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

09/14/2017
Yang Seok Ki
Director of Memory Solutions Lab
Samsung Semiconductor Inc.
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.
Agenda

• Cloud: A New Era
• Scalability: A New Challenge
• Key Value SSD: A New Technology
  – Samsung Key Value SSD
• Ecosystem
• Use Case and Performance Studies
• Q&A
Agenda

• Background
• Concept
• Key Value SSD
• Ecosystem
• Use Case and Performance Studies
• Standards
• Q&A
Cloud: A New Era & Challenges
What happens in an internet minute?

2016

- Facebook: 701,389 Facebook logins
- WhatsApp: 20.6 million messages
- YouTube: 2.78 million video views
- Tinder: 972,222 Swipes
- Spotify: 1.04 million Vine loops
- Vine: 38,194 posts to Instagram
- LinkedIn: 347,222 new tweets
- Amazon: 1,389 Uber rides
- Netflix: 69,444 hours watched
- Email: 150 million emails sent
- Snapchat: 527,760 photos shared
- Google: 38,052 hours of music
- YouTube: 2.4 million search queries
- Spotify: 38,194 posts to Instagram
- Amazon: 2.4 million search queries
- Apple: 1.389 Uber rides
- Netflix: 701,389 Facebook logins

2017

- Facebook: 900,000 logins
- Google: 3.5 million search queries
- YouTube: 70,017 hours watched
- Netflix: $751,522 spent online
- Instagram: 4.1 million videos viewed
- Snapchat: 342,000 apps downloaded
- Vine: 46,200 posts uploaded
- Tinder: 452,000 tweets sent
- Spotify: 1.8 million snaps created
- Apple: 1.5 million GIPs sent via Messenger
- Amazon: 15,000 apps downloaded
- LinkedIn: 120 new accounts created
- Vine: 50 voice-first devices shipped
- Spotify: 40,000 hours listened
- Amazon: 156 million emails sent
- Tinder: 990,000 swipes
Challenges in Cloud Era

- Scalable TCO
- Capacity
- Throughput
- Power
- Latency
A driver (host) is responsible for parking (data management)
Object: Valet Parking

- A parking facility (storage) is responsible for parking (data management)
Key Value SSD: New Scalable Technology
Everything is object!

Block vs Key Value

OSD Object Storage
- ID
- Attributes
- User Data

KV Storage
- Key
- Value
Key Value Stores are Common in Systems at Scale
Key Value in Systems at Scale: Twitter Timeline Service

Write API

Fanout

twemproxy

Followers

Key Value Store
Key Idea

Key Value Store is everywhere!

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
Samsung KV-PM983 Prototype

NGSFF KV SSD

Form factor: NGSFF/U.2
Capacity: 1-16TB
Interface: NVMe PCIe Gen.3
KV SSD Design Overview

- **Key/Value Range**
  - Key: 4~255B
  - Value: 64B~2GB (32B granularity)
  - The large value is stored into multiple NAND pages
Key Value SSD is a Scalable Solution with Better TCO

- **Performance**
- **Capability**

**Scale-Up**
- **Capacity**
- **Performance**

**Scale-In**
- CPU
- Server

**Scale-Down**
- TCO
- Power

**Scale-Out**
- Capacity
- Performance

TCO($)

---

[Image: Collaborate. Innovate. Grow.]

18
KV SSD Ecosystem
Ecosystem in Block and KV Device Era

Block SSD

S/W KV Store

Database

Filesystem

SNS

Cache

CDN

Learning

E-commerce

Cache

KV SSD

KV Stacks

Database

Cloud

CEDR

NetApp

Ceph

LinkedIn

Facebook

Twitter

Pinterest

redis

Caffe2

SAMSUNG

COLLABORATE. INNOVATE. GROW.
KV SSD Ecosystem

Key Value SSD

- Partners
- SDK
- Applications
- Product
- Standard

Supporting technologies and platforms:
- Linux
- Windows
- RocksDB
- MongoDB
- redis
- Ceph

Applications for KV SSD

NoSQL DB

- KV Device
- KV Stacks
- KV Adapter
- API

Distributed DB

- KV Device
- KV Stacks
- KV Adapter
- Storage Engine
- API

Object Storage System

- KV Device
- KV Stacks
- Storage Engine
- Swift API
- OSD
- API
KV SSD Ecosystem

Key Value SSD

- Standard
- Product
- Partners
- SDK
- Applications

- SNIA
- Invitation to Party
- nvm EXPRESS
- Samsung

- Linux
- Windows
- RocksDB
- MongoDB
- redis
- ceph
Key Value SW Stacks

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

![Diagram showing Key Value SW Stacks with 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, and Log. On the right side, it shows Datacenter S/W Infra, Storage Plugin Interface, Key Value Glue Logic, Key Value API, Thin KV Library, TX/s, WAF, RAF, Latency, KV Interface, KV Device Driver, Command Protocol, Index, KV Device, and Log.]
### 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)

- store/retrieve/delete/exist
- KV Pair
- namespace

<table>
<thead>
<tr>
<th>Linux Kernel Device Driver</th>
<th>Linux User-space Device Driver</th>
<th>Windows Device Driver</th>
</tr>
</thead>
</table>
kvbench: Key Value Benchmark Suite

