diablo technologies™

PROBLEM SOLVED.

Memory Channel Storage™

Maher Amer
CTO Diablo Technologies

diablo technologies™ | PROBLEM SOLVED.

DIABLO TECHNOLOGIES **HIGHLIGHTS**

Invented MCS™: New system architecture for non-volatile memory

Strong Financial Backing: $36M from Tier-1 Investors

Established ecosystem of industry partners, OEMs, ISVs, and end-users

Forged strategic partnerships with IBM, VMware, and SanDisk

Significant Time to Market Advantage

IT'S ALL ABOUT THE **APPLICATIONS!**

LOW LATENCY APPLICATIONS

+ TRANSACTION LOGGING
+ TRANSACTION PLAYBACK
+ LOW LATENCY MESSAGING
+ ELECTRONIC TRADING

VIRTUAL DESKTOPS

+ DATA CACHING
+ SERVER CONSOLIDATION
+ VM GOLDEN IMAGES
+ VSAN

DATABASE/ HYPERSCALE

+ FREQUENTLY ACCESSED TABLES
+ LOG FILES
+ TEMPDB FILES
+ INDEXING
+ PARTITION TABLES

BIG DATA ANALYTICS

+ LOG REDUCTION
+ LOG PARSING
+ EVENT LOGGING
+ MEMCACHED
+ IN MEMORY DATA GRIDS

SERVER VIRTUALIZATION

+ VSAN
+ SOFTWARE DEFINED STORAGE
+ VIRTUALIZED DATABASES
+ DATA CACHING

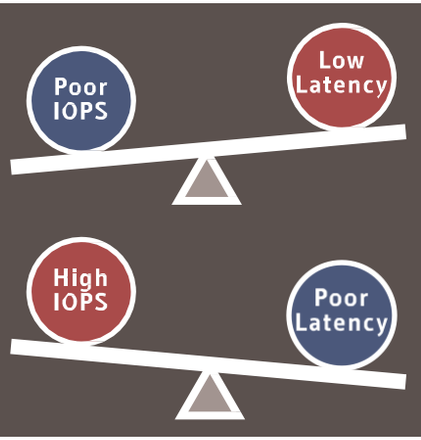
THE PERFORMANCE TRADE-OFF

+ Traditionally customers have faced a suboptimal trade-off in storage system design:

OPTIMIZE LATENCY → **SACRIFICE IOPS**

OR

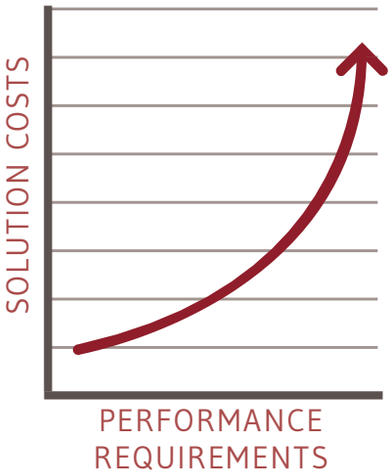
OPTIMIZE IOPS → **SACRIFICE LATENCY**



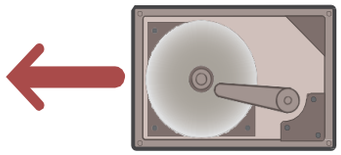
A Painful Workaround…

+ When SSD "IOPS vs. Latency" trade-offs are unacceptable, adding expensive RAM is a traditional recourse.

+ However, adding RAM can create an imbalance between incremental performance requirements and rapidly growing solution cost.



FLASH STORAGE EVOLUTION THUS FAR



(Decades of evolution)

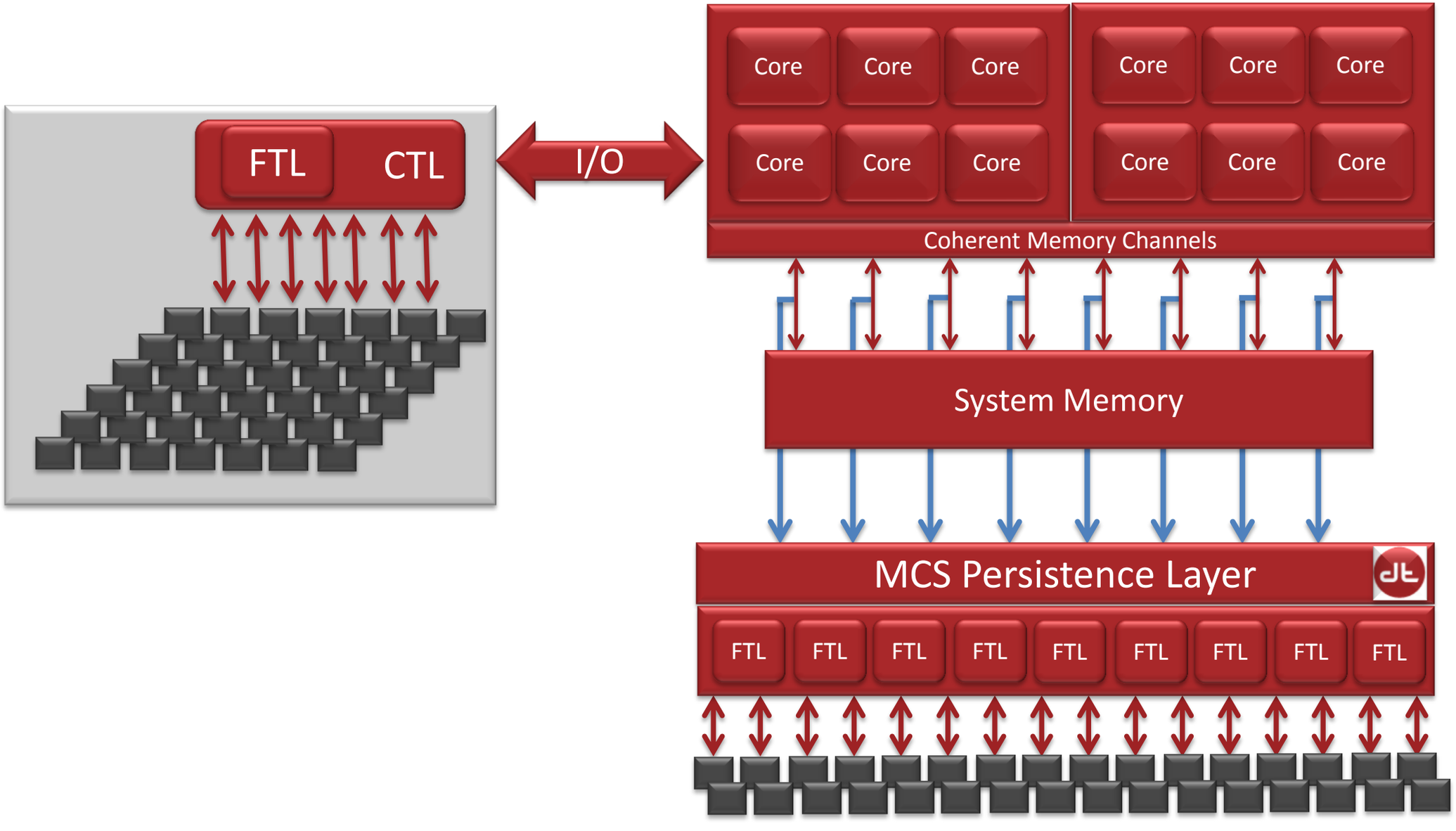
— SATA SSDs move FLASH into the Enterprise

— PCI SSDs bring FLASH closer to the CPU

????

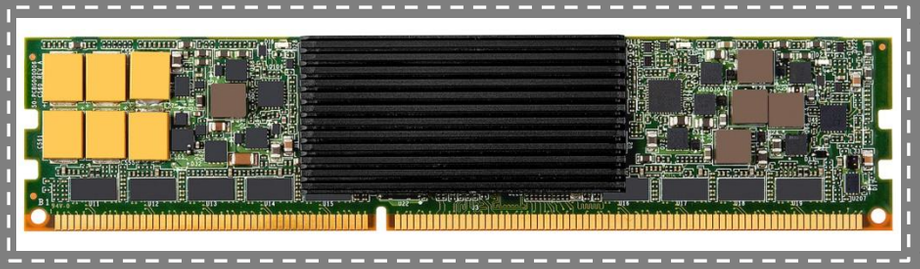
2000 2005 2010 NOW 2015

Enter Memory Channel Storage (MCS™)



Massive flash capacity exposed through the low-latency memory subsystem

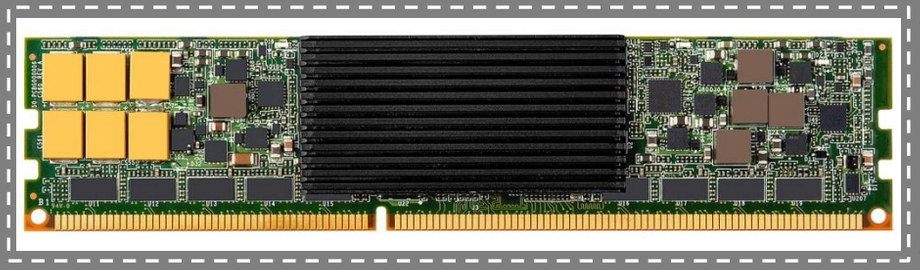
LEVERAGING THE MCS PLATFORM: **TODAY**



- Block Interface
- No App/OS Changes Required
- Functionally Replaces Existing Solutions

- Rapid Deployment
- Seamless Integration
- Storage Acceleration

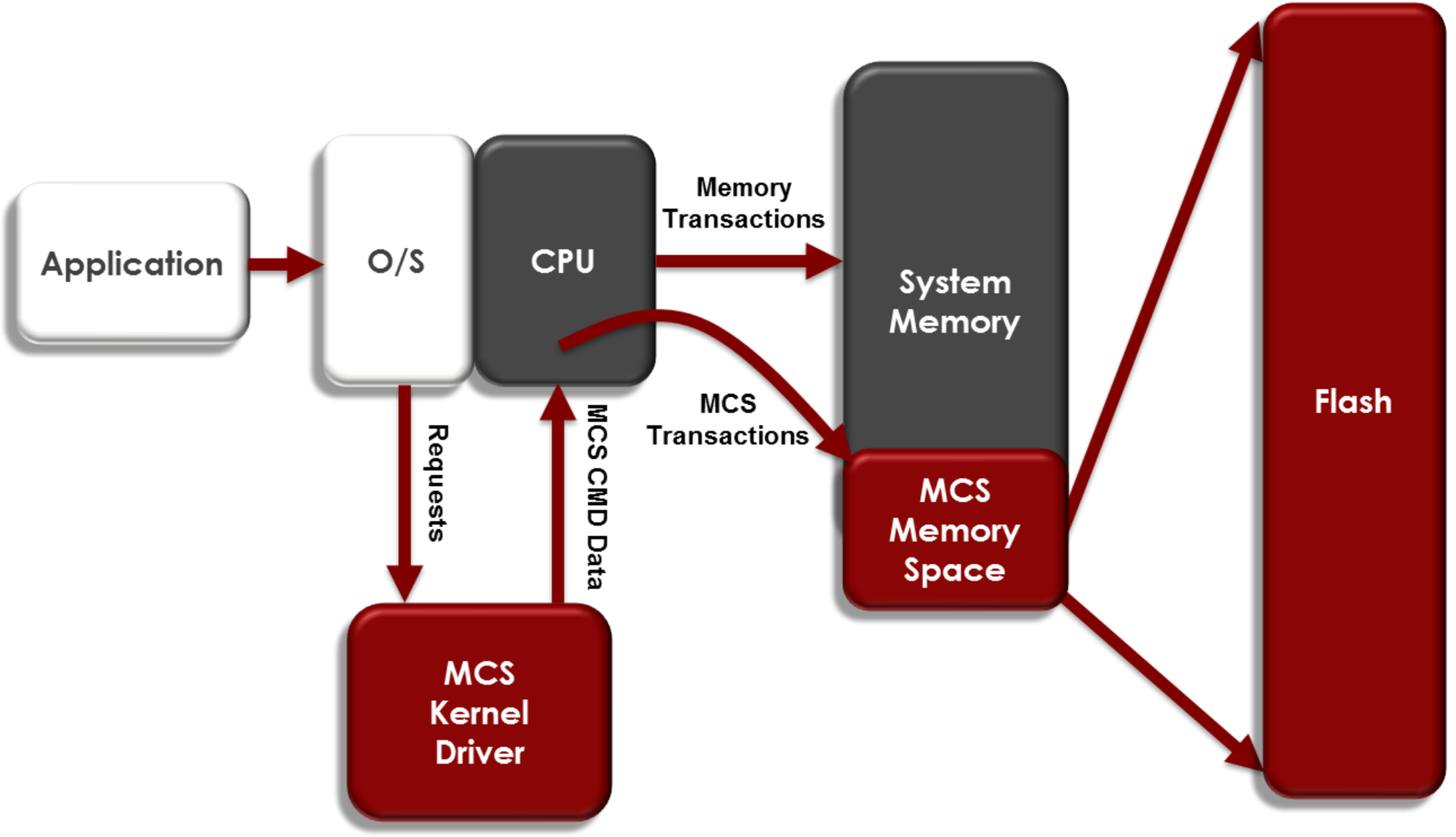
LEVERAGING THE MCS PLATFORM: **TOMORROW**



