Key Value SSD Explained – Concept, Device, System, and Standard

YANG SEOK KI
Samsung Electronics
Disclaimer

This presentation and/or accompanying oral statements by Samsung representatives collectively, the “Presentation”) is intended to provide information concerning the SSD and memory industry and Samsung Electronics Co., Ltd. and certain affiliates (collectively, “Samsung”). While Samsung strives to provide information that is accurate and up-to-date, this Presentation may nonetheless contain inaccuracies or omissions. As a consequence, Samsung does not in any way guarantee the accuracy or completeness of the information provided in this Presentation.

This Presentation may include forward-looking statements, including, but not limited to, statements about any matter that is not a historical fact; statements regarding Samsung’s intentions, beliefs or current expectations concerning, among other things, market prospects, technological developments, growth, strategies, and the industry in which Samsung operates; and statements regarding products or features that are still in development. By their nature, forward-looking statements involve risks and uncertainties, because they relate to events and depend on circumstances that may or may not occur in the future. Samsung cautions you that forward looking statements are not guarantees of future performance and that the actual developments of Samsung, the market, or industry in which Samsung operates may differ materially from those made or suggested by the forward-looking statements in this Presentation. In addition, even if such forward-looking statements are shown to be accurate, those developments may not be indicative of developments in future periods.
Outline

- Background
- Concept
- Key Value SSD
- Ecosystem
- Standards
- Use Case Studies
What happens in an internet minute?

2016

- 701,389 Facebook logins
- 150 million Emails Sent
- 2.78 million Video Views
- 2.4 million Search Queries
- 972,222 Snapchats
- 38,102 hours of Music
- 104 million Vine loops
- 347,222 New Tweets
- 120 new LinkedIn accounts

1.3X

2017

- 900,000 logins
- 16 million text messages
- 4.1 million videos viewed
- 342,000 apps downloaded
- 751,522 spent online
- 347,222 new tweets
- 1.8 million snaps created
- 50 voice-first devices shipped

1.3X

Created By: @OfficiallyChadd

@LoriLewis
BC/AD in IT

Source: Human Computer Interaction & Knowledge Discovery

Structured Data

Unstructured Data

IDC and EMC project that data will grow to 40 ZB by 2020.

Before Cloud

Anno Datum

SDC 17

2017 Storage Developer Conference

All Rights Reserved.
Everything is object!

Block VS Object

OSD Object Storage
- ID
- Attributes
- User Data

KV Storage
- Key
- Value
Key Value Stores are Common
Key Idea

Host S/W

Block Device Driver

Block Device

Traditional KV Store

Thin KV Library

TX/s

WAF, RAF, Latency

KV Device Driver

KV Device

KV Stacks

2017 Storage Developer Conference. ©
Samsung KV-PM983 Prototype

NGSFF KV SSD

Form factor: NGSFF/U.2
Capacity: 1-16TB
Interface: NVMe PCIe Gen.3
Key Value SSD is a Scalable Solution

- Performance
- Capability

KV SSD

Scale-Up
- Capacity
- Performance

Scale-In
- CPU
- Server

Scale-Down
- TCO
- Power

Scale-Out
- Capacity
- Performance
KV SSD Ecosystem

Key Value SSD

- Standard
- Product
- Applications
- SDK
- Partners

Software logos include:
- RocksDB
- MongoDB
- redis
- AllCeph
- Linux
- Windows

Brands included:
- SNIA
- nvm EXPRESS
- Samsung
Key Value SW Stacks

- SSD with native key value interface through hardware software co-design

### Datacenter S/W Infra
- Storage Plugin Interface
- Key Value Glue Logic

#### Key Value API
- Index
- S/W Key Value Store
- Log
- POSIX API
- Block Map
- File System
- Journal

#### Block Interface
- Block Device Driver

#### Command Protocol
- Map
- Block Device
- Log

### Datacenter S/W Infra
- Storage Plugin Interface
- Key Value Glue Logic

#### Key Value API
- Thin KV Library

#### KV Interface
- KV Device Driver

### TX/s vs WAF, RAF, Latency

- Index
- KV Device
- Log
KV SSD Design Overview

- SSD that supports native key value commands

**Storage Server**

- Read/Write User Data
- Key Size Range ?
- Value Size Range ?
- Key Size
- Value Size
- Key Value I/F Command
- Get (key) / Put (Key, Value)
- Key Value SSD device driver

