Overview of the NVMe Management Interface Specification

SNIA SDC INDIA,
May 26th 2016
Karthik Venkatasubba
Manjunath AM
Dell R&D, Bangalore
Agenda

• NVMe
  – Legacy storage stacks on modern SSS – no performance improvement
  – NVMe’s improvisation over legacy protocols exploiting flash characteristics

• NVMe Management
  – In-band v/s Out-of-Band management paradigms
  – Out-of-band mgmt. protocol framework
  – OSI Model
  – Architectural Model

• Overview of Features in NVMe Management Interface
  – Control Primitives
  – NVMe Management Commands
  – NVMe Admin Commands
  – PCIe Commands

• Q & A
NVMe’s Comparative Performance

### Random IOPs

- **NVM Express**
  - Random Read 4K: 450,000
  - Random Write 4K: 130,000

- **SAS 12Gbps**
  - Random Read 4K: 175,000
  - Random Write 4K: 175,000

- **SATA 6Gbps**
  - Random Read 4K: 75,000
  - Random Write 4K: 100,000
  - Random Read 4K: 36,000

### Sequential Bandwidth

- **NVM Express**
  - Sequential Read: 2800 MB/s
  - Sequential Write: 1100 MB/s

- **SAS 12Gbps**
  - Sequential Read: 500 MB/s
  - Sequential Write: 2000 MB/s

- **SATA 6Gbps**
  - Sequential Read: 765 MB/s
  - Sequential Write: 460 MB/s
Current Performance Bottleneck (Resolved)

- 1 queue
- 32 commands/queue

Parallel Operation improves performance

- 65535 queues
- 64,000 commands/queues
Software Stack Improvements

- Submission latency and CPU cycles reduced > 50%
  - SAS: 6.0 us, 19,500 cycles
  - NVMe: 2.8 us, 9,100 cycles
### Storage Protocols Compared

<table>
<thead>
<tr>
<th></th>
<th>SATA</th>
<th>SAS</th>
<th>SOP/PQI</th>
<th>PCIe</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Drive Form Factors</strong></td>
<td>SATA 1.8”,2.5”,3.5”</td>
<td>SAS 2.5”, 3.5”</td>
<td>2.5”</td>
<td>NVMEExpressCard</td>
</tr>
<tr>
<td><strong>No of Ports/Lanes</strong></td>
<td>1</td>
<td>1,2</td>
<td>1, 2, 4</td>
<td>1, 2, 4 (8 on card)</td>
</tr>
<tr>
<td><strong>Command Set/Queue Interface</strong></td>
<td>ATA / SATA-IO</td>
<td>SCSI / SAS</td>
<td>SCSI /SAS</td>
<td>NVM Express</td>
</tr>
<tr>
<td><strong>Transfer Rate</strong></td>
<td>6 Gb/s</td>
<td>12 Gb/s</td>
<td>12 Gb/s</td>
<td>8 Gb/s</td>
</tr>
<tr>
<td><strong>Drive Connector</strong></td>
<td>SFF-xxxxxx</td>
<td>SFF-8680</td>
<td>SFF-8639</td>
<td>SFF-8639 (2.5”),CEM (Edge-Card)</td>
</tr>
<tr>
<td><strong>Express Bay Compatible?</strong></td>
<td>Yes, 2.5”</td>
<td>Yes, 2.5”</td>
<td>Yes, 2.5”</td>
<td>Yes, 2.5”</td>
</tr>
<tr>
<td><strong>Drive Power (Typical)</strong></td>
<td>9W Typical</td>
<td>9W Typical</td>
<td>Upto 25W</td>
<td>Upto 25W</td>
</tr>
<tr>
<td><strong>Max Bandwidth</strong></td>
<td>0.6 GB/s (x2)</td>
<td>4.8 GB/s (x4)</td>
<td>9.6 GB/s (x4)</td>
<td>8 GB/s (x4)</td>
</tr>
<tr>
<td><strong>Host Driver Stack (Stg Ctrlr/Direct Drives)</strong></td>
<td>AHCI</td>
<td>IHV</td>
<td>IHV</td>
<td>Common Driver (SOP/PQI)</td>
</tr>
</tbody>
</table>
Command Submission

1. Host writes command to Submission Queue
2. Host writes updated Submission Queue tail pointer to doorbell

Command Processing
3. Controller fetches command
4. Controller processes command
Command Completion

1. Queue Command
2. Ring Doorbell, New Tail
3. Submission Queue Tail Doorbell
4. Fetch Command
5. Process Command
6. Queue Completion
7. Process Completion
8. Ring Doorbell, New Head

**NVMe Controller**

**Command Completion**

5. Controller writes completion to Completion Queue
6. Controller generates MSI-X interrupt
7. Host processes completion
8. Host writes updated Completion Queue head pointer to doorbell
Why NVMe is becoming popular?