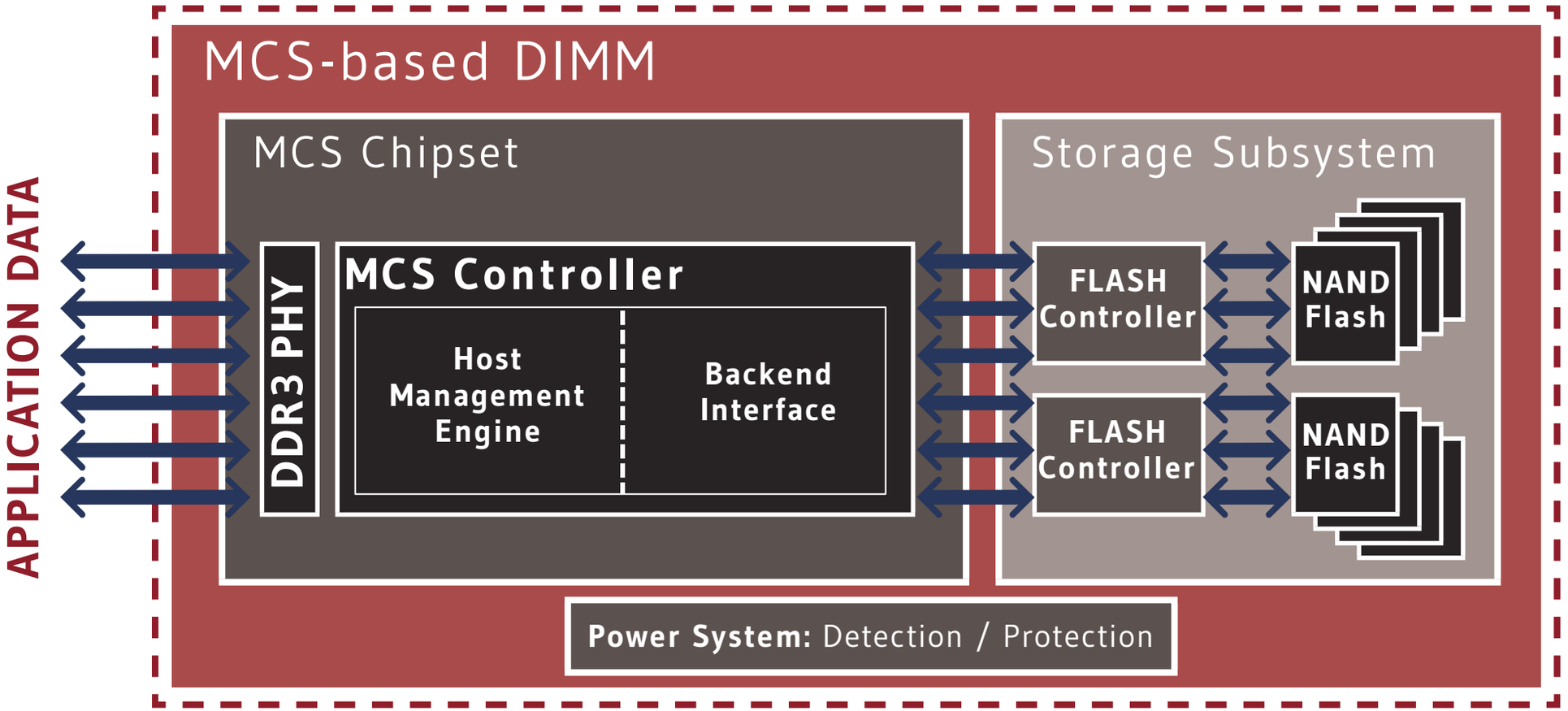
- Cache Line Interface
- Apps Optimized To Leverage MCS
- New Usage Models Enabled

