NVMe: Solid-state drives meet PCI Express

Helen Koike <helen.koike@collabora.com>
Overview

- SSD/PCIe/NVMe vs HDD/SATA/AHCI
- NVM express specification
  - Core concepts and mechanics (queues)
  - Command types
- Linux Support
- Features in specification version 1.3
Bottlenecks

Why NVM express?
SSD++; HDD--;

Replace

SATA/AHCI

SATA/AHCI
SSD vs Disk

- No moving parts, lower latency
- Internal parallelism:
  - Typical page size: 4KB
  - Multiple pages at the same time
Bottleneck

- Sata/AHCI
  - Optimized for disks
- Sata III
  - 600 MB/s
- My Notebook (PCIe 3.0)
  - 2.4 GB/s (read speed)
PCI Express 3.0

- 1 GB/s per lane
- M.2 form factor → x4 lanes
- X16 → ~16GB/s (per PCIe Port)
- >>>> Sata III (600 MB/s)
AHCI vs NVMe

- **AHCI**
  - Optimized for high latency and low parallelism

- **NVMe – Non Volatile Memory Express**
  - Designed from scratch
  - Originally to PCI Express
  - Fabrics – thousands of devices
Concepts and mechanics

NVM express specification
NAND Flash - Internals

- **Physical Page**
  - Minimum data to read/write
- **Physical Block**
  - Collection of pages
  - Bits - set to 0 - individually
  - Bits - set to 1 - after an erase cycle
  - Minimum data to erase
NAND Flash - Internals (2)

• Update content
  - Only flip bits to 0 → OK
  - Flip bits to 1:
    • read, erase, re-write

• Limited number of program/erase cycles
  - Frequent updates can damage the block
  - Wear-leveling
NVMe - Logical Blocks

- Target for generic storage (not only for NAND/NOR flash technology)
- Abstracts physical pages and blocks
- Logical Block: minimum data to read/write
- Specification above erases and wear leveling management
NVMe Subsystem

- Namespace
  - Collection of logical blocks formatted in a given size
- Host
  - Send commands to the SSD
- Controller
  - Execute commands from the Host
NVMe Subsystem (2)
NVMe Subsystem (3)

PCle Port

<table>
<thead>
<tr>
<th>PCle Function 0</th>
<th>PCle Function 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Controller 0</td>
<td>Controller 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>NSID 1</th>
<th>NSID 2</th>
<th>NSID 2</th>
<th>NSID 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>NS A</td>
<td>NS B</td>
<td>NS C</td>
<td></td>
</tr>
</tbody>
</table>
NVMe Subsystem (4)
System bus registers

- PCI Express specific
  - Device vendor
  - Interruption
  - Errors / Device status
  - Power management
  - Latency / Clock
  - etc
Controller registers

- Independent of PCI Express bus
- Usually memory mapped
- MM address configured through PCI Express registers BAR0 and BAR1
- Basic configuration
Queue management

NVM express specification
**Queues**

**Submission Queue**
- Head
- Tail

**Completion Queue**
- Head
- Tail
Admin And I/O Queues

Host memory

Admin Queue

I/O Queues

Controller
Instantiating Queues

- Memory allocated in the Host
- Admin queue
  - Base address - Controller registers
- I/O queues
  - Created/deleted using admin commands
AHCI comparison

• AHCI
  - Single queue
  - 32 outstanding commands

• NVMe
  - 65,535 I/O queues
  - 64K outstanding commands per queue
  - Capitalize on parallelism
  - A queue per CPU – avoid lock contention
Doorbell – Submission Queue Tail

- Host – sending a new command
  - Fill a new entry in SQ
  - Update SQ Tail pointer
- Controller register called Doorbell
- A Doorbell register per queue
- Controller fetches new entries from SQ
Completion queue entry

- Posted by the controller
- Command concluded

Completion queue entry layout

<table>
<thead>
<tr>
<th>Command Specific</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reserved</td>
</tr>
<tr>
<td>SQ Identifier</td>
</tr>
<tr>
<td>Status Field</td>
</tr>
<tr>
<td>Command Identifier</td>
</tr>
</tbody>
</table>
Submission Queue - Head/Tail

Host's memory

SQ 6
0  CMD ID 9
1  CMD ID 22
2  CMD ID 44
4

CQ 4
0  HEAD: 1
   CMD ID 22
   SQ 6
   HEAD 2
2  TAIL: 2
3
4

Controller Registers

Doorbell: SQ 4
Doorbell: SQ 5
Doorbell: SQ 6 3
Doorbell: SQ 7

Completion Queue - Head

Host's memory

<table>
<thead>
<tr>
<th></th>
<th>SQ 6</th>
<th>CQ 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>CMD ID 9</td>
<td>CMD ID 22</td>
</tr>
<tr>
<td>1</td>
<td>CMD ID 22</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>CMD ID 44</td>
<td>CMD ID 22</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td>SQ 6</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td>HEAD-2</td>
</tr>
</tbody>
</table>

