PERFORMANCE ANALYSIS OF NVME SSDS
AND THEIR IMPLICATION ON REAL WORLD DATABASES
Non-Volatile Memory Express (NVMe)

- A storage **protocol standard** on top of PCIe

- NVMe SSDs connect through PCIe and support the standard
  - Since 2014 (Intel, Samsung)
  - Enterprise and consumer variants
Agenda

- NVMe Overview
- NVMe drive characterization
  - Order-of-magnitude better performance than SATA SSDs
  - How/where does the performance come from?
- End-to-end performance benefits for applications
  - Exemplify using MySQL, Cassandra, and MongoDB
Why PCIe SSD? (1/3)

Legacy interfaces became a bandwidth bottleneck for SSDs

80-160MB/sec

SATA : 600MB / sec

Bottleneck!

PCIe 3.0: 1GB / sec * 4 lanes

3-4GB/sec

CPU
Why PCIe SSD? (2/3)

Legacy interfaces add unnecessary delays

- SATA SSDs connect through either:
  - Slow-clock on-board PCH or Host-Bus-Adapter (HBA)
  - NVMe SSDs connect directly to the PCIe root port
Why PCIe SSD? (3/3)
Legacy protocols/software limit performance

- Legacy protocols were built for HDDs
  - A sequential device: read one block at a time
- But, SSDs are inherently parallel
  - Multiple channels, multiple flash chips

N. Agrawal et al., “Design Tradeoffs for SSD Performance”, USENIX '08
NVMe Standardizes PCIe SSDs

- Several vendors developed proprietary PCIe SSD variants
  - Fusion IO, Violin Memory, Micron
- Lack of standard hindered adoption
- NVMe standardized the interface
  - Command set, feature set, etc.
  - Enables standard drivers and inter-vendor interoperability
NVMe Interface

- Architected from ground up for NVM, and with **multicore** in mind
- Multiple, paired Submission and Completion queues
- Enterprise-friendly features
  - Multiple namespaces, priorities, dataset management hits, robust error reporting

Figure: “NVMe Express: Optimizing for PCIe SSD Performance”, [http://www.snia.org/](http://www.snia.org/)

---

Controller Management

Admin Submission Queue
Admin Completion Queue

Core 0
I/O Submission Queue
I/O Completion Queue

Core 1
I/O Submission Queue
I/O Completion Queue

Core N
I/O Submission Queue
I/O Completion Queue

Host

NVMe Controller

MSI-X

---
NVMe Raw Performance Analysis

Experimental Environment

- Dual-socket, 8-cores Xeon E5-2670
- 3 enterprise-class devices

<table>
<thead>
<tr>
<th>Family</th>
<th>Drive</th>
</tr>
</thead>
<tbody>
<tr>
<td>SATA HDD</td>
<td>Seagate Enterprise Performance 15K</td>
</tr>
<tr>
<td>SATA SSD</td>
<td>Samsung 843 Pro Enterprise</td>
</tr>
<tr>
<td>NVMe SSD</td>
<td>Samsung XS1715</td>
</tr>
</tbody>
</table>

- Ubuntu 12.04, kernel v3.14
- fio used for traffic generation
  - Bypassing the page-cache
- Blktrace used to characterize the software stack
  - Via instrumentation of the NVMe driver
**NVMe SSD Bandwidth**

*Order-of-magnitude improvement over SATA-SSDs*

<table>
<thead>
<tr>
<th>Storage Type</th>
<th>4K Random Read IOPS (IOPS)</th>
<th>Sequential Read Bandwidth (MB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SATA-HDD</td>
<td>190</td>
<td>245MB/s</td>
</tr>
<tr>
<td>SATA-SSD</td>
<td>70k</td>
<td>530MB/s</td>
</tr>
<tr>
<td>NVMe</td>
<td>750k</td>
<td>3GB/s</td>
</tr>
</tbody>
</table>

4K Random Read IOPS: NVMe offers a 10x improvement over SATA-SSDs.