- Flexible Access
- Deep Integration
- Memory Extension

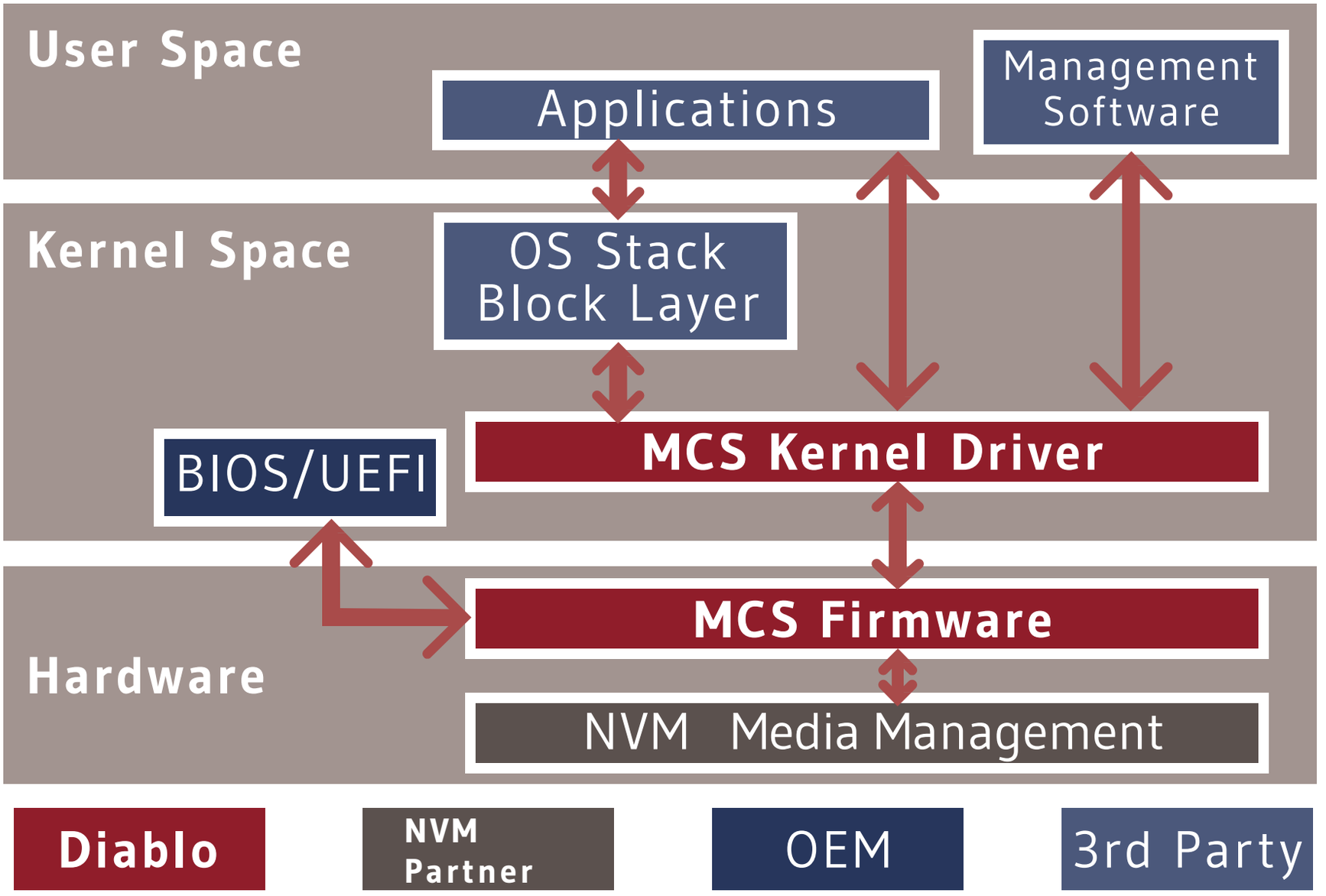
MCS IN SYSTEM MEMORY MAP



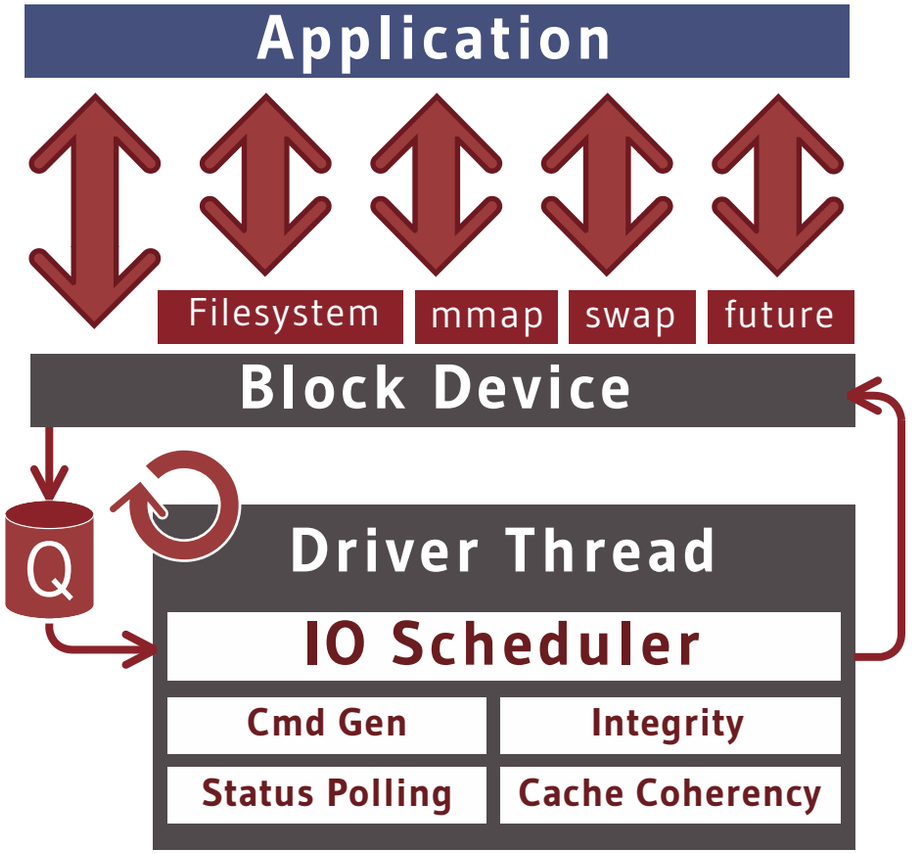
HARDWARE ARCHITECTURE



SOFTWARE ARCHITECTURE

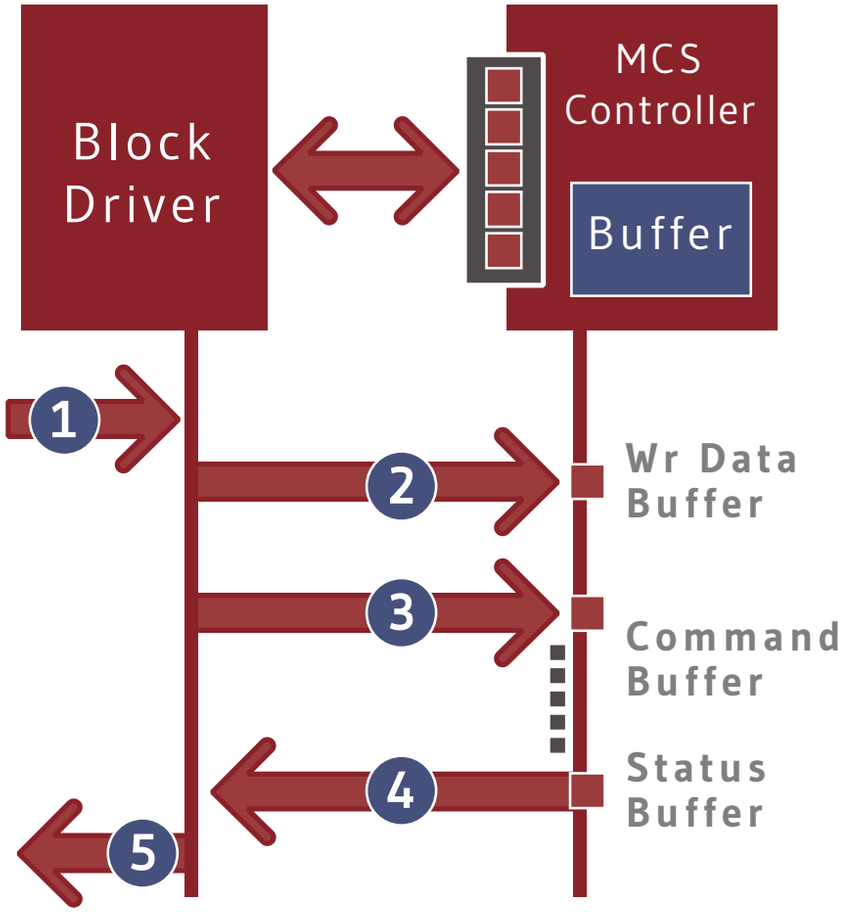


DRIVER DETAILS



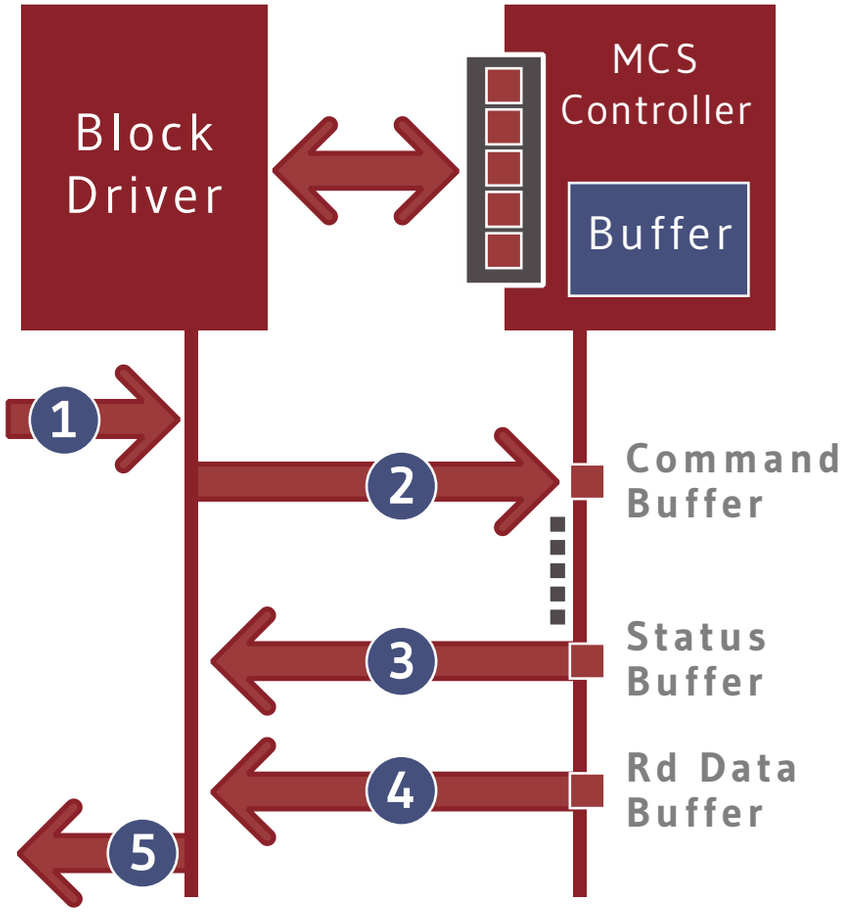
+ **Plugs into block layer:**
  + Bypasses SCSI/SATA on Linux
  + Emulates SCSI on Windows and VMware
+ **Handles req's asynchronously:**
  + Kernel posts requests into driver's incoming request queue.
  + Driver thread generates commands, posts to device, checks status, and copies data.
+ **Handles data and control req's:**
  + 512B – 4kB native atomics
  + Up to 32kB atomics with FW aid
  + SMART logs, thermal data, stat, events etc.

EXAMPLE WRITE



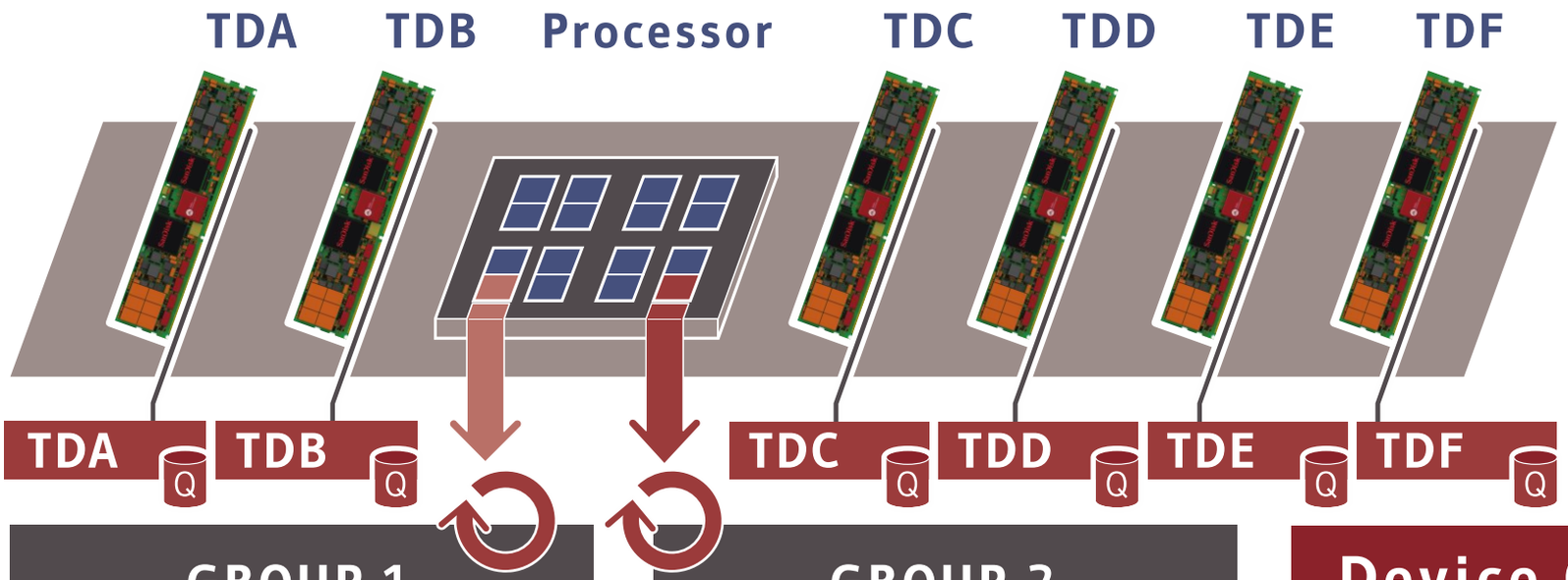
1. **OS requests a write.**
2. **Driver writes data to write buffer.** (4kB data plus optional metadata)
3. **Driver constructs a Diablo MCS protocol command, and writes it to a command buffer.** (encodes intent, LBA, buffer number, and E2E integrity metadata)
4. **Driver checks status.**
5. **Driver completes the write.**

EXAMPLE READ



1. **OS requests a read.**
2. **Driver constructs a Diablo MCS protocol command, and writes it to a command buffer.**
   (encodes intent, LBA, buffer number, and E2E integrity metadata)
3. **Driver checks status.**
4. **Driver reads data from read Buffer, and validates integrity.**
   (4kB data plus optional metadata)
5. **Driver completes the read.**

CONFIGURABLE DEVICE GROUPS



GROUP 1

CPU: 1

DEVICES: A, B

GROUP 2

CPU: 4

DEVICES: C, D, E, F

NOTE 1: Shown only for one NUMA node, but this pattern is replicated on each node.

NOTE 2: Devices can be combined in any combination.

Device Grouping:

+ Configurable CPU affinity
+ 1 Thread round robins between active devices
+ Efficiency through driver/device locality
+ Flexible prioritization of latency vs. CPU usage

TECHNOLOGY COLLABORATION TO CREATE **THE FIRST MCS-ENABLED PRODUCT**

diablo technologies™

+ Reference architecture design
+ DDR3 to SSD ASIC/firmware
+ Kernel and application level software development
+ OEM System Integration and enterprise application domain knowledge

+

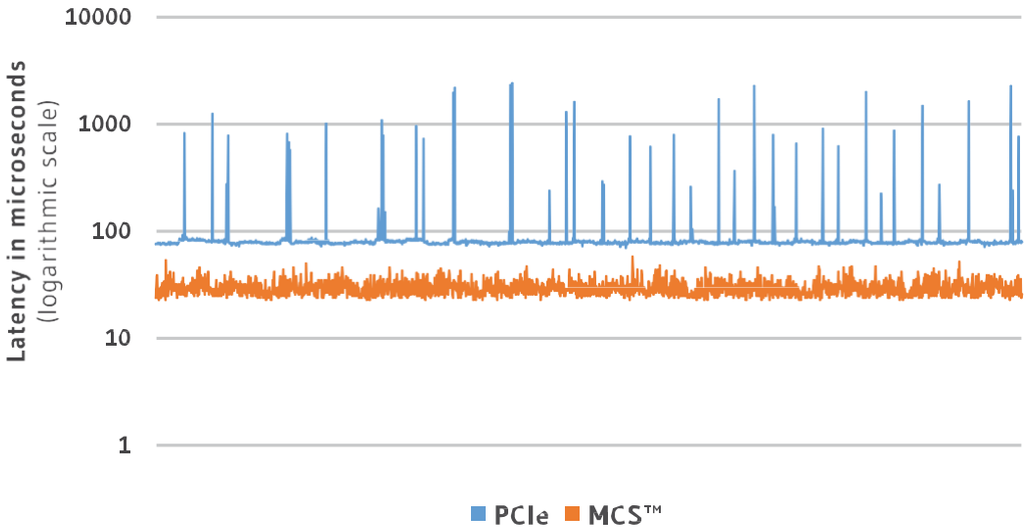
SanDisk®

+ Guardian Technology for enterprise applications
+ SSD controller & FTL firmware development and test
+ Supply Chain and Manufacturing with flash partner
+ System Validation

