Standardizing Zones Across the Storage Ecosystem

Sponsored by NVM Express™ organization, the owner of NVMe™, NVMe-oF™ and NVMe-MI™ standards

Speaker

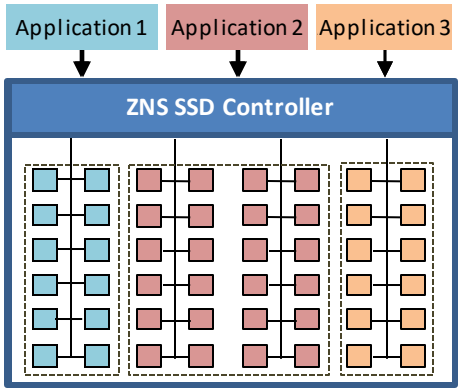
Dave Landsman
Director of Industry Standards

Beyond Random IO – New Use Cases for NVMe

- Over past couple years, industry debate on Open Channel vs. "Traditional" SSDs
- Debate is really about the emergence of two new SSD use case categories

IO Isolation

- Focus on Latency/QoS
- Host control over SSD physical topology



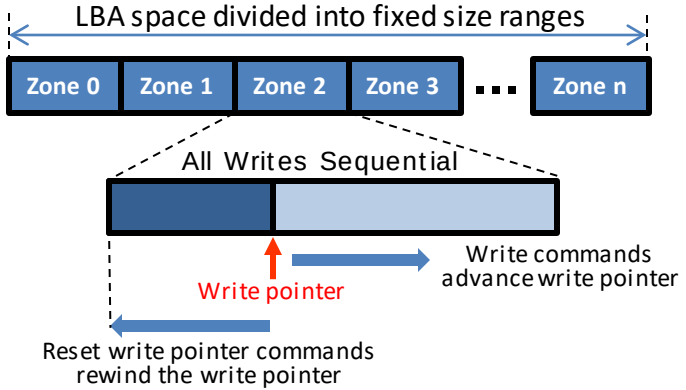
- **NVMe 1.4:** IO Determinism
  - NVM Sets & Endurance Groups
  - Predictable Latency Mode

- **New:** Endurance Group Mgmt
  - Enable standard provisioning of topology isolation use cases

Write Amp Reduction

- Focus on Capacity/Cost
- Host/SSD collaboration on GC and Wear Leveling

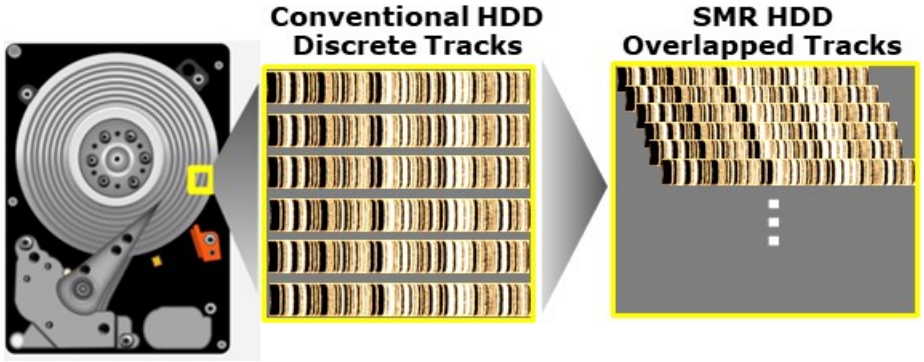
New: Zoned Namespaces



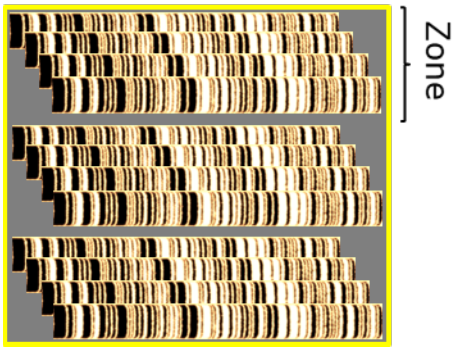
Zoned Block Storage already in HDDs

Take advantage of SMR capacity growth

- SMR (Shingled Magnetic Recording)
  - Enables areal density growth
  - Causes magnetic media to act like flash
    - Data must be erased to be re-written



