ZNS+ Simulator

Prof. Dongkun Shin (dongkun@skku.edu)
TA – Youngjae Lee(yjlee4154@gmail.com)
TA – Jeeyoon Jung(wjdwldbs1@skku.edu)
Embedded Software Laboratory
Sungkyunkwan University
http://nyx.skku.ac.kr

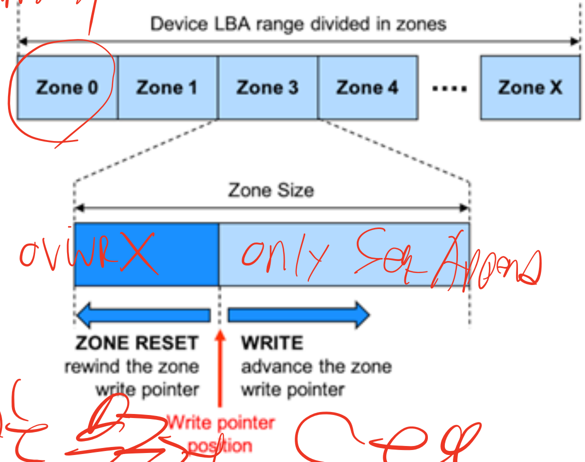
ZNS SSD

Zoned Namespace FW 복잡

- SSD interface exposing NAND flash chracteristics
- Zoned Interface
  - Logical address space is divided into fixed-size zones
  - Each zone must be written in sequential order
  - User explicitly reset zone to erase data
- Benefits
  - GC-less
    - Predictable latency
  - Less resource usage
    - DRAM
    - Over-provision

← GC 없으니

Mapping 없음 : 수십GB


Device LBA range divided in zones
Zone 0
Zone 1
Zone 3
Zone 4
....
Zone X
Zone Size
ZONE RESET
rewind the zone write pointer
WRITE
advance the zone write pointer
Write pointer position

ovwrX

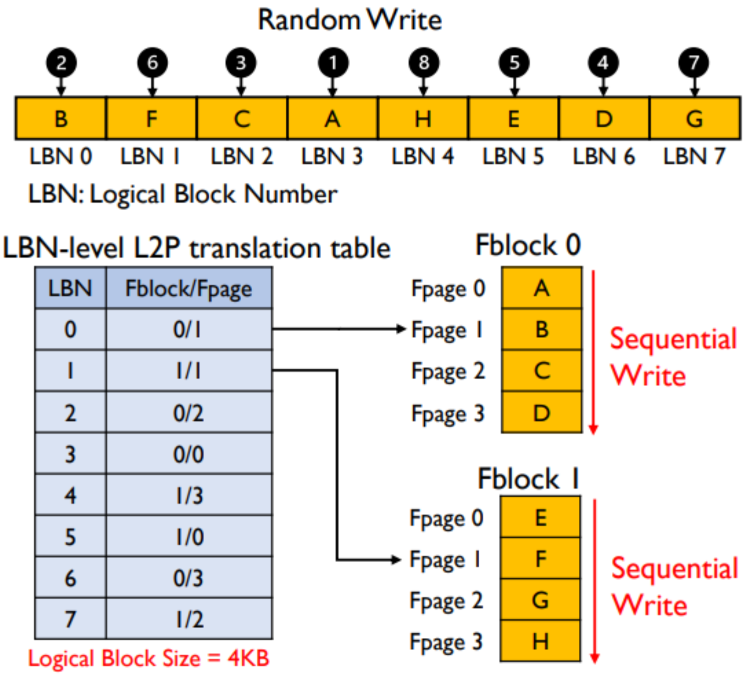
only Seq Append

Zone 단위 Seq.

Why ZNS? Small L2P Table

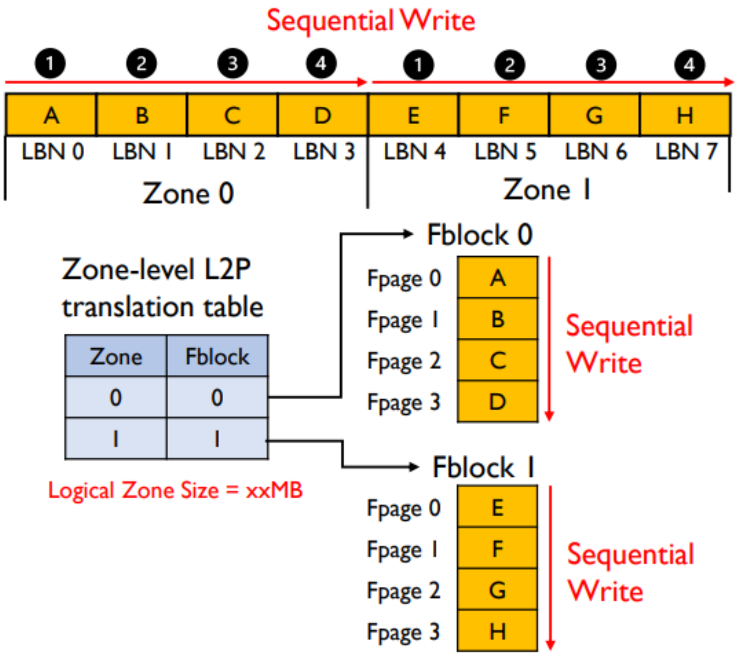
일반 => L2P 大

여러 LBN → Zone.