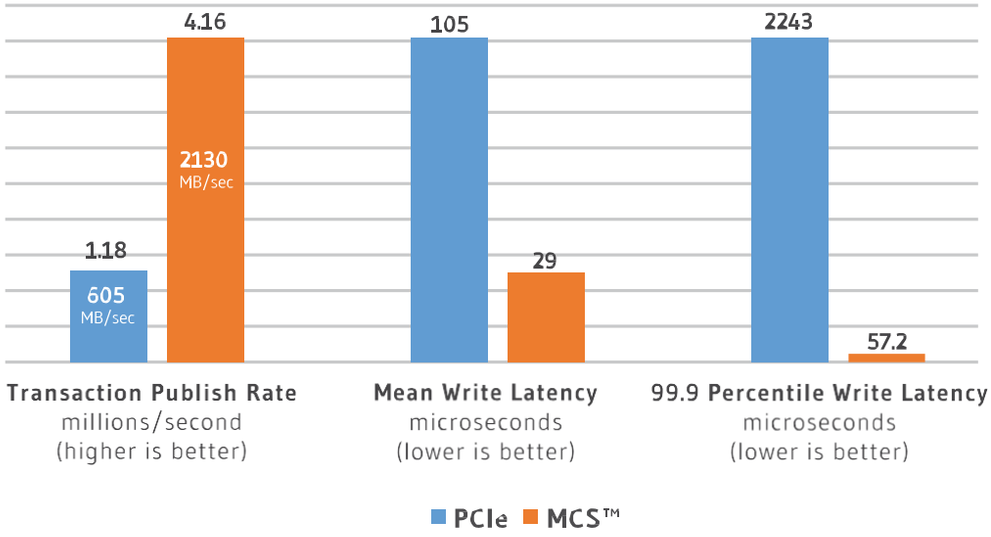
REDUCED LATENCY
ENABLES REAL-TIME ANALYTICS

