NAND Flash Controller

Lee Hao Zhi

2022/4/29

新竹交大

- Duration: 2h/week
- Time: 10am-12pm

Date	Topic
4/29	Lecture - Introduction to NAND Flash & Controller Lecture - Introduction to FW concepts (basic) Lab - FUSE environment set-up (before next lecture)
5/6	Lecture - Introduction to FW concepts (advanced) Lab - Challenge
6/B	Students provide report and source code

AGENDA

NAND Flash Market

NAND Flash Device

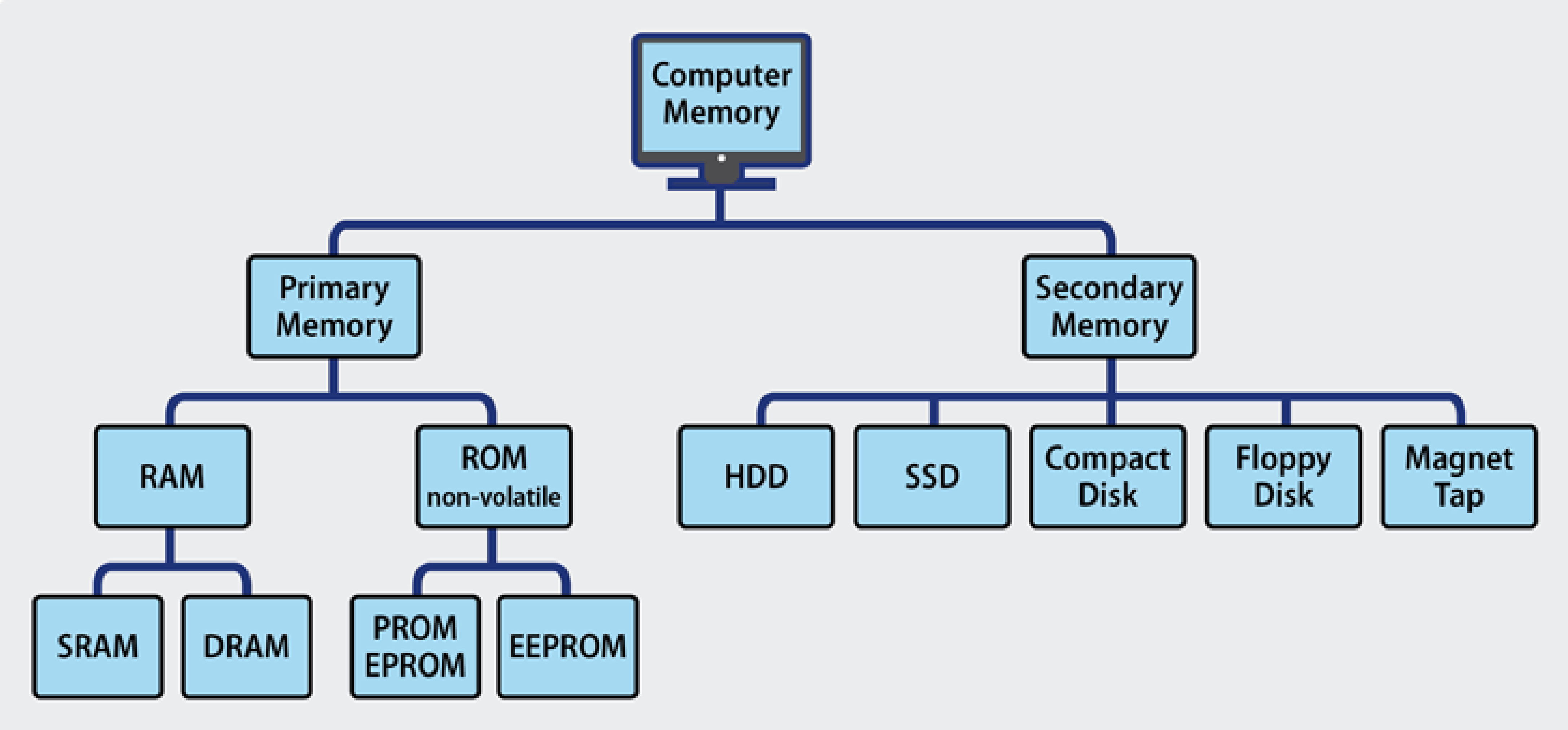
FW Concepts (Basic)

Summary

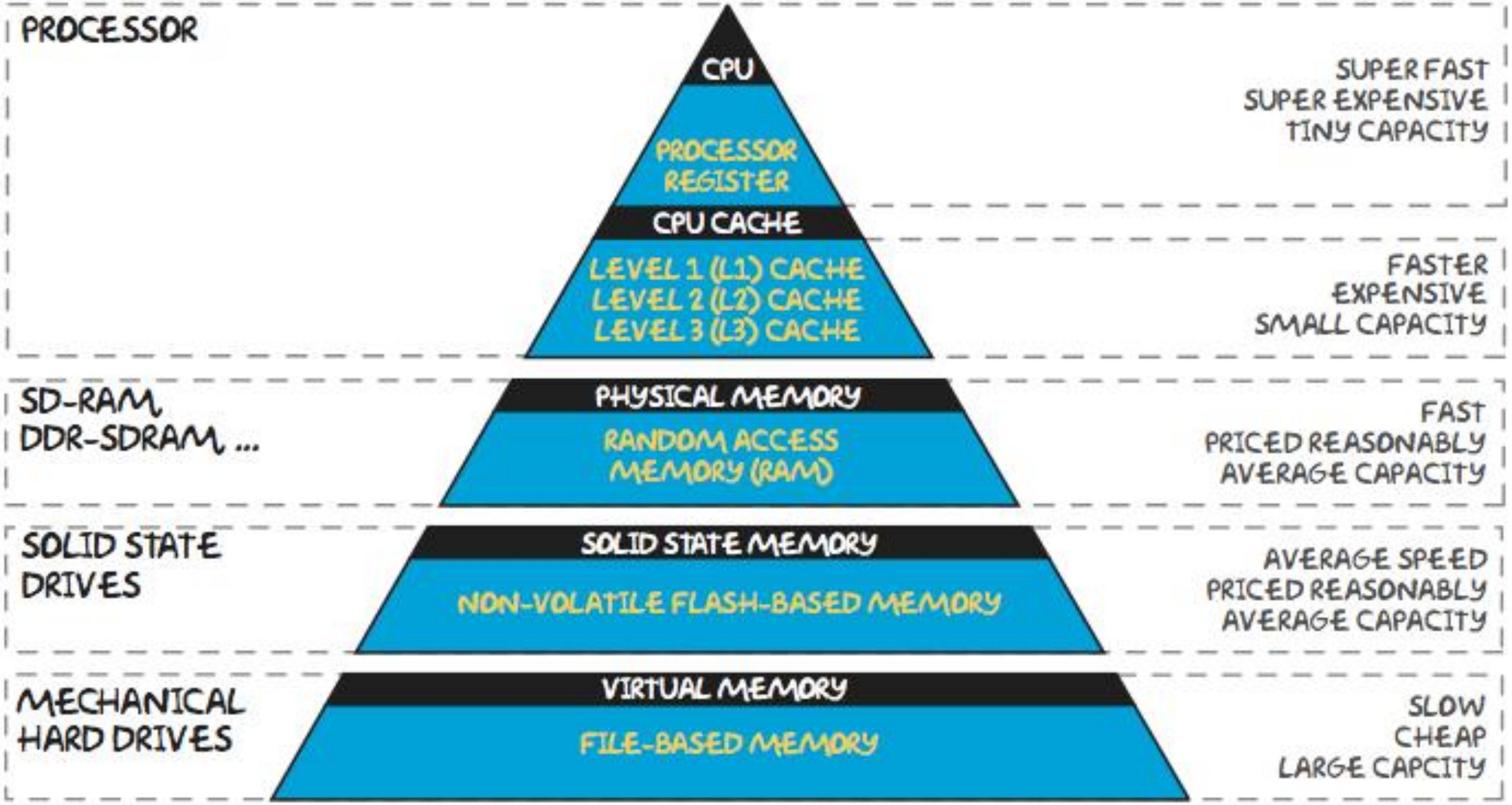
NAND Flash Market

PHISON

Memory Classification



Memory Hierarchy



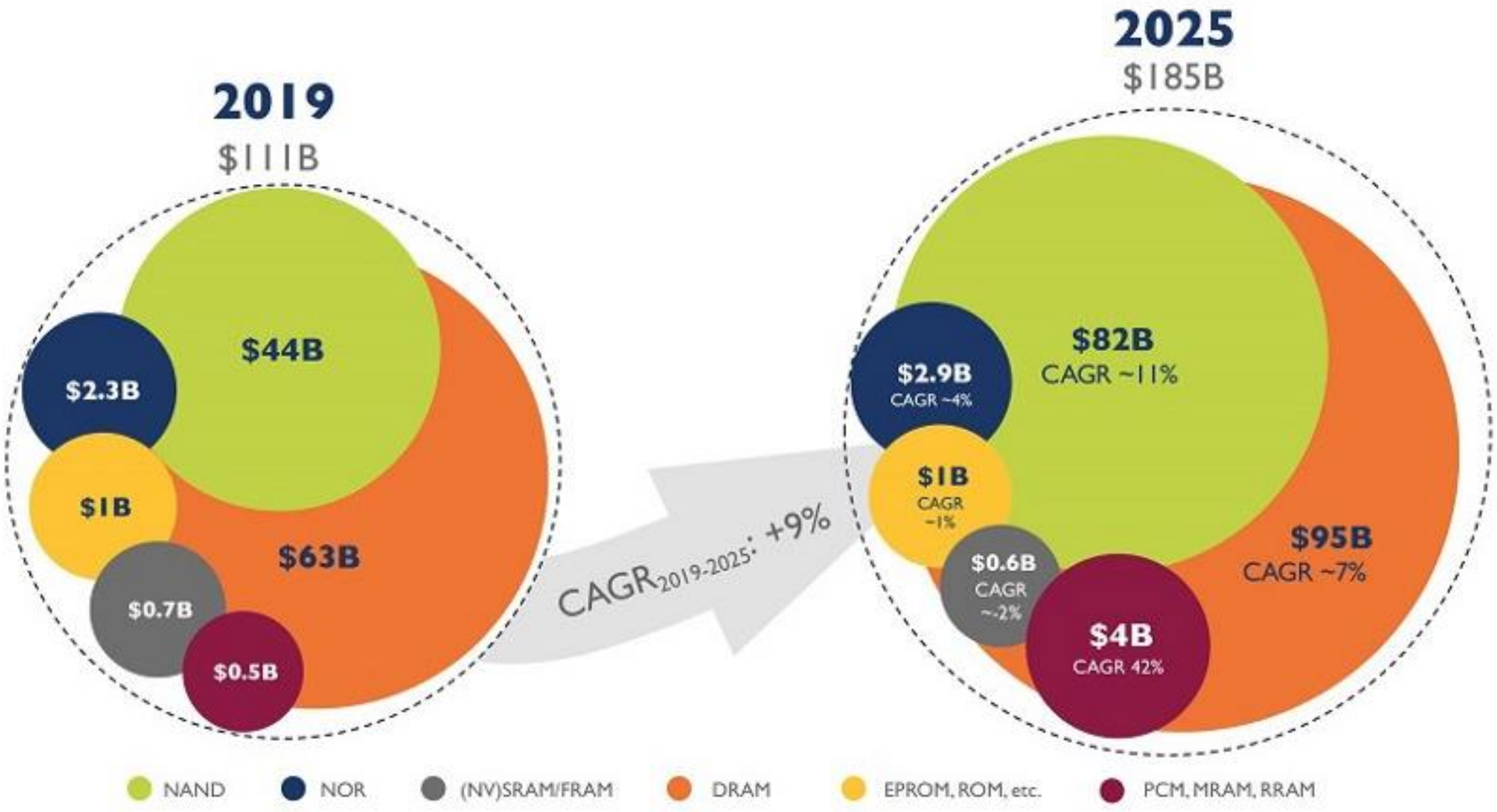
PHISON

Question

Which type of memory has the highest market cap growth in 5 years time?

Which type of memory has the highest CAGR in 5 years time?

Market of Memory



NAND Flash Is Everywhere

PC Computer

Netflix/FB/Amazon Servers

Smartphone

Game Consoles

Smart Speaker

Google Glass

PHISON

Smart Robots

Drones

Smart Treadmills

Smart Projectors

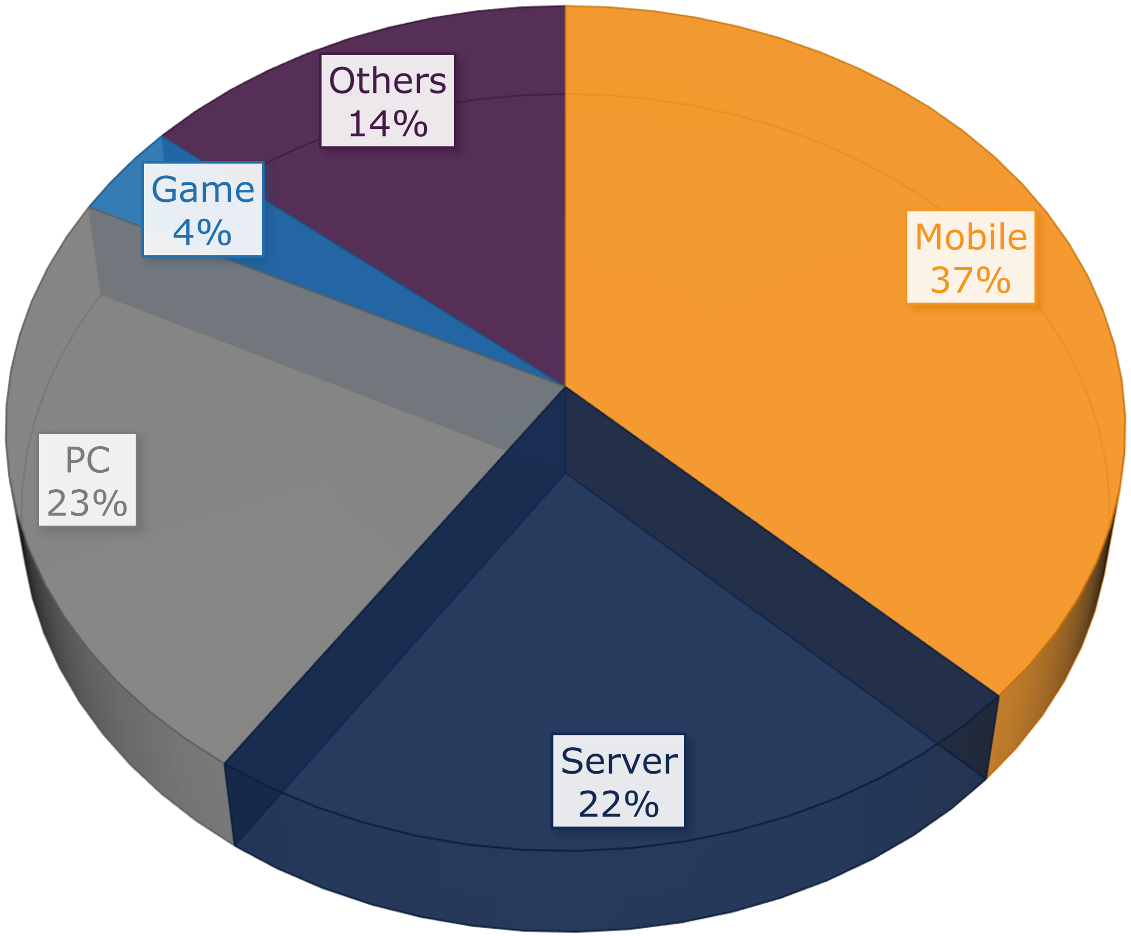
Electric Cars

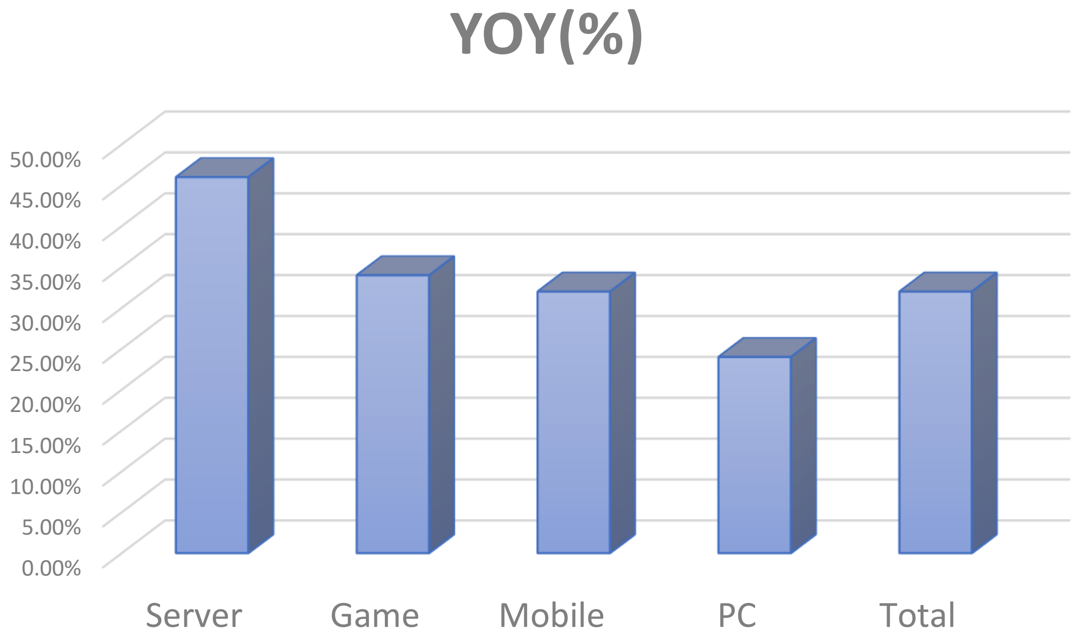
Question

Which type of application used the most of NAND Flash?

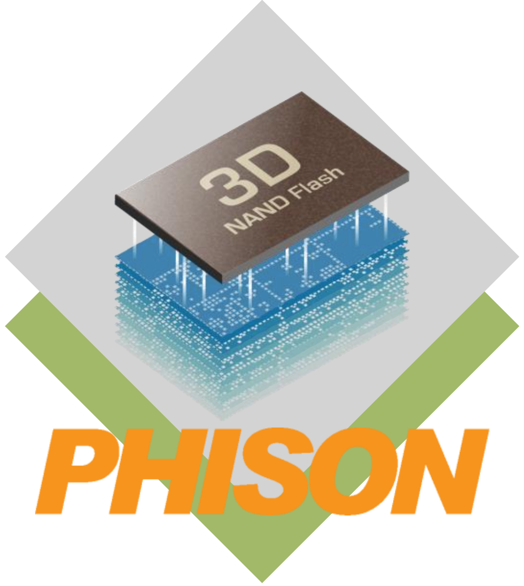
Which type of application that uses NAND Flash has the highest growth?