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

Regular SSD

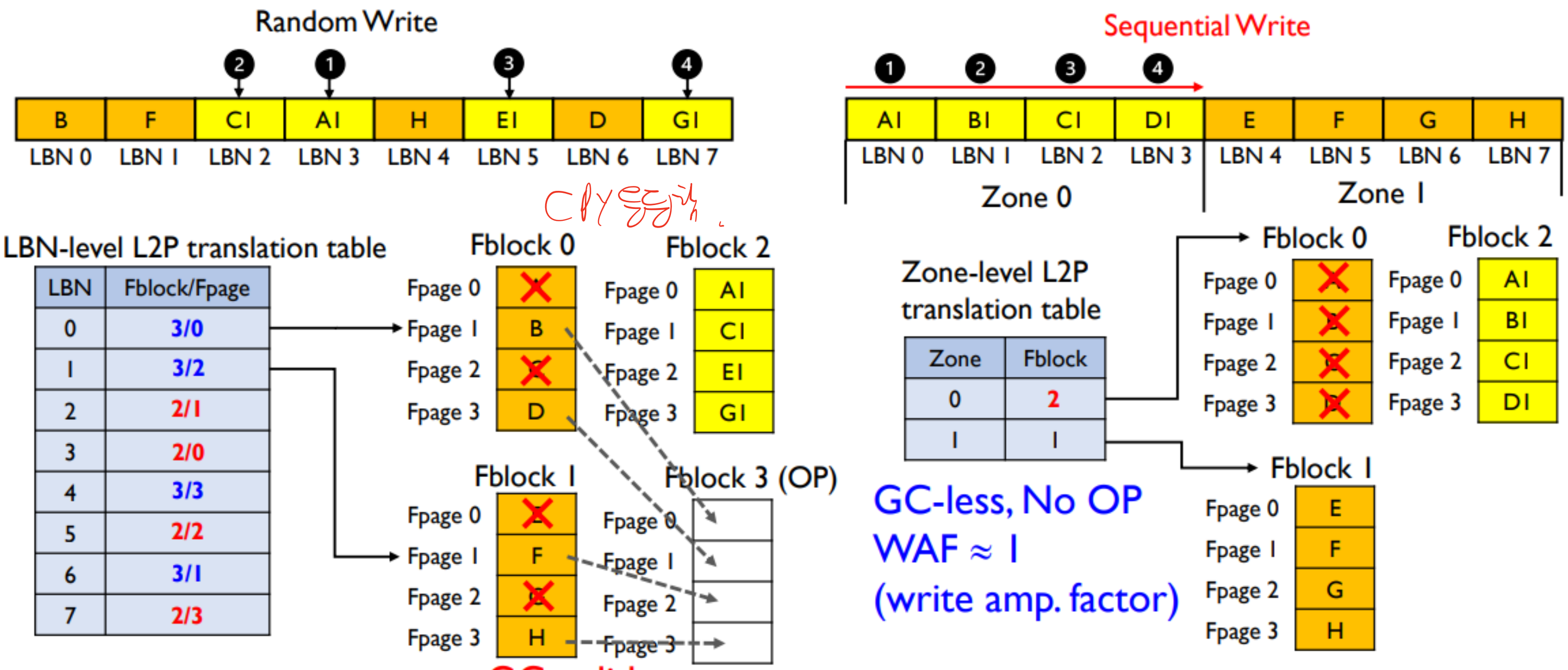


Zone	Fblock
0	0
1	1

ZNS SSD

Why ZNS? GC-less, Predictable

애초에 overWR X
User가 erase 함.



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

Zone-level L2P translation table

Zone	Fblock
0	2
1	1

GC-less, No OP
WAF ≈ 1
(write amp. factor)

Regular SSD

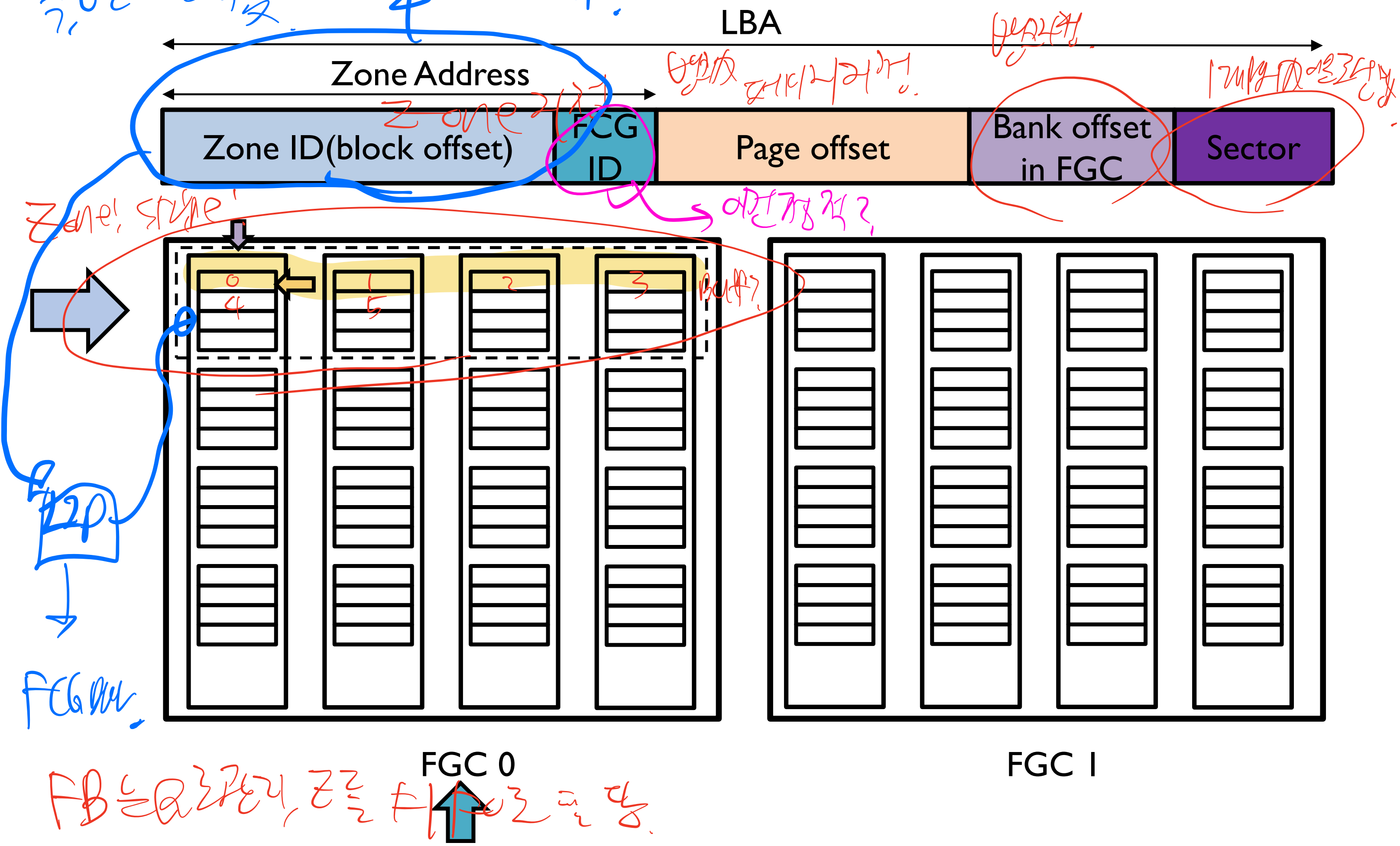
GC: valid page copy
➔ write amplified, unexpected delay