60East TECHNOLOGIES + diablo technologies™

15% Read Mix



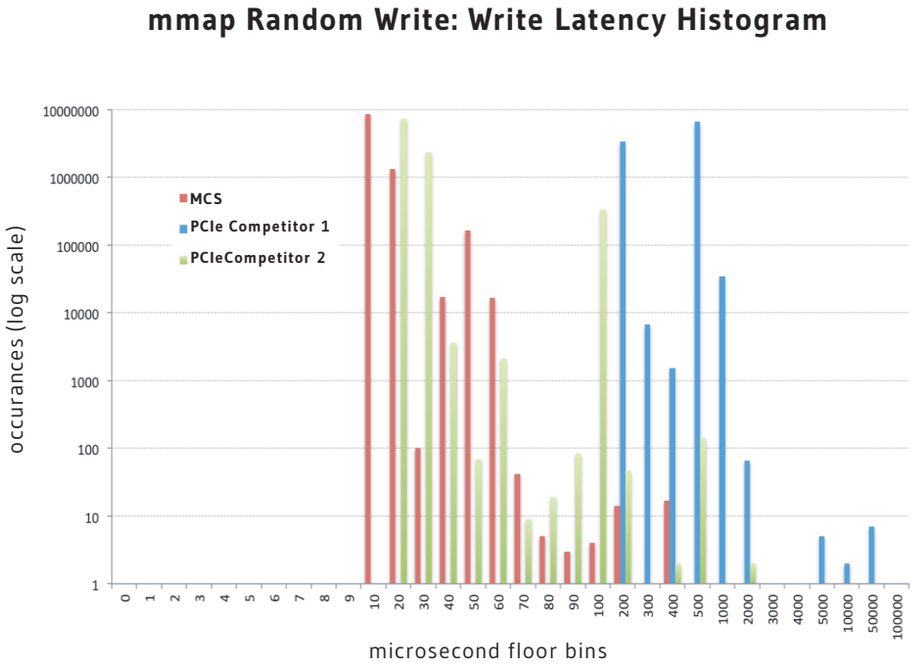
15% Read/Write Ratio Overview

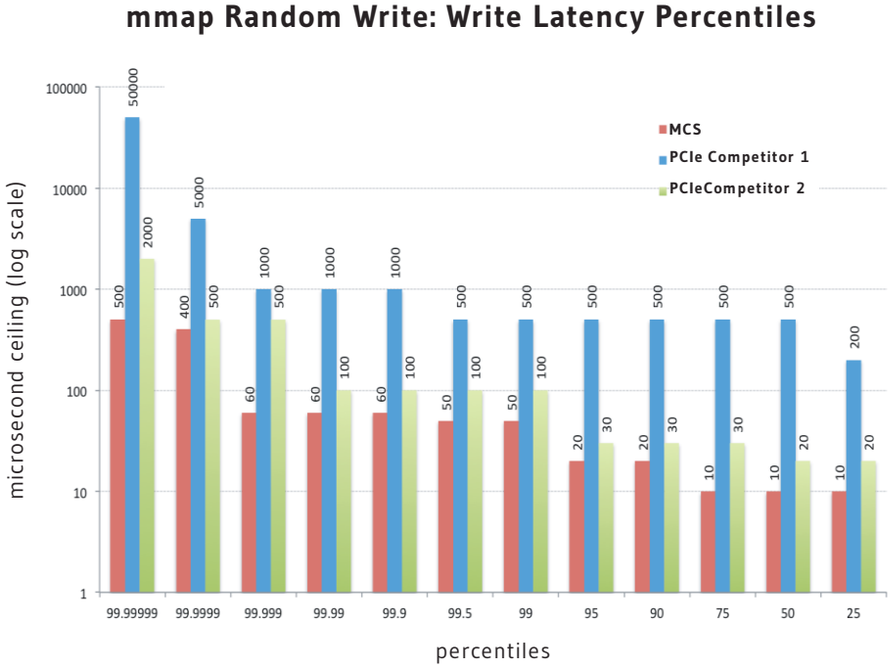


+ THE APPLICATION HAS BECOME THE BOTTLENECK IN E-TRADING

MEMORY MAPPED **I/O ACCELERATION**

10 million records (20GB mmap) using synchronous msync calls

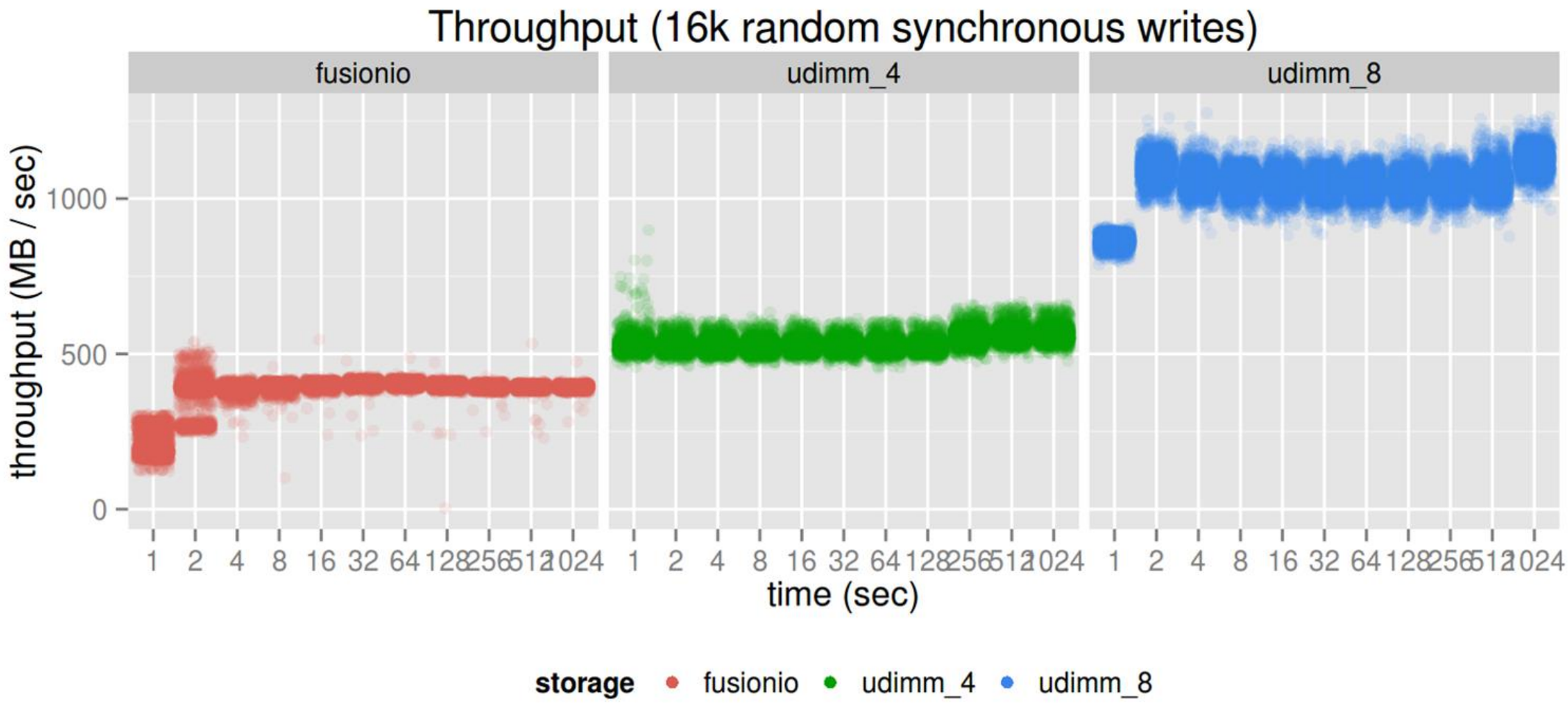




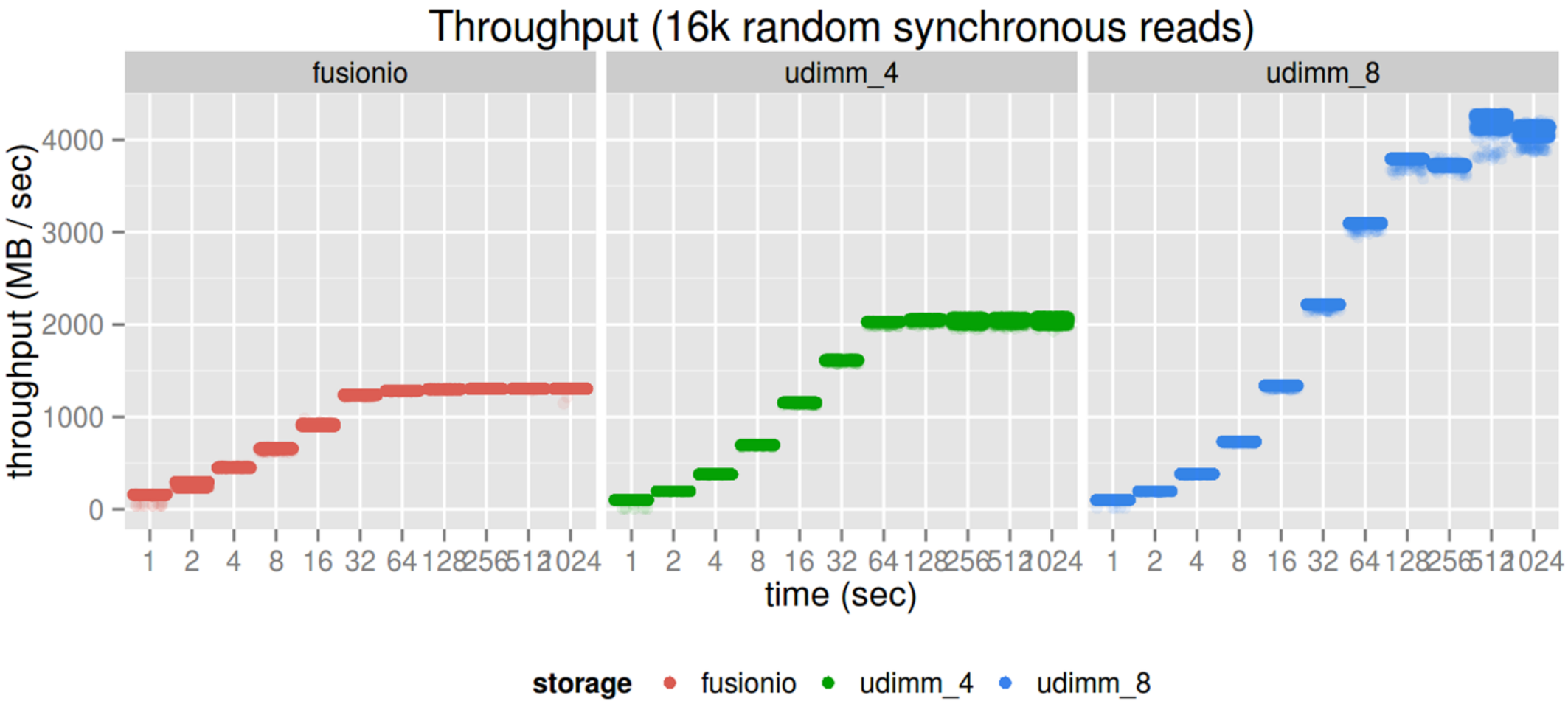
+ MCS 99th-percentile latency is **2x lower than Competitor 2** and **10x lower than Competitor 1**

