SNIA DSI DATA STORAGE INNOVATION CONFERENCE

Creating Higher Performance Solid State Storage with Non-Volatile Memory Express (NVMe)

J Metz, Ph.D

Cisco

SNIA

Agenda

- SCSI and NVMe Differences
- How NVMe Works
- Consequence of Efficiency
- NVMe and Ethernet
- Additional Resources

What This Presentation Is...

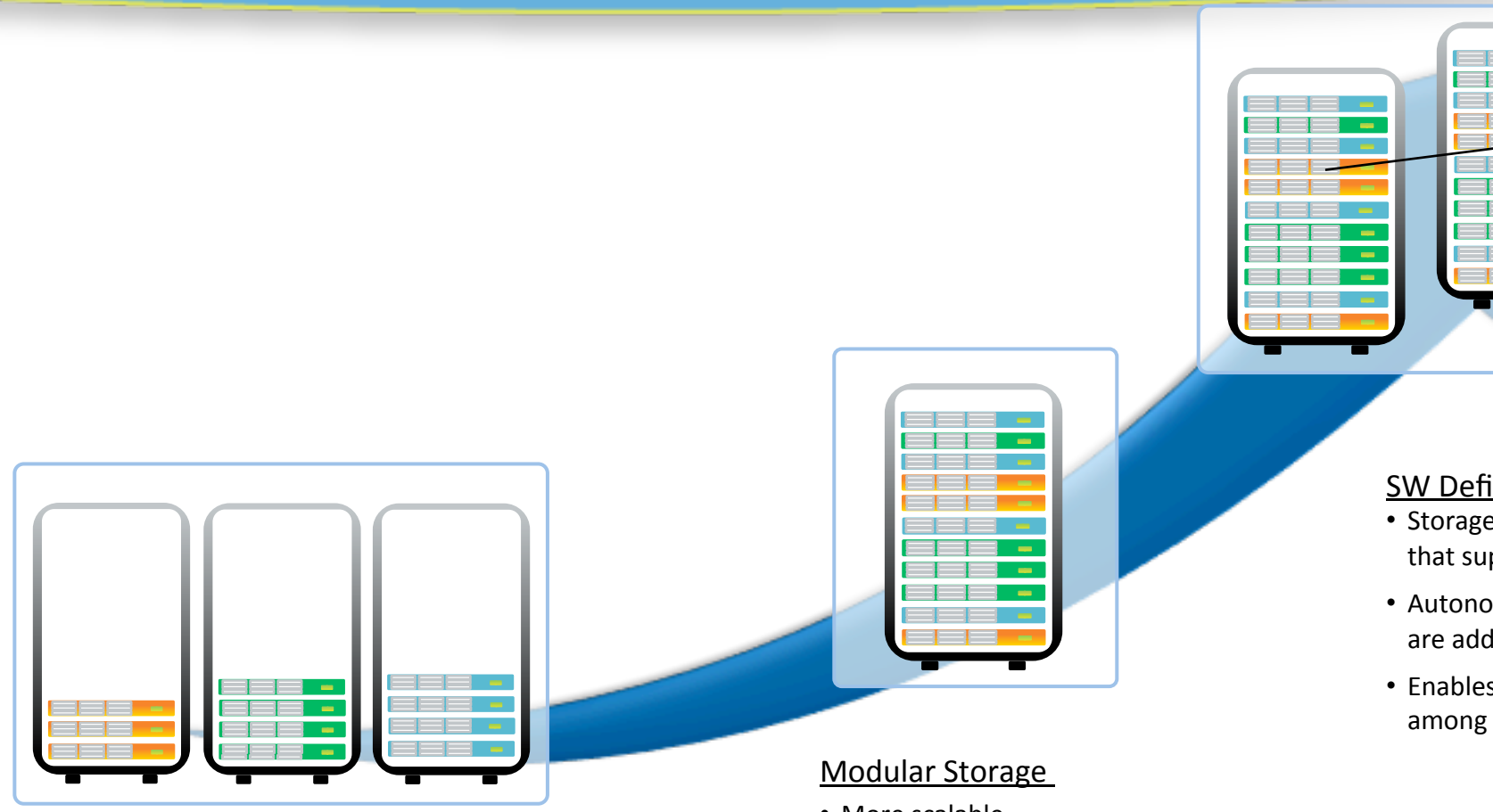
- Conversation about what NVMe is and why it's cool
- Start slow and get deeper as we go along
  - (bring everyone onto the same page)
- Some examples of NVMe efficiencies at work
- How NVMe works
- Future work and projects

What This Presentation Is *NOT*...

- Exhaustive
- A bit-by-bit primer
- Charts and graphs *ad nauseum*

Starting at the Beginning

Storage Market is Transforming



Fixed Storage Resources

- Separate storage networks
- Limited scalability

Modular Storage

- More scalable
- Faster time to deploy
- Pooled storage accessed by multiple servers/clients

SW Defi

- Storage that sup
- Autono are add
- Enables among

Storage Market is Transforming

SW Defined Storage

- Storage software installed on any standard off-the-shelf server that supports a service level objective
- Autonomous scaling and selection of storage tiers as resources are added or removed
- Enables applications to use heterogeneous storage resources among varying underlying infrastructure

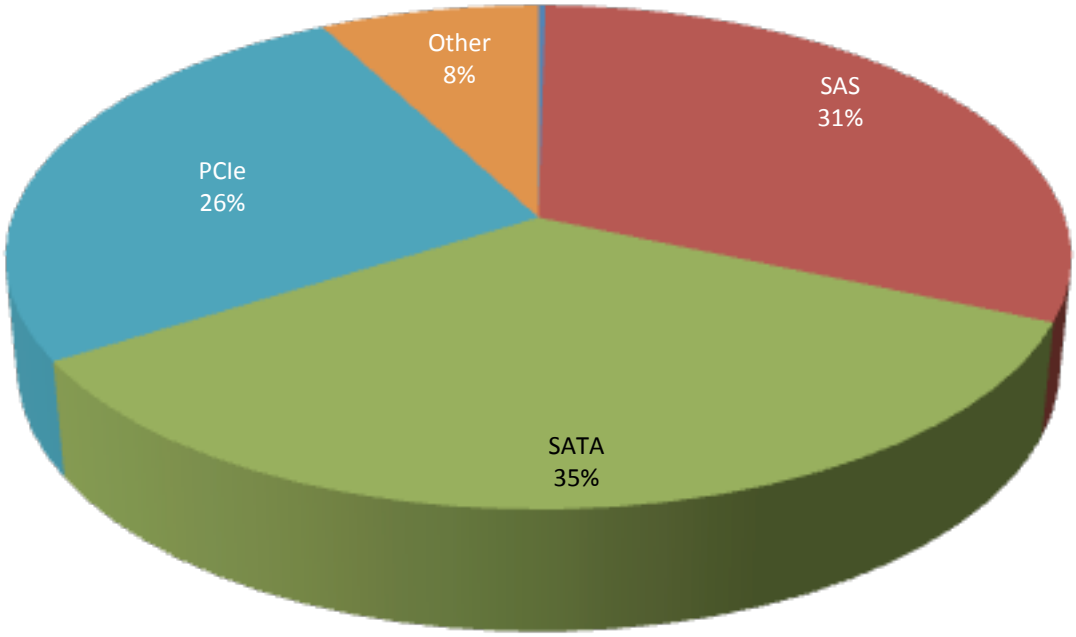
Modular Storage

- More scalable
- Faster time to deploy
- Pooled storage accessed by multiple servers/clients

