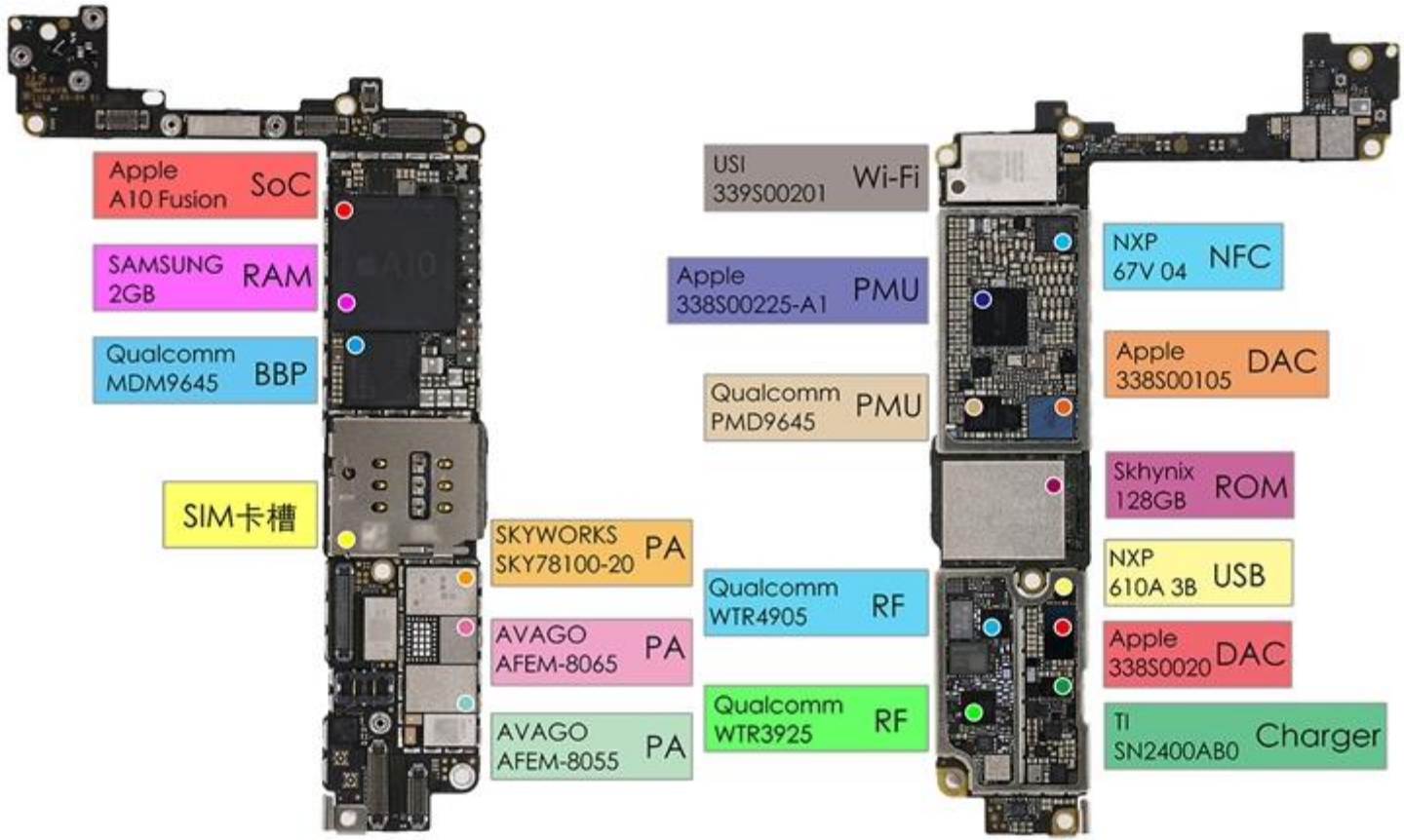
# 高级计算机系统结构

沈海华
shenhh@ucas.ac.cn

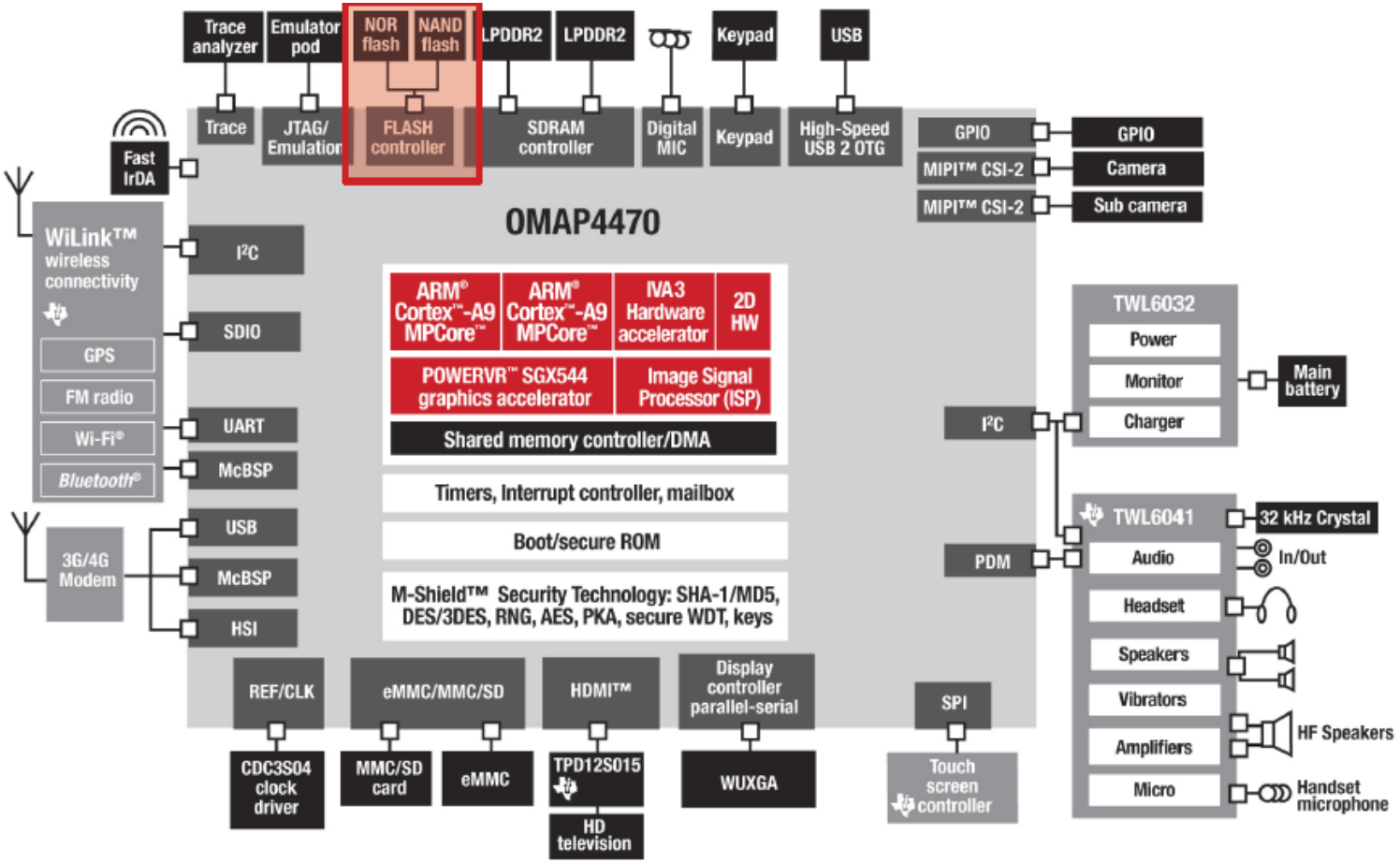
# 第五讲 Flash Storage

- 简介
- 分类
- 应用
- 特性

# 手机里面最主要的芯片有什么？



# Diagram of a CellPhone SoC




- Flash is ubiquitous in cellphones and tablets
  - And increasingly common in notebooks and servers

