NVM Express™: Unlock the Potential

Forum A-11

Audio-Visual Sponsor
The Maturity of NVM Express™

Tom Lenny
Principal Program Manager
Seagate Technology
- NVM Express is a scalable host controller interface designed for Enterprise and Client SSDs that utilize PCI Express® (PCIe®)
- NVM Express work group transitioned to an incorporated industry standards organization in March 2014
- More than 50 member companies
- Go to www.nvmexpress.org for information on how to join
PCIe SSDs taking top share in datacenter

PCIe will be leading interface for datacenter SSD by 2018

**Performance:** PCIe bandwidth is much greater than SAS and SATA, and NVMe™ protocol with new software stacks have far lower latency

**Industry Standards:** Transitioning from proprietary PCIe SSD hardware and software to standard NVMe™ software interface and standard 2.5” SFF 8639 form factor

---

PCle overtakes SATA in Client SSD

PCle performance and new standard M.2 form factor driving transition

Client SSD Unit Shipment Forecast by Interface

<table>
<thead>
<tr>
<th>Year</th>
<th>SATA</th>
<th>PCle</th>
</tr>
</thead>
<tbody>
<tr>
<td>2014</td>
<td>86%</td>
<td>14%</td>
</tr>
<tr>
<td>2015</td>
<td>74%</td>
<td>26%</td>
</tr>
<tr>
<td>2016</td>
<td>58%</td>
<td>42%</td>
</tr>
<tr>
<td>2017</td>
<td>40%</td>
<td>60%</td>
</tr>
<tr>
<td>2018</td>
<td>25%</td>
<td>75%</td>
</tr>
</tbody>
</table>

Source: Forward Insights

Interface Performance

- SATA 6Gb/s: 0.6 G B/s
- PCIe 3.0 x4: 4 G B/s

M.2 Form Factor

- 42mm
- 60mm
- 80mm
- 110mm
NVMe™ Interoperability

• NVMe Organization along with UNH-IOL continue to develop new compliance and interoperability tests to drive for mature production devices with plug and play experience

• 1st Plugfest, May 2013, had 11 participant companies
• 2nd Plugfest, February 2014, had 14 companies
• 3rd Plugfest scheduled for November 10-13, 2014 at UNH-IOL

• Register for plugfest:
  www.iol.unh.edu/services/testing/NVMe/plugfest.php
1st NVM Express* Products

Intel PCIe with NMVe SSDs Announced

Intel has announced a new line of powerful SSD that combine PCI Express with NVM Express (NVMe) technology for substantial performance gains over SATA SSDs and traditional hard disk drives. The Intel SSD Data Center Family for PCIe are designed to meet the increasing needs for high-performance, consistent, and reliable storage solutions in the data center, all the while helping to lower total cost of ownership. The new line includes the Intel SSD DC P3700 Series for write-intensive applications, the Intel SSD DC P3600 Series for mixed workloads, and the Intel SSD DC P3500 Series for read-heavy uses.

“…our first look at an NVMe drive, and the results are impressive.” AnandTech

Samsung Announces Industry’s First 2.5-Inch NVMe SSD

Samsung has announced the XS1715, a 2.5-inch Non-Volatile Memory Express (NVMe Express) PCIe SSD. According to Samsung, the 1.6TB SPY-832 NVMe SSD provides a sequential read speed at 3,000MB/s, six times faster than the company’s current high-end enterprise SSD. The XS1715’s random read performance is specified at up to 740,000 IOPS, more than 19 times as fast as existing SSD options.

“The NVMe interface can provide up to six times the throughput of the SATA 6Gb/s interface.” The SSD Review
Track Agenda

- NVMe Ecosystem Development – Amber Huffman (Intel)
- NVMe 1.2 Feature Overview – Dave Landsman (Sandisk)
- NVMe Management Interface – Peter Onufryk (PMC-Sierra)
- An Implementer’s Perspective – Ken Okin (HGST)

**Afternoon session A12 3:15 - 4:25:**

- Scaling from the Datacenter to the Client – KeunSoo Jo (Samsung)
- NVMe in the real world – Dave Allen (PMC-Sierra)
NVM Express Ecosystem Development

Amber Huffman
Sr. Principal Engineer
Intel Corporation
Outline

- Ecosystem Embrace of PCIe* Storage
- NVM Express Overview
- Features for Client and Datacenter
- Native Driver Support for NVMe™
- Form Factors and Connectors
- Using Hot Plug with NVMe
- Interoperability Program for NVMe
PCI Express* (PCIe) is scalable enabling an OEM to select the right performance point for an individual drive.