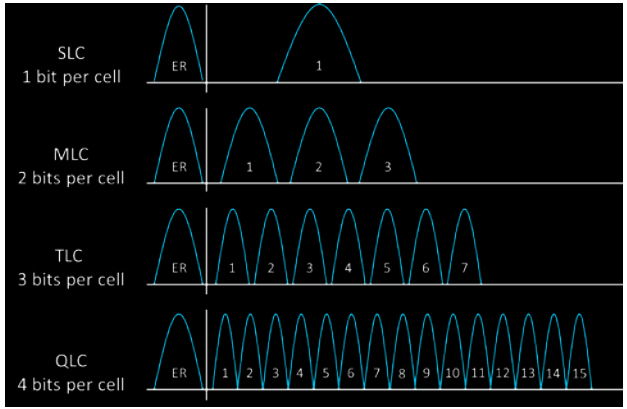
- Zoned Block access for HDDs
  - Drive formatted into fixed sized regions
  - Host/Device enforce sequential writes in LBA space to mitigate RMW effects of SMR
  - Zoned Block interface standardized in T13/T10
    - Zoned ATA Commands (ZAC): SATA
    - Zoned Block Commands (ZBC): SAS



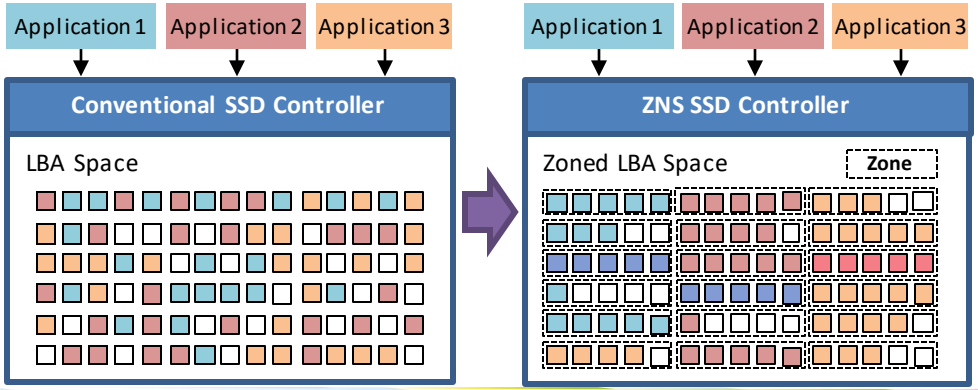
Why Zoned Block Storage for SSDs

Take advantage of TLC/QLC capacity growth

- TLC & QLC increases capacity but at cost of
  - Less endurance
  - Lower performance
  - More DRAM to map higher capacity