# 闪存Flash

- Flash是一种非易失性（ Non-Volatile ）存储器
  - 比较：DRAM 属于易失存储器
    - 停止供电内存中的数据就无法保持，因此每次开机都需要把数据重新载入内存。
  - Flash作为非易失性（ Non-Volatile ）存储器，在断电的条件下也能够长久保持数据，其存储特性相当于硬盘
- 闪存分类：NAND Flash vs NOR Flash
  - NOR Flash（Intel 1988，替代EPROM和EEPROM)
  - NAND Flash（东芝 1989，成本低，容量大，有利于大规模普及)

# Flash vs EPROM（或EEPROM）

- Flash是EPROM和EEPROM的升级换代产品。
- 比较：

  相同点：

  - 都是非易失性（ Non-Volatile ）存储器
  - 技术上是EPROM和EEPROM二者的结合

  不同点：

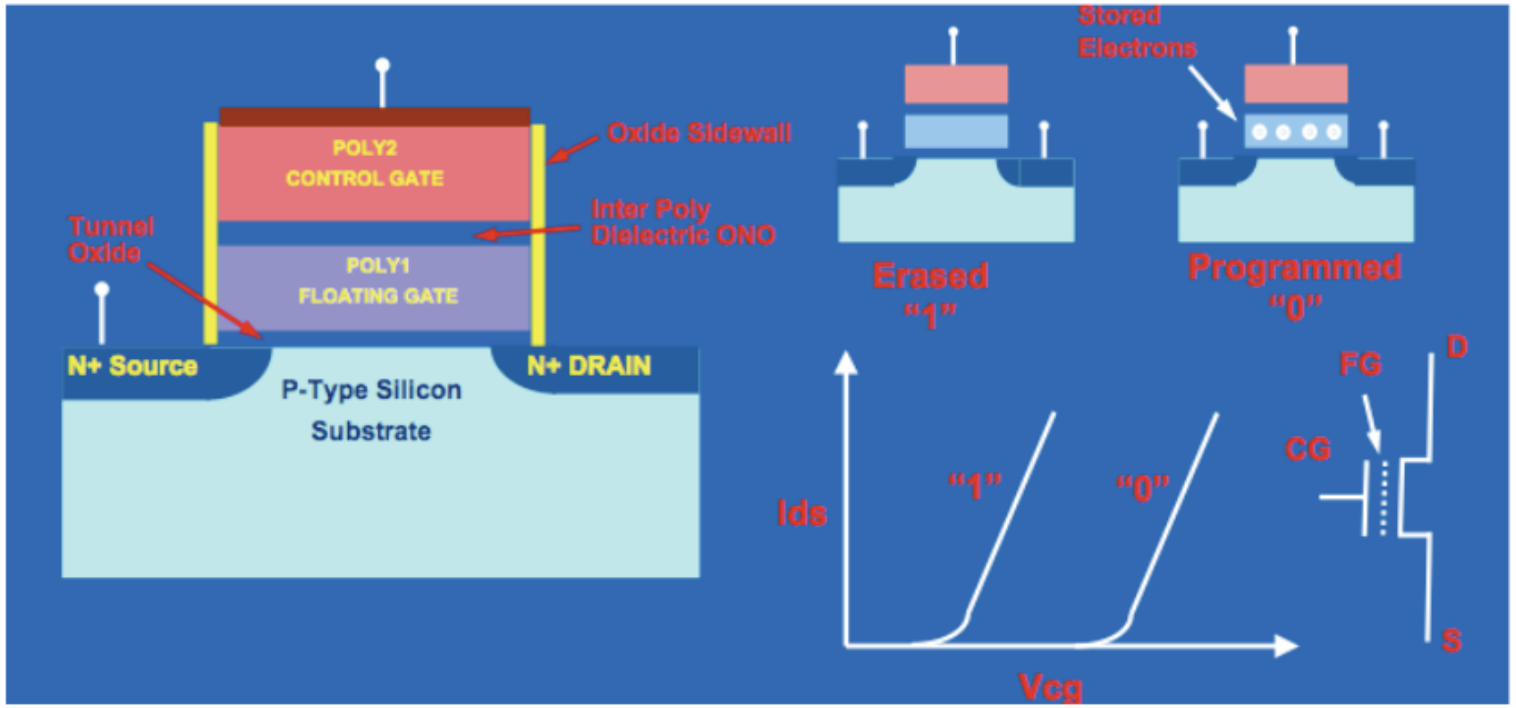
  - Flash对芯片提供整块（big block）擦除，降低芯片设计复杂性
  - Flash单元结构较EEPROM省一个晶体管，单位面积容量更大、集成度更高。
  - 改进了工艺，写入速度更快。
  - EEPROM可以按位擦写，Flash只能按块（block）擦写，由于RAM通常需要按字节修改，Flash ROM还做不到，所以目前不能替代内存。
  - 抗震、无噪声、速度快、耗电低，基本取代硬盘。

# 为什么要用Flash？

- Flash: semiconductor, non-volatile memory
- Compared to a hard disk
  - Lower latency
  - Lower power
  - Lighter weight, smaller size, shock resistance
- Rough comparisons for DRAM : Flash : Disk
  - Cost per bit: 100 : 10 : 1
  - Access latency: 1 : 5,000 : 1,000,000

DDR4 0.5USD/Gbits
TLC 0.014USD/Gbits　MLC 0.065USD/G bit　SLC 0.825USD/Gbits
Disk 0.005 USD/Gbits

