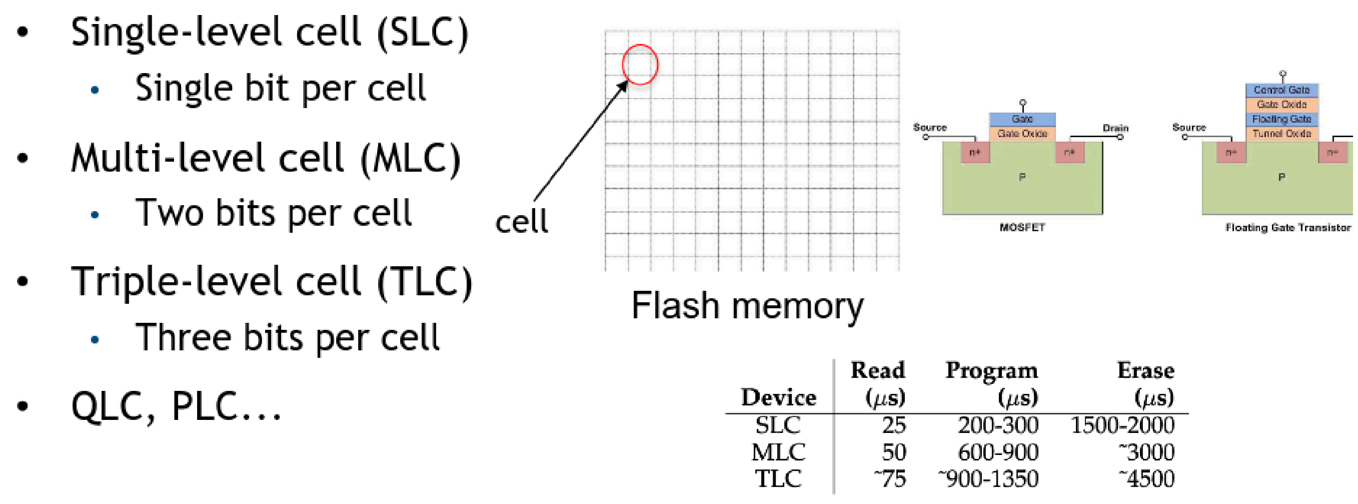
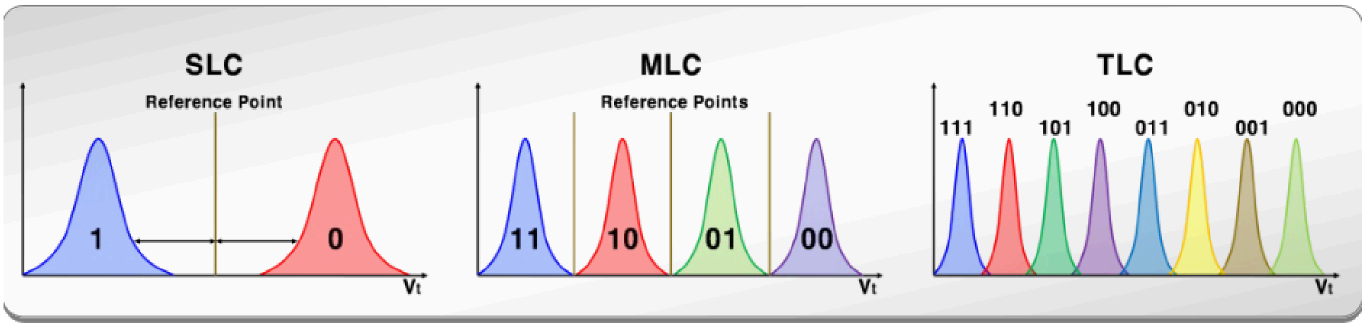
SSD - Solid State Disk

Based on flash memory, in the last 10 years we have SSD disks because of the cost(much lower than before). They are electronic, build up of transistors and in case of power loss you don't loss the power state of the disk(no data loss). The system is limited by the speed of the controller. SSDs cost 10% more than HDD and this is why data center use hybrid solutions or use only HDD.