Sequential Read Bandwidth: NVMe provides a 6x improvement over SATA-SSDs.
Latency Comparison

Unloaded read latency reduced to 0.7X

Access Latency (µs)

5.3ms

80µs

135µs

80µs

94µs

39µs

SATA-HDD
SATA-SSD
NVMe

Read Latency
Write Latency
~40us Reduction
Software Overheads

NVMe SSD considerably reduces software overheads along the I/O path

Pct. of time spent in each software section out of total IO access latency

- Application
- Block layer
- Kernel other
- NVMe driver

SATA HDD: 0%  
SATA SSD: 28%  
NVMe SSD: 8%
Simplified Software Stack

- NVMe bypasses the block-layer
  - The block-layer was designed for HDD and is inefficient for SSDs
    - Request reordering and combining is unneeded
    - Single request queue
    - Lock contention across threads/cores

- The block layer re-architected in current Linux version
  - Blk-mq for SATA (v3.17)
  - Blk-mq for NVMe (v3.18)
Application Performance

Evaluate how NVMe raw performance translate to application speedup
MySQL, Cassandra, MongoDB

Compare three Configurations:

1 SATA SSDs (Samsung 843 Pro)
4 SATA SSDs (Samsung 843 Pro)
1 NVMe SSD (Samsung XS1715)

Single server-client setup
Dual socket, Xeon E5 server
10 GbE
TPC-C Performance

Workload Setup

- Database: MySQL
- Driver: HammerDB
- Working-set:
  - 1024 warehouses, 95GB dataset
  - 64 virtual users
- Run:
  - 6 hours, report on Transaction-per-Minute (tpmC)
- Disk-intensive workload
  - Random mix of two reads to one write traffic
TPC-C Performance
Single SATA SSD

Speedup

Machine is mostly waiting for I/O
Relatively low disk BW
TPC-C Performance
Single SATA SSD

Speedup

- Load distributed across 4 drives
- Wait time reduced by 3X
TPC-C Performance
Single NVMe SSD

- **Speedup**

<table>
<thead>
<tr>
<th>Relative performance (tpmc) Improvement</th>
<th>One SSD</th>
<th>Four SSD</th>
<th>One NVMe</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.0</td>
<td>1.5</td>
<td>3.5</td>
</tr>
</tbody>
</table>

- **Minimal I/O wait-time**
- **User-level activity at 30%**
TPC-C Performance
Tmpfs (in-memory)

Speedup

<table>
<thead>
<tr>
<th>Relative performance (tpmc) Improvement</th>
<th>One SSD</th>
<th>Four SSD</th>
<th>One NVMe</th>
<th>Tmpfs</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.0</td>
<td>1.5</td>
<td>3.5</td>
<td>5.0</td>
</tr>
</tbody>
</table>

CPU Util.

(a) OneSSD_64vUsers
(b) FourSSD_64vUsers
(c) OneNVMe_64vUsers
(d) Tmpfs-64vUsers
Cassandra Performance

Large speedups for read-heavy workloads

- 8.5x over a single-SATA SSD
- 4x over 4-SATA SSD

Transactions-Per-Second (Normalized)

<table>
<thead>
<tr>
<th></th>
<th>Single SSD</th>
<th>Four SSD</th>
<th>One NVMe</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A</strong> (Read=50%, Update=50%) Zipfian</td>
<td>1</td>
<td>4.1</td>
<td>5.7</td>
</tr>
<tr>
<td><strong>B</strong> (Read=95%, Update=5%) Zipfian</td>
<td>1</td>
<td>2.1</td>
<td></td>
</tr>
<tr>
<td><strong>C</strong> (Read=100%)</td>
<td>1</td>
<td>1.7</td>
<td></td>
</tr>
<tr>
<td><strong>D</strong> (Read=95%, Insert=5%) Latest</td>
<td>1</td>
<td>1.1</td>
<td>1.5</td>
</tr>
</tbody>
</table>
MongoDB Performance
NVMe SSD delivers the same performance as 4-SATA SSDs

Client Throughput (Normalized)

Relative Client Throughput

A
Read=50%
Update=50%
Zipfian
1.56
1.56
1

B
Read=95%
Update=5%
Zipfian
2.24
2.22
1

C
Read=100%
1.3
1.38
1

D
Read=95%
Insert=5%
Latest
1.61
1.63
1
Acknowledgements

Project team:

Huzefa Siyamwala
Mrinmoy Ghosh
Tameesh Suri
Manu Awasthi
Zvika Guz
Anahita Shayesteh
Vijay Balakrishnan

This project was done @ Memory Solution Lab, Samsung Semiconductor Inc, summer 2014, and published @ SYSTOR’15
SUMMARY

NVMe architected for NVM and SSD
  Overcomes drawbacks of traditional interfaces
    Shorter latencies, higher bandwidth, more parallelism
NVMe SSDs leverage the interface to deliver superior perf
  5X to 10X over SATA-SSD
Raw performance translates to overall application speedups
  3.5X for MySQL
  Up to 8.5 for Cassandra

THANK YOU!
QIUMIN@USC.EDU