



Flash Memory Summit

Design challenges and tradeoffs with QLC-based ZNS SSDs

Guanying Wu @ Silicon Motion
gwu@siliconmotion.com

Outline

Why use ZNS with QLC?

Why is QLC so hard to use?

Design options and tradeoffs

Conclusion

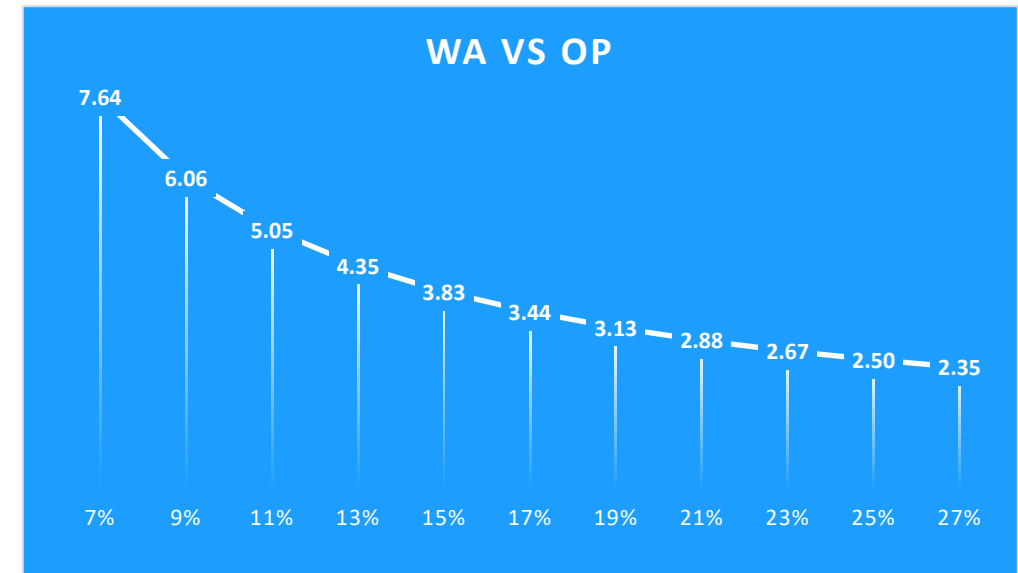


Why Use ZNS with QLC?

Why Use ZNS with QLC?

Why ZNS?

- GC/WA is expensive
- SSD doesn't know how to do it right
- → Let hosts worry about it
- → Only sequential writes allowed



Why Use ZNS with QLC?

Why QLC?

- Lower cost: 70~80% of TLC
- Good read performance
 - ~100us read latency
 - Less than 100 dies to saturate PCIe Gen 5 x4 under random read workload
- Fair write performance
- Poor endurance: 1.5K~3K PEC
 - TLC: WA of 5 → Good DWPD
 - QLC: WA of 1 → Good DWPD, as well

Source: ISSCC'2022

	Layer	Plane	Area density	Prog BW	tR	Interface
Kioxia/WD, 1Tb	162	4	15.0Gb/mm2	60MB/s	65us	2400MTs
Micron	176	4	14.7Gb/mm2	40MB/s	90us	1600MTs
SK-Hynix	176	4	14.8Gb/mm2	40MB/s	90us	1600MTs
Samsung	(NA) close to 176	4	11.55Gb/mm2	164MB/s	45us	2400MTs



Why Use ZNS with QLC?

ZNS + QLC can serve

hosts who

- Do sequential writes (low WA) only
 - Do no GC at all, or
 - Do GC on their own
- Desire good read performance
- Desire lower cost



Good Value!

- **OLAP**
- **Big data + AI**
- **Media streaming**
- **NoSQL**
- **And more ...**



Why is QLC So Hard to Use?