Driving a Need for New Levels in Storage Performance

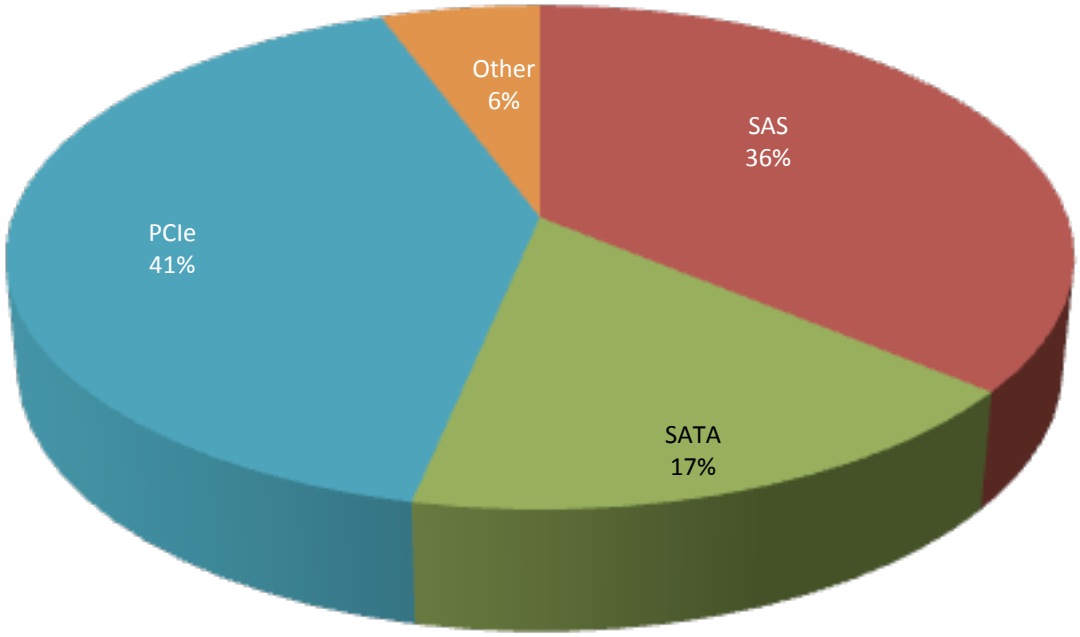
Interface Transitions

WW Enterprise SSD Revenue Interface Forecast, 2014



Interface Transitions

WW Enterprise SSD Revenue Interface Forecast, 2017



Expect multiple interfaces to be in multiple market segments, PCIe expecting to grow dramatically

The Flash Conundrum

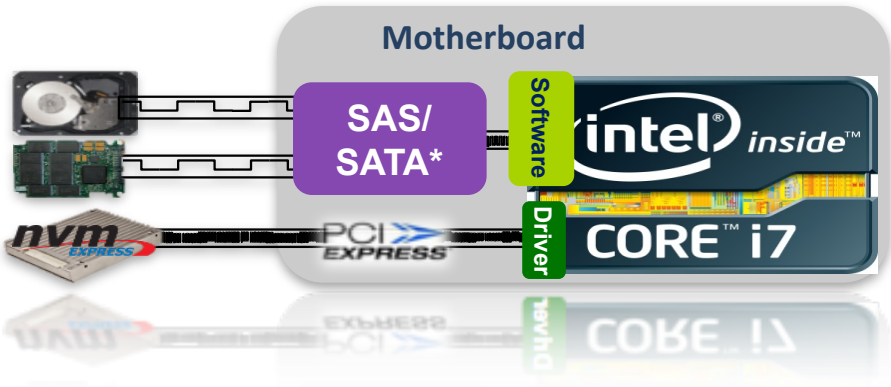
- Flash
  - Requires far fewer commands than SCSI
  - Does not rotate (no latency time, exposing latency of a one command/one queue system)
  - Thrives on random (non-linear) access
    - Both read and write
- Nearly all Flash storage systems use SCSI for access
  - But they don't have to!

What's So Great About PCIe for SSDs?

- PCIe is High Performance
  - Full duplex, multiple outstanding requests, and out of order processing
  - Scalable port width (x1 to x32)
  - Scalable link speed (2.5 GTps, 5 GTps, 8 GTps)
  - Low latency (no HBA overhead or protocol translation)
- PCIe is Low Cost
  - High volume commodity interconnect
  - Direct attach to CPU subsystem eliminates HBA cost
- PCIe is Power Efficient
  - Direct attach to CPU subsystem eliminates HBA power
  - Link power management (ASPM & dynamic link width/speed)
  - Optimized Buffer Flush/Fill (OBFF) & L1 Substates
  - Power Budgeting and Dynamic Power Allocation
  - Slot Power Limit

Enter NVM Express (NVMe)

- NVM Express (NVMe) is a standardized high performance host controller interface for PCIe Storage, such as PCIe SSDs
- Architected from the ground up for this and next generation Non-volatile Memory to address Enterprise and Client system needs
- Developed by an open industry consortium, directed by a 13 company Promoter Group
- Architected for on-motherboard PCIe connectivity
- Capitalize on multi-channel memory access with scalable port width and scalable link speed



NVMe - A Parable

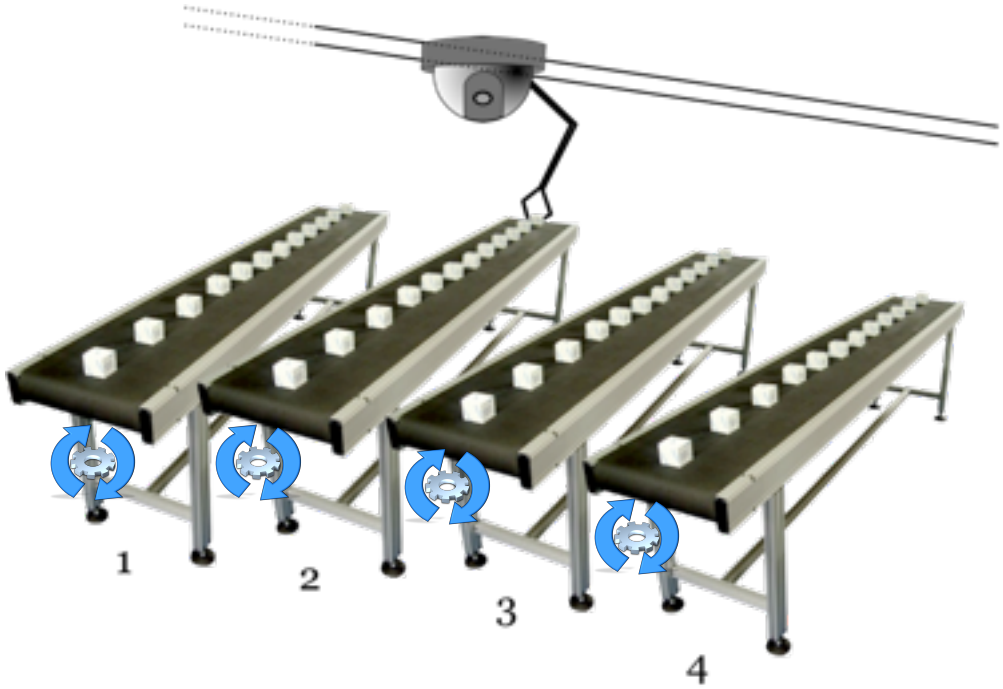
(With more than just a little snark)

SCSI Fundamentals

- SCSI is the command set used in traditional storage
  - Is the basis for all storage used in the data center
  - Obviously, is the most obvious starting point for working with Flash storage
- These commands are transported via:
  - Fibre Channel
  - Infiniband
  - iSCSI (duh!)
  - Other stuff
- Works great for data that can't be accessed in parallel (like disk drives that rotate)
  - Any latency in protocol acknowledgement is far less than rotational head seek time

Imagine...

