Flash Memory Summit

Architecture of a query accelerating KVCSD in a HPC System

Woosuk Chung,

Director, Memory Forest x&D, SK hynix Inc.

Legal Disclaimer

The information contained in this document is claimed as property of SK hynix. It is provided with the understanding that SK hynix assumes no liability, and the contents are provided under strict confidentiality.

This document is for general guidance on matters of interest only. Accordingly, the information herein should not be used as a substitute for consultation or any other professional advice and services.

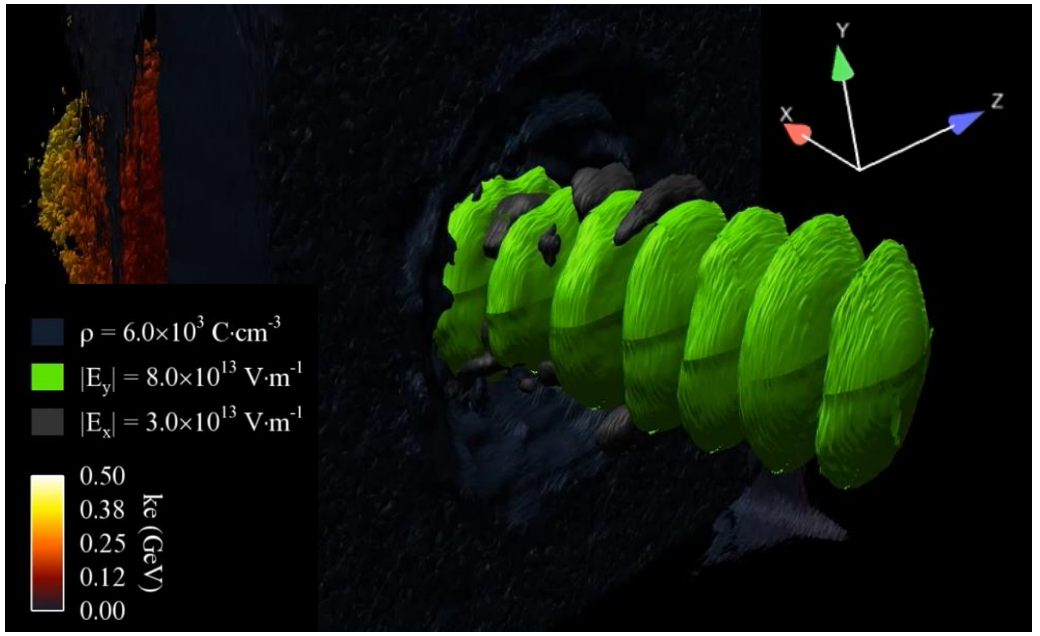
SK hynix may have copyrights and intellectual property right. The furnishing of document and information disclosure should be strictly prohibited.

SK hynix has right to make changes to dates, product descriptions, figures, and plans referenced in this document at any time. Therefore the information herein is subject to change without notice.

Recent Trends: Analysis is being complicated

- Data growth is triggering the rapid increase of highly selective data queries.
  - Queries tend only to find a small subset of data.
  - Seeing all of a simulation's data output fetches too many unnecessary data.
  - Overhead increases as data grows.

- Increasing cases to analyze using various value fields.
  - User wants to find refined information to make analysis more meaningful.



* Source: Los Alamos National Laboratory "Ordered Key-Value Computational Storage Device Video" Flash memory summit, 2022

Example : Tracing the state of few high-energy particles
SELECT X, Y, Z FROM particles **WHERE** Ke >= 1.5

Background: Scientific Simulation & Analysis

- HPC application performs scientific simulation and then performs analysis.
  - Simulation produces petabytes of data and trillion records.
  - Analysis finds meaningful data in a specific value range.

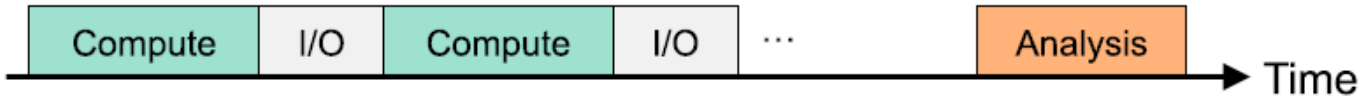
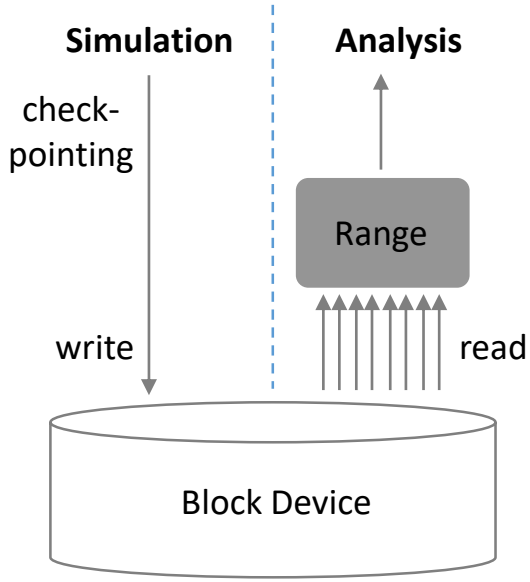