ZNS SSD

ZNS striping policy

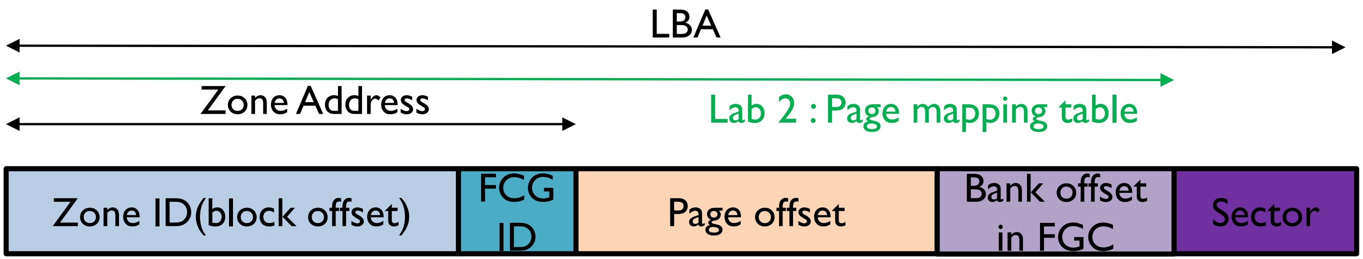
즉, 0번 zone은 [illegible]

정적X! - 고정.



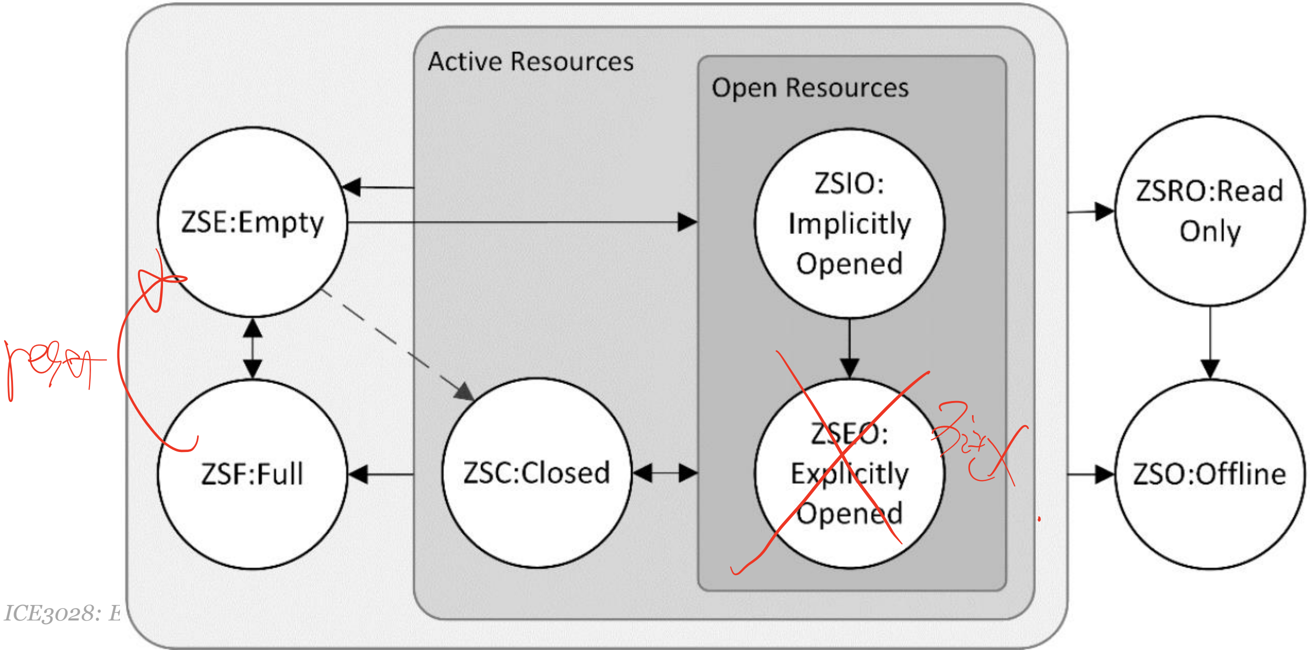
ZNS striping policy

- Must use Zone Address → Physical Address mapping table in SSD side DRAM
  - Lab2 : page mapping table use X 2^(Page offset + Bank offset in FGC) DRAM than ZNS simulator
- Total bank : 2^(FCG ID + Bank offset in FGC)