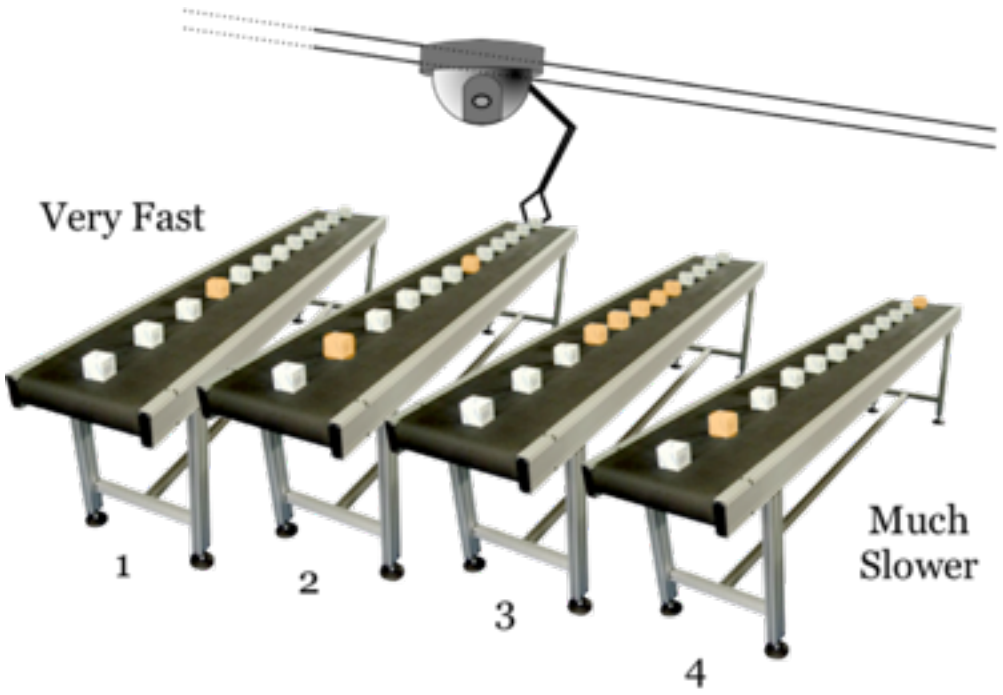
- You're programming a robot in a factory
  - "Go get me a block"
- Robot is on a track that moves back and forth to pick up blocks as they go by on a conveyor belt



Blog: "A Beginner's Guide to NVMe" http://sniaesfblog.org/?p=368

Speed Change-Up

- Worse, each conveyor belt is slower than the last
- Robot must
  - Move from one conveyor belt to the next (+time)
  - Wait for the block to come around (+time)
  - Move around more if the blocks are not in sequence (+time)



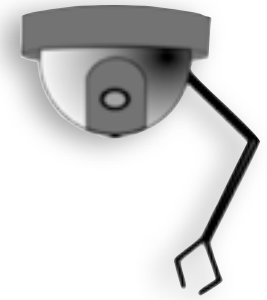
Blog: “A Beginner’s Guide to NVMe” http://sniaesfblog.org/?p=368

SCSI = Robot

- Your control program (in this example) is SCSI
- Does things one-at-a-time (serial commands)
- Built for the nature of rotational disks
- Works for non-volatile memory, like Flash and SSDs
- But is it the best way to do it?

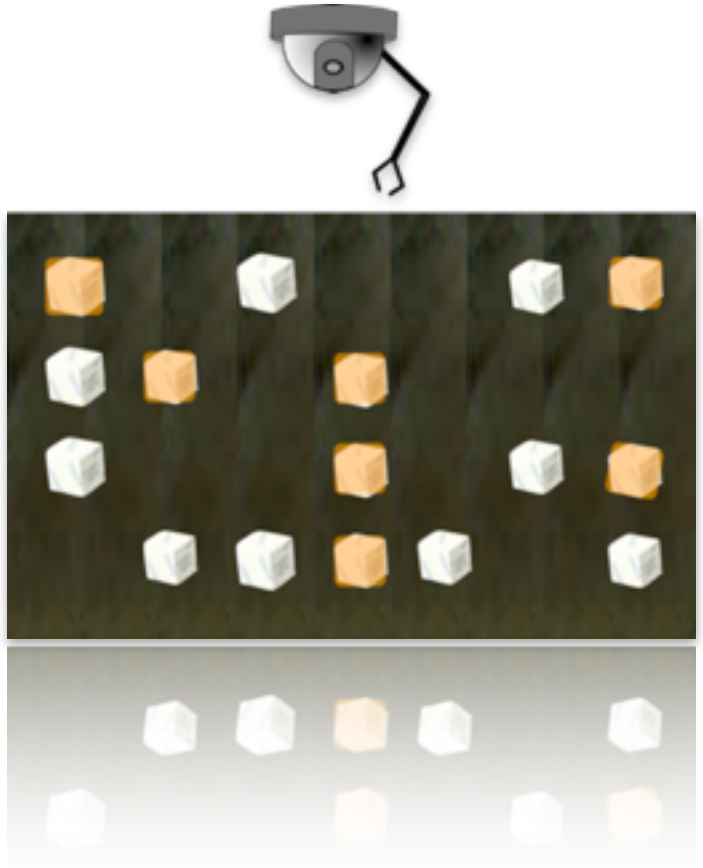
command: “get block off belt”

SCSI Commands



SCSI on Flash (NVM)

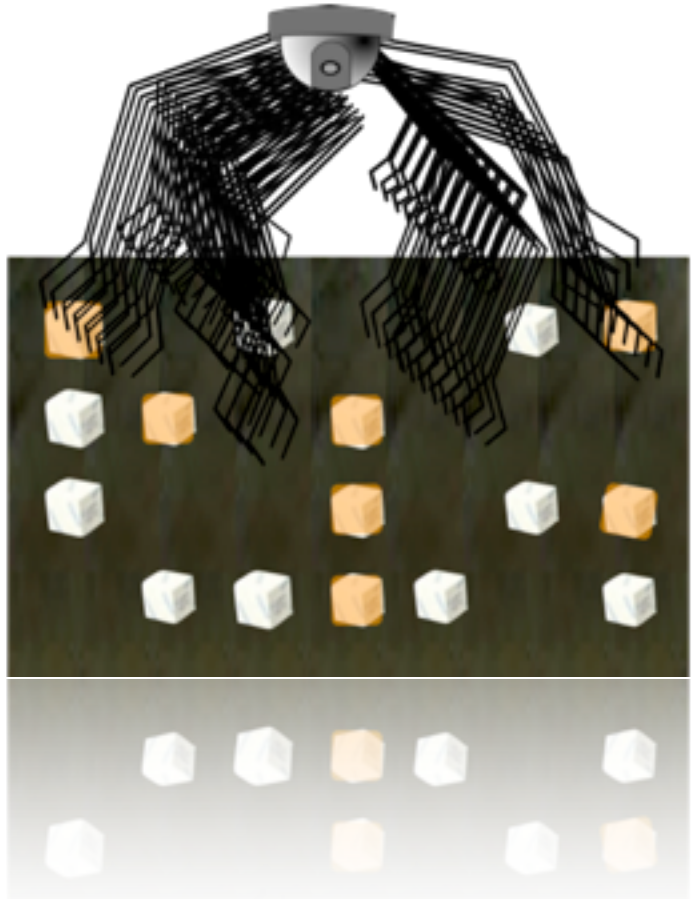
- Flash does away with the conveyor belts
  - No variance in media speed (-time)
  - All blocks are accessible (-time)
  - No need for data to be kept sequential (-time)
- But...
  - Must select blocks one at a time (one command <--> one queue)



Blog: “A Beginner’s Guide to NVMe” http://sniaesfblog.org/?p=368

NVMe on Flash (NVM)

- NVMe is like a robot with 64,000 arms (potentially)
  - Built to take advantage of the static nature of Non-Volatile Memory (including Flash)
  - Each arm can process up to 64k commands
    - NVMe command set is highly optimized (only 13 required)
- Realistic queue usage
  - 4 - to - 8 queues
  - Applications need to be written to take advantage of multi-queue access



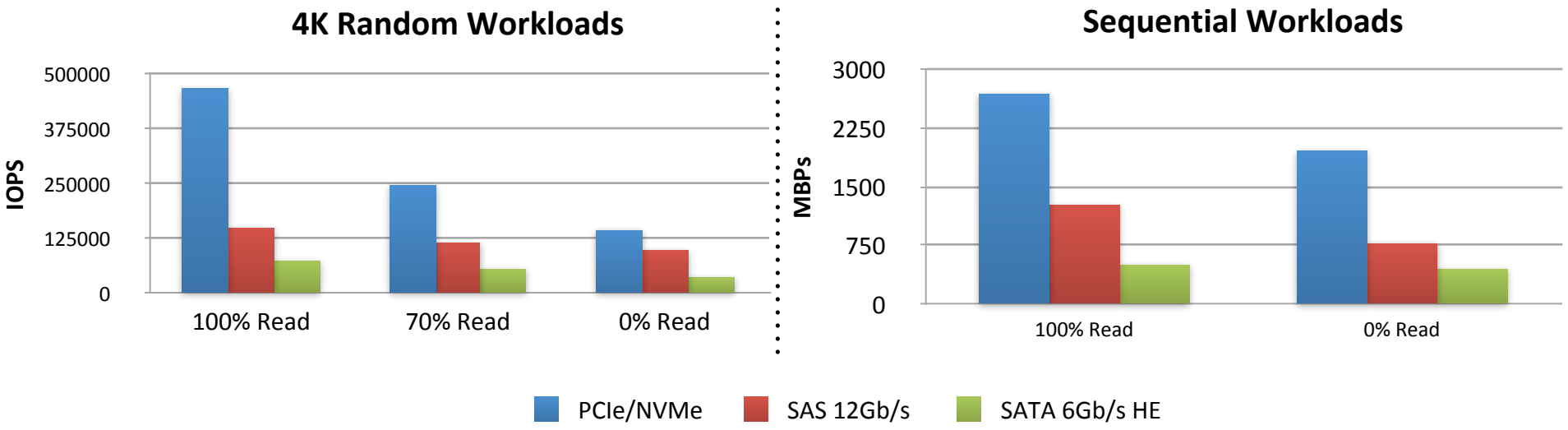
Blog: “A Beginner’s Guide to NVMe” http://sniaesfblog.org/?p=368