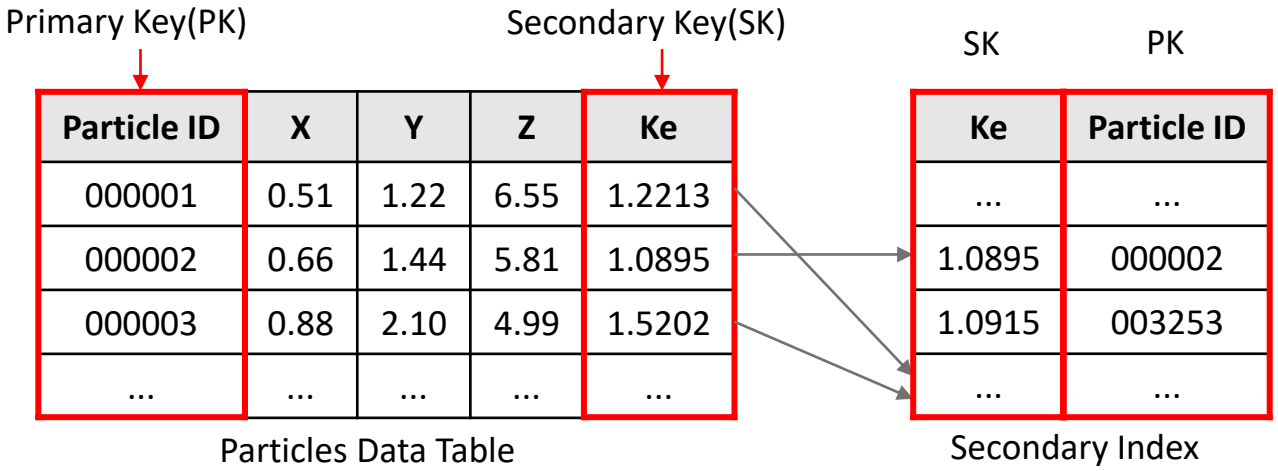
Illustration of a typical scientific workflow consisting of a bulk-synchronous parallel simulation application acting as a data writer and a subsequent data analysis program acting as a data reader with the execution of the writer further divided into iterations of non-overlapping compute and I/O phases.

* Source: “Streaming Data Reorganization at Scale with DeltaFS Indexed Massive Directories”, ACM Trans. Storage, 2020