+ MCS has the **tightest latency distribution**

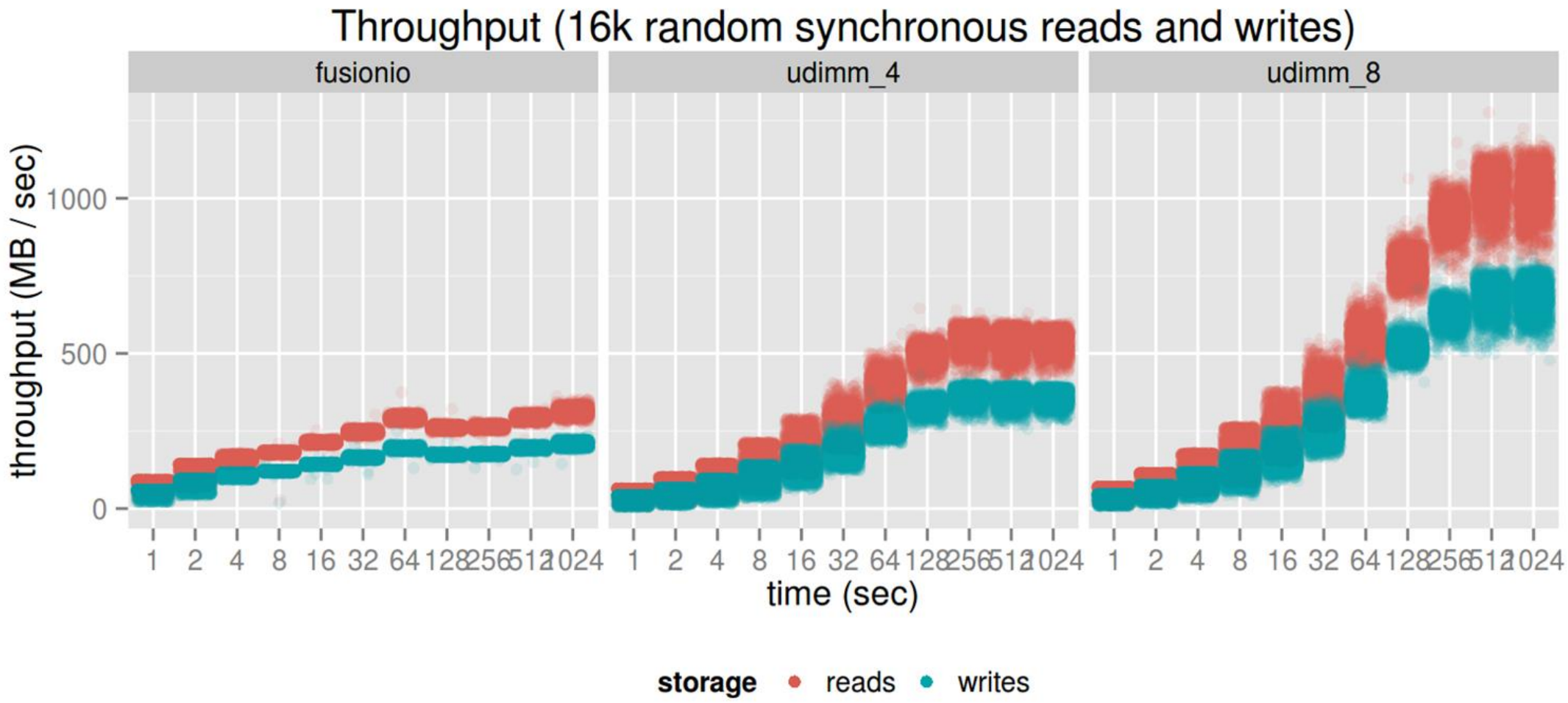
SYNCHRONOUS WRITES



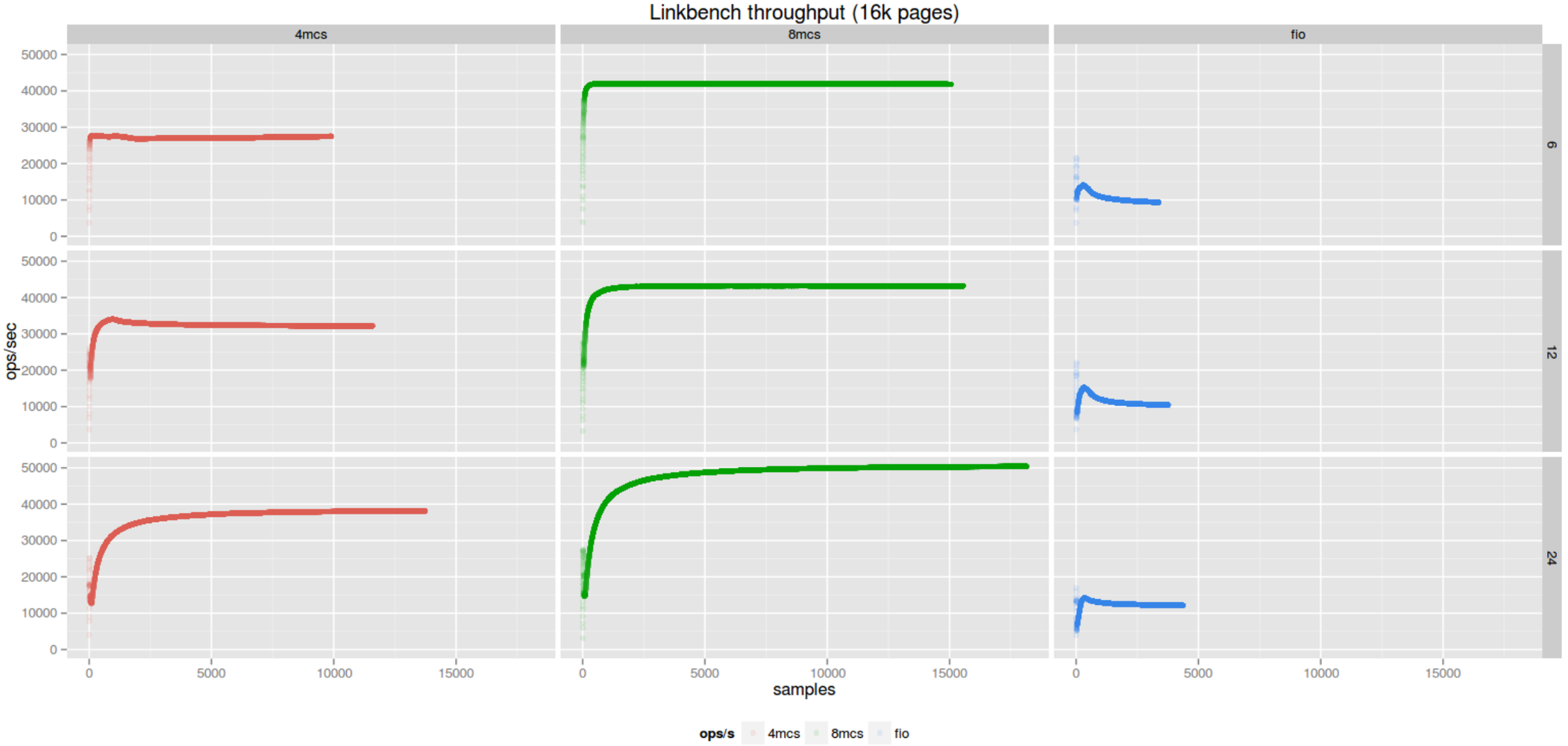
SYNCHRONOUS READS



SYNCHRONOUS MIXED

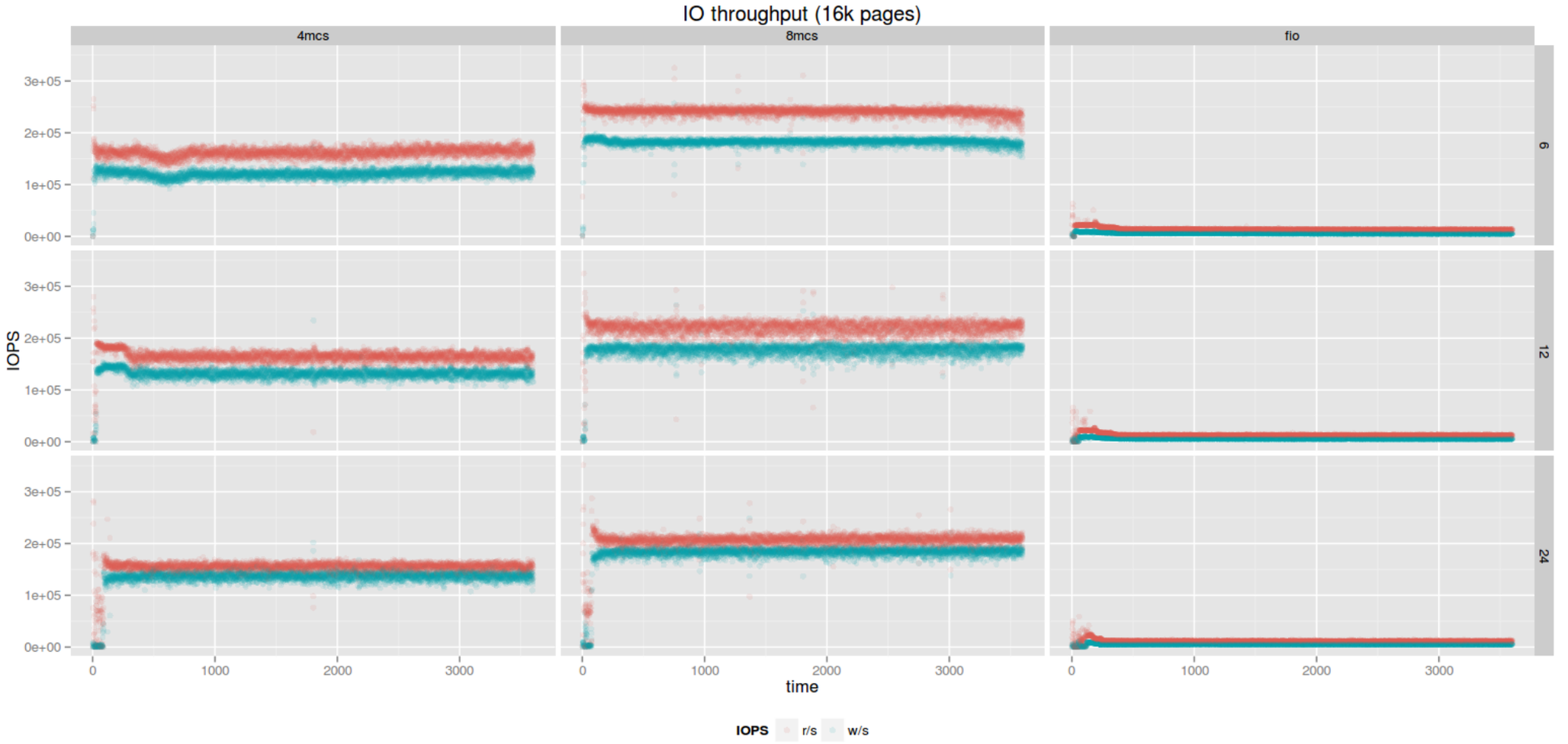


Linkbench MySQL Load



-Linkbench is CPU bound with MCS – more than 70% of CPU time is spent in USR
-Linkbench is IO bound with Fusion – more than 70% of CPU time is spent in iowait

Linkbench



-MCS based solution is not IO bound
-Adding more CPU power WILL increase server productivity

MCS FOR VIRTUALIZATION

VIRTUAL MACHINE ACCELERATION

- Ultra low response times for virtualized applications
- Ideal as VM file swapping data store
- Fully certified technology for ESXi vSphere platform

PERFECT SOLUTION FOR VSAN

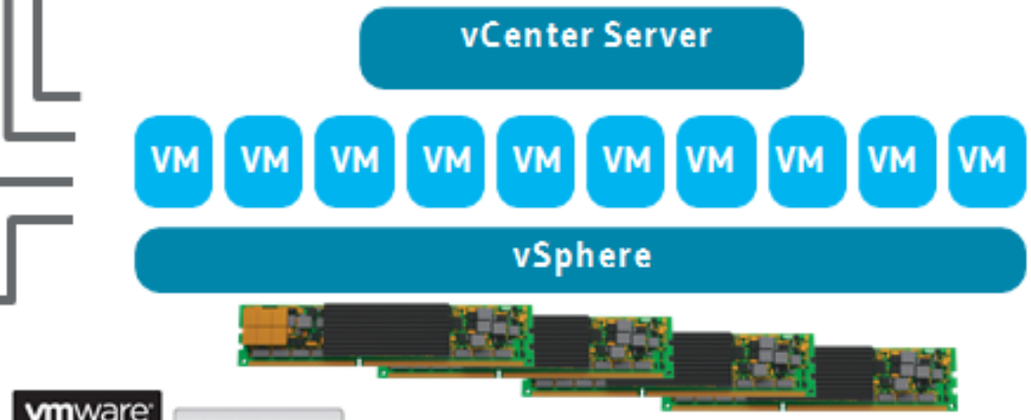
- MCS eliminates the need for external storage arrays
- Extremely fast commit to clustered nodes for HA
- Predictable IOPS and latency for heavy workloads

CACHING SOLUTION FOR VIRTUALIZED ENVIRONMENTS

- Primary cache for hot data stored externally
- Ideally suited for random, mixed virtualized workloads
- Strong solution for virtualized relational and in-memory databases

REDUCED TOTAL COST OF OWNERSHIP

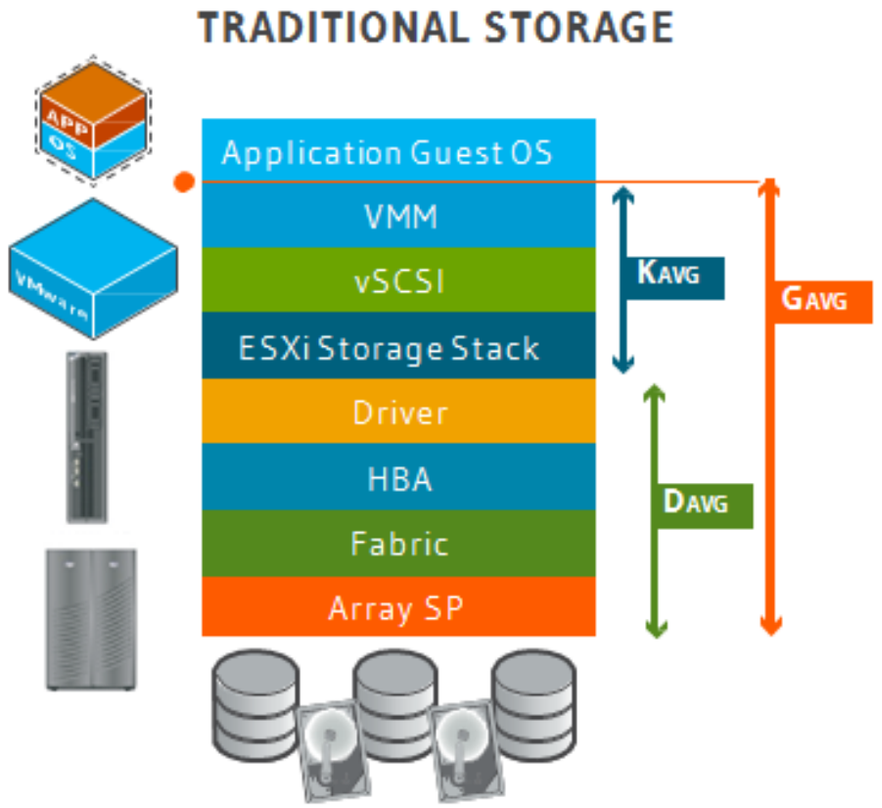
- Requires 1/6 typical memory/VM as MCS offsets DRAM
- Allows up to 4x VMs per host compared to PCIe-based flash
- Improved VDI user experience with lower Capex/Opex

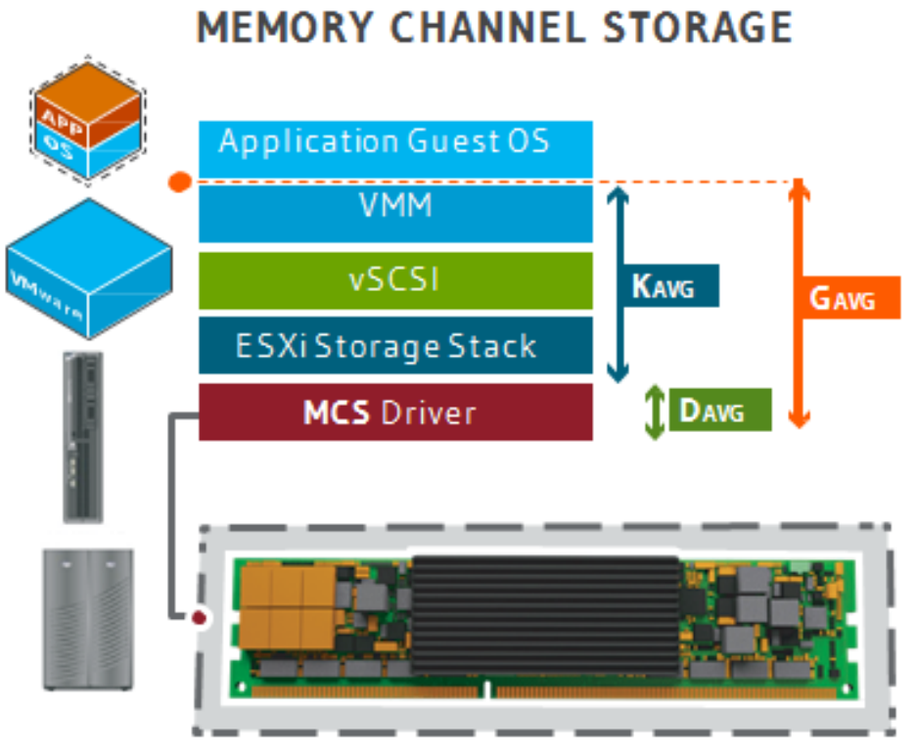


vmware PARTNER TECHNOLOGY ALLIANCE

vmware READY

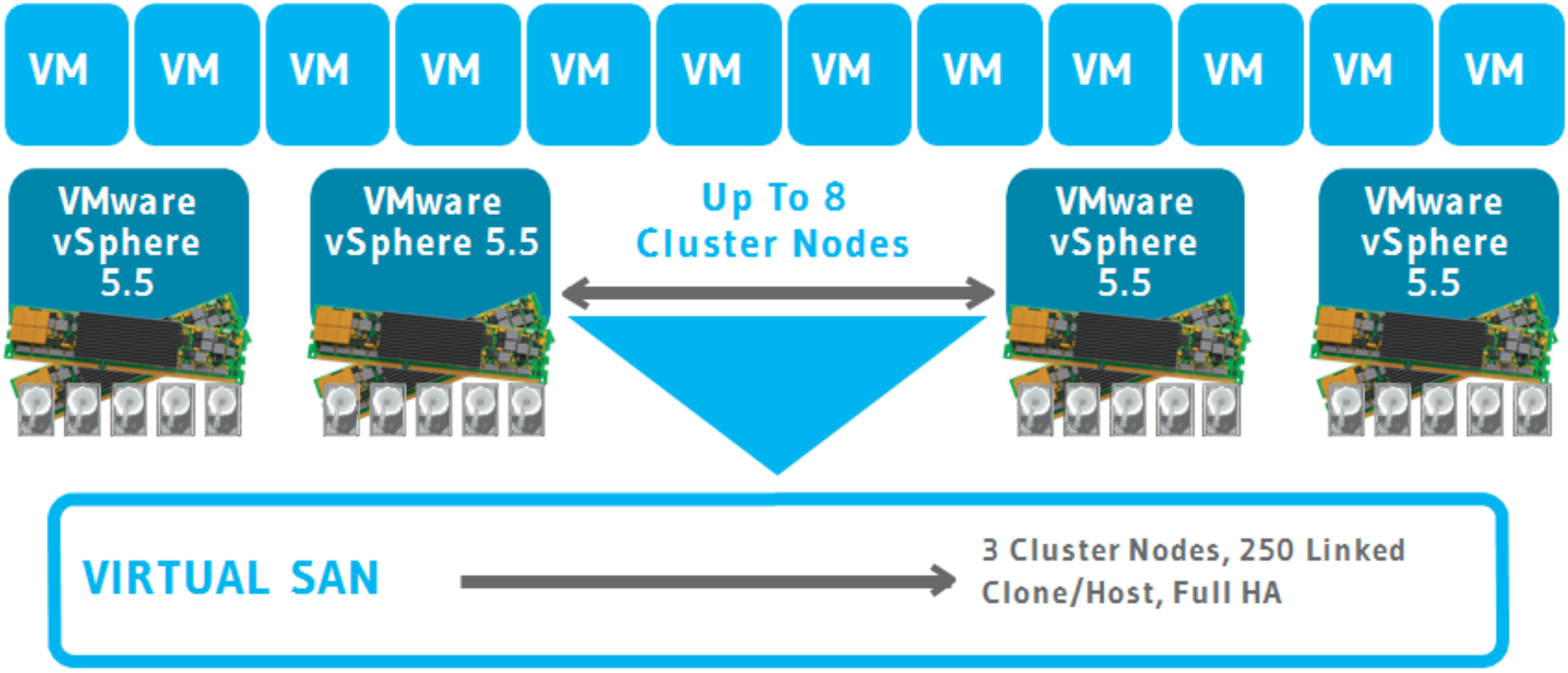
SOLVING THE I/O RESPONSE TIME ISSUE





K_{AVG}: Total Kernel Time **D_{AVG}**: Device Latency **G_{AVG}**: Total VM Latency

