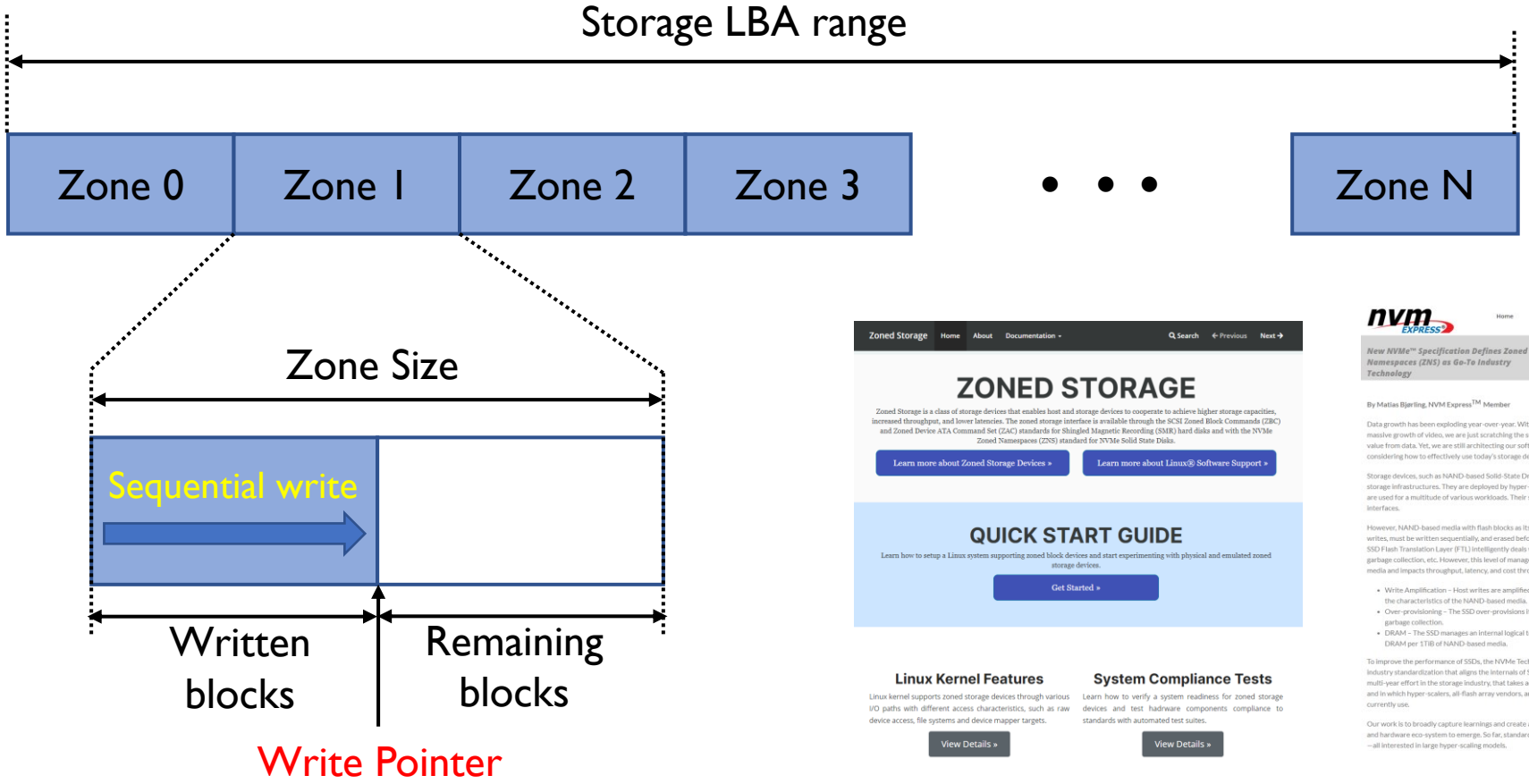
ZNS+: Advanced Zoned Namespace Interface for Supporting In-Storage Zone Compaction

Kyuhwa Han[1,2], Hyunho Gwak[1], Dongkun Shin[1], and Joo-Young Hwang[2]

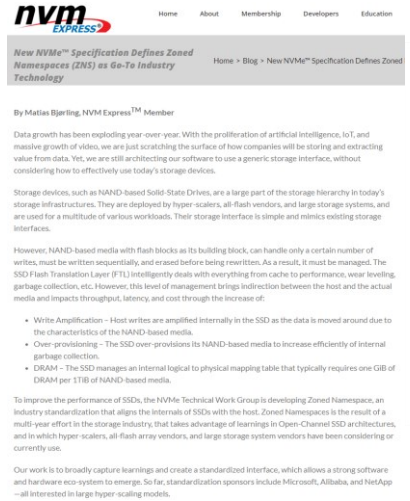
[1]Sungkyunkwan University, [2]Samsung Electronics

Zoned Name Space (ZNS) Storage

- The logical address space is divided into fixed-sized zones.
- Each zone must be written sequentially and reset explicitly for reuse



