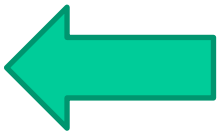




Storage: SSD, a new hope

The topics of the course: what are we going to see today?

A. **HW Infrastructures**:
   - **System-level**: Computing Infrastructures and Data Center Architectures, Rack/Structure;
   - **Node-level**: Server (computation, HW accelerators), **Storage (Type, technology)**, Networking (architecture and technology);
   - **Building-level**: Cooling systems, power supply, failure recovery



B. **SW Infrastructures**:
   - **Virtualization**: Process/System VM, Virtualization Mechanisms (Hypervisor, Para/Full virtualization)
   - **Computing Architectures**: Cloud Computing (types, characteristics), Edge/Fog Computing, X-as-a service
   - **Machine and deep learning-as-a-service**

C. **Methods**:
   - **Reliability and availability of datacenters** (definition, fundamental laws, RBDs)
   - **Disk performance** (Type, Performance, RAID)
   - **Scalability and performance of datacenters** (definitions, fundamental laws, queuing network theory)

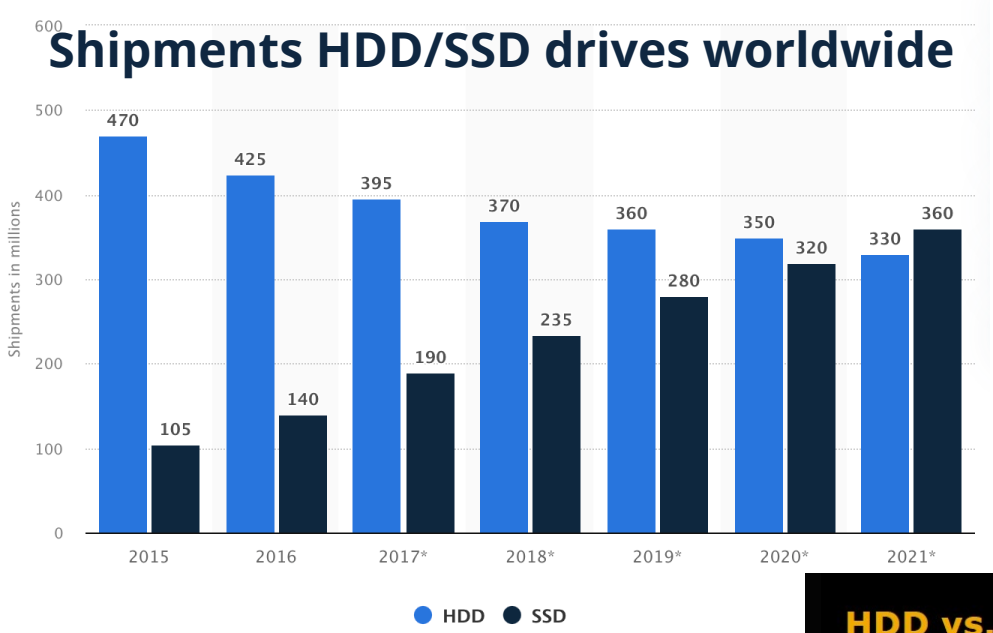
SSD: Overview

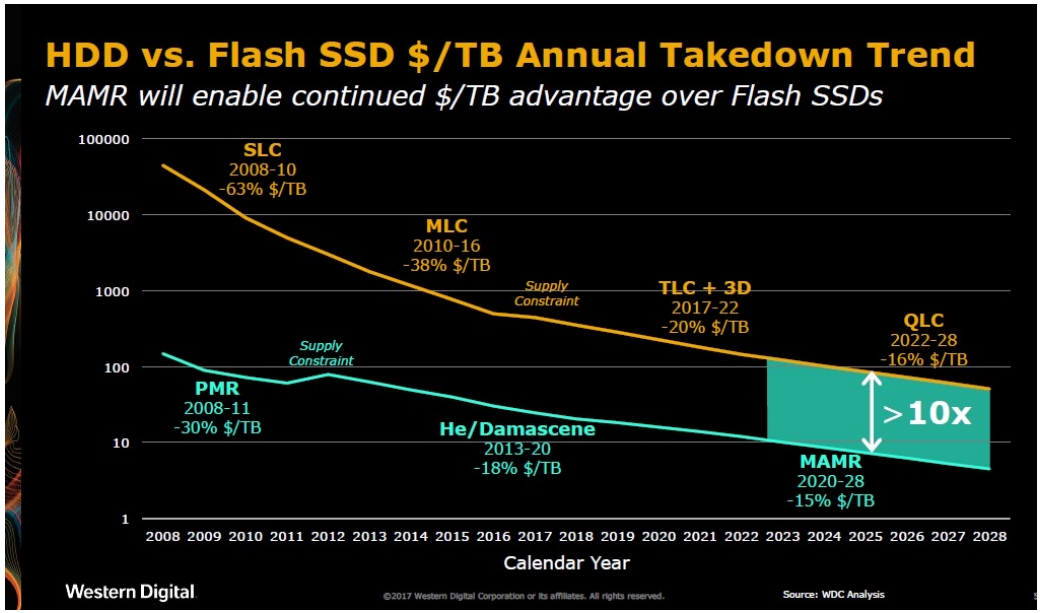
Solid-state Drive (SSD)

- No mechanical or moving parts like HDD
- Built out of transistors (like memory and processors)
- Retain information despite power loss unlike typical RAM
- A controller is included in the device with one or more solid state memory components
- It uses traditional hard disk drive (HDD) interfaces (protocol and physical connectors) and form factors
- Higher performance than HDD

Platters are replaced by NAND-based memories

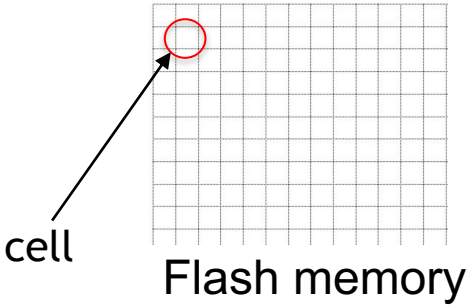
HDD vs SSD Major trends



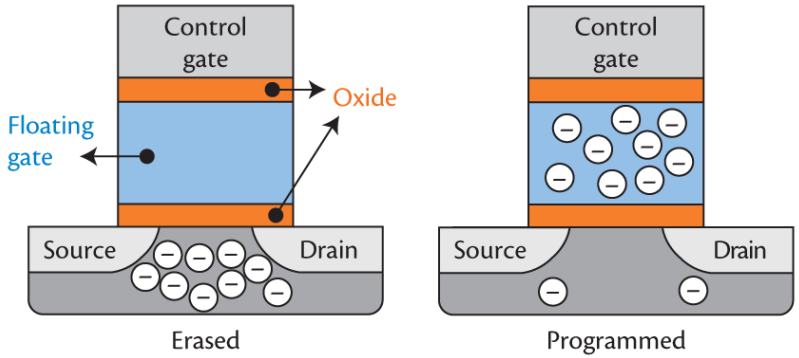


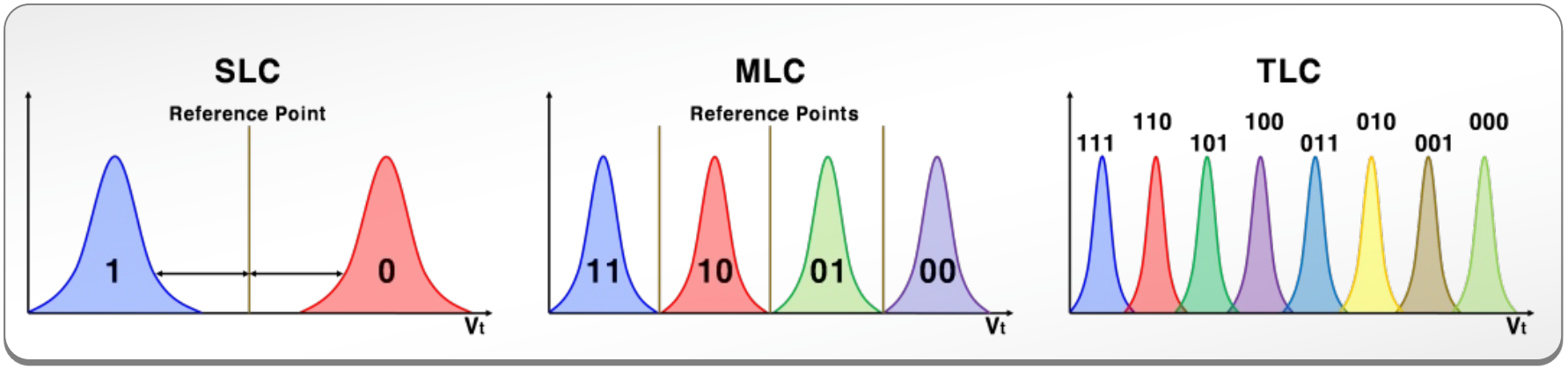
SSD: Storing Bit

- 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

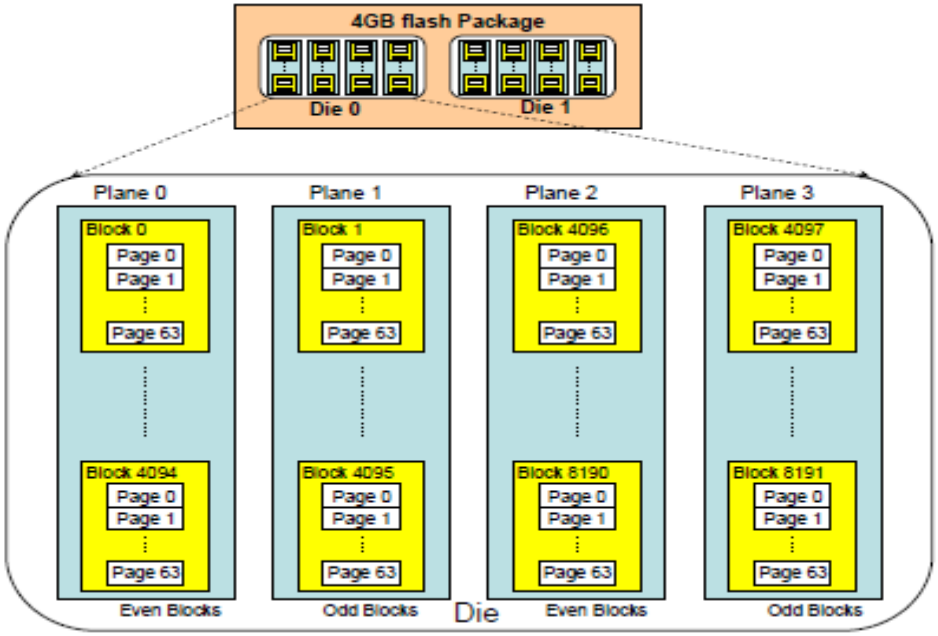




SSD: Internal organization (1)

NAND flash is organized into **Pages** and **Blocks**

- 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



SSD: Internal organization (2)