Background: Secondary Index

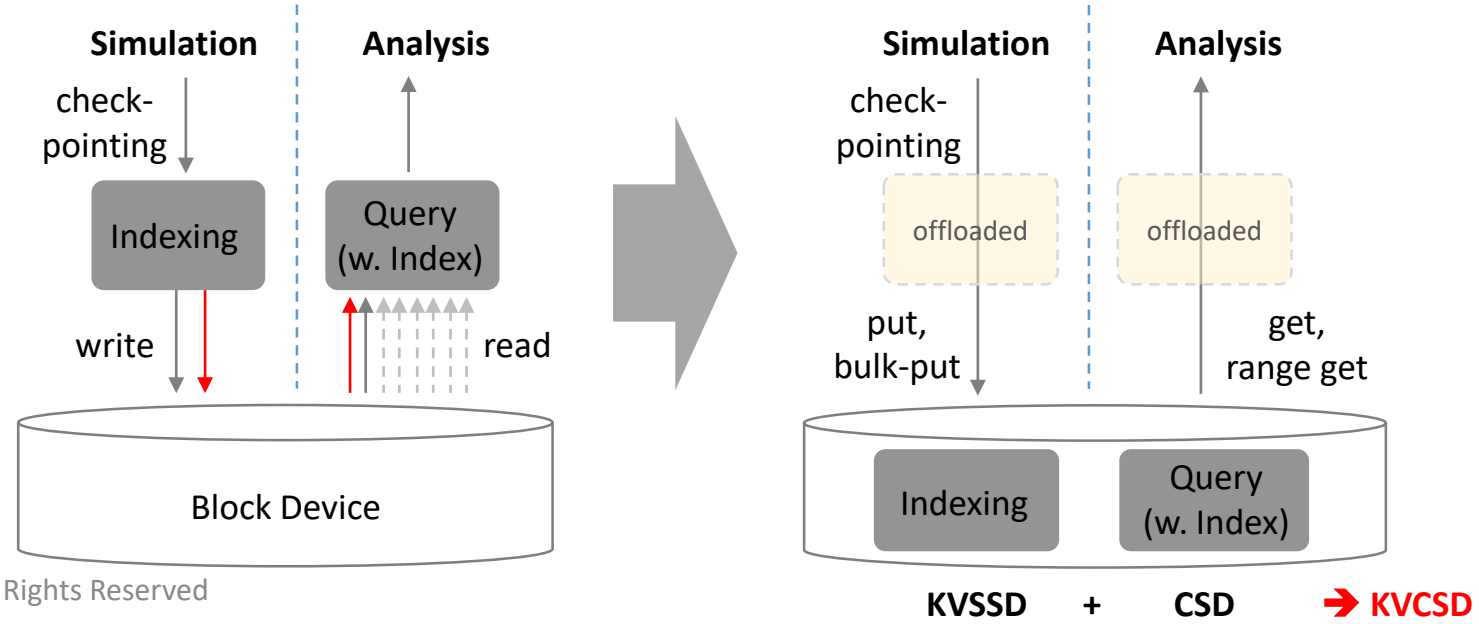
- Access records using value attributes other than primary key.
  - Secondary indexes are built using the target attributes you want to search.
- Additional data structure is created to support secondary index.
  - Building secondary indexes require additional resources and time.
- Secondary index improves query performance and reduces storage overhead.
  - Secondary index can reduce the amount of data that needs to be scanned.



Able to query using “Ke” attributes.
SELECT X, Y, Z FROM particles **WHERE** Ke >= 1.5

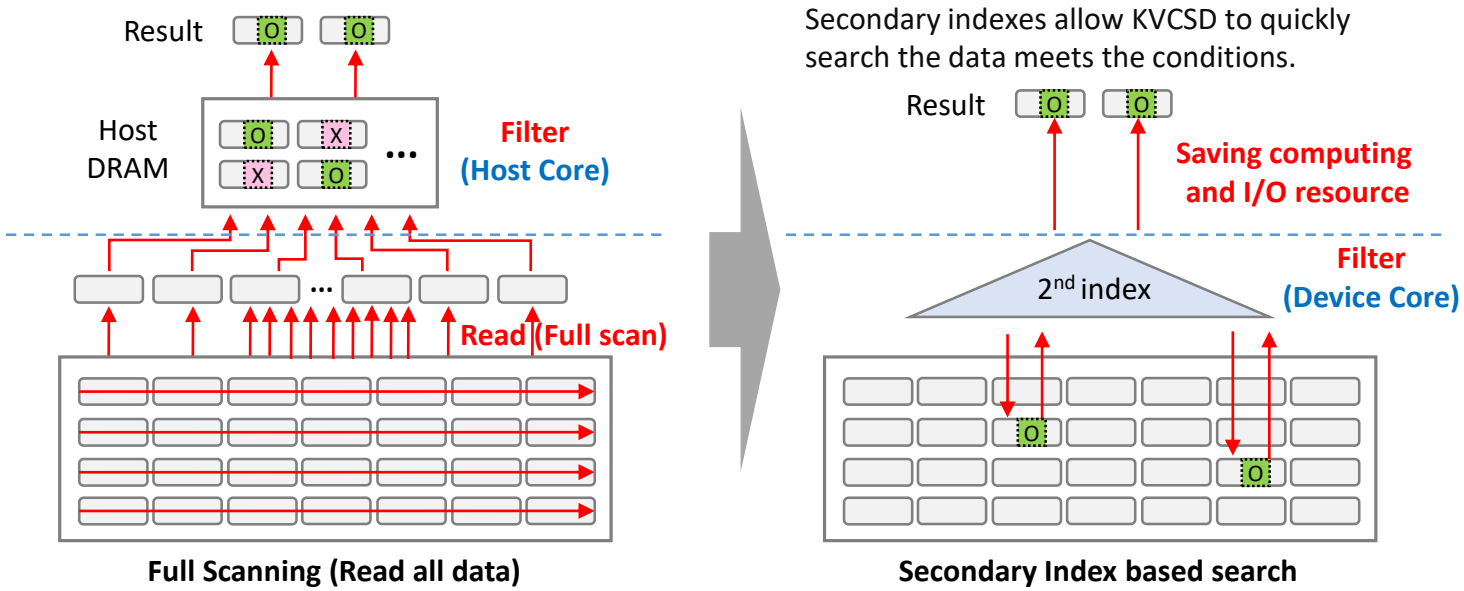
Our Goal: High Performance KVCSD for Analytics