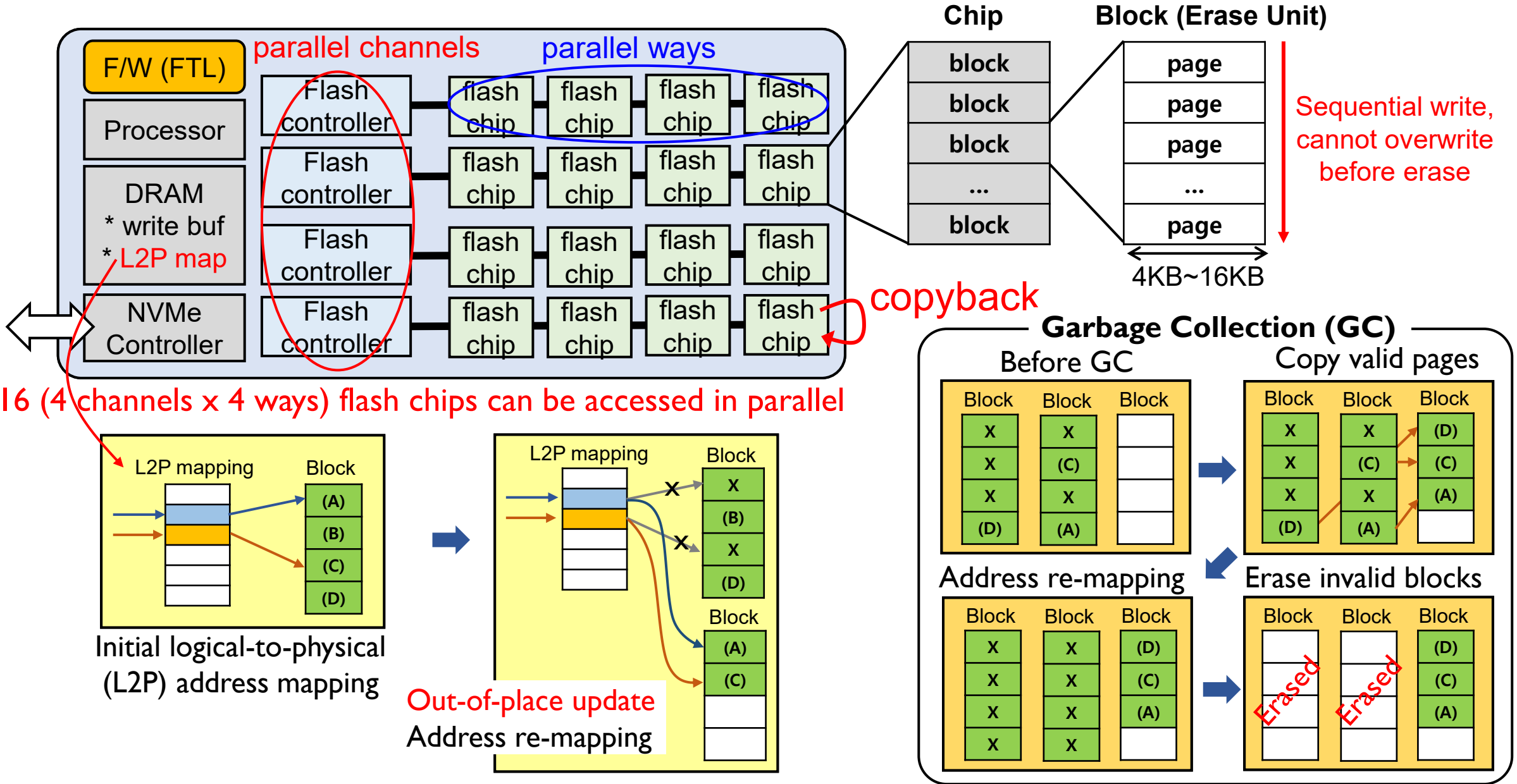
https://zonedstorage.io/



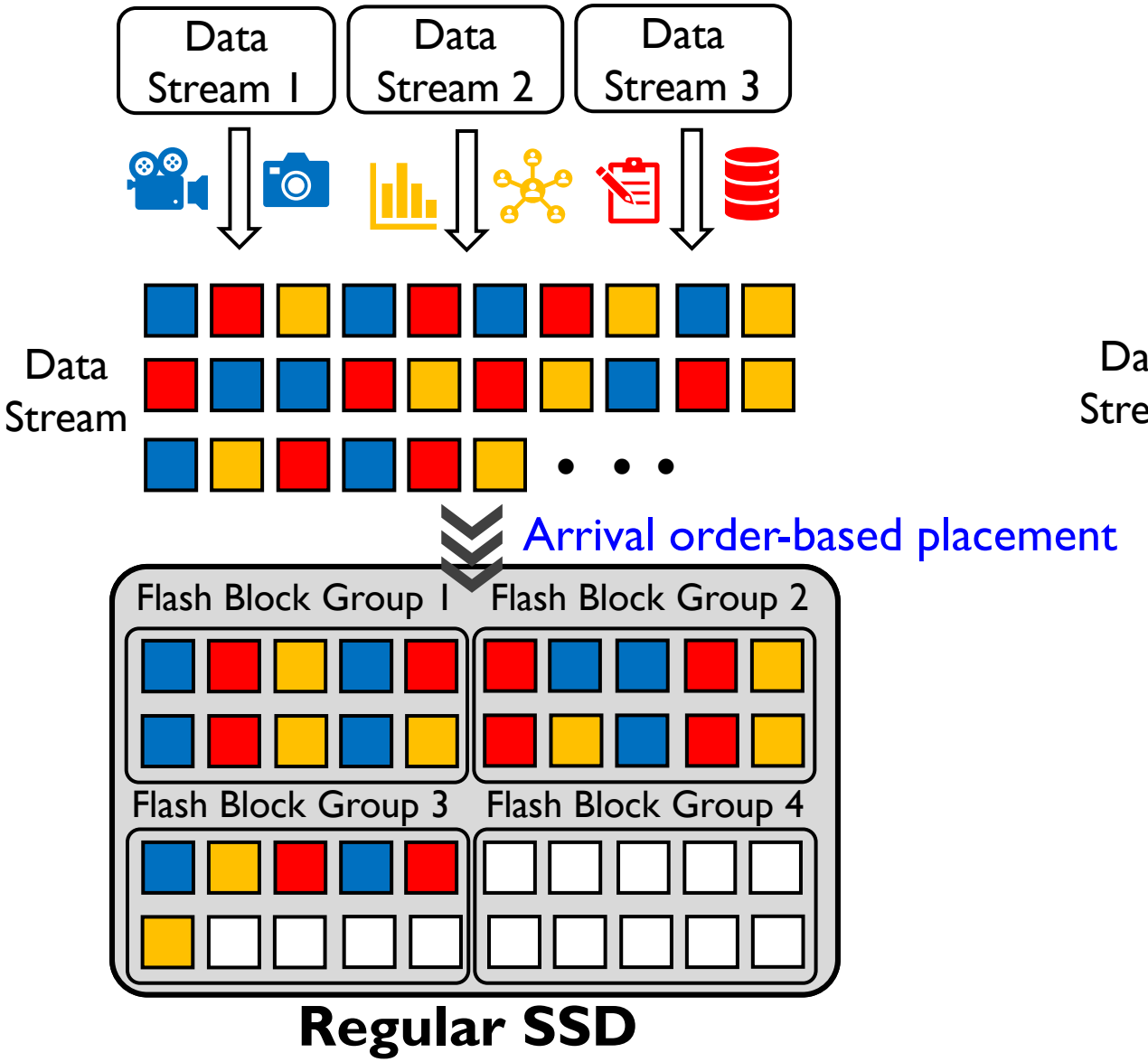
https://nvmexpress.org/

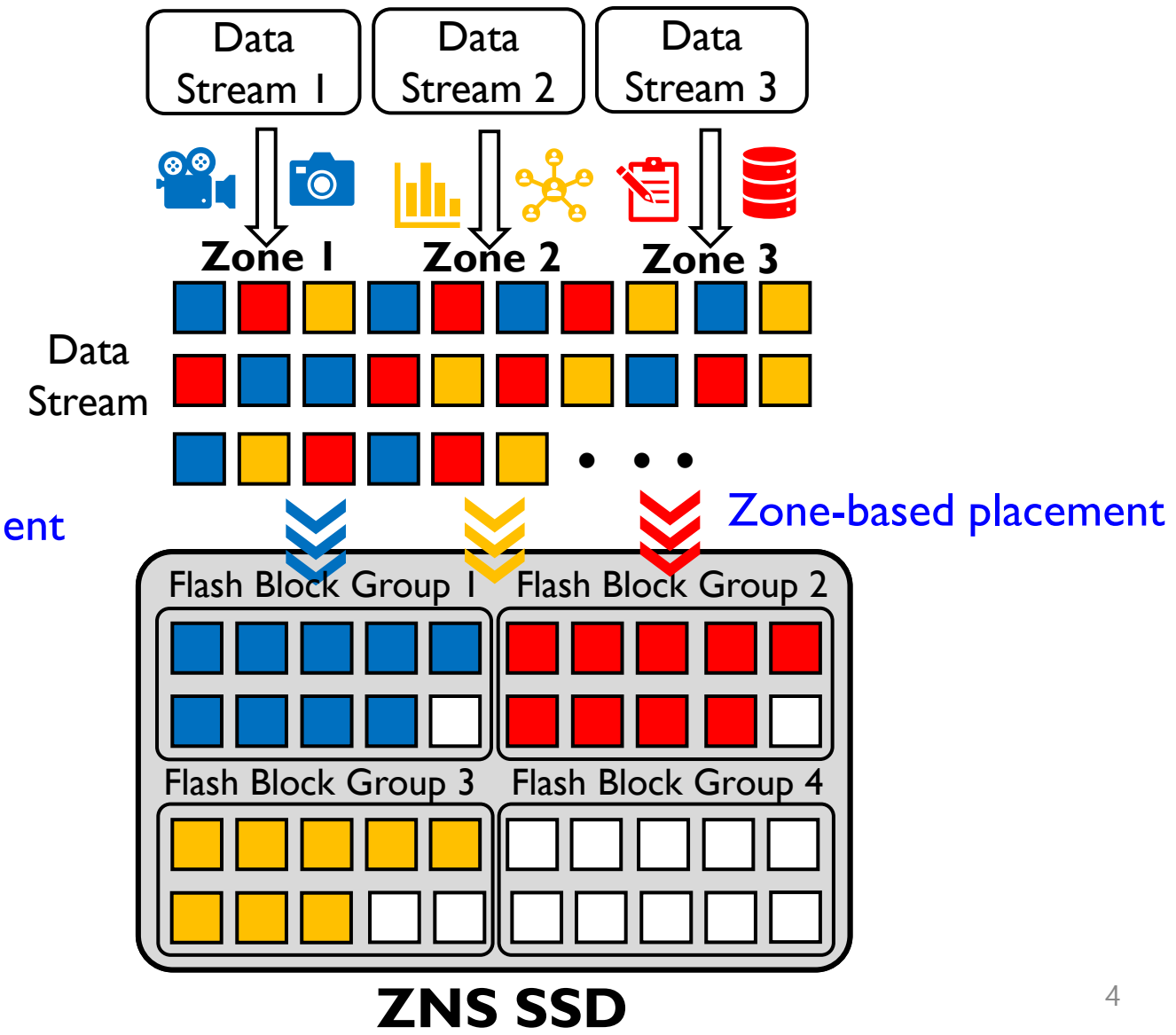
QLC(4bit) ZNS SSD (June 2021)