• There is an increasing gap in the performance of DRAM and hard drives. NVMe in the form of Solid State Drives is filling this gap.
PCle SSD Form Factors

- Add-in Card (AIC)
- 2.5” SSD FF (hot plug)
NVME MANAGEMENT
## Management Protocol Stack

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Transmission unit</th>
<th>Endpoints</th>
<th>Operations</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>SMBus</td>
<td>Bytes</td>
<td>Master, Slave</td>
<td>Commands - READ/WRITE</td>
<td>Half duplex</td>
</tr>
<tr>
<td>PCIe VDM</td>
<td>TLP</td>
<td>Requester/Completer</td>
<td>Transactions - Memory, IO, configuration</td>
<td>Full duplex</td>
</tr>
<tr>
<td>MCTP binding spec</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MCTP</td>
<td>Messages</td>
<td>Source/Destination</td>
<td>Control commands</td>
<td>Medium-specific</td>
</tr>
<tr>
<td>Application-defined</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Packet layering**

- **SMBus/PCle VDM**
- **MCTP**
- Application-define payload

**Application-defined payload**
In-Band vs Out-of-Band Management

1. NVMe driver communicates to NVMe controllers over PCIe per NVMe Spec

2. Two OOB paths: PCIe VDM and SMBus
   • Note: PCIe VDMs are completely separate from in-band PCIe traffic though they share the same physical connection
In-band vs Out-of-Band Management (cont’d)

• In-Band Management (OS agents)
  • Many host OSes to support (Windows, Linux, VMWare, etc.)
  • Several different flavors/distros of each OS
  • Developing/maintaining/validating a management application for every OS variant is resource/cost-prohibitive
  • New revisions of OS and NVMe driver released over time
  • If given a choice, customers would want to do away with installing management agents in the OS which continuously consume CPU cycles
  • Security implications
  • Management features vary per OS

• Out-of-Band Management (Agent-free)
  • Develop management application in one operating environment (i.e. BMC)
  • Works the same across any host OS
  • Works across no OS cases (pre-boot, deployment)
  • Doesn’t consume host CPU cycles
NVMe-MI

– A programming interface that allows *out-of-band management* of an NVMe Field Replaceable Unit (FRU) or an embedded NVMe NVM Subsystem

Four pillars of systems management:
– Inventorying
– Configuration
– Monitoring
– Change Management

Management operational times:
– Deployment (No OS)
– Pre-OS (e.g. UEFI/BIOS)
– Runtime
– Decommissioning
– Auxiliary Power
NVMe-MI OSI Model

MCTP defines the transport layer
  • Refer to DMTF Specs: DSP0236, DSP0237, DSP0238, DSP0235

NVMe-MI defines:
  • Messages for BMC (aka SP or MC) to NVMe (aka device or PCIe SSD) out-of-band communication
  • Additional flow control and exception handling on top of MCTP
  • VPD access
NVMe-MI Architectural Model

• **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**: One Controller/Port

**NVM Subsystem’s anatomy**
An NVMe FRU consists of one and only one NVM Subsystem with

- One or more PCIe ports (PCIe VDM)
- Optional SMBus/I2C port
- Management Endpoint per port
- Two Command Slots per Management Endpoint
- Controller Management Interface per NVMe Controller
- FRU Information Device
NVME MANAGEMENT COMMANDS
NVMe-MI Message Types

Types of MCTP Messages

- Request Message
  - Command Message
    - NVMe-MI Command
    - PCIe Command
    - NVMe Admin Command
  - Control Primitive
- Response Message
  - Success
  - Error
- Other MCTP Messages (e.g., MCTP control)
Control Primitives

- Control Primitives enable a Management Controller to utilize flow control and to detect and recover from errors

- Control Primitives fit into a single packet and do not require message assembly

<table>
<thead>
<tr>
<th>Control Primitive</th>
<th>O/M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pause</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Resume</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Abort</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Get State</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Replay</td>
<td>Mandatory</td>
</tr>
</tbody>
</table>
Command Slots

- Each NVMe-MI Management Endpoint has two Command Slots to service Command Messages
- Each Command Slot follows this state machine
Management Interface Command Set

