Building True Computational Storage

Sebastien Jean, Phison Electronics Corporation

Computational Storage Drives (CSD)

- General CSD Objectives
  - Reduce data movements
  - Reduce processing time
  - Offload the host

- Basic CSD Use Cases
  - Compression (Available on most Enterprise SSD)
  - Encryption (Available on most Enterprise SSD)
  - Patter Search
  - Data Deduplication

- Advanced CSD Use Cases
  - Implementing a Key Value Interface
  - Video Transcoding

CSD & Fault Tolerance

Flash Memory Summit

- Common Fault Tolerance Types
  1. Individual Drives w/ failover pool
  2. RAID 5,6,7 or Erasure Coding
  3. Raid 1 or Replication

Type	Data Elements	Cost	Redundant
1	Complete	Low	No
2	Fractional	Mid	Yes
3	Complete	High	Yes

- Basic CSD Use Cases
  - Compression (Available on standard Enterprise SSD)
  - Encryption (Available on standard Enterprise SSD)
  - Patter Search
  - Data Deduplication

Type	Data Elements	Compression	Encryption	Pattern Search	Dedup
1	Complete	Yes	Yes	Yes	Yes
2	Fractional	Yes	Yes	No	No
3	Complete	Yes	Yes	Yes	Yes

- Advanced CSD Use Cases
  - Implementing a Key Value Interface
  - Video Transcoding

Type	Data Elements	KV Interface	Video Coding
1	Complete	Yes	Yes
2	Fractional	No	No
3	Complete	Yes	Yes

- The more cost-effective RAID and Erasure Coding environments add significant complexity

CSD & Compute Power

- Non-floating point
  1. Enterprise CPU 500K (typ) to 2,400K MIPS (max)
  2. SSD Core 2K to 5K MIPS (1% at best and 0.1% at worse)

- Floating point
  1. GPU 20,000 TFLOPS
  2. Advanced SSD Core 0.05 TFLOPS (0.03%)

- It is possible to add more SSD guest cores to increase the capability

- Though compute does scale with each CSD added, it's important to take a deep look at the ROI

- SSD wear out, so how much additional cost per unit can your business tolerate?

Taking another look at efficiency

- Broader Objective
    1. Reduce data movements
    2. Reduce processing time
    3. Offload the host

  > Increase compute efficiency

- There are use-cases that benefit from processing data element on the SSD
- Adding cost to the solution greatly limits the adoption rate
- Alternative - allow compute and storage focus on their key strengths, but improve efficiency

SSD is a memory peer

- PCIe accelerated in 2018 after 7 years of stagnation

DRAM Channel
DDR3 @ 6 – 17 GB/s
DDR4 @ 12 – 25 GB/s
DDR5 @ 38 – 57 GB/s

Level 1

Industry Acceptance

CXL

Longer Latency

CXL DRAM Expansion
Gen 4x4 @ 7 GB/s
Gen 5x4 @ 14 GB/s
Gen 6x4 @ 28 GB/s

Level 2

Industry Acceptance ???

CXL

Longer Latency

SSD Channel
Gen 4x4 @ 7 GB/s
Gen 5x4 @ 14 GB/s
Gen 6x4 @ 28 GB/s

Level 3

- NVMe SSD are now memory peers to DDR and CXL DRAM Expansion
  - Enterprise DDR bandwidth 57 GB/s * 8 = 456 GB/s
  - Enterprise PCIe bandwidth 7 GB/s * 128 = 896 GB/s

- Remove overhead of file system, drivers and protocol → use MemRd & MemWr

- Adding a secondary CXL interface to an SSD namespace allows CPU, GPU, CXL DRAM, and SSD to communicate with minimal overhead while maintaining coherency

Summary

- Computational storage is about increasing efficiency
- There are use-case that benefit from compute directly on the SSD
- There are cases where that need a different solution
    1. Very large data sets
    2. Sensitive to SSD cost or power
    3. Replication is not feasible
    4. Very high compute requirements
- A CXL SSD enables coherent computational efficiency at scale