# Flash 基本单元

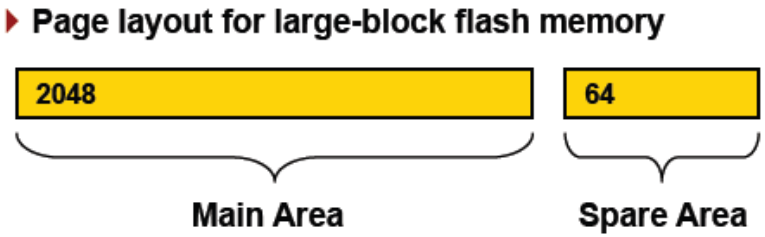


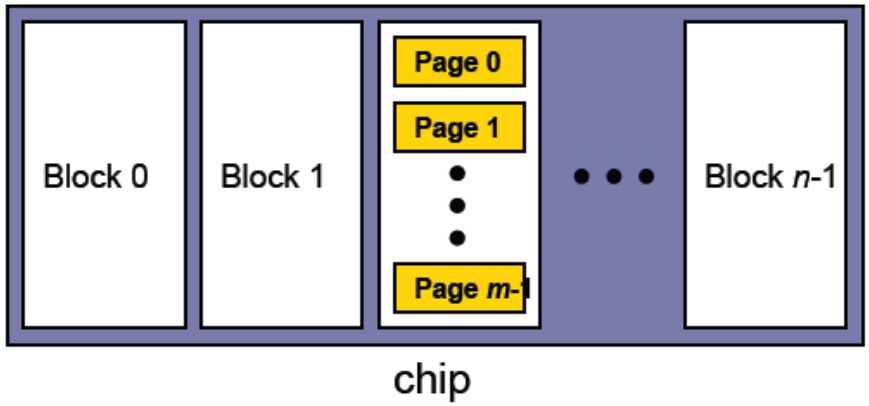

- Store bit as charge trapped in floating gate
  - Charge modulates Vth of underlying transistor
  - Writing/erasing by applying high/low Vcg

# Flash的类型

- NOR flash
  - Fast read (~100ns), slow writes (200usec), very slow erase (1sec)
  - 10K to 100K erase cycles
  - Used for instruction memory in mobile systems
- NAND flash (our focus today)
  - Denser (bits/area, ~40% of NOR), cheaper per GB
  - Slow read (20-50usec), slow writes (200usec), slow erase (2msec)
  - 100K to 1M erase cycles
  - Used for data storage (phones, USB keys, solid-state drives, …)
- Both types have durability issues
  - Damaged after some number of write/erase cycles

# Flash 芯片的基本结构




- Single (SLC) or multiple (MLC) bits per cell
- Page: minimum unit of read/write
  - 0.5Kb – 8Kb of data + spare area for error coding
- Block: minimum unit of erasing
  - 64 – 128 pages
- Chip: 1 – 16GB
  - Upto 16K blocks per chip

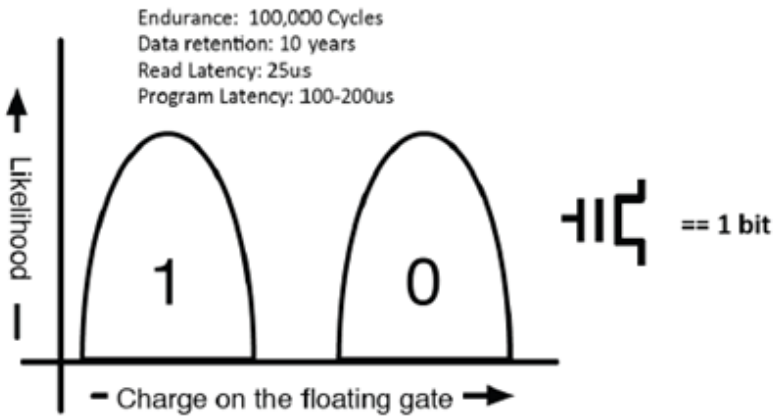
# Flash 基本操作

- Read the contents of a page
  - 20-50us
- Write (program) data to a page
  - Only 1→ 0 transitions are allowed
  - Writing within a block must be ordered by block
  - 100-300us
- Erase all bits in a block to 1
  - Pages must be erased before they can be written
  - Update-in-place is not possible
  - 0.5-3ms

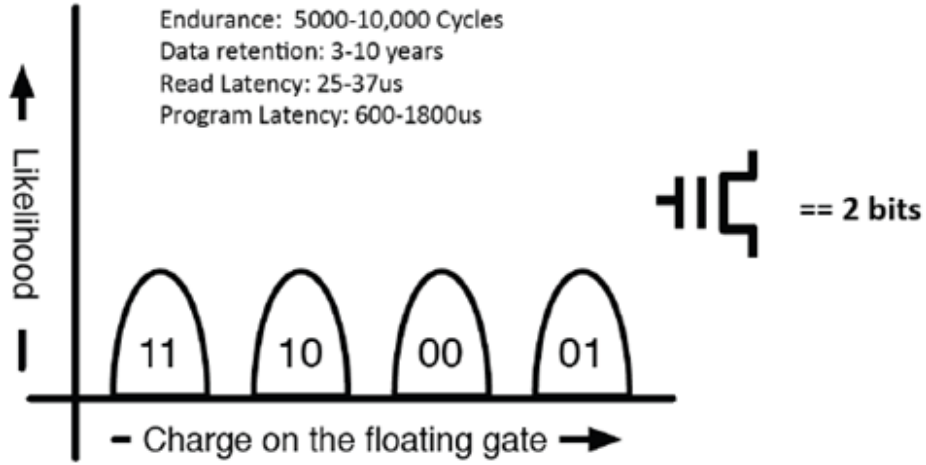
# Flash的可靠性

- Wear out
  - Flash cells are physically damaged programming and erasing them
- Writing disturb
  - Programming pages can corrupt the values of other pages in the block
- Read disturb
  - Reading data can corrupt the data in the block
  - It takes many reads to see this effect