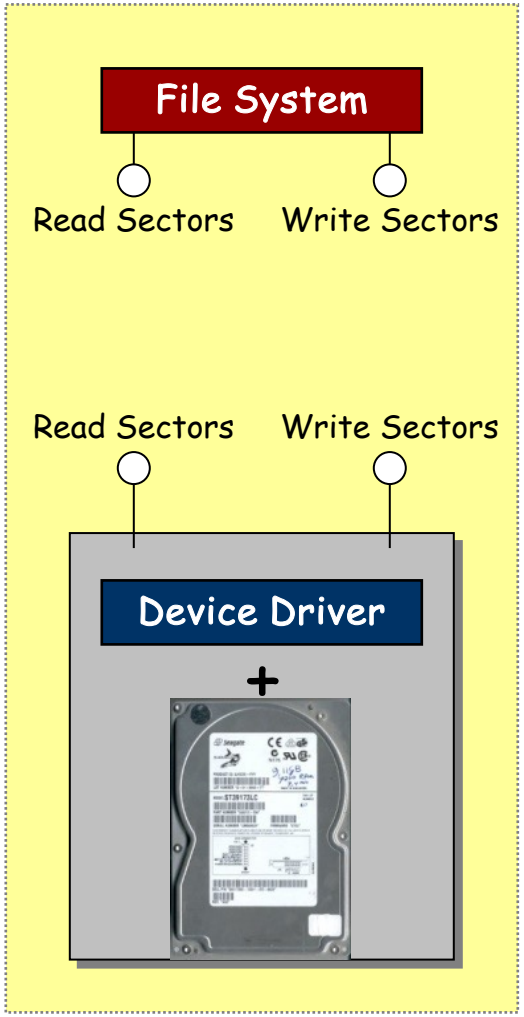
- **Blocks** (or Erase Block): smallest unit that can be erased
  - It consists of multiple pages and can be cleaned using the **ERASE** command
- **Pages**: smallest unit that can be read/written.
  - It is a sub-unit of an erase block and consists of the number of bytes which can be read/written in a single operations through the ***READ*** or ***PROGRAM*** commands
- Pages can be in *three states*:
  - Dirty (or *INVALID*):
    - they contain data, but this data is no longer in use (or never used)
  - Empty (or *EREASED*):
    - they do not contain data
  - In use (or *VALID*):
    - The page contains data that can be actually read

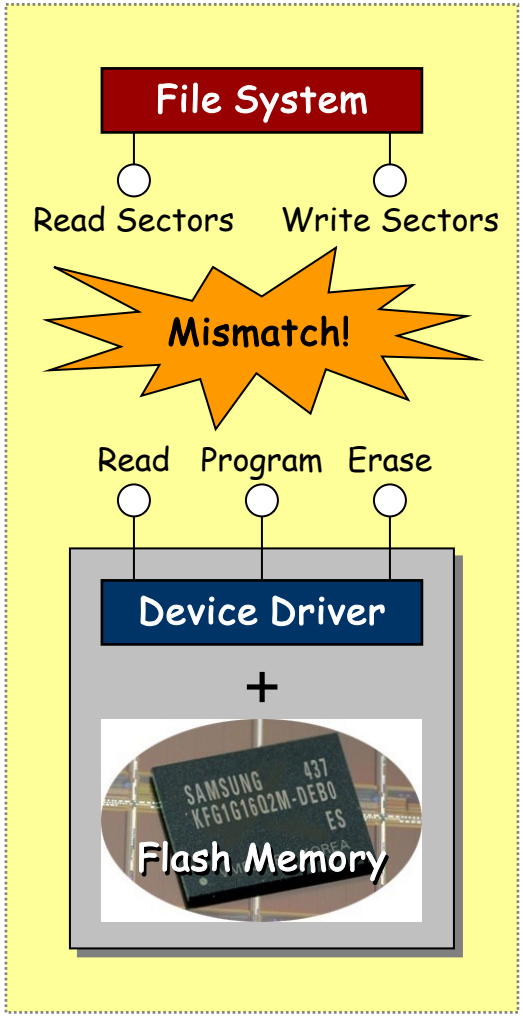
Page states – writing and erasing

- **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" ("valid") state**
- **If no empty page exists, some dirty page must be erased**
  - If no block containing just dirty or empty pages exists, then special procedures should be followed to gather empty pages over the disk
  - To erase the value in flash memory the original voltage must be reset to neutral before a new voltage can be applied, known as write amplification

Remark: we can write and read a single page of data from a SSD but we have to delete an entire block to release it

Hard Disk and Flash SSD

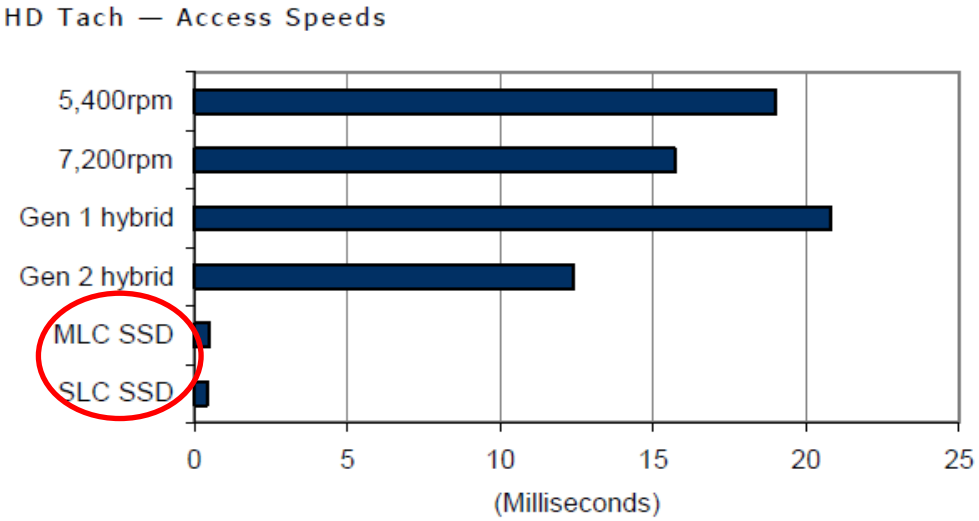




This mismatch is one of the cause for the WRITE AMPLIFICATION problem!

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

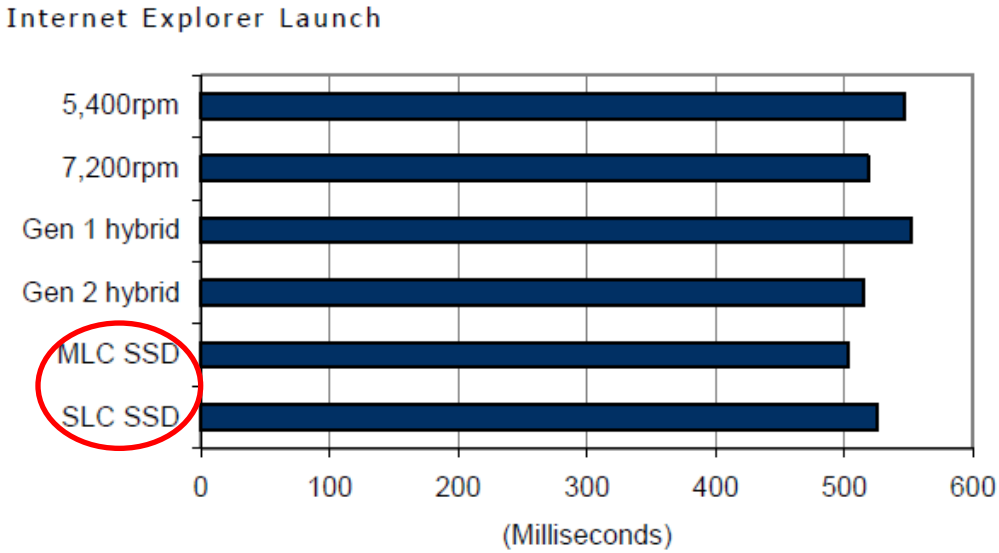
SSD: Performance: from theory...to practice



Access speed
measured at device level in a controlled laboratory environment
HDD vs SSD
~ 15-20x improvement

Same device integrated into a notebook PC with a real application test
~ same performace !!!

Why such behavior ?

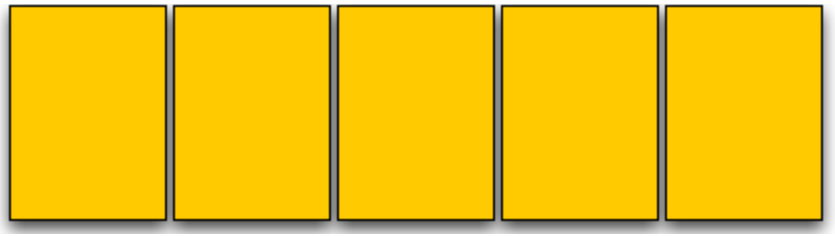


Source: Exposing the Strengths and Weaknesses of HDDs, SSDs, IDC and Hybrids, IDC, 2008

Example:

- Hypothetical SSD:

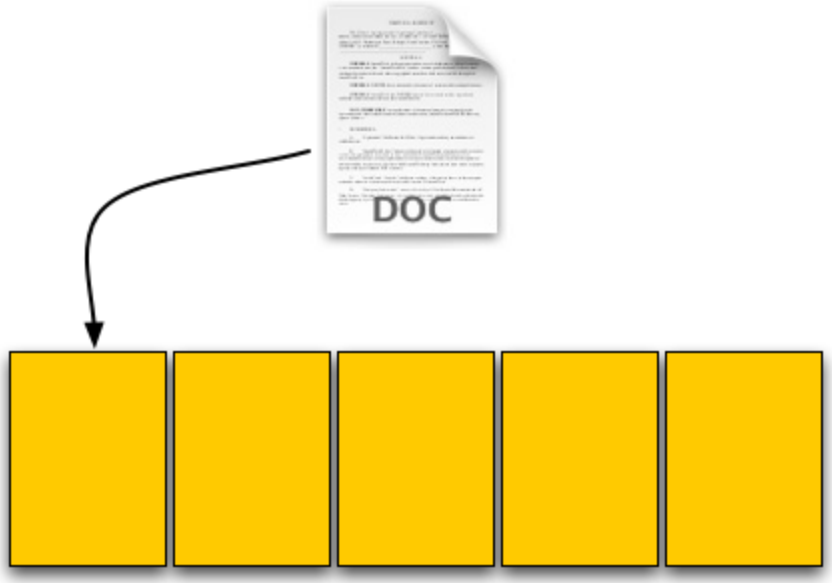
Page Size:	4KB
Block Size:	5 Pages
Drive Size:	1 Block
Read Speed:	2 KB/s
Write Speed:	1 KB/s



Example:

- Let's write a 4KB text file to the brand new SSD
- Overall Writing Time:
  - 4 seconds!