Consequence of Efficiency

IOPS and Sequential Performance

- 100% random reads: >3X better IOPs than SAS 12Gbps
- 70% random reads: >2X better IOPs than SAS 12Gbps
- 100% random writes:~ 1.5X better IOPs than SAS 12Gbps

- 100% reads:>2X better performance than SAS 12Gbps
- 100% writes:>2.5X better performance than SAS 12Gbps



Server Setup

- Basic 4U Intel® Xeon® E5 processor based server
- Out-of-the-box software setup
- Moderate workload: 8 workers, QD=4, random reads

Note: PCI Express* (PCIe*)/NVM Express* (NVMe) Measurements made on Intel® Core™ i7-3770S system @ 3.1GHz and 4GB Mem running Windows* Server 2012 Standard O/S, Intel PCIe/NVMe SSDs, data collected by IOmeter* tool. PCIe/NVMe SSD is under development. SAS Measurements from HGST Ultrastar* SSD800M/1000M (SAS) Solid State Drive Specification. SATA Measurements from Intel Solid State Drive DC P3700 Series Product Specification.

Analyzing What Matters

- Not *just* IOPS and/or bandwidth
  - Look at efficiency of software stack, latency, consistency

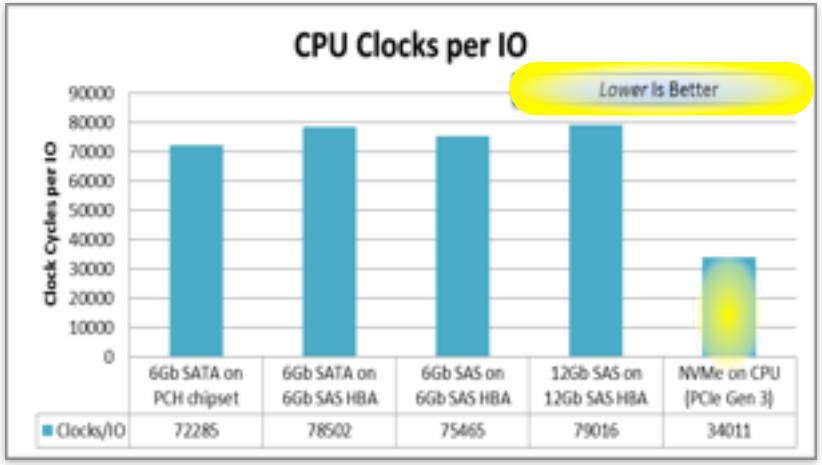
Storage Protocols Evaluated					
Interface	6Gb SATA	6Gb SATA	6Gb SAS	12Gb SAS	NVMe PCIe Gen 3
Attach Point	PCH chipset	6Gb SAS HBA	6Gb SAS HBA	12Gb SAS HBA	CPU

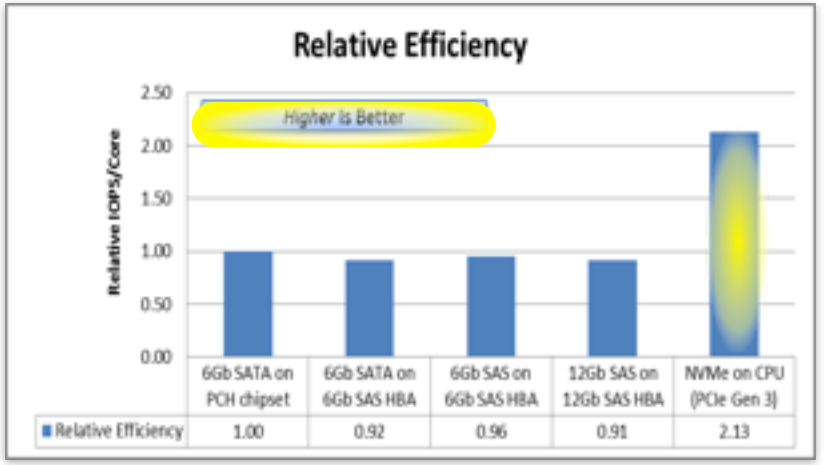
Not strenuous on purpose – evaluate protocol and not the server.

NVM Express* (NVMe)
PCI Express* (PCIe*)

Efficiency of NVMe

- CPU cycles in High Performance Computing are precious
- Each CPU cycle required for an IO adds latency
- NVM Express takes less than half the CPU cycles per IO as SAS

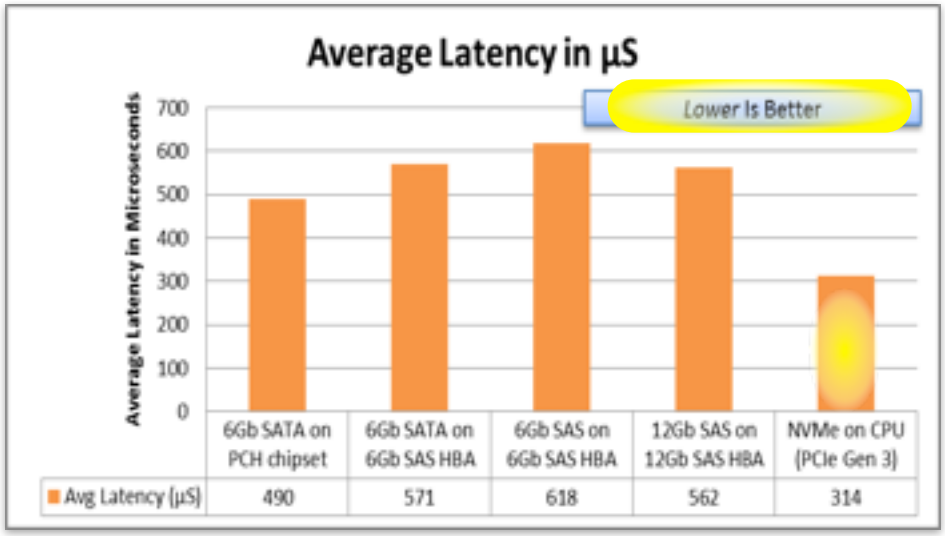


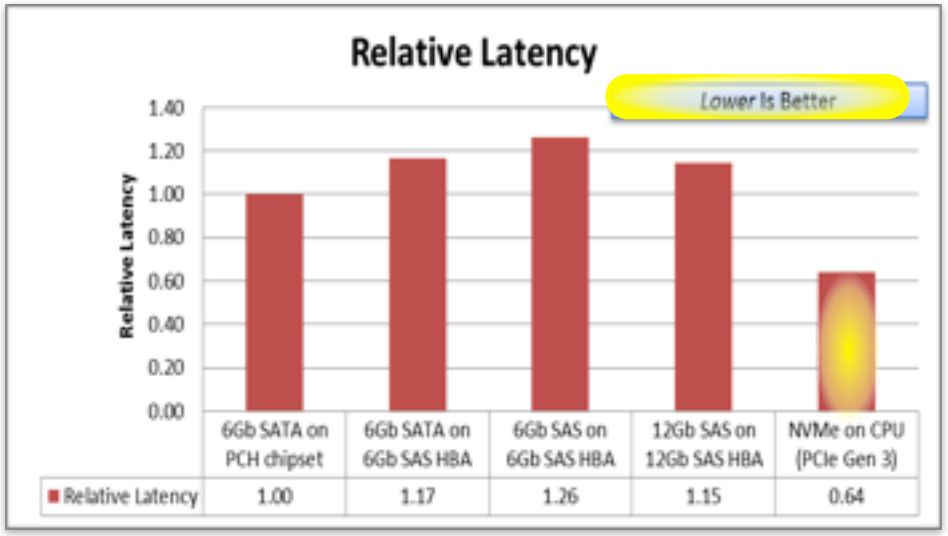


With equivalent CPU cycles, Intel SSD DC P3700 plus NVM Express delivers over 2X the IOPs of SAS!