SSD Architecture 101

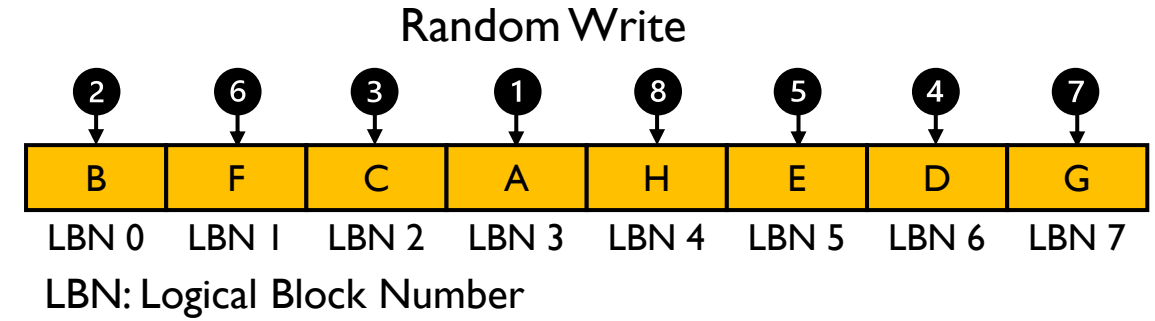


Why ZNS? Isolation, hot/cold separation





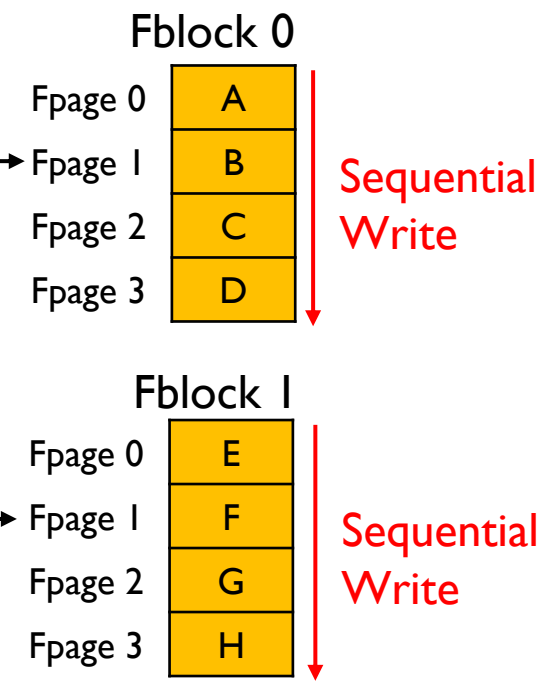
Why ZNS? Small L2P Translation Table



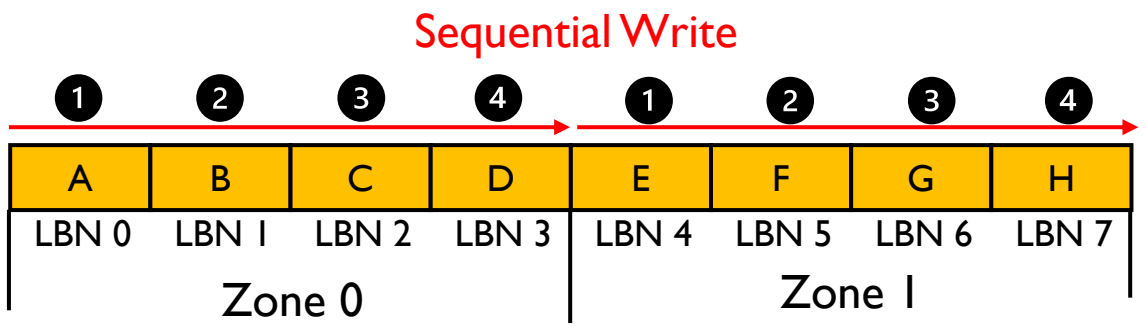
LBN-level L2P translation table

LBN	Fblock/Fpage
0	0/1
1	1/1
2	0/2
3	0/0
4	1/3
5	1/0
6	0/3
7	1/2

Logical Block Size = 4KB



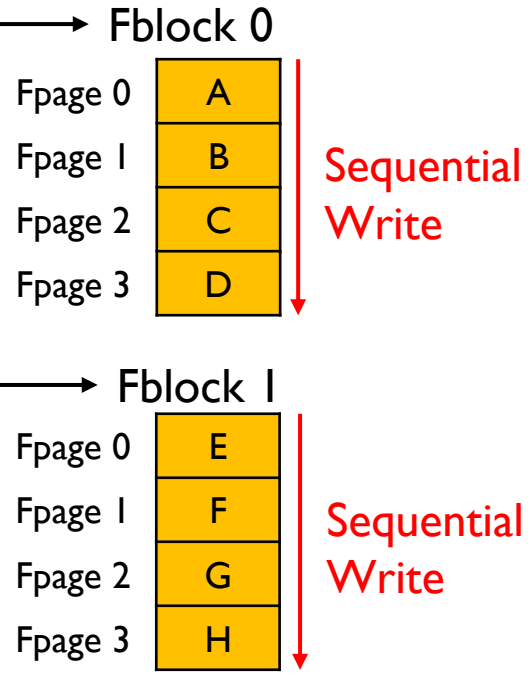
Regular SSD



Zone-level L2P translation table

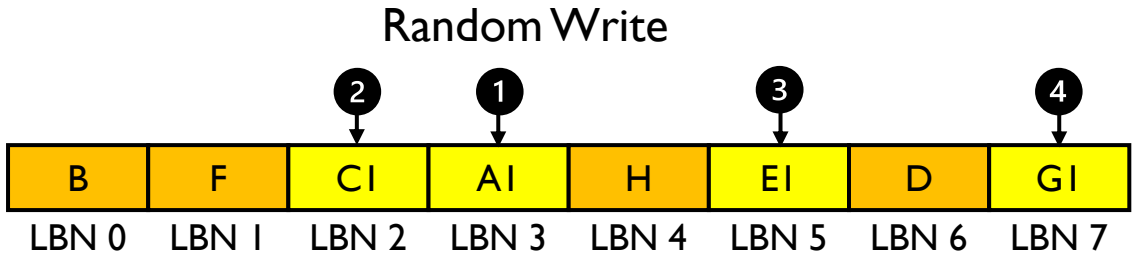
Zone	Fblock
0	0
1	1