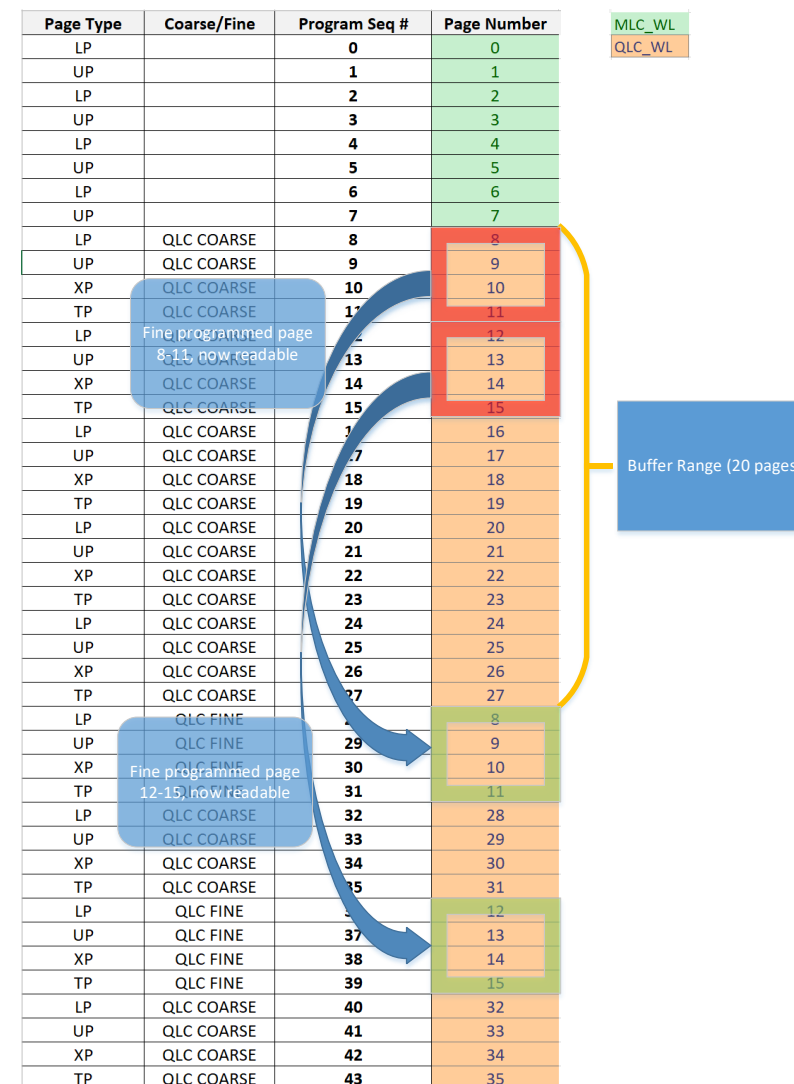
Why is QLC So Hard to Use?



Flash Memory Summit

Coarse-fine program sequence

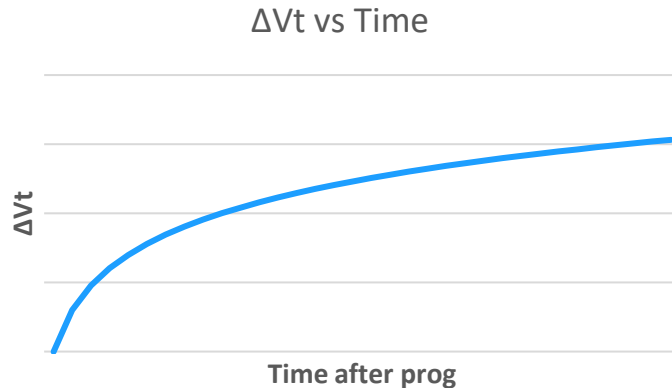
- Charge-Trapping QLC NAND
- 1st pass: coarsely prog WL (precharge the cells); unreadable until 2nd/fine prog pass (full charge)
- Coarse and fine prog on the same WL are separated by many pages
- ∴ Coarse-prog'd pages must be persisted in some way
- ∴ Lots of data even for a single-stream drive
 - ~20 pages per open MP block = 1280KB
 - 128 dies & 1 stream
 - **160MB**



Why is QLC So Hard to Use?