Workload Generator

- RocksDB
- WiredTiger
- ForestDB

Performance Profiler

- KV API/Lib
- KV ADI
- KV Device Driver

Filesystem

Block Device Driver

Block Device

KV Device
KV Virtualization

Application

KV Virtualization

Capacity Management  Key Space Distribution  Load Balancing

KV devices
Key Value Software Development Stacks

Key Value Library & Tools

- Cache
- AIO
- Multi-Queue
- Multi-Device
- Memory Manager
- Tools

KV Abstract Device Interface (ADI)

- store/retrieve/delete/exist
- KV Pair
- namespace

- Linux Kernel Device Driver
- Linux User-space Device Driver
- Windows Device Driver
Key Value SSD Use Case Studies
Use Case Study

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>Single</th>
<th>Scale-Up</th>
<th>Scale-Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>KV Stacks</td>
<td>KV Bench</td>
<td>RocksDB</td>
<td>KV Stacks</td>
</tr>
<tr>
<td>Key Value Store</td>
<td>Single</td>
<td>Scale-Up</td>
<td>Scale-Out</td>
</tr>
<tr>
<td>Device</td>
<td>Single</td>
<td>Scale-Up</td>
<td>Scale-Out</td>
</tr>
</tbody>
</table>
Use Case Study

<table>
<thead>
<tr>
<th>Device</th>
<th>Key Value Store</th>
<th>Benchmark</th>
<th>Scale-Up</th>
<th>Scale-Out</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NVMeoF</td>
<td>KV BENCH</td>
<td>KV BENCH</td>
<td>KV BENCH</td>
</tr>
<tr>
<td></td>
<td>KV STacks</td>
<td></td>
<td>RocksDB</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>NVMeoF</td>
</tr>
</tbody>
</table>
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 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

<table>
<thead>
<tr>
<th>Block SSD</th>
<th>KV SSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>RocksDB</td>
<td>Key Value API</td>
</tr>
<tr>
<td>Filesystem</td>
<td>Key Value ADI</td>
</tr>
<tr>
<td>Block Driver</td>
<td>KV Driver</td>
</tr>
</tbody>
</table>

PM983 | KV-PM983
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

* Workload: 100% random put, 16 byte keys of random uniform distribution, 4KB-fixed values on single PM983 and KV-PM983 in a clean state
Use Case Study

<table>
<thead>
<tr>
<th>Device</th>
<th>Single</th>
<th>Scale-Up</th>
<th>Scale-Out</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmark</td>
<td></td>
<td>KV Bench</td>
<td></td>
</tr>
<tr>
<td>Key Value Store</td>
<td></td>
<td>RocksDB vs KV Stacks</td>
<td></td>
</tr>
<tr>
<td>Device</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
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

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 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.
## Use Case Study

<table>
<thead>
<tr>
<th></th>
<th>Single</th>
<th>Scale-Up</th>
<th>Scale-Out</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Benchmark</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>KV Bench</td>
<td></td>
</tr>
<tr>
<td><strong>Key Value Store</strong></td>
<td></td>
<td>RocksDB</td>
<td>KV Stacks</td>
</tr>
<tr>
<td></td>
<td></td>
<td>VS</td>
<td></td>
</tr>
<tr>
<td><strong>Device</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Scale-Out: RocksDB & KV Stacks Configuration

RocksDB vs KV Stacks

NVMeoF over RDMA

Mission Peak KV-PM983 SSDs

Software CentOS 7.3 w/ KV SW
NIC 2x 100GbE
CPU Xeon E5-2699V4 CPU @2.20GHz
1-node 2-socket
44-core 88-thread
DRAM 256 GB
Local vs NVMeoF PUT Latency

Average Latency

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

Fill Random

Scaling w/ 2 KV Servers

- KQPS: 9.4M
- Scaling: 13.8X
- # of KV Clients: 1 - Client, 2 - Client, 3 - Client, 4 - Client

Fill Sequential

Scaling w/ 2 KV Servers

- KQPS: 9.4M
- Scaling: 5.87X
- # of KV Clients: 1 - Client, 2 - Client, 3 - Client, 4 - Client

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

Avg Utilization 10% Higher

2.1 M QPS
Key Value SSD Standards

Bill Martin
Principal Engineer
Memory Solutions Lab
Key Value SSD layers

Application(s)

KV API

e.g. SNIA KV API

SNIA KV Library

KV Protocol Client Interface

KV Wire Protocol

e.g. NVMe KV commands

KV Protocol Provider Interface

KV Device

e.g. KV SSD

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

**New Key Value Commands**
- PUT
- GET
- DELETE
- EXISTS

**Existing Command Extension**
- Admin command
- Identify commands for KV
- Other non-block specific commands
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
    • Contributors and Promoters have access to working proposals

• SNIA work is proceeding in SNIA Object Drive Technical Working group
  – www.snia.org
    • Members may join the Object Drive TWG and have access to working proposals
Key Value SSD is a Scalable Solution with Better TCO

- 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