Distribution and Trend by Applications





Data Storage



Floppy Disk

1.44MB

Optical Disc

CD: 700MB
DVD: 4.7GB
Blu-ray Disc: 25GB~128GB

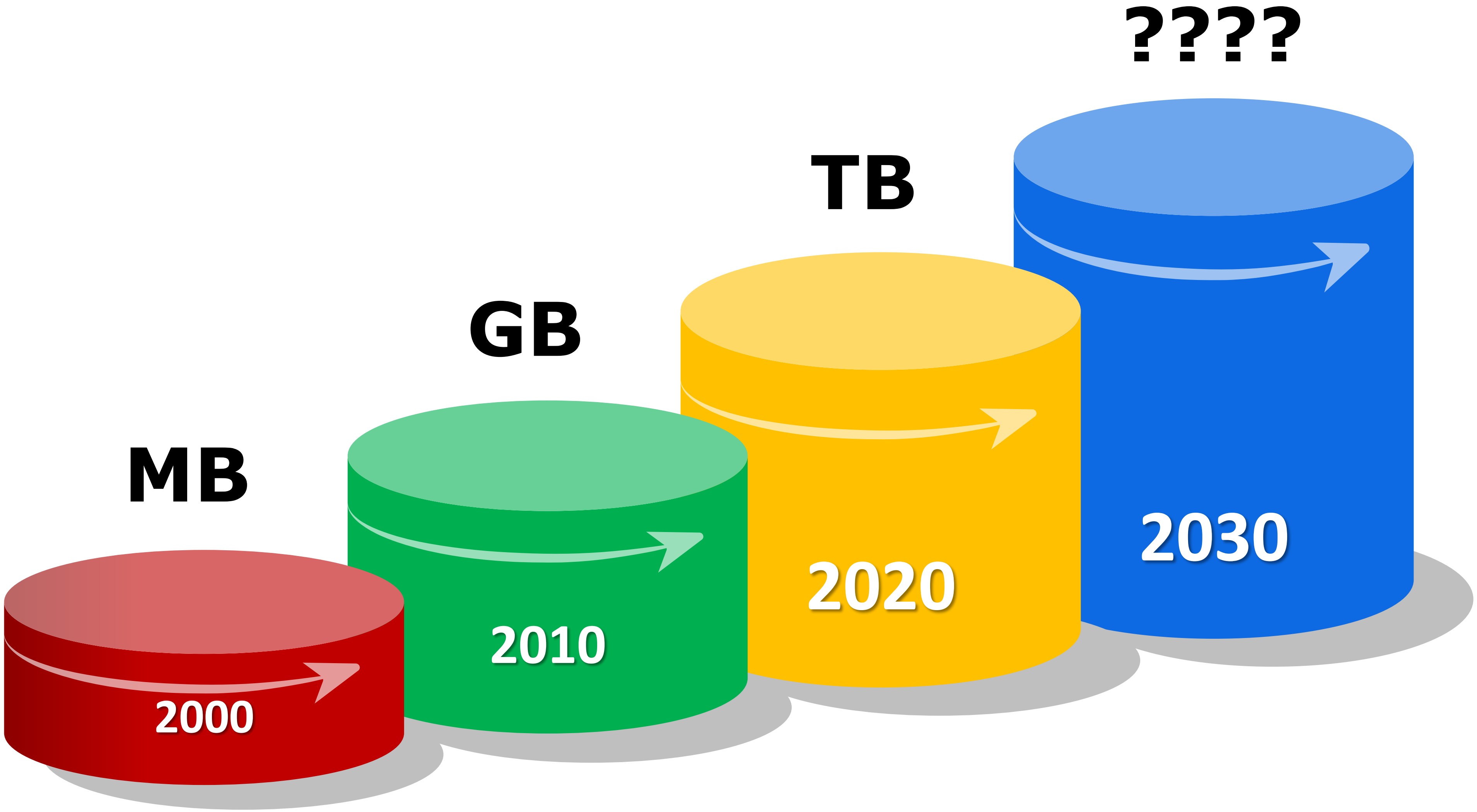
HDD

36GB~1TB~Beyond

NAND Storage

1GB~1TB~8TB~Beyond

Demand Never Stop



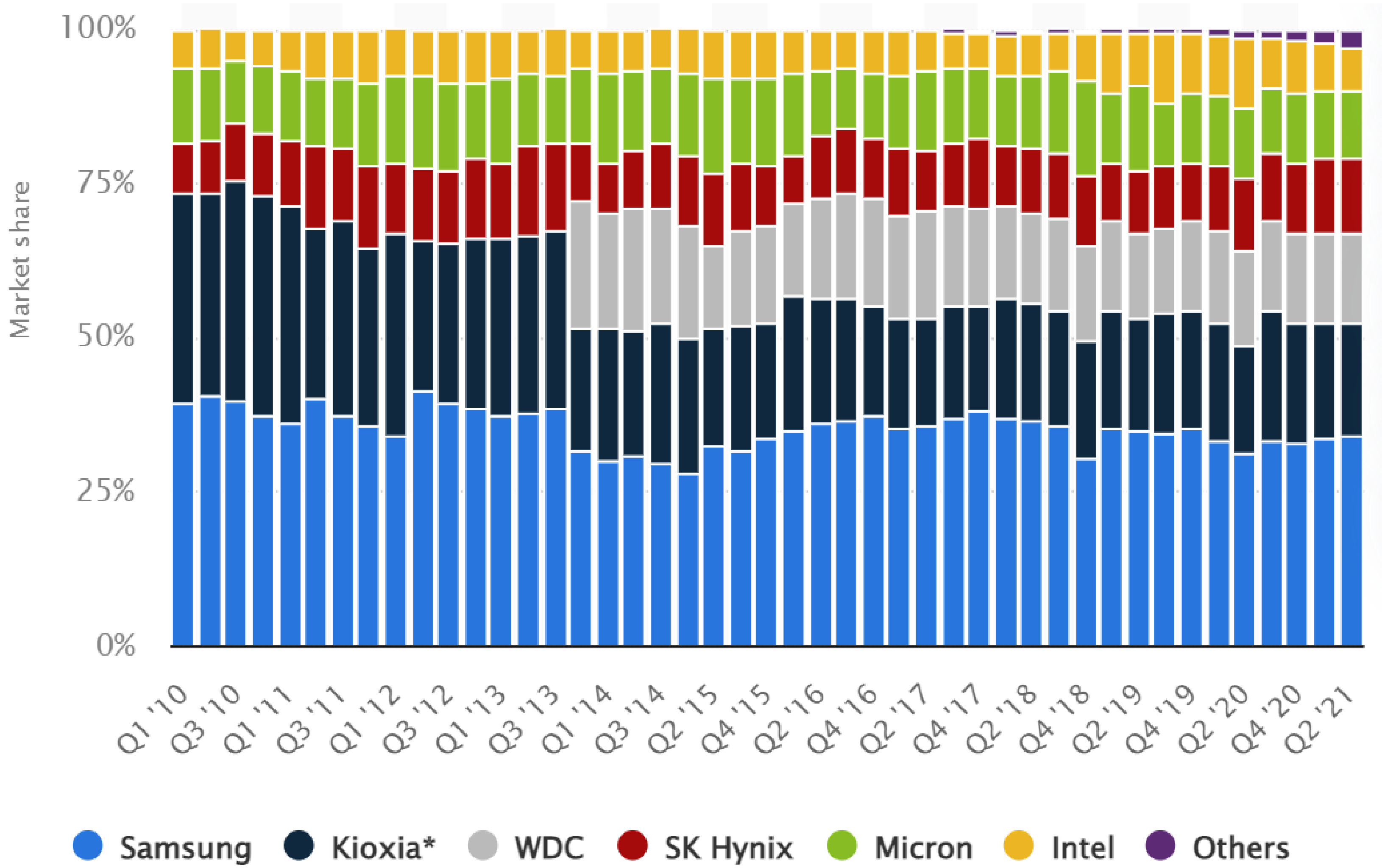
Question

Which company invented NAND Flash?

A NAND Flash company founder used to be a farmer, guess what did he used to grow?

NAND Flash Players

NAND Flash Players



ECO System of NAND Flash Storage System

NAND Flash Vendors

KIOXIA
WD Western Digital
Micron
intel
SAMSUNG
SK hynix

NAND Flash Controller Vendors

PHISON
MARVELL
REALTEK
KIOXIA
WD Western Digital
SiliconMotion
Micron
intel
Asolid
SAMSUNG
SK hynix

NAND Flash Module Vendors

PHISON
Kingston TECHNOLOGY
KIOXIA
WD Western Digital
SEAGATE
Micron
intel
CORSAIR
PNY
SAMSUNG
SK hynix
ADATA
Transcend

Storage System and Application Vendors

DELL EMC
IBM
H3C 数字化解决方案领导者
HUAWEI
NetApp
DELL
hp
acer
Lenovo
ASUS
vivo
oppo
mi
HUAWEI

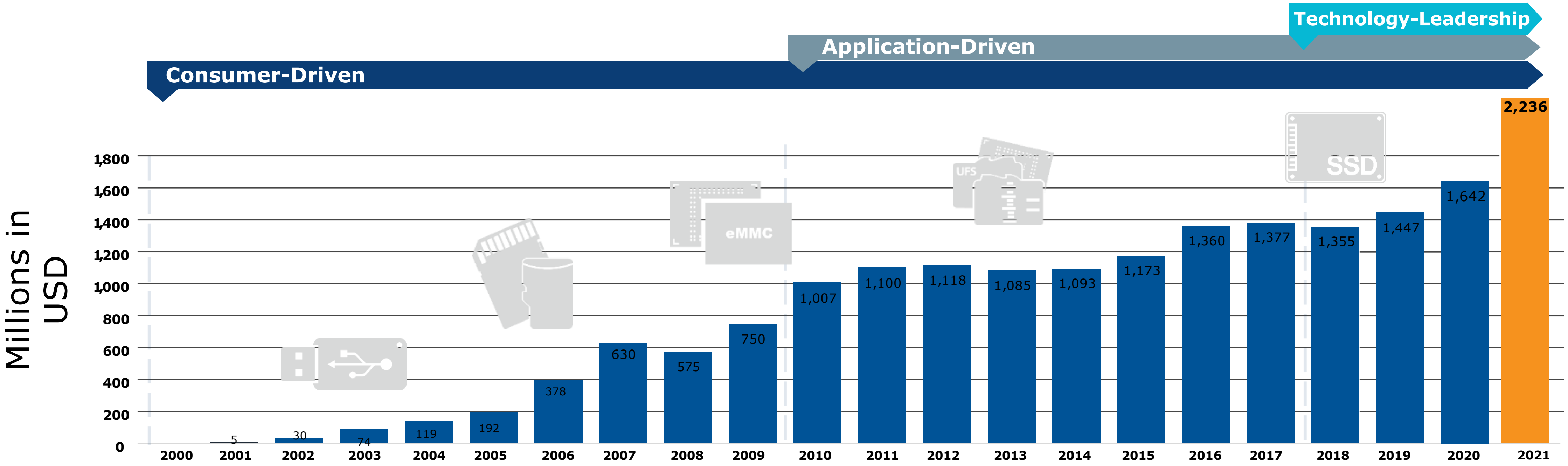
About Phison

World No.1
NAND Flash Controller Solution

Taiwan Top 4
IC Design House

20%+ Worldwide
SSD Controller Market Share

Gaming **Enterprise** **Automotive** **Industrial** **Embedded ODM**



PHISON

Key Facts

Experience
21+ Years

80%+
RD Exp./OPEX

3400+ HC
75% R&D

20%+
Global SSD Controller Market Share

2000+
Worldwide patents

$2.23B USD
2021 Revenue

Overview

MEDIATEK, Qualcomm, intel, Amlogic

PHISON Knows What You Need

Top NAND Flash Controller solution in the world

SAMSUNG, SK hynix, KIOXIA, Western Digital, Micron, intel, YMTC

Application

Filesystem

Host Driver

Host: PC, Tablet, Smartphone, Smartwatch, etc.

Protocol: USB, SD, eMMC, UFS, SATA, NVMe, etc.

Controller

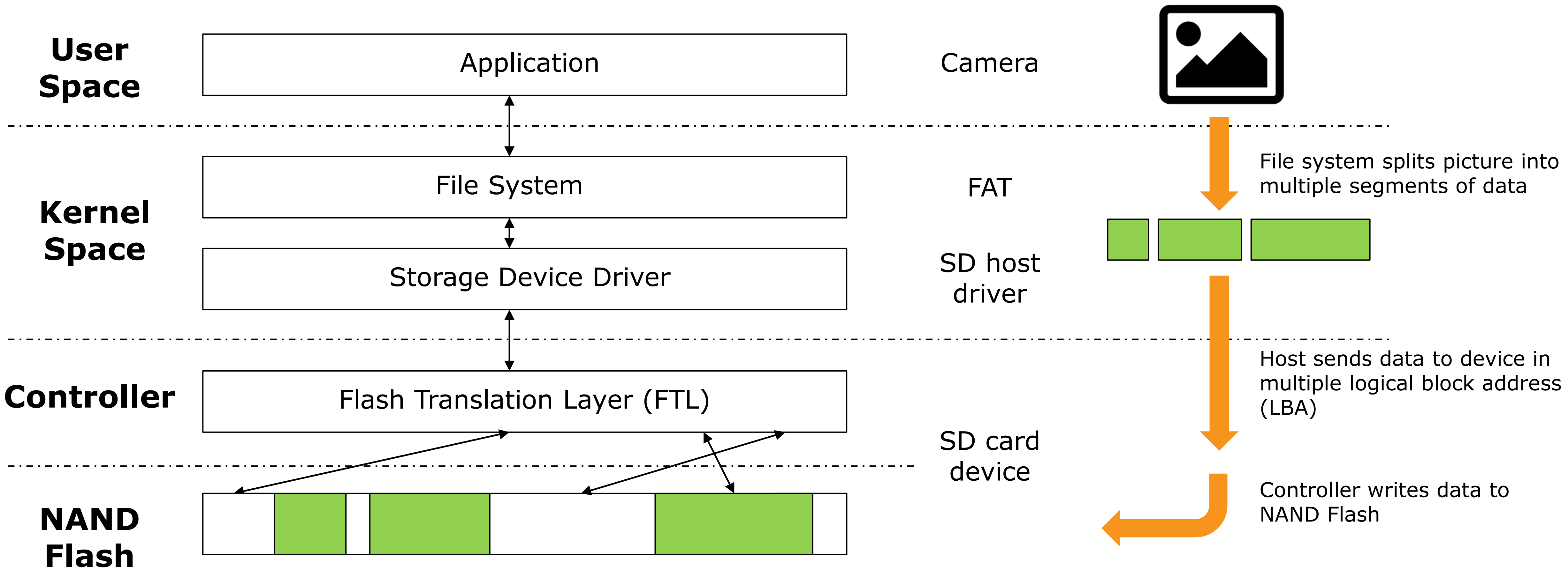
NAND Flash

Host layer

Flash Translation layer (FTL)