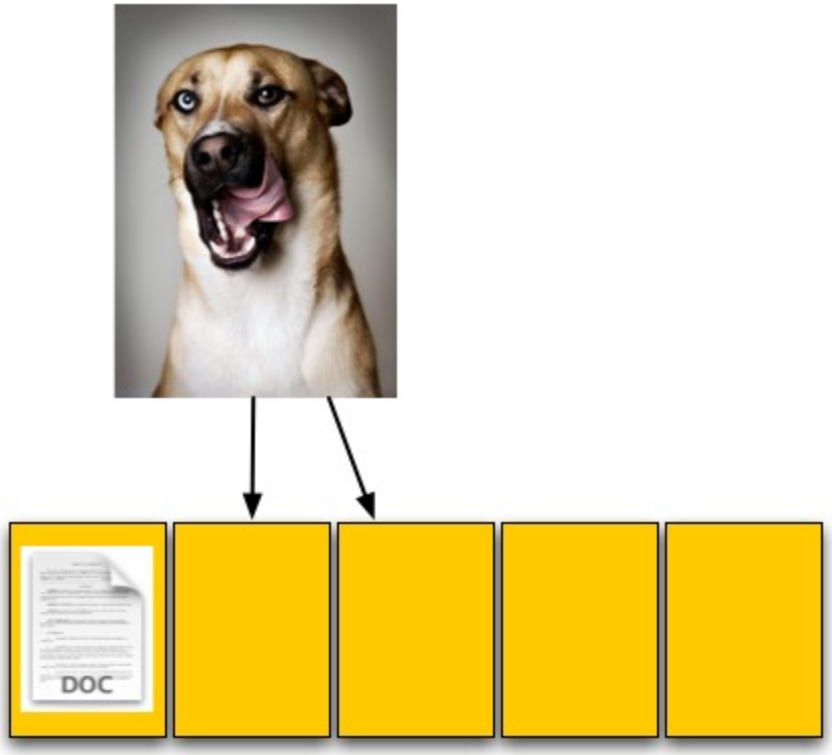
- Hypothetical SSD:
  - Page Size: 4KB
  - Block Size: 5 Pages
  - Drive Size: 1 Block
  - Read Speed: 2 KB/s
  - Write Speed: 1 KB/s



Example:

- Now let’s write a 8KB pic file to the almost brand new SSD, thankfully there's space
- Overall Writing Time:
  - 8 seconds!

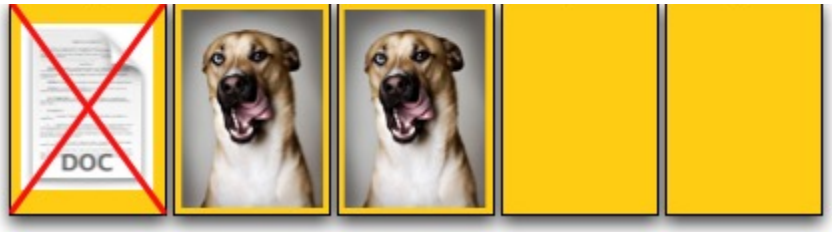
- Hypothetical SSD:
  - Page Size: 4KB
  - Block Size: 5 Pages
  - Drive Size: 1 Block
  - Read Speed: 2 KB/s
  - Write Speed: 1 KB/s



Example Cont.

- Let's now consider that the txt file in the first page is not more needed

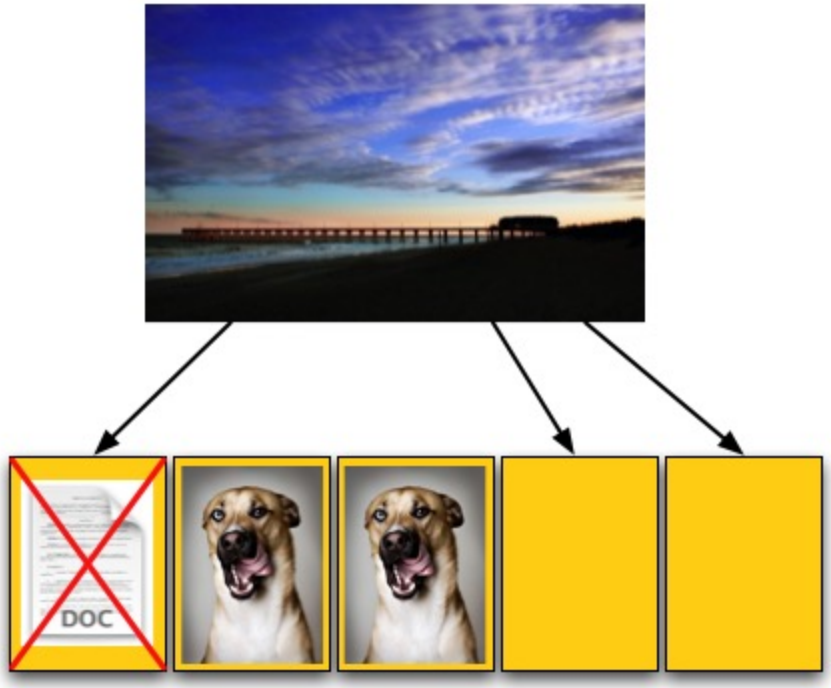
- Hypothetical SSD:
  - Page Size: 4KB
  - Block Size: 5 Pages
  - Drive Size: 1 Block
  - Read Speed: 2 KB/s
  - Write Speed: 1 KB/s



Example Cont.

- Finally, let's write a 12KB pic to the SSD
- How long should it take? 12s?

- Hypothetical SSD:
  - Page Size: 4KB
  - Block Size: 5 Pages
  - Drive Size: 1 Block
  - Read Speed: 2 KB/s
  - Write Speed: 1 KB/s

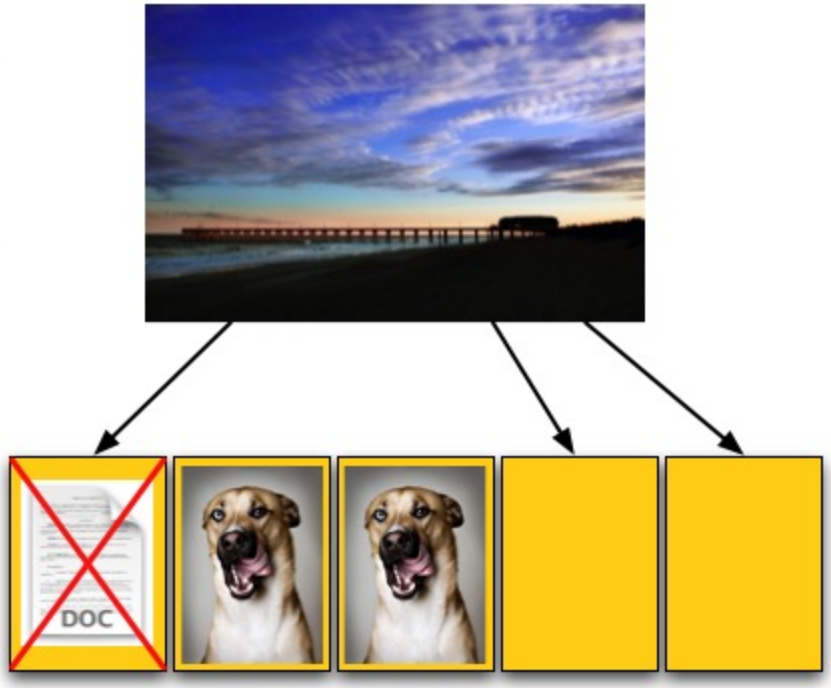


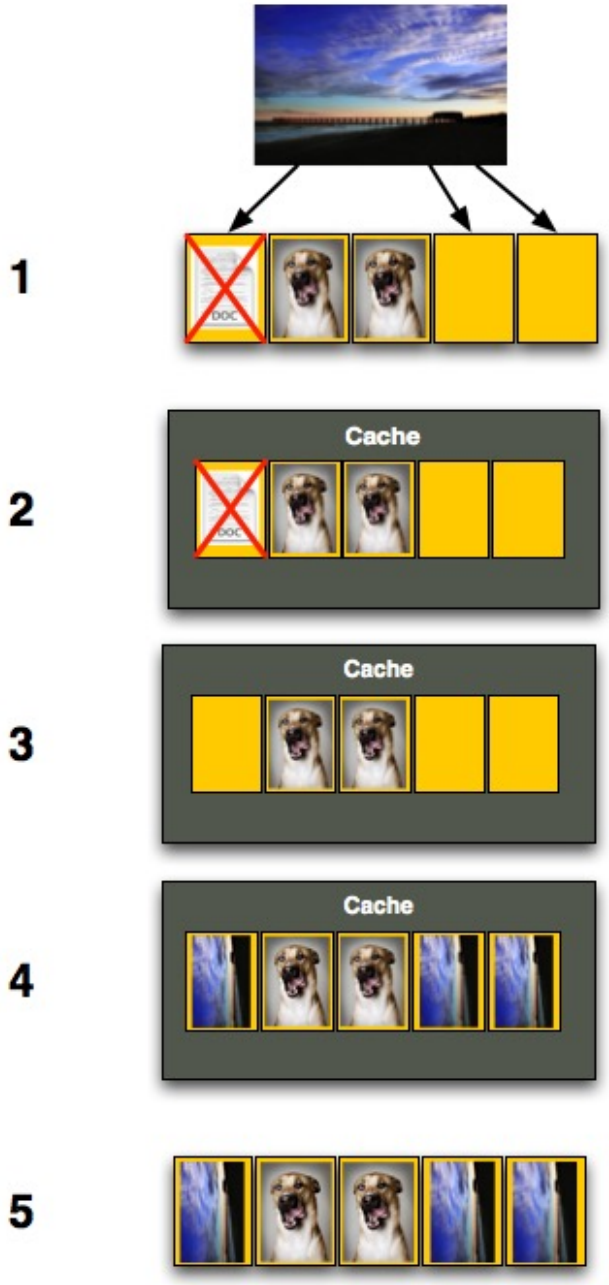
Example Cont.

- Finally, let's write a 12KB pic to the SSD
- How long should it take? 12s?
  - **NO!!!!! 24 SECONDS!!!!!**

- Hypothetical SSD:
  - Page Size: 4KB
  - Block Size: 5 Pages
  - Drive Size: 1 Block
  - Read Speed: 2 KB/s
  - Write Speed: 1 KB/s

The OS is told there are 3 free pages on the SSD when there are only 2 "empty".