Latency of NVMe

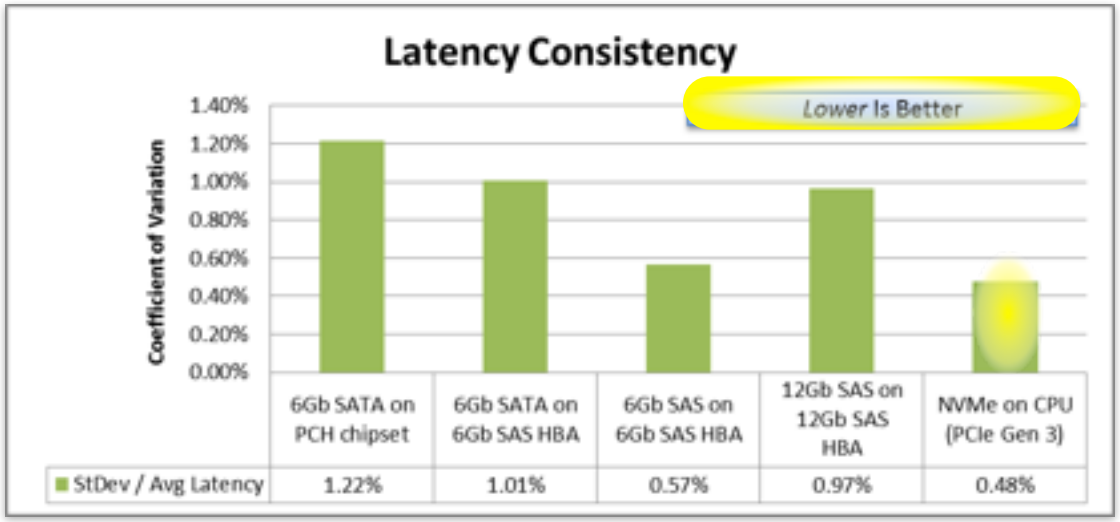
- The efficiency of NVM Express directly results in leadership latency
- When doubling from 6Gb to 12Gb, SAS only reduces latency by ~ 60 µS
- NVMe is more than 200 µs lower average latency than 12 Gb SAS





Consistency of NVMe

- NVM Express* (NVMe) leadership on latency and efficiency is consistently amazing
- SAS is a mature software stack with over a decade of tuning, yet the first generation NVM Express software stack has 2 to 3X better consistency

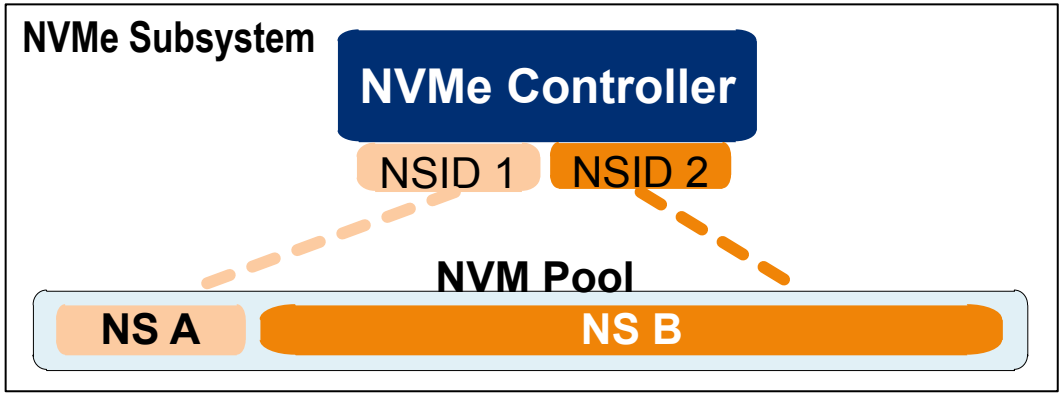


NVMe is already best in class, with more tuning yet to come.

NVMe - How it Works

Namespace Management

- A namespace is a region of NVM, made visible to applications as collection of logical blocks, which has defined Format, Features, PI, etc.
- Each namespace is independent of other namespaces in the subsystem.



This example:

OS sees two drives

- NS A = Disk 0
- NS B = Disk 1
- Logical partitions on A and B

- Today, creation/configuration of namespaces is vendor specific

OEM's and Customers wanted standard solution:
Configure any vendor's drive with same tool

NVMe Namespace Management - Command Set

- Namespace Management (new)
  - Create, Modify, or Delete namespaces
- Namespace Attachment (new)
  - Attach/Detach
  - Control visibility of namespaces by controllers and applications
- Identify Device (changes)
  - Enumerate namespaces, controllers, and attachment status of both, in subsystem

Create/Modify Namespace Properties

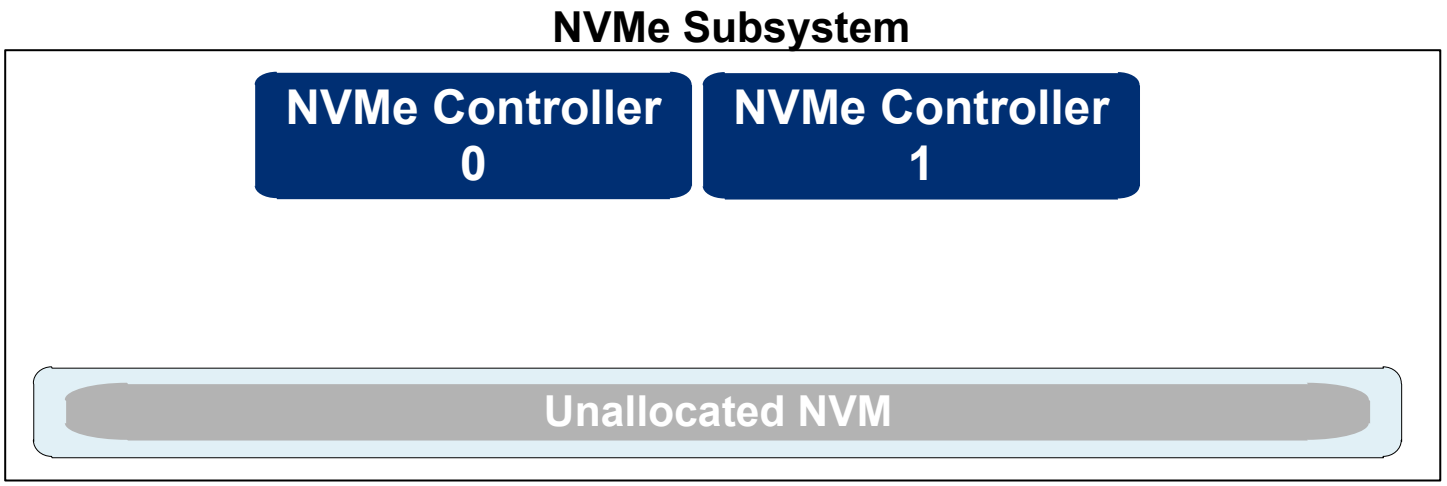
Namespace Size (NSZE)
Namespace Capacity (NCAP)
Formatted LBA Size (FLBAS)
Namespace Usage Hints (NUH)
End-to-end Data Protection Setting (DPS)
Multi-path I/O & Sharing Cap (NMIC)
End-to-end Data Protection Cap (DPC)
Per Namespace Atomicity Values
Namespace Utilization (NUSE)
Namespace Features (NSFEAT)
Number of LBA Formats (NLBAF)
Metadata Capabilities (MC)
Reservation Capabilities (RESCAP)
NVM Capacity (NVMCAP)
IEEE Extended Unique Identifier (EUI64)
LBA Format 0 Support (LBAF0)
LBA Format 1 Support (LBAF1)
…
LBA Format 15 Support (LBAF15)