NAND layer

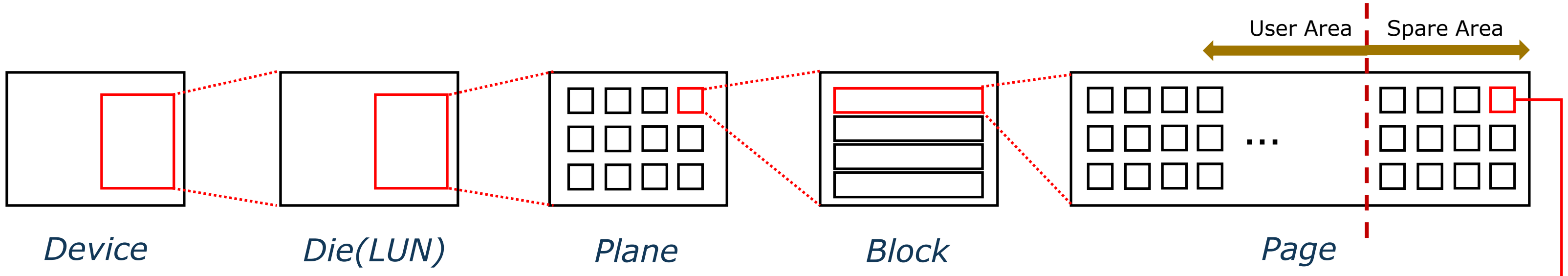
Application & Flash Storage Device

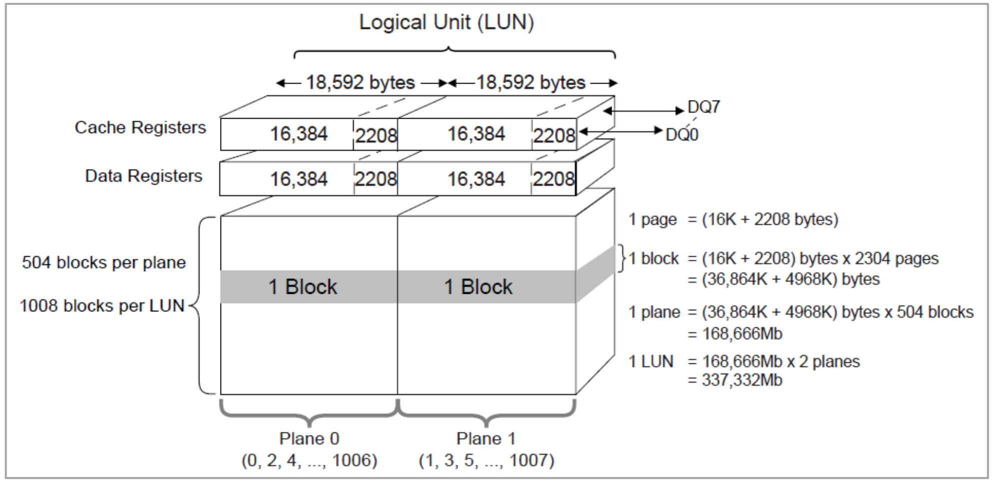


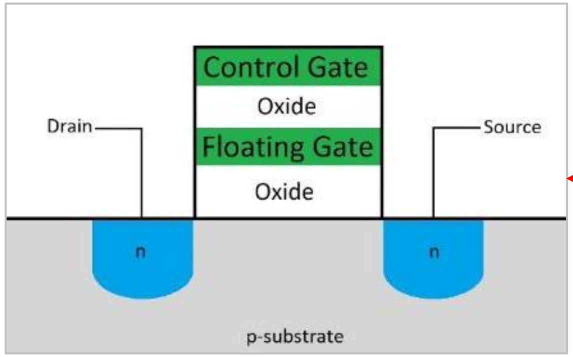
NAND Flash Device

PHISON

NAND Flash Structure

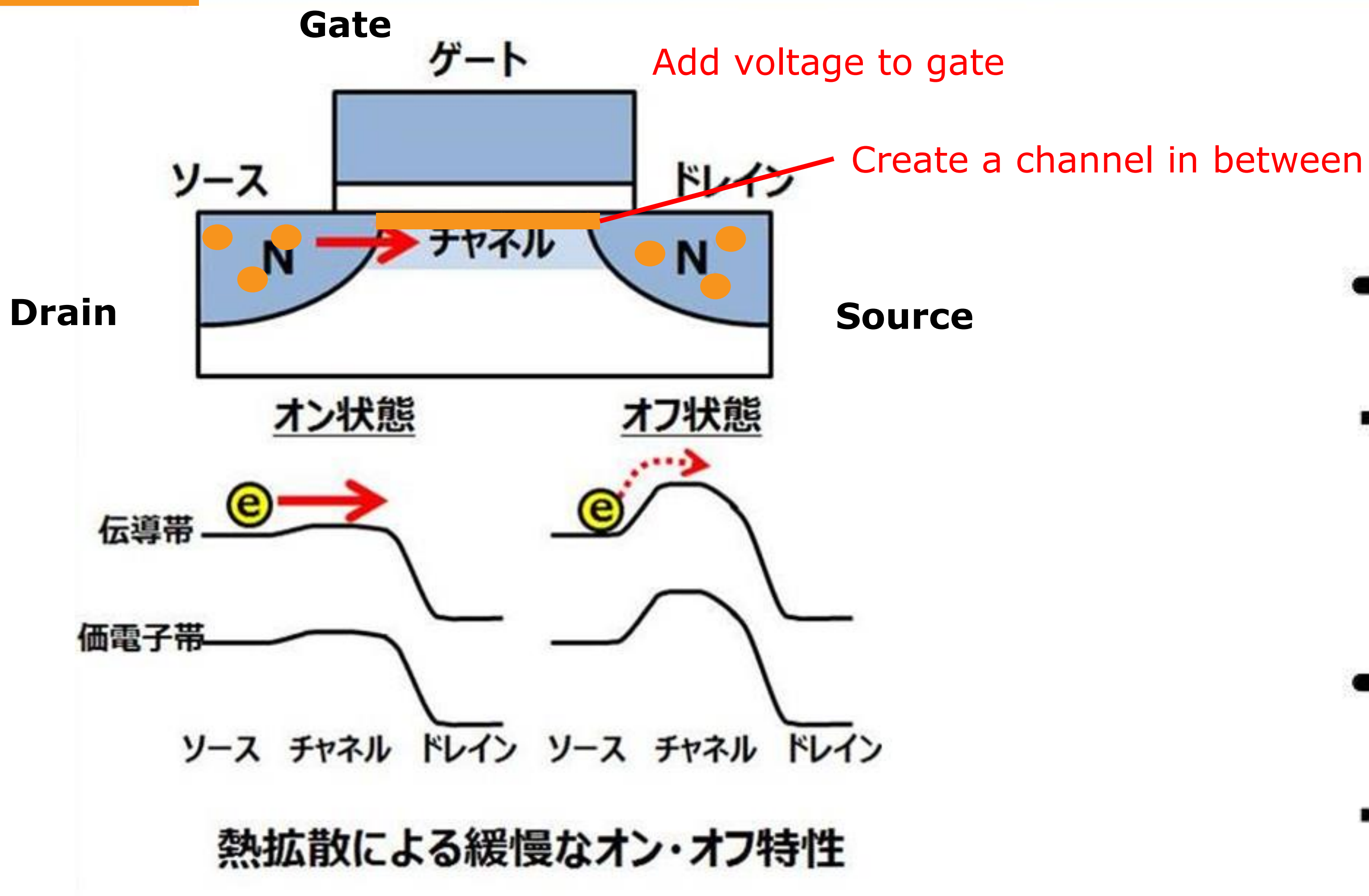


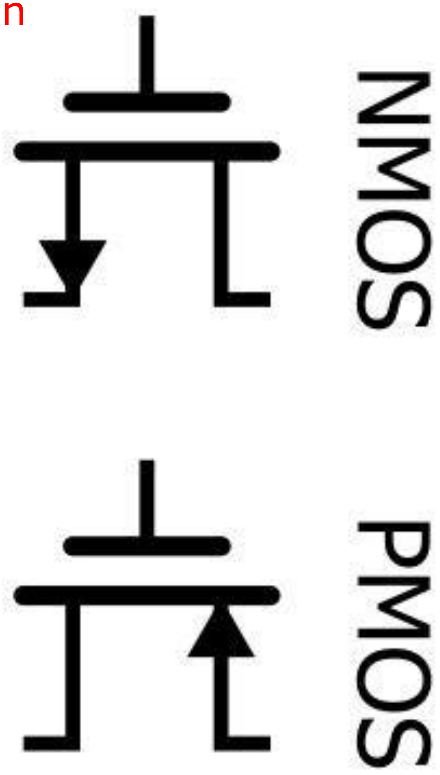




Memory Cell

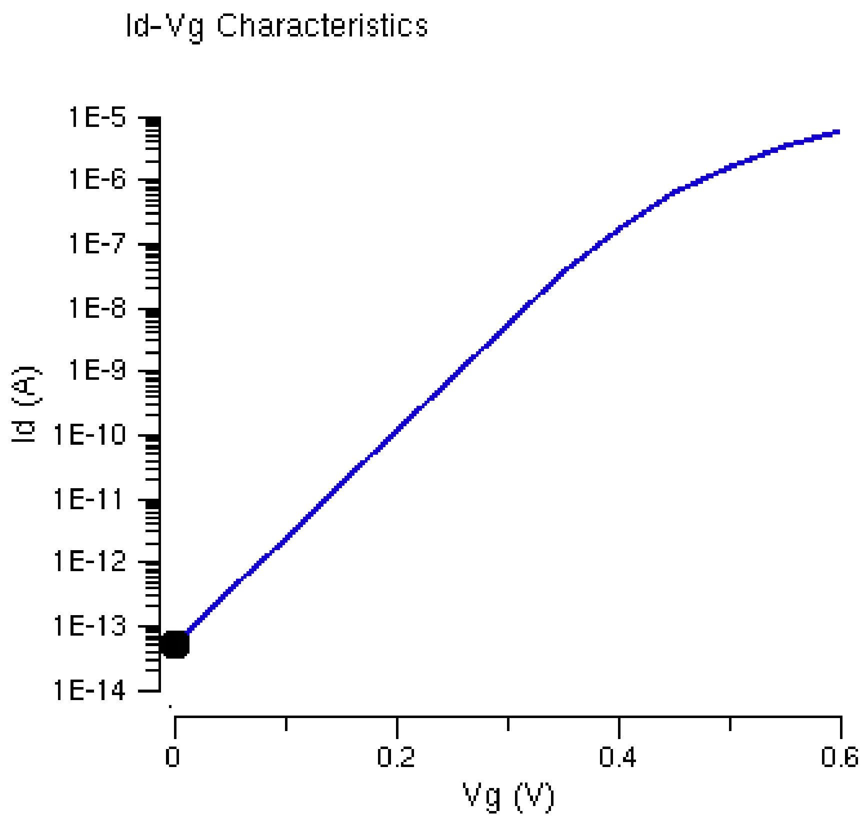
What’s MOSFET

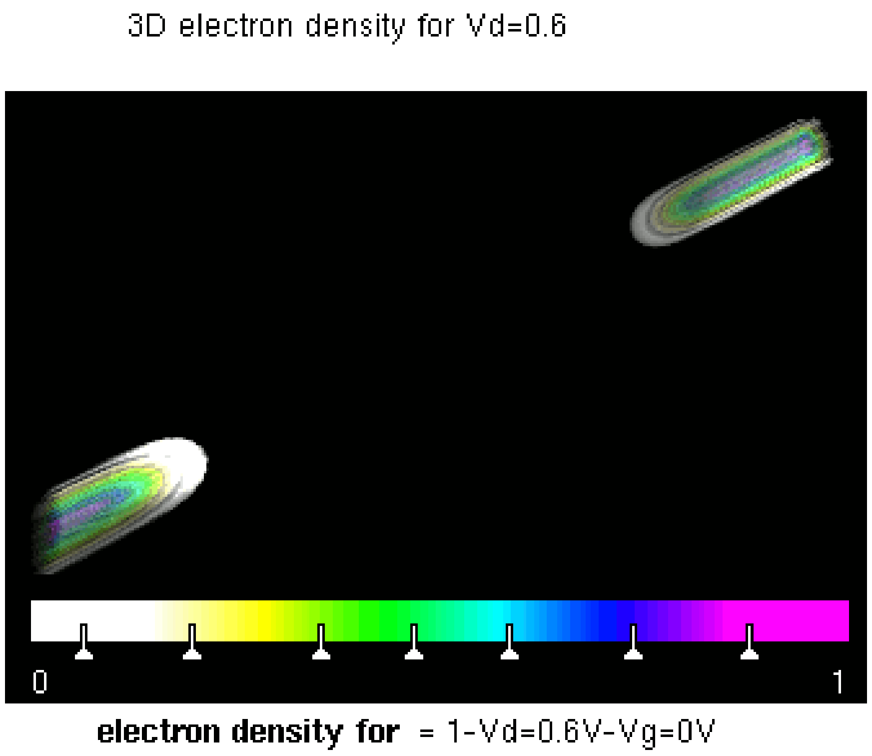




D. Kahng & Martin Atalla, 1960, Bell Lab

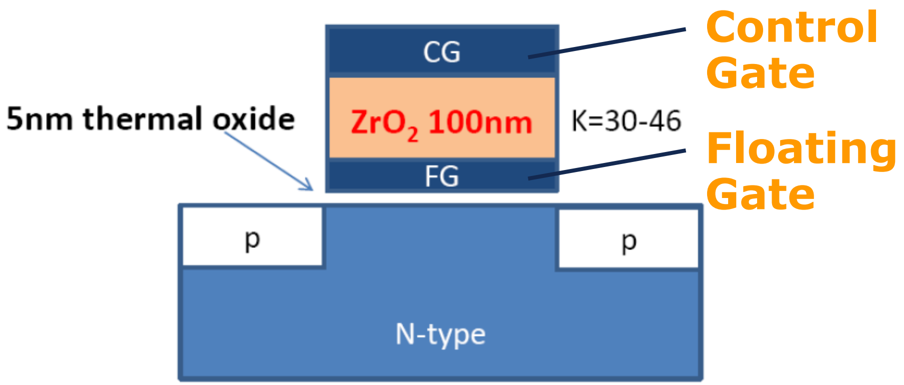
What's Threshold Voltage?

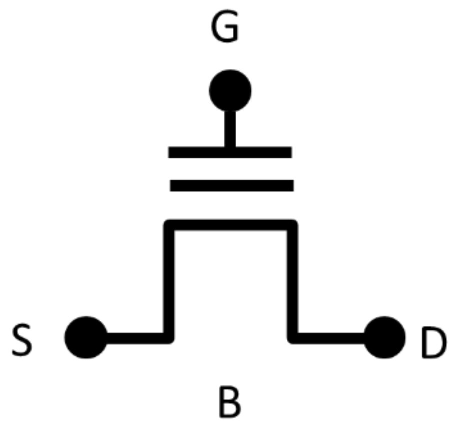




- As known as V_{th}, V_t :a gate voltage which can "turn on" the channel of a transistor.

What’s Floating Gate MOSFET





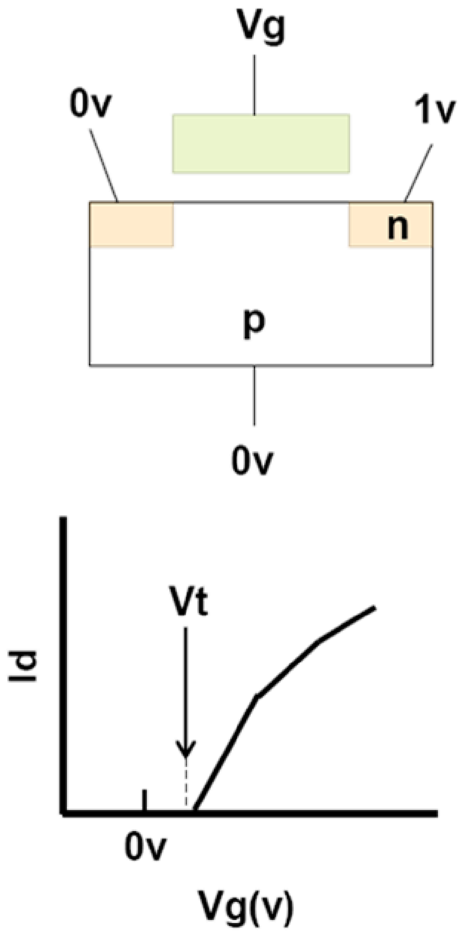
Floating Gate (FG)-NVM cell, 1967 (Bell)

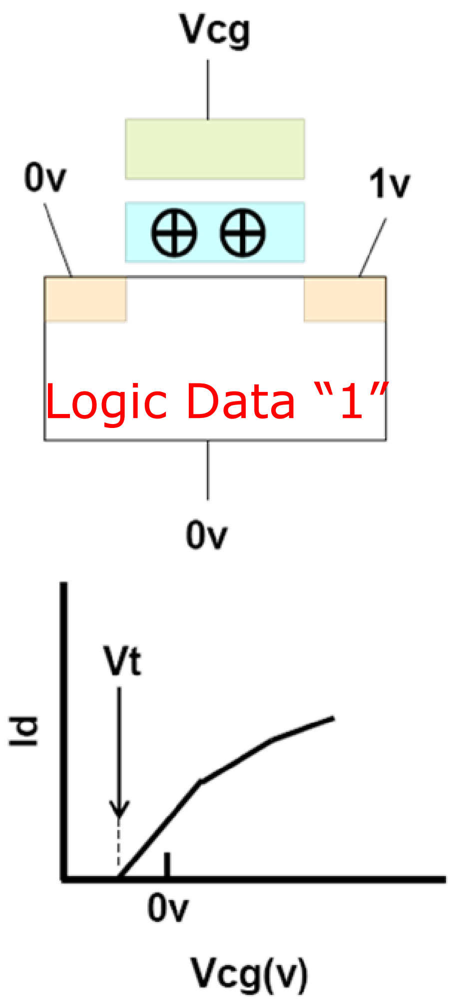
- ✓ **Electrically isolated**
- ✓ **Charge contained in it remains unchanged for long periods of time**