- Discover Capabilities
- Optimized Health Monitoring/polling
- Initialize & troubleshoot NVMe-MI
- Efficiently manage NVMe at the FRU level
- Sub-system level

<table>
<thead>
<tr>
<th>Command</th>
<th>O/M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configuration Set</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Configuration Get</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Controller Health Status Poll</td>
<td>Mandatory</td>
</tr>
<tr>
<td>NVM Subsystem Health Status Poll</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Read NVMe-MI Data Structure</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Reset</td>
<td>Mandatory</td>
</tr>
<tr>
<td>VPD Read</td>
<td>Mandatory</td>
</tr>
<tr>
<td>VPD Write</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Vendor Specific</td>
<td>Optional</td>
</tr>
</tbody>
</table>
NVMe Admin Commands

- NVMe-MI defines mechanism to send existing NVMe Admin Commands out-of-band

- Admin Commands target a controller in the NVM subsystem

<table>
<thead>
<tr>
<th>Command</th>
<th>O/M</th>
</tr>
</thead>
<tbody>
<tr>
<td>Get Features</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Get Log Page</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Identify</td>
<td>Mandatory</td>
</tr>
<tr>
<td>Firmware Activate/Commit</td>
<td>Optional</td>
</tr>
<tr>
<td>Firmware Image Download</td>
<td>Optional</td>
</tr>
<tr>
<td>Format NVM</td>
<td>Optional</td>
</tr>
<tr>
<td>Namespace Management</td>
<td>Optional</td>
</tr>
<tr>
<td>Security Send</td>
<td>Optional</td>
</tr>
<tr>
<td>Security Receive</td>
<td>Optional</td>
</tr>
<tr>
<td>Set Features</td>
<td>Optional</td>
</tr>
<tr>
<td>Vendor Specific</td>
<td>Optional</td>
</tr>
</tbody>
</table>
PCIe Commands

• PCIe Commands provide optional functionality to read and modify PCIe memory

<table>
<thead>
<tr>
<th>Command</th>
<th>O/M</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCIe Configuration Read</td>
<td>Optional</td>
</tr>
<tr>
<td>PCIe Configuration Write</td>
<td>Optional</td>
</tr>
<tr>
<td>PCIe Memory Read</td>
<td>Optional</td>
</tr>
<tr>
<td>PCIe Memory Write</td>
<td>Optional</td>
</tr>
<tr>
<td>PCIe I/O Read</td>
<td>Optional</td>
</tr>
<tr>
<td>PCIe I/O Write</td>
<td>Optional</td>
</tr>
</tbody>
</table>
Basic Management Command

• Simple and optional command
• Intended for Vendors/System integrators looking for a light-weight NVMe out-of-band device monitoring
• Does not use MCTP
• Limited set of attributes could be monitored by the host via SMBus like: Temperature, Critical Warnings and Life Remaining
• Mode of operation is very much like a typical VPD access to FRU information device

Example SMBus block read of the drive’s status (status flags, SMART warnings, temperature):

```
<table>
<thead>
<tr>
<th>Start</th>
<th>Addr</th>
<th>W</th>
<th>Ack</th>
<th>Cmd Code</th>
<th>Ack</th>
<th>Addr</th>
<th>R</th>
<th>Ack</th>
<th>Length</th>
<th>Ack</th>
<th>Status Flags</th>
<th>Ack</th>
<th>SMART Warnings</th>
<th>Ack</th>
<th>Temp</th>
<th>Ack</th>
<th>Drive Life Used</th>
<th>Ack</th>
<th>Reserved</th>
<th>Ack</th>
<th>Reserved</th>
<th>Ack</th>
<th>PEC</th>
<th>Ack</th>
</tr>
</thead>
<tbody>
<tr>
<td>D4h</td>
<td>00h</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>D5h</td>
<td></td>
<td></td>
<td>06h</td>
<td></td>
<td>BFh</td>
<td></td>
<td>FFh</td>
<td></td>
<td>1Eh</td>
<td></td>
<td>01h</td>
<td></td>
<td>00h</td>
<td></td>
<td>00h</td>
<td></td>
<td>10h</td>
<td></td>
</tr>
</tbody>
</table>
```
NVMe-MI standardizes an **out-of-band** management interface to discover, monitor and configure NVMe devices
NVMe-MI adds the ability to manage NVMe **at the FRU level**

References

• SSD Form Factor Working Group, http://www.ssdformfactor.org/
• SMBus, http://smbus.org/
• PCI SIG, https://pcisig.com/
• DMTF, http://dmtf.org/
• NVMe, www.nvmexpress.org