MCS AS FLASH TIER IN SPINDLE-BASED VSAN

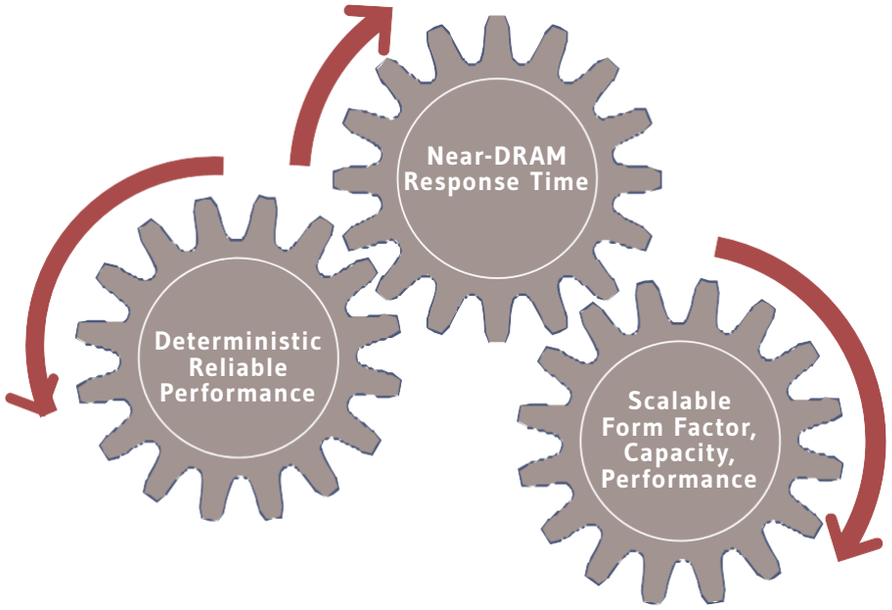


SUMMARY

Memory Channel Storage

+ Leverages parallelism and scalability of the memory channel

+ Significantly reduces data persistence latencies and improves single thread throughput



Benefits of MCS

+ 200GB to tens of TB's of flash in standard DIMM form factor and DDR3-CPU interface

+ Disruptive performance accelerates existing applications and enables new flash use cases

+ Scalability facilitates economic, "right-sized" system solutions

+ Form factor enables high-performance flash in servers, blades, and storage arrays

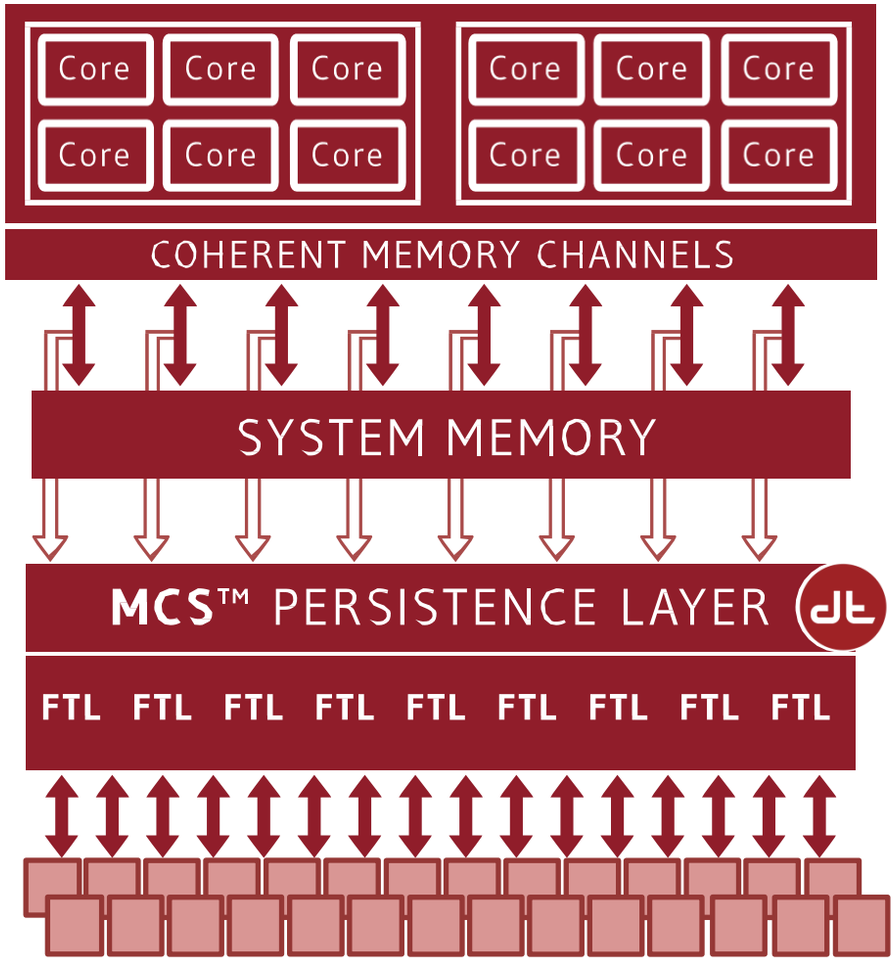
+ Future proofed with ability to utilize NAND-flash and future non-volatile memories

THANK YOU!

mamer@diablo-technologies.com

diablo technologies™ | PROBLEM SOLVED.

MCS SYSTEM VIEW

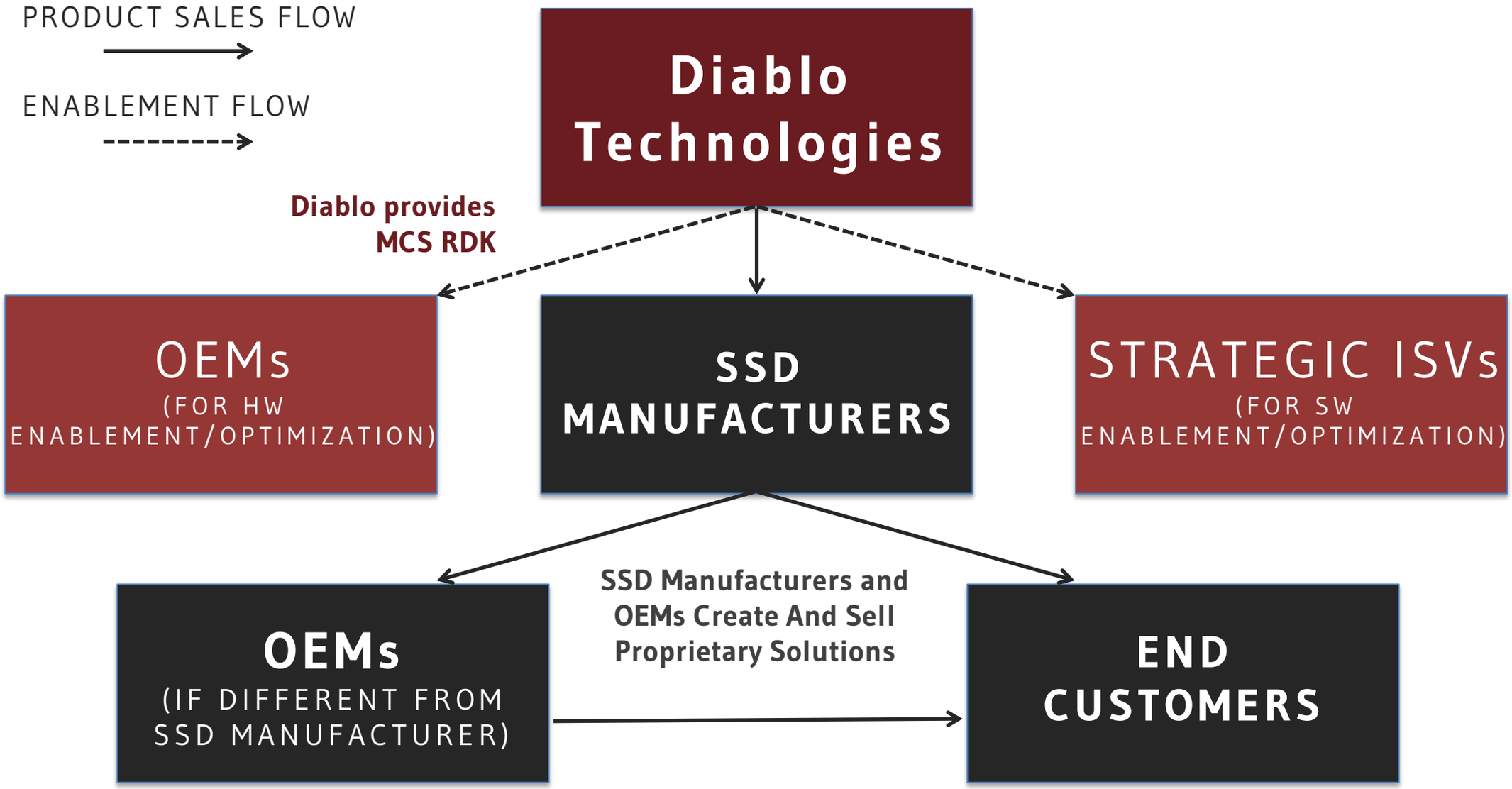


Leveraging the **Power of Parallelism…**

+ Massive Flash capacity exposed through the low-latency memory subsystem.

PROBLEM SOLVED.