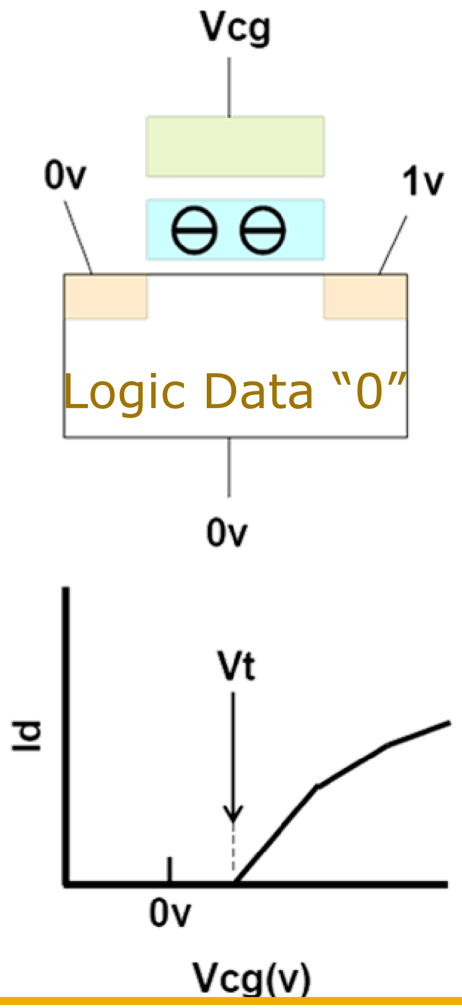
半導體之父 施敏教授研發的「浮閘記憶體」

Threshold Voltage of Memory Cell

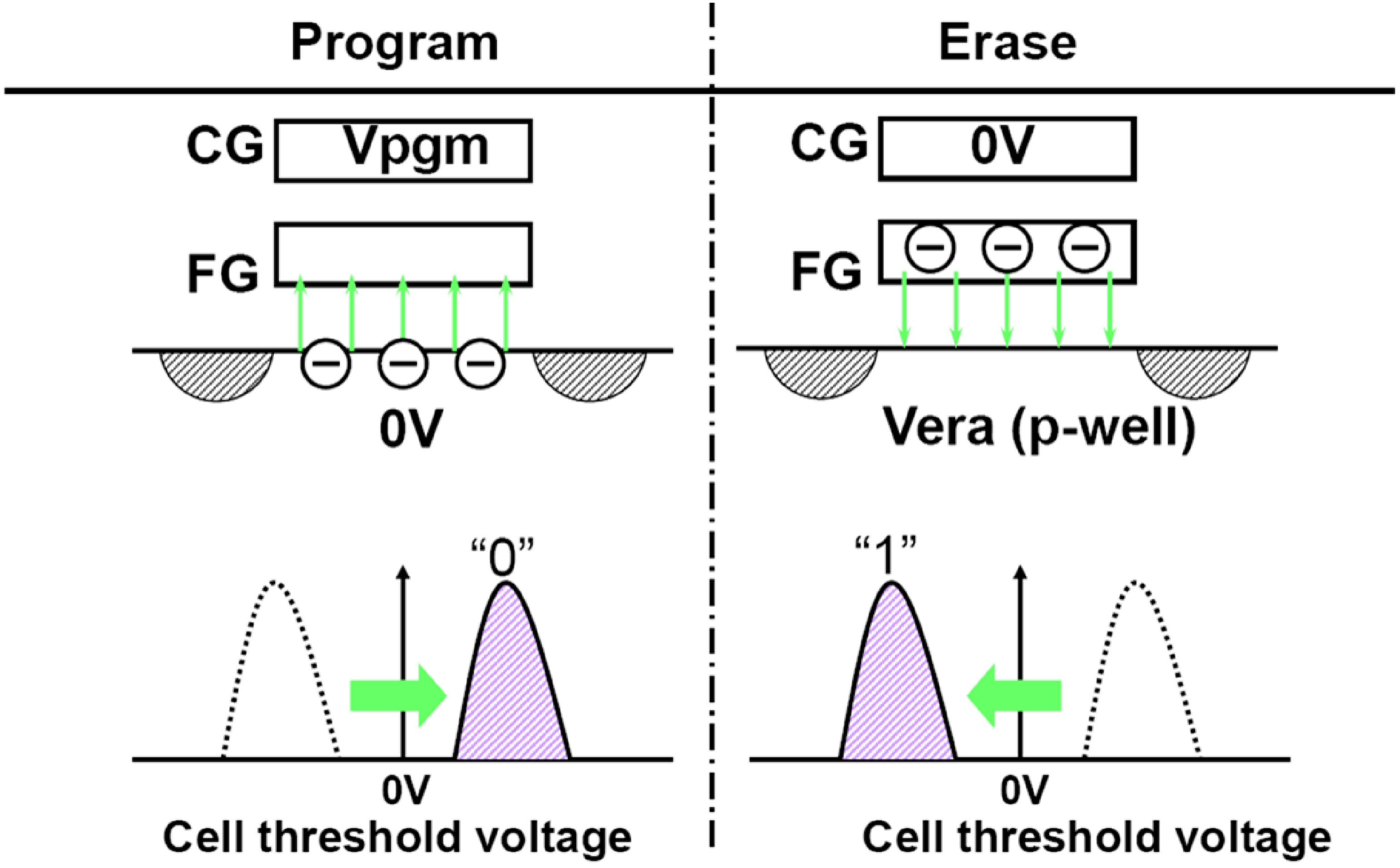
MOSFET



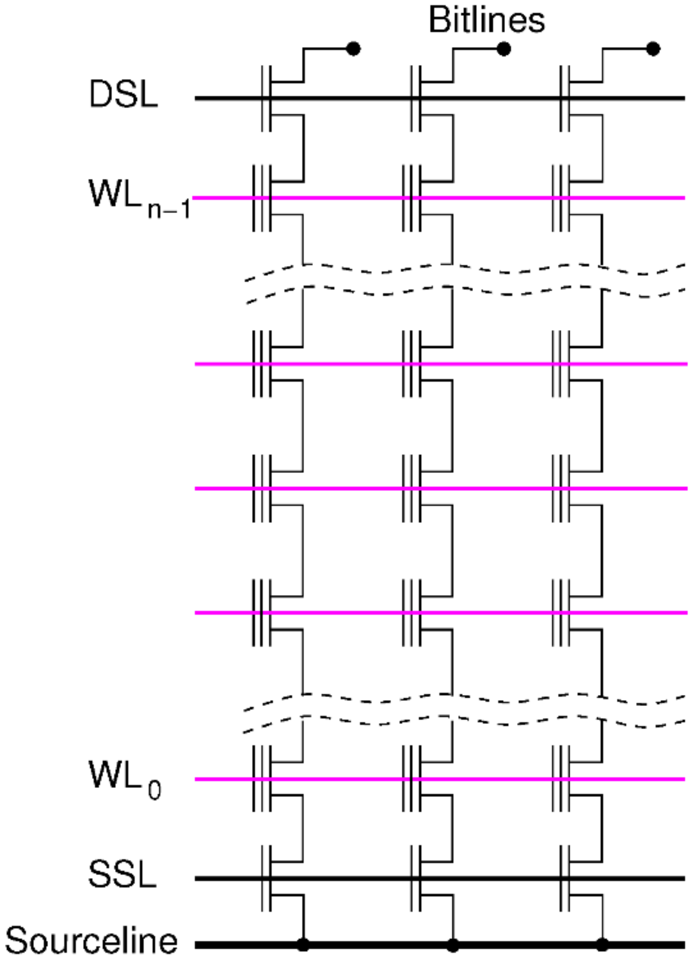


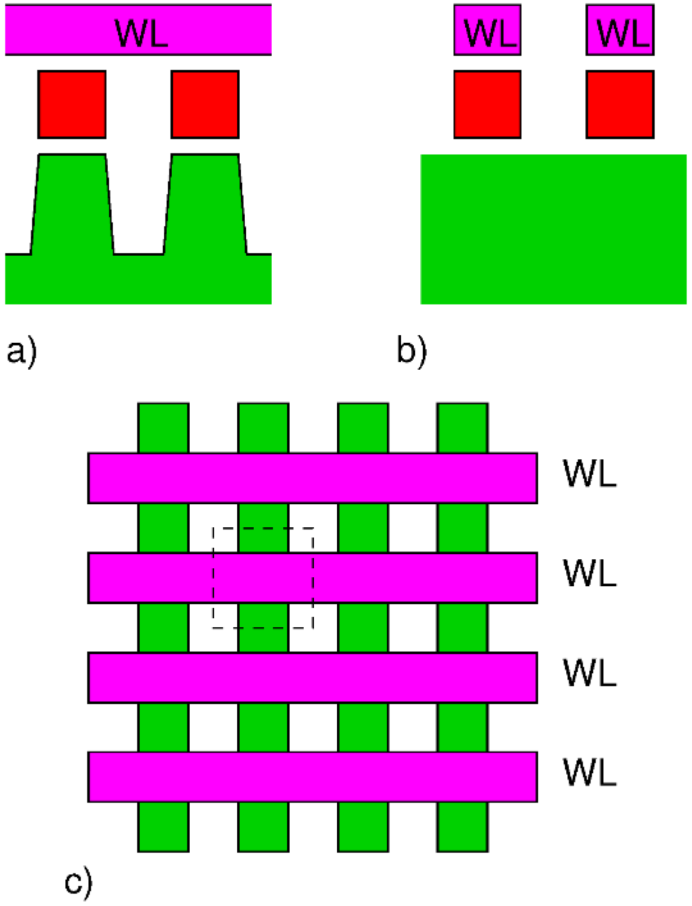


Principle of NAND Memory Programming and Erasing



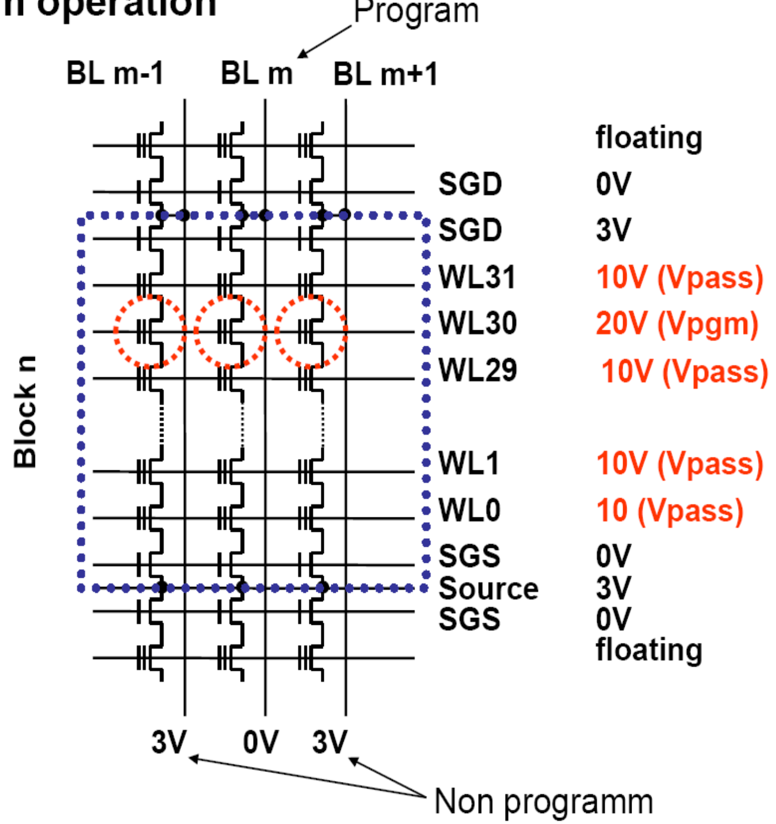
NAND Flash Array



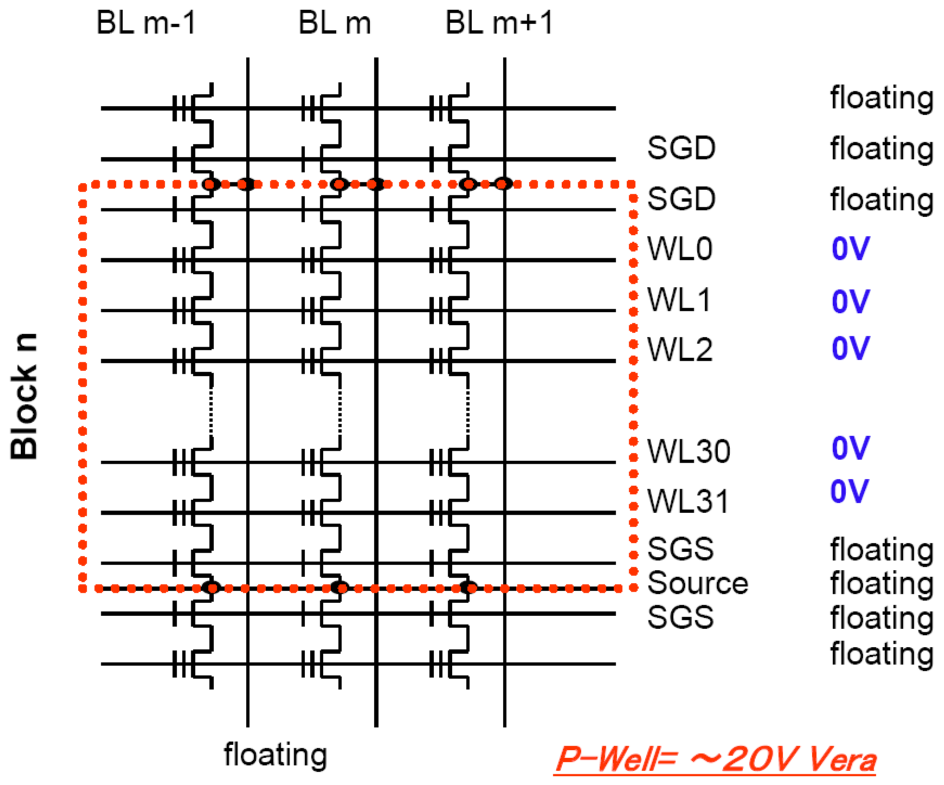


Basic Operation of NAND Flash- P/E

Program operation

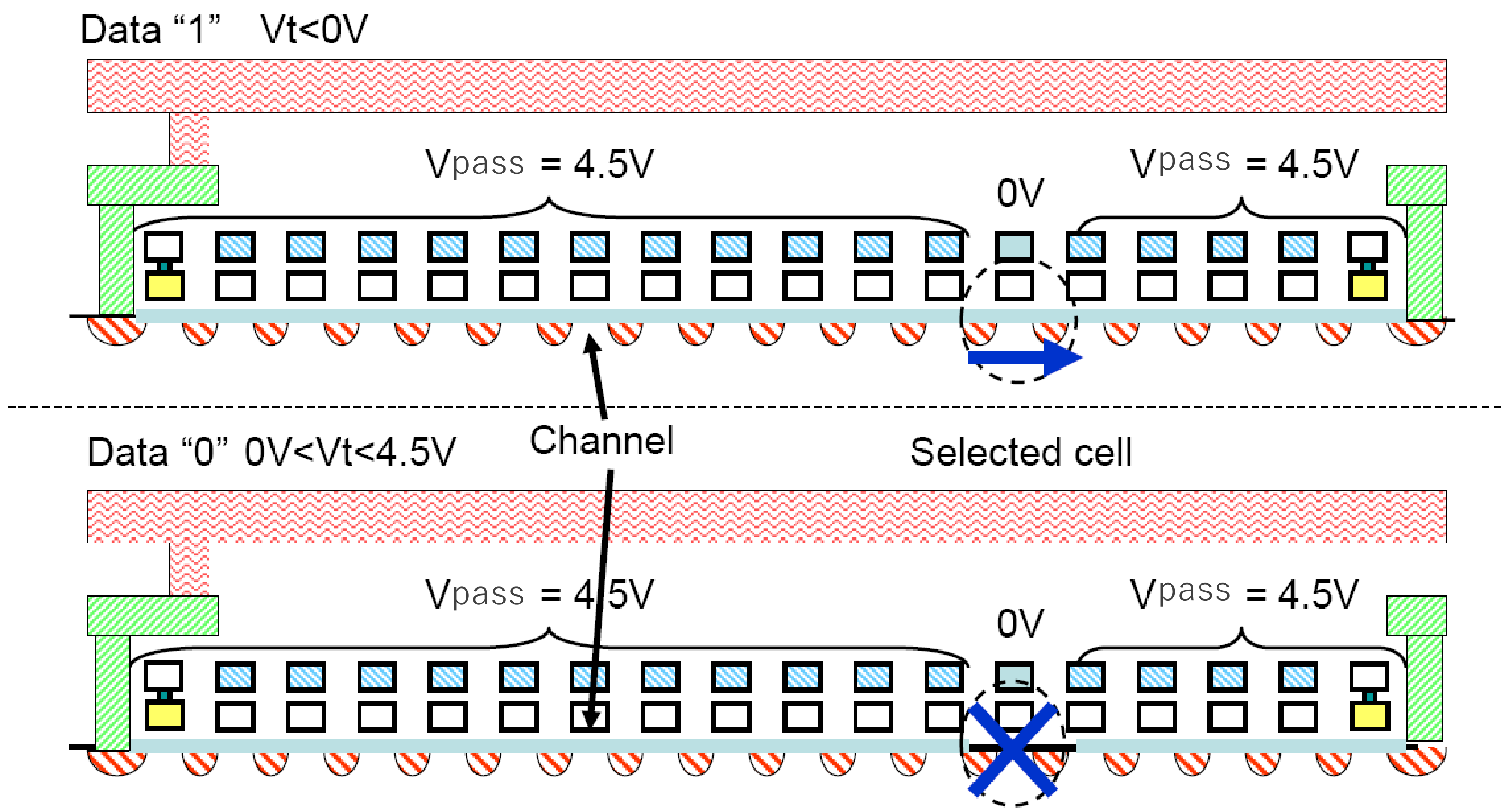


Erase operation (Block erasing)



Basic Operation of NAND Flash- Read

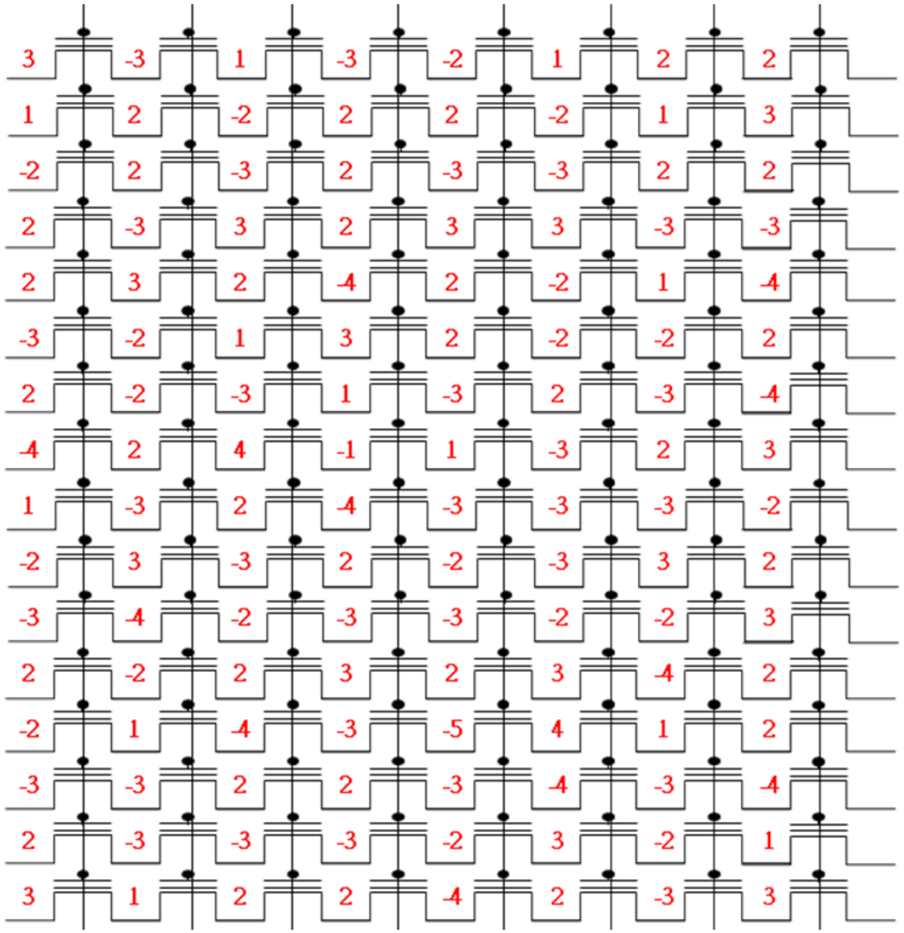
Read Operation



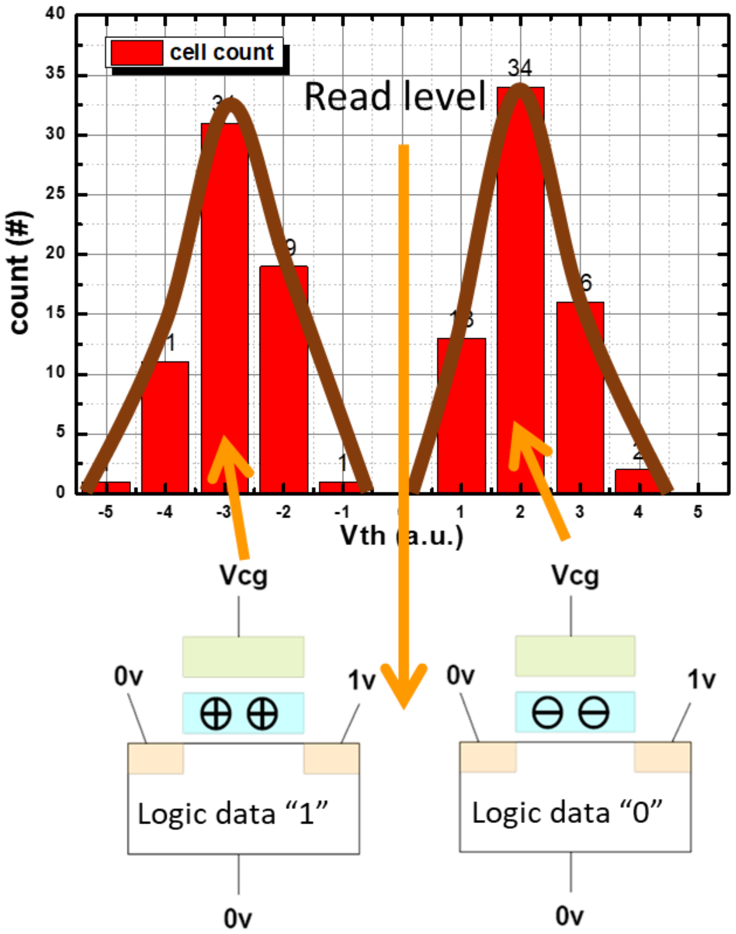
PHISON

Threshold Voltage Distribution

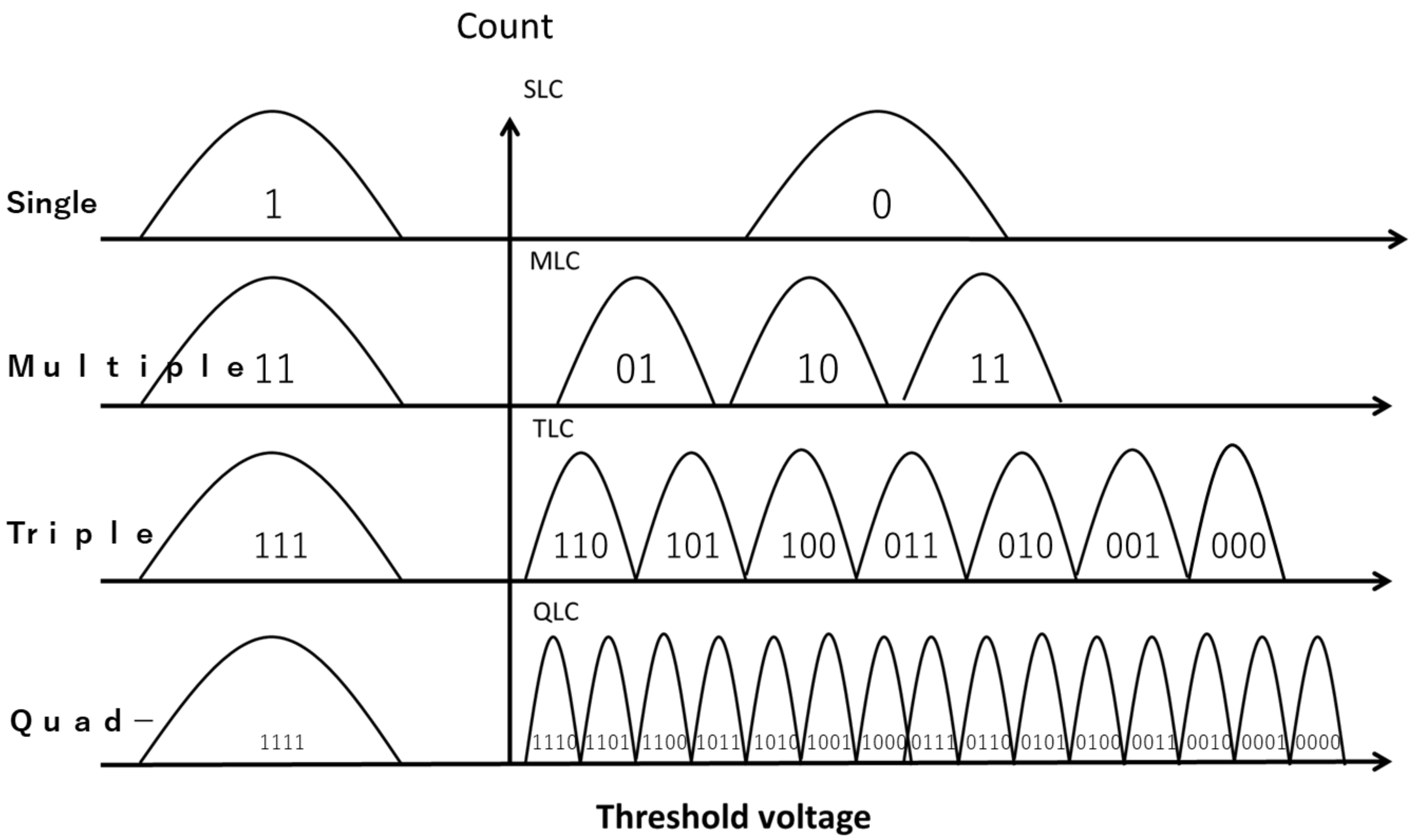
V_{th} of a 128 bit NAND Array:



V_{th} distribution of the array:



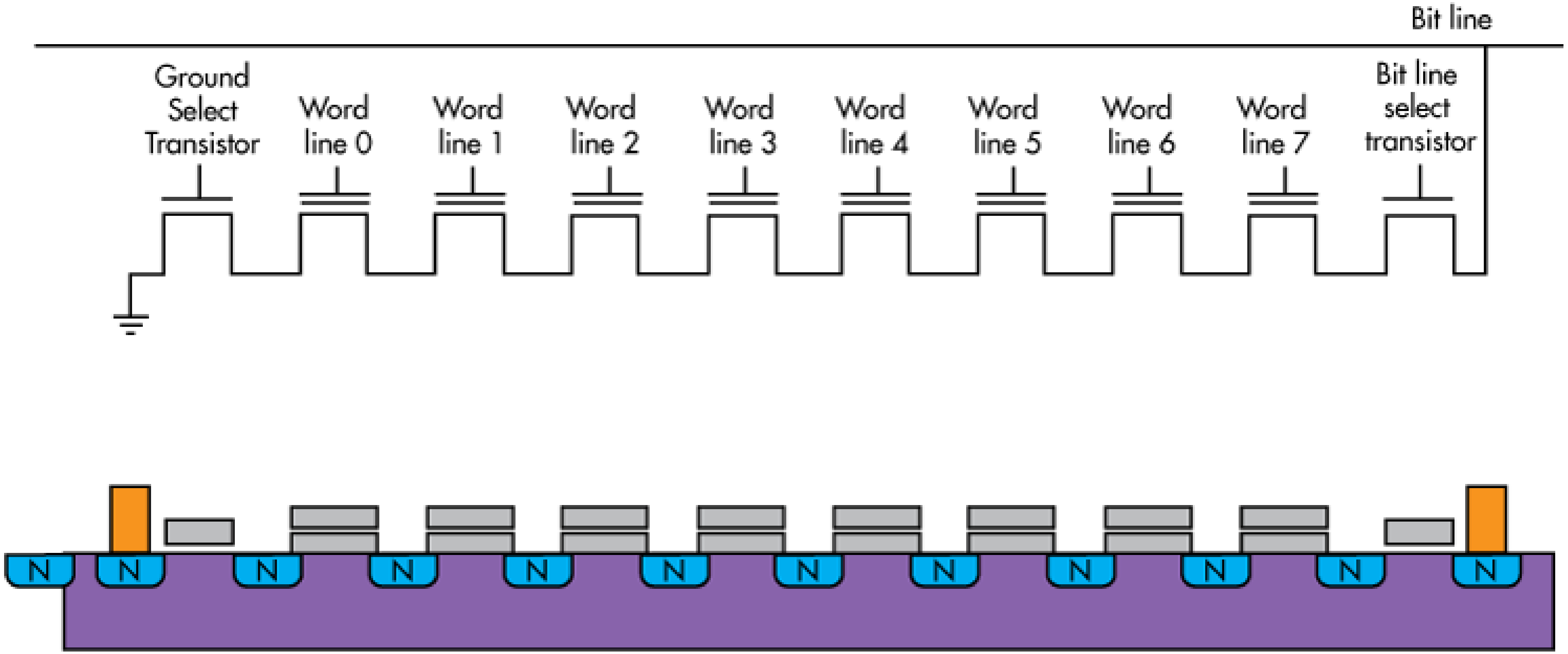
XLC Application



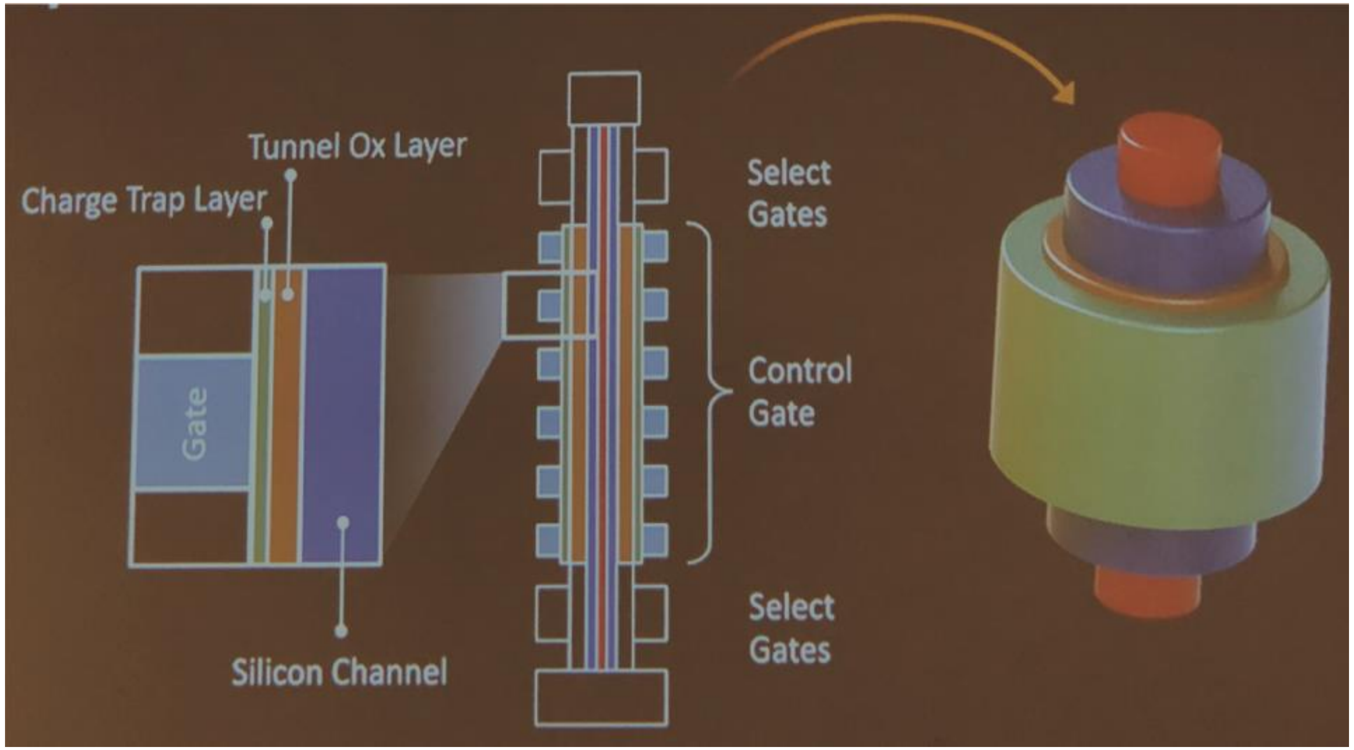
Question

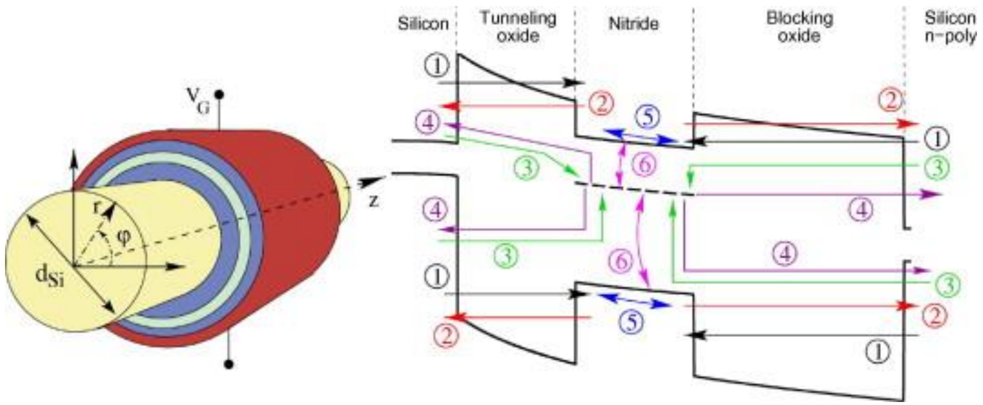
Why do we need XLC?