- Called Key Value Computational Storage Drive (KVCSD).
  - KVCSD supports basic Key-Value operations and Queries. (KVSSD).
  - KVCSD offloads computations including indexing and query processing to the device (CSD).
- Offloading both primary and secondary index operations to the device.
  - Achieves high I/O performance by eliminating index overheads on host.
  - Reduces data movement and accelerates query performance.



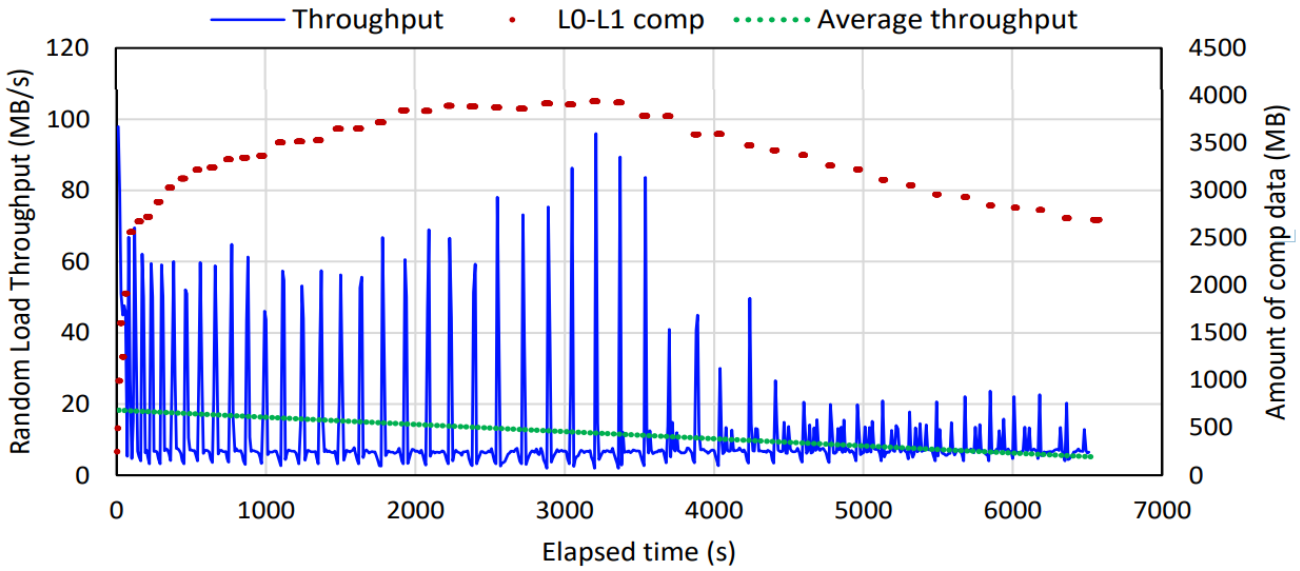
Motivation: Reducing the burden of host

- To find the data that satisfies a range of value attributes...
  - Full Scan : Access, transfer all the data to the host then filtering.
  - Secondary Index : Finds only data within range in tiny secondary index and reads only the data you need.
- Full scan takes up too much expensive host resources.
- On the other hand, using secondary index in KVCSD...
  - Just call the secondary index API and get the results. Everything is handled in KVCSD.
- The more data you have, the better the effect of secondary index.



Motivation: Reducing the burden of host

- Write optimized LSM key-value store dynamically transform data using compaction.
  - This key-value store must run inside an application process with limited host resource.
  - Due to processing compaction, there are cases where inserts are blocked, leading to long I/O delays.
- KVCSD allows compaction to be deferred and processed asynchronously in the device.
  - Hiding compaction latency from applications and preventing bottlenecks such as write stall.
  - User only needs to experience write time, with compaction deferred and offloaded to the KVCSD.



