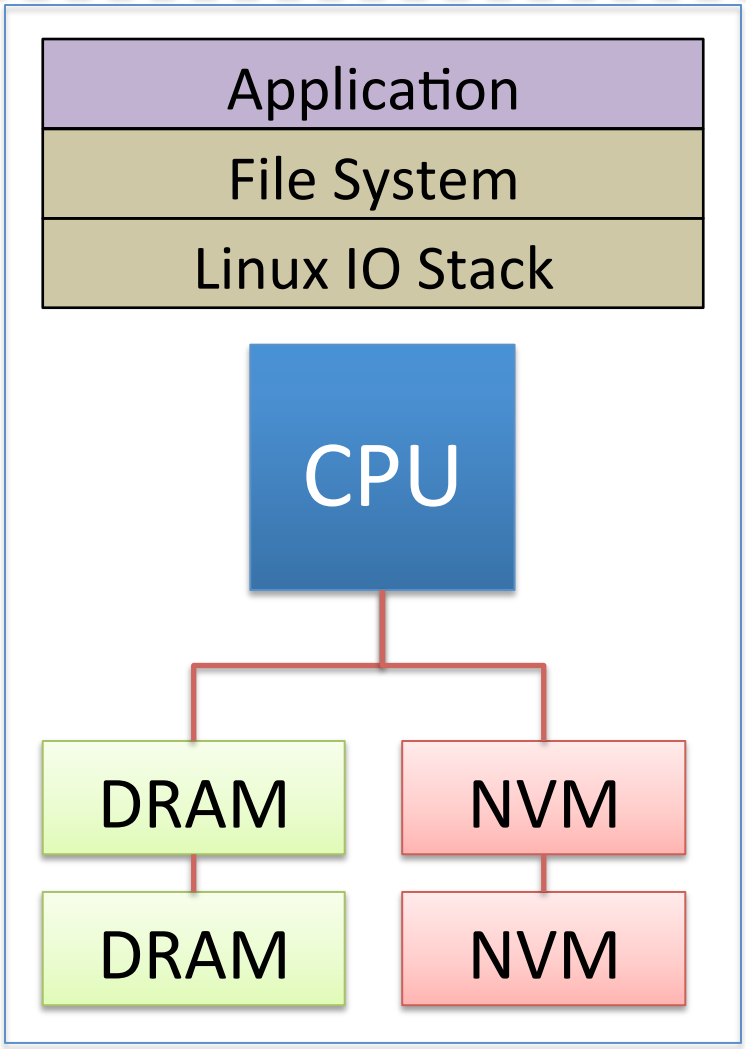
102GB Sorted



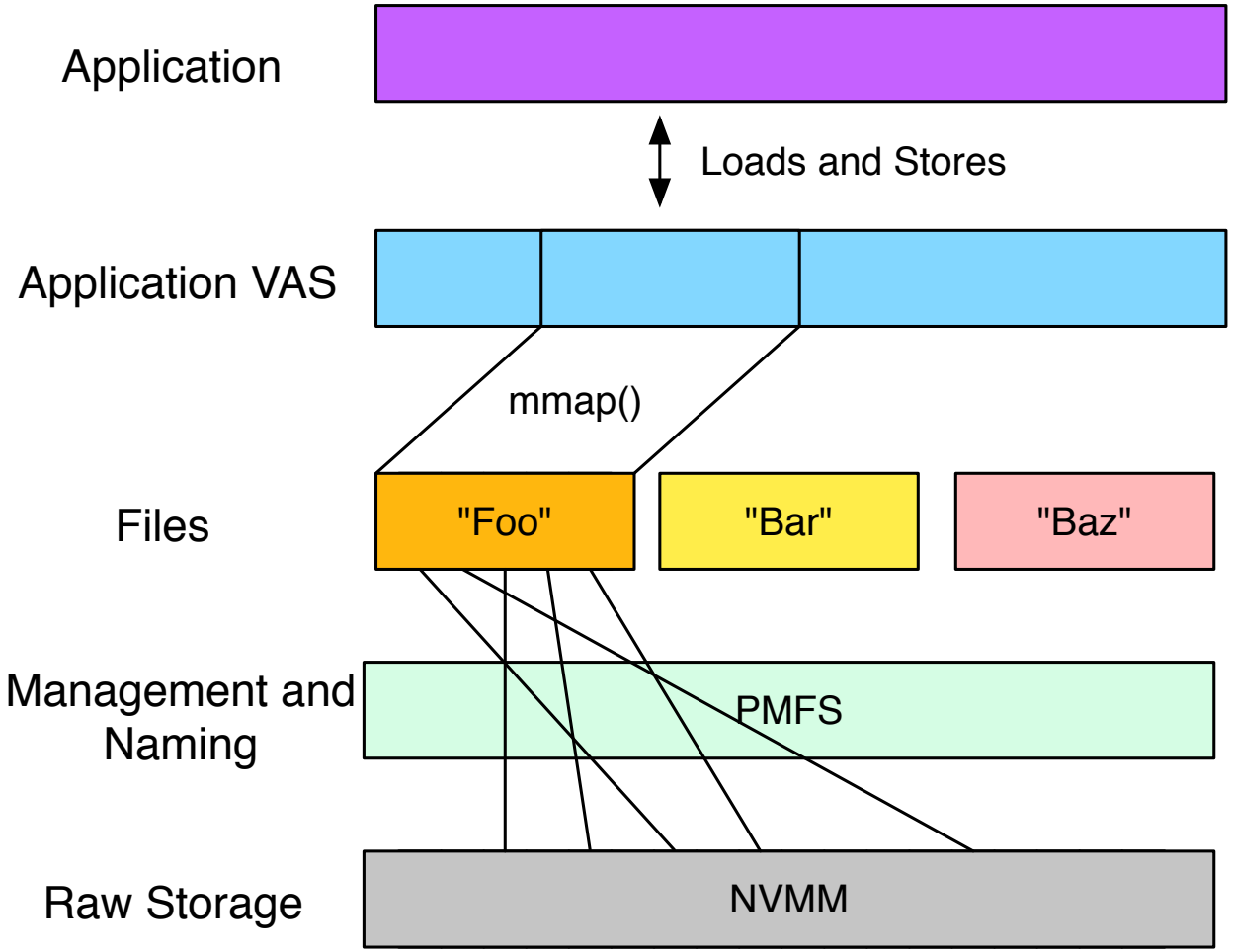
The Storage Landscape



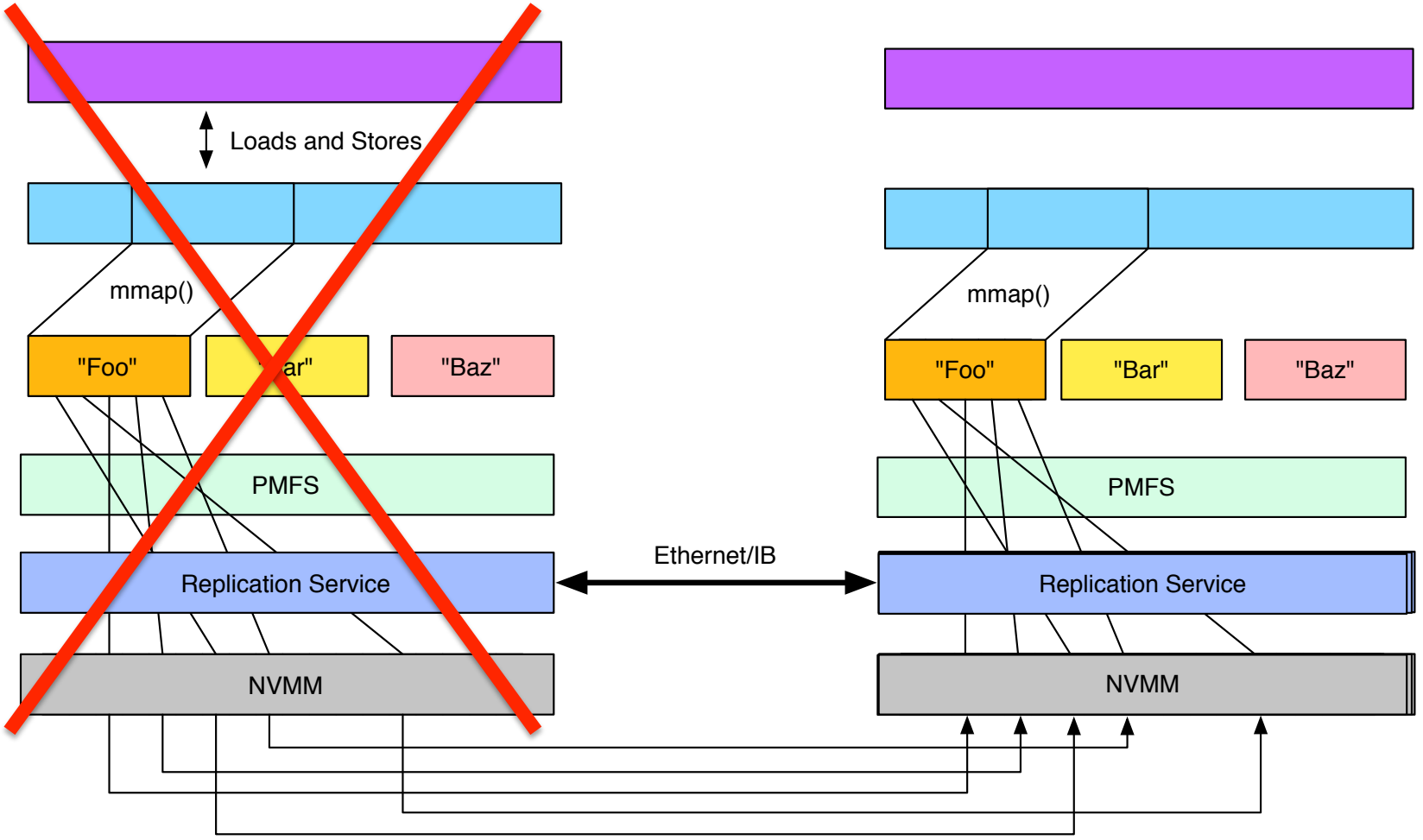
Direct-attached NVMs



System Usage Model

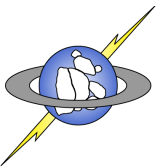


Mojim: Replicated NVM Main Memory



Mojim: Replicated NVM Main Memory (NVMM)

- Kernel service provides replicated address space regions
  - File systems
  - NVMalloc
- Extended msync() provide atomic group commit of updates.
- Configurable consistency-performance trade-offs
  - Async: Better performance, weak consistency
  - Sync: Stronger consistency, increased latency