Do I need a 4 GB/s device for my application?

...Or an 8 GB/s device?


*Other names and brands may be claimed as the property of others.
PCIe* SSD Benefits

- Performance: 1 GB/s per lane.. 4 GB/s, 8 GB/s, 16 GB/s per device..
- Lower latency: Direct CPU connection
- No host bus adapter (HBA): Lower power ~ 10W and cost ~ $15
- Increased I/O opportunity: Up to 40 PCIe lanes per CPU socket
- Form factor options: PCIe add-in-card, SFF-8639, M.2, SATA Express, BGA
NVM Express Overview

- NVM Express (NVMe) is a standardized high performance host controller interface for PCIe Storage, such as PCIe SSDs
  - Standardizes register set, feature set, and command set where there were only proprietary PCIe solutions before
  - Architected from the ground up for NAND and next generation NVM
  - Designed to scale from Enterprise to Client systems

- NVMe was developed by an open industry consortium and is directed by a 13 company Promoter Group
Technical Basics of NVMe

- All parameters for 4KB command in single 64B command
- Supports deep queues (64K commands per queue, up to 64K queues)
- Supports MSI-X and interrupt steering
- Streamlined & simple command set (13 required commands)
- Optional features to address target segment (Client, Enterprise, etc.)
  - Enterprise: End-to-end data protection, reservations, etc.
  - Client: Autonomous power state transitions, etc.
- Designed to scale for next generation NVM, agnostic to NVM type used
NVMe Delivers on Performance

- NVM Express delivers versus leadership SAS & SATA products

- Random Workloads
  - > 2X performance of SAS 12Gbps
  - 4-6X performance of SATA 6Gbps

- Sequential Workloads
  - Realize almost 3 GB/s reads
  - > 2X performance of SAS 12Gbps
  - > 4X performance of SATA 6Gbps

Note: PCIe/NVMe Measurements made on Intel(R)Core(TM) 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.

* Other names and brands are property of their respective owners
Real Workload Benefit of NVMe

Oracle TimesTen In-Memory Database Restart

Oracle TimesTen database checkpoint file loaded to memory

Intel® SSD DC P3700 Series with NVMe*
NVMe Scales from Client to Datacenter

- NVMe allows devices to flexibly report capabilities
  - Number of queues, number of power states, etc

- NVMe features may be focused on Datacenter, on Client, or both

How do you determine the NVMe features needed for your use case?

Different features, form factor, capacity, etc, based on market segment targeted
## Current Feature Guidance

<table>
<thead>
<tr>
<th>Feature</th>
<th>Datacenter Guidance</th>
<th>Client Guidance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Queues</td>
<td>16 to 128</td>
<td>2 to 8</td>
</tr>
<tr>
<td>Number of Namespaces</td>
<td>1 to 16 (use case dependent)</td>
<td>1 to 4 (use case dependent)</td>
</tr>
<tr>
<td>Number of Power States</td>
<td>1 to 4</td>
<td>3 to 4</td>
</tr>
<tr>
<td>Arbitration Mechanism</td>
<td>Weighted Round Robin with Urgent Priority Class or Round Robin</td>
<td>Round Robin</td>
</tr>
<tr>
<td>End-to-end Data Protection</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Security Tunneling</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Autonomous Power State Transitions</td>
<td>For small form factors</td>
<td>Yes</td>
</tr>
<tr>
<td>Scatter/Gather List</td>
<td>Yes for Storage Customers</td>
<td>No</td>
</tr>
<tr>
<td>Reservations (and Dual Port)</td>
<td>Yes for Storage Customers</td>
<td>No</td>
</tr>
</tbody>
</table>
## Revision 1.2 Feature Guidance