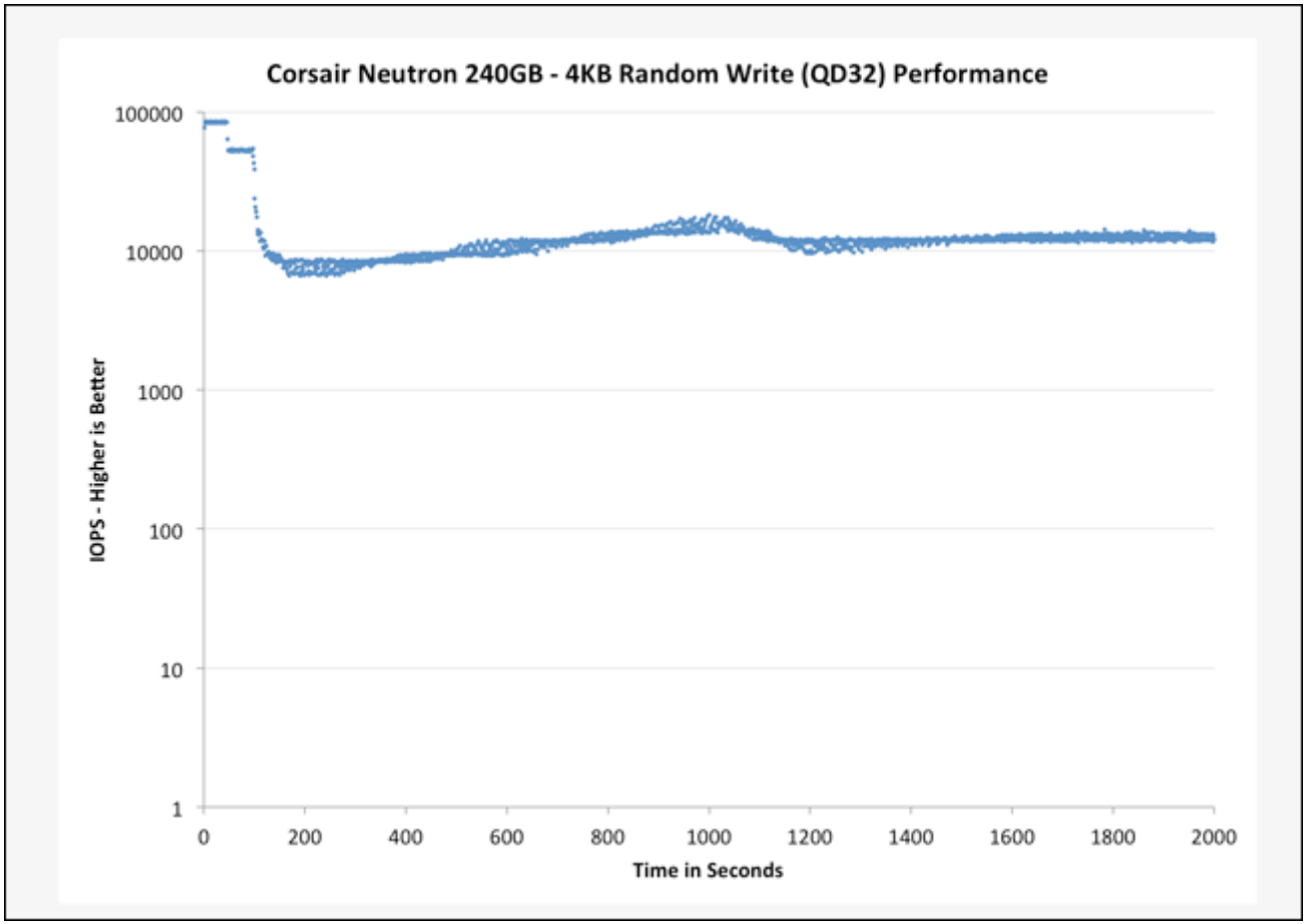


- Step 1: Read block into cache
- Step 2: Delete page from cache
- Step 3: Write new pic into cache
- Step 4: Delete the old block on SSD
- Step 5: Write cache to SSD

- The OS only thought it was writing 12 KBs of data when in fact the SSD had to read 8 KBs (2 KB/s) and then write 20KBs (1 KB/s), the entire block.
- The writing should have taken 12 secs but actually took 4 + 20 = 24 seconds, resulting in a write speed of 0.5KB/s not 1KB/s

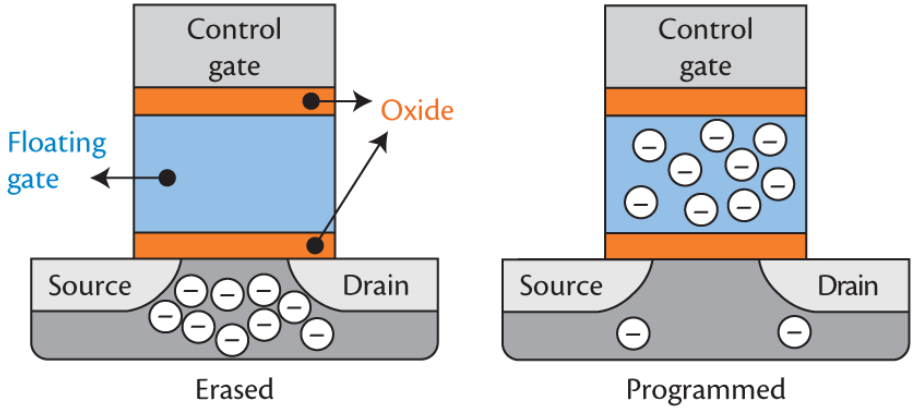
Performance degradation

Write amplification degrades a lot the performance of an SSD as time passes:



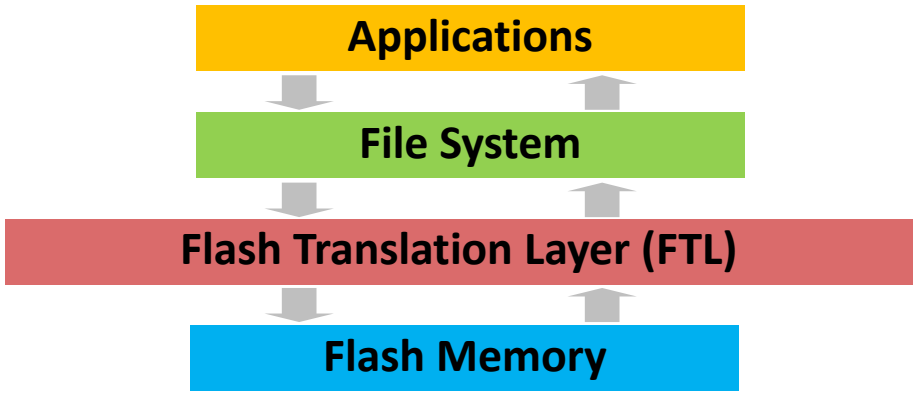
Flash cells wear out

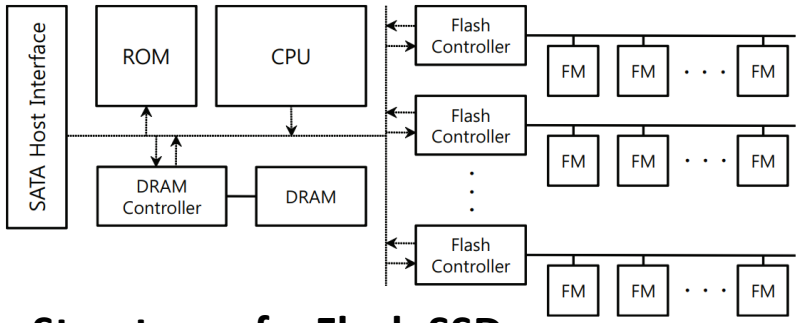
- Wear out: breakdown of the oxide layer within the floating-gate transistors of NAND flash memory
- The erasing process hits the flash cell with a relatively large charge of electrical energy
- Each time a block is erased:
  - the large electrical charge actually degrades the silicon material
  - after enough write-erase cycles, the electrical properties of the flash cell begin to break down and the cell becomes unreliable



Flash Translation Layer

- Direct mapping between Logical to Physical pages is not feasible
- FTL is an SSD component that make the SSD “look as HDD”
  - Data Allocation and Address translation
    - Efficient to reduce Write Amplification effects
    - Program pages within an erased block in order (from low to high pages)
      - **Log-Structured FTL**
  - Garbage collection
    - Reuse of pages with old data (Dirty/Invalid)
  - 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

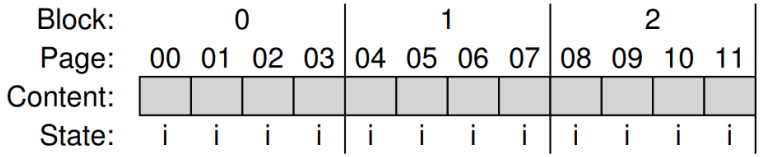




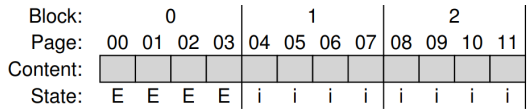
Structure of a Flash SSD

Example: Log-Structured FTL

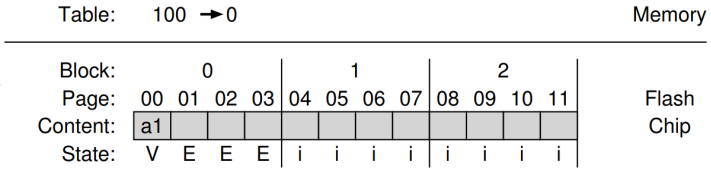
- Assume that a page size is 4 Kbyte and a block consists of four pages.
- Write(pageNumber, value)
  - Write(100, a1)
  - Write(101, a2)
  - Write(2000, b1)
  - Write(2001, b2)
  - Write(100, c1)
  - Write(101, c2)
- The initial state is with all pages marked `INVALID(i)`



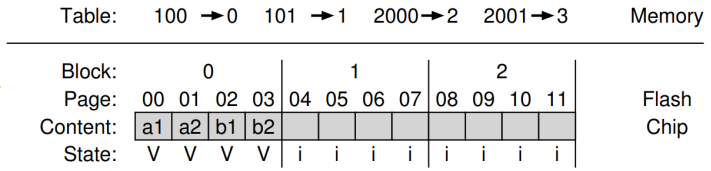
- Erase block 0.



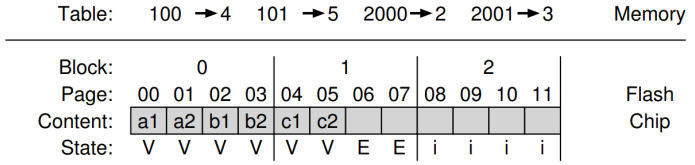
- Program pages in order and update mapping information.