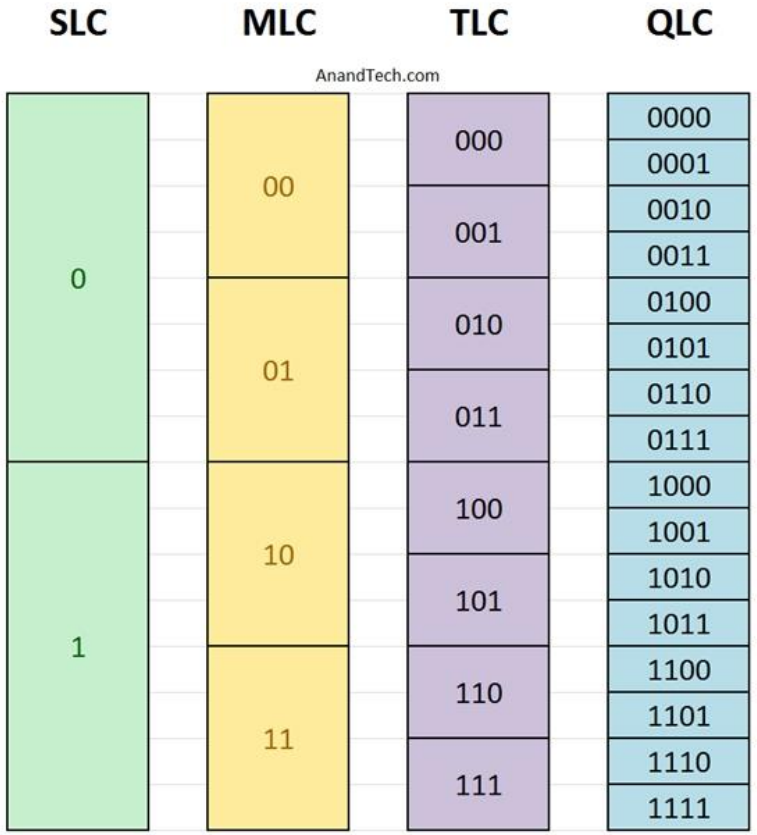
# Multi-level Cells




SLC




MLC



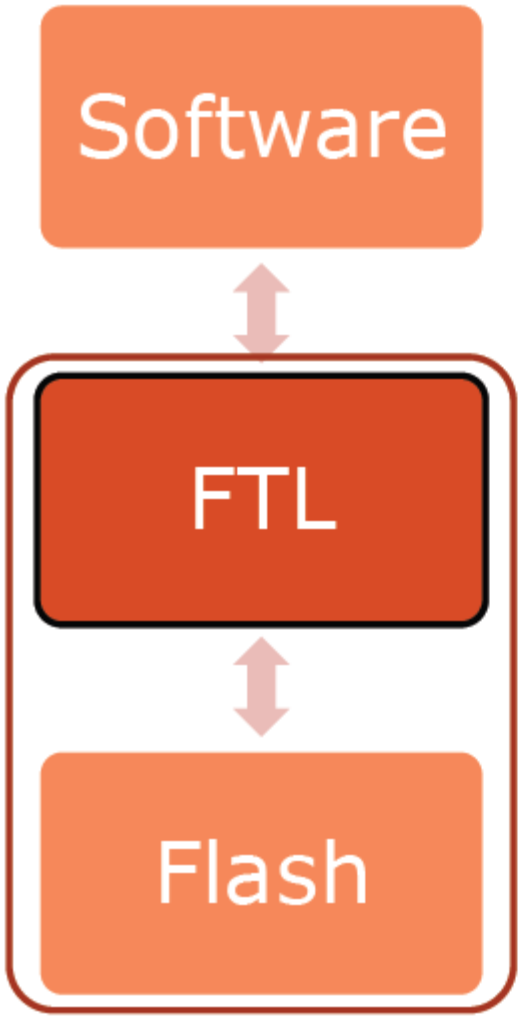
# MLC vs SLC

- SLC – single-level cell
  - Faster but less dense
  - More reliable (100K -1M erase cycles)
  - $5.60/GB
  - Used in “enterprise” drives (i.e. Intel Extreme SSDs)
- MLC – multilevel cell
  - Slower but denser
  - Less reliable (1K - 10K erase cycles)
  - $0.53 /GB
  - Used in consumer drives (flash cards, thumb drives, cheap SSDs, etc.)

# Making Flash Useful

- Raw flash is not terribly useful
  - The interface is ugly; rules are onerous
- Instead, we want it to look like a disk
  - Build a "flash translation layer (FTL)
  - Exposes a block-based interface (like a disk)
  - Manages program/erase granularity mismatch
  - Equalizes wear
  - Delivers high performance (not always…)
- FTL implemented using a microcontroller & SRAM/DRAM buffers
  - See any issue with the latter?

# Flash Translation Layer (FTL)



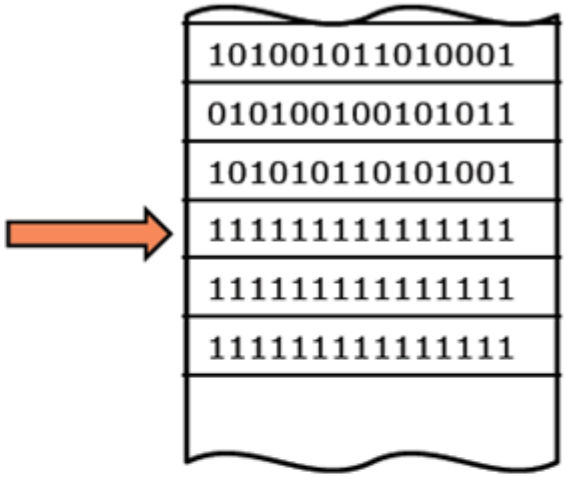

- User
  - Logical Block Address
- Flash
  - Write pages in order
  - Erase/Write granularity
  - Wears out
- FTL
  - Logical → physical map
  - Wear leveling
  - Power cycle recovery

# Centralized FTL State

**Map**

| LBA | Physical Page Address | |
|---|---|---|
| 0 | Block 5 | Page 7 |
| 2k | Block 27 | Page 0 |
| 4k | Block 10 | Page 2 |

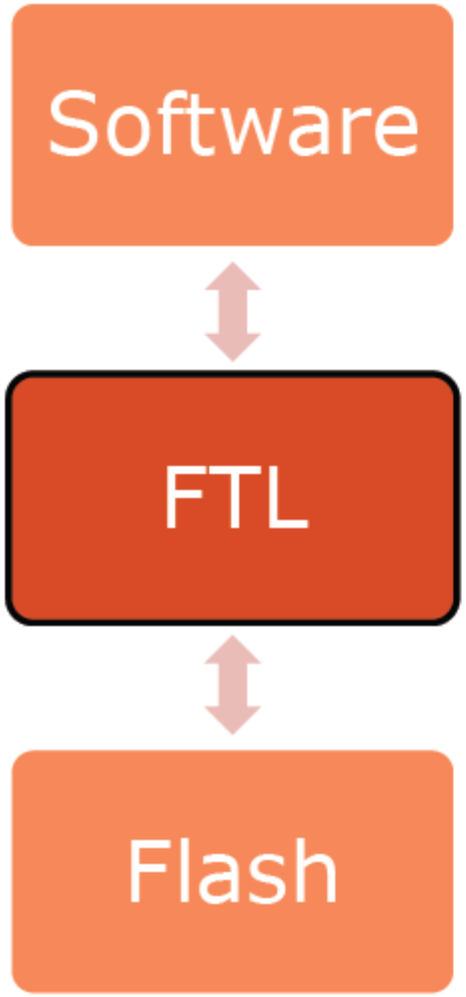
**Write Point**




**Block Info Table**

| Block | Erased | Erase Count | Valid Page Count | Sequence Number | Bad Block Indicator |
|---|---|---|---|---|---|
| 0 | False | 3 | 15 | 5 | False |
| 1 | True | 7 | 0 | - | False |
| 2 | False | 0 | 4 | 9 | False |

# Read




**1. Read Data at LBA 2k**

**2. Map**

| LBA | Physical Page Address | |
|---|---|---|
| 0 | Block 5 | Page 7 |
| 2k | Block 27 | Page 0 |
| 4k | Block 10 | Page 2 |

**3. Flash Operation**

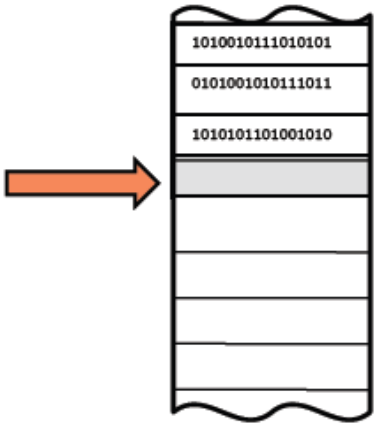
# Write – Mid Block

**Write** 0101101011001010 **to LBA 2k**

**Write Point = Block 2, Page 5**

**Map**

| LBA | Physical Page Address | |
|---|---|---|
| 0 | Block 5 | Page 7 |
| 2k | Block 0 | Page 0 |
| 4k | Block 10 | Page 2 |