<table>
<thead>
<tr>
<th>Feature</th>
<th>Datacenter Guidance</th>
<th>Client Guidance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namespace Management</td>
<td>Yes (if multiple supported)</td>
<td>Yes (if multiple supported)</td>
</tr>
<tr>
<td>Namespace Inventory Change Notices</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Temperature Threshold</td>
<td>Yes</td>
<td>Minimal support (e.g., 1 sensor)</td>
</tr>
<tr>
<td>Power State Enhancements and Runtime D3 Entry/Exit</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Host Memory Buffer</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Controller Memory Buffer</td>
<td>Yes for Storage Customers</td>
<td>No</td>
</tr>
<tr>
<td>Replay Protected Memory Block</td>
<td>No</td>
<td>Yes for mobile oriented SKU</td>
</tr>
<tr>
<td>Command Effects Log for NVMe Passthrough Support</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
**Driver Ecosystem Flourishing**

<table>
<thead>
<tr>
<th>Operating System</th>
<th>Key Features</th>
</tr>
</thead>
</table>
| Windows*         | - Windows 8.1 and Windows Server 2012 R2 include native driver  
                  - Open source driver in collaboration with OFA |
| Linux*           | - Stable OS driver since Linux kernel 3.12 |
| Unix             | - FreeBSD driver upstream |
| Solaris*         | - Solaris driver will ship in S12 |
| VMware*          | - Open source vmklinux driver available on SourceForge |
| UEFI             | - Open source driver available on SourceForge |

Native OS drivers already available, with more coming.
Serviceable Form Factor for Datacenter

- A serviceable (hot pluggable) form factor is critical in Datacenter

- The SFF-8639 form factor / connector supports NVMe, SAS, and SATA
  - Enables OEMs to transition at their own speed

- SFF-8639 can be used with existing platforms using a PCIe* adapter
Client Form Factors

- There are a variety of form factors depending on need
- M.2 is an optimized SSD only form factor for laptops
- The SATA Express connector supports 2.5” form factor drives when allowing for HDD/SSHD is needed
- Intel, SanDisk, Toshiba, and HP proposed a BGA solution for standardization in PCI SIG for behind-the-glass usages (e.g., 2-in-1 laptops) – join PCI SIG to participate!

Form factors are there to support NVMe adoption in client in 2015.

*Other names and brands may be claimed as the property of others.
Hot Plug: Why PCIe* is Different

- For SATA & SAS, the host storage controller and driver manage hot plug
- With PCIe, the storage controller moves into the SSD
- Challenge: There is no storage driver present to handle events for hot plug

Hot plug needs support of the PCI bus driver, BIOS, and host software.
Surprise removal is not normal and not recommended!

- Device removal with I/O in flight invites data loss; not recommended for SAS either.
- However, a random device failure or operator error may look like surprise removal.

Storage controller driver must check for Master Abort

- On all reads to the device, the driver checks register for FFFF_FFFFh.
- If data is FFFF_FFFFh, then driver reads another register expected to have a value that includes zeroes to verify device is still present.

Time order of removal notification is unknown

- e.g. Storage controller driver via Master Abort, or PCI Bus driver via Presence Change interrupt, or RAID software may signal removal first.

Software hardening for surprise removal is under development.
Normal Hot Plug Use Cases

- Hot Add & Remove are software managed events
  - The operator takes explicit action, including software configuration

- During boot, the system must prepare for hot plug:
  - Configure PCI Express* Slot Capability registers
  - Enable/register for hot plug events to higher level software (e.g., RAID)
  - Pre-allocate slot resources (Bus IDs, interrupts, memory regions) using ACPI*

- No updates to host hardware (CPU/chipset) is needed

Existing BIOS and Windows*/Linux* OS are prepared to support PCIe hot plug today.
The University of New Hampshire Interoperability Lab (UNH-IOL) has collaborated with NVMe to deliver a robust interop program.

Two plugfests have been held in the past year, with another slated for the end of this year.

Seven devices are already on the NVMe Integrator's List.

More details at www.iol.unh.edu/services/testing/NVMe.
Conformance and Interoperability

- NVMe Interop program tests interoperability and conformance
  - Interoperability: Ensure the device interoperates with many hosts
  - Conformance: Ensure the device complies to the specification

- Interoperability: Test against five different hosts – must pass with at least four including both a Windows and Linux reference driver

- Conformance: Test tools used to check proper response to different NVMe host requests

Comprehensive testing ensures high quality NVMe ecosystem.
Summary

- NVM Express is the interface architected from the ground up for non-volatile memory

- NVMe performance meets the needs of demanding applications

- A robust NVMe ecosystem is developing, including native OS drivers, form factors, and an interoperability program
  - Remember plugfest #3 the week of November 10

Join NVMe. Learn more at nvmexpress.org.
What’s New in NVMe™ Rev 1.2?