RocksDB's random write performance and L0-L1 compactions. The blue line shows the random write throughput. In the graph, the blue line has zero values in some areas because inserts are blocked by write stalls. Write stall occurs when the host CPU becomes a bottleneck due to compaction.

* Source: "MatrixKV: Reducing Write Stalls and Write Amplification in LSM-tree Based KV Stores with Matrix Container in NVM", USENIX ATC, 2020

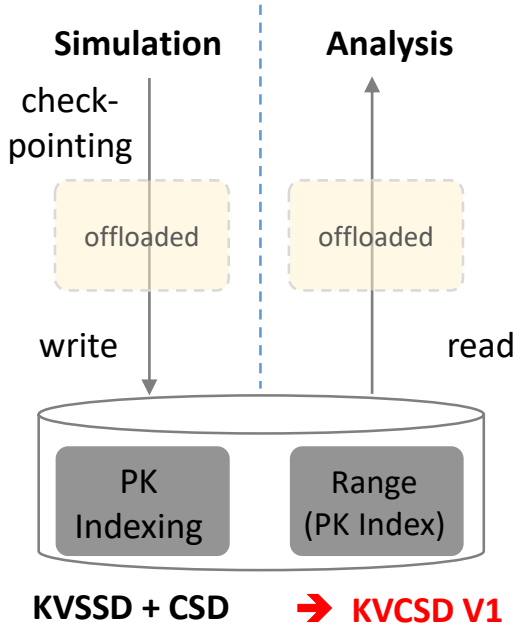
Recap: KVCSD Version1 – FMS`22

- Adopt fully ordered data structure(LSM) and offer ZNS optimization.
  - Optimizes I/O and search performance by sorting data within a zone.
  - Simplify internal storage management and reduce garbage collection overheads with ZNS.

- Offloading Indexing and query processing to the device.
  - Improve performance by minimizing host processing and data transfer.
  - Better performance compared to RocksDB: Write x3, Read x2, Bulk Store x4.7.

- Supports “Range Query” which shows much better performance.
  - Range read can read all data within a specific range with a single command.
  - About 40% performance improvement compared to RocksDB.

- KVCSD V1 can only find the data by key index (Primary key).



Overview: SK hynix KVCSD Features

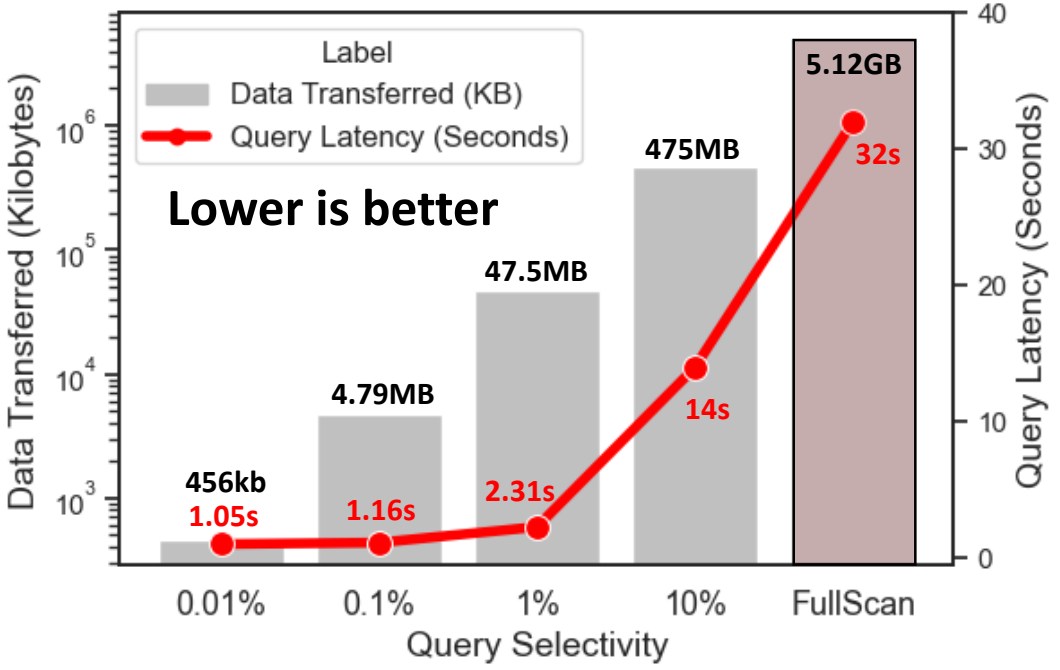
<KVCSD Prototype>

- Key-Value Interface
  - Eliminate duplicate layer and fine-grained management.
- Analytics Query Acceleration
  - Support Range Query, 2nd Index, and Histogram.
- LSM Data Structure
  - Offload sorting data and building indexes into KVCSD.
- ZNS Optimization
  - Improve Write/Read amplification and remove GC.