MEMORY CHANNEL STORAGE ECOSYSTEM



MEMORY CHANNEL STORAGE
REFERENCE DESIGN KITS (RDKs)

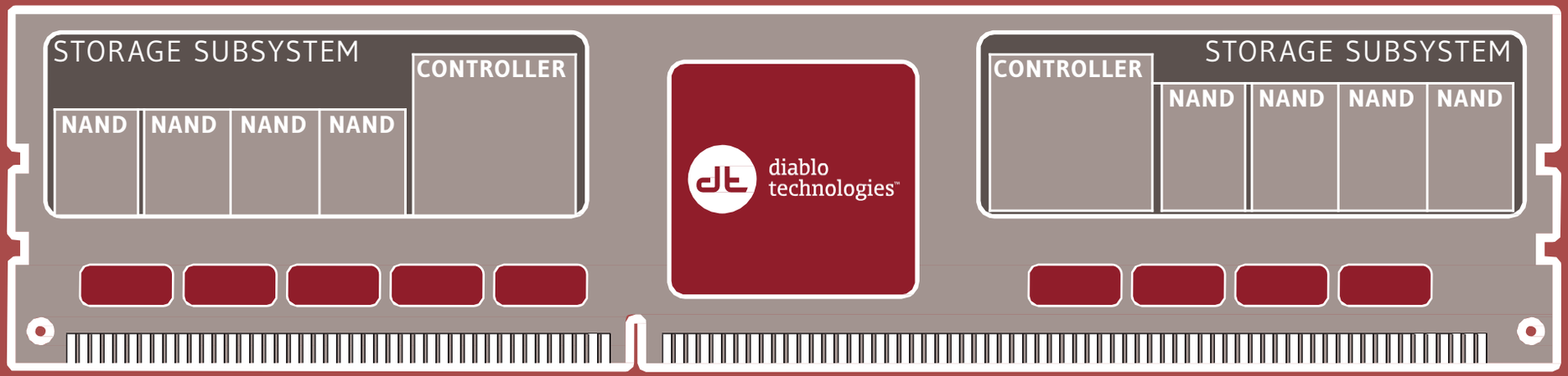
Modular, Reference Solutions for Enablement/Evaluation by SSD manufacturers, OEMs, and ISVs

Each **RDK includes:**

- MCS Chipset
  - Enables hardware interface via Memory Channel
  - Includes full firmware
- MCS Drivers
  - Manages communication between Host and MCS Module(s)
  - Diablo drivers for Windows, VMware ESXi and popular Linux distributions/kernels
- Storage Subsystem
  - Reference Non-Volatile Memory (NVM) solution
  - Final NVM solution will vary according to SSD Manufacturer/OEM preference

MEMORY CHANNEL STORAGE **CARBON$_1$**

- The First Commercialized MCS RDK
  - Enables NAND Flash to Directly Interface on the Memory Channel
- Presents as a Block I/O Device
  - Can be Managed just like Existing Storage Devices
- DDR3 Interface, Standard RDIMM Physical Form Factor
  - Plugs into Standard DIMM Slots
  - Self-contained, No External Connections Required



MCS CARBON$_1$:
SYSTEM REQUIREMENTS & COMPATIBILITY

+ Hardware and BIOS Requirements

- + Server enabled with MCS UEFI BIOS modifications
- + DDR3-compatible processor
  - + Compatible with standard JEDEC-compliant 240-pin RDIMMs
  - + Supports DDR3-800 through DDR3-1600
- + 8GB of standard memory (RDIMM) installed in the system
- + Follows standard server DIMM population rules

+ Initial OS Support

- + Linux (RHEL, SLES)
- + Windows Server
- + VMware ESXi

vmware

Microsoft

ANATOMY OF AN ISV ENGAGEMENT: **PERCONA**

PERCONA

+ Percona Tested Memory Channel Storage devices
  + Percona is oldest and largest independent MySQL provider
  + Experts in MySQL and InnoDB Performance
  + Serving more than 2,000 customers in 50+ countries
  + Provide and support Percona Server MySQL distribution

+ Performance Consulting
  + MySQL architecture and design reviews
  + Diagnosing and solving MySQL performance problems
  + Optimization of MySQL on SSD infrastructure
  + Performance Audits to identify performance improvements

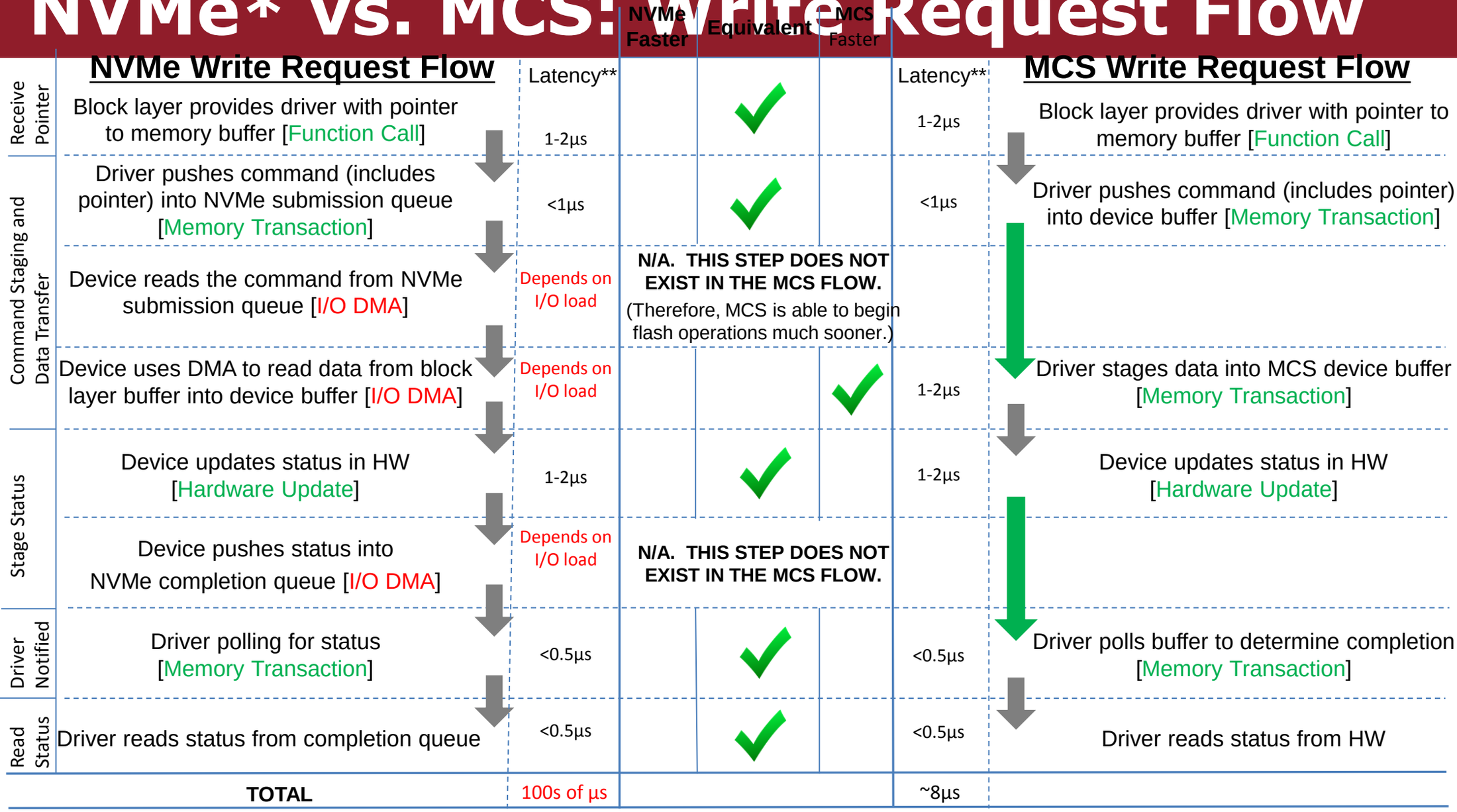
+ Diablo ISV Partnering Analysis identified Percona as critical partner

ANATOMY OF AN ISV ENGAGEMENT: **PERCONA**

PERCONA

- Percona Memory Channel Storage Testing
  - Tested Carbon$_1$ reference design
  - Tested ULLtraDIMM Carbon$_1$ based product
- Benchmark Testing
  - Sysbench Benchmarks
  - Linkbench
- Metrics Measured
  - Reads/Writes/Mixed Workload
  - Throughput
  - Operations per Second
  - 95th Percentile Response Time

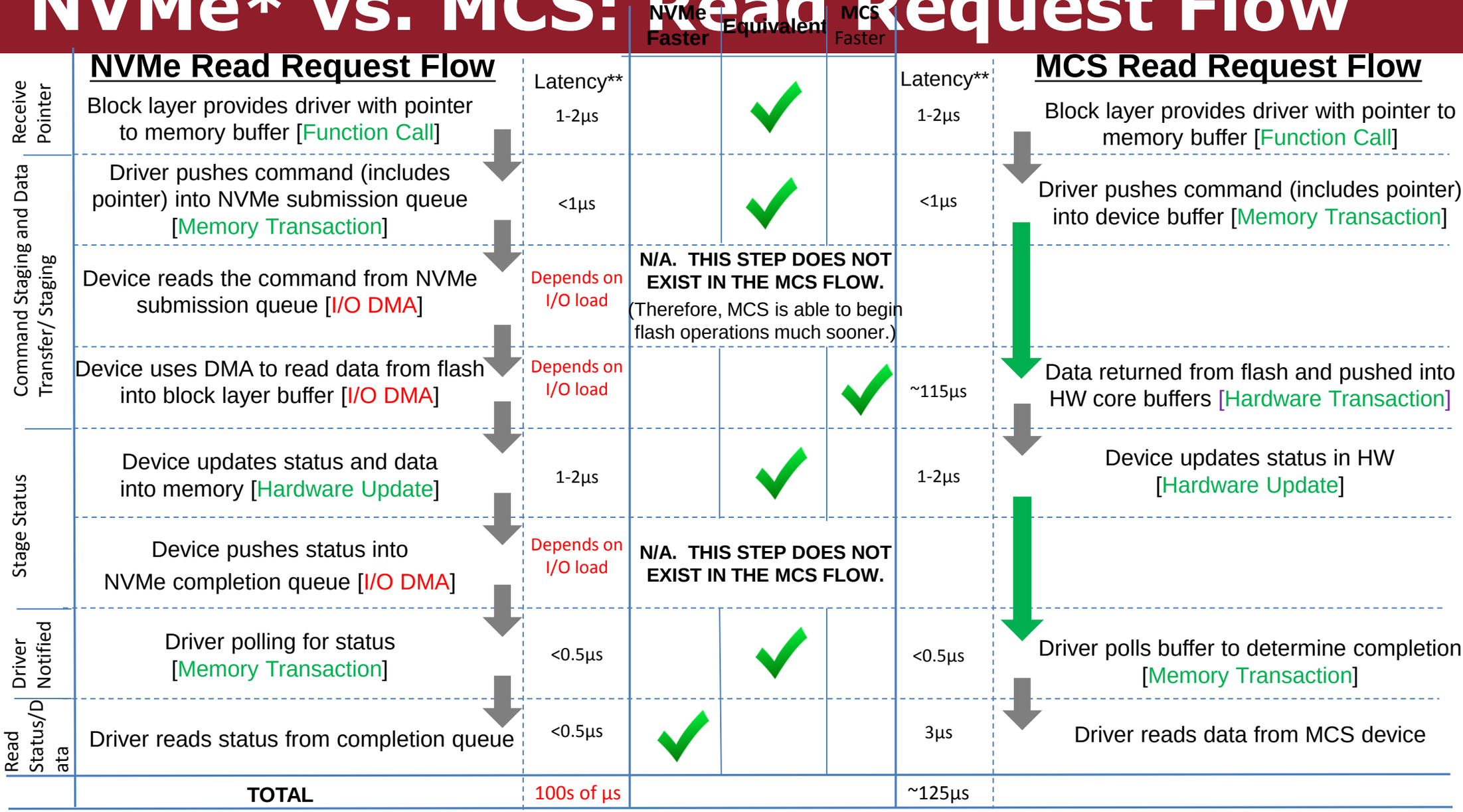
NVMe* vs. MCS: Write Request Flow



- **Memory transactions (and transactions occurring within device hardware) are very deterministic and faster than I/O DMAs**
- **I/O DMAs involve the I/O controller and are non-deterministic (subject to conflicts with other system I/O)**

**NVMe flow depicted since the current PCIe flow (through SCSI stack) is commonly accepted as inefficient.*
***Latencies under heavy I/O load (high IOPS)*

NVMe* vs. MCS: Read Request Flow



- **Memory transactions (and transactions occurring within device hardware) are very deterministic and faster than I/O DMAs**
- **I/O DMAs involve the I/O controller and are non-deterministic (subject to conflicts with other system I/O)**

**NVMe flow depicted since the current PCIe flow (through SCSI stack) is commonly accepted as inefficient.*

***Latencies under heavy I/O load (high IOPS)*