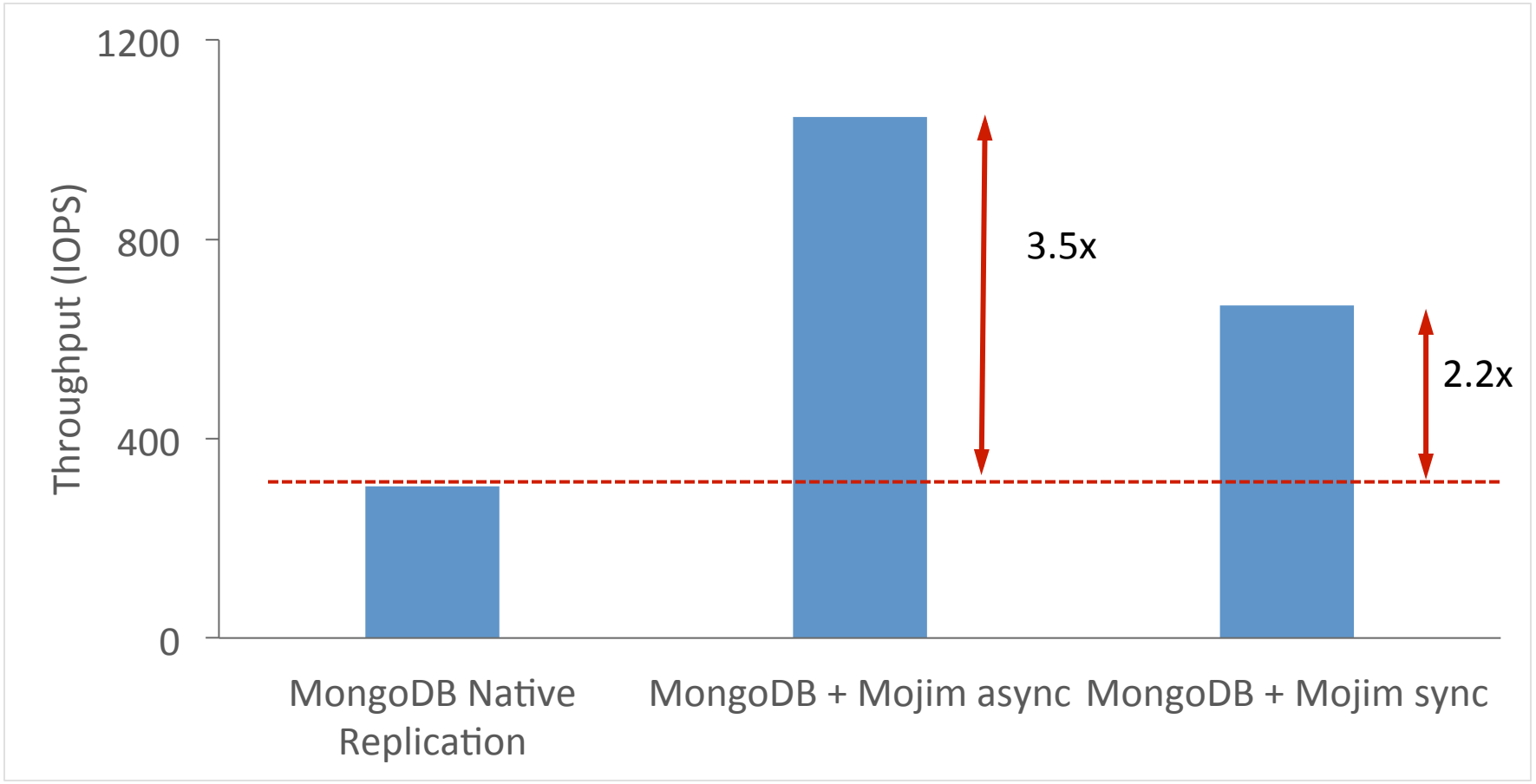


Use Case – Key-Value Store

- MongoDB
  - (mmapped) data store and journal
  - Async / sync replication across nodes
- With Mojim
  - Replaced memory-mapped, paged files with direct-mapped NVMM
  - Leverage Mojim atomic group commit for replication and consistency