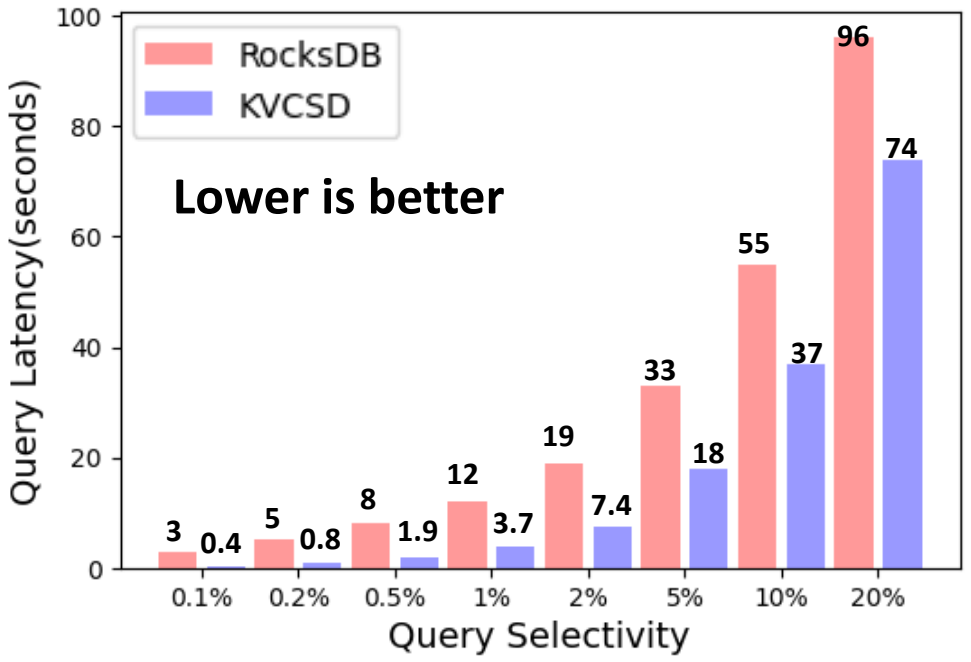
Evaluation: Boosted Performance with KVCSD

- KVCSD V2 shows up to 32X faster compared with KVCSD V1.
- KVCSD V2 is faster than current state-of-the-art software-based key-value stores.
  - Up to 7.4x faster than RocksDB with much less use of host resources.