- Single-level cell (SLC)
  - Single bit per cell
- Multi-level cell (MLC)
  - Two bits per cell
- Triple-level cell (TLC)
  - Three bits per cell
- QLC, PLC…

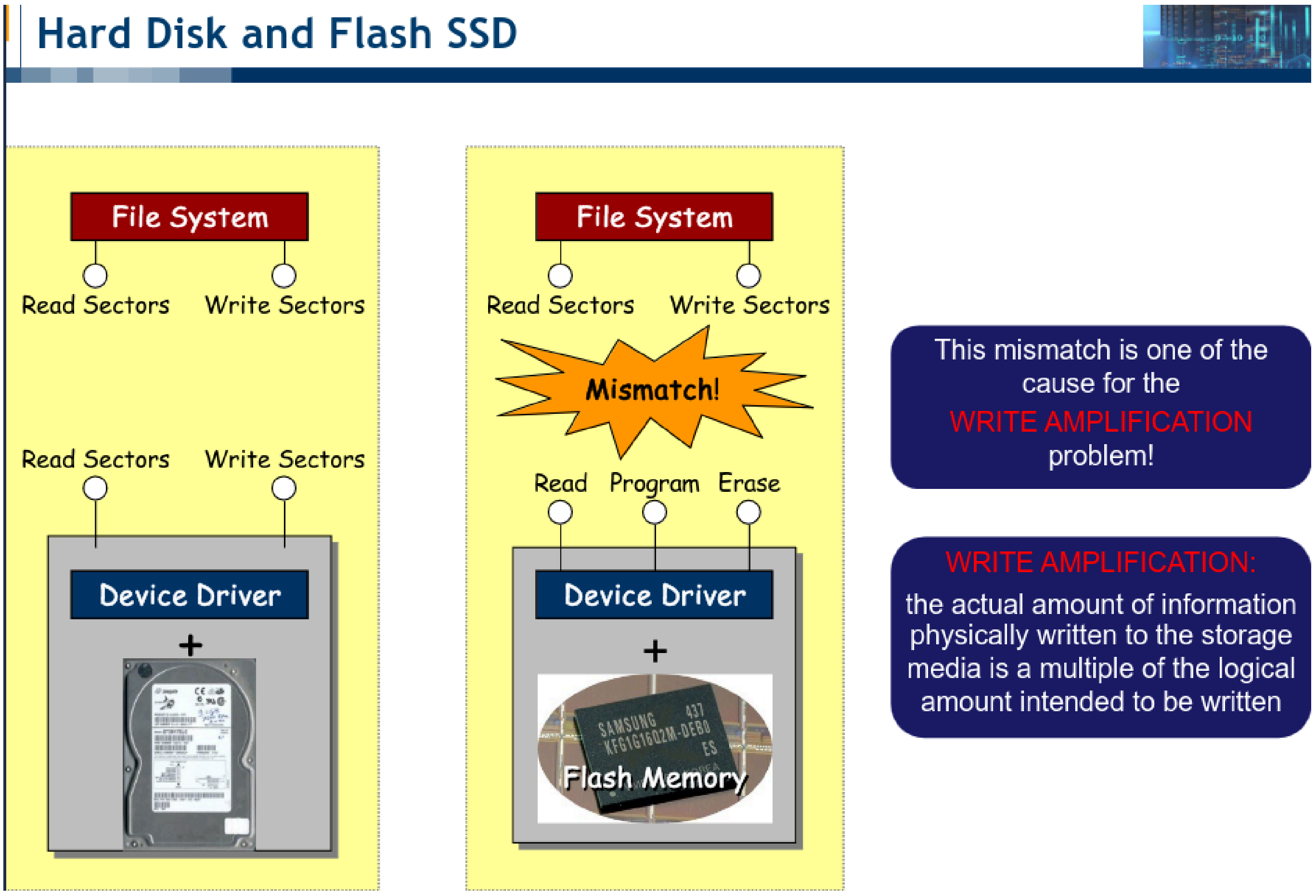
Device	Read (µs)	Program (µs)	Erase (µs)
SLC	25	200-300	1500-2000
MLC	50	600-900	~3000
TLC	~75	~900-1350	~4500



Increasing the layers helps in saving money as you are having more cells in the same space but you are also increasing the time for Reads and you are increasing the reading error. Writes are 10 times slower than reads and Erase are slower than Writes.
They are realised as follows:

- NAND flash is organized into Pages and Blocks
- You can read and write a individuals pages, but you can erase pages at the block level only
- A page contains multiple logical block (e.g. 512B-4KB) addresses (LBAs)
- A block typically consists of multiple pages (e.g. 64) with a total capacity of around 128-256KB
- Block/Page terminology in the SSD context can clash with previous use
- Blocks (or Erase Block): smallest unit that can be erased
  - It consists of multiple pages and can be cleaned using the ERASE command