Data retention

- Not only 16 states
- But also, charge leaks for days



Solutions

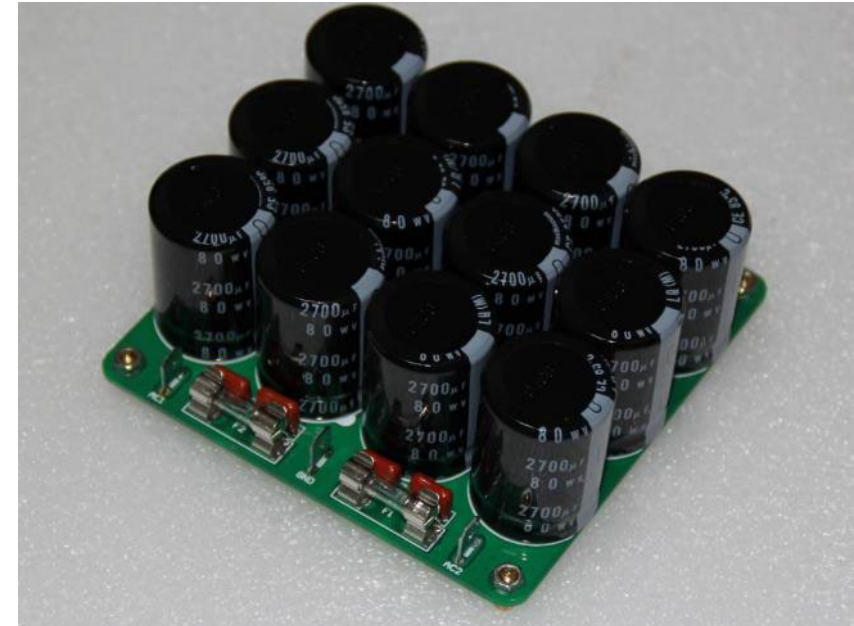
1. NAND gives the best guesses
 - Only if your NAND can support it
 - Overhead on the read latency
2. Predict ΔV_t using time after prog
 - Retain prog time
 - Less reliable
 - Lose track upon power cycle
3. Track ΔV_t by periodical scanning
 - Retain V_t with an interval tree



Why is QLC So Hard to Use?

ZNS / multi-stream makes worse ...

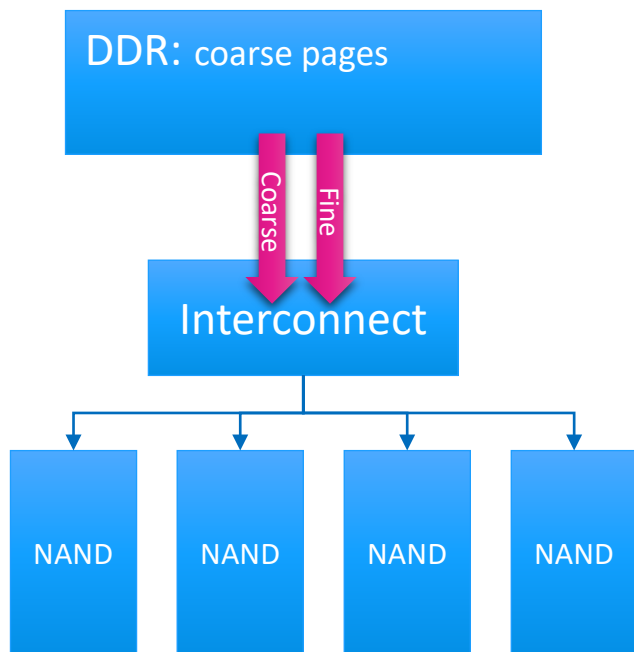
- Customers want lots of open zones
- Say, 16 open blocks per die → **2560MB**
- + larger data structure to track Vt