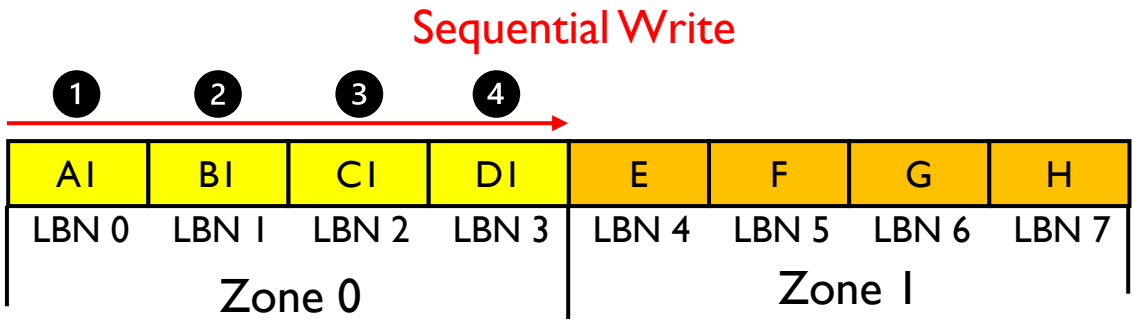
Logical Zone Size = xxMB



ZNS SSD

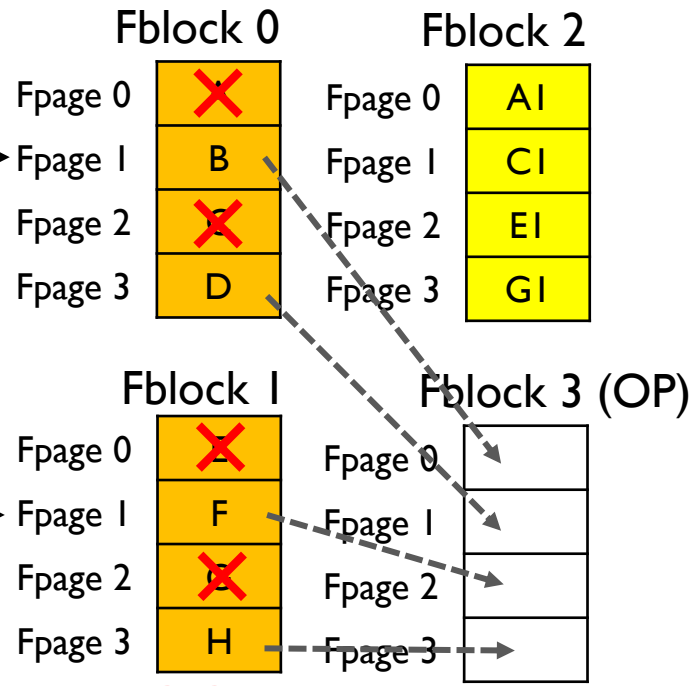
Why ZNS? GC-less, Predictable





LBN-level L2P translation table

LBN	Fblock/Fpage
0	3/0
1	3/2
2	2/1
3	2/0
4	3/3
5	2/2
6	3/1
7	2/3



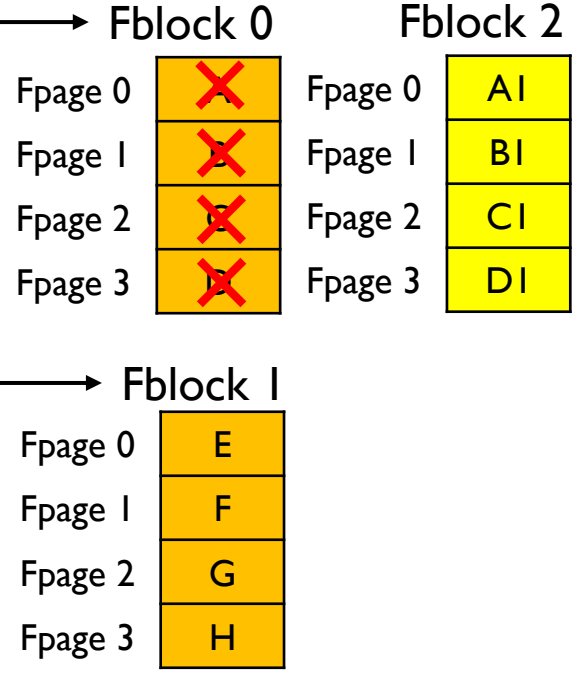
Regular SSD

GC: valid page copy

➔ write amplified, unexpected delay

Zone-level L2P translation table

Zone	Fblock
0	2
1	1



GC-less, No OP

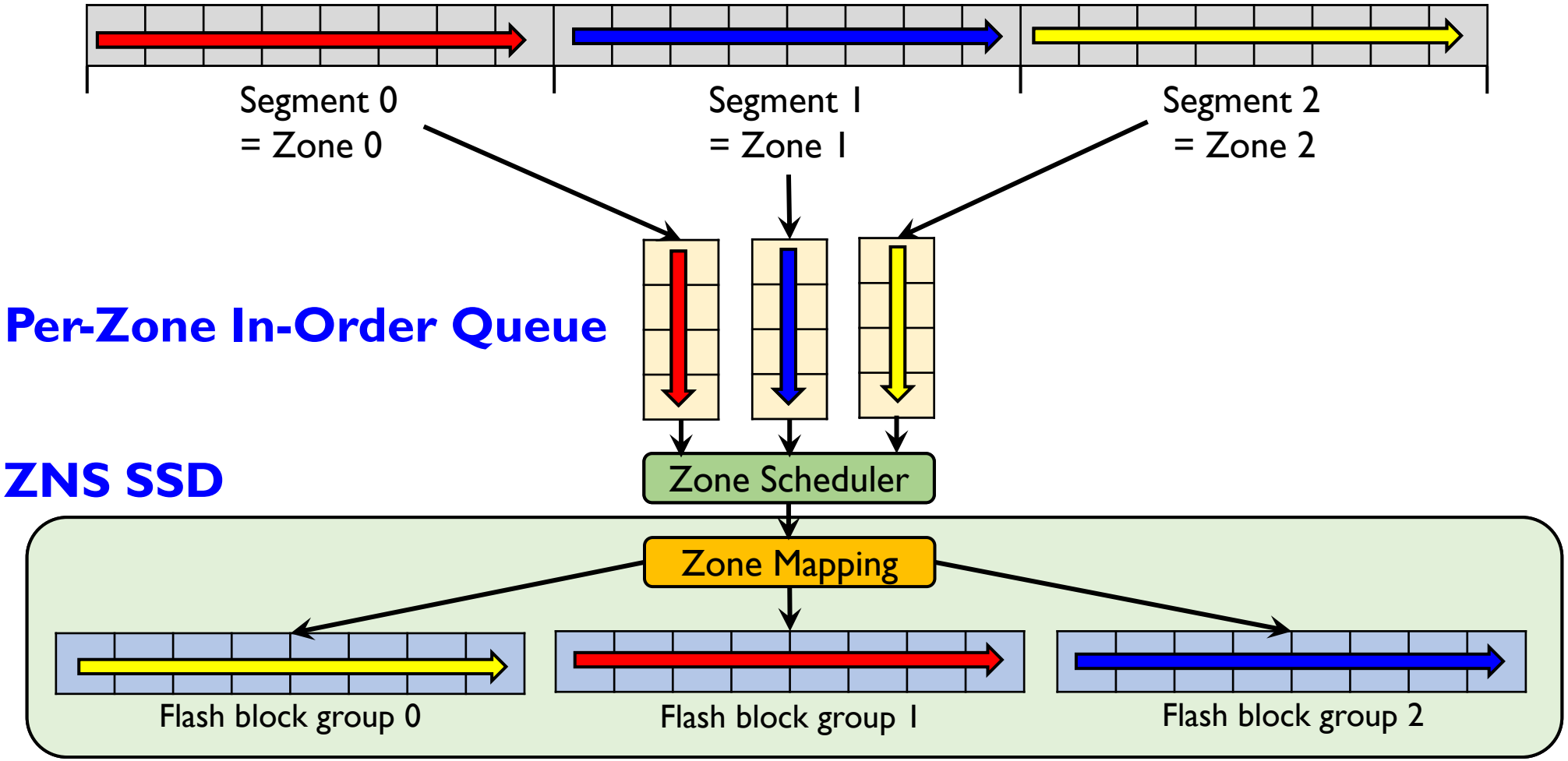
WAF ≈ 1

(write amp. factor)