- After performing four writes

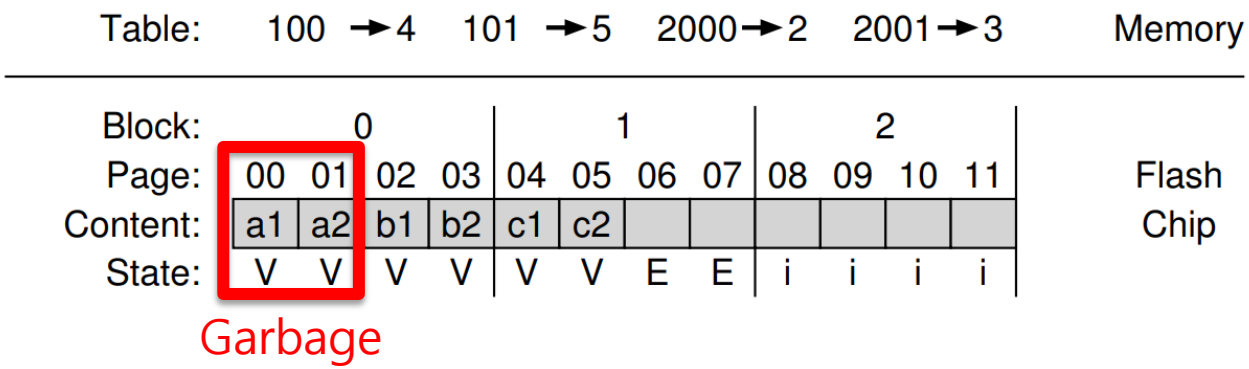


- After udating 100 and 101



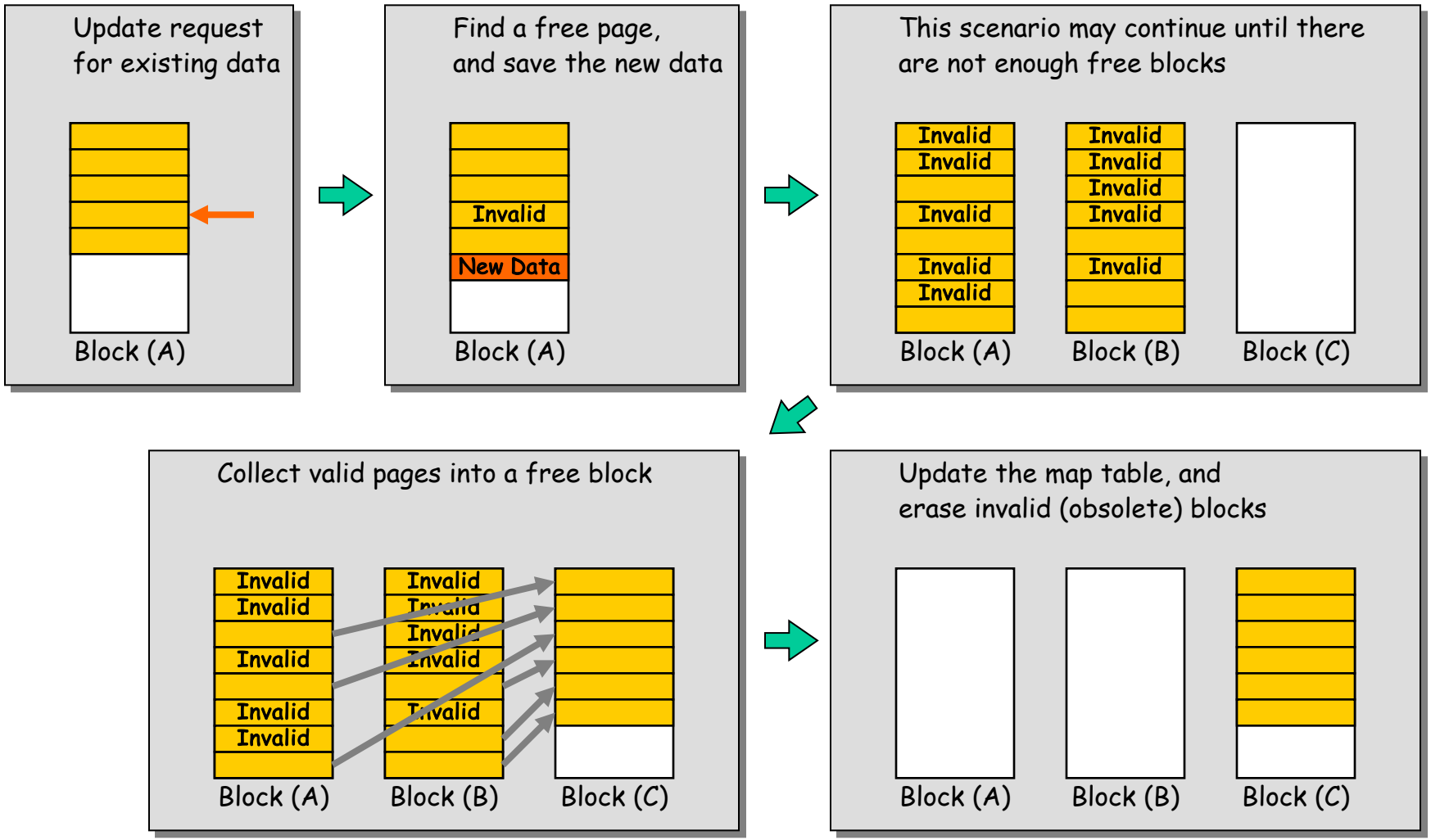
Garbage collection

- 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
  - Simple process for fully garbage blocks
  - More complex for partial cases. I.e. Basic steps:
    - Find a suitable partial block
    - Copy non-garbage pages
    - Reclaim (erase) the entire block for writing

Garbage collection (Example)

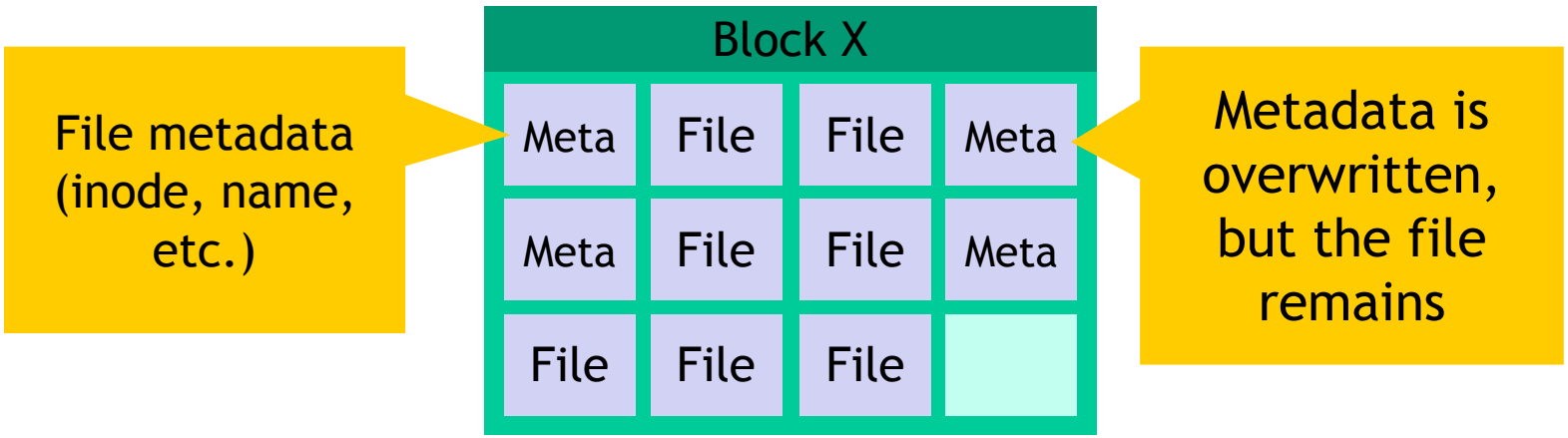


Garbage Collection Cost

- 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
- The cost of Garbage Collection depends on the amount of data blocks that have to be migrated
- Solutions to alleviate the problem:
  - Overprovision the device by adding extra flash capacity
    - Cleaning can be delayed
  - Run the Garbage Collection in the background using less busy periods for the disk

Garbage Collection: The Ambiguity of Delete

- When performing background GC the SSD assumes to know which pages are invalid
- Problem: most file systems don't actually delete data
  - E.g., on Linux, the "delete" function is unlink() and it removes the file meta-data, but not the file itself

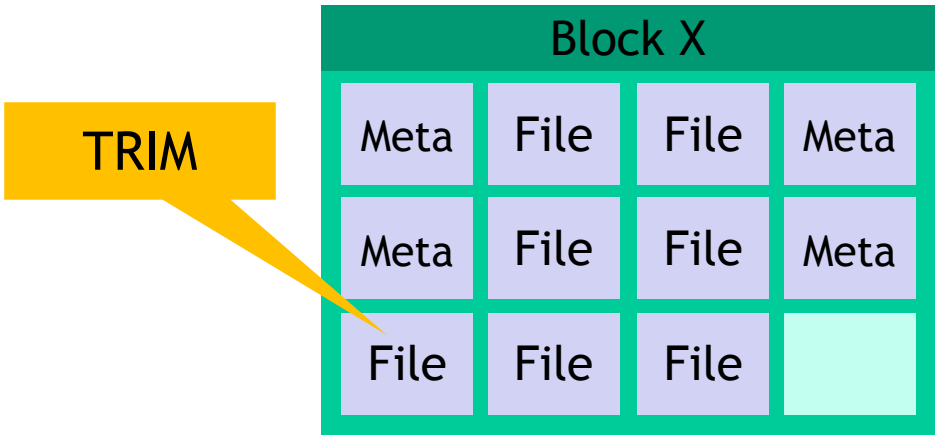


1. File is written to SSD
2. File is deleted
3. The GC executes
   - 9 pages look valid to the SSD
   - The OS knows only 2 pages are valid

- Lack of explicit delete means the GC wastes effort copying useless pages
- Hard drives are not GCed, so this was never a problem

Garbage Collection: The Ambiguity of Delete

- When performing background GC the SSD assumes to know which pages are invalid
- Problem: most file systems don't actually delete data
  - E.g., on Linux, the "delete" function is unlink() and it removes the file meta-data, but not the file itself
- New SATA command TRIM (SCSI – UNMAP)
  - The OS tells the SSD that specific LBAs are invalid and may be GCed
  - OS support for TRIM
    - Win 7, OSX Snow Leopard, Linux 2.6.33, Android 4.3

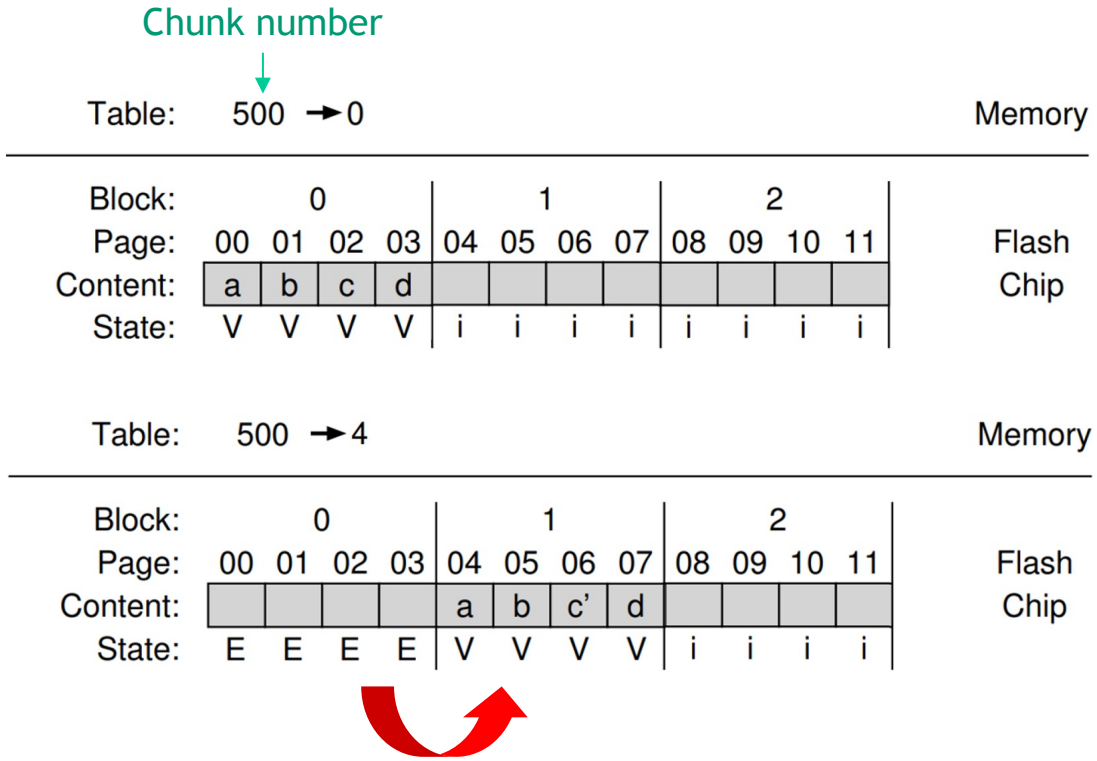


Mapping Table Size

- The size of page-level mapping table is too large
  - With a 1TB SSD with a 4byte entry per 4KB page, 1GB of is needed for mapping
- Some approaches to reduce the costs of mapping
  - Block-based mapping
    - Coarser grain approach
  - Hybrid mapping
    - Multiple tables (log -> page; data -> block)
  - Page mapping plus caching
    - Exploiting Data Locality