Set by host software during Create or Modify

Controller generated or fixed subsystem value

NVMe Namespace Management Example - Before Configuration

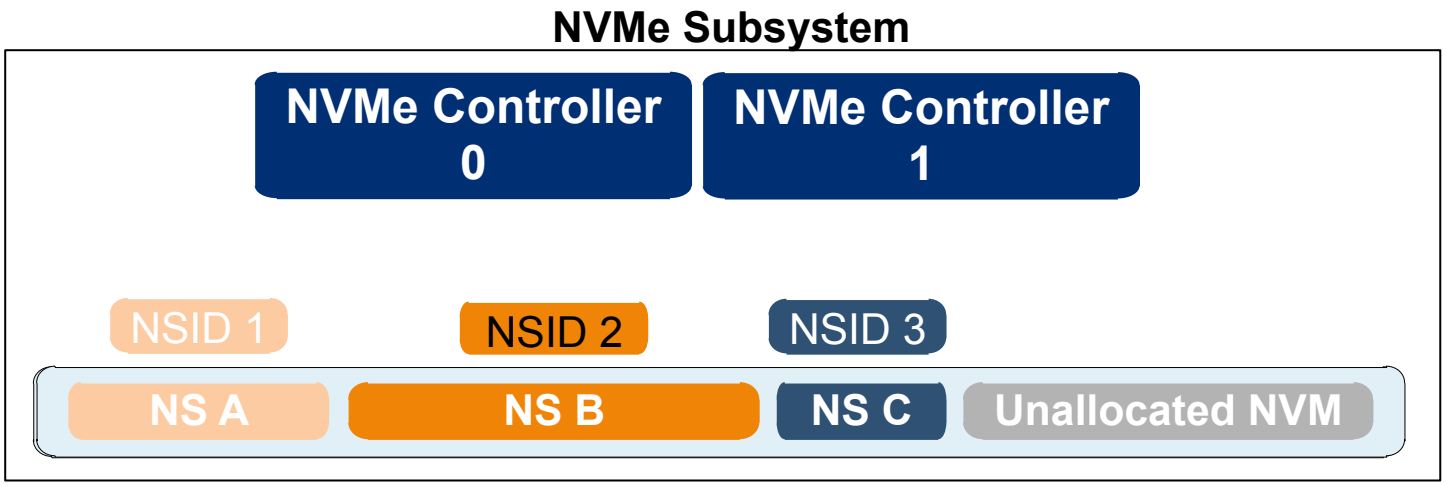
- No storage configured



Command	Issued to	After Create	Attach Status

NVMe Namespace Management Example - After Create

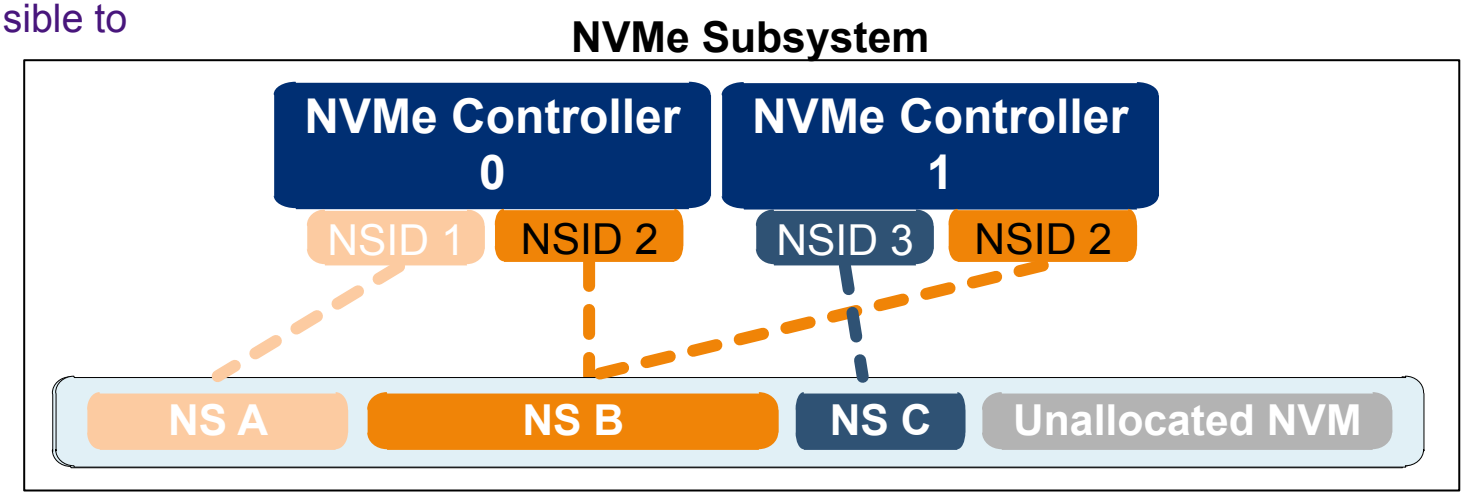
- Namespace structures created by controllers
- NSID's (handles) mapped to NS's
- NS's still not visible to apps



Command	Issued to	After Create	Attach Status
Create NS A (private, …)	0	NSID 1 ⇔ NS A	Not attached
Create NS B (shared, …)	1	NSID 2 ⇔ NS B	Not attached
Create NS C (private, …)	1	NSID 3 ⇔ NS C	Not attached

NVMe Namespace Management Example - After Attach

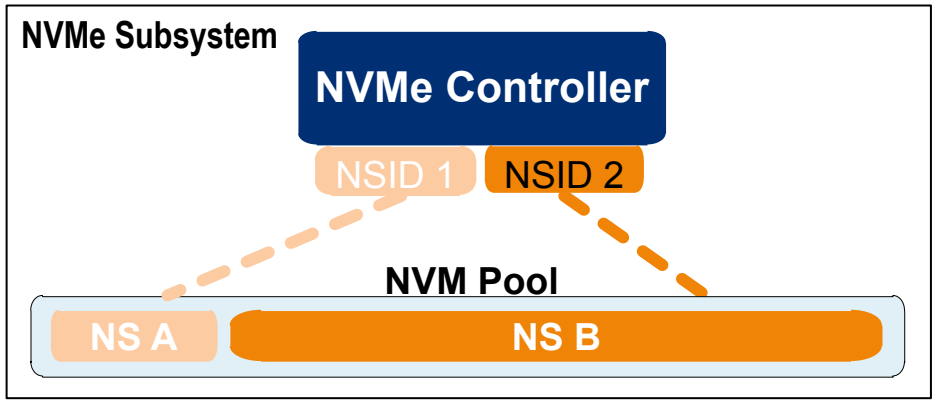
- Namespaces assigned to controllers and visible to applications



Command	Issued to	After Create	Attach Status
Create NS A (private, …)	0	NSID 1 ⇔ NS A	
Create NS B (shared, …)	1	NSID 2 ⇔ NS B	
Create NS C (private, …)	1	NSID 3 ⇔ NS C	
Attach (NS A, Controller 0)	0		NS A ⇔Controller 0
Attach (NS B, Controller 0, 1)	0		NS B ⇔Controller 0 and 1
Attach (NS C, Controller 1)	1		NS C ⇔Controller 1