ZNS SSD

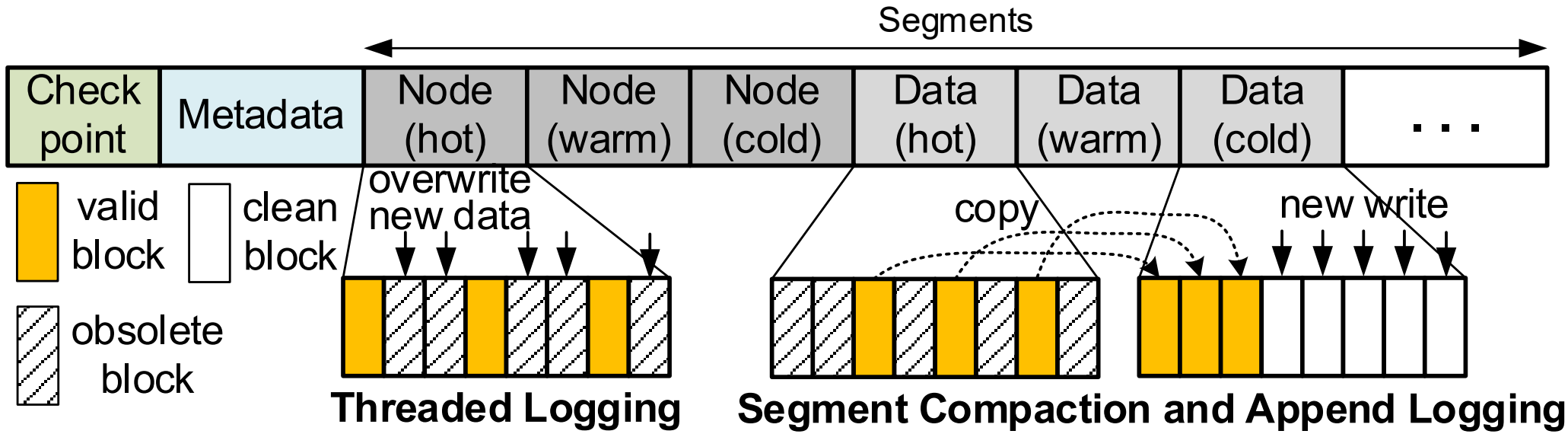
New IO Stack for ZNS

Log-Structured File System (LFS) – Append Logging (Sequential Write)



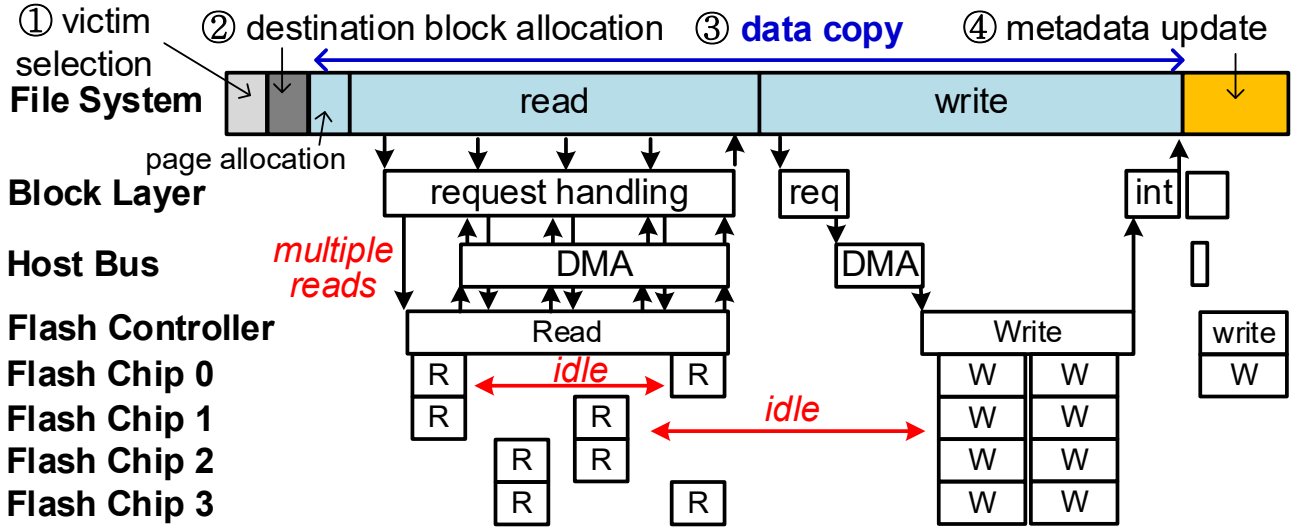
F2FS (Flash-Friendly File System)

- One of actively maintained Log-structured File Systems (LFS)
- Six types of segments: hot, warm, and cold segments for each node/data
- Multi-head logging
- Supports both append logging (AL) and threaded logging (TL)
- A patch version for ZNS is available (threaded logging is disabled)