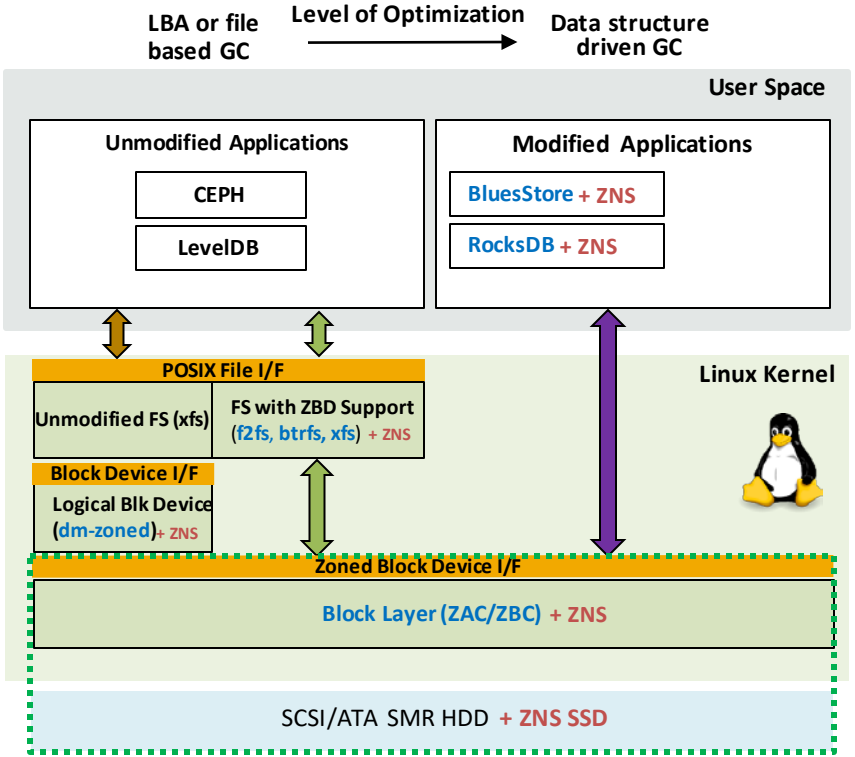


- Zoned Block access for SSDs
  - SSDs are intrinsic Zoned devices due to flash characteristics
  - Host/SSD cooperate (distributed FTL) using sequential access
  - No complex topology provisioning; Zones are logical
  - Reduces write amplification and internal data movement
- Result
  - Reduced wear
  - Improved latency outliers and throughput
  - Reduced DRAM in SSD (smaller L2P)
  - Reduced drive Over Provisioning



Why Zoned Block Storage for SSDs?

Synergy w/ ZAC/ZBC software ecosystem

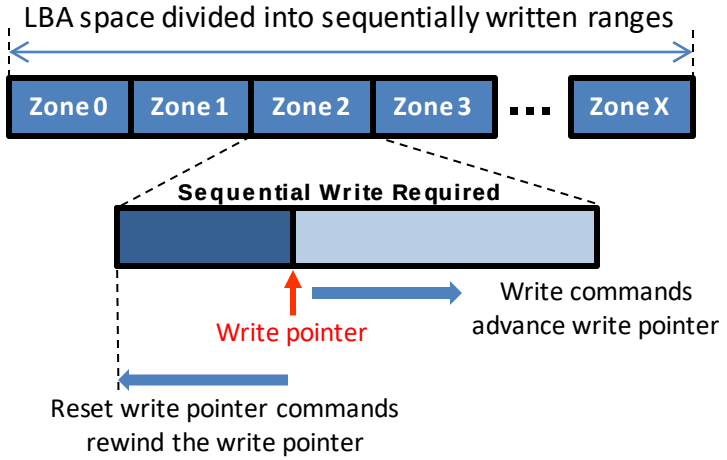


- SW ecosystem optimized at multiple levels
- Foundation is the Zoned Block Device
- Different optimization paths
  - App -> POSIX File -> Unmodified FS -> dm-zoned -> ZBD
  - App -> POSIX File -> Modified FS -> ZBD
    - FS knows which LBAs in which file
  - Modified App -> ZBD
    - No file system; app knows most about data; e.g., may use zones as containers for objects
- We are adding changes for ZNS

Zoned Namespaces TP

Ongoing in the NVMe™ working group

- Inherits NVM Command Set
- Namespace divided into fixed sized Zones
  - Sequential Write Required is only zone type supported for now
- Aligned to host-managed ZAC/ZBC model, with some SSD optimizations
  - Zone Capacity
  - Zone Append