**Block Info Table**

| Block | Erased | Erase Count | Valid Page Count | Sequence Number | Bad Block Indicator |
|---|---|---|---|---|---|
| 0 | False | 3 | 15 | 5 | False |
| 1 | True | 7 | 0 | - | False |
| 2 | False | 0 | 4 | 9 | False |

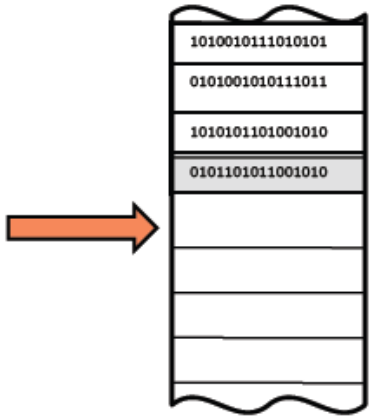
# Write – Mid Block

**Write** 0101101011001010 **to LBA 2k**

**Write Point = Block 2, ~~Page 5~~ Page 6**

**Map**

| LBA | Physical Page Address | |
|---|---|---|
| 0 | Block 5 | Page 7 |
| 2k | Block ~~0~~ 2 | Page ~~0~~ 5 |
| 4k | Block 10 | Page 2 |




**Block Info Table**

| Block | Erased | Erase Count | Valid Page Count | Sequence Number | Bad Block Indicator |
|---|---|---|---|---|---|
| 0 | False | 3 | ~~15~~ 14 | 5 | False |
| 1 | True | 7 | 0 | - | False |
| 2 | False | 0 | ~~4~~ 5 | 9 | False |

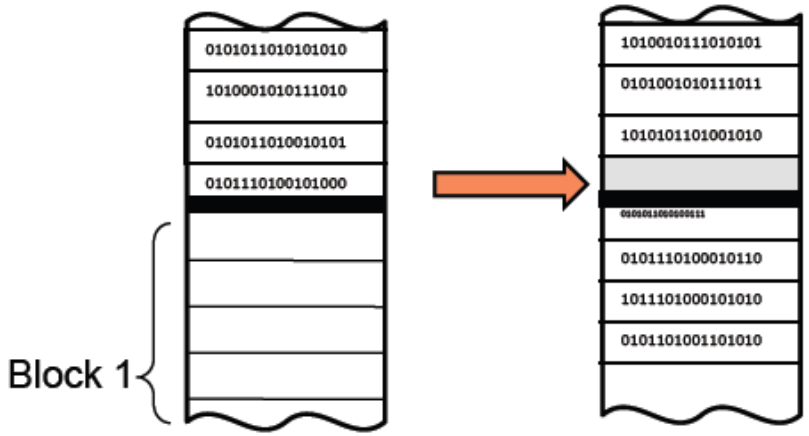
# Write – Block Jump (1)

**Write** 0101001010100110 **to LBA 2k**

**Write Point = Block 2, Page 63**

**Map**

| LBA | Physical Page Address | |
|---|---|---|
| 0 | Block 5 | Page 7 |
| 2k | Block 0 | Page 5 |
| 4k | Block 0 | Page 2 |




**Block Info Table**

| Block | Erased | Erase Count | Valid Page Count | Sequence Number | Bad Block Indicator |
|---|---|---|---|---|---|
| 0 | False | 3 | 15 | 5 | False |
| 1 | True | 7 | 0 | - | False |
| 2 | False | 0 | 4 | 9 | False |

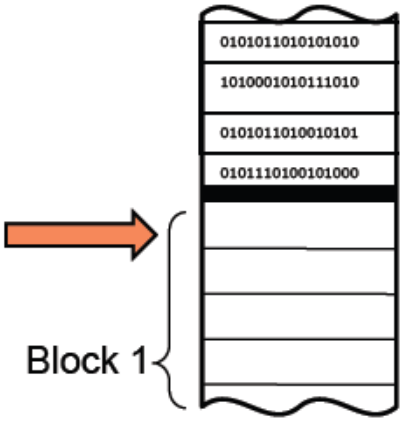
# Write – Block Jump (1)

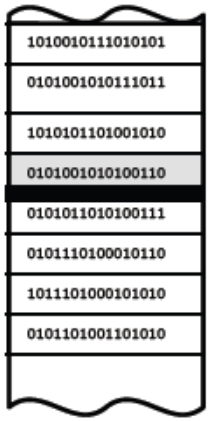
**Write** 0101001010100110 **to LBA 2k**

**Write Point =** ~~**Block 2, Page 63**~~ **Block 1, Page 0**

**Map**

| LBA | Physical Page Address | |
|---|---|---|
| 0 | Block 5 | Page 7 |
| 2k | Block ~~0~~ 2 | Page ~~5~~ 63 |
| 4k | Block 0 | Page 2 |




**Block Info Table**

| Block | Erased | Erase Count | Valid Page Count | Sequence Number | Bad Block Indicator |
|---|---|---|---|---|---|
| 0 | False | 3 | ~~15~~ 14 | 5 | False |
| 1 | True | 7 | 0 | - | False |
| 2 | False | 0 | ~~4~~ 5 | 9 | False |

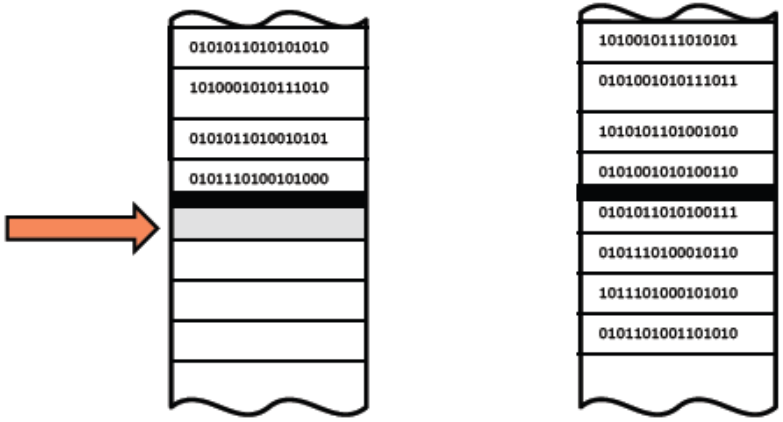
# Write – Block Jump (2)

**Write** 1101000101101001 **to LBA 4k**

**Write Point = Block 1, Page 0**

**Map**

| LBA | Physical Page Address | |
|---|---|---|
| 0 | Block 5 | Page 7 |
| 2k | Block 2 | Page 63 |
| 4k | Block 0 | Page 2 |




**Next Sequence Number: 12**

**Block Info Table**

| Block | Erased | Erase Count | Valid Page Count | Sequence Number | Bad Block Indicator |
|---|---|---|---|---|---|
| 0 | False | 3 | 14 | 5 | False |
| 1 | True | 7 | 0 | - | False |
| 2 | False | 0 | 5 | 9 | False |