2D (Planar) NAND Flash



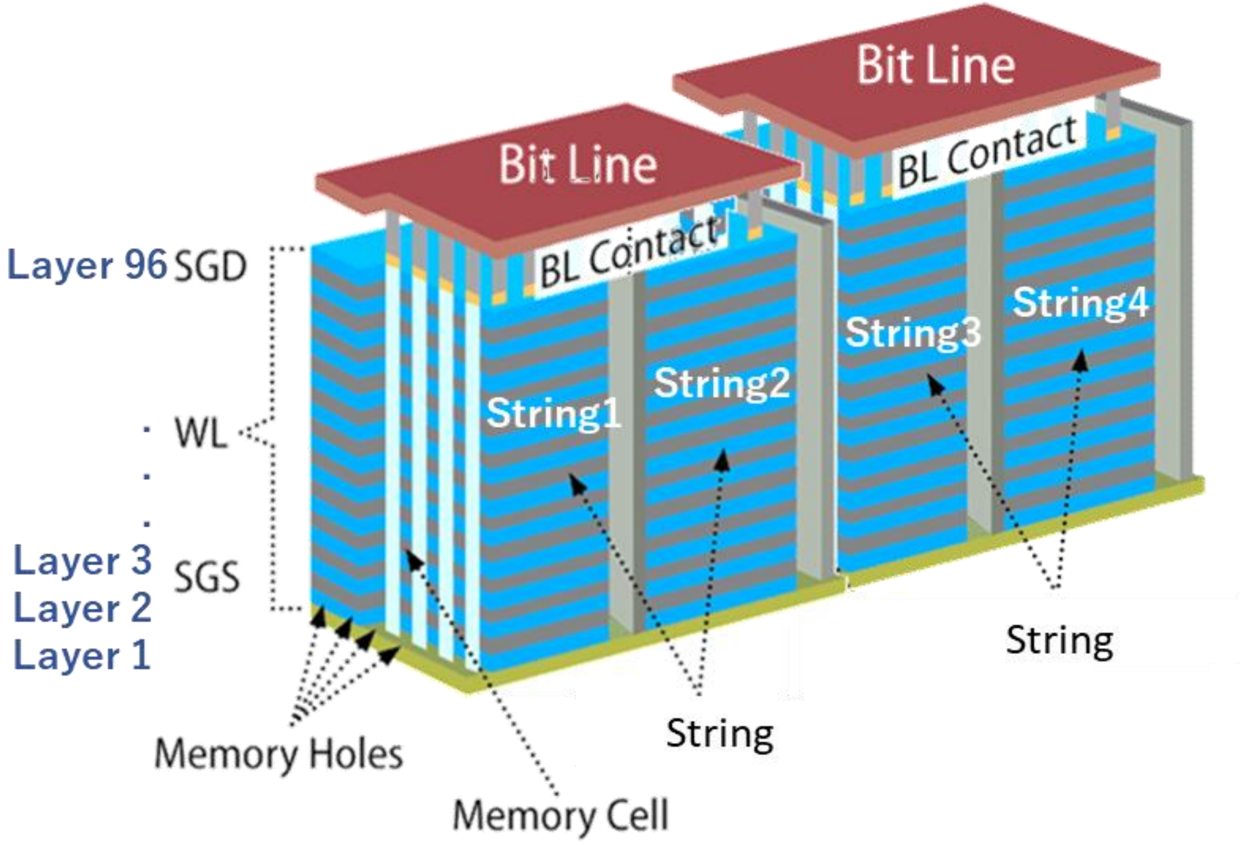
3D NAND Flash



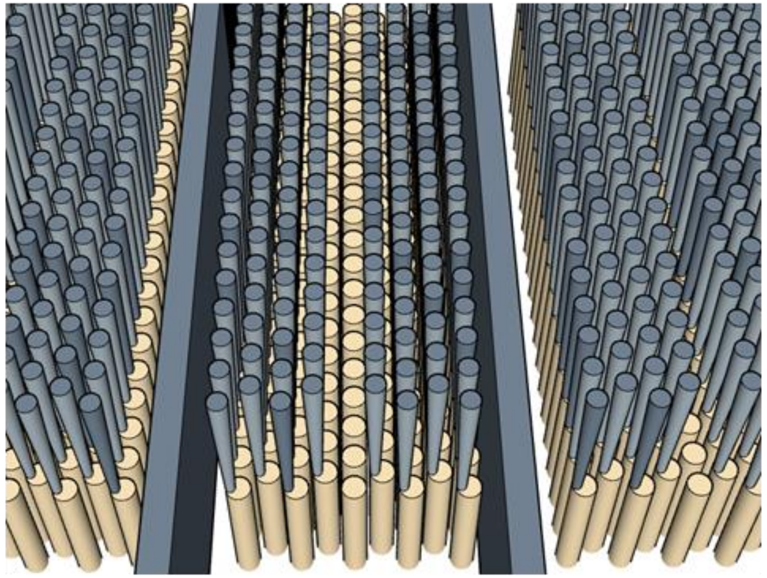


3D NAND Array

3D NAND Architecture



◆ Top View



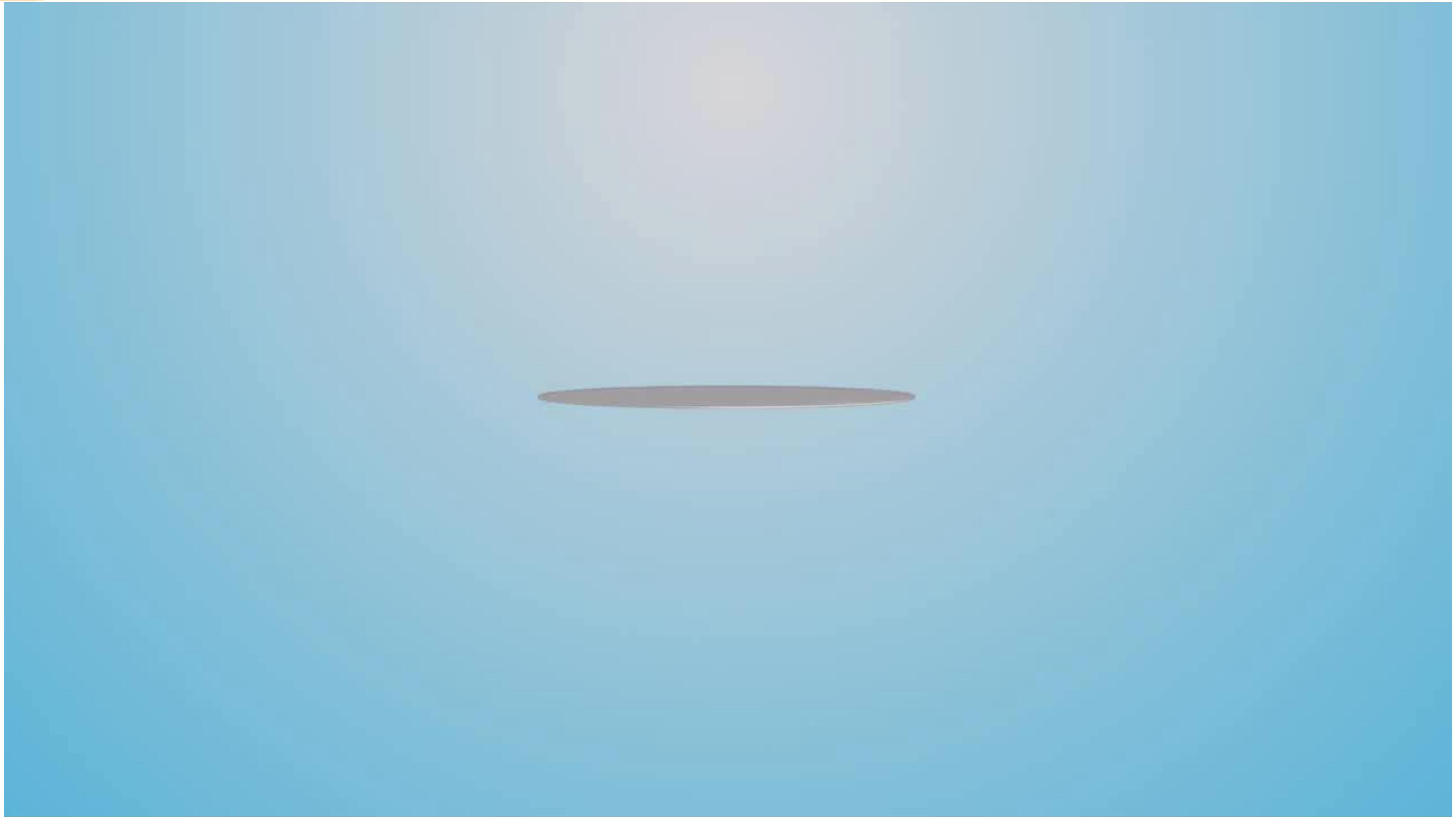
Page Size = 16K Byte (16*1024*8)

1 WL = 4 strings =4 Pages

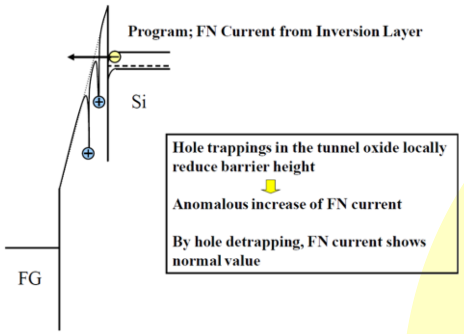
1Block = 4 * 96 WLs = 384 Pages (SLC)

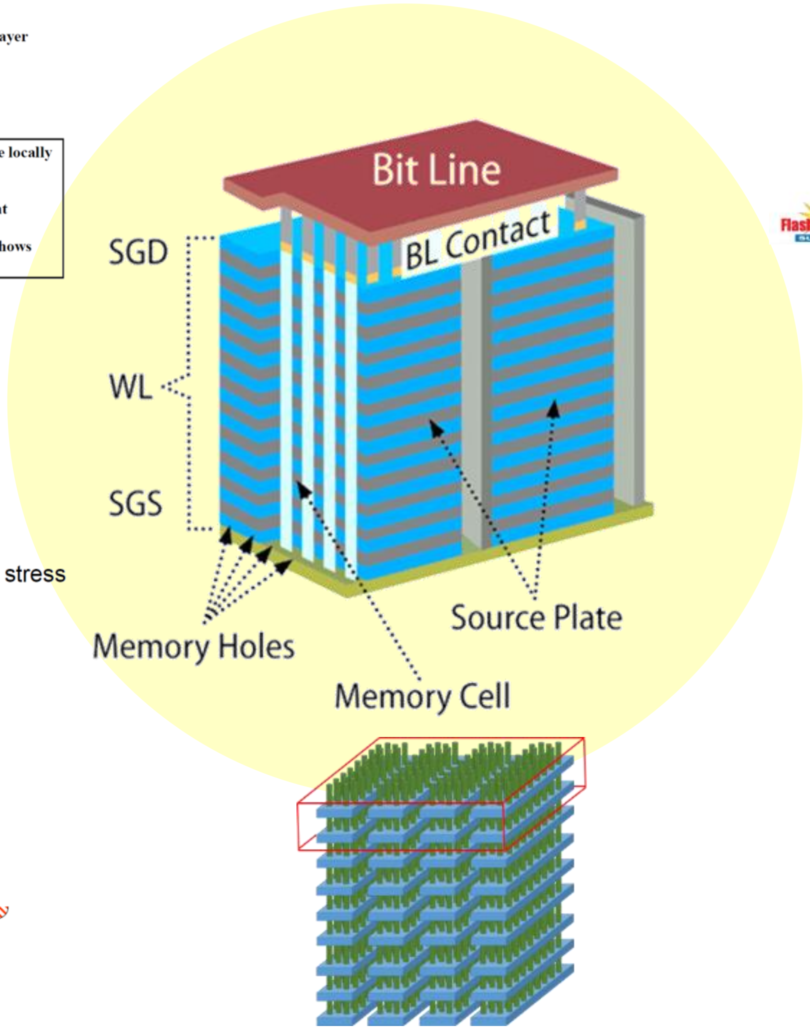
PHISON

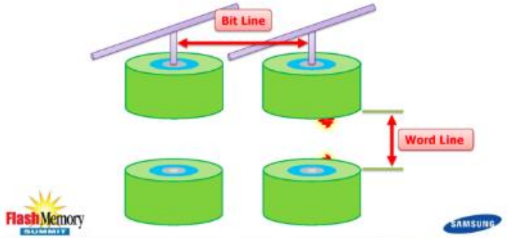
Xtacking

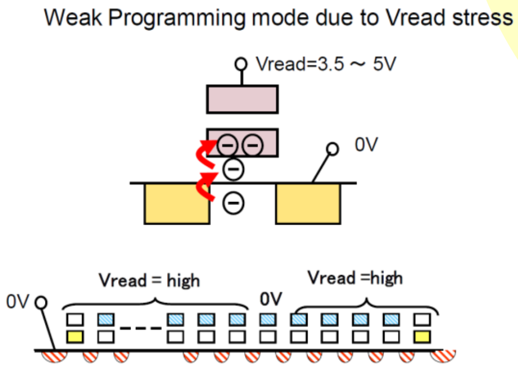


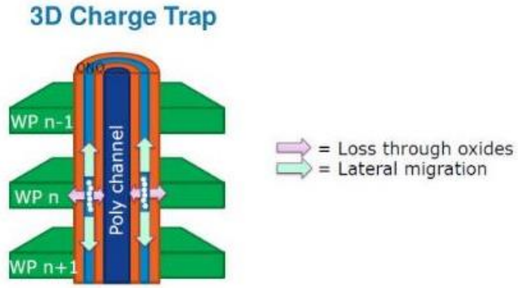
Reliability Issues of NAND Flash











Source: G. Van Den Bosch, IMW 2014

Question

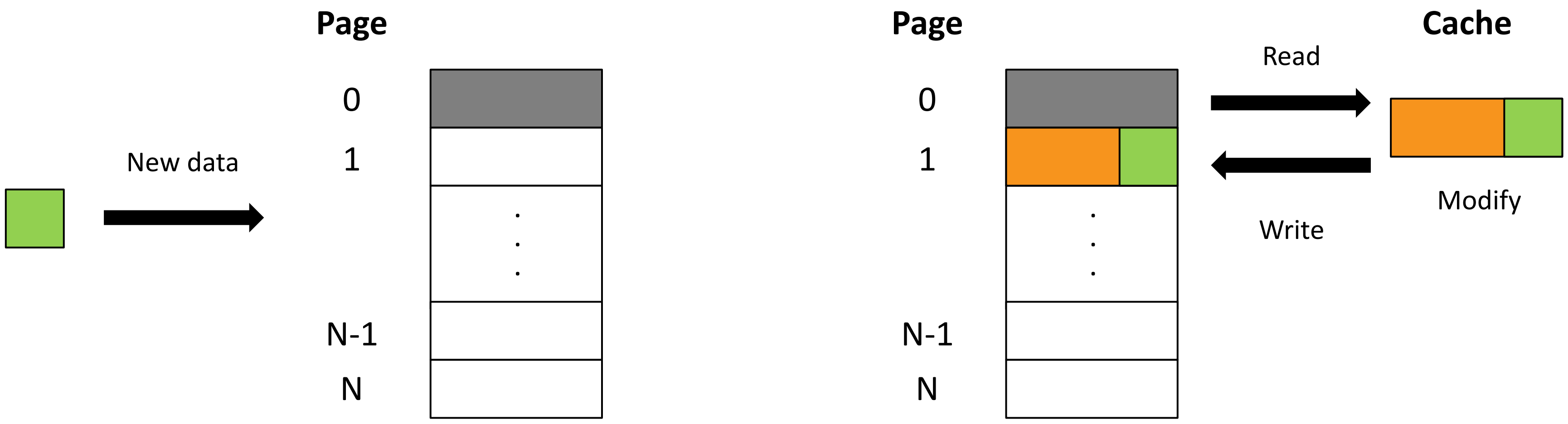
Why do we need NAND Flash Controller?

What does NAND Flash Controller need to do?

NAND Flash Limitations

- **Page** is the smallest unit for read and program
- **Block** is the smallest unit for erase
- Must erase before program (cannot overwrite)

Mapping Table & Garbage Collection



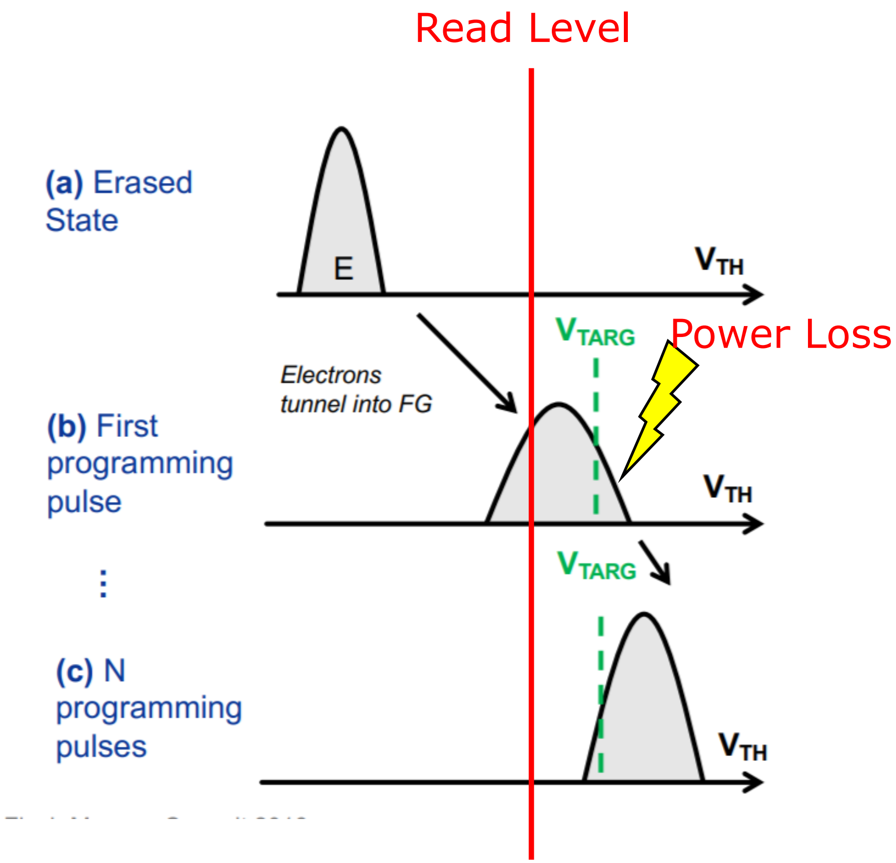
- Each read/program/erase operation has busy time to complete

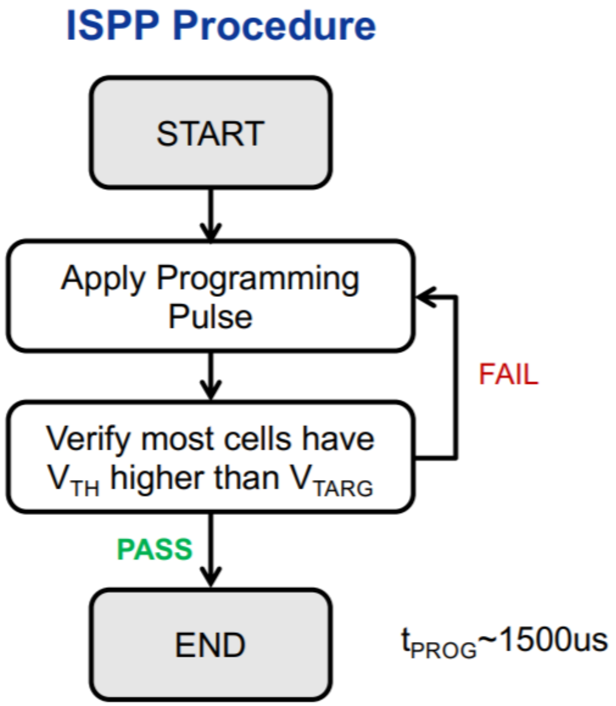
I/O scheduling

NAND Flash Limitations

- What if sudden power loss happens before VT reaches programmed state?