**Key Value SSD**

- User/Device Hash Key
- Physical Location / Offset
- Index
- NAND Page (32KB)
- Meta data
- Key
- Value

Lookup / Check hash collision
### Key Value Software Development Stacks

#### Key Value Library & Tools

<table>
<thead>
<tr>
<th>Cache</th>
<th>AIO</th>
<th>Multi-Queue</th>
<th>Multi-Device</th>
<th>Memory Manager</th>
<th>Tools</th>
</tr>
</thead>
</table>

#### KV Abstract Device Interface (ADI)

<table>
<thead>
<tr>
<th>store/retrieve/delete/exist</th>
<th>KV Pair</th>
<th>namespace</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Linux Kernel Device Driver</th>
<th>Linux User-space Device Driver</th>
<th>Windows Device Driver</th>
</tr>
</thead>
</table>
Application Integration with KV SSD

NoSQL DB
- MongoDB
- RocksDB
- Redis

Distributed DB
- KV Stacks
- KV Device

Object Storage System
- Swift API
- OSD

KV Adapter

KV Stacks

KV Device

Storage Engine

API
Key Value SSD Layers

Application(s)

KV API

SNIA KV Library

KV Protocol Client Interface

KV Wire Protocol

KV Protocol Provider Interface

KV Device

e.g. SNIA KV API

e.g. C Library, Java, etc. Written by vendors, open source, etc.

e.g. NVMe KV commands

e.g. KV SSD

All Rights Reserved.
Key Value SSD Standard Activities

- NVMe
  - Work on a technical proposal is being discussed by the NVMe working group
  - The group is defining the scope of the work
  - This will be a new device type

- SNIA
  - A proposal for a Key Value API has been submitted to the SNIA Object Drive Technical Working Group
  - Discussion on the minimum necessary commands to meet basic Key Value needs is progressing
Key Value, not Object Drive

- Both standards efforts are focused on Key Value SSD not Object Drive
  - Key Value is a means to submit a Key and put or get a Value
  - Object Drive would include more extensive commands to query the Key Value database
### NVMe Extension for Key Value SSD

- Defines a new device type for a Key Value device
- A controller performs either KV or traditional block storage commands

<table>
<thead>
<tr>
<th>New Key Value Commands</th>
<th>PUT</th>
<th>GET</th>
<th>DELETE</th>
<th>EXISTS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin command</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Identify commands for KV</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Other non-block specific commands</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Existing Command Extension</th>
<th>Admin command</th>
<th>Identify commands for KV</th>
<th>Other non-block specific commands</th>
</tr>
</thead>
</table>
SNIA Key Value API

- The Key Value API (Application Programming Interface) has been presented to SNIA for consideration in the Object Drive Technical Working Group.
- Defines a Tuple
  - Key
  - Value
- Defines KV specific constants
  - Max Key Length
  - Alignment Unit
- Key type supported
  - 4 byte fixed
  - 8 byte fixed
  - Variable length character string
  - Variable length binary string
- The API defines the calls that an application may make to the Key Value device interface
  - These calls are independent of any specific implementation
  - These calls support the basic commands proposed for the NVMe standard
    - Open/Close
    - Store/Retrieve
    - Exist
    - Delete
    - Containers/groups
Call for Participation

- NVMe work is proceeding in the NVMe working group
  - [www.nvmexpress.org](http://www.nvmexpress.org)
    - Contributors and Promoters have access to working proposals
- SNIA work is proceeding in SNIA Object Drive Technical Working group
  - [www.snia.org](http://www.snia.org)
    - Members may join the Object Drive TWG and have access to working proposals
Use Case Studies
Use Case Study

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>KV Bench</th>
<th>Scale-Up</th>
<th>Scale-Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scale-Up</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Scale-Out</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Key Value Store

- RocksDB
- KV Stacks

Device

- SDC
- NVMeoF
- Samsung
Single Component Performance: RocksDB vs. KV Stacks

- RocksDB
  - Originated by Facebook and Actively used in their infrastructure
  - Most popular embedded NoSQL database
  - Persistent Key-Value Store
  - Optimized for fast storage (e.g., SSD)
  - Uses Log Structured Merge Tree architecture

- KV Stacks on KV SSD
  - Benchmark tool directly operates on KV SSD through KV Stacks
RocksDB: Key Value Database
RocksDB vs. KV Stacks Performance Measurement