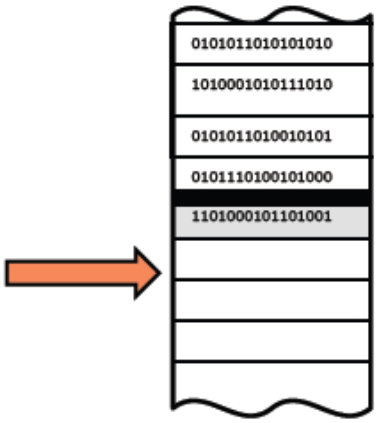
# Write – Block Jump (2)

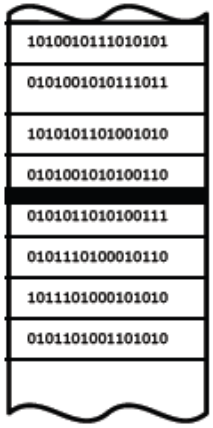
**Write** 1101000101101001 **to LBA 4k**

**Write Point = Block 1, ~~Page 0~~ Page 1**

**Map**

| LBA | Physical Page Address | |
|---|---|---|
| 0 | Block 5 | Page 7 |
| 2k | Block 2 | Page 63 |
| 4k | Block ~~0~~ 1 | Page ~~2~~ 0 |




**Block Info Table**

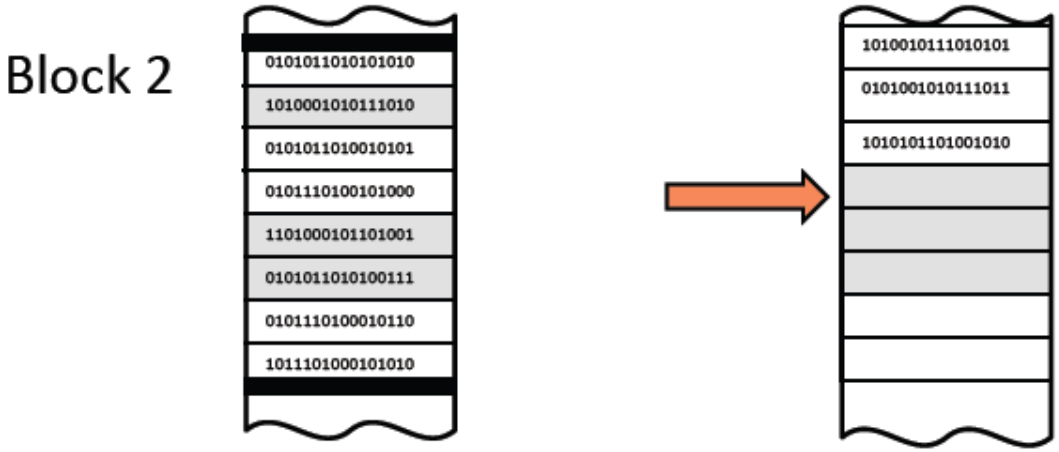
| Block | Erased | Erase Count | Valid Page Count | Sequence Number | Bad Block Indicator |
|---|---|---|---|---|---|
| 0 | False | 3 | ~~14~~ 13 | 5 | False |
| 1 | ~~T~~ F | 7 | ~~0~~ 1 | 12 | False |
| 2 | False | 0 | 5 | 9 | False |

# Erase

**Block Info Table**

| Block | Erased | Erase Count | Valid Page Count | Sequence Number | Bad Block Indicator |
|---|---|---|---|---|---|
| 0 | False | 3 | 13 | 5 | False |
| 1 | False | 7 | 1 | 12 | False |
| 2 | False | 0 | 3 | 9 | False |

**Move Valid Pages**

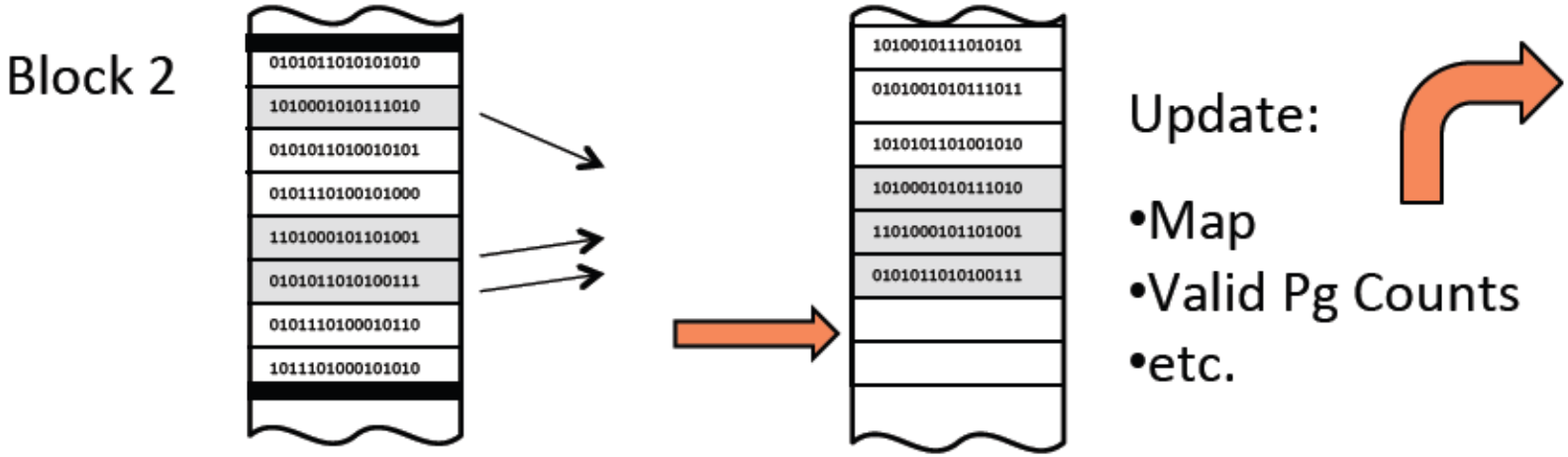


# Erase

**Block Info Table**

| Block | Erased | Erase Count | Valid Page Count | Sequence Number | Bad Block Indicator |
|---|---|---|---|---|---|
| 0 | False | 3 | 13 | 5 | False |
| 1 | False | 7 | 1 | 12 | False |
| 2 | False | 0 | ~~3~~ 0 | 9 | False |

**Move Valid Pages**



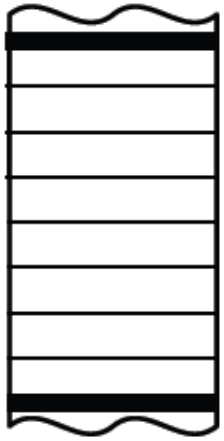
# Erase

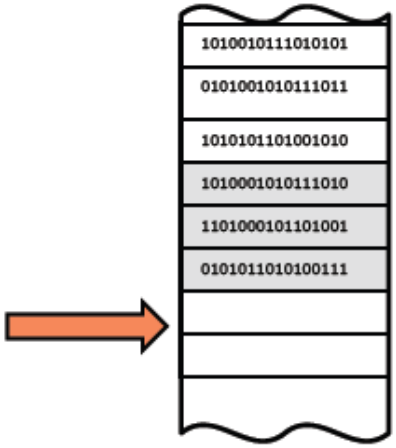
**Block Info Table**

| Block | Erased | Erase Count | Valid Page Count | Sequence Number | Bad Block Indicator |
|---|---|---|---|---|---|
| 0 | False | 3 | 13 | 5 | False |
| 1 | False | 7 | 1 | 12 | False |
| 2 | ~~F~~ T | ~~0~~1 | 0 | - | False |