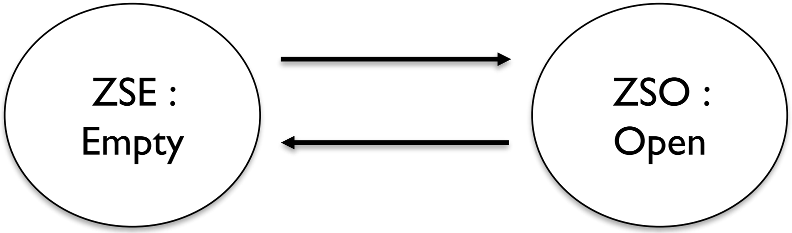
Zone State Machine

- In NVMe spec
  - Manage total 7 state
  - Devide Open Resources & Active Resources
    - Max open Zone & Max Active Zone



Zone State Machine

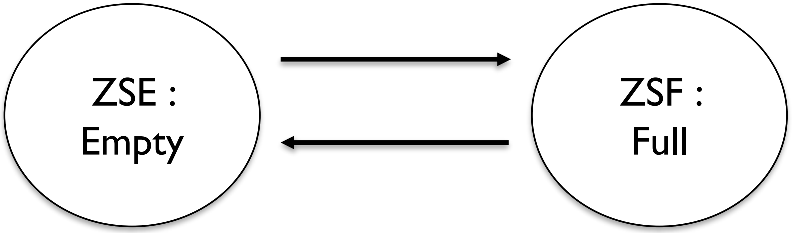
- Empty → Open
  - When – Host writes first LBA of zone, Zone opened automatically
  - Host sends ~~**Zone Management Send**~~ Command (Empty → Open) 필요X
  - If open resource is full, then report error
- Open → Empty by Reset. (Full 아니어도 ㅇㅋ)
  - Host sends **Zone Management Send** Command (Open → Empty)



Zone State Machine

존이 [illegible] 자동적

- Empty → Full
  - ~~Host sends **Zone Management Send** Command (Empty → Full)~~
- Full → Empty
  - Host sends **Zone Management Send** Command (Full → Empty)



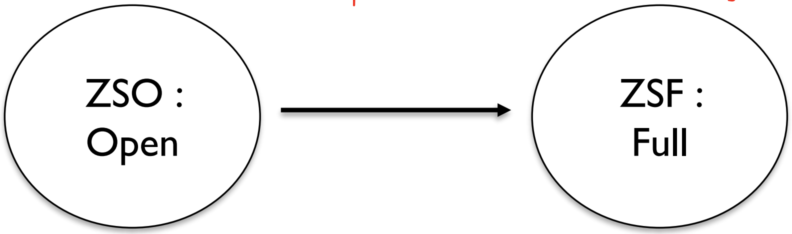
ZSE : Empty

ZSF : Full

Zone State Machine

- Open → Full
  - Host sends **Zone Management Send** Command (Open → Full) 1. 억지로 Command
  - Host writes last LBA of zone, Zone Finished automatically 2. → Zone 끝.
- Full → Open
  - Full state zone must be reset and open
  - It can't be happened, if this occur return error

Full → open 반드시 Reset 거침.



ZSO : Open

ZSF : Full

Zone Management Command

- Zone_Descriptor : must manage in DRAM to manage Zone State
  - Manage Zone's Start LBA, Size, Write Pointer, State

- Zone management Send & Receive
  - Zone Management Send
    - Change state of the zone
  - Zone Management Receive
    - Return Zone Descriptor

=> " Zone 정보를 RST / FNSH 시켜라! "

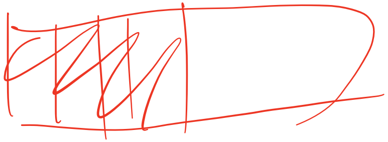
" Zone 정보의 metadata? "
([illegible])

Zone의 시작LBA
size
상태, 어디까지 썼는지

DRAM usage

- Zone_Descriptor ✓ ← zone 개수대로
- Logical Zone to Physical Zone mapping table ✓
- Data buffer(must keep not page aligned data)
  - Keep # max open zone Data buffer(size = 1page)

4씩에 써놓니?



⇒ 버퍼링! (∵ 최소 PG 단위)

ZNS+

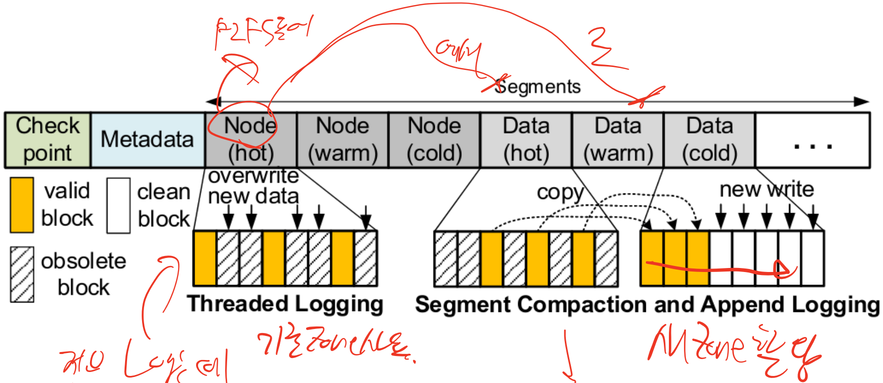
ZNS+

- **ZNS+: Advanced Zoned Namespace Interface for Supporting In-Storage Zone Compaction**
  - **OSDI'21**
    - **https://www.usenix.org/conference/osdi21/presentation/han**
  - **ZNS extension for F2FS file system**
    - **Offload file system management to ZNS SSD**