KVCSD V1 vs KVCSD V2



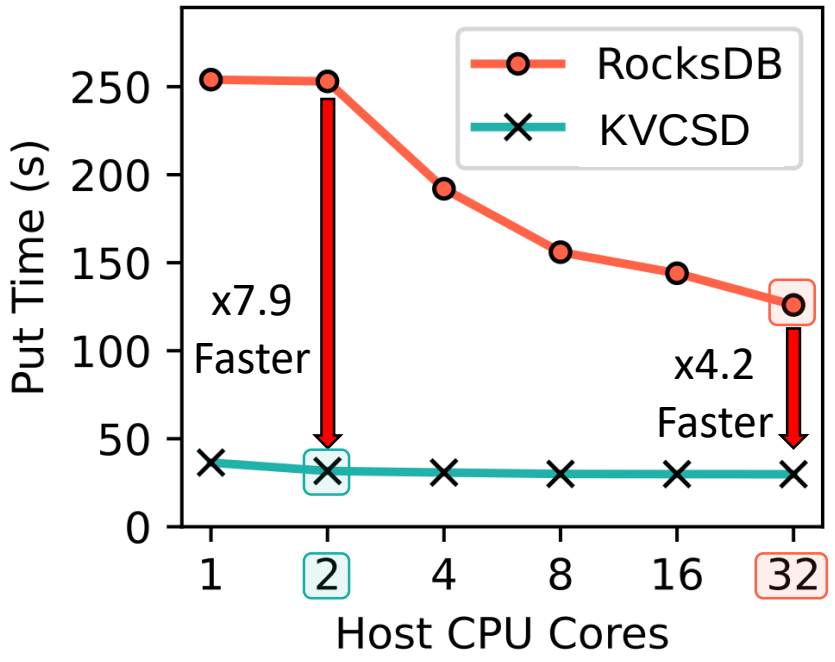
KVCSD V2 vs Rocks DB with 2nd index



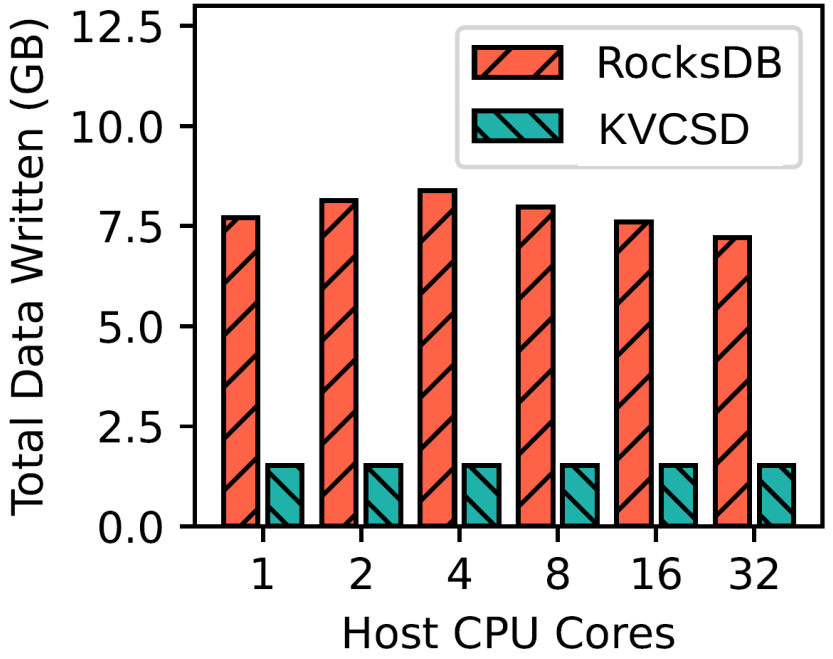
Evaluation: Saturated Performance with fewer core

- RocksDB requires **32** dedicated host CPU cores to maximize performance.
- KVCSD only needs **2** CPU cores to reach peak performance.
- KVCSD also reduces total data written resulting in a reduced host burden.

Put Time(32M KV pair) Comparison



Total Data Written Comparison

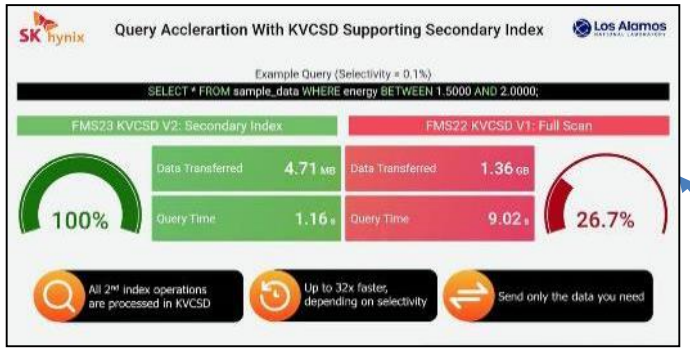


Summary

- Scientific simulation and analysis
  - Large-scale data analytics is a key element of scientific discovery.
  - Data grows rapidly, data queries are being highly selective, host overhead increases.
  - Indexes accelerate analysis, but host-side indexing degrades overall simulation performance.
  - Computational storage technology present a unique opportunities to overcome these challenges.

- In-drive, indexing, query accelerating KVCSD
  - Offload both primary, secondary indexing and query processing to KVCSD.
  - Reduce the burden on host compute resources and enable highly fast, efficient data queries.
  - KVCSD enriches the analysis using multiple secondary indexes.
  - Adopt ordered data structure and offer ZNS optimization.

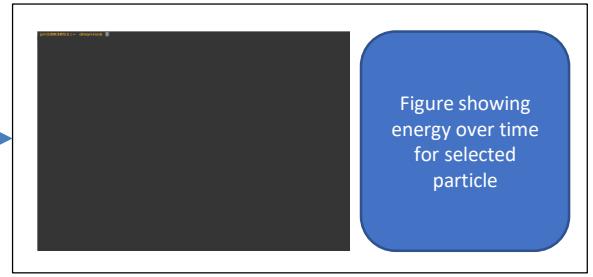
Co-demonstration with Los Alamos National Lab



Video Demo
(SKH & LANL Video)