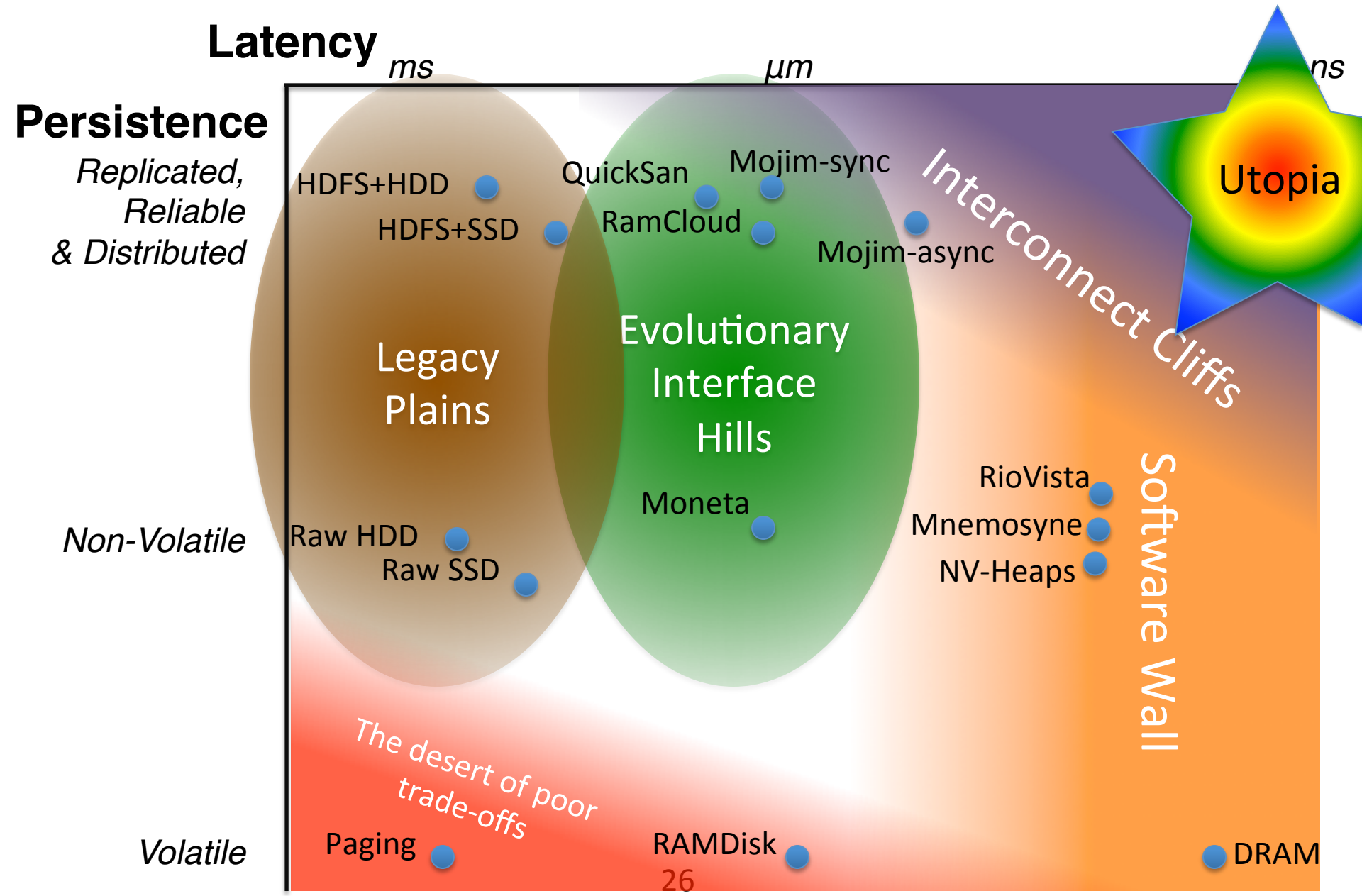
MongoDB Throughput

- Workload: random key-value pair inserts



25

The Storage Landscape



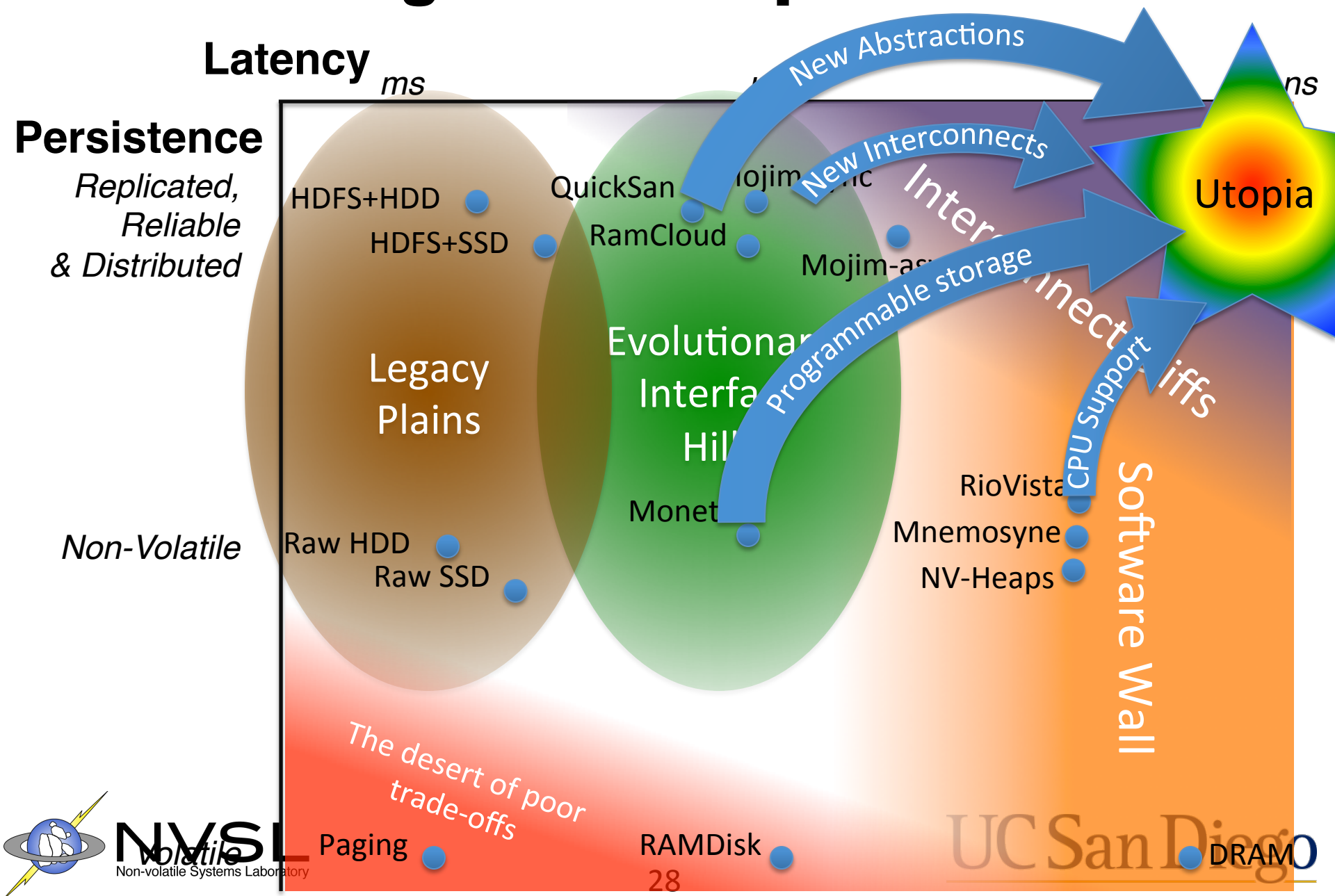
Reaching Utopia

- Move apps to the data
- New storage/memory abstractions
- Processor support
- New interconnects

- *Embed computation close to memory*
- *Low-latency*
- *High Bandwidth*
- *Energy efficiency*
- *Direct CPU*
- *Reliable execution*
- *Atomicity*
- *Low-latency transport*
- *(non-)volatility*
- *Silicon photonics*

- *File system-like management*
- *Memory-like access*
- *Language support*
- *Integration*
- *Hardware support*

The Storage Landscape



Thanks!