F2FS (Flash-Friendly File System)

- F2FS → overwrite안되는 FS 예 block 6번을 새zone3번으로!
  - One of actively maintained Log-structured File Systems (LFS)
    WAF:
  - Six types of segments: hot, warm, and cold segments for each node/data
    Host는 valid, invalid 가려 알려주나? (FS) ← 응용 ○.
  - Multi-head logging
  - Internal GC to reclaim invalid blocks
    - Append logging (AL)
    - Threaded logging (TL)
  - Support ZNS SSD

다른 모드로.

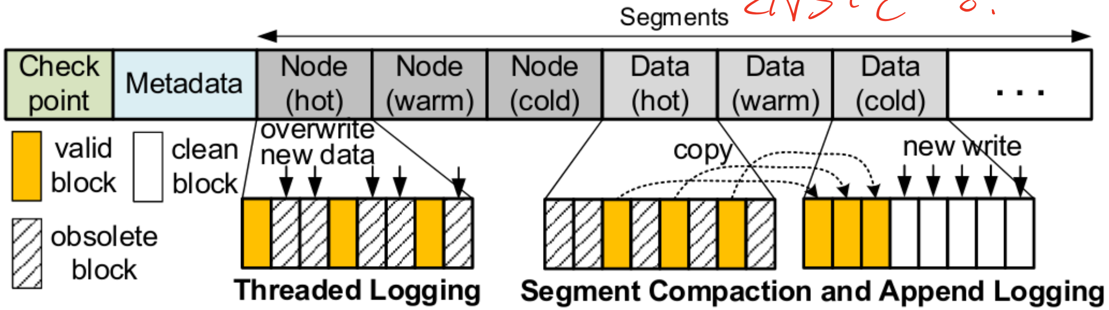


zns: Host의 명시적 도움.

F2FS (Flash-Friendly File System)

- Garbage Collection
  - Segment Compaction
    - Select a victim segment with lowest compaction cost
    - Copy valid data to new free segment via host-initiated R/W
    - Update metadata
  - ~~Threaded Logging~~ → overwrite! (강하게 금지)
    - ~~Enabled when free segment become insufficient~~
    - ~~Overwrite invalid blocks with new data~~
    - Disabled for ZNS SSD

ZNS 가려 (: SWR)
→ ZNS는 금지.



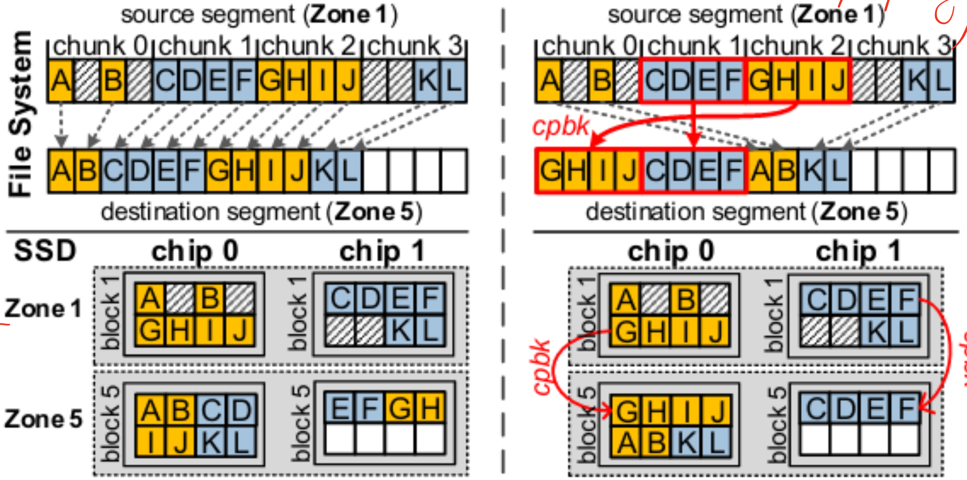
ZNS+: LFS-Aware ZNS

- **Internal Zone Compaction** 중요!
  - **Accelerate segment compaction**
  - **Copy blocks within SSD**
    - **Reduce host-device traffic**

이번 file 단위마다 (그차이)