Controller Registers

- Doorbell: SQ 4
- Doorbell: CQ 4
- Doorbell: SQ 5
- Doorbell: CQ 5
- Doorbell: SQ 6
- Doorbell: CQ 6
- Doorbell: SQ 7
- Doorbell: CQ 7
Completion Queue - Tail

- Updated by the controller
- Inverts Phase Tag bit
- Interruption to the host (if configured)
Completion Queue - Tail (2)

CQ

HEAD/TAIL: 0
1. P = 0
2. P = 0
3. P = 0
4. P = 0

CQ

HEAD: 0
1. P = 1
2. P = 1
3. P = 1
4. P = 0

New 3 entries

TAIL: 3
4. P = 0
Ordering

- Execution order not ensured
- Neither between queue nor within a queue
- Wait for completion queue entry
- Fused operation – 2 cmds - atomic unity
Types of commands

NVM express specification
Set of commands

- 13 mandatory commands
  - 10 Admin
  - 3 I/O
- Low overhead
Admin mandatory commands

- Delete I/O Submission Queue
- Create I/O Submission Queue
- Delete I/O Completion Queue
- Create I/O Completion Queue
- Get log page
- Identify
- Abort
- Set Features
- Get Features
- Asynchronous Event Request
I/O mandatory commands

• Write
• Read
• Flush
Asynchronous Event Request

- Messages not associated with a specific command
- Host post a Asynchronous Event Request command associated with an event type
- Controller only completes the command when the event happens
- Support multiple pending commands
Recent features and Linux Kernel support

- Linux Kernel
  - Good support over PCI Express
  - Multi-queue introduced with Kernel 3.13
  - Tuning of NVMe over fabrics
- Doorbell Buffer Config Command
- Directives Command
- Overview
Doorbell Buffer Config Command

- VM Exit when writing to Doorbell
- Overhead from context switching

OSSNA 2017
Doorbell Buffer Config Command (2)

- Avoid writing to Doorbell register
- Doorbells values in the Host’s memory
- Emulated controller fetches the Doorbell value when convenient
- Shadow Doorbell buffer
- Host can still write to the Doorbell register
Doorbell Buffer Config Command (3)

- EventIdx buffer
  - Hint to ring the Doorbell when its value is greater then a threshold
  - Avoid accumulating commands
- Shadow Doorbell / Event Index buffers
  - Configured through Doorbell Buffer Config Command in the admin queue
Doorbell Buffer Config Command (4)

- 4x IOPS with fio v2.1 in Kernel 4.12 at GCE
- Spec v1.3 / Included in Kernel 4.12

43.9K IOPS

184K IOPS
Directives

• NAND Flash
  - Entire block needs to be erased to re-use a page
  - Abstracted by NVMe
  - Garbage collector
Directives (2)

Block

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Invalid</td>
<td>Invalid</td>
<td>E</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>G</td>
<td>H</td>
<td>I</td>
<td>Invalid</td>
</tr>
<tr>
<td></td>
<td>J</td>
<td>K</td>
<td>Invalid</td>
<td>L</td>
</tr>
</tbody>
</table>

Garbage Collection

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>E</td>
<td>F</td>
<td>G</td>
<td>H</td>
</tr>
<tr>
<td></td>
<td>I</td>
<td>J</td>
<td>K</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>Free</td>
<td>Free</td>
<td>Free</td>
<td>Free</td>
</tr>
</tbody>
</table>
Directives (3)

- Place pages with similar life time in the same block
- Avoid unnecessary copying and erase operations
- Improve performance and endurance
- Spec v1.3 – write hints
Directives (4)

- Writes with the same Stream ID are associated
- Exposes 4 hint levels to user space through fcntl system call
  - RWH_WRITE_LIFE_SHORT
  - RWH_WRITE_LIFE_MEDIUM
  - RWH_WRITE_LIFE_LONG
  - RWH_WRITE_LIFE_EXTREME
Directives (5)

- Merged in Kernel 4.13
- According to Jens Axboe (patch author):
  - 25% reduction in NAND writes in a RocksDB setup
Other new features in spec 1.3

- Device self test command
- Sanitize command
- Boot partitions
- Telemetry
- Virtualization Enhancements
- Host Controlled Thermal Management
Questions?

Helen Koike <helen.koike@collabora.com>

Creative Commons Attribution-ShareAlike 4.0