Dave Landsman
Director, Standards and Industry Associations
SanDisk
NVMe Development Timeline

**NVMe 1.0 – Mar-2011**
- Queuing Interface
- Command Set
- End-to-End Protection
- Security
- PRP’s

**NVMe 1.1 – Oct-2012**
- Multi-Path IO
- Namespace Sharing
- Reservations
- Autonomous Power Transition
- Scatter Gather Lists

**NVMe 1.2 – Q4 2014**
- Host Memory Buffer
- Replay Protected Memory
- Active/Idle Power and RTD3
- Temperature Thresholds
- Namespace Management
- Enhanced Status Reporting
- Pass through support
- Controller Memory Buffer
- Firmware Update w/ no Reset

Flash Memory Summit 2014
Santa Clara, CA
NVMe Evolving for Client

- Client has become Mobile
  - Push for lower cost, power, size

- Storage occupying larger % of area in ultra-thin, 2-in-1’s, tablets

- NVMe Rev 1.2 adding support for cost/power sensitive SFF
  - Host Memory Buffer
  - Replay Protected Memory Block
  - Power Management Enhancements

Storage (22x80 M.2) is ~ 15% of area

Source: Intel
Host Memory Buffer (HMB)

- Allow the host driver to allocate system memory for the SSD’s exclusive use

- Enables DRAM savings
  - E.g., store translation tables in host DRAM

- Spec
  - Host can enable/disable feature, preserve/reassign buffer after RTD3
  - Device can specify minimum and preferred buffer size
Replay Protected Memory Block (RPMB)

- Allow host to store data to authenticated and replay protected area of memory
  - e.g. WWAN provisioning

**RPMB Layout (up to 7)**

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
</table>
| **Authentication Key** | • Write Once, no Erase or Read  
                      | • Used to generate MAC |
| **Write Counter**     | • Read Only, cannot be reset  
                      | • # of authenticated write requests |
| **Data Area**         | • Read/Write, no Erase  
                      | • May be read and written after authentication |
RTD3 Entry/Exit Latency

Enable host to be more aggressive using RTD3

- In NVMe Rev 1.1, RTD3 is a bit of a guessing game
  - Driver issues shutdown intent, polls till device says ready
  - Host does not know entry/exit latency to calculate ROI of device shutdown

- NVMe Rev 1.2 adds RTD3 entry and exit latency reporting
  - Knowing entry time reduces polling effort
  - Knowing entry + exit time enables better host policy
Active and Idle Power

Example Device State Definitions

<table>
<thead>
<tr>
<th>Power Descriptor Field</th>
<th>State 1</th>
<th>State 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Max Instantaneous</td>
<td>X</td>
<td>Y</td>
</tr>
</tbody>
</table>

NVMe Rev 1.1 Power Descriptors

<table>
<thead>
<tr>
<th>Max Instantaneous</th>
<th>Operational State</th>
</tr>
</thead>
<tbody>
<tr>
<td>Relative Read Throughput</td>
<td>Relative Write Throughput</td>
</tr>
<tr>
<td>Entry Latency</td>
<td>Exit Latency</td>
</tr>
<tr>
<td>Relative Read Latency</td>
<td>Relative Write Latency</td>
</tr>
</tbody>
</table>

- New power-related fields enable better host policy

New NVMe Rev 1.2 Descriptors
- "In-state" power, with workload assumptions
- Example Host Policy
  - State 1 for “Instant Go” workload
  - State 2 for “Compute” workload
Temperature Management - Threshold Model

- Set alert for over temp
- If exceeded, poll temp till below critical level

NVMe Rev 1.1
Critical Temp and Over Alert

NVMe Rev 1.2
Add Warning Temp & Under Alert

- In region A, set Over Alert to Warning
- If warning temp exceeded (now in B)
  - Assign Over Alert to Critical
  - Assign Under Alert to Warning
  - Begin cooling measures
- If Under Alert triggered (back in A)
  - Cooling measures worked
  - Re-assign Over and Under alerts
- If Over Alert triggered
  - Cooling did not work
  - Shut down or other measures
Temperature Management
- Additional Temp Sensors