- **Better Performance**
  - Lean software stacks
  - Overhead moved to device

- **IO Efficiency**
  - Reduction of host traffic to devices

### Hardware
- Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz
- 96 GB RAM
- PM983(Block) & KV-PM983 SSD

### Software
- Ubuntu 16.04
- RocksDB v5.0.2 on XFS
- 50M records, 16B Key, 4KB value

---

Client: kvbench

Block SSD vs. KV SSD

RocksDB

Filesystem

Block Driver

KV SSD

Key Value API

Key Value ADI

KV Driver

PM983

KV-PM983

SAMSUNG
Performance: Random PUT

- 8x more QPS (Query Per Second) with KV Stacks than RocksDB on block SSD
- 90+% less traffic goes from host to device with KV SSD than RocksDB on block device
Testbed System for Scaling

- CPU: Xeon 8160 CPU @ 2.10GHz
- 1-node 2-socket, 48-core 96-thread
- # SSDs: 30x 1TB

Full Box: 7.8 MIOPS
- Fully Balanced Architecture
- No PCIe bottle Neck

Fully utilize the Skylake CPU and PCIe switch bandwidth
Scale-Up Storage: RocksDB

- **Linear Scaling**
  - More devices, more throughput and capacity

- **IO Efficiency**
  - Reduction of host traffics to devices

- **Less CPU utilization**
  - Small number of cores or less CPU utilization for performance
Scale-up Performance: Random Key PUT

- **15x IO performance over S/W key value store on block devices**

Relative performance to the maximum aggregate RocksDB random Put QPS for 1 SSD with a default configuration for 1 PM983 SSD in a clean state. System: Ubuntu 16.04.2 LTS, Ext4, RAID0 for block SSDs, Actual CPU utilization could be 70-90% at CPU saturation point.

Workload: 100% puts, 16 byte keys of random uniform distribution for RocksDB v. 5.0.2, 4KB-fixed values, 24 RocksDB instances with 4 client threads, 50GB/Instance or 1.2TB Data is used.
Scale-up Performance: Sequential Key PUT

- **3.4x** IO performance over S/W key value store on block devices

**Graph 1:**
- Relative QPS vs. number of SSDs
- **RocksDB (PM983)** vs. **KV Stacks (KV-PM983)**
- **3.4x** IO performance over S/W key value store on block devices

**Graph 2:**
- Device IO/User IO
- **2.0** for RocksDB (PM983)
- **1.0** for KV Stacks (KV-PM983)

**Notes:**
- Relative performance to the maximum aggregate RocksDB random Put QPS for 1 SSD with a default configuration for 1 PM983 SSD in a clean state.
- System: Ubuntu 16.04.2 LTS, Ext4, RAID0 for block SSDs, Actual CPU utilization could reach 85%-90% at CPU saturation point.
- Workload: 100% puts, 16 byte keys of random uniform distribution for RocksDB v. 5.0.2, 4KB-fixed values, 36 RocksDB instances with 1 client thread, 34GB/Instance or 1.2TB Data is used.
Scale-Out: RocksDB & KV Stacks Configuration

RocksDB vs KV Stacks

NVMeoF over RDMA

Mission Peak KV-PM983 SSDs
Local vs NVMeoF PUT Latency

Average Latency

@Qdepth: 1-8
Overhead: 4-7us

Microseconds

Queue Depth

0 200 400 600 800 1000 1200

0 1 2 4 8 16 32 64 128

RDMA Switch

kvbench

KV Stacks

ADI + KV User Driver

RDMA Switch

@Qdepth: 1-8
Overhead: 4-7us
Performance and Capacity Scale-Out: PUT Throughput

```
Client RocksDB: CentOS 7.3, Ext4, RAID0 for block SSDs, Workload: 100% puts, 16 byte keys of random uniform distribution for RocksDB, 4KB-fixed values, 24 RocksDB instances with 8 client threads, 50GB/Instance or 1.2TB Data is used.
Client KV Stacks: CentOS 7.3, KV Load Generator, 100% 4K PUTs, 16 byte keys.
KV Server: Mission Peak w/ NVMeoF KV Target
```
CPU Utilization for Clients

Fill Random

Fill Sequential

KV Stacks

Avg 170K QPS@72% CPU

Avg 400K QPS@80% CPU

Avg 2.1M QPS@30% CPU

2.1 M QPS
Conclusion

- Linear performance and capacity scaling
- TCO reduction
- CPU or server reduction
- Dense performance and capacity scaling
- Lean host software stacks
Questions?

kvssd@ssi.samsung.com