- Pages: smallest unit that can be read/written.
  - It is a sub-unit of an erase block and consists of the number of bytes which can be read/written in a single operations through the READ or PROGRAM commands. They can be in three states: Empty(or Erased, they don't contain data), Dirty(or Invalid, they contain data but this data is no longer in use(or never used)), In Use(or Valid, the page contains data that can be actually read).
  Only empty pages can be written. Only dirty pages can be erased but this must be done at the block level(all the pages in the block must be dirty or empty). It is meaningful to read only pages in the "in use" state. This derives in huge constraints to the use of the SSD.



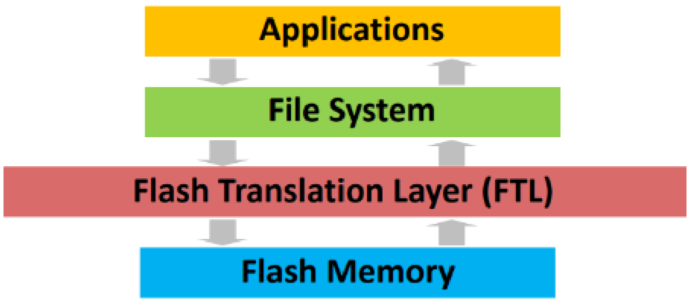
Hard Disk and Flash SSD

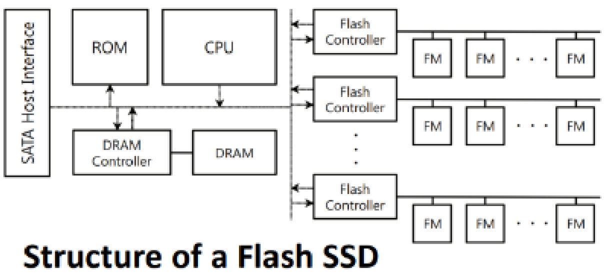
This mismatch is one of the cause for the WRITE AMPLIFICATION problem!

WRITE AMPLIFICATION: the actual amount of information physically written to the storage media is a multiple of the logical amount intended to be written

Flash cells can wear out due to the erasing of the cells. The silicon is degraded every time electricity flows in it. We don't want to reach the wear out but we want to reach this wear out state equally on all the cells in the system. We can use some empty space to replace weared out cells. The Flash Translation Layer is the component that is relegated to the job of mapping the cells and blocks and the wear state. With a static mapping you are always writing and erasing the same block in various cycle reaching the wear out of the cell faster. The FTL has to perform the data allocation, reduce the Write Amplification problem. Reads and Writes can flip the state of near cells. So you try to write the blocks in order.
When an existing page is updated old data becomes obsolete! Old version of data are called garbage and (sooner or later) garbage pages must be reclaimed for new writes to take place. Garbage Collection is the process of finding garbage blocks and reclaiming them. 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.