- NVMe Rev 1.1
  - Single overall device/subsystem Temperature (K°)

- NVMe Rev 1.2
  - Keep original abstract temperature
    - Renamed “Composite Temperature”
  - Add Temperature Sensors 1-8
    - Actual temperature (K°)
  - Platform can use Sensors 1-8 for overall system temperature control
    - E.g., Composite Temp may be ok for NVMe device, but laptop case getting hot
Namespace Management

- A namespace is a region of NVM, made visible to applications as a 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

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

<table>
<thead>
<tr>
<th>Created/Modify Namespace Properties Set by host software during Create or Modify Controller generated or fixed subsystem value</th>
</tr>
</thead>
</table>
NVMe Namespace Management Example - Before Configuration

- No storage configured

<table>
<thead>
<tr>
<th>Command</th>
<th>Issued to Controller</th>
<th>After Create</th>
<th>Attach Status</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- NVMe Controller 0
- NVMe Controller 1

Unallocated NVM
- 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 Controller** | **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

<table>
<thead>
<tr>
<th>Command</th>
<th>Issued to Controller</th>
<th>After Create</th>
<th>Attach Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create NS A (private, ...)</td>
<td>0</td>
<td>NSID 1 ↔ NS A</td>
<td>NS A ↔ Controller 0</td>
</tr>
<tr>
<td>Create NS B (shared, ...)</td>
<td>1</td>
<td>NSID 2 ↔ NS B</td>
<td>NS B ↔ Controller 0 and 1</td>
</tr>
<tr>
<td>Create NS C (private, ...)</td>
<td>1</td>
<td>NSID 3 ↔ NS C</td>
<td>NS C ↔ Controller 1</td>
</tr>
<tr>
<td>Attach (NS A, Controller 0)</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attach (NS B, Controller 0, 1)</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Attach (NS C, Controller 1)</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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
As NVMe gets deployed in larger systems, e.g. PCIe JBODs with PCIe switches for scale, Thunderbolt and other interbox PCIe, etc., fabric latency and variability can increase.

1. Command written to host memory
2. Host PCIe Write to SQ doorbell
3. Controller PCIe Read to fetch command
4. Command data returned from host
5. Data + Completion

PCIe read/write ordering rules affect latency as more read/write cross-traffic.

PCIe reads take resources in switches, affecting performance.
Controller Memory Buffer

- Enable host dedicated access to a region in controller memory space for commands and data
- Revised sequence:
  1. Command/data written to controller memory + Host PCIe Write to SQ doorbell
  2. Controller PCIe Read to fetch SQ entry
  3. SQ Returned from host entry
  4. Data + Completion
- Reduces latency/variability
- Increase performance
Firmware Update w/o Reset

- In NVMe Rev 1.1 firmware update requires reset
  - Queues/Commands lost

- In NVMe Rev 1.2
  - "No Reset" option added to Firmware Commit

Firmware Commit in Rev 1.2

Firmware Slots: 0 1 2 3 4 5 6 7

- Successful
  - Running image from Slot 2
- Not successful
  - Image applied at next reset
- Not successful
  - Max time exceeded; command must be re-issued
Other NVMe Rev 1.2 Features

- Atomicity enhancements
- Command Effects Log (pass-through)
- Enhanced Status Reporting
NVMe Expanding, Maturing

- **NVM Express**
  - Architected for NVM
  - Simple Command Set
  - Low Latency
  - Exploits Parallelism

- **Rev 1.2 continues the growth**
  - Client
  - Enterprise
  - General Usability

---

**NVMe 1.0 – Mar-2011**
- Queuing Interface
- Command Set
- End-to-End Protection
- Security
- PRP's

**NVMe 1.1 – Oct-2012**
- Multi-Path IO
- Namespace Sharing
- Reservations
- Autonomous Power Transition
- Scatter Gather Lists

**NVMe 1.2 – Q4 2014**
- Host Memory Buffer
- Replay Protected Memory
- Active/Idle Power and RTD3
- Temperature Thresholds
- Namespace Management
- Enhanced Status Reporting
- Pass through support
- Controller Memory Buffer
- Firmware Update w/ no Reset
Thank You!
NVM Express Management Interface