Power Loss Rebuild



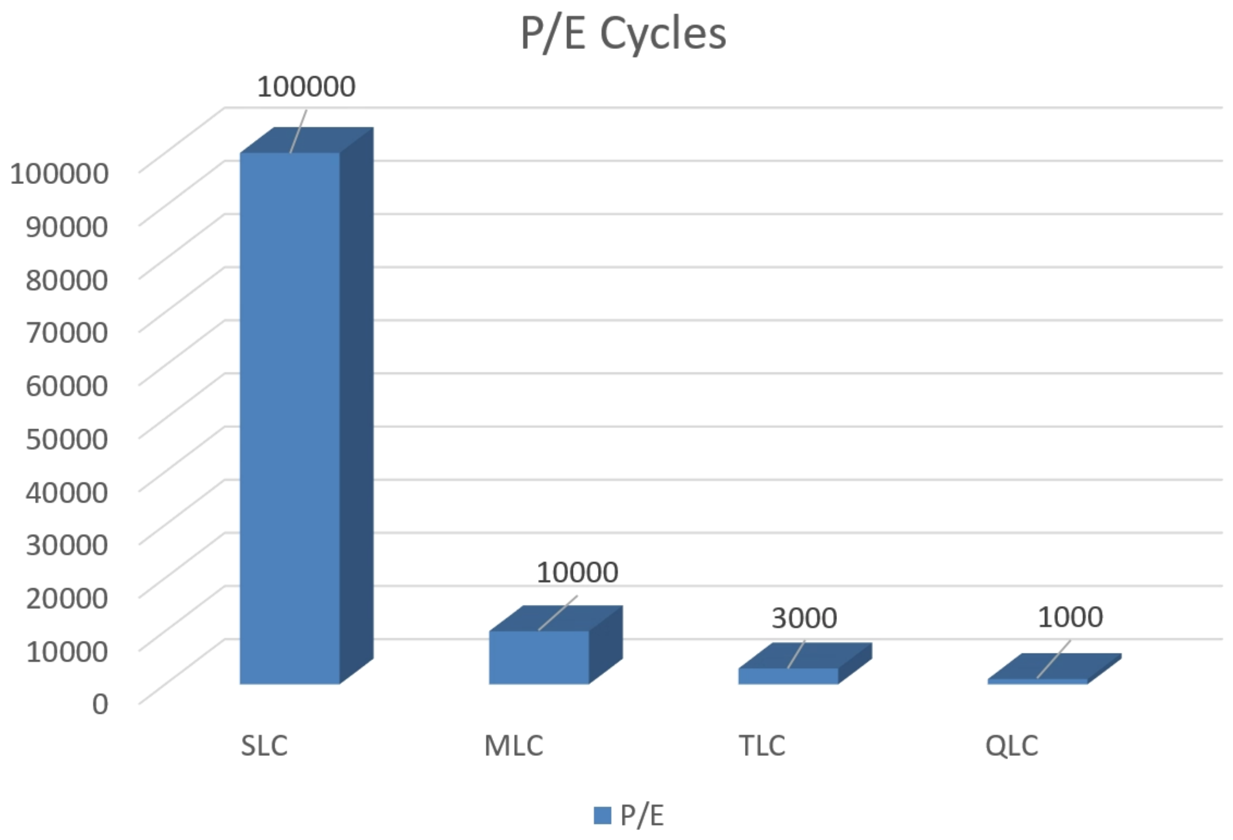


NAND Flash Limitations

- Program/Erase (PE) Cycle: SLC = 100k, MLC = 10k, TLC = 3k
- Initial and runtime bad blocks

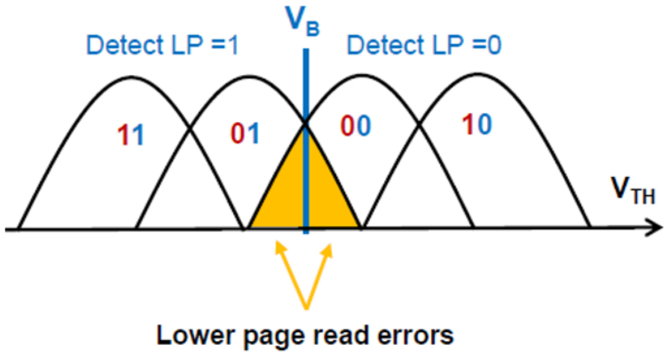
Wear Leveling

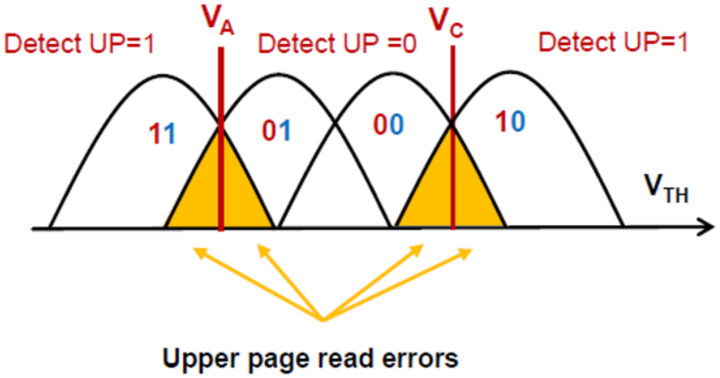
Bad Block Management



NAND Flash Limitations

- Main sources of noise that would cause abnormal Vth distribution and read errors:
  - **Program/Erase cycling stress**: Program/erase pulses lead to degraded reliability of the underlying NAND flash cells.
  - **Cell-to-cell interference**: Threshold voltage of 'victim' cell is strongly affected by programming of neighboring 'aggressor' cells.
  - **Data retention**: Over time, electrons can escape from the programmed flash cells, causing a loss of threshold voltage.
  - **Read disturbance**: When reading a particular page in a block of NAND flash, a voltage is applied to all other WL in order to 'deselect' them. This applied voltage can affect the Vth distribution of the unselected WLs.

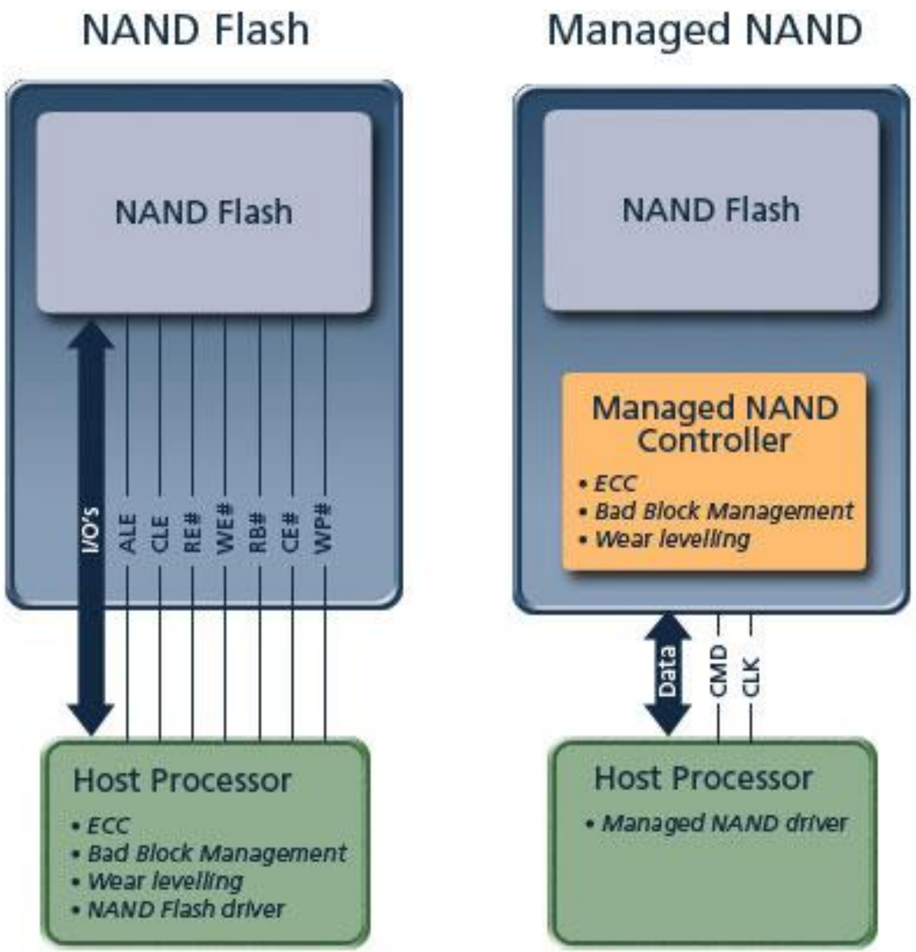




Error correction code &
Read disturb management

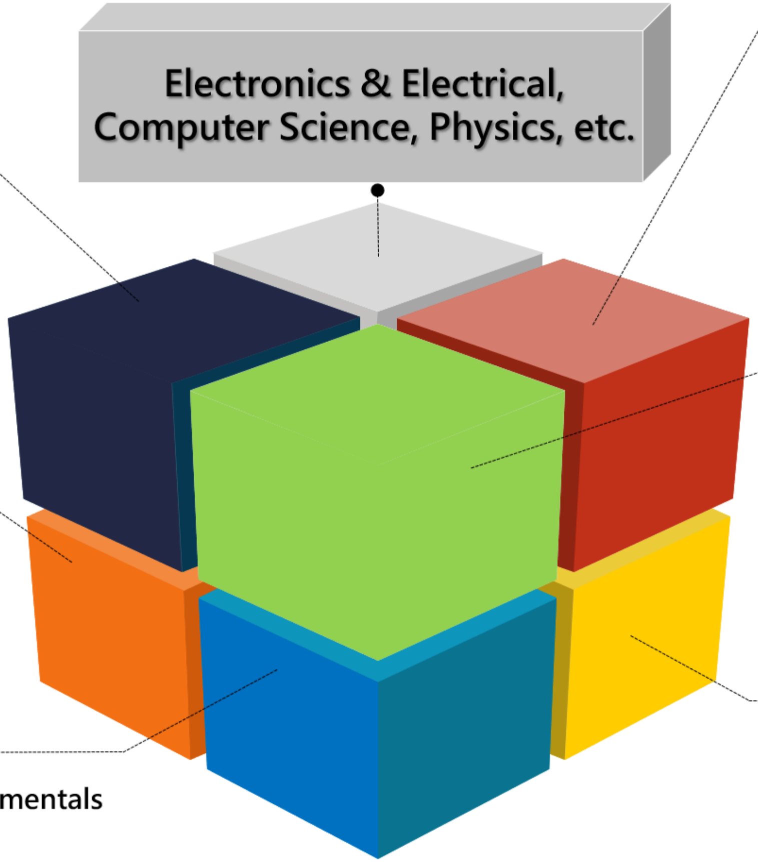
PHISON

NAND Flash Controller



PHISON

Key Departments in Controller House



Firmware Engineer
- ✓ Skills :
  - C/C++
  - Python
  - Computer Architecture
- ✓ Firmware development
- ✓ FTL Algorithm design
- ✓ RMA Analysis

Software Engineer
- ✓ Skills :
  - C/C++
  - Visual C++
  - Python
- ✓ V&V pattern design

Field Application Engineer
- ✓ Skills :
  - Electronics & Electrical fundamentals
- ✓ Circuitry design
- ✓ Customer support

Analog IC Engineer
- ✓ Skills :
  - Analog IC Design
- ✓ Power analog circuit design
- ✓ Analog/Mixed signal design
- ✓ High Speed IO/transceiver circuit designs

Digital IC Engineer
- ✓ Skills :
  - FPGA、Verilog
  - Digital Systems
  - Design & Simulation Tools
- ✓ IP design & validation
- ✓ CP/FT

NAND Flash Engineer
- ✓ Skills :
  - C/C++
  - Semiconductor Device Physics
- ✓ Error correction algorithm
- ✓ NAND Flash characteristic analysis

Block Base Mapping

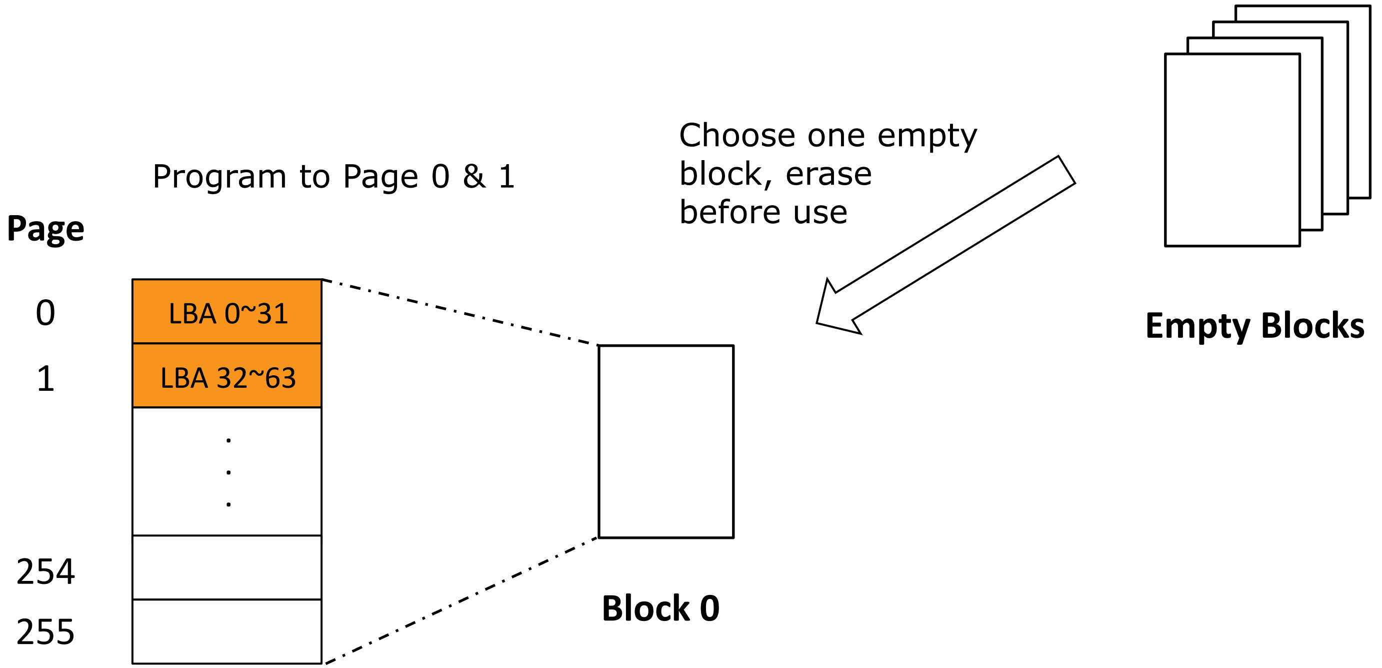
Definitions & Rules

- 8GB NAND Flash
  - 1 die = 2048 blocks
  - 1 block = 256 pages
  - 1 page = 16KB = 32 sectors
  - 1 LBA/sector = 512B

- Limitation
  - **Page** is the smallest unit for read and program
  - **Block** is the smallest unit for erase
  - Must erase before program (cannot overwrite)

Write

- Host writes 2 chunk of data (16KB each)
  - LBA 0~31
  - LBA 32~63

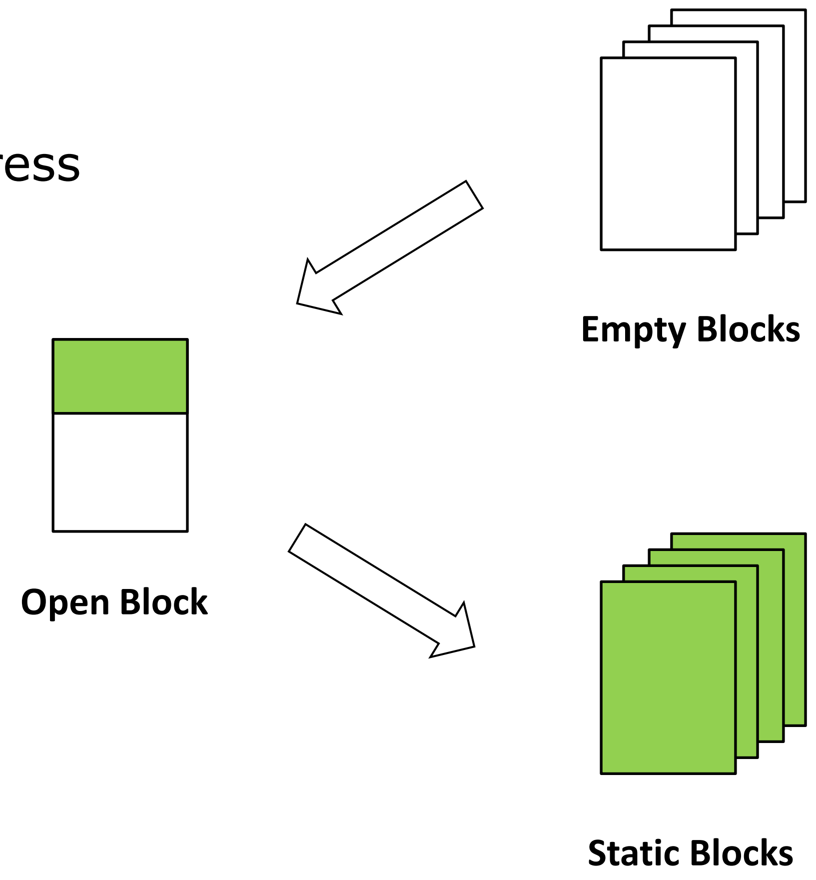


Write

Q: How do we build mapping table?

- Host continues to write large chunk size data…
- Open block becomes static when full
- We record logical to physical block mapping
- Each Logical Block (LB) index maps to Block (PB) address

LB	PB
0	0
1	1
	. . .
2046	0xFFFF
2047	0xFFFF



- Address = (PB)
- Might need 2B to store address

Read

- Host wants to read LBA 32~63
- How do we know where is the data located in NAND Flash?

Read

Q: What is the formula to calculate offset?

- Logical to Physical (L2P) translation

  Step 1: Calculate Logical Block (LB) Index

$$LB = \frac{LBA}{\text{Total Physical Sector Number (in block)}}$$

  Step 2: Get Physical Block (PB) address from mapping table

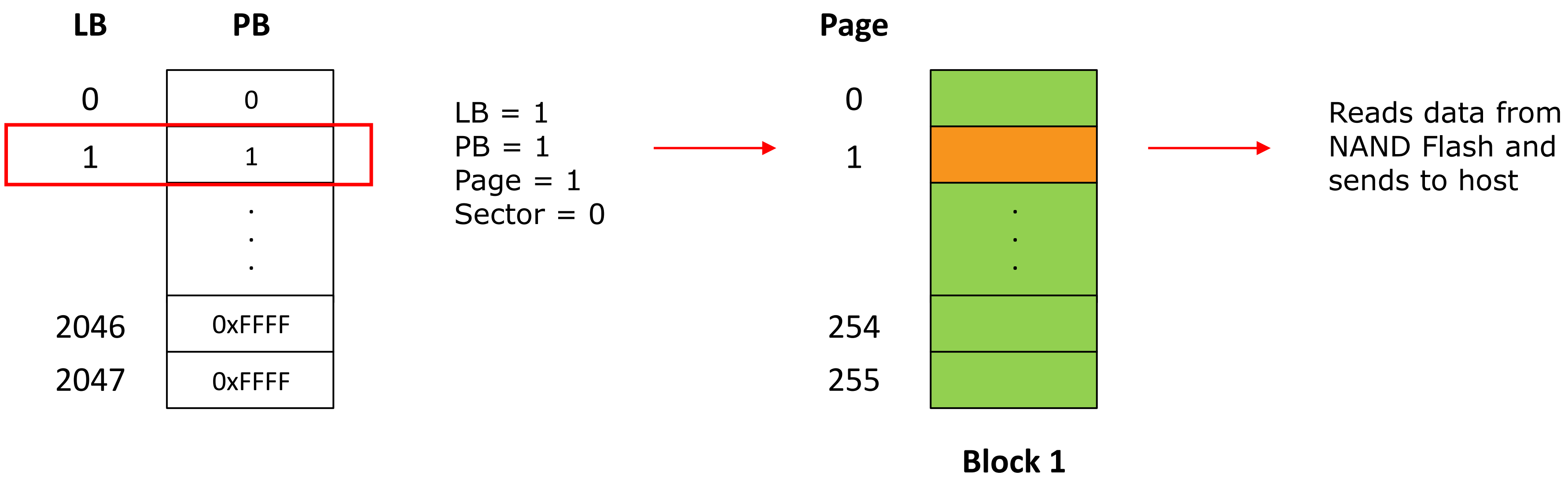
  Step 3: Calculate Offset

$$Page = \frac{LBA \,\%\, \text{Total Physical Sector Number (in block)}}{\text{Total Physical Sector Number (in page)}}$$

$$Sector = LBA \,\%\, \text{Total Physical Sector Number (in block)} \,\%\, \text{Total Physical Sector Number (in page)}$$

Read

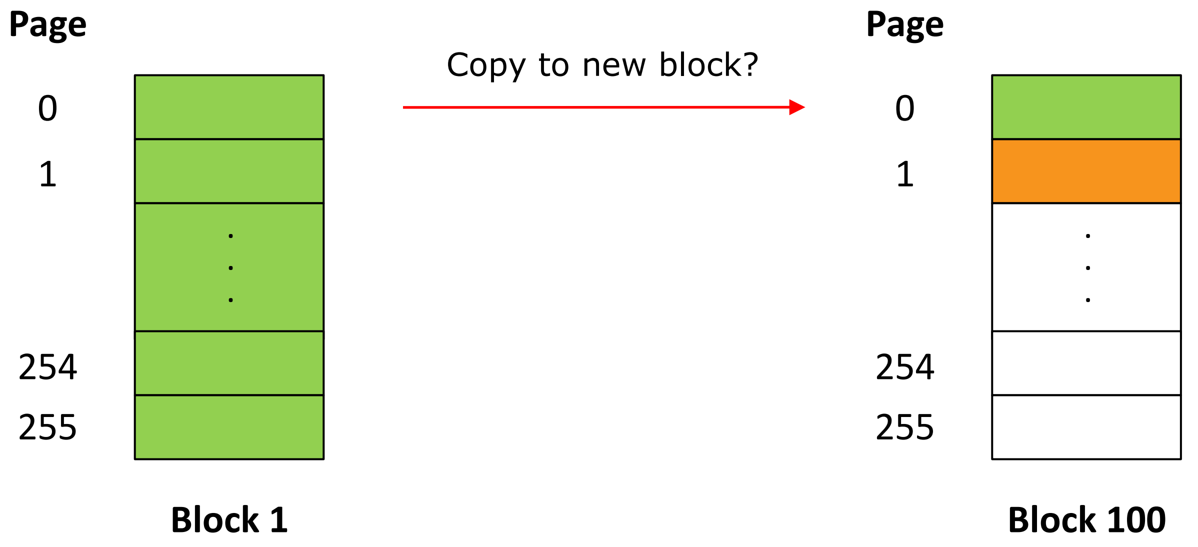
- Host wants to read LBA 8224~8255



Overwrite

Q: How do we handle overwrite page?