가능하다

Copy



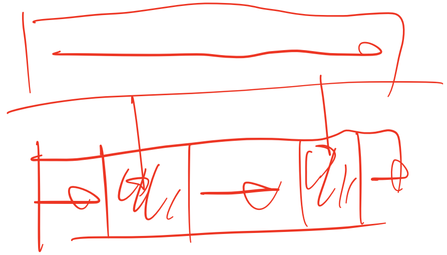
(a) normal block allocation

(b) copyback-aware allocation

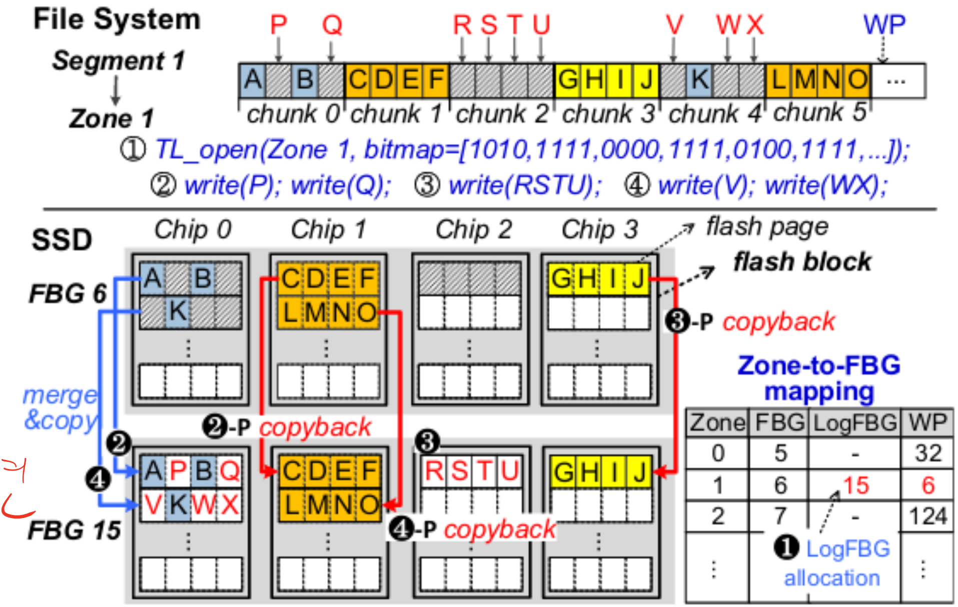
ZNS+: LFS-Aware ZNS

- **Sparse Sequential Write** 중간②
  - **Host can overwrite a zone sparsely**
    - **Support F2FS threaded logging on ZNS SSD**

threaded Logging 주의



중간 Implement 주의



① TL_open(Zone 1, bitmap=[1010,1111,0000,1111,0100,1111,...]);
② write(P); write(Q); ③ write(RSTU); ④ write(V); write(WX);

Zone-to-FBG mapping

Zone	FBG	LogFBG	WP
0	5	-	32
1	6	15	6
2	7	-	124
⋮			⋮

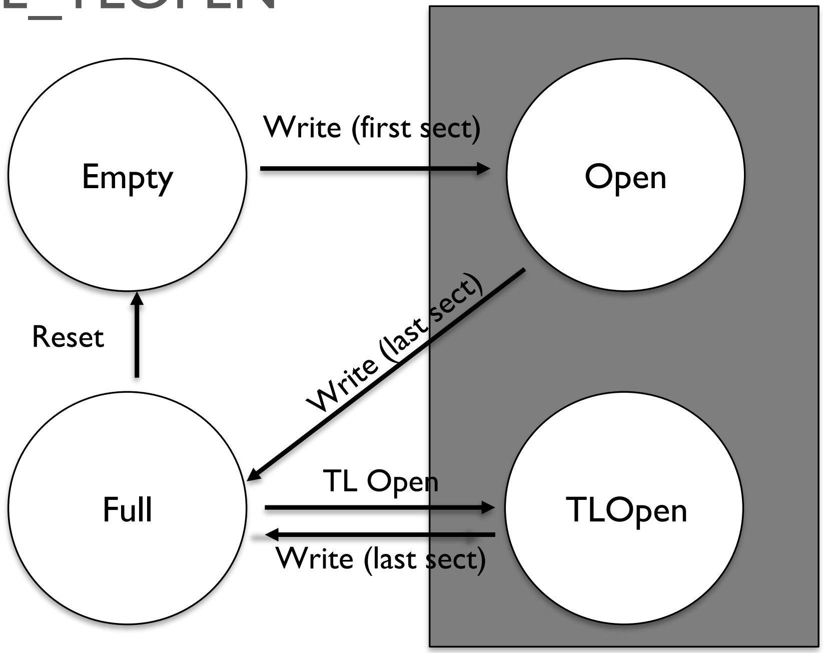
Lab 5 : ZNS+ Simulator

ZNS+ Simulator

- Develop a ZNS+ simulator
  - Simulate the operations ZNS SSD
    - Zoned I/O interface: Read, Write, Reset , Finish
      LBA단위, zone 단위로 LBA 범위.
    - Zone metadata
  - We also implement ZNS+ operations
    - Zone Compaction ✓
    - Threaded Logging ✓

ZNS+ Simulator

- Zone State for our Assignment
  - Total 4 states
  - Manage Open resource for ZONE_OPE and ZONE_TLOPEN

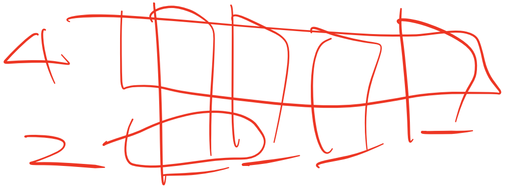


ZNS Operations

동적세팅.

- zns_init(nbank, nblk, npage, dzone)
  - Initialize ZNS simulator
    - Zone descriptors (state, write pointer, start lba)
    - Buffer for unaligned write
    - Internal metadata (zone-to-fbg, threaded logging, free fbg list)
  - nbank, blk, npage: NAND flash dimension
    - nbank, npage: power-of-two
  - Dzone: zone interleaving degree
    - Degree is guaranteed to be a power-of-two

FBG랑 연결?



각 Open된 Zone 마다 1개 FBG 크기의 버퍼를 SRAM에 둔다.

ZNS Operations

여러 zone에 걸치는 일X 발생!