SKH Live Demo

LANL Live Demo



SKH Benchmark Application

SKH KVCSDs

SKH Demo Server

LANL Demo Server

Shelf

LANL VPIC Application

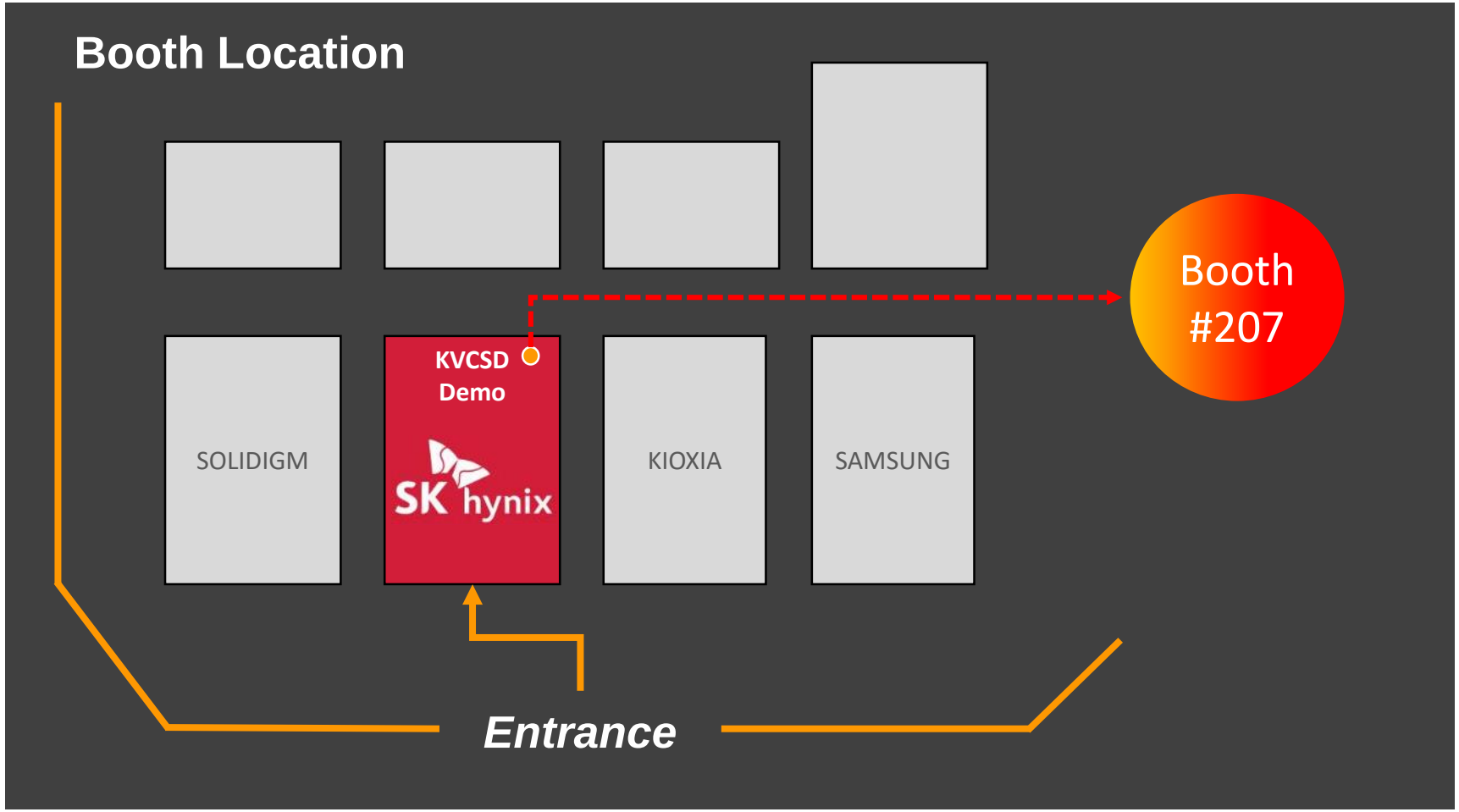
SKH KVCSDs

SKH Demo compares the performance of KVCSD V1(Full scan) and KVCSD V2(Secondary Index) with typical query used in real-world scientific applications.

In video demo, SKH describes KVCSD main features and its effects. LANL introduces VPIC and describe the need for KVCSD and its effect.

LANL Demo simulates real-world scientific data, VPIC, and shows query performance using secondary index.

Learn more about SK hynix



Visit Booth #207 and Experience SK hynix products and demos