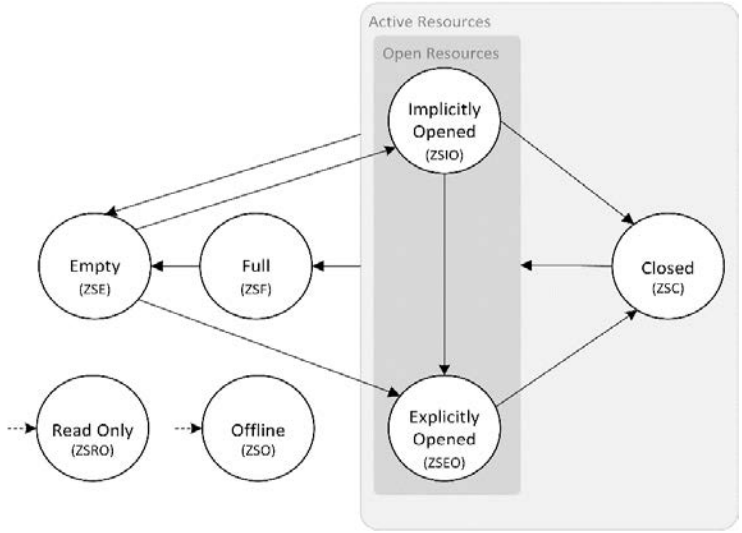
Zoned Namespaces TP

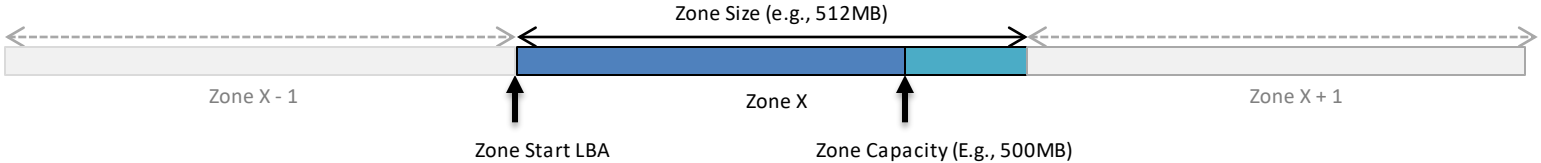
Zone Capacity

- ZNS model similar to ZAC/ZBC
  - States: Empty, Full, Implicit Open, Explicit Open, Closed, Read Only, Offline
  - State Changes: Write, Zone Management Command (Open, Close, Finish, Reset), Device Resets

- Zone Size vs. Zone Capacity(NEW)
  - Zone Size is fixed
  - Zone Capacity is variable

ZNS State Machine

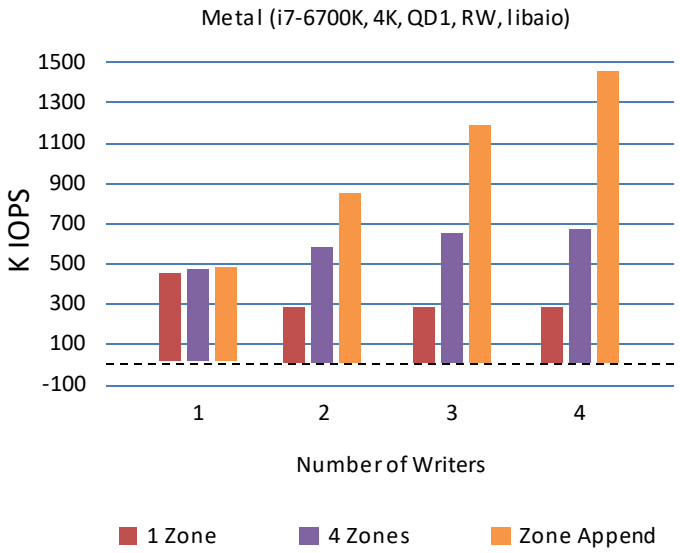




Zoned Namespaces TP

Zone Append

- ZAC/ZBC requires strict write ordering
  - Limits write performance, increases host overhead
- Low scalability with multiple writers to a zone
  - One writer per zone -> Good performance
  - Multiple writers per zone -> Lock contention
- Performance improves somewhat by writing to multiple Zones
- With Zone Append, we scale
  - Append data to a zone with implicit write pointer
  - Drive returns LBA where data was written in zone

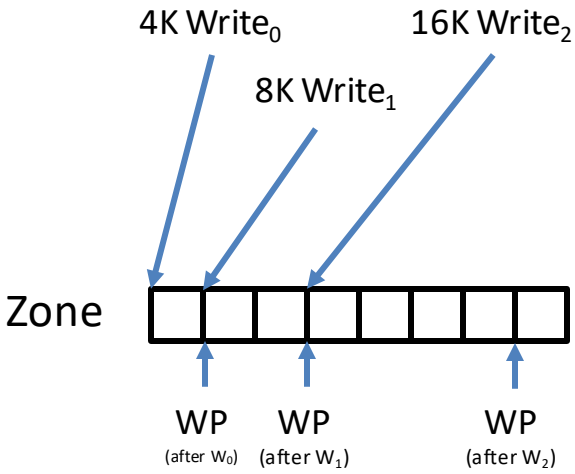


Zoned Namespaces TP

How does Zone Append work?