Design options and tradeoffs

Option 1: DDR + Capacitors



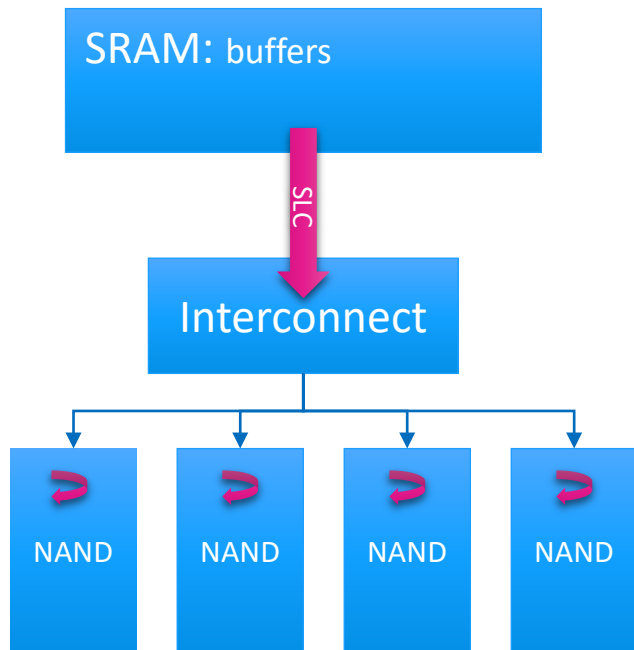
Pros

- Easy implementation
- Low overhead
 - Use all available NAND space
 - No performance hit

Cons

- Large capacitors
 - $3000\mu\text{F} = \text{X00MB}$
 - \therefore only a couple of open QLC blocks/die on 128-die drive
 - \therefore won't allow many open zones
- QLC prog must follow the host
 - Difficult to track Vt

Option 2: SLC copyback



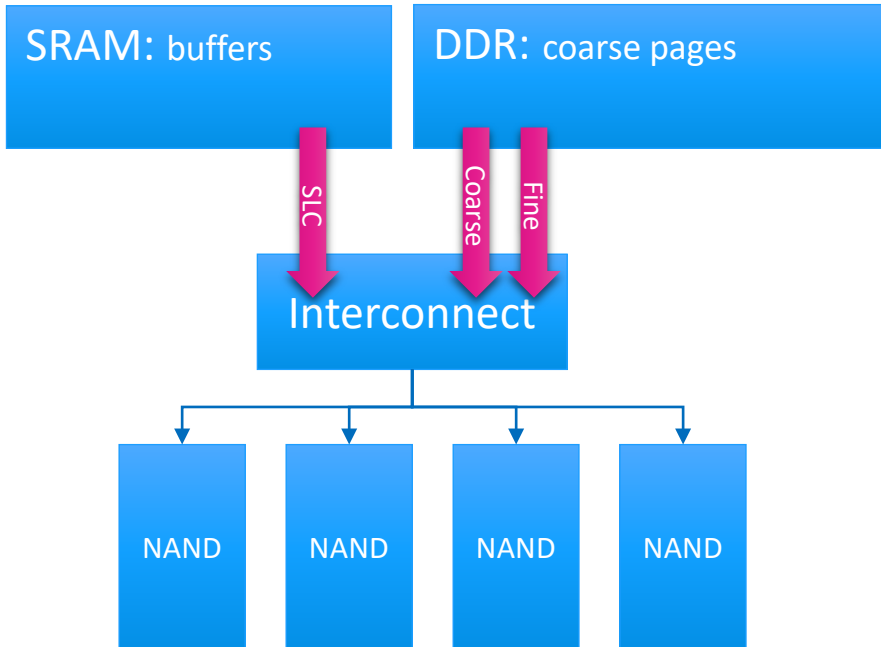
Pros

- Allows DRAM-less
- Small cap works
- Low NAND chan usage
- Decouples QLC prog and the host
 - Making Vt tracking easier

Cons

- SLC RBER → HRER
 - Retention > days
 - EOF
 - ∴ Careful tracking
- 5~6% blocks for SLC
- QoS
 - SLC prog/copyback/erase (**10%** performance hit)
 - Long NAND sequence
 - Challenging implementation
 - Lots of support from NAND vendors

Option 3: SLC backing up DDR



Pros

- Small cap works
- Allows many open zones
- Easy implementation
- Easy to change design to SLC copyback
- Moderate space/performance hits