- What if host overwrites same address LBA 8224~8255?
- Remember, we cannot overwrite a page that has been programmed



- Not so efficient

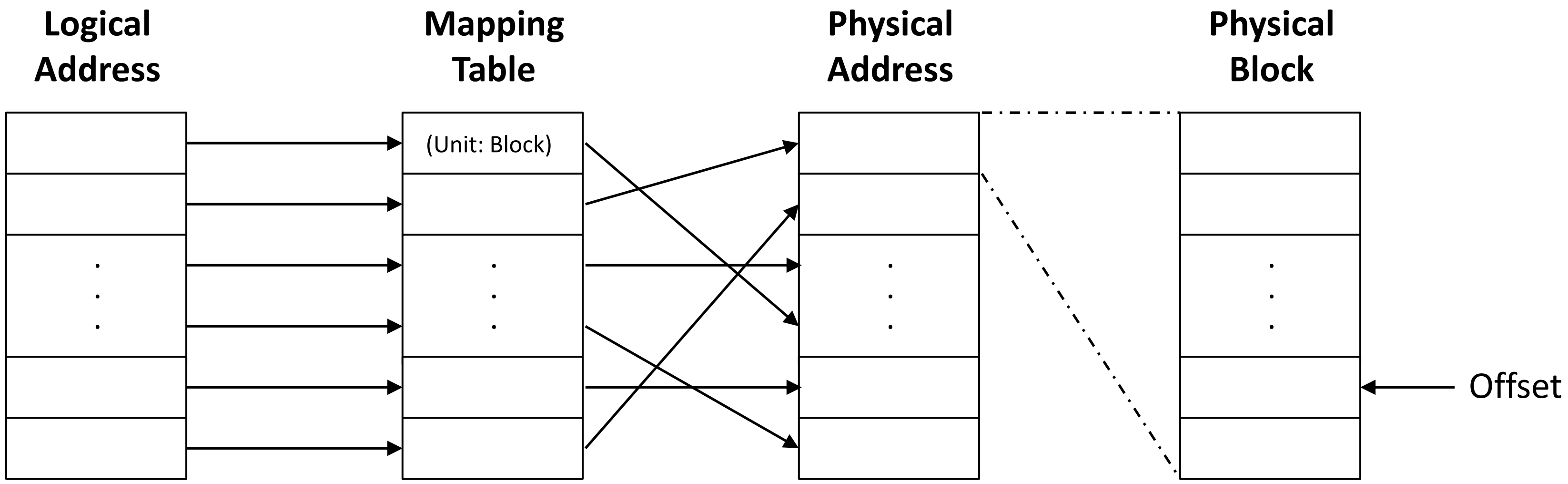
Table Size (for Block Mapping)

- For 8GB NAND Flash, total table size required is 4KB
  - Table Size = 2048 blocks × 2B = 4KB
  - Table Entries = 2048

LB	PB
0	0
1	1
	. . .
2046	0xFFFF
2047	0xFFFF

Block Base Mapping Table

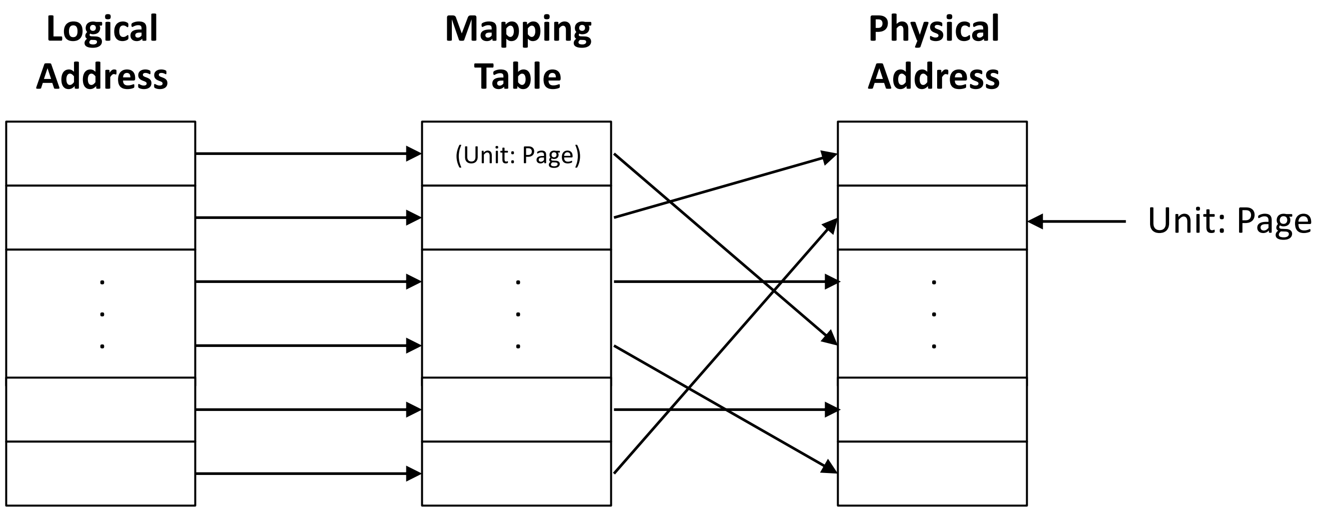
- In a block level address mapping, a logical page address is made up of both a **logical block number** and an **offset**



Page Base Mapping

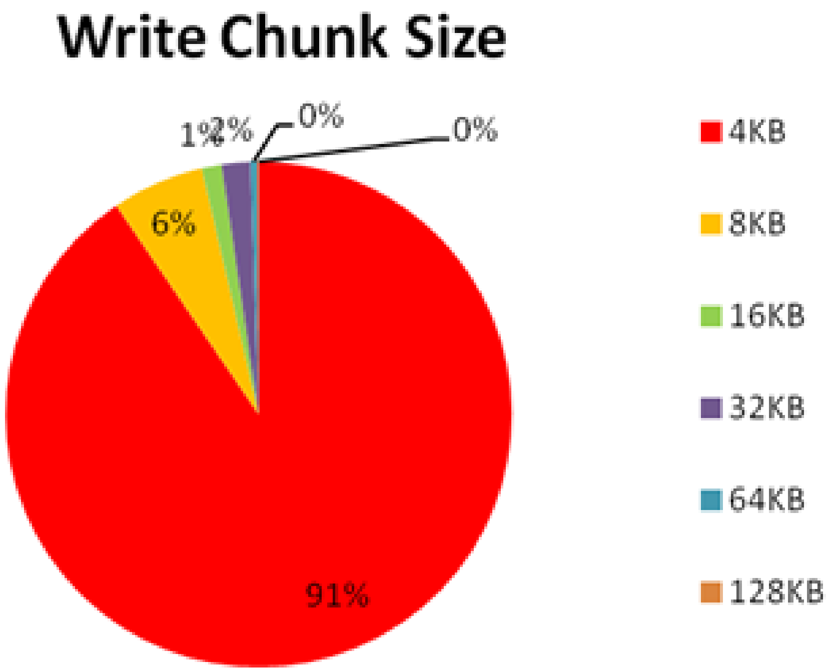
Page Base Mapping Table

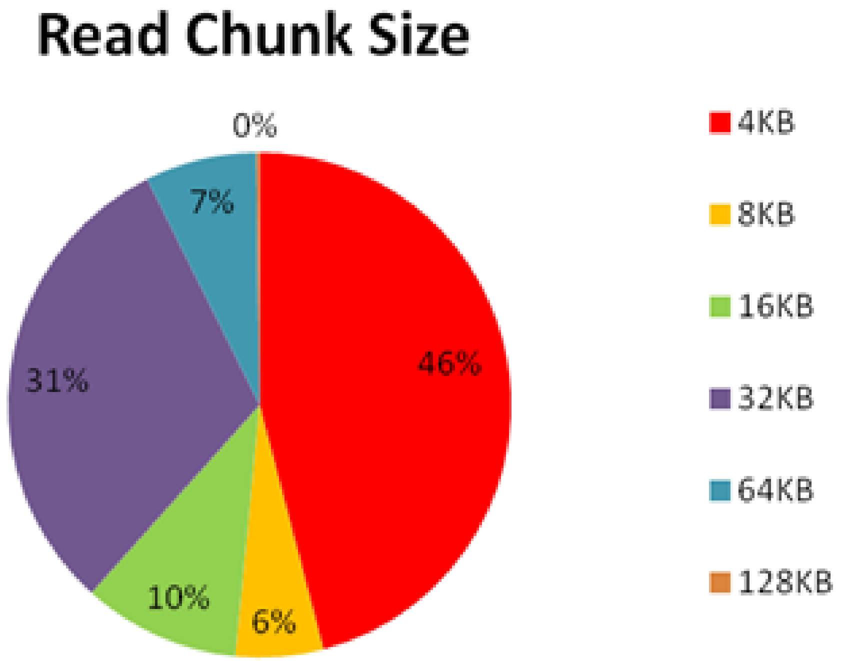
- In a page level address mapping, a logical page can be **mapped into any physical page** in flash memory



Page Base Mapping Table

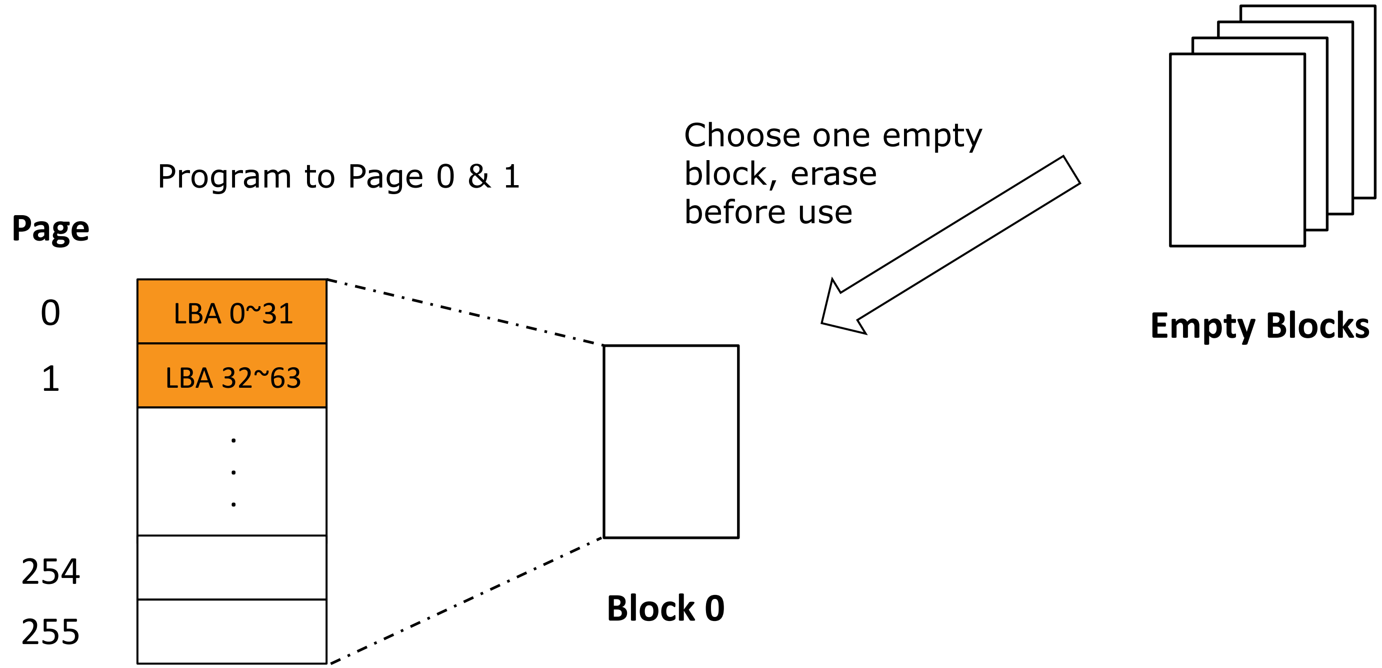
- Data structure of host (eMMC) operations on WHCK performance test
- 4KB chunk size has higher percentage than other chunk size





Write

- Host writes 2 chunk of data (16KB each)
  - LBA 0~31
  - LBA 32~63



Write

Q: How do we record page base mapping?

- Instead of block mapping, we now record logical to physical page mapping
- Each Logical Page (LP) index maps to Physical Page (PP) address

LP	PP
0	0, 0, 0
1	0, 1, 0
	. . .
542286	0xFFFFFFFF
542287	0xFFFFFFFF

- Address = (Block, Page, Sector)
- Might need 4B (instead of 2B) to store address

Write

- Host writes additional 2 chunk of data (16KB each)
  - LBA 64~95
  - LBA 32~63 (overwrite)

Page	Block 0
0	LBA 0~31
1	LBA 32~63
2	LBA 64~95
3	LBA 32~63
	. . .
254	
255	

No need to copy data to new block during overwrite, old data becomes invalid

LP	PP
0	0, 0, 0
1	0, 3, 0
2	0, 2, 0
3	0xFFFFFFFF
	. . .
542286	0xFFFFFFFF
542287	0xFFFFFFFF

Read

- Host wants to read LBA 32~63
- How do we know where is the data located in NAND Flash?

Read

Q: What is the formula to calculate Logical Page?

- Logical to Physical (L2P) translation

  Step 1: Calculate Logical Page (LP) Index

$$LP = \frac{LBA}{Total\ Physical\ Sector\ Number\ (in\ page)}$$

  Step 2: Get Physical Page (PP) address from mapping table

Read

- Host wants to read LBA 32~63

LP	PP
0	0, 0, 0
1	0, 3, 0
2	0, 2, 0
3	0xFFFFFFFF
	. . .
542286	0xFFFFFFFF
542287	0xFFFFFFFF

LP = 1
Block = 0
Page = 3
Sector = 0

Page	Block 0
0	LBA 0~31
1	LBA 32~63
2	LBA 64~95
3	LBA 32~63
	. . .
254	
255	

Reads data from NAND Flash and sends to host

Erase

Q: How do we erase LBA?

- Host wants to erase LBA 64~95

LP	PP
0	0, 0, 0
1	0, 3, 0
2	0xFFFFFFFF
3	0xFFFFFFFF
	. . .
542286	0xFFFFFFFF
542287	0xFFFFFFFF

LP = 2
Block = 0
Page = 2
Sector = 0

Page	Block 0
0	LBA 0~31
1	LBA 32~63
2	LBA 64~95
3	LBA 32~63
	. . .
254	
255	

Block 0

- Mark corresponding LP as invalid

Table Size (for 16KB Page Mapping)

- For 8GB NAND Flash, total table size required is 2MB
  - Table Size = 2048 blocks × 256 pages × 4B = 2MB
  - Table Entries = 2048 blocks × 256 pages = 524288

LP	PP
0	0xFFFFFFFF
1	0xFFFFFFFF
	⋮
542286	0xFFFFFFFF
542287	0xFFFFFFFF

Table Size (for 4KB Page Mapping)

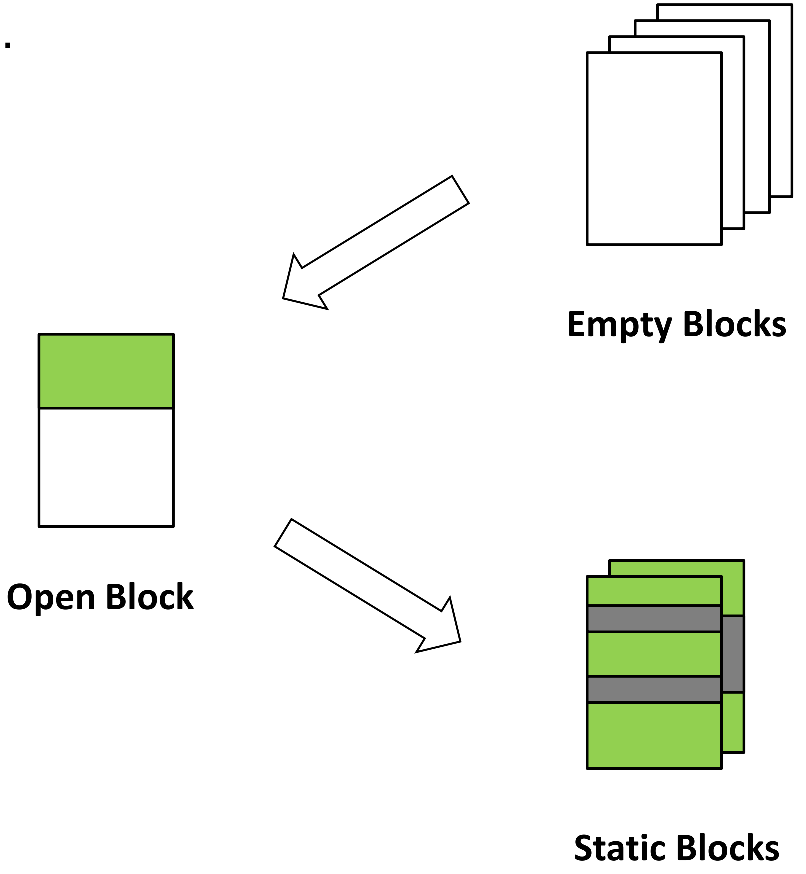
- For 8GB NAND Flash, total table size required is 8MB
  - Table Size = 2048 blocks × 256 pages × 4 nodes × 4B = 8MB
  - Table Entries = 2048 blocks × 256 pages × 4 nodes= 2097152

LP	PP
0	0xFFFFFFFF
1	0xFFFFFFFF
	⋮
2097150	0xFFFFFFFF
2097151	0xFFFFFFFF

Garbage Collection (GC)

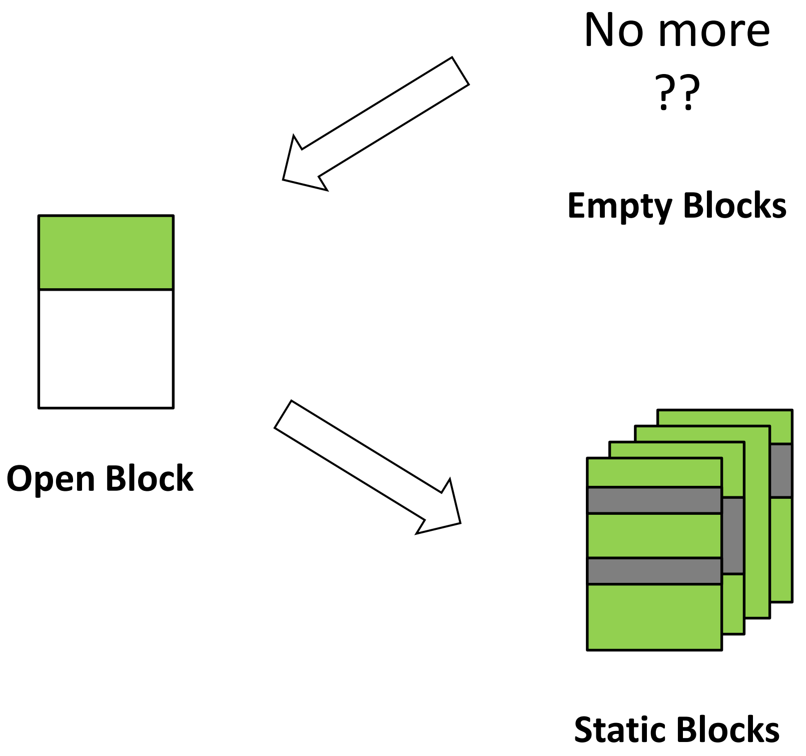
Garbage Collection

- What is GC and why do we need to do GC?
- Host continues to write large chunk size data…
- Open block becomes static when full
- Some physical page become invalid



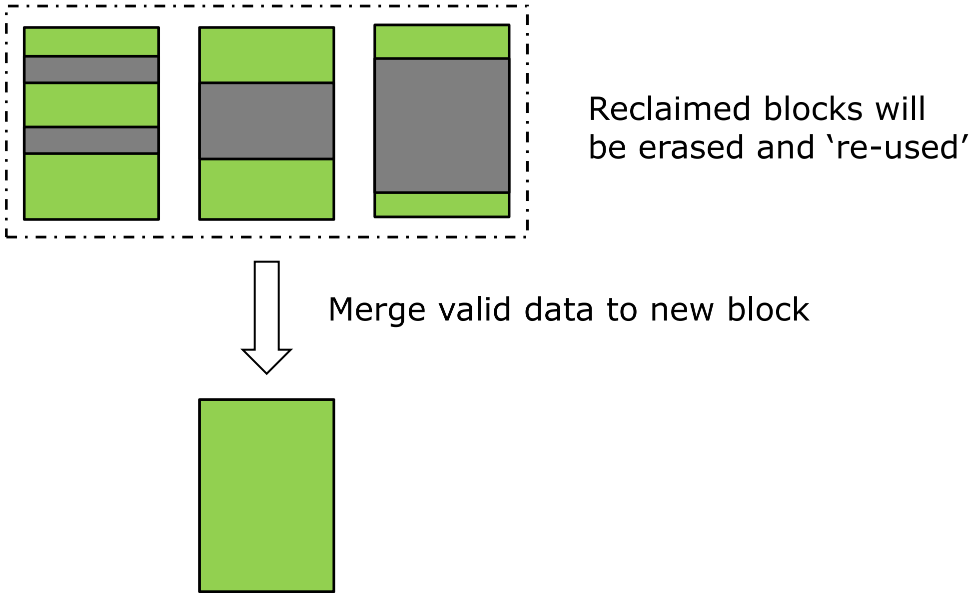
Garbage Collection

- What if host keeps writing until empty block runs out?
- Before empty block runs out, we should do Garbage Collection (GC)



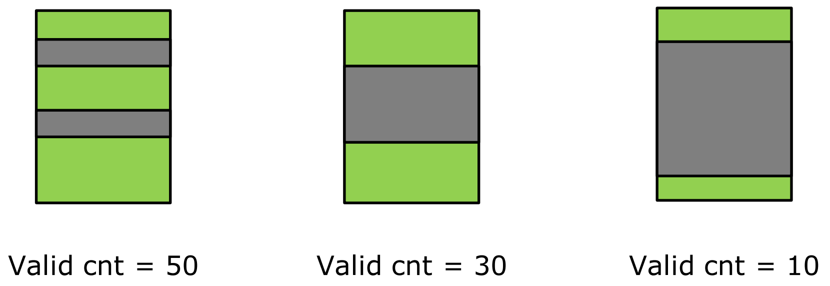
Garbage Collection

- We collect valid data to new block and reclaim blocks filled with invalid data
- So that we can erase the reclaimed blocks and use them for new data

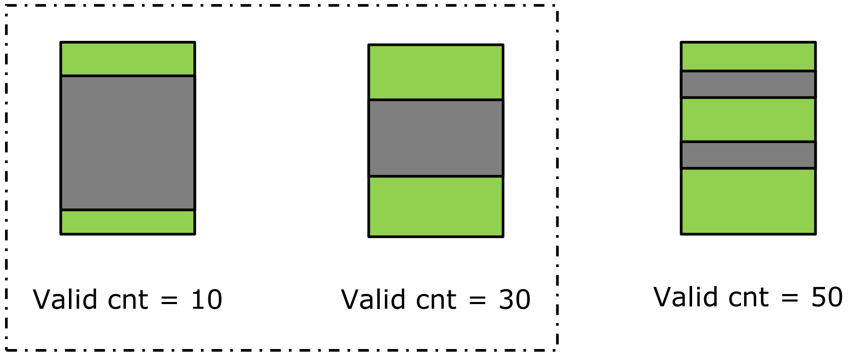


Garbage Collection

- How do we pick static blocks as source of GC?