- zns_write(lba, nsect, data)
  - If `lba` does not match to the write pointer or zone is in `ZONE_FULL` state

    모두 seq하게 써야하므로.
    - Write fails and return -1
  - If `lba` is start of the zone

    open zone수 ↑↑
    - Change the zone state to `ZONE_OPEN` and increase open count
    - If total open zone count has already exceeded `MAX_OPEN_ZONE`, then write fails and return -1
    - Allocate FBG
      - Free FBG should be managed in FIFO manner

    큐[illegible]

큐에서 넣고 [illegible] 빼야함

ZNS Operations

- znsֹ_write(lba, nsect, data)
  - Write data to the NAND Striping 주의
    - Stripes data across the banks mapped to the zone
    - If the data does not fit into the NAND page, store in buffer 8 sct
      ↳ over되면 page 미만은 buffer!
    - Increase `wp` by `nsect`
    - If the last sector of zone is written, change state to ZONE_FULL
  - [lba, lba+nsect) range is guaranteed to be included within single zone opencnt ←—;
) 쪼개버려

ZNS Operations

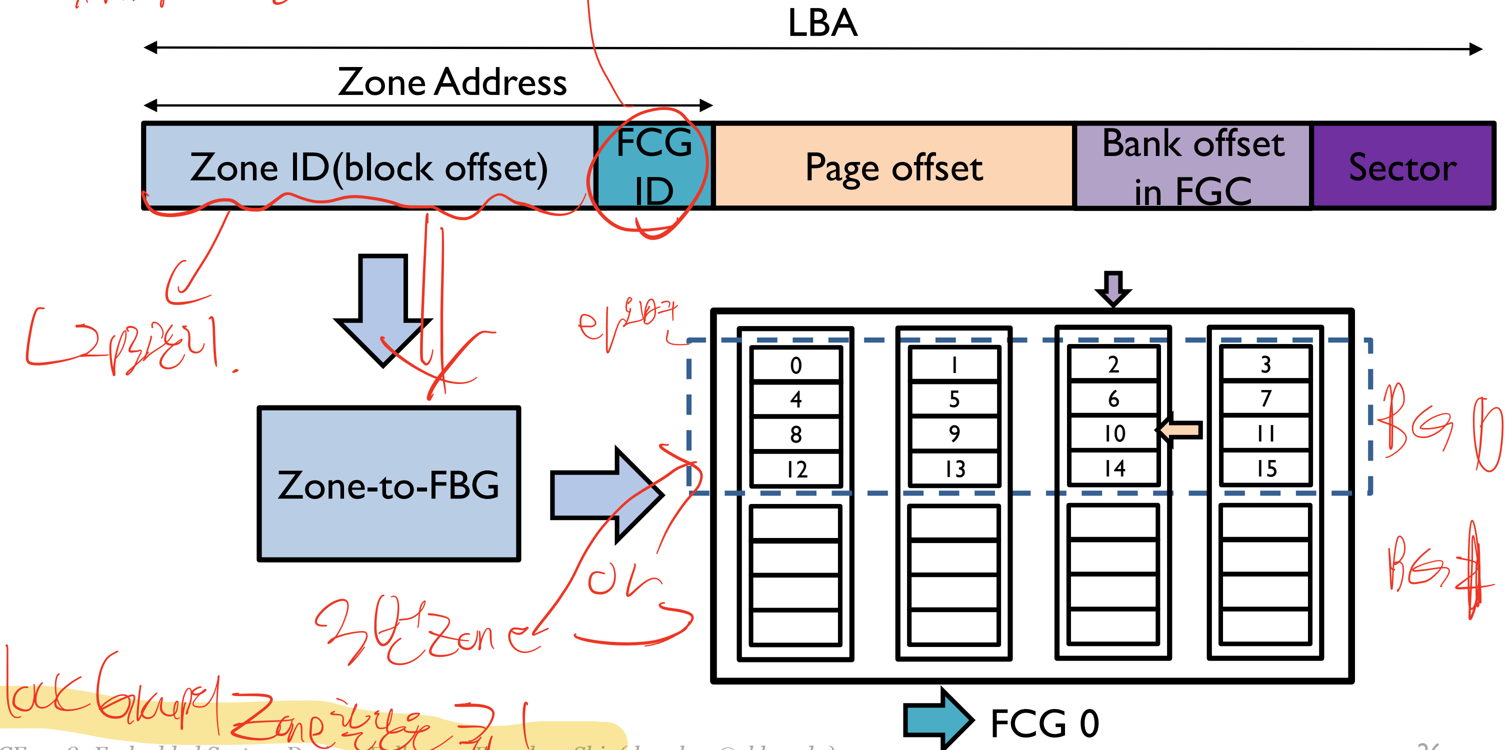
FCG: Flash Chip Group

디오론에서부에 따라 다르게 동작.

- zns_write(lba, nsect, data) (cont.)
  - Write striping

ex) 32 Bank에서 4개씩이면 FCG는 8개

striping 개는 잘 해야함.



Block Group이 Zone 같은 거!

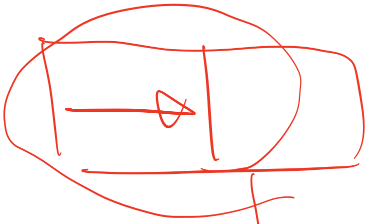
ZNS Operations

- zns_read(lba, nsect, data)
  - Fill `data` with written data and return
  - Read has no restriction (max open, writer pointer)
  - If target sector is not written yet, fill all bytes with `0xff`
  - You should also check the data stored in the buffer
  - [lba, lba+nsect) range is guaranteed to be included within single zone

Write가 쌓이므로

Zone 걸쳐진 없음.

Buffer에도 있지!



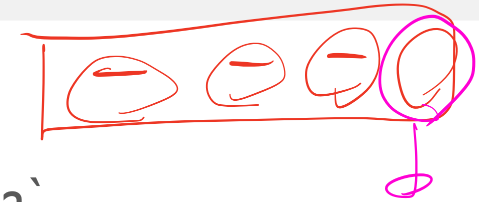
empty 하면 0xff다.

0xff 리턴!

ZNS Operations

Zone의 시작지점 입력.