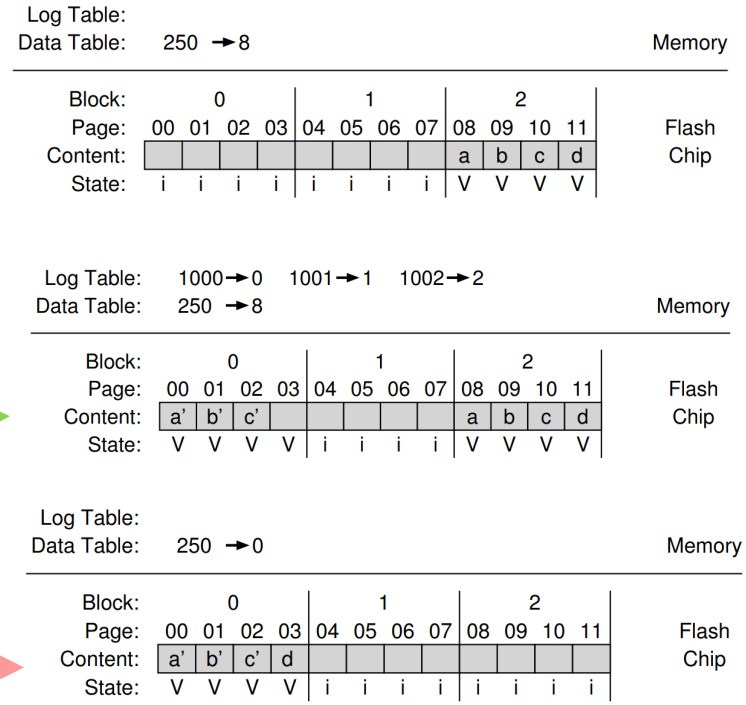
Block Mapping

- Mapping at block granularity
  - To reduce the size of a mapping table
- **Small write problem:**
  - The FTL must read a large amount of live data from the old block and copy them into a new one.
- **Example:**
  - Write(2000, a)
  - Write(2001, b)
  - Write(2002, c)
  - Write(2003, d)
  - Write(2002, c')



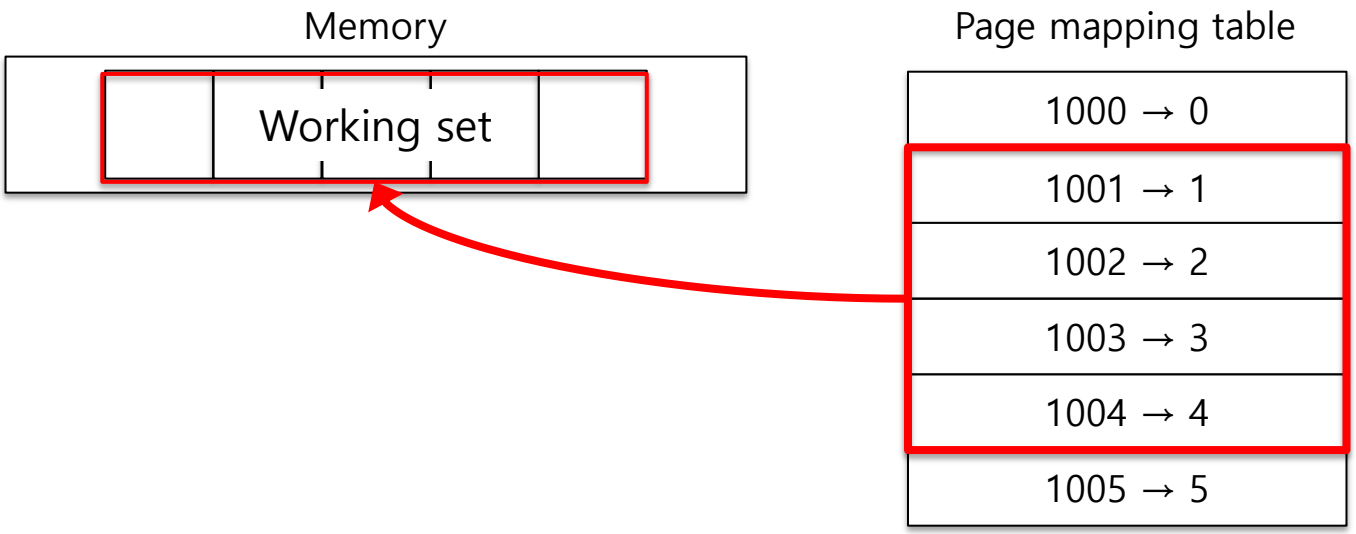
Hybrid mapping

- FTL maintains two tables:
  - Log blocks: page mapped
  - Data blocks: block-mapped
- When looking for a particular logical block, the FTL will consult the page mapping table and block mapping table in order
- Example:
  - Let's suppose the past sequence
    - Write(1000, a)
    - Write(1001, b)
    - Write(1002, c)
    - Write(1003, d)
  - Let's update some pages
    - Write(1000, a')
    - Write(1001, b')
    - Write(1002, c')
    - FTL updates only the page mapping information
  - When needed FTL can perform MERGE operations



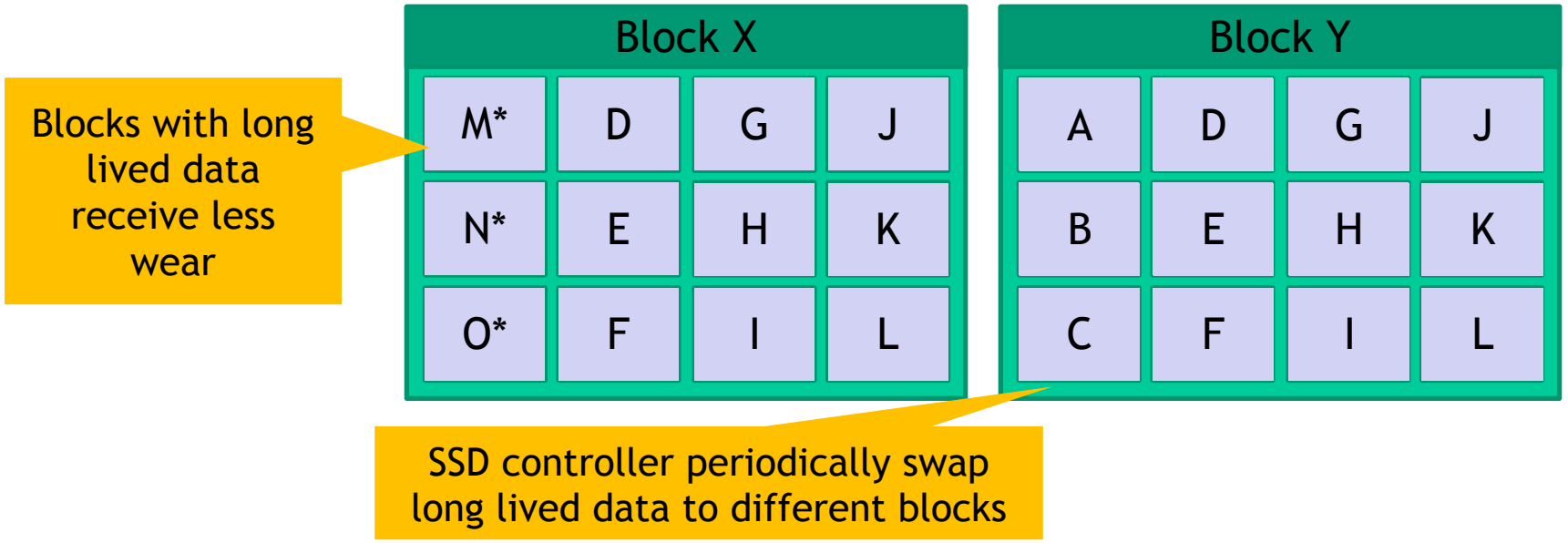
Page mapping plus caching

- Basic idea is to cache the active part of the page-mapped FTL
  - If a given workload only accesses a small set of pages, the translations of those pages will be stored in the FTL memory
- High performance without high memory cost if the cache can contain the necessary working set
- Cache miss overhead exists



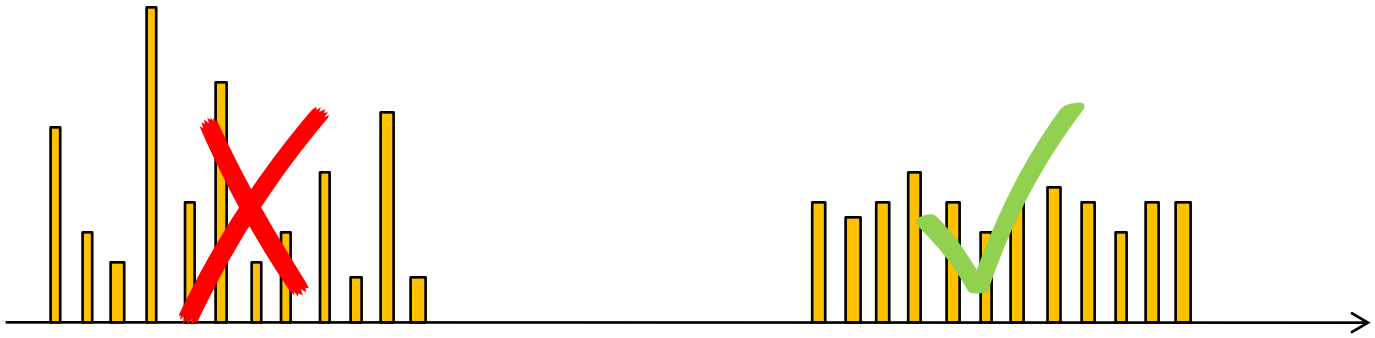
Wear Leveling

- 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

- 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
      - Maintain |Max(EW cycle) – Min(EW cycle)| < e



Solid State Disks – summary

- They cost more than the conventional HDD
- Flash memory can be written only a limited number of times (wear):
  - have a shorter lifetime
  - error correcting codes
  - over-provisioning (add some spare capacity)
- Different read/write speed
  - Write amplification
- Write performance degrades of one order of magnitude after the first writing
- Often the controller become the real bottleneck to the transfer rate