**Move Valid Pages**

Block 2



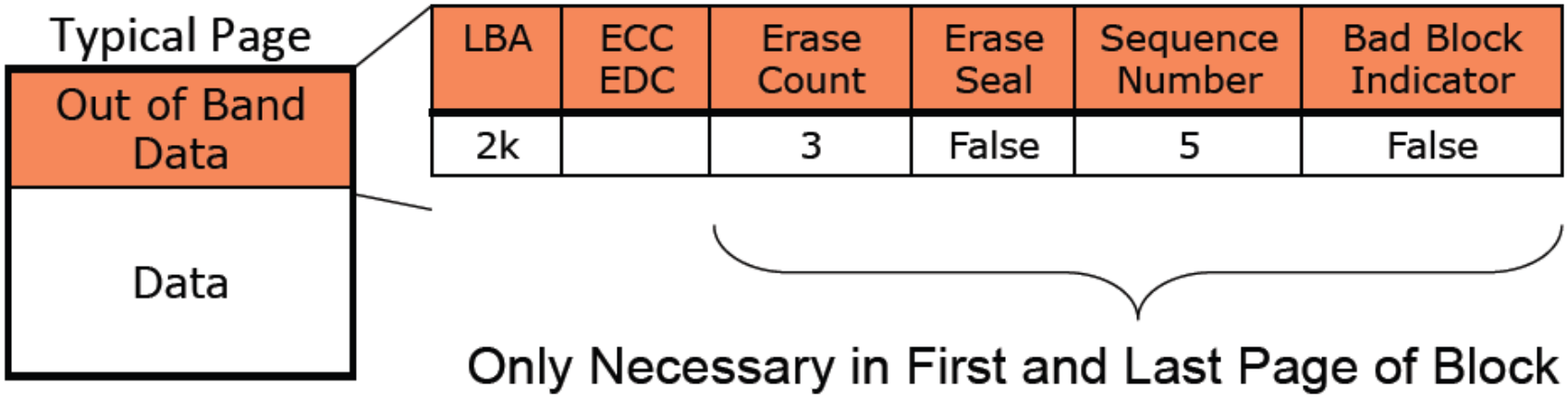



Update:

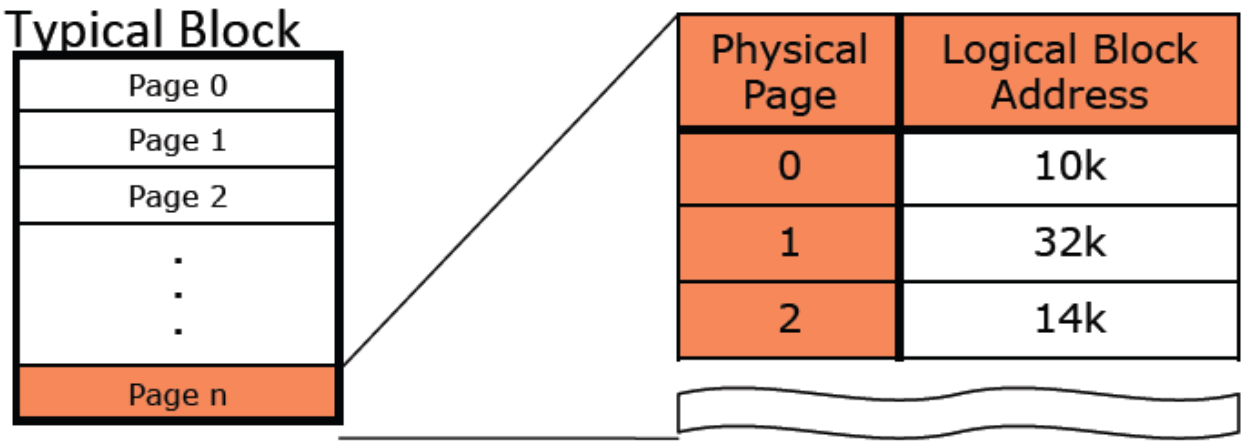
- Map
- Valid Pg Counts
- etc.

# Distributed FTL State

**Metadata**




**Summary Page**



# Power Cycle

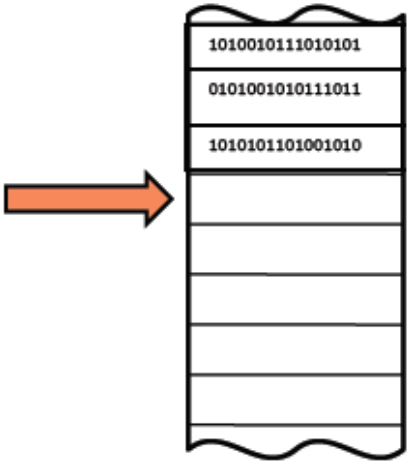
Scan each block:

1. Summary page
2. First Page
3. All Pages

**Map**

| LBA | Physical Page Address | |
|---|---|---|
| 0 | Block 5 | Page 7 |
| 2k | Block 27 | Page 0 |
| 4k | Block 10 | Page 2 |