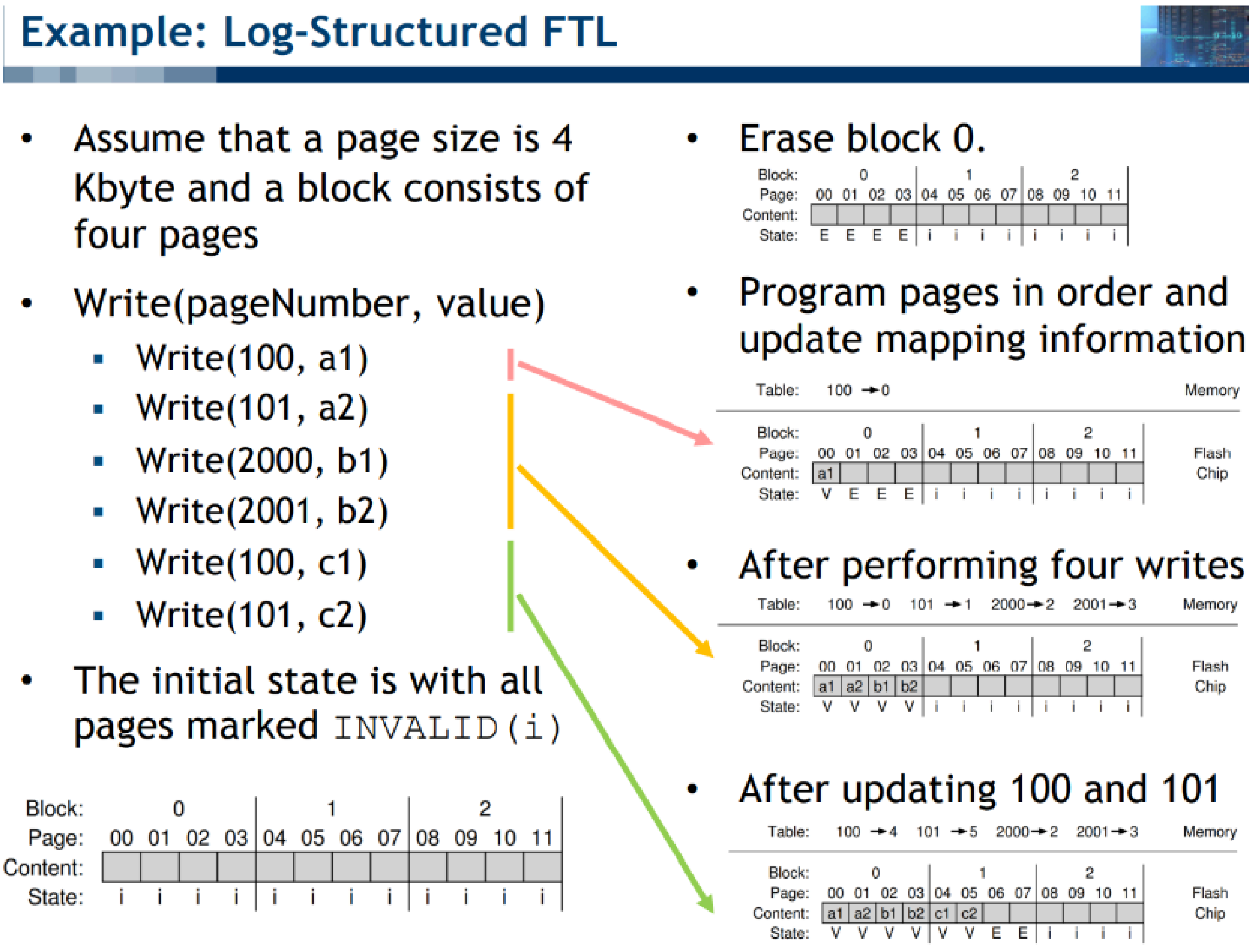
Flash Translation Layer





Structure of a Flash SSD

Direct mapping between Logical to Physical pages is not feasible
FTL is an SSD component that make the SSD “look as HDD”
We use the approach called as Log-Structured FTL, we try to limit the write amplification and try to program the pages in order to diminish the possibility of flipping bits in the neighboring pages.



Example: Log-Structured FTL

- Assume that a page size is 4 Kbyte and a block consists of four pages
- Write(pageNumber, value)
  - Write(100, a1)
  - Write(101, a2)
  - Write(2000, b1)
  - Write(2001, b2)
  - Write(100, c1)
  - Write(101, c2)
- The initial state is 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

- 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

Table: 100 →0 Memory

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

- 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	Flash Chip
Content:	a1	a2	b1	b2									
State:	V	V	V	V	i	i	i	i	i	i	i	i	

- After updating 100 and 101

Table: 100 →4 101 →5 2000→2 2001→3 Memory

Block:	0				1				2				
Page:	00	01	02	03	04	05	06	07	08	09	10	11	Flash Chip
Content:	a1	a2	b1	b2	c1	c2							
State:	V	V	V	V	V	V	E	E	i	i	i	i	

In this case we are maintaining the same structure but we are using a garbage collector to clean the location where there is invalid data. This situation is time consuming so we try to have the minimal delay from the garbage collecting task.

Mapping Table Size

The 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.

Wear Leveling

Erase/Write cycle is limited in Flash memory. Skewness in the EW cycles shortens the life of the SSD. All blocks should wear out at roughly the same time. Log-Structured approach and garbage collection helps in spreading writes. However, 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. Simple Policy: Each Flash Block has EW cycle counter so
$Maintain = |Max(EWcycle) – Min(EWcycle)| < e$