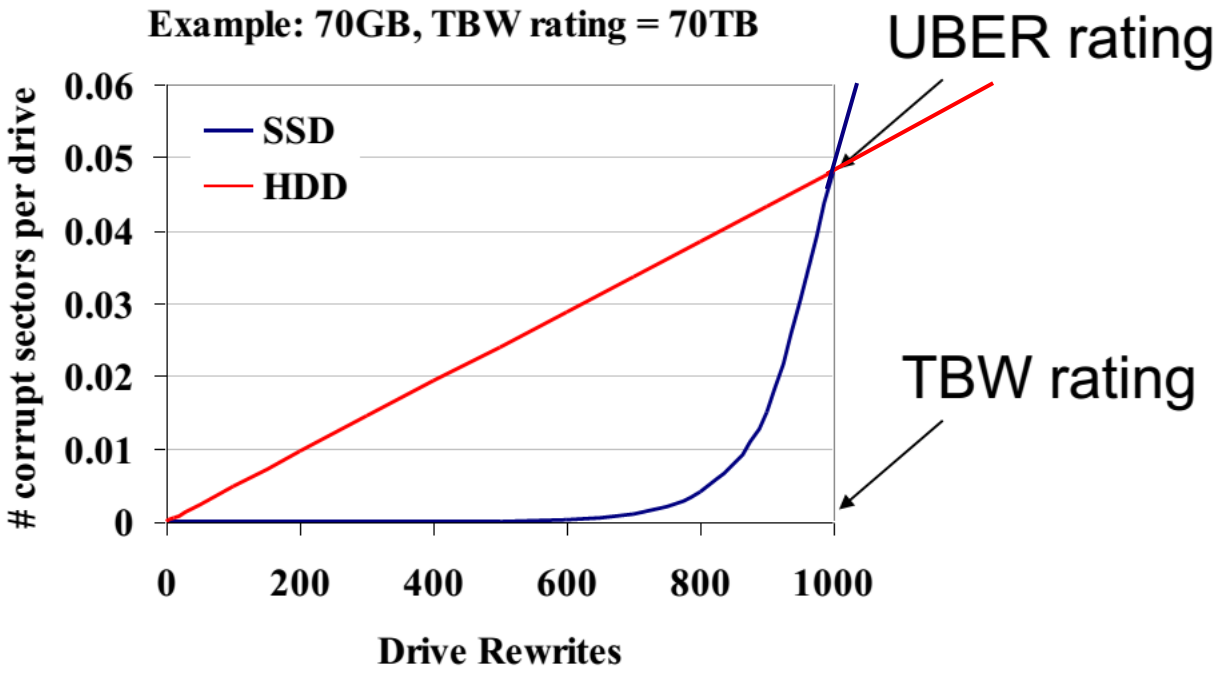
Solid State Disks – summary

- SSD are not affected by data-locality and must not be defragmented (actually, defragmentation may damage the disks)

- Flash Translation Layer is one of the key components
  - Data Allocation
  - Address Translation
  - Garbage Collection
  - Wear Leveling

HDD vs SSD

- **Unrecoverable Bit Error Ratio (UBER):** A metric for the rate of occurrence of data errors, equal to the number of data errors per bits read
- **Endurance rating:** Terabytes Written (TBW is the total amount of data that can be written into an SSD before it is likely to fail). The number of terabytes that may be written to the SSD while still meeting the requirements



Capacity and TBW

- **Memory cells can accept data recording between 3,000 and 100,000 during its lifetime**
  - Once the limit value is exceeded, the cell "forgets" any new data
- **A typical TBW for a 250 GB SSD is between 60 and 150 Terabytes of data written to the drive**
  - This means that in order to overcome, for example, a TBW of 70 Terabytes, a user should write 190 GB every day for a year or fill his SSD on a daily basis for two thirds with new files for a whole year
- **It is difficult to comment on the duration of SSDs**
  - Dell, in an old study (2011), spoke of an estimated duration between three months and ten years explaining, however, that there are so many factors (temperature and workload) that may depend on the life of an SSD that is very difficult to make predictions

Some data concerning recent SSD Seagate

Specifications	2TB	1TB	500GB	250GB
Model Number	ZA2000CM10003	ZA1000CM10003	ZA500CM10003	ZA250CM10003
Interface	SATA 6 Gb/s	SATA 6 Gb/s	SATA 6 Gb/s	SATA 6 Gb/s
NAND Flash Memory	3D TLC	3D TLC	3D TLC	3D TLC
Form Factor	2.5 in × 7 mm	2.5 in × 7 mm	2.5 in × 7 mm	2.5 in × 7 mm
Performance				
Sequential Read (Max, MB/s), 128KB[1]	560	560	560	560
Sequential Write (Max, MB/s), 128KB[1]	540	540	540	540
Random Read (Max, IOPS), 4 KB QD32[1]	90,000	90,000	90,000	90,000
Random Write (Max, IOPS), 4 KB QD32[1]	90,000	90,000	90,000	90,000
Endurance/Reliability				
Total Bytes Written (TB)	1,170	600	300	150
Mean Time Between Failures (MTBF, hours)	1,800,000	1,800,000	1,800,000	1,800,000
Warranty, Limited (years)	5	5	5	5
Power Management				
Active Power, Average (W)	2.8	2.7	2.5	2.5
Idle Power, Average (mW)	128	120	115	116
DevSleep (mW)	5	5	5	5
Environmental				
Temperature, Operating Internal (°C)	0°C – 70°C	0°C – 70°C	0°C – 70°C	0°C – 70°C
Temperature, Non-operating (°C)	-40°C – 85°C	-40°C – 85°C	-40°C – 85°C	-40°C – 85°C
Shock, Non-operating: 0.5 ms (Gs)	1,500	1,500	1,500	1,500
Physical				
Height (mm/in, max)	7.1 mm/0.279 in	7.1 mm/0.279 in	7.1 mm/0.279 in	7.1 mm/0.279 in
Width (mm/in, max)	70.1 mm/2.759 in	70.1 mm/2.759 in	70.1 mm/2.759 in	70.1 mm/2.759 in
Depth (mm/in, max)	100.35 mm/3.951 in	100.35 mm/3.951 in	100.35 mm/3.951 in	100.35 mm/3.951 in
Weight (lb/g)	50 g/0.11 lb	50 g/0.11 lb	50 g/0.11 lb	50 g/0.11 lb
Bulk Pack Specifications				
Carton Unit Quantity	20	20	20	20
Cartons Per Pallet/Layer	84 / 12	84 / 12	84 / 12	84 / 12
Special Features				
TRIM	Yes	Yes	Yes	Yes
S.M.A.R.T.	Yes	Yes	Yes	Yes
Halogen-free	Yes	Yes	Yes	Yes
RoHS compliance	Yes	Yes	Yes	Yes

POLITECNICO DI MILANO



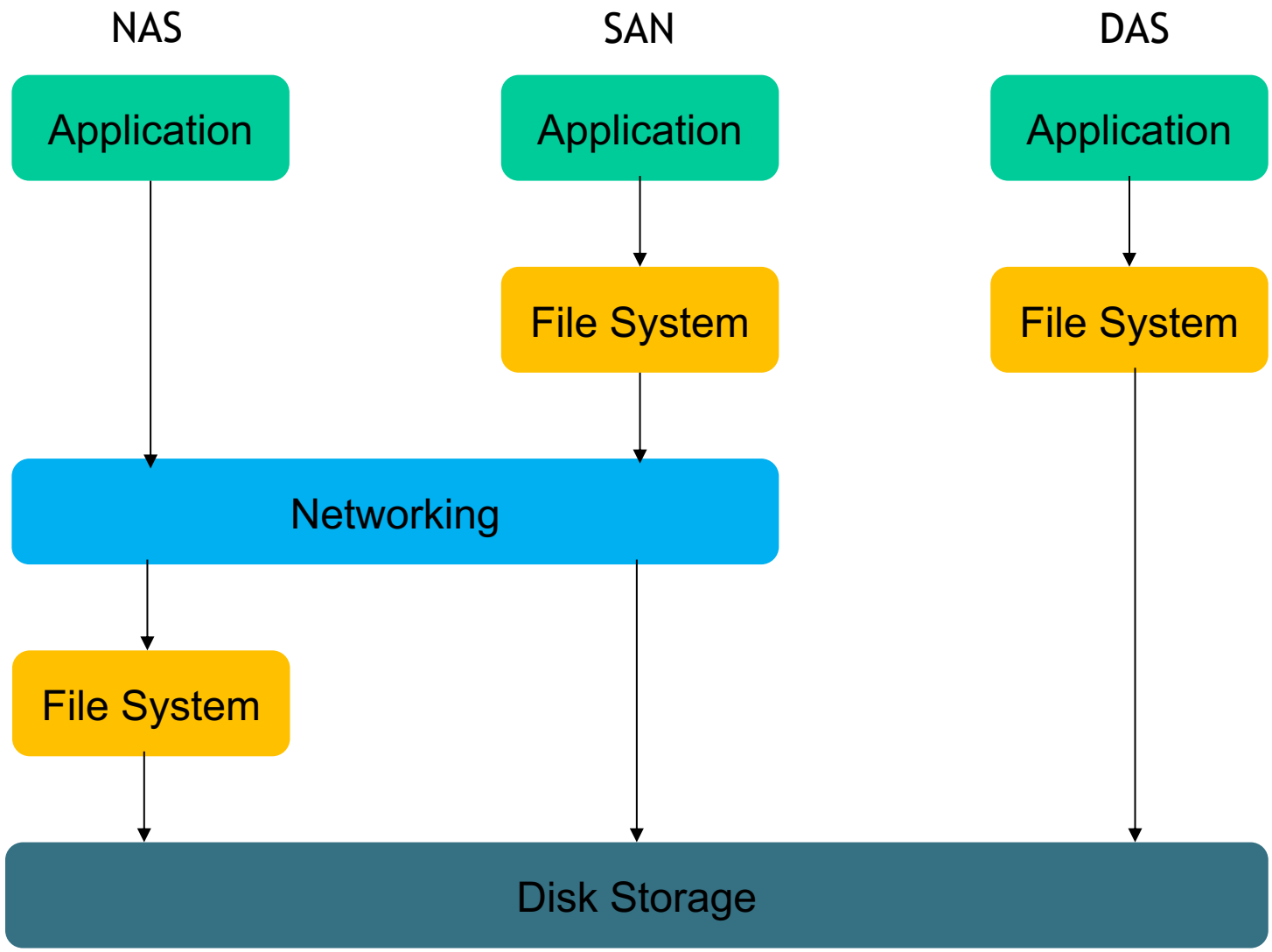


Storage systems: DAS, NAS and SAN

DAS, NAS and SAN

- A **Direct Attached Storage (DAS)** is a storage system directly attached to a server or workstation. They are visible as disks/volumes by the client OS
- A **Network Attached Storage (NAS)** is a computer connected to a network that provides only file-based data storage services (e.g., FTP, Network File System and SAMBA) to other devices on the network and is visible as File Server to the client OS
- **Storage Area Networks (SAN)** are remote storage units that are connected to a server using a specific networking technology (e.g., Fiber Channel) and are visible as disks/volumes by the client OS