**Write Point**




**Block Info Table**

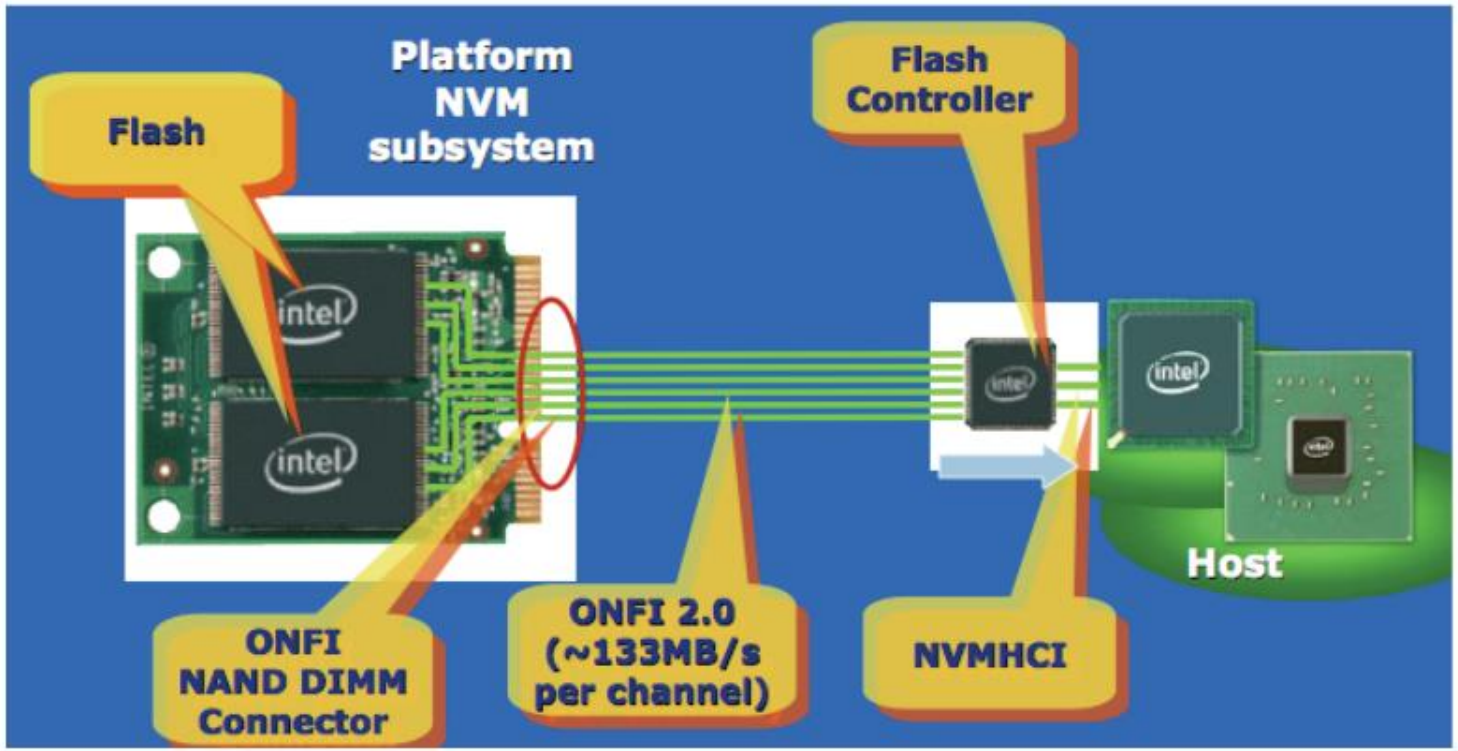
**Next Sequence Number: 12**

| Block | Erased | Erase Count | Valid Page Count | Sequence Number | Bad Block Indicator |
|---|---|---|---|---|---|
| 0 | False | 3 | 15 | 5 | False |
| 1 | True | 7 | 0 | 0 | False |
| 2 | False | 0 | 4 | 9 | False |

# FTL Options

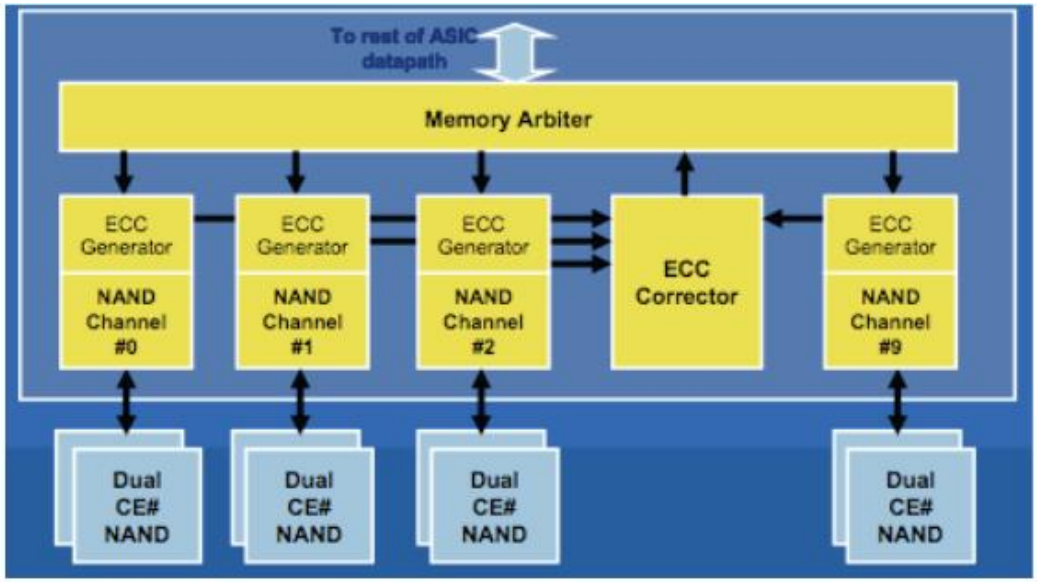
- Per-page vs per-block mapping
  - Per-page: flexibility at the cost of large tables
  - Per-block: small tables but write-amplification issue
    - Must copy whole block when 1 page updated
- Log-based systems
  - Log changes to data sequentially in empty blocks
  - Makes writes sequential but reads can be scattered
- . . .
- Lots of FTL variations overall
  - Performance can vary significantly with FTL choice
  - Implementation cost can also vary

# Connecting Flash to the System




- ONFI standard for NAND Flash
  - Allows Flash chips to talk to controllers
  - Controller can be integrated

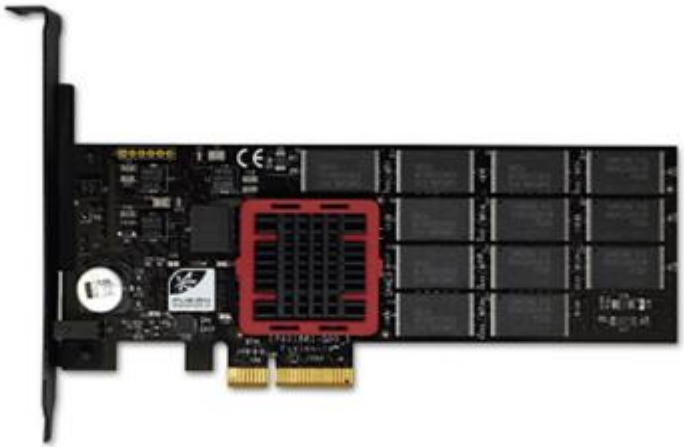
# Flash SSD




- A SATA device
  - Flash chips + control/buffering chip + ROM
  - They sometimes include a capacitor or small battery
    - Needed to flush buffered data on power loss
  - Internal concurrency through multiple channels

# Higher Performance Flash

- SSDs connected to PCI-e
  - Similar latencies but higher bandwidth (2-4x)
  - Highly banked architectures, multi-queue interface
- Examples: Fusion-IO storage cards



2014年7月，Fusion-IO举行了PCI-e闪存卡发布会。
2015年2月，闪迪Sandisk收购Fusion-IO。

# Other Uses of Flash

- Notebooks
  - In addition to hard disk
    - Store code to accelerate program launch
  - Replacement for hard disk
- Servers and data centers
  - High bandwidth (IOPS) storage system
  - Performance & energy savings
  - Good for systems not limited by capacity
    - Cost/IOPS > cost/bit
    - Or for hot data only

# Flash Power Consumption

- Flash vs Disk
  - Sleep state: <0.3W vs >2W
  - Read/write: 2-3W vs 2-10W
  - Bandwidth: 500MB/s vs 60MB/sec
- Flash vs DRAM
  - Sleep state: <0.3W vs ~1W
  - Read/write: 2-3W vs 2-3W
  - Capacity: 128GB vs 8GB

# 致谢：

本讲内容参考了M.I.T. Daniel Sanchez教授的课程讲义，特此感谢。