Zone Write Example

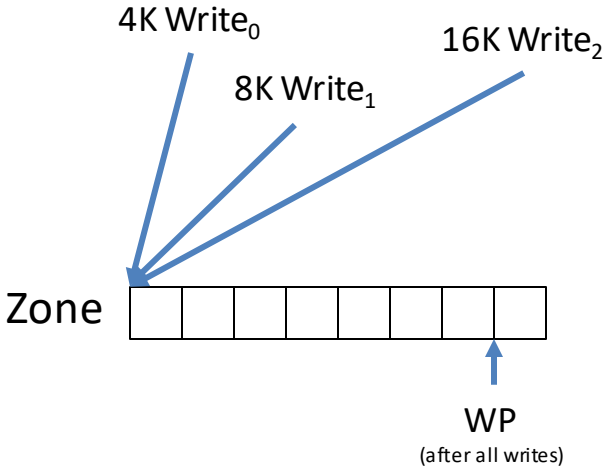
Queue Depth = 1



- Host serializes I/O, forces low queue depth
- Insignificant lock contention when using HDDs
- Significant lock contention when using SSDs

Zone Append Example

Queue Depth = 3



- No host serialization; higher queue depth
- Scalable for HDDs and SSDs

Summary – Zoned Namespaces

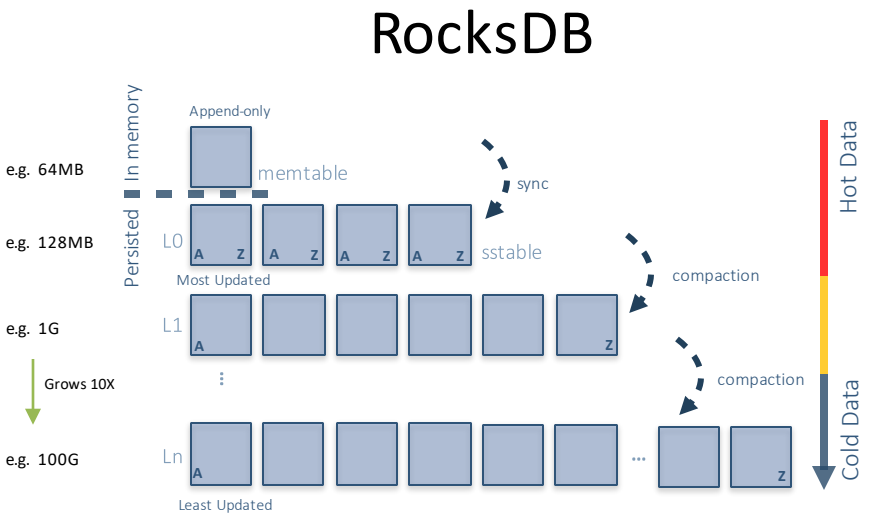
- Standardizes interface for key evolving SSD use case
  - Sequential-write centric workloads
  - Host/SSD cooperate on GC and WL
- Enables lower cost solutions
  - Reduced wear
  - Reduces SSD DRAM
  - Reduced overprovisioning
- SW model synergy w/ SMR HDD ecosystem
- Specification nearing completion in NVMe™ WG

Questions?