Peter Onufryk
Sr. Director Product Development
PMC Sierra

Austin Bolen
Storage Development Principal Engineer
Dell

Audio-Visual Sponsor
Shannon Systems
NVMe Storage Device Management

- Example Pre-boot Management
  - Inventory, Power Budgeting, Configuration, Firmware Update
- Example Out-of-Band Management During System Operation
  - Health Monitoring, Power/Thermal Management, Firmware Update, Configuration
Why Standardize NVMe Storage Device Management?

- **Reduces Cost and Broadens Adoption**
  - Allows OEMs to source storage devices from multiple suppliers
  - Eliminates need for NVMe storage device suppliers to develop custom OEM specific management features

- **Consistent Feature Set**
  - All storage devices that implement management implement a common baseline feature set
  - Optional features are implemented in a consistent manner

- **Industry Ecosystem**
  - Compliance tests / program
  - Development tools
NVM Express Organization

**NVMe Promoters**

“Board of Directors”

**Technical Workgroup**
- Queuing Interface
- Admin Command Set
- NVMe I/O Command Set
- Driver Based Management

**Management Interface Workgroup**
- Out-of-Band Management over PCIe VDM and SMBus
NVMe Architecture (review)

- **NVM Subsystem** - one or more controllers, one or more namespaces, one or more PCI Express ports, a non-volatile memory storage medium, and an interface between the controller(s) and non-volatile memory storage medium.

NVM Subsystem with Two Controllers and One Port

NVM Subsystem with Two Controllers and Two Ports
An NVMe Storage Device consists of one NVM Subsystem with
- One or more PCIe ports
- An optional SMBus/I2C interface
Driver vs. Out-of-Band Management

Diagram showing the interaction between different components in a system, including NVMe drivers, operating system, platform management, and NVM subsystem, with connections via PCIe, SMBus/I2C, and PCIe VDM.
Management Interface Workgroup

Mission

- Create, define, and specify out-of-band management that is independent of the physical transport and its associated protocol.

- Define mapping of the management interface to one or more out-of-band physical interfaces.

- Create, define and specify a management command set for NVMe devices.

- Provide feedback and recommendations into the Technical Workgroup on command and interface behaviors required for management.
Key Management Interface Tenets

- Works across operational times
  - Deployment (No OS)
  - Pre-OS (e.g., UEFI/BIOS)
  - Runtime
  - Aux Power (VPD/FRU)
  - Decommissioning
- Does not create new security holes
- Supports robust error reporting and recovery
- No host or OS dependencies (e.g., out-of-band management)
- Form factor agnostic
- Comprehends surprise add/remove (hot-plug)
Management Interface Protocol Layering

- Management Applications (e.g., Remote Console)
- Management Controller (BMC or Host Processor)
- NVMe Management Interface
- Management Component Transport Protocol (MCTP)
- MCTP over SMBus/I2C Binding
- MCTP over PCIe VDM Binding
- SMBus/I2C
- PCIe
- PCIe SSD
# NVMe Management Interface Command Set Overview (preliminary)

<table>
<thead>
<tr>
<th>Command Type</th>
<th>Command</th>
<th>Command Type</th>
<th>Command</th>
</tr>
</thead>
<tbody>
<tr>
<td>NVMe Management Interface Specific Commands</td>
<td>Controller Inventory</td>
<td>NVMe Commands</td>
<td>Get Log Page</td>
</tr>
<tr>
<td></td>
<td>Read / Write VPD</td>
<td></td>
<td>Identify</td>
</tr>
<tr>
<td></td>
<td>Run Diagnostics</td>
<td></td>
<td>Set Feature</td>
</tr>
<tr>
<td></td>
<td>Health Status</td>
<td></td>
<td>Get Feature</td>
</tr>
<tr>
<td></td>
<td>Command Flow Control</td>
<td></td>
<td>Firmware Activate</td>
</tr>
<tr>
<td></td>
<td>Exception Handling</td>
<td></td>
<td>Firmware Image Download</td>
</tr>
<tr>
<td>PCIe Command</td>
<td>Configuration Read</td>
<td></td>
<td>Vendor Specific</td>
</tr>
<tr>
<td></td>
<td>Configuration write</td>
<td></td>
<td>Format NVM</td>
</tr>
<tr>
<td></td>
<td>I/O Read</td>
<td></td>
<td>Security Send</td>
</tr>
<tr>
<td></td>
<td>I/O Write</td>
<td></td>
<td>Security Receive</td>
</tr>
<tr>
<td></td>
<td>Memory Read</td>
<td></td>
<td>…</td>
</tr>
<tr>
<td></td>
<td>Memory Write</td>
<td></td>
<td>…</td>
</tr>
</tbody>
</table>