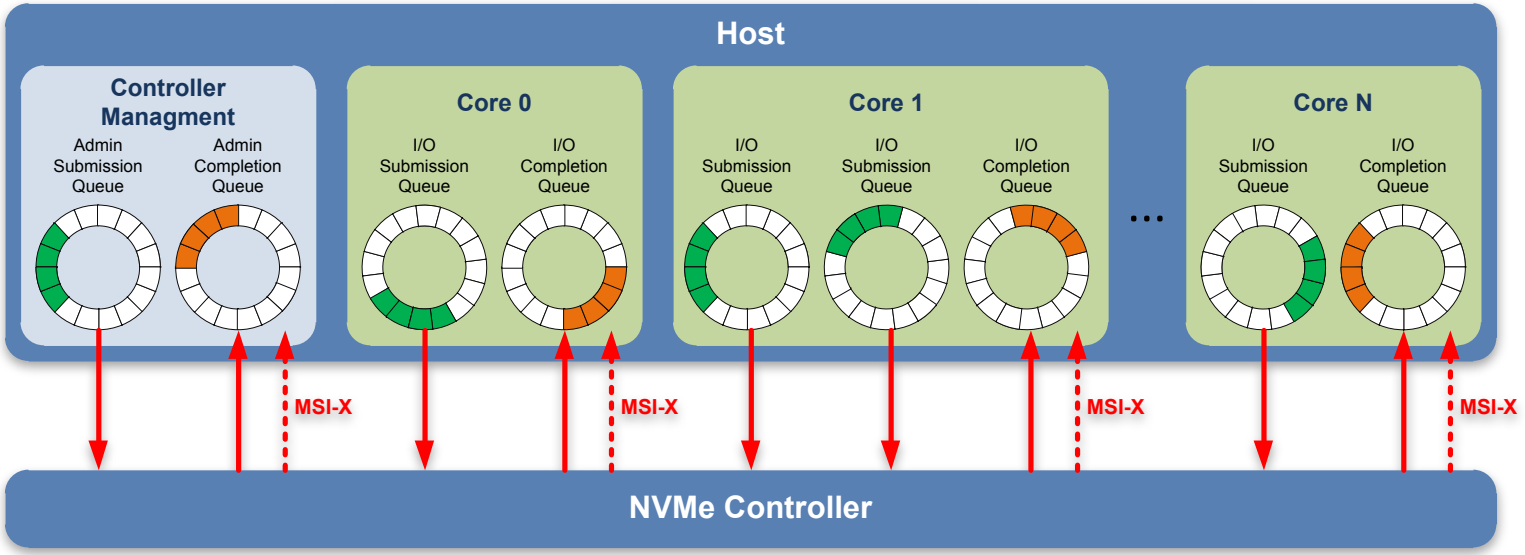
NVMe Namespace Management - Summary

- Namespace Management (new)
  - Create, Modify, or Delete namespaces
- Namespace Attachment (new)
  - Attach/Detach (i.e., control visibility of) namespaces
- Identify Device (changes)
  - Enumerate namespaces, controllers, and attachment status of both, in subsystem



Configure any vendor's NVMe drive with tools based on standard NVMe commands

NVMe Queues



- Multi Core Queuing Support
  - One or more I/O submission queues, completion queue, and MSI-X interrupt per core
  - High performance and low latency command issue
  - No locking between cores
- Up to ~2^32 outstanding commands
  - Support for up to **~64K** I/O submission and completion queues
  - Each queue supports up to **~64K** outstanding commands

Simplified Command Set

Admin Commands
Create I/O Submission Queue
Delete I/O Submission Queue
Create I/O Completion Queue
Delete I/O Completion Queue
Get Log Page
Identify
Abort
Set Features
Get Features
Asynchronous Event Request
Firmware Activate (optional)
Firmware Image Download (optional)

NVM Admin Commands
Format NVM (optional)
Security Send (optional)
Security Receive (optional)

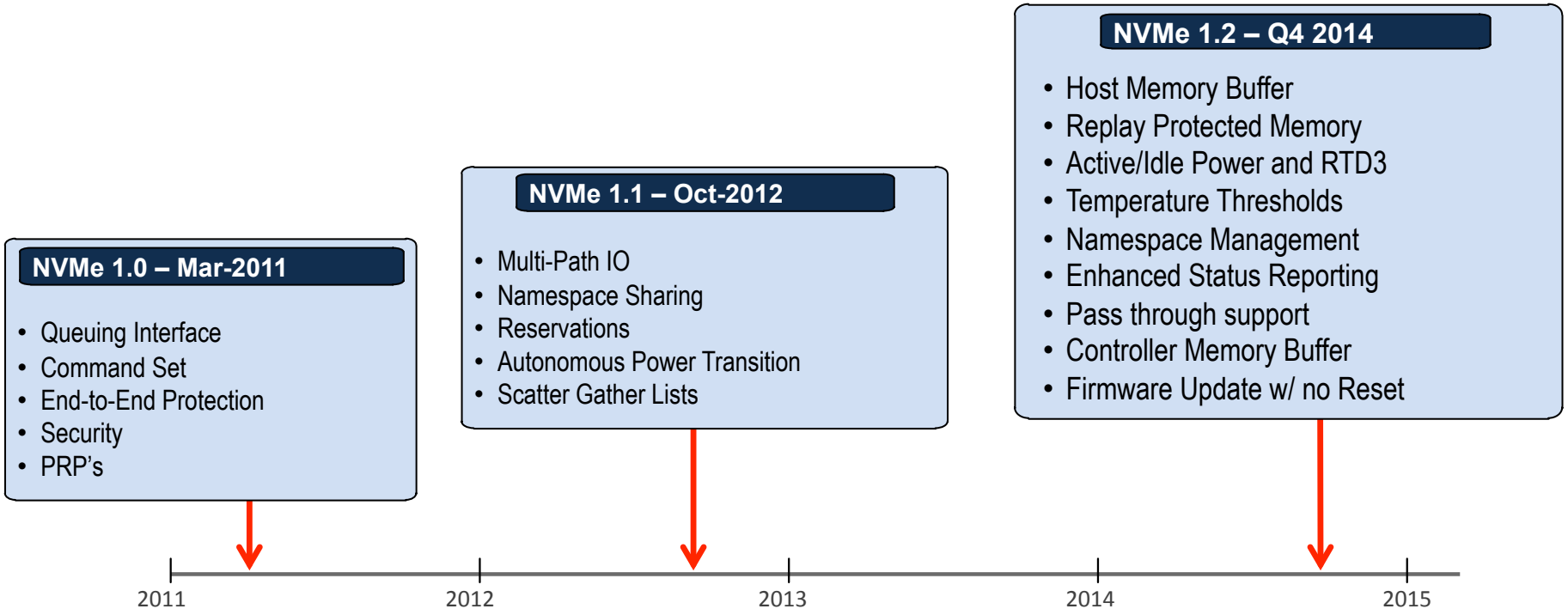
NVM I/O Commands
Read
Write
Flush
Write Uncorrectable (optional)
Compare (optional)
Dataset Management (optional)
Write Zeros (optional)
Reservation Register (optional)
Reservation Report (optional)
Reservation Acquire (optional)
Reservation Release (optional)

Only 10 admin commands and 3 I/O commands are required

Driver Development on Major OS'

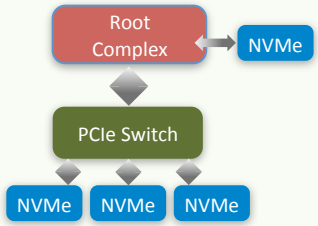
Windows*	• Windows* 8.1 and Windows* Server 2012 R2 include inbox driver • Open source driver in collaboration with OFA
Linux*	• Native OS driver since Linux* 3.3 (Jan 2012)
Unix	• FreeBSD driver released
Solaris*	• Delivered to S12 and S11 Update2 • Compliant with 1.0e
VMware*	• vmklinux driver certified targeted for Q2 '14 release
UEFI	• Open source driver available on SourceForge

NVMe Development Timeline



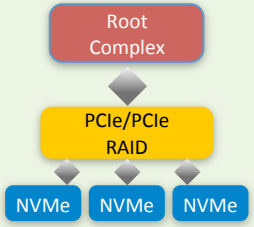
What's Next: NVMe over Fabrics

NVMe Usage Models



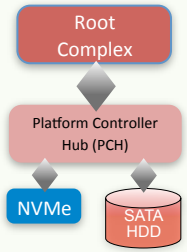
Server Caching

- Used for temporary data
- Non-redundant
- Used to reduce memory footprint



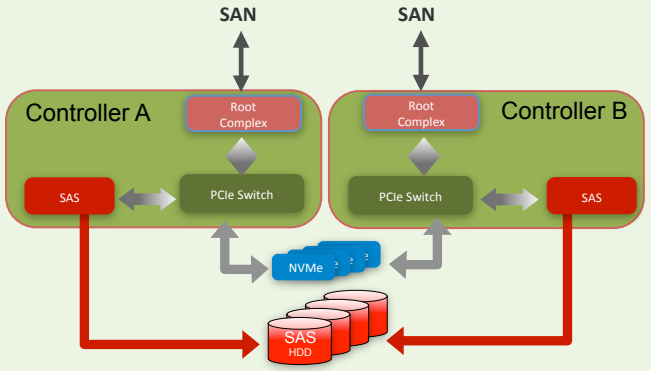
Server Storage

- Typically for persistent data
- Commonly used as Tier-0 storage
- Redundant (i.e. RAID'ed)