Segment Compaction

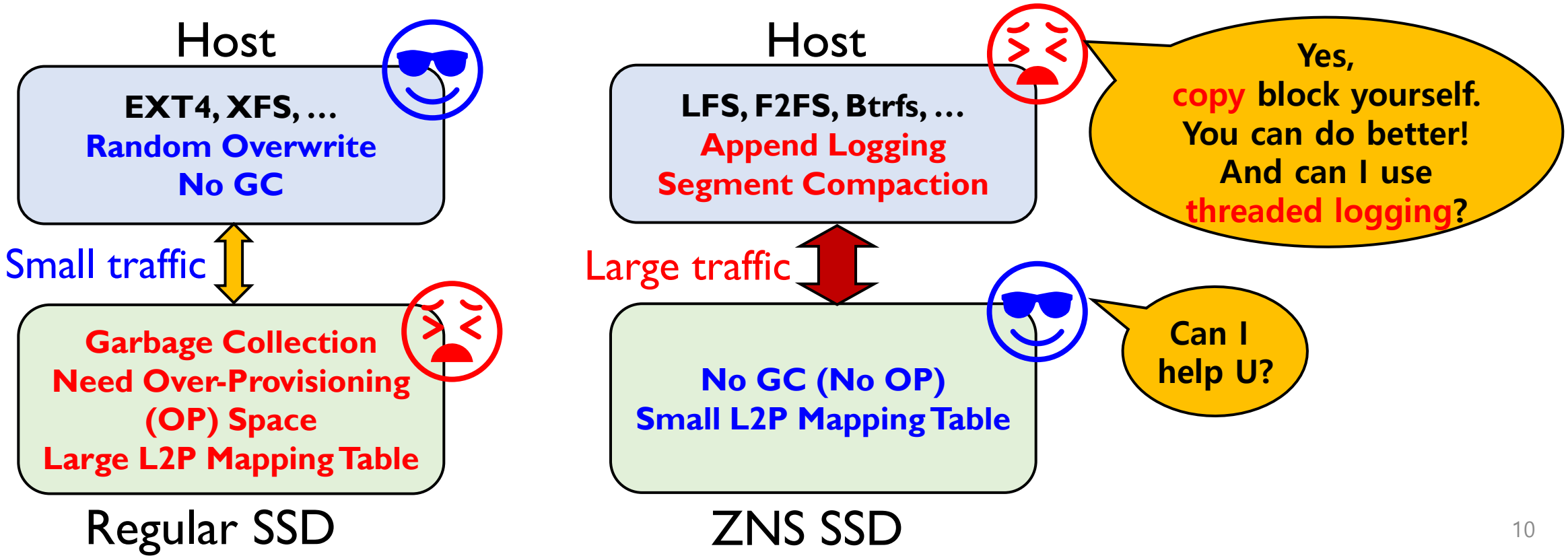
1. Victim segment selection - a segment with the lowest compaction cost
2. Destination block allocation - contiguous free space
3. Valid data copy - moves all valid data in the victim segment to the destination segments via host-initiated read and write requests
   - Many idle intervals of flash chips
4. Metadata & checkpoint update



Normal segment compaction via host-level copy

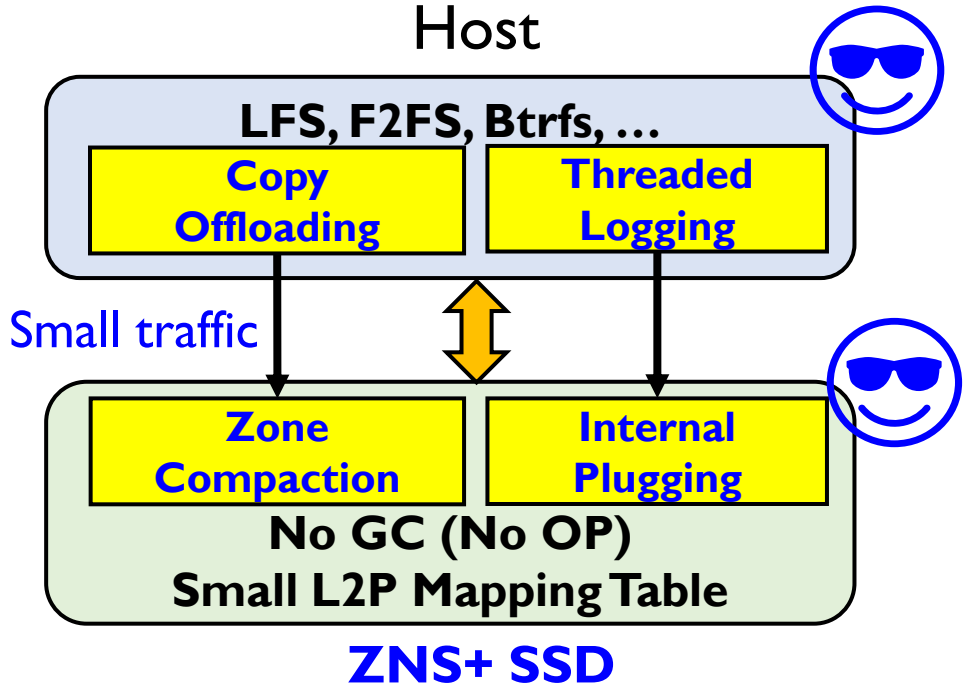
Robbing Peter to Pay Paul?

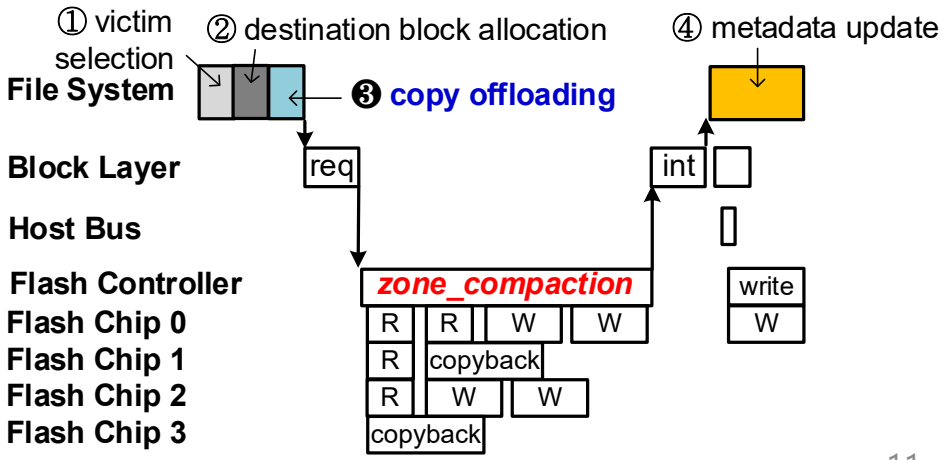
- Host-side GC in exchange for using GC-less ZNS SSD
- Host-side GC overhead > Device-side GC overhead
  - IO Request Handling, H2D Data Transfer, Page Allocation, Metadata Update



ZNS+: LFS-Aware ZNS

- **Internal Zone Compaction (IZC)**
  - `zone_compaction` command
  - Can accelerate zone compaction
  - Copy blocks within SSD
  - Reduce host-to-device traffic
  - SSD can schedule flash operations efficiently
- **Sparse Sequential Write**
  - `TL_open` command
  - Can avoid zone compaction w/ threaded logging
  - The host can overwrite a zone sparsely
    - ✓ The block addresses of the consecutive writes must be in the increasing order
  - Transformed to dense request w/ internal plugging by SSD
  - Can hide latency by utilizing idle flash chips