- zns_reset(lba)
  - Reset the zone starting with given `lba`
    - Erase all NAND blocks in target zone
  - Return -1 if target zone is not in ZONE_FULL state
  - Zone metadata
    - Reset `wp` to the `slba`
    - Change `state` to `ZONE_EMPTY`



이번 erase X?

ㅇㅋ.

[illegible]

[illegible]

시작lba 가리키게.

zns_finish없는듯?

ZNS Operations

메타데이터 가져오기 descriptor 가져오기 =>그래서 [illegible]

- zns_get_desc(~~szone~~, nzone, descs) 여기에 채워주기

  lba

  - Fill `descs` with the zone descriptors
    - `state`: zone state
    - `slba`: start address of the zone
    - `wp`: write pointer of the zone

    메타 데이터

  - `nzone` descriptors from `szone` to `szone+nzone-1`
  - The length of the `descs` array is `nzone`

ZNS+ Operations

internal zone compaction

full 상태의 zone을　　빈 zone으로 copy

- zns_izc(src_zone, dest_zone, copy_len, copy_list)
  - Copy sectors from source zone to destination zone
  - Append sectors in `copy_list` from the beginning of the destination zone
    - Each entry is sector offset from the beginning of the zone
  - Reset source zone
  - If total open zone count has already exceeded `MAX_OPEN_ZONE`, then write fails and return -1
  - Return -1 if target zones are not in suitable state
    - src_zone != dest_zone
    - src_zone: ZONE_FULL, target_zone: ZONE_EMPTY

~~BANK에~~

1줄이 여러 BANK에 걸쳐있는?

ex) (예의

d개의 블록을 [illegible] data는 넣게

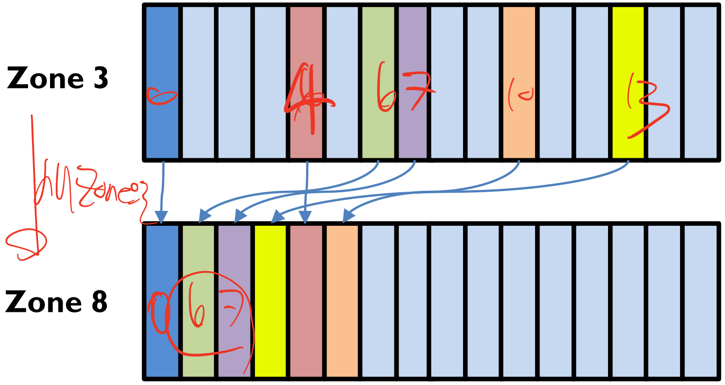
ZNS+ Operations

- zns_izc(src_zone, dest_zone, copy_len, copy_list)
  - Example

IZC(3, 8, 6, {0, 6, 7, 13, 4, 10})

3zから8zまで

6개 src cpy, 2칸-



화이팅!

ZNS+ Operations 원래', 쓸께 write 해면 그대로 이어

Zone 번호인거!!!

- zns_tl_open(zone, valid_arr) 에 들어갈
  - Open a zone for threaded logging
    - Allow sparse overwrite zns_write on TL opened zone jmp 표 cnt++
    - Increase open zone count same as normal open 0 | 이면 jmp 뒤 cpy
      - Allocate one more FBG for target zone
    - Sectors specified in bitmap will be skipped by host write
      - Copy skipped sectors from original NAND blocks
    - Target zone should be in `ZONE_FULL` state
      - If not, return -1
  - Each bank has `MAX_OPEN_ZONE` OP blocks for TL
  - After the zone becomes FULL, erase source NAND blocks in FBG

: OP블록.

ZNS+ Operations [illegible]X (: [illegible] 이[illegible] 함)

- zns_tl_open(zone, valid_arr) (cont.)

zone 1개 16개 [illegible]

1인줄, jmp

TL_OPEN(3, {1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1})

thread 의 [illegible]

WRITE(49, 3)

WRITE(53, 1)

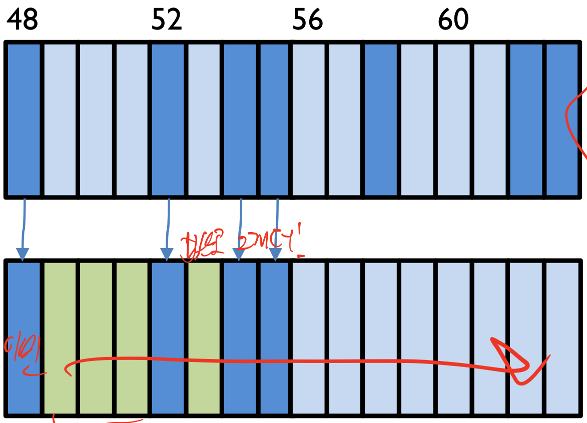
[illegible]

READ(48, 4) from new zone

READ(56, 4) from old zone

read 시에는

[illegible]! 느림 대박!



Zone 유지 (내부적으로는)

다른 [illegible] erase 된다

Zone becomes FULL after write sector 61

f: 물, Physical
논: Logical

[illegible] 1개 zone이 2개 [illegible] 유지 된다

Grading Policy

- Recommended environment : GCC on Linux
  - You can do it in Windows, but be sure that your work also runs in Linux
  - Use only standard C library or POSIX C library functions

- Personal Project

- Submissions will be graded based on the number of test cases passed
  - We will use larger test cases for scoring

- Submit to the icampus
  - Due: 10/31(Sun.) 23:59:00
  - Submission file name: `<student_id>.tar.gz` (includes `zns.c` only)
  - Modify student id in `Makefile` and use `make submit` command

- Late penalty : -20 % / day (Up to 3 days)

Q&A