Client Storage

- Used for Boot/OS drive and/or HDD cache
- Non-redundant
- Power optimized



External Storage

- Used for just metadata or all data
- Multi-ported device
- Redundancy based on usage

Applicable for Enterprise, Data Center and Client

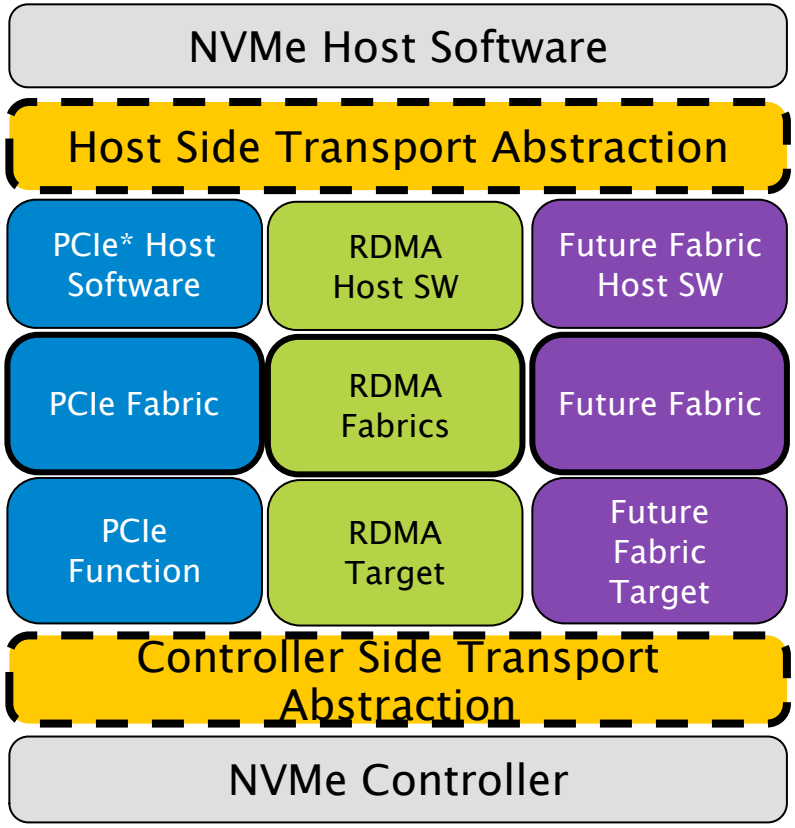
Why NVM Express over Fabrics?

- Simplicity, Efficiency and End-to-End NVM Express (NVMe) Model
- NVMe supports up to 64K I/O Queues with 3 required commands
  - Inherent parallelism of multiple I/O Queues is exposed
- Simplicity of protocol enables hardware automated I/O Queues – transport bridge
- No translation to or from another protocol like SCSI (in firmware/software)
- NVMe commands and structures are transferred end-to-end
- Maintains consistency between fabric types by standardizing a common abstraction

Goal: Make remote NVMe equivalent to local NVMe, within ~ 10 µs latency.

Architectural Approach

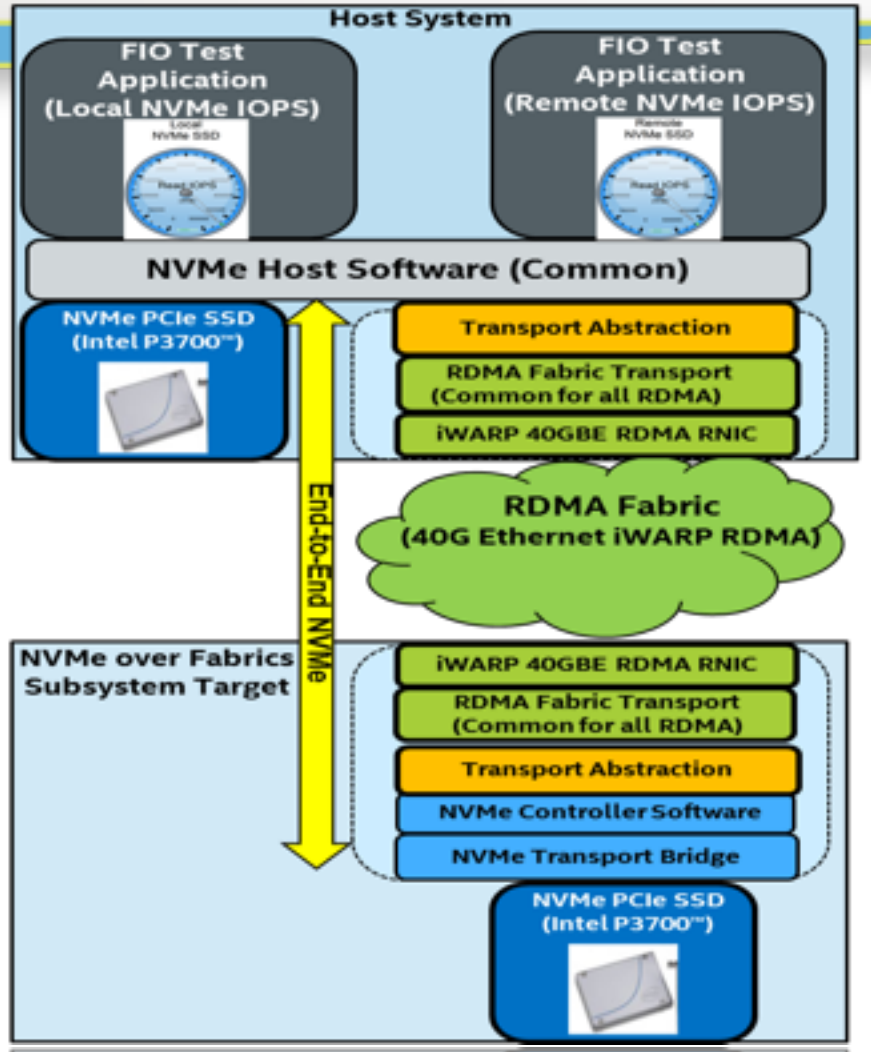
- The NVM Express (NVMe) Workgroup has started the definition of NVMe over Fabrics
- A flexible transport abstraction layer is under definition, enabling a consistent definition of NVMe over many different fabrics types
- The first fabric definition is the RDMA protocol family – used with Ethernet (iWARP and RoCE) and InfiniBand™
- Expect future fabric definitions; such as Fibre Channel/FCoE and Intel® Omni-Path fabrics



NVMe over Fabrics Prototype on iWARP

- Example shown in 2014 by Intel
- Recall: Goal is remote NVM Express (NVMe) equivalent
  - IOPS to local NVMe and no more than 10 µs added latency
- Prototype delivers 460K IOPs for both the local and remote PCIe NVMe SSD devices
- Remote NVMe adds 8 µs latency versus local NVMe access (4K Read & Write; QD=1)
- Demonstrates the efficiency of NVMe End-to-End;
  NVMe Target software running on one CPU core (two SMT threads) at 20% utilization

Remote storage equivalent to local storage, within ~ 10 µs latency



Source: Intel IDF, 2014

SNIA DSI DATA STORAGE INNOVATION CONFERENCE

Additional References

SNIA™

Additional References

- Official Website
  - **http://nvmexpress.org**
  - (Videos, specification docs, and more)
- Blogs
  - A Beginner's Guide to NVMe
    - http://sniaesfblog.org/?p=368
  - Introduction to NVMe Technology
    - https://www.osr.com/nt-insider/2014-issue4/introduction-nvme-technology
- Technical Working Groups
  - SNIA: http://www.snia.org/forums/sssi/nvmp
  - NVMExpress: http://www.nvmexpress.org/join-nvme

Note: Portions of this presentation are used with permission of the NVM Express group