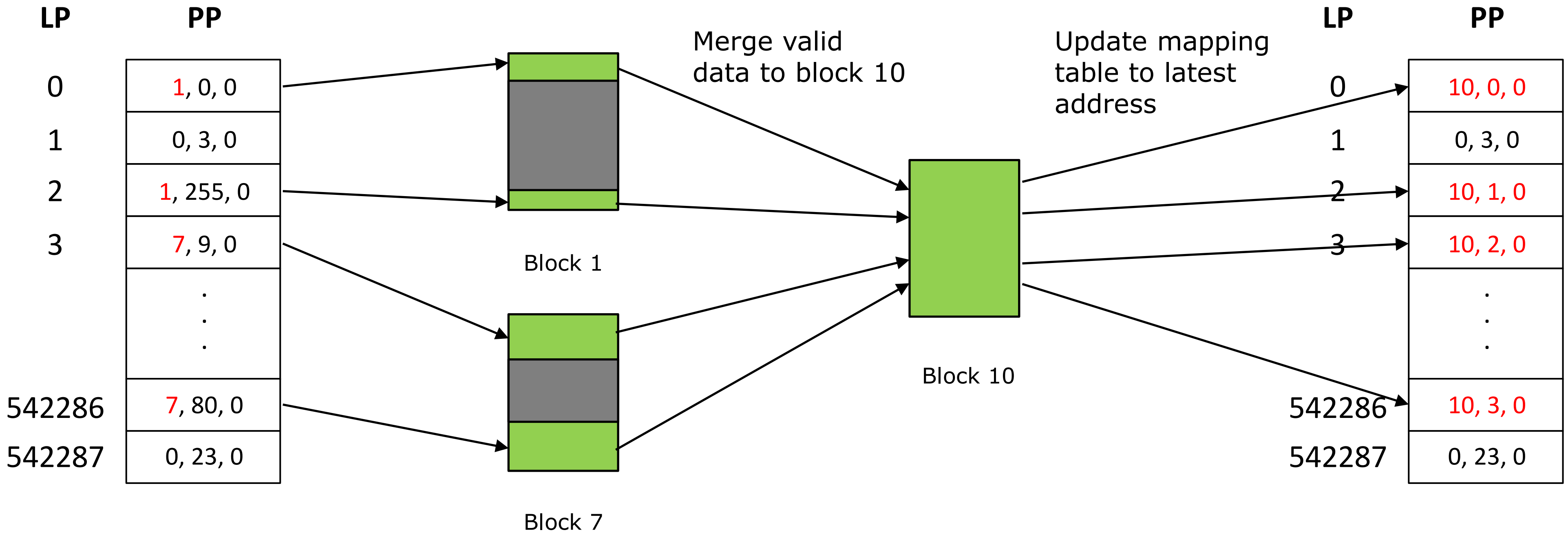
- Pick the block(s) with least valid count



Garbage Collection

Q: What should we update to complete GC?

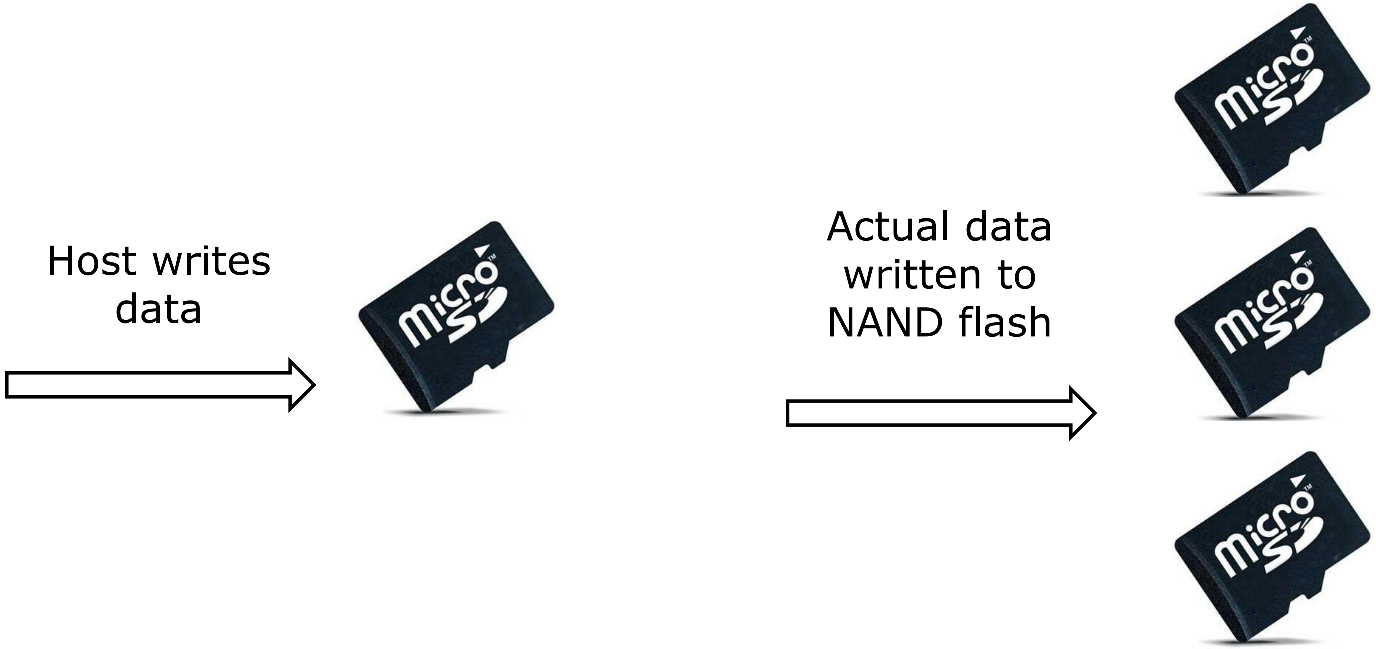
- How do we know which data is valid or invalid in the source blocks?
- Check the mapping table, look for the entries that point to these source blocks



Write Amplification Factor (WAF)

Definition

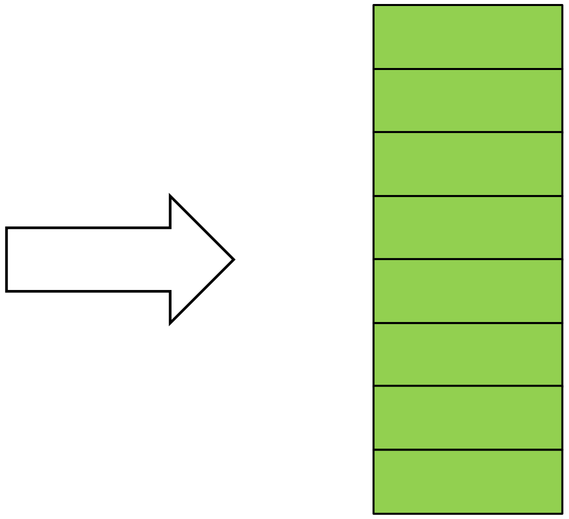
$$\text{Write Amplification Factor (WAF)} = \frac{\text{Data written to NAND Flash}}{\text{Data written by host}}$$



Example

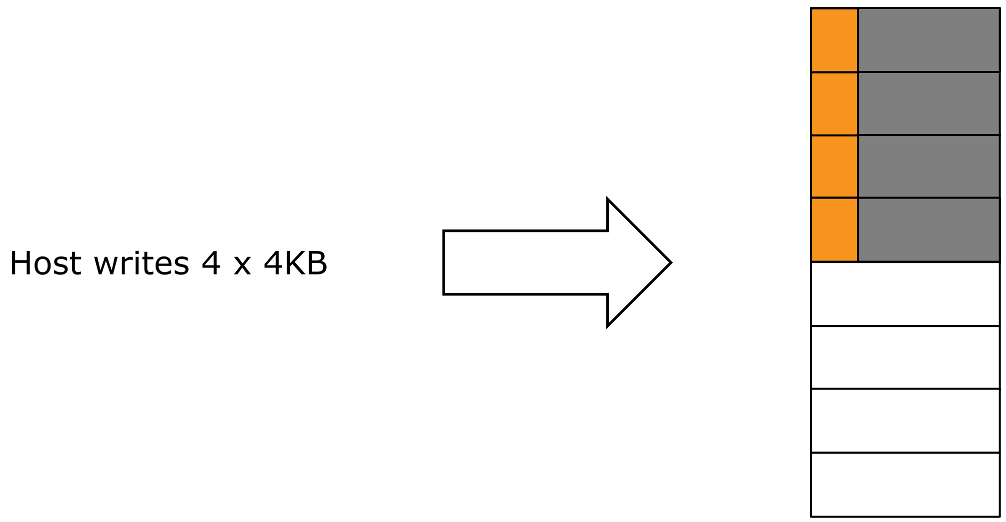
- WAF = 1

Host writes large sequential data



Example

- WAF = 4



- Still remember what happens when host overwrites data of same LBA using block level mapping?

Summary

- Disadvantages of high WAF
  - Low performance
  - High erase count

Terabytes Written (TBW)

Q: What is the formula to calculate TBW?

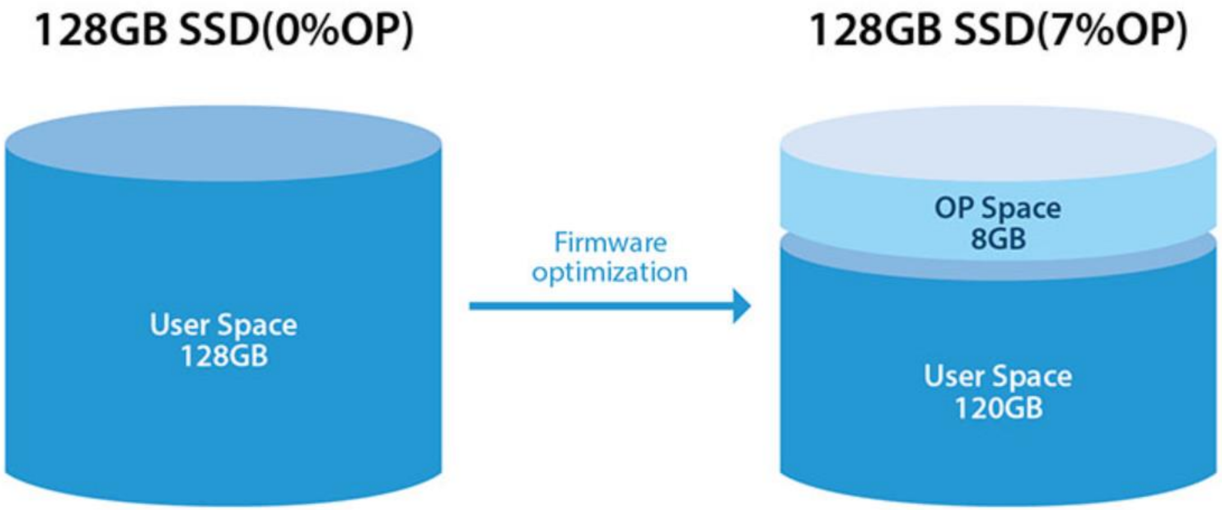
- Total amount of data that can be written to a storage device until it reaches its lifetime

$$\text{Terabytes Written (TBW)} = \frac{\text{User Capacity (GB)} \times \text{NAND P/E Cycles}}{\text{WAF} \times 1024}$$

Over Provision (OP)

Definition

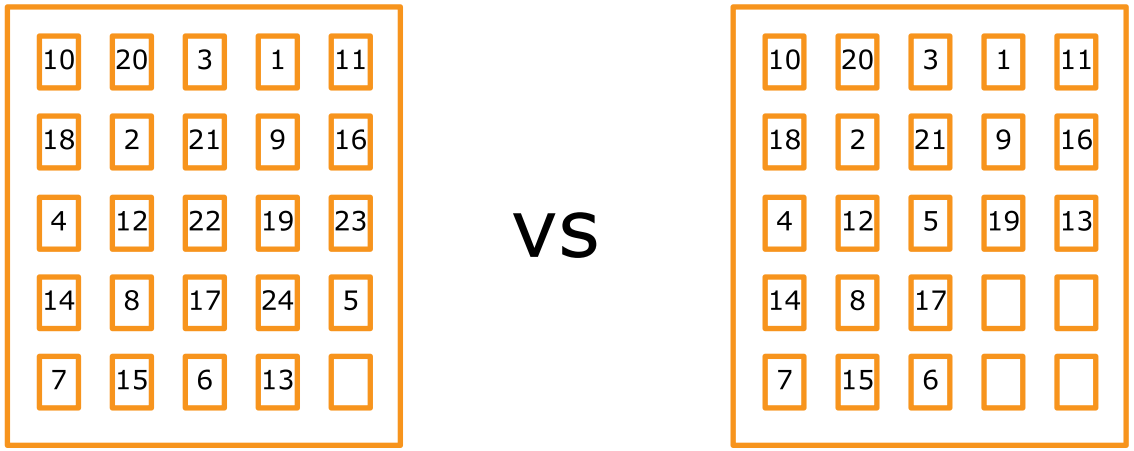
$$\text{Over-Provision} = \frac{\text{Physical capacity} - \text{User capacity}}{\text{User capacity}}$$



PHISON

Example

- The larger the size of the spare area, the higher the operating efficiency, and the better the performance become



The Sliding Puzzle

OP vs WAF

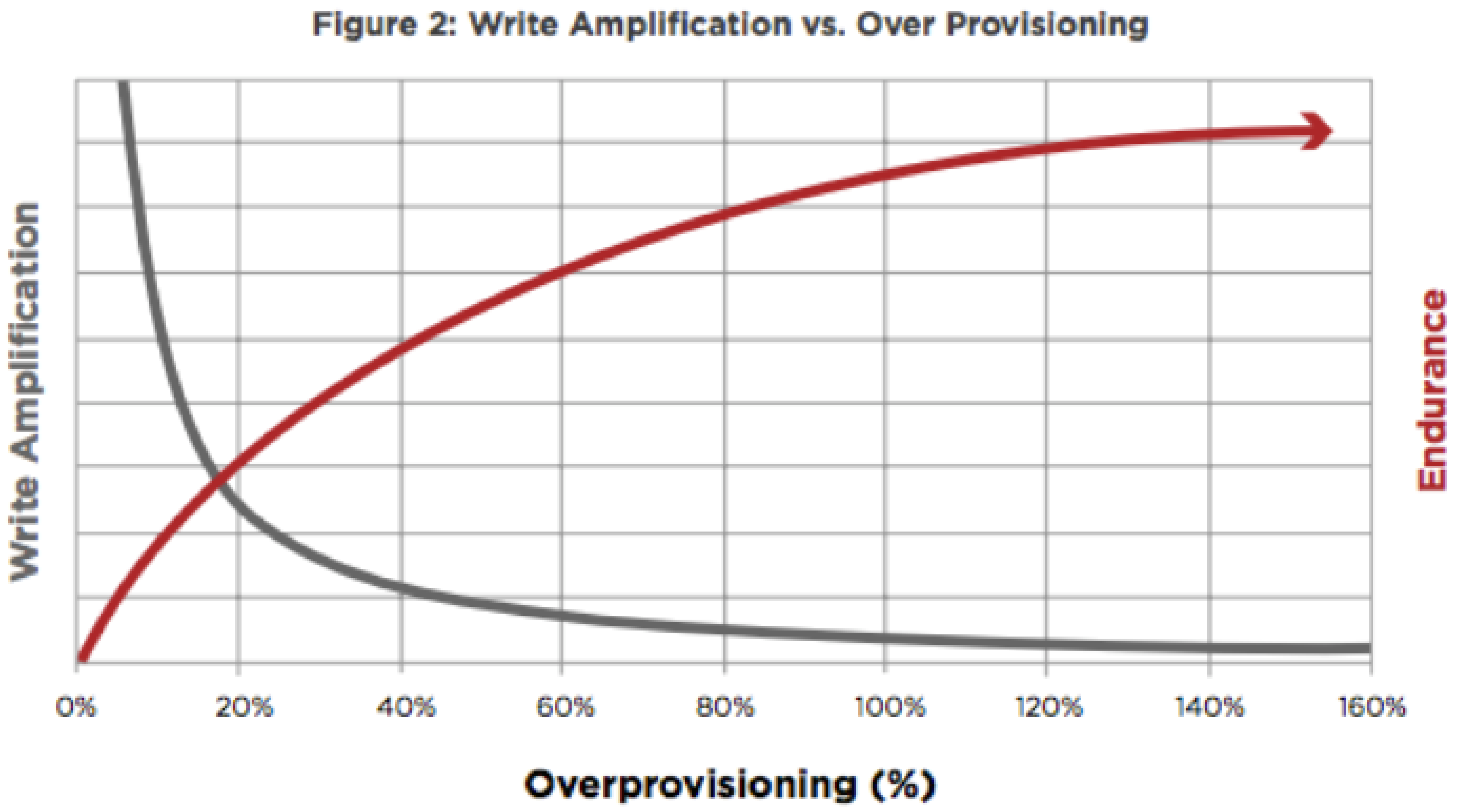


Figure 2: Write Amplification vs. Over Provisioning
Write Amplification
Endurance
0%
20%
40%
60%
80%
100%
120%
140%
160%
Overprovisioning (%)

Summary

- Advantage(s) of higher OP due to less background data movement required
  - Higher performance
  - Better endurance (lower WAF)
- Disadvantage(s) of higher OP due to more reserved spare blocks
  - Less usable storage space

Summary

NAND Flash Limitation

- **Page** is the smallest unit for read and program
- **Block** is the smallest unit for erase
- Must erase before program (cannot overwrite)

Page Base vs Block Base

Q: Page vs Block Base, which is better?

	Page Base	Block Base
Mapping Unit	Page (or 4KB)	Block
Table Size	Large (store in NAND Flash)	Small (store in SRAM)
R/W Performance	Excellent random write performance	Slow random write performance, but impressive read performance
Garbage Collection	Collect valid nodes when empty block becomes insufficient	Collect valid nodes when overwrite previous data or erase data
WAF	Low (efficient block utilization)	High (expensive merge operation)

Cold Facts

- Do you know that your storage device with higher OP (less user space) has better performance and lifetime?
- Do you know that when you keep your storage usage full, the performance and lifetime become worst (due to frequent GC operation)?
- When purchasing storage device, consider performance vs lifetime (such as WAF or TBW)

Advanced Questions

- What if power cycle occurs when we program NAND Flash?
- What if bad block occurs when we program NAND Flash?
- What if read error occurs when we read NAND Flash?
- What if certain blocks wear out quickly than other blocks?

FTL

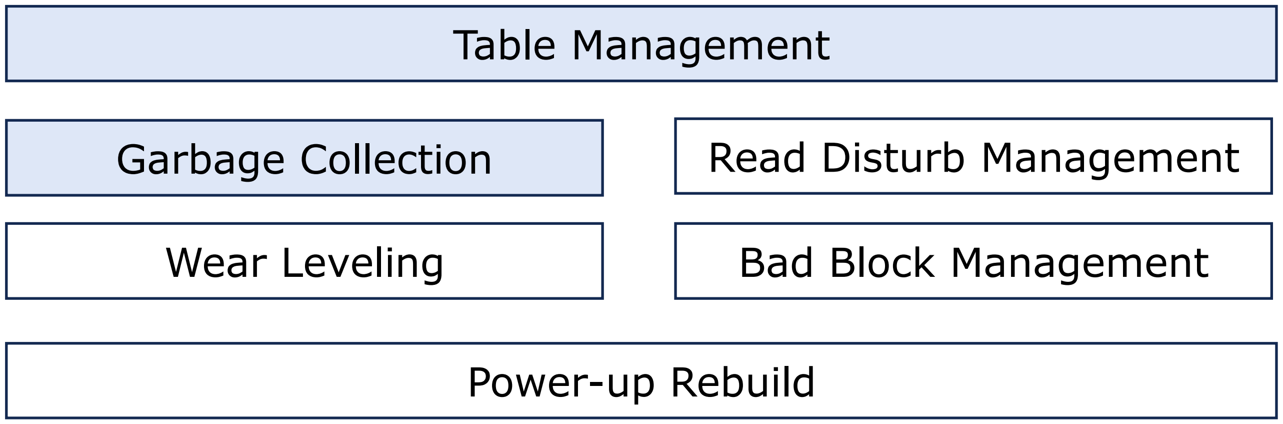


Table Management
Garbage Collection
Read Disturb Management
Wear Leveling
Bad Block Management
Power-up Rebuild

THANK YOU!