---

Flash Memory Summit 2014
Santa Clara, CA
**NVM Subsystem Architectural Model**

- **NVMe Management Endpoint** – An MCTP endpoint that is the terminus and origin of MCTP packets/messages and which processes MCTP and NVMe Management Interface commands

- **NVMe Controller Management Interface** – An interface associated with each NVMe controller in the NVM subsystem that is responsible for executing management operation on behalf on an NVMe Management Endpoint
1. Management Controller sends request message to NVMe Management Endpoint
2. Management Endpoint processes command
3. NVMe Management Endpoint sends response message to Management Controller
1. Management Controller sends request message to NVMe Management Endpoint
2. NVMe Management Endpoint forwards request to appropriate NVMe Controller Management Interface
3. NVMe Controller Management Interface executes command on NVMe Controller
4. NVMe Management Endpoint sends response back to NVMe Management Endpoint
5. NVMe Management Endpoint sends response message to Management Controller
Two or more requests received on NVMe Management Endpoints which contain PCIe or NVMe commands that target the same NVMe Controller.
Management Concurrency (case 2)

- Two requests received on a NVMe Management Endpoint which contain NVMe Management Interface Specific Commands that conflict
A management command that targets an NVMe Controller that is executing an NVMe command
Handling Management Concurrency

- **Case 1:** Two or more requests received on NVMe Management Endpoints which contain PCIe or NVMe commands that target the same NVMe Controller
  - Concurrent management commands executed on an NVMe Controller shall be executed in parallel and the result shall be equivalent to some sequential execution of commands

- **Case 2:** Two requests received on a NVMe Management Endpoint which contain NVMe Management Interface Specific Commands that conflict
  - Concurrent management commands executed on an NVMe Management Endpoint shall be executed in parallel and the result shall be equivalent to some sequential execution of commands

- **Case 3:** A management command that targets an NVMe Controller that is executing an NVMe command
  - No hardware enforcement of concurrency between management commands and NVMe commands
Vital Product Data (VPD)

- Vital Product Data (VPD) contains information about the storage device
  - Examples:
    - Manufacturer
    - Serial number
    - Device configuration
    - Power requirements
  - See IPMI FRU information

- VPD accessible using I2C serial EEPROM read/write operations and NVMe Management Interface commands over MCTP

- Two I2C addresses
  - I2C serial EEPROM access (VPD device)
  - MCTP Endpoint (NVMe controller ASIC)

- VPD accessibility during power modes
  - During Auxiliary Power
    - I2C serial EEPROM read/write
  - During Main Power
    - I2C serial EEPROM read/write
    - NVMe Management Interface commands
SMBus/I2C Topologies

Shared SMBus/I2C

- Management Controller
- SMBus/I2C
- Other SMBus Device
- Other SMBus Device
- NVMe SSD
- NVMe SSD
- NVMe SSD

Requires Unique SMBus/I2C addresses

Segmented SMBus/I2C

- Management Controller
- SMBus/I2C Mux
- SMBus/I2C
- Other SMBus Device
- Other SMBus Device
- NVMe SSD
- NVMe SSD
- NVMe SSD
- NVMe SSD

Repeated SMBus/I2C Addresses Supported
SMBus/I2C Addressing

- **During Auxiliary Power (if supported)**
  - I2C serial EEPROM read/write access at default SMBus/I2C address 0xA6, but may be modified using ARP

- **During Main Power**
  - MCTP Endpoint at default SMBus/I2C address 0xD4, but may be modified using ARP
  - I2C serial EEPROM read/write access
    - If auxiliary power was provided, then SMBus/I2C address shall be maintained if modified using ARP; otherwise, the default address is 0xA6
    - SMBus/I2C address may be modified using ARP