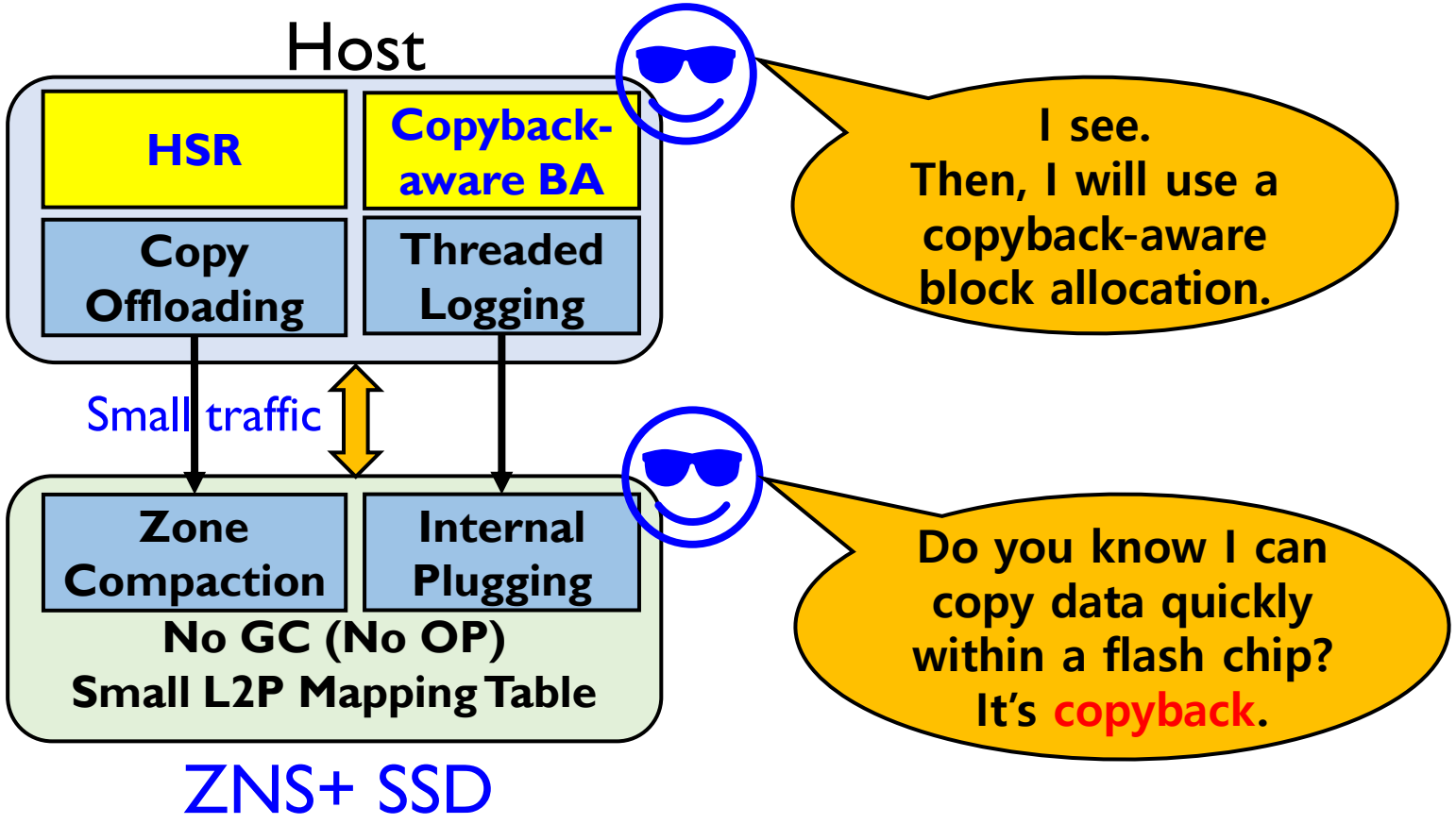


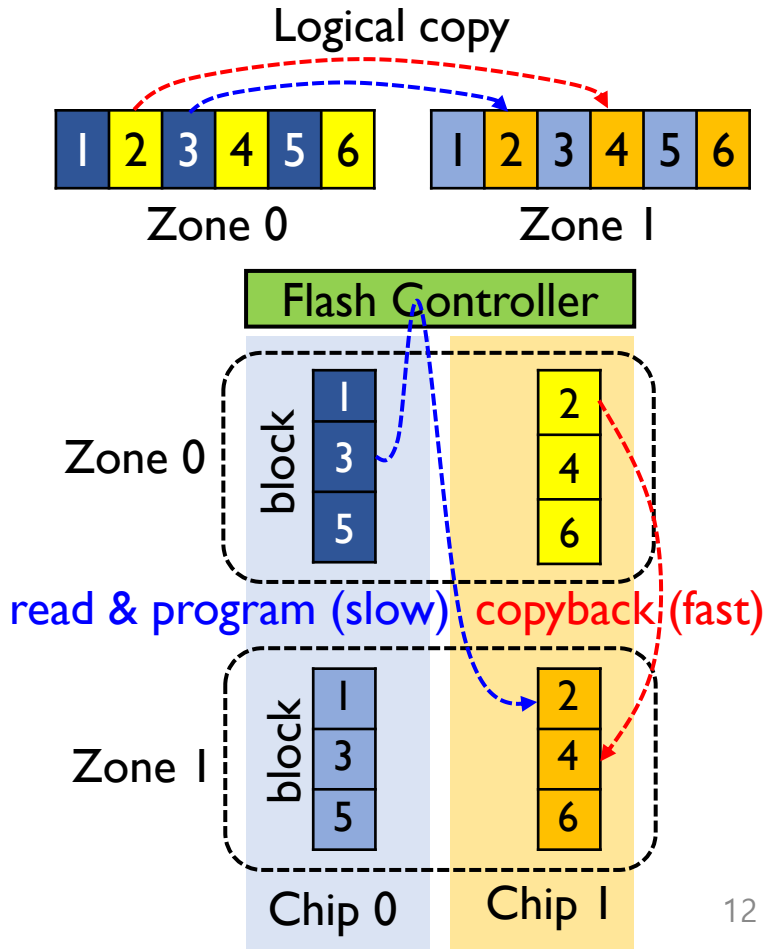


Efficient internal scheduling

ZNS+-aware LFS

- Hybrid segment recycling (HSR): segment cleaning vs. threaded logging
- Copyback-aware block allocation: maximize on-chip copyback operations





Experimental Setup

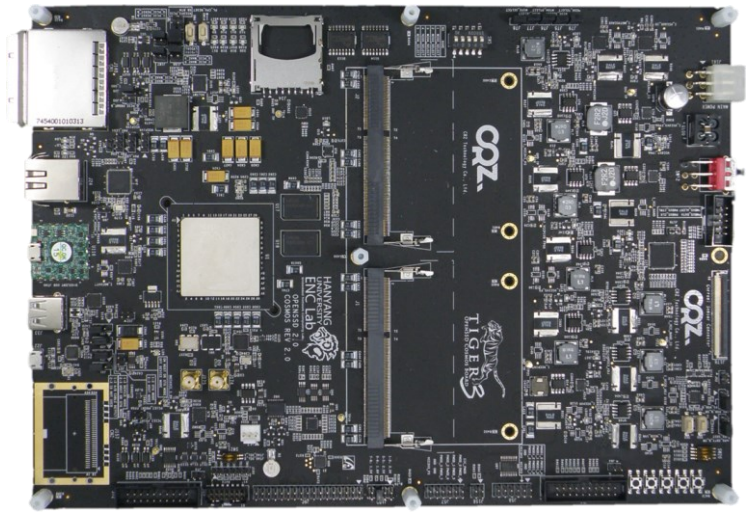
- ZNS+ emulator based on FEMU
- Real ZNS+ implemented at Cosmos+ OpenSSD
- Modified F2FS 4.10
- Comparison
  - ZNS vs. IZC (internal zone compaction, no TL) vs. ZNS+ (IZC and TL)

The CASE of FEMU:
Cheap, Accurate, Scalable and Extensible Flash Emulator

Huaicheng Li, Mingzhe Hao, Michael Hao Tong,
Swaminatahan Sundararaman†, Matias Bjørling‡, Haryadi S. Gunawi
University of Chicago †Parallel Machines ‡CNEX Labs