BACKUP

RocksDB and Zoned Block Devices

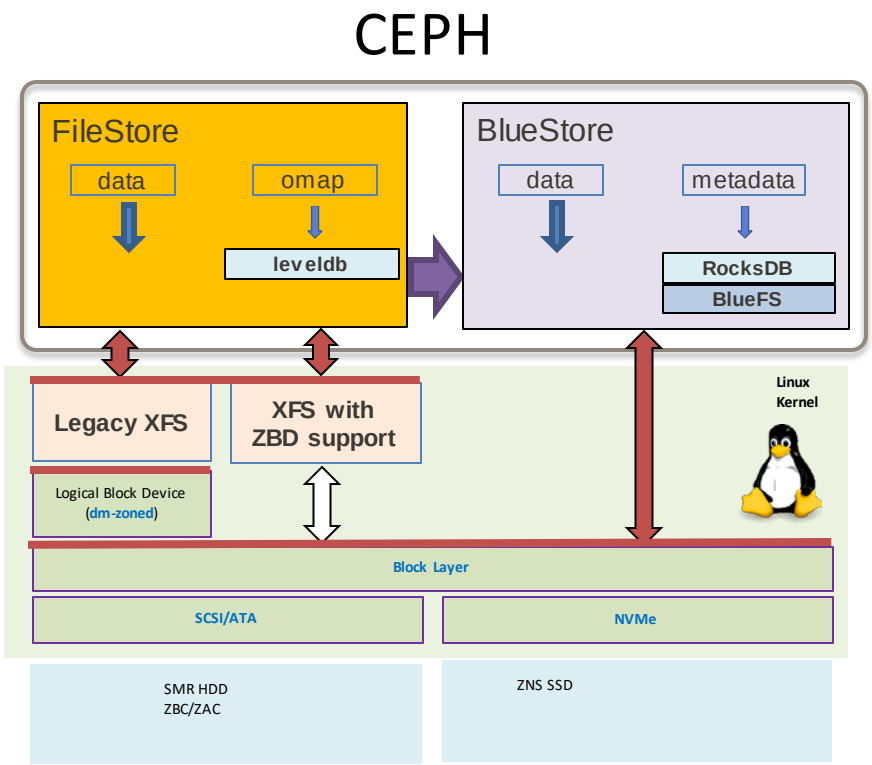
- Embeddable key-value persistent store where keys and values are arbitrary byte streams.
- Optimized for fast storage, many CPU cores and low latency
- Based on Log-Structured Merge (LSM) Tree data structure
- The LSM structure aligns with Zones, and enables significant optimizations
- Integrates with MySQL databases (MyRocks)
- Patches are in progress



Ceph and Zoned Block Devices

Zettabyte Infrastructure

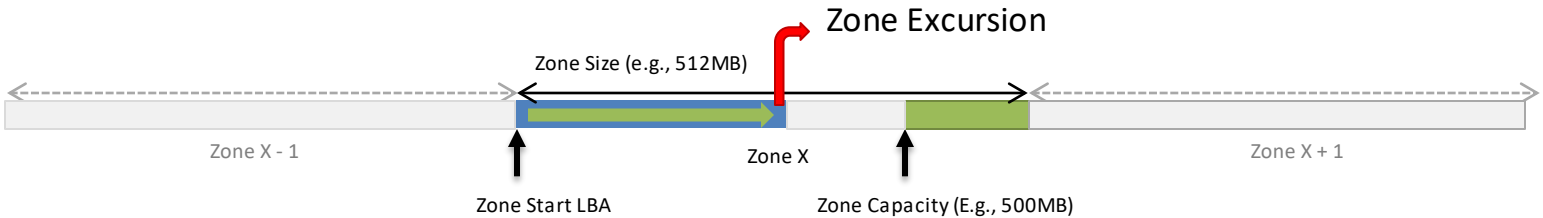
- Distributed File-System providing object, block and file-level storage.
- Enable Ceph to utilize Zoned Block Devices
- Ceph Bluestore removes the local file system from the equation
  - BlueFS backend writes data directly to the block device and can handle the sequential constraints
  - RocksDB uses LSM-trees that naturally generate no/few random updates and can easily be stored on ZBDs as well
- Zones or group of zones can map to natural failure domains that may be smaller than the whole device
  - Mapping OSDs to such failure domains would naturally ensure that recovery from failure would involve less network utilization and fewer I/Os



Zoned Namespaces TP

Attributes: Zone Excursions & Variable Capacity

- For NVMe™ devices that implement the Zoned Command Set, there is optional support for:
  - Variable Capacity
    - The completion of Reset Zone command may result in a notification that zone capacity has changed
  - Zone Excursions
    - The device can transition a zone to Full before writes reaches the Zone Capacity. Host will receive an AEN and write failure if writing after the transition
- If device implements, the host shall implement as well
  - Incoherent state model if not – Software should be specifically be written to know that zone capacity can change, or writes may suddenly fail



Architected for Performance