- Supports both shared and segmented SMBus/I2C environments
Summary

- We are standardizing out-of-band management interface for NVMe storage devices
  - PCIe VDM and SMBus/I2C

- The NVMe management interface is leveraging other management specifications/standards
  - Complementary and not a replacement

- The specification is planned to be completed at the end of this year
References

- MCTP Overview:
- MCTP Base Spec:
  http://www.dmtf.org/sites/default/files/standards/documents/DSP0236_1.2.0.pdf
- MCTP SMBus/I2C Binding:
  http://www.dmtf.org/sites/default/files/standards/documents/DSP0237_1.0.0.pdf
- MCTP over PCIe VDM Overview:
- MCTP PCIe VDM Binding:
  http://www.dmtf.org/sites/default/files/standards/documents/DSP0238_1.0.1.pdf
- IPMI Platform Management FRU Information Storage Definition:
NVM Express – An Implementer’s Perspective

Ken Okin
Systems Architect
HGST

Audio-Visual Sponsor
Shannon Systems
Contents

- Where we are now
- Latest Additions
- Future Possibilities
Where we are now…

- NVMe was a clean slate built from the ground up for high performance SSDs and future non-volatile storage possibilities
  - Defines Device Commands and Transport
  - Leverages PCIe, T10 (SGLs, Security)

- Simple yet extensible
  - Storage addressed by NAMESPACE.LBA

- Multi-queue based communication: Host and controllers write to submission and completion queues in Host memory. PCIe is only read by the Host only at initialization.
Managing Command Processing Latency

- Relative Latency  1Ghz -> 1ns clock cycle
- L1 / L2 (L3)  Caches very close:
- Memory hundreds of clock cycles away
- Storage device thousand(s) of clock cycles away
Where we are now

- Can build a simple client controller
- Can build a very high performance controller from the same specification
  - Multiple submission / completion queues from different processor packages
  - Controller can reorder command execution
  - Controller can deliver data out of order
- Can create a complex NVMe Subsystem with multiple controllers using multiple physical links and/or multiple functions on a link
Latest Additions

- Standardized Namespace Management
- Flexibility to easily reallocate Disks / Partitions
Latest Additions

- **Workload Optimized Namespaces**
  - Database logging Namespace optimized for sequential 512 byte writes
  - Main Database Namespace optimized for random reads and writes

`NVMe SSD`
Latest Additions

- Command effects Log (Pass Through Commands)
  - Allows OS to understand side effects of commands and properly insert them into Submission Queues

- Allows device / application innovation without requiring OS framework updates
Latest Additions

- System Memory Architecture Enhancements
  - Host Memory Buffer
    - Controller Scratchpad
    - V2P table for low cost or really huge drives
Latest Additions

- Today SQ / PRP lists / SGLs in HOST memory
- Command execution latency
Latest Additions

- Move SQ / PRP lists / SGLs to device reduces command initiation latency
- Simplify Controller Front End
Promise for the Future

- Weighted Round Robin with Priority Submission queues.
Promise for the Future

- I/O Command Prioritization
  - Today Applications cannot tell the driver the priority of a given command.
  - Once they can, The NVMe Weighted Round Robin with Priority arbitration can meter commands into the Controller and keep lower priority commands in the Host memory.
  - OS Challenge: Deliver application I/O priority from the application through to the driver.
Promise for the Future – NVMe Specification evolution

- Namespace Management
  - Richer set of manageable Namespace attributes
    - R/W vs R/O
    - Controller Protection

![Diagram showing namespace management and controller protection](image)
Future Possibilities

- Simple interface can interact with more complex data storage systems
- High availability Namespace Attribute: Don’t post a write to the Completion queue until the data is safely stored in a remote device.
The existing NVMe command set can interact with more complex data storage systems

- Transparent data migration: Data may reside in a hidden storage hierarchy but still be referenced by namespace and LBA
Namespaces as files:

- File system should resolve naming, protection, and gross device utilization, but not micro allocate storage within files.
- The controller is always shuffling data around, let it allocate the blocks and communicate about subsystem “fullness” back to the file system as needed.
Thanks

Come see HGST’s NVMe technology demo in the NVMe Booth