Memory Systems
Lec12 – Flash Memory and Solid-State Drives

Chin-Fu Nien (粘儆夫)

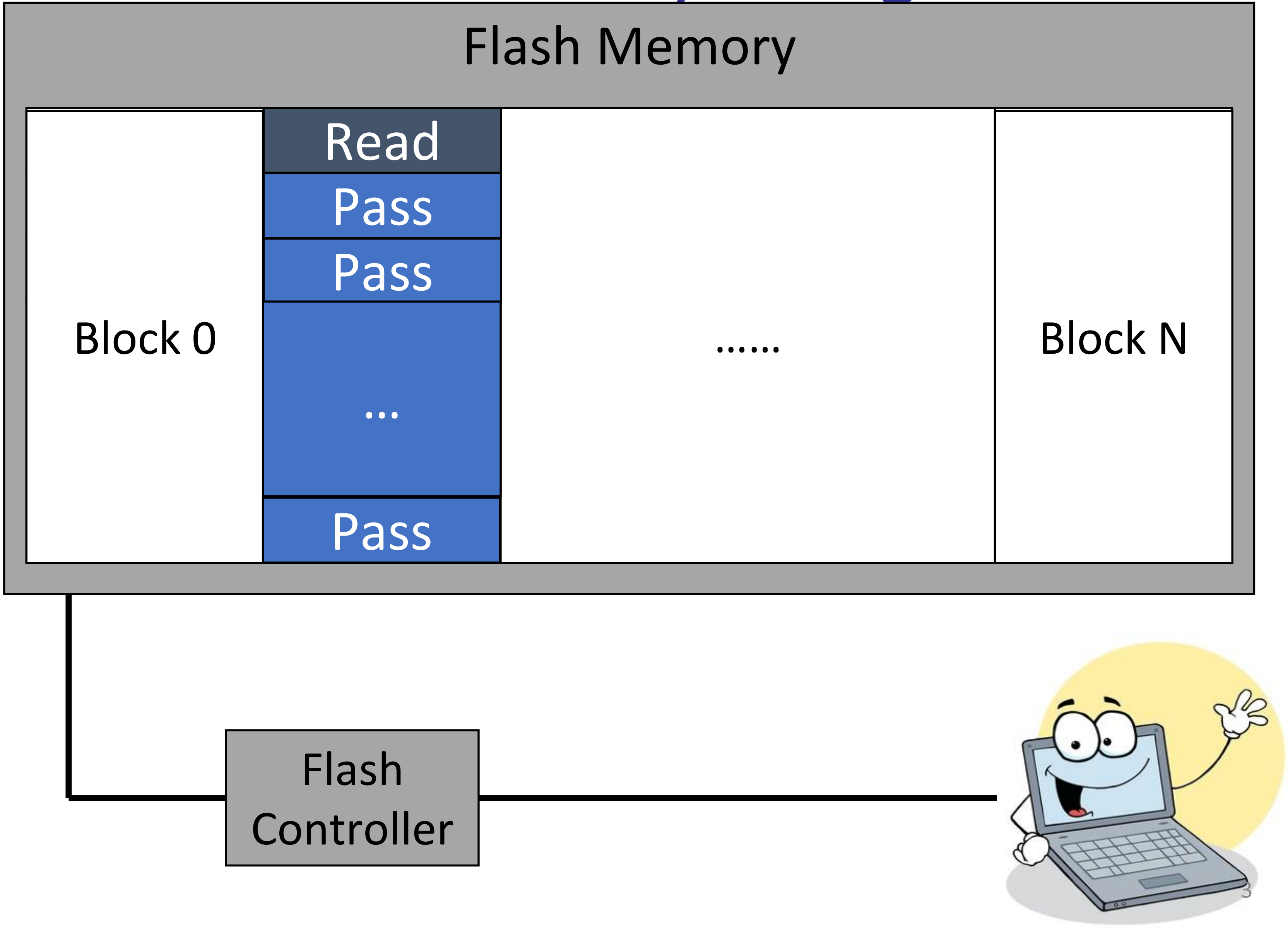
(本節內容改自Prof. Onur Mutlu, “Computer Architecture,” 16th and 17th Lecture, Fall 2023 課程講義)

Module 3: Advanced Architectural Topics

Short Background on NAND Flash Memory Operation

(本節內容改自Prof. Onur Mutlu, “Computer Architecture,” 16th Lecture, Fall 2023 課程講義)

NAND Flