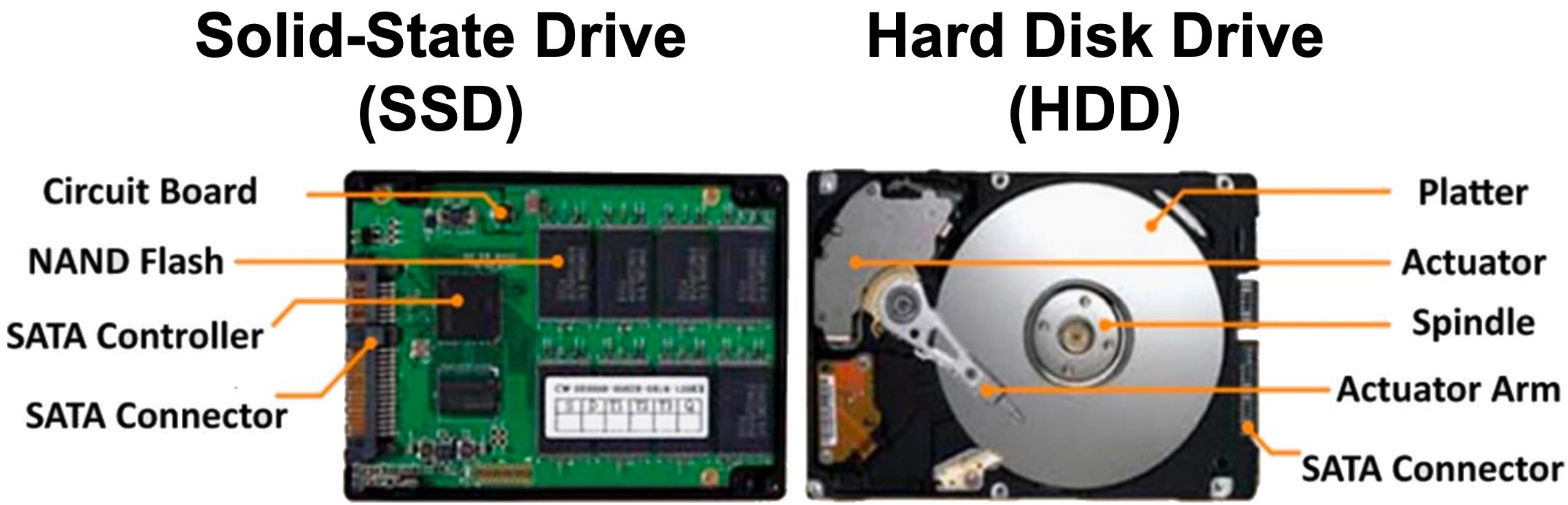
Operating Systems

Lecture 16: File Systems Part IV: Solid-State Drive

Hong Xu

https://github.com/henryhxu/CSCI3150

Solid-State Drive vs. Hard Disk Drive (SSD vs. HDD)



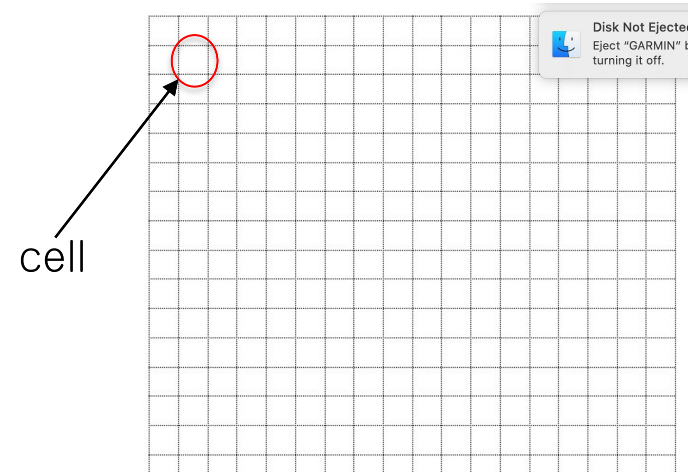
- SSD's advantages:
  - Faster performance
  - No vibrations or noise; shock-resistance
  - Lighter, smaller

Overview

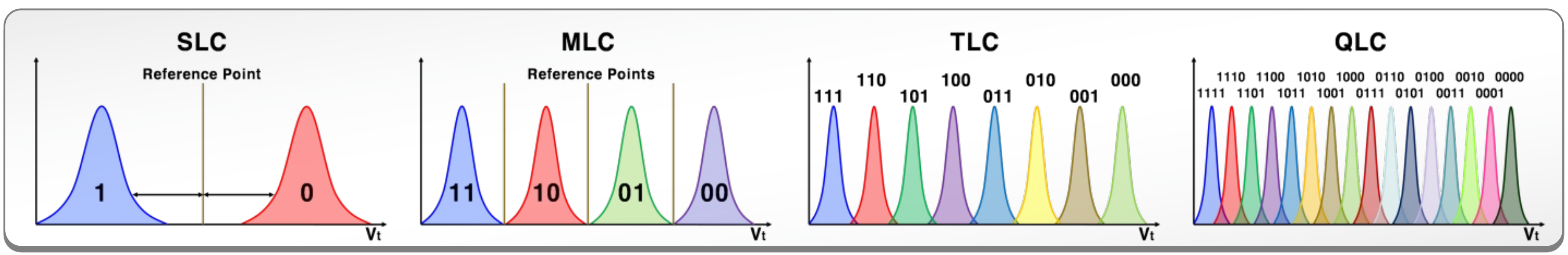
- Solid-state storage device
  - No mechanical or moving parts like HDD.
  - Built out of transistors (like memory and processors).
  - Retain information despite power loss unlike typical RAM.
- NAND Flash based SSD
  - **To write** to a given chunk of it, you **have to erase** a bigger chunk.
  - The number of erase/write is limited.

Storing Bits

- Single-level cell (SLC): a single bit per cell
- Multi-level cell (MLC): two bits per cell
- Triple-level cell (TLC): three bits per cell
- QLC, PLC



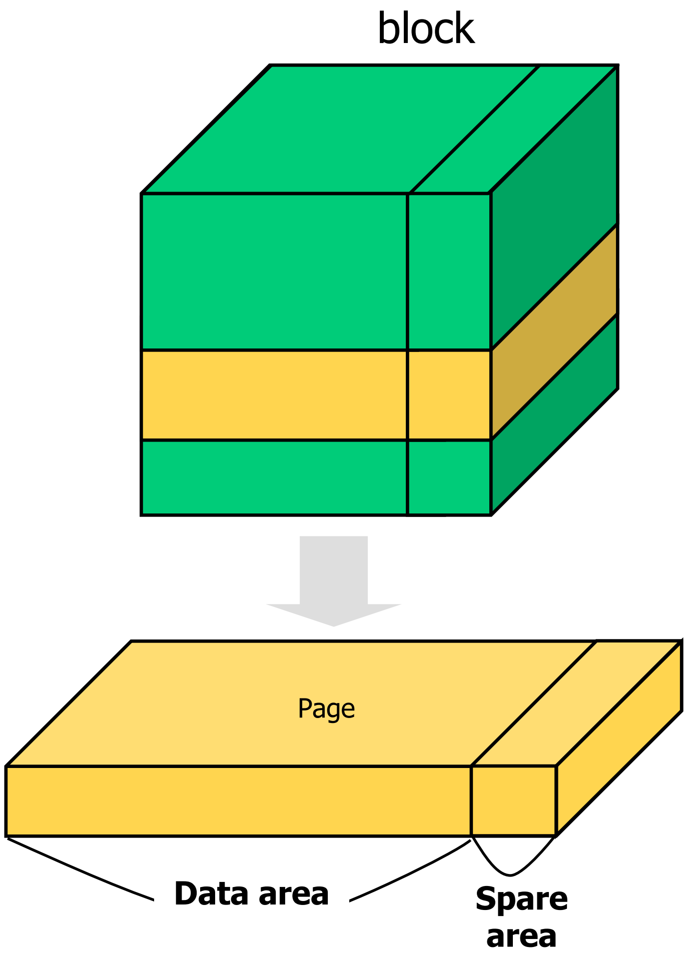
Flash memory



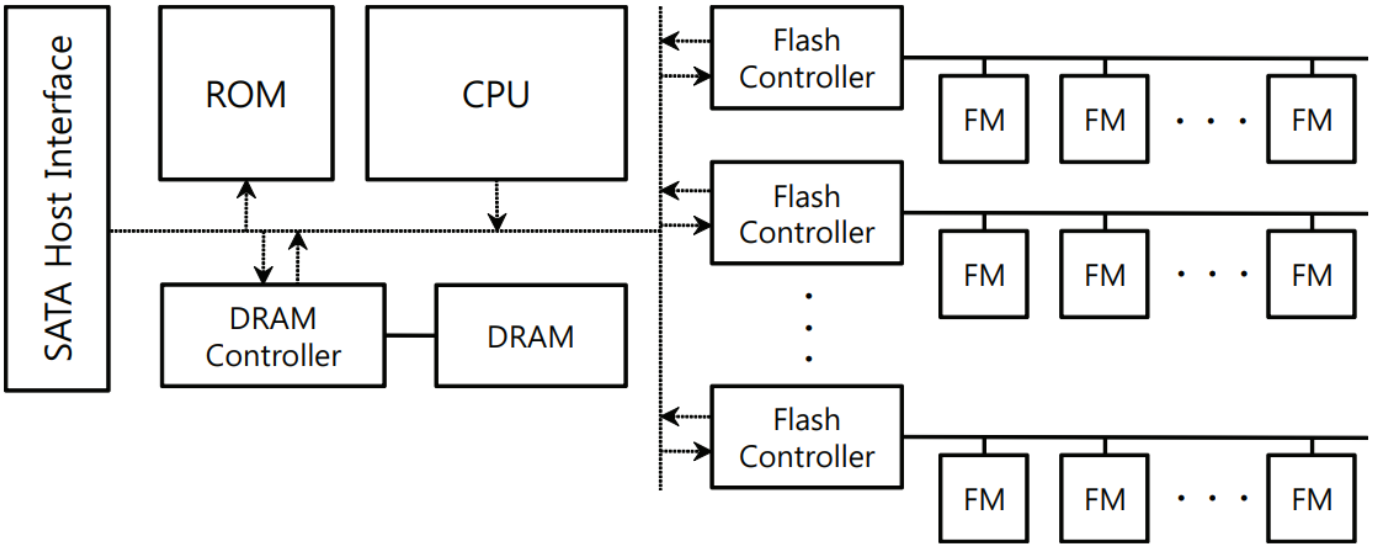
Structure of Flash

Flash cells -> Pages

Pages-> Blocks

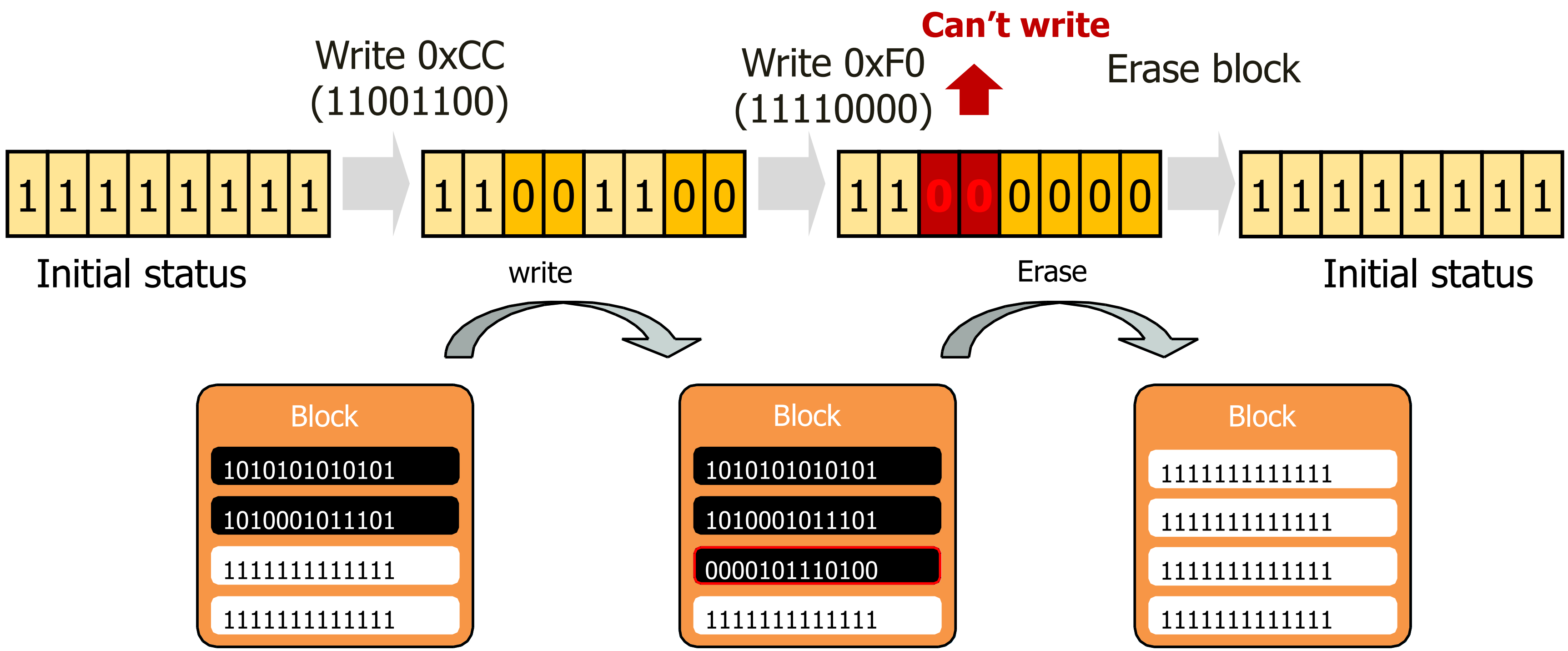


Structure of Flash SSD



Asymmetric Operation Units

- Read; Write (program): 1 → 0: in page unit.
- Erase: 0 → 1: in block unit
- Write-once property: A flash page cannot be overwritten until the residing block is erased first.

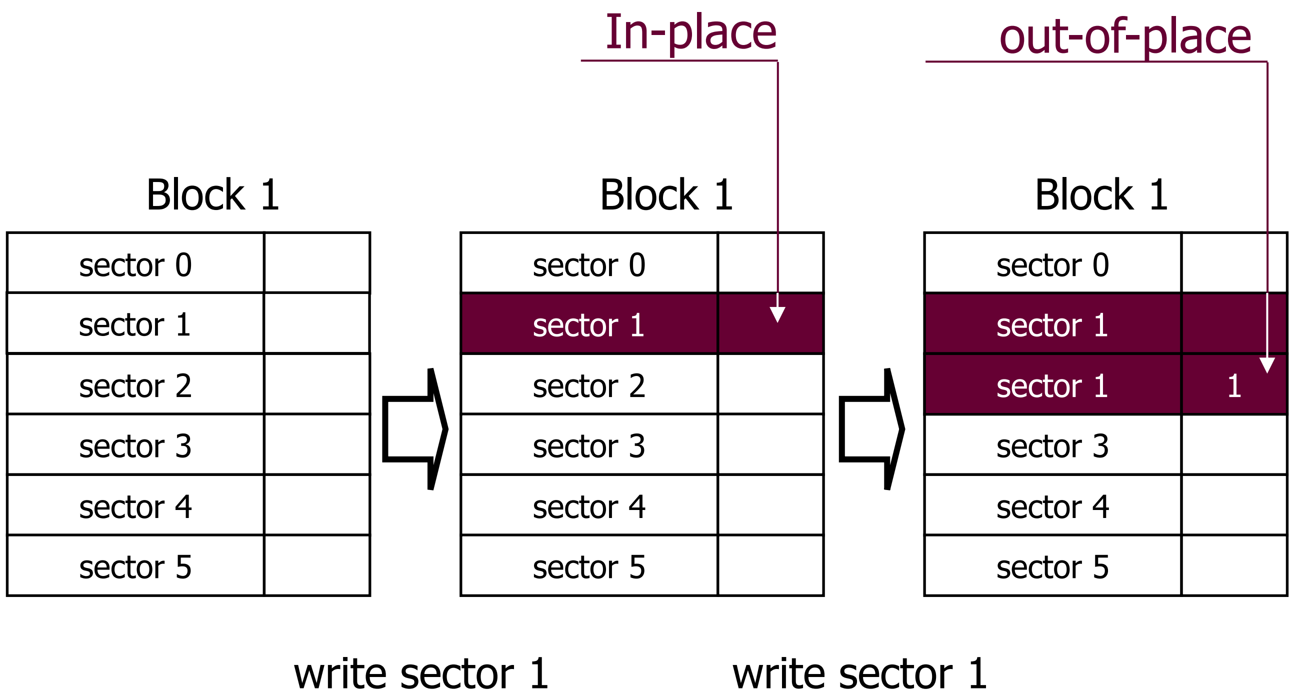


Reliability of Flash

- Wear out
  - Flash cells wear out as we program/erase it (P/E cycles)
  - Eventually the block becomes unusable.
  - Typical erase/wear out cycle
    - MLC-based block: 10,000 P/E cycles
    - SLC-based block: 100,000 P/E cycles
- Disturbance
  - When accessing a page, it is possible that some bits in the neighboring pages get flipped (interference)
  - It is called read disturbance or program disturbance.

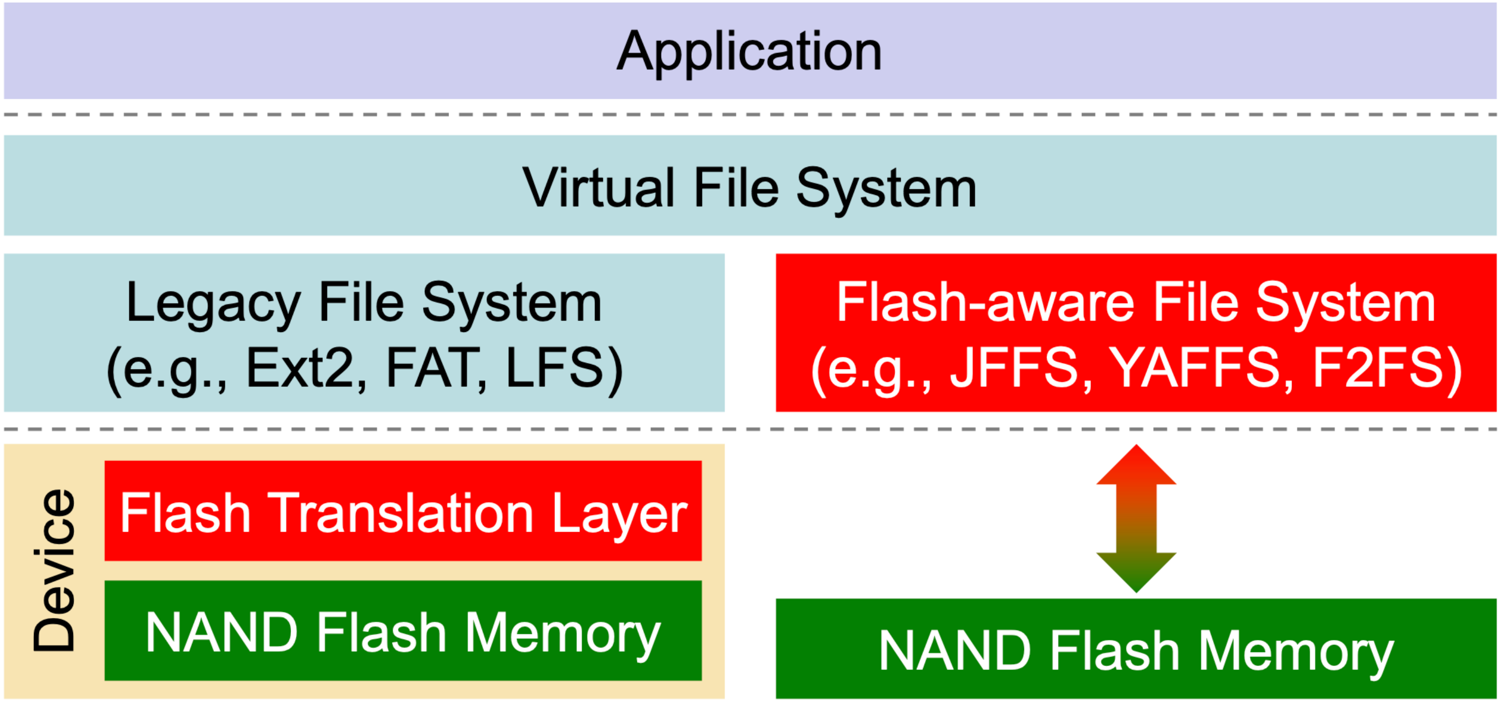
Out-of-place update in Flash memory

- Flash memory should be erased before written.
- Flash SSD uses out-of-place update for write operation.



System Architecture

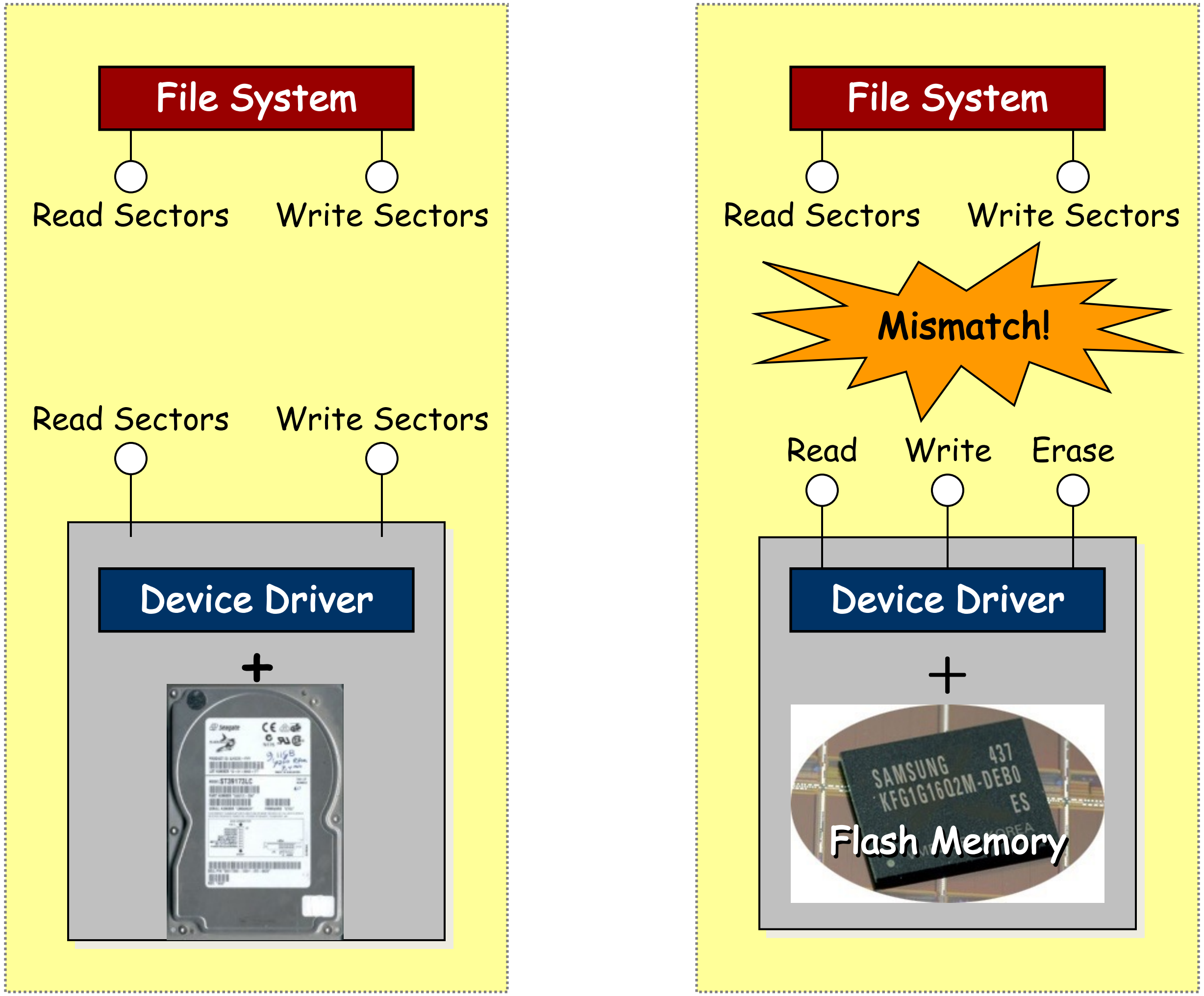
- There are two typical ways to address the inherent challenges of flash memory
  - Implementing a **flash translation layer** in the **device**
  - Designing a **flash-aware file system** in the **host**



Flash Translation Layer

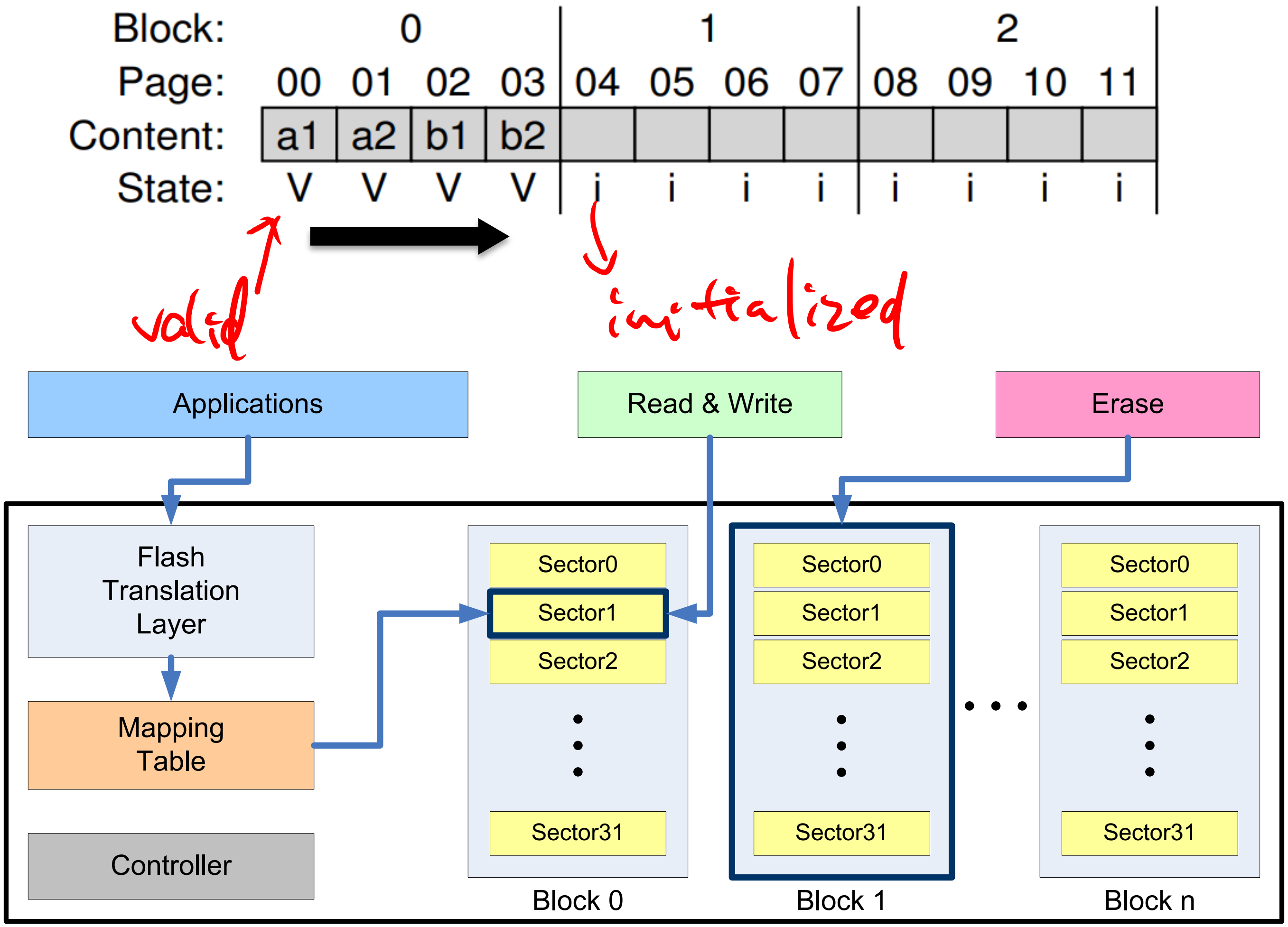
- Software that makes the SSD look like HDD
  - Address translation
    - Program pages within an erased block in order (from low page to high page)
  - Wear leveling
    - FTL should try to spread writes across the blocks of the flash
    - Ensuring that all of the blocks of the device wear out at roughly the same time.
  - Garbage collection

Hard Disk and Flash SSD



Flash Translation Layer

- Append the write to the next free spot in the currently-being-written-to block.



Example

- Assume that a page size is 4 Kbyte and a block consists of four pages.
- Write(~~page number~~) logical block addr.
  - Write(100) with contents a1
  - Write(101) with contents a2
  - Write(2000) with contents b1
  - Write(2001) with contents b2
- The initial state of SSD, with all pages marked `INVALID(i)`.

Block:	0				1				2			
Page:	00	01	02	03	04	05	06	07	08	09	10	11
Content:												
State:	i	i	i	i	i	i	i	i	i	i	i	i

Example

- Erase block 0.

Block:	0				1				2			
Page:	00	01	02	03	04	05	06	07	08	09	10	11
Content:												
State:	E	E	E	E	i	i	i	i	i	i	i	i

- Program pages in order and update mapping information (logical block address to physical page address)

Table: 100 → 0 (Memory)

Block:	0				1				2			
Page:	00	01	02	03	04	05	06	07	08	09	10	11
Content:	a1											
State:	V	E	E	E	i	i	i	i	i	i	i	i

Flash Chip

- After performing four writes.

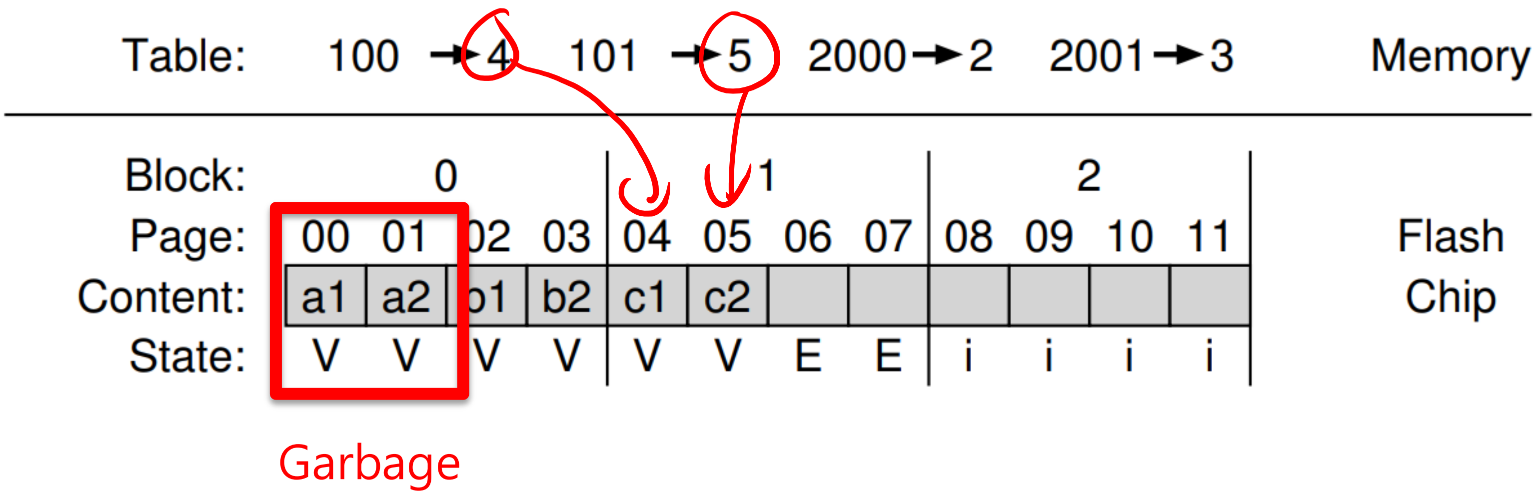
Table: 100 → 0 101 → 1 2000 → 2 2001 → 3 (Memory)

Block:	0				1				2			
Page:	00	01	02	03	04	05	06	07	08	09	10	11
Content:	a1	a2	b1	b2								
State:	V	V	V	V	i	i	i	i	i	i	i	i

Flash Chip

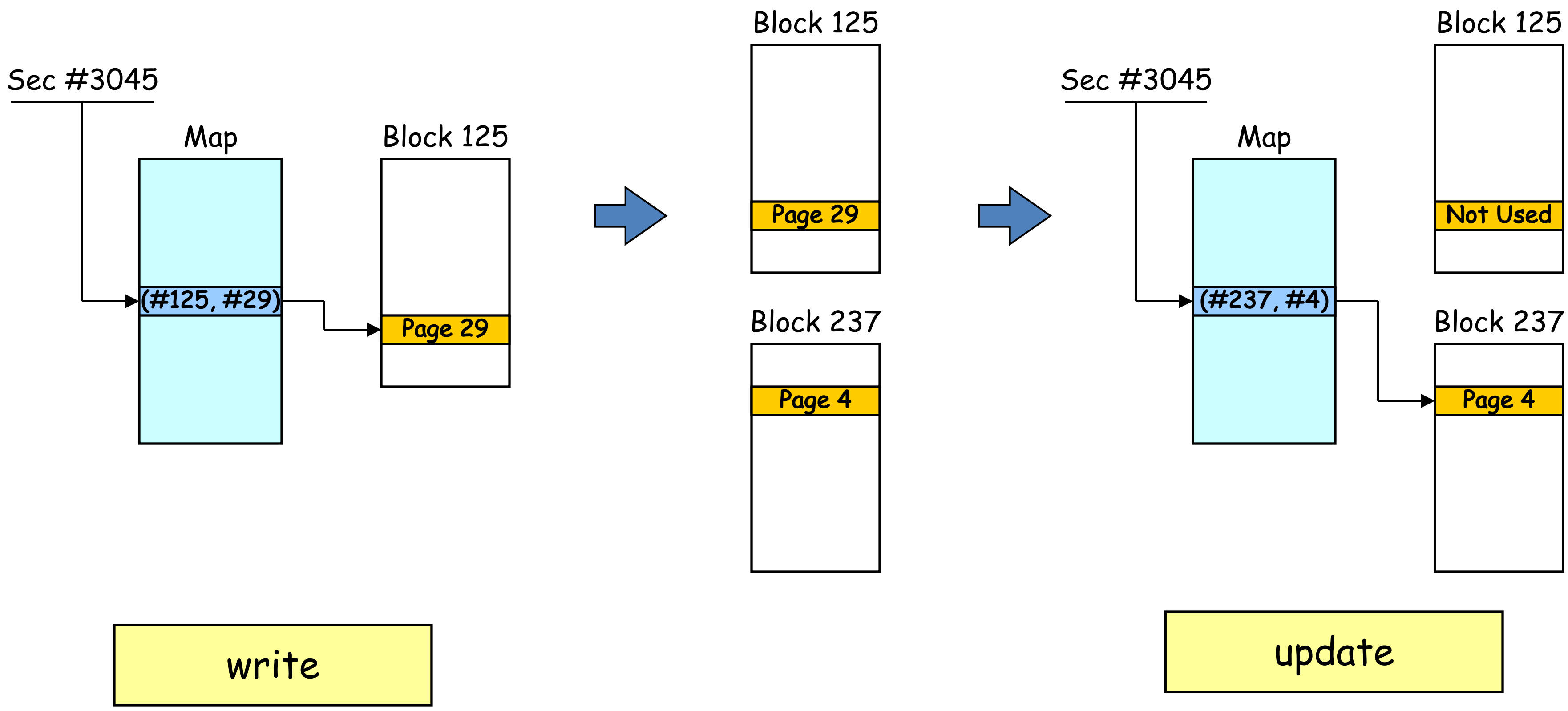
Garbage collection

- Update the existing page → old version of data becomes obsolete.
- Update logical blocks 100 and 101 with contents c1 and c2.



- The pages 0 and 1 in block 0 is old version data and these are called garbage.
- These garbage pages must be reclaimed for new writes to take place.

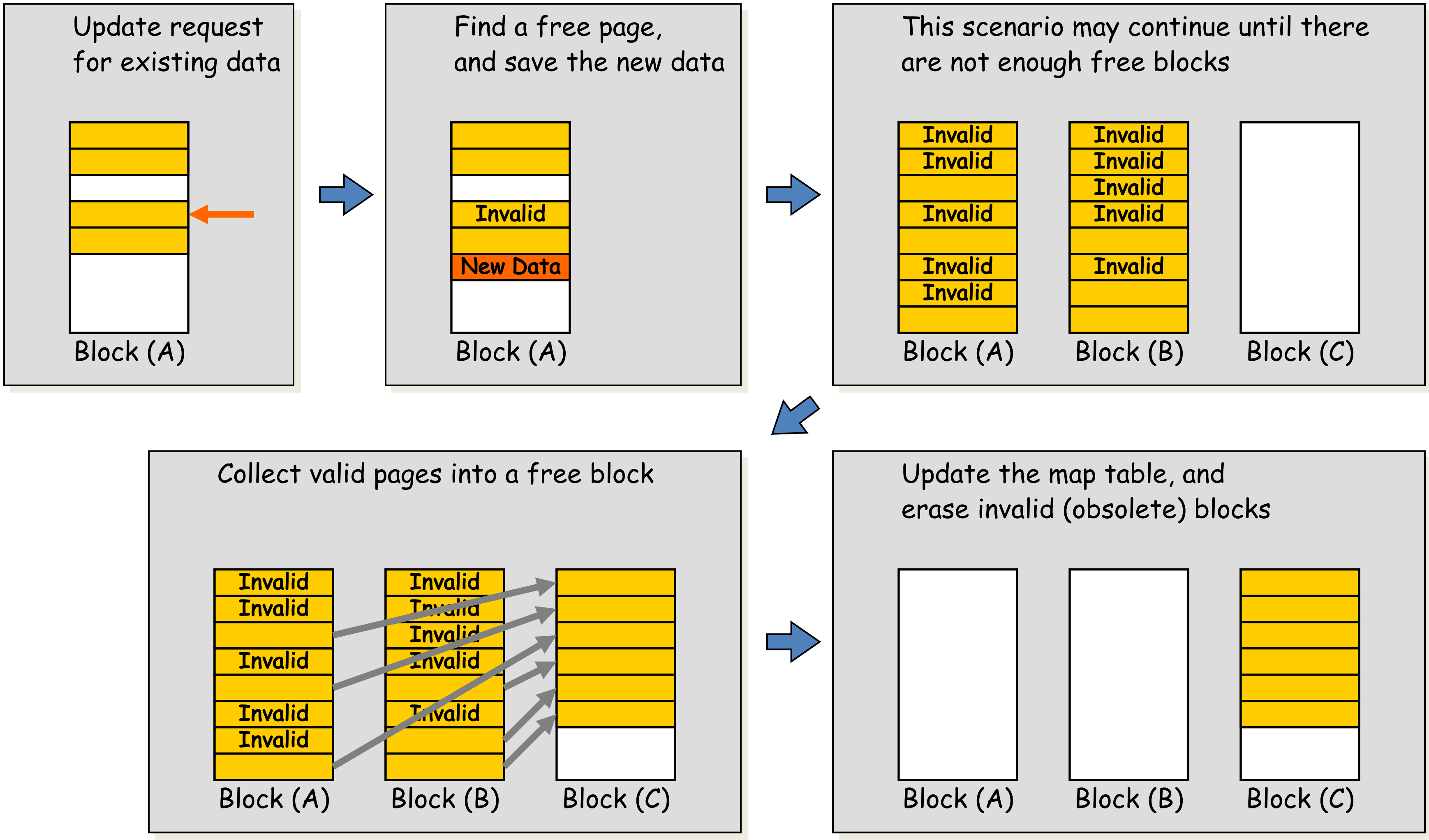
Address Translation



Garbage collection

- the process of finding garbage blocks and reclaiming them.
- Basic process
  - Find a block that contains one or more garbage pages.
  - Read in the live(non-garbage) pages from that block.
  - Write out those live pages to the next writable pages.
  - Reclaim the entire block for use in writing.

Garbage collection



Garbage collection

- Garbage collection is expensive
  - Require reading and rewriting of live data.
  - Ideal garbage collection is reclamation of a block that consists of only dead pages.

- Cost of Garbage Collection
  - Amount of data blocks that are migrated.
  - Overprovision the device by adding extra flash capacity→ Cleaning can be delayed.
  - Run the GC in the background.

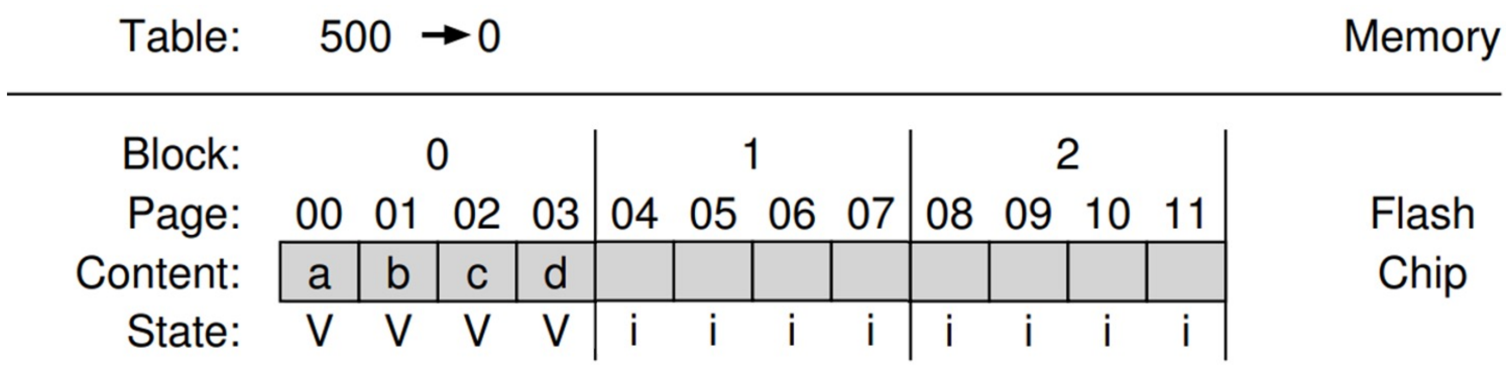
Mapping Table Size

- Size of page-level mapping table is too large
  - With a 1TB SSD with a 4byte entry per 4KB page, 1GB of DRAM is needed for mapping.
- Some approaches to reduce the costs of mapping
  - Block-based mapping
  - Hybrid mapping
  - Page mapping plus caching

Block Mapping

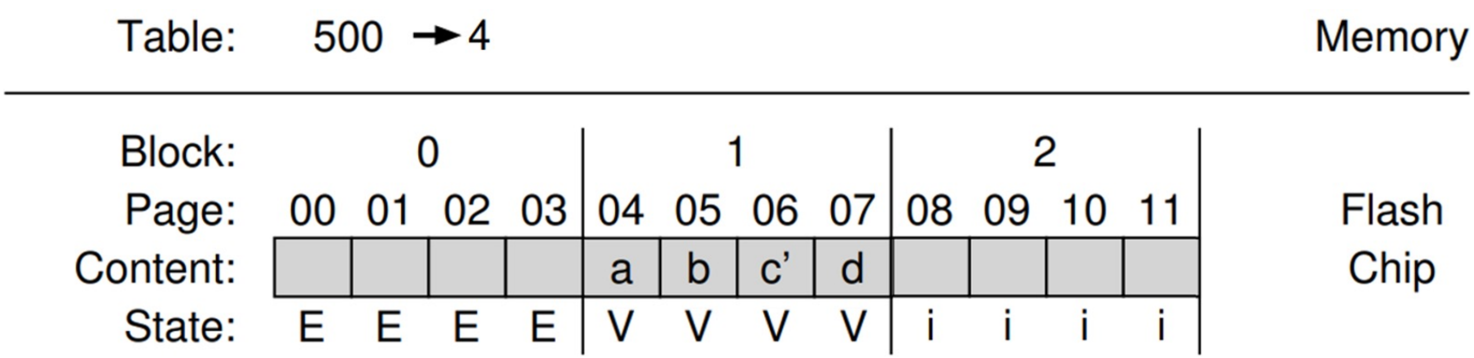
- Mapping at block granularity.
  - To reduce the size of a mapping table.
- Small write problem: When a small write occurs, the FTL must read a large amount of live data from the old block and copy them into a new one.

- Logical blocks 2000, 2001, 2002, and 2003 are at the same Flash block (500), and have different offsets (0, 1, 2, and 3).
- The FTL records that chunk 500 maps to block 0.



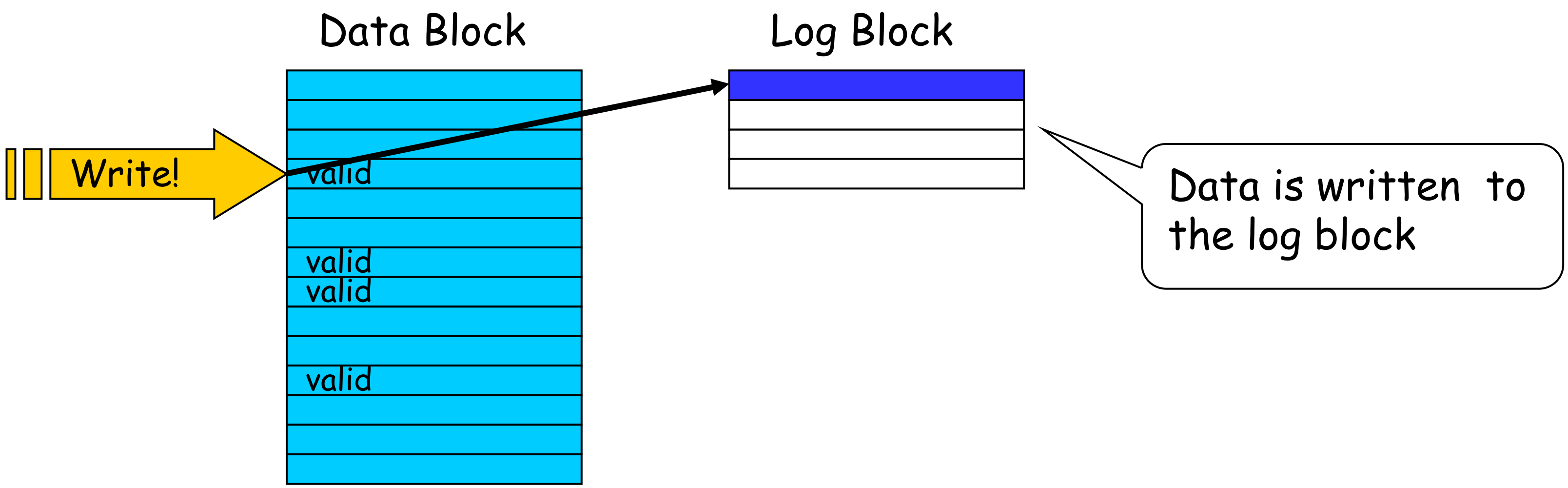
Example

- If the logical block 2002 is updated with contents c',
  - FTL must read in 2000, 2001, and 2003.
  - Write out all four logical blocks in a new location.
  - Update the mapping table.



Hybrid mapping

- In hybrid mapping, FTL maintains
  - Log blocks: page-mapped
  - Data blocks: block-mapped
- Write to log blocks first
- When looking for a particular logical block, the FTL will consult the page mapping table and block mapping table in order.



Example

- In the following situation,

Log Table:
Data Table: 250 → 8 — Memory

Block:	0				1				2				Flash Chip
Page:	00	01	02	03	04	05	06	07	08	09	10	11	
Content:									a	b	c	d	
State:	i	i	i	i	i	i	i	i	V	V	V	V	

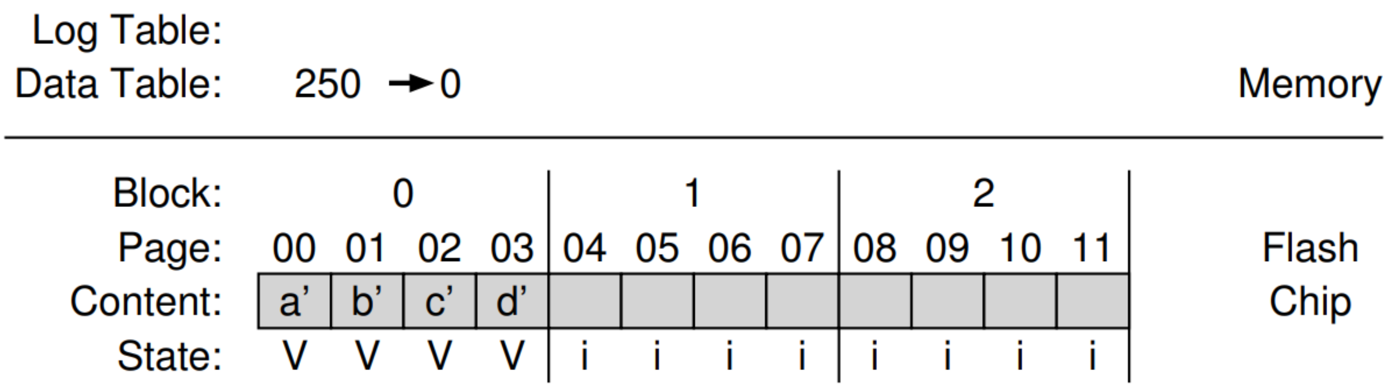
- Update these pages (with data a', b', c', and d'). → write them to the log block. FTL updates the page mapping information.

Log Table: 1000 → 0 1001 → 1 1002 → 2 1003 → 3
Data Table: 250 → 8 — Memory

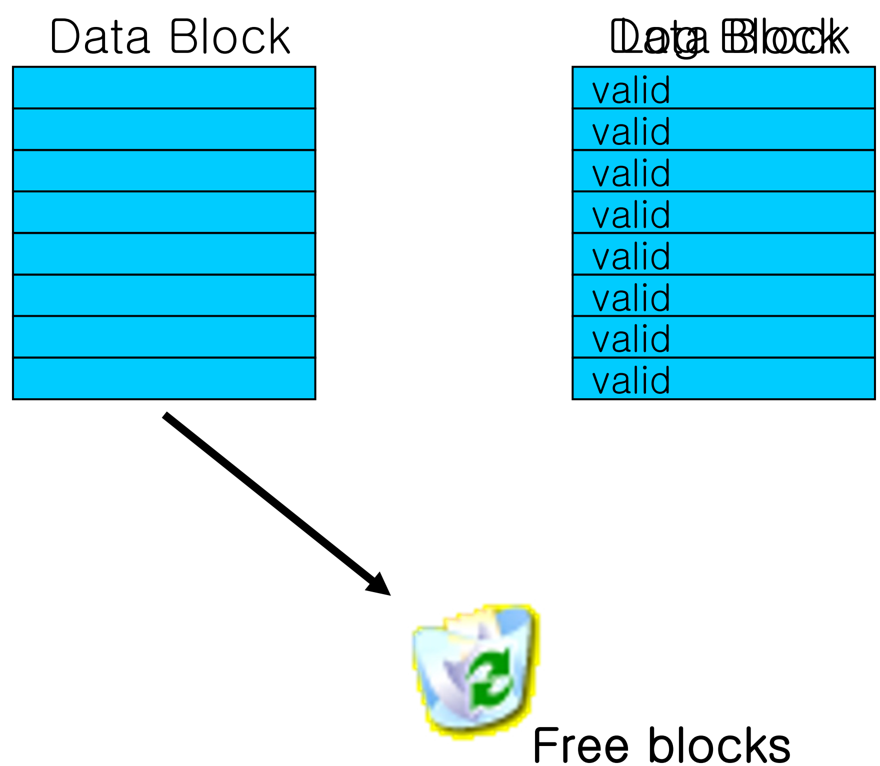
Block:	0				1				2				Flash Chip
Page:	00	01	02	03	04	05	06	07	08	09	10	11	
Content:	a'	b'	c'	d'					a	b	c	d	
State:	V	V	V	V	i	i	i	i	V	V	V	V	

Switch Merge

- When log block is full, perform merge.
  - Switch merge



Switch Merge



Partial Merge

- Client overwrites logical block 1000 and 1001 partially. What happened?

Log Table:
Data Table: 250 → 8

Memory

Block:	0				1				2				
Page:	00	01	02	03	04	05	06	07	08	09	10	11	Flash
Content:									a	b	c	d	Chip
State:	i	i	i	i	i	i	i	i	V	V	V	V	

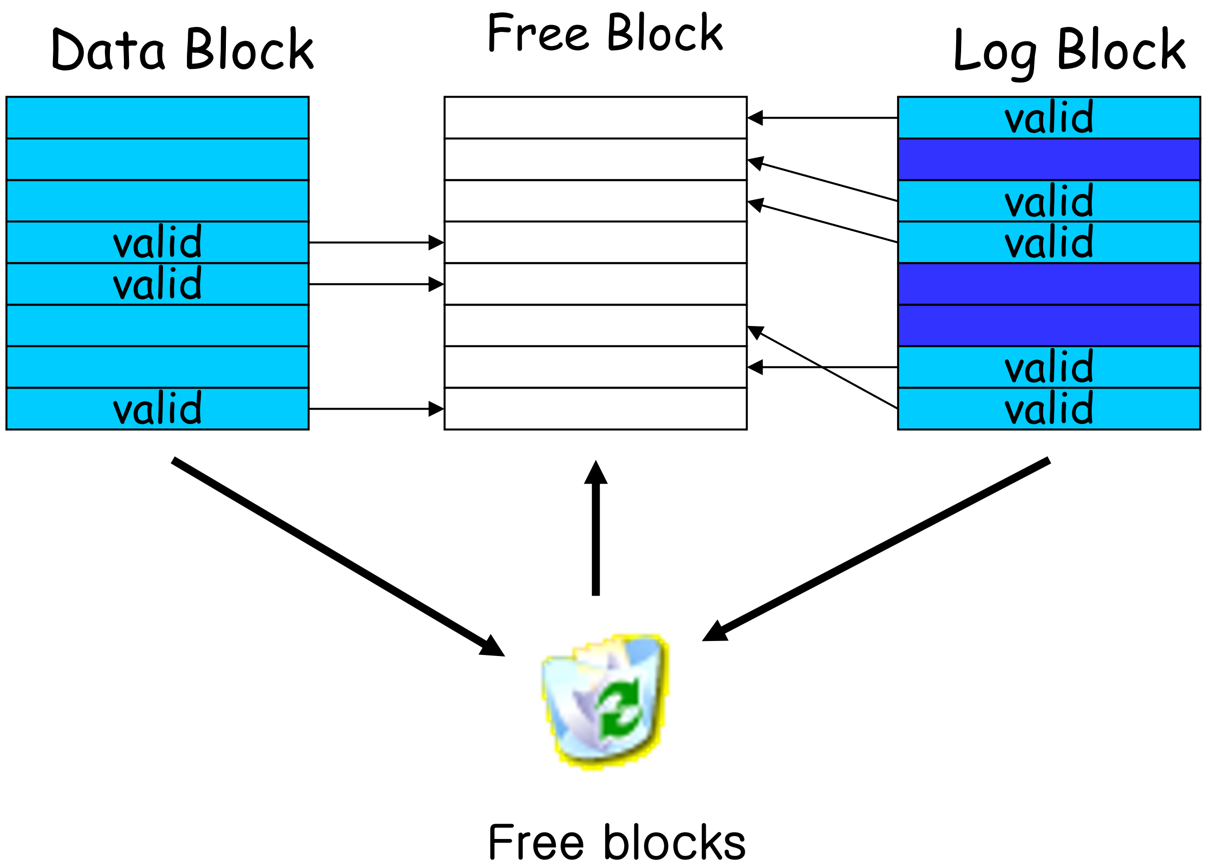
- The FTL writes logical block 1000 and 1001(contents a' and b') to the available pages in new block.
- And then, the FTL appends other live data.

Log Table: 1000 → 0 1001 → 1
Data Table: 250 → 8

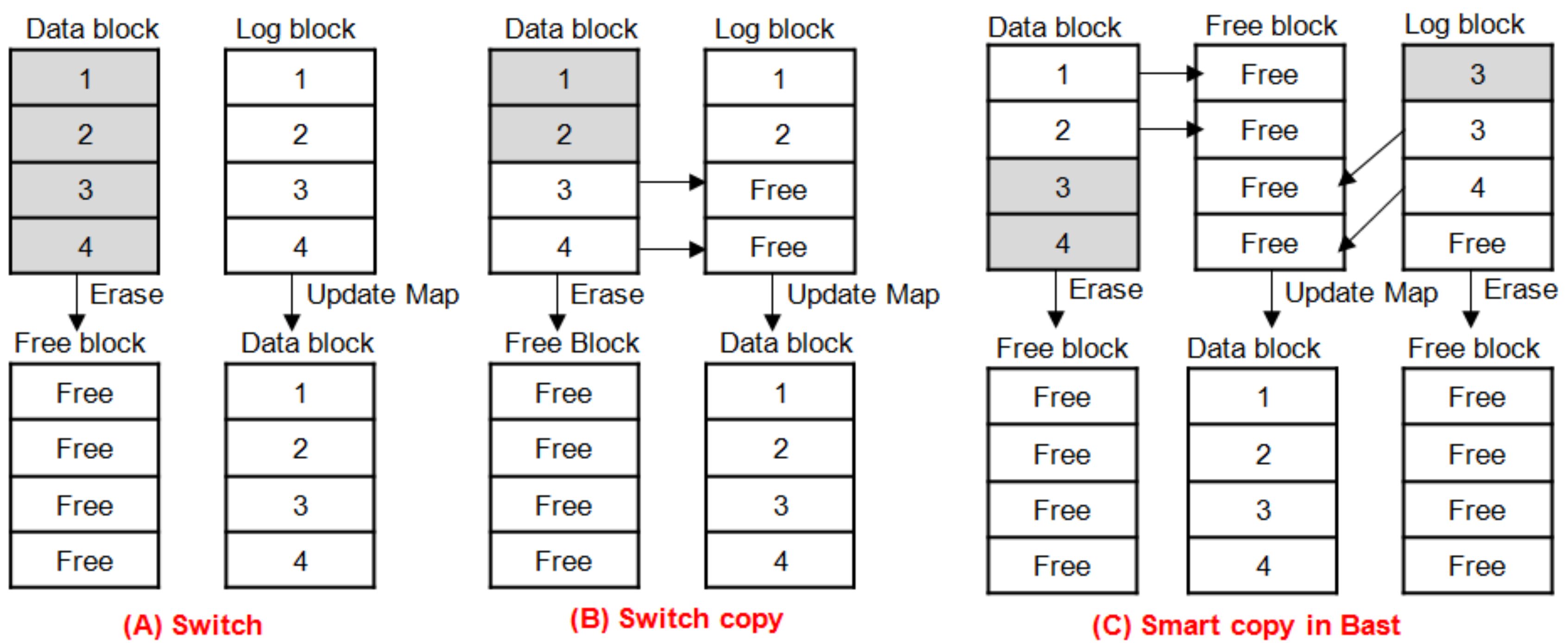
Memory

Block:	0				1				2				
Page:	00	01	02	03	04	05	06	07	08	09	10	11	Flash
Content:	a'	b'							a	b	c	d	Chip
State:	V	V	i	i	i	i	i	i	V	V	V	V	

Partial Merge



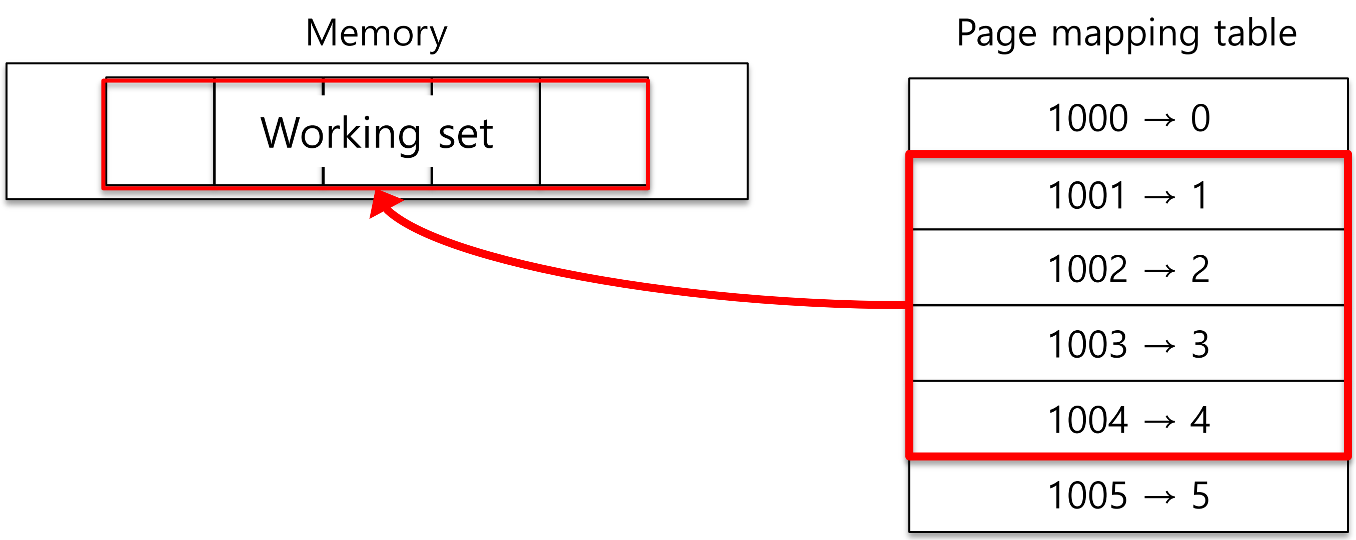
Switch and Merge



Page mapping plus caching

- Caching only the active part of the page-mapped FTL in memory.
  - If a given workload only accesses a small set of pages, the translations of those pages will be stored in the in-memory FTL.

- Performance will be excellent without high memory cost.
- Cache miss overhead exists.

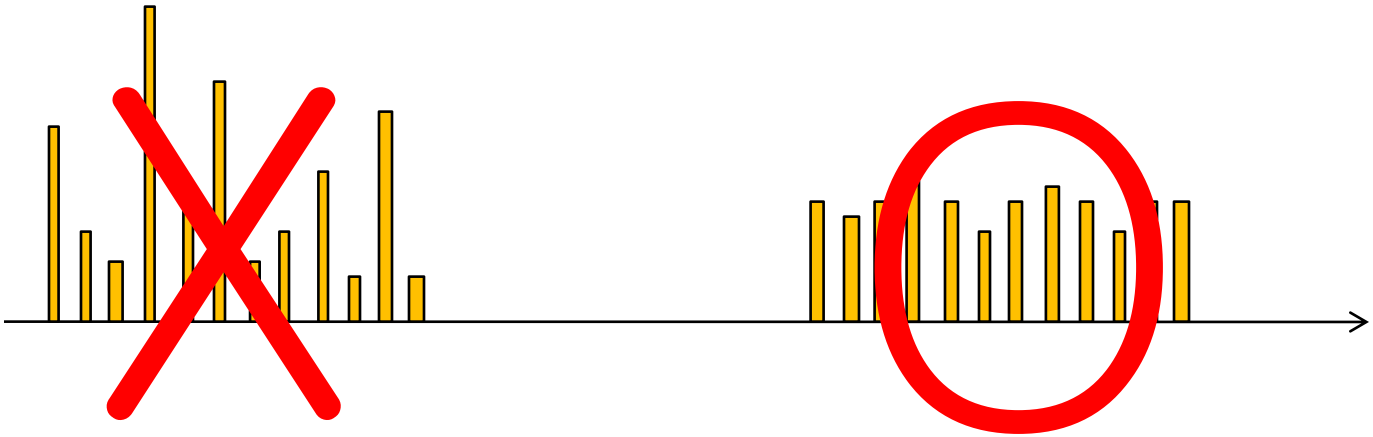


Wear Leveling

- Program/Erase cycle is limited in Flash memory.
- If P/E cycle is skewed, that shortens the lifespan of the entire Flash storage.

All blocks should wear out at roughly the same time.

- A block may consist of cold data.
  - The FTL must periodically read all the live data out of such blocks and re-write it elsewhere.
  - Wear leveling increases the write amplification of the SSD and decreases performance.
- Sample Policy: Each Flash Block has a P/E cycle counter.
  - Maintain |Max(PE cycle) – Min(PE cycle)| < e
  - Maintain |Max(PE cycle) – Min(PE cycle)|/Max(PE cycle) < e



Summary

- Flash-based SSDs are a common presence in ...
  - laptops, desktops, and servers inside the datacenters.

- Flash Translation Layer
  - Address Translation
  - Wear Leveling
  - Garbage Collection