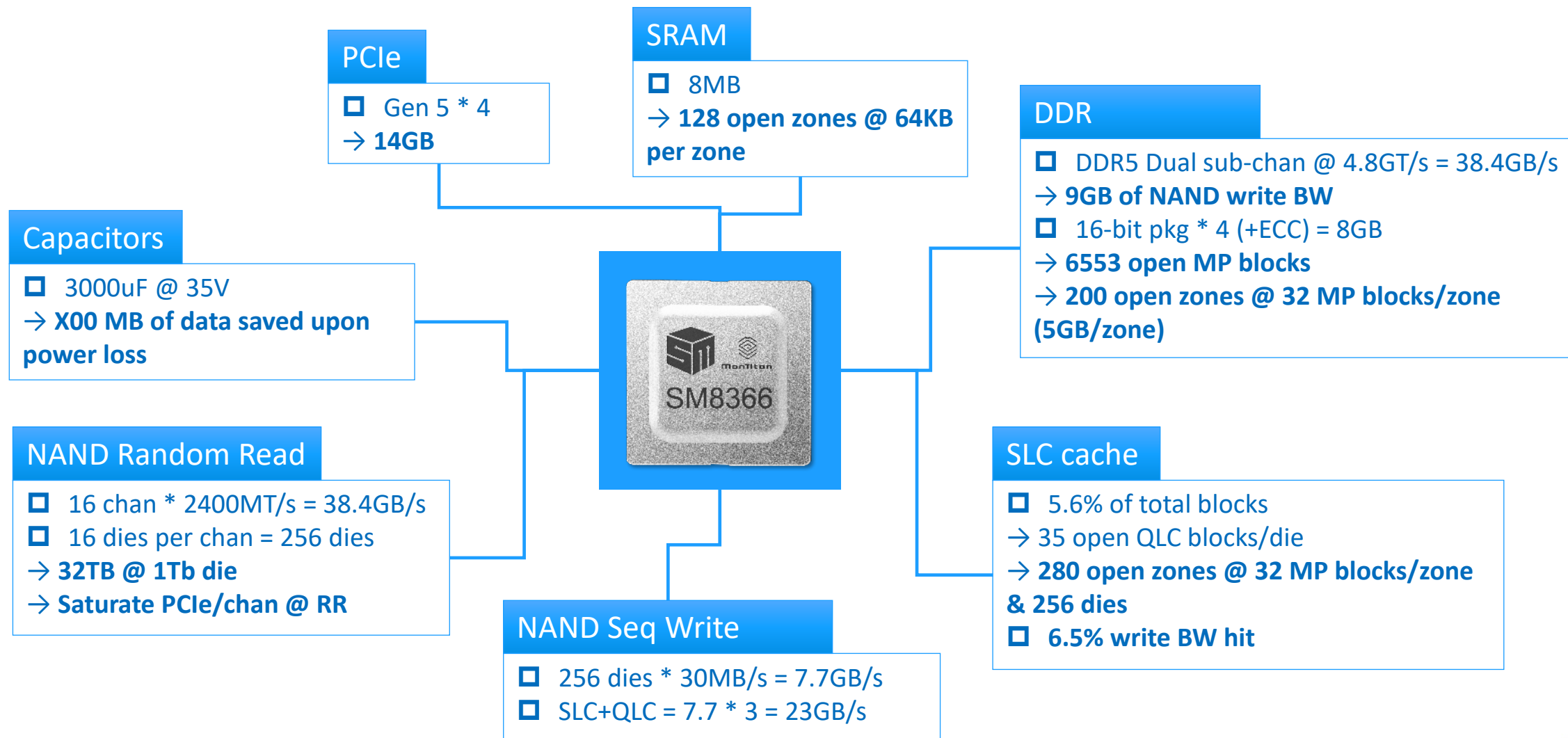
Cons

- DDR bandwidth?
 - DDR5 will do
- NAND chan bandwidth?
 - 1600 MB/s → ~200 dies

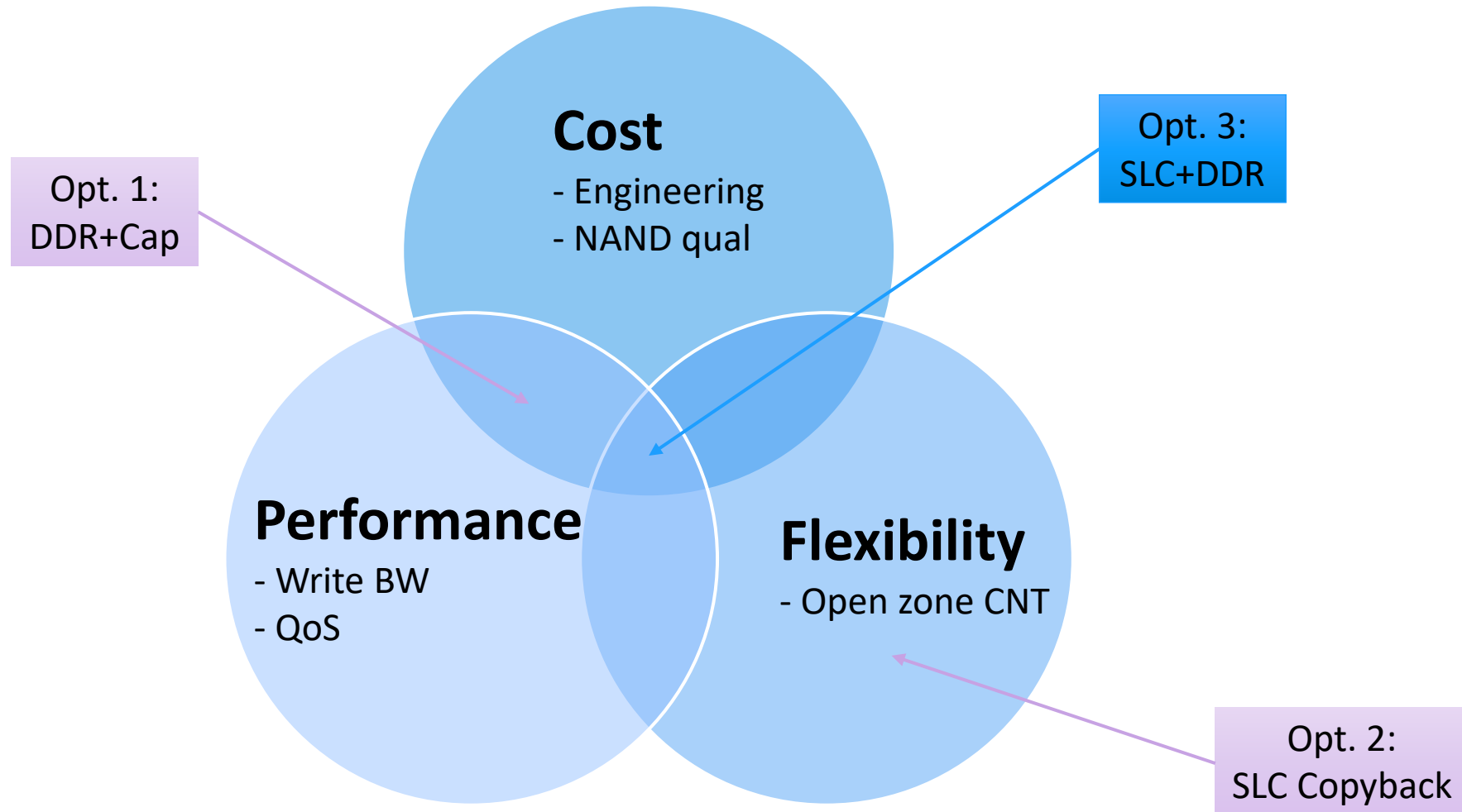
Option 3: A Quantitative Analysis



Flash Memory Summit



Conclusion: we choose **SLC** backing up **DDR**





Please stop by Booth 311 to see the latest offerings and technology demonstrations from



SiliconMotion