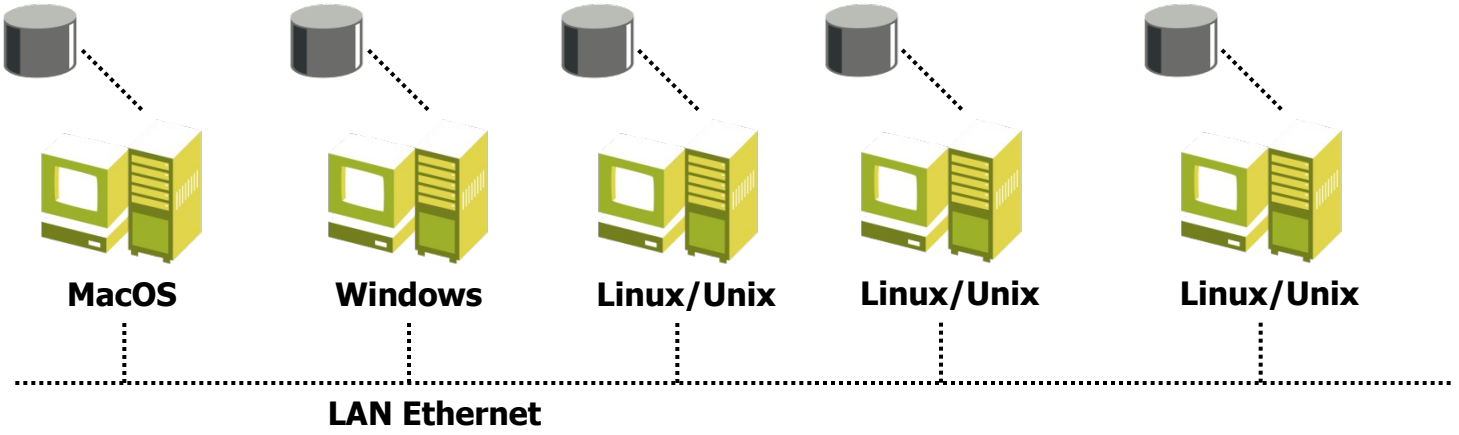
DAS, NAS, SAN: an architectural comparison



DAS
Direct Attached Storage

Direct Attached Storage

- DAS is a storage system directly attached to a server or workstation
- The term is used to differentiate non-networked storage from SAN and NAS (that will be described later)



Direct Attached Storage (DAS): physical model

Main features:

- limited scalability
- complex management
- to read files in other machines, the “file sharing” protocol of the OS must be used

Internal and external:

- DAS does not necessary mean “internal drives”
- All the external disks, connected with a point-to-point protocol to a PC can be considered as DAS

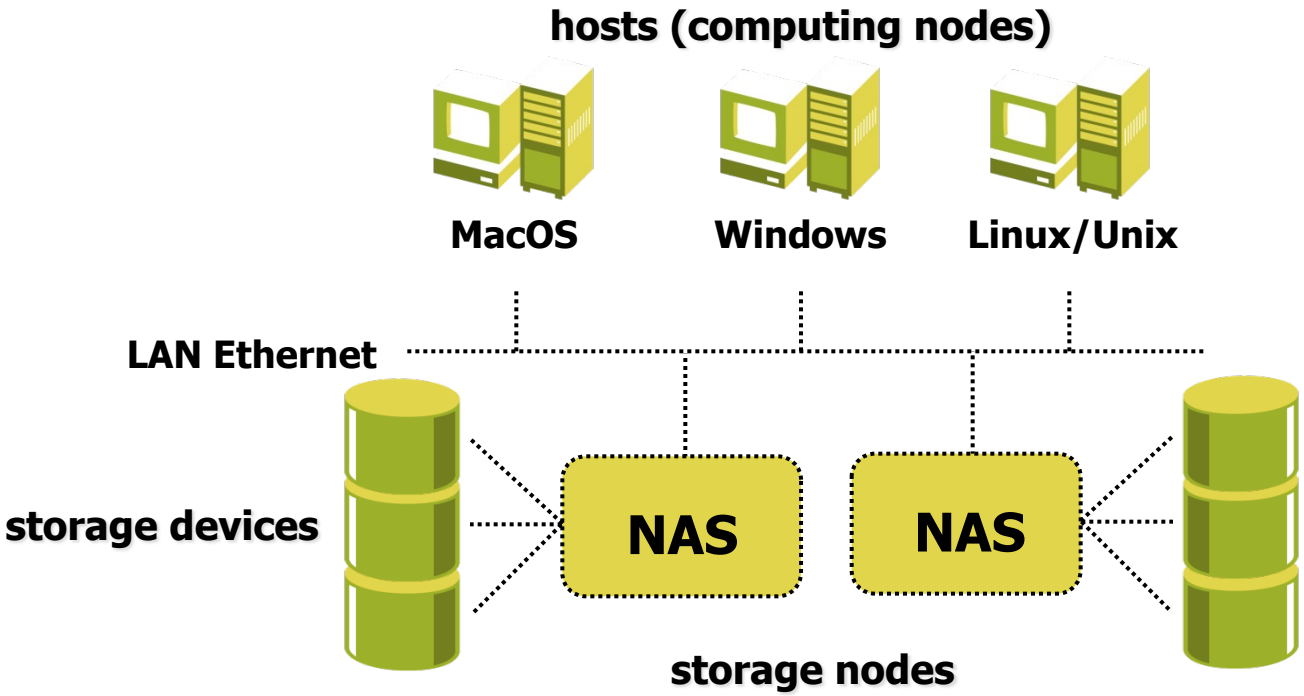
NAS
Network Attached Storage

Network Attached Storage (NAS)

- A NAS unit is a computer connected to a network that provides only file-based data storage services to other devices on the network
- NAS systems contain one or more hard disks, often organized into logical redundant storage containers or RAID
- Provide file-access services to the hosts connected to a TCP/IP network though Networked File Systems/SAMBA

Network Attached Storage (NAS): physical model

- Each NAS element has its own IP address
- Good scalability (incrementing the devices in each NAS element or incrementing the number of NAS elements)



NAS vs DAS

- The key differences between direct-attached storage (DAS) and NAS are
  - DAS is simply an extension of an existing server and is not necessarily networked
  - NAS is designed as an easy and self-contained solution for sharing files over the network

- Comparing NAS with local (non-networked) DAS, the performance of NAS depends mainly on the speed of and congestion on the network

SAN:

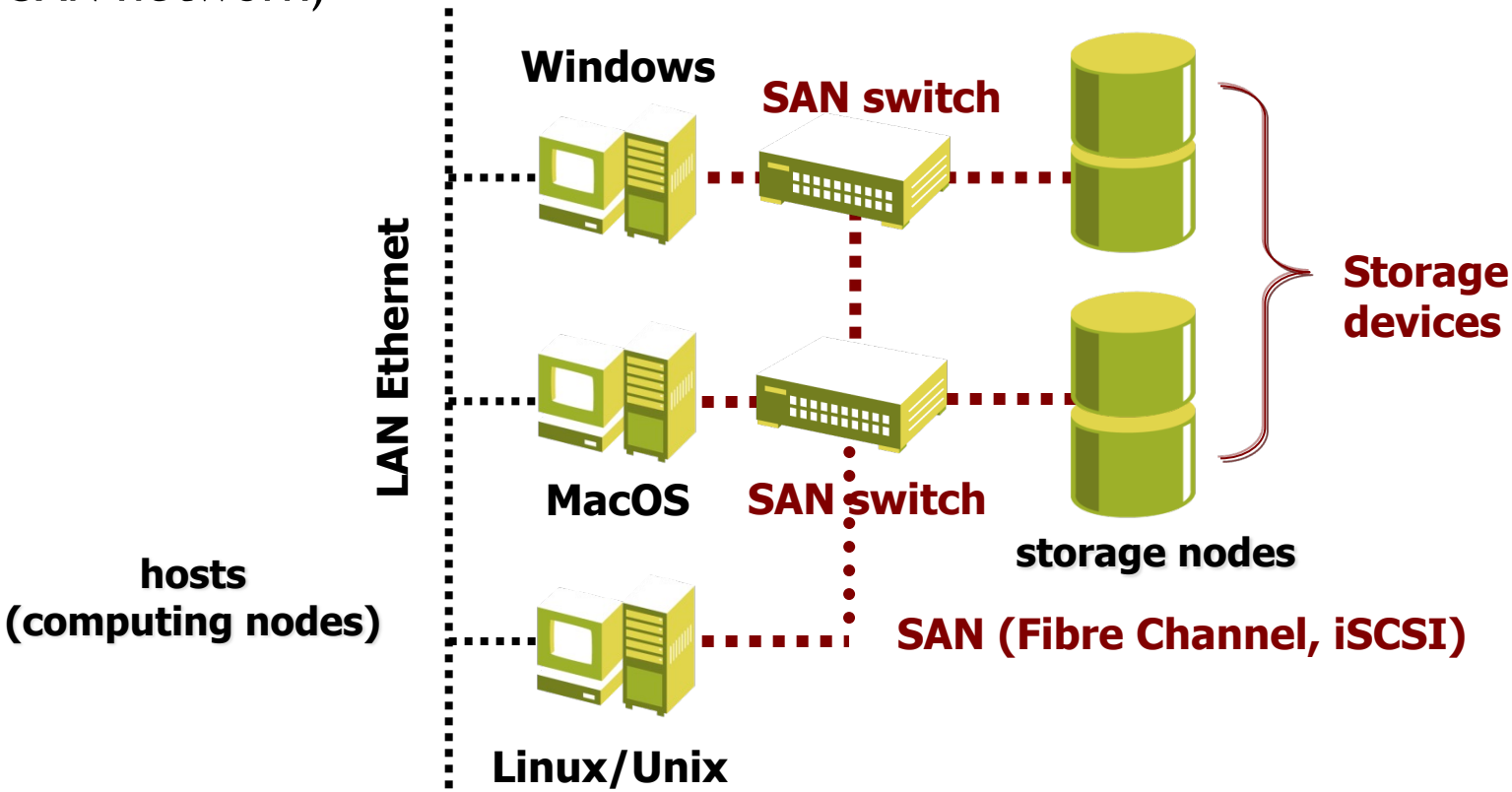
Storage Area Network

Storage Area Networks

Storage Area Networks, are remote storage units that are connected to a PC/server using a specific networking technology

SAN architecture

- SANs have a special network devoted to the accesses to storage devices
- Two distinct networks (one TCP/IP and one dedicated network, e.g., Fiber Channel)
- High scalability (simply increasing the storage devices connected to the SAN network)



NAS vs SAN

- NAS provides both storage and a file system
- This is often contrasted with SAN which provides only block-based storage and leaves file system concerns on the "client" side

- One way to loosely conceptualize **the difference between a NAS and a SAN** is that
  - **NAS appears to the client OS (operating system) as a file server (**the client can map network drives to shares on that server)
  - **a disk available through a SAN still appears to the client OS as a disk**: it will be visible in the disks and volumes management utilities (along with client's local disks), and available to be formatted with a file system

- Traditionally:
  - NAS is used for low-volume access to a large amount of storage by many users
  - SAN is the solution for petabytes (10^{12}) of storage and multiple, simultaneous access to files, such as streaming audio/video

DAS vs. NAS vs. SAN

	Application Domain	Advantages	Disadvantages
DAS	• Budget constraints • Simple storage solutions	• Easy setup • Low cost • High performance	• Limited accessibility • Limited scalability • No central management and backup
NAS	• File storage and sharing • Big Data	• Scalability • Greater accessibility • Performance	• Increased LAN traffic • Performance limitations • Security and reliability
SAN	• DBMS • Virtualized environments	• Improved performance • Greater scalability • Improved availability	• Costs • Complex setup and maintenance