A QEMU-based and DRAM-backed NVMe SSD Emulator

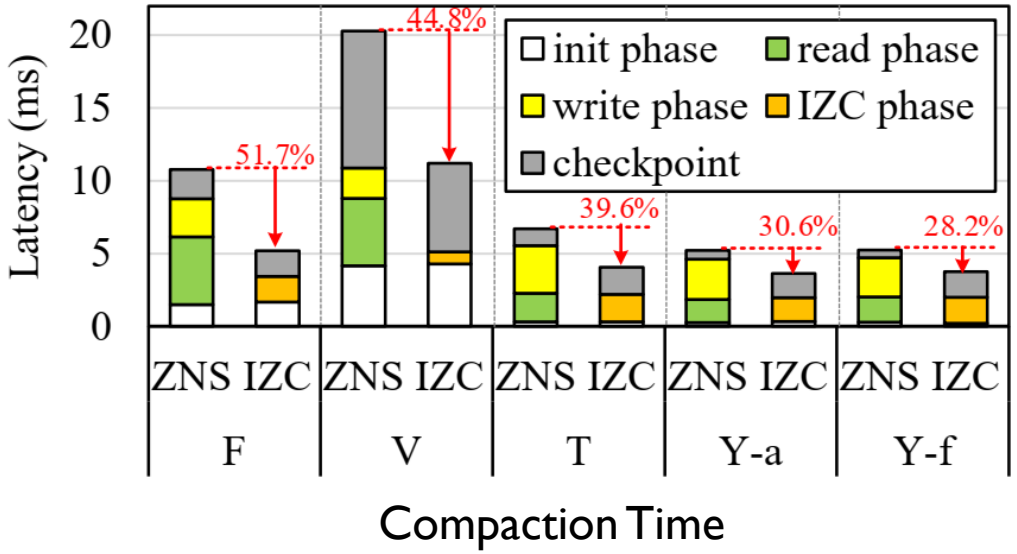
https://github.com/ucare-uchicago/femu

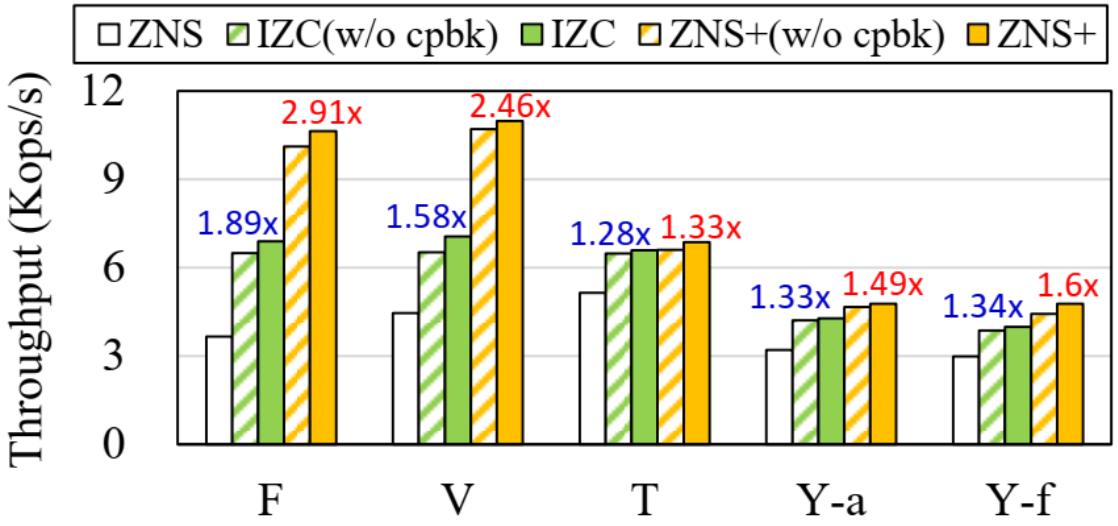


Cosmos+ OpenSSD

Performance

- IZC
  - 28.2–51.7% reduction at compaction time
  - 1.3–1.9x higher throughputs than ZNS
- ZNS+
  - 1.3–2.9x higher throughputs than ZNS
  - > 86% of the reclaimed segments are handled by threaded logging
  - Metadata write traffic reduction by 48%



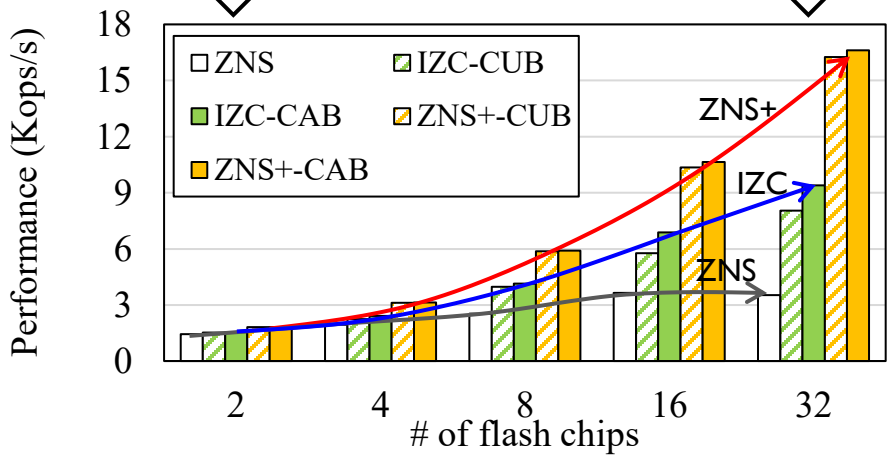


Flash Chip Parallelism

- ZNS+ shows a faster increase rate on performance
  - Increased compaction cost can be hidden in the background by the internal plugging.

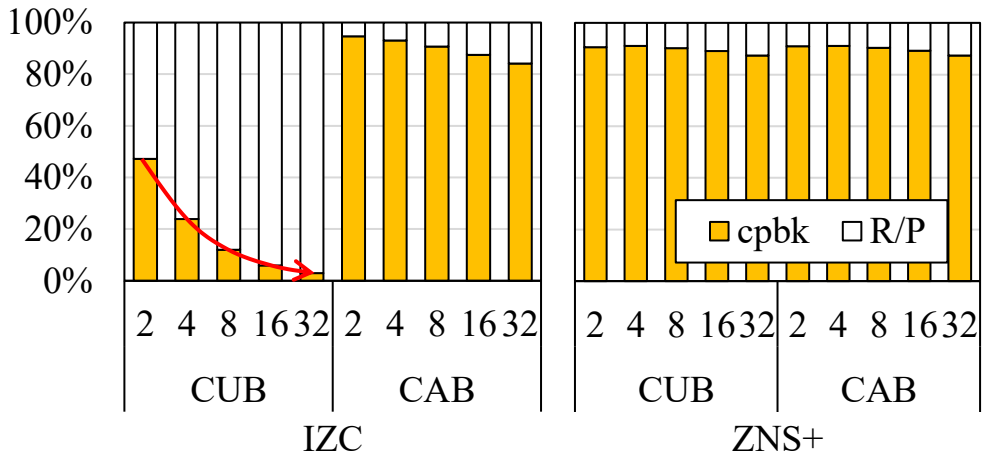
Low internal B/W
ZNS ≈ ZNS+

High internal B/W
ZNS << ZNS+



# of flash chips ↑ ➔ Zone size ↑ ➔ Compaction cost ↑

- Copyback-Aware (CAB) vs. -Unaware (CUB)
  - CUB: cpbk ratio decreases linearly
  - CAB: > 80% of the copy requests are processed by copyback
  - ZNS+ increases cpbk ratio w/ thread logging (internal plugging)



copyback (cpbk) and read-and-program (R/P) distribution

Conclusion

- SSD evolution
  - Black-box model – Regular SSD
    - Log-on-Log, Unpredictable Delay
  - Gray-box model – Multi-streamed SSD
    - Host can specify the stream ID of each write request ➔ GC optimization
  - White-box model – Open-Channel SSD, ZNS
    - Exposes SSD hardware geometry to the host
    - Current ZNS imposes a high storage reclaiming overhead on the host to simplify SSDs.
  - ZNS+
    - Place each storage management task in the most appropriate location
    - Make the host and the SSD cooperate
- Code is available
  - https://github.com/eslab-skku/ZNSplus

Thank You

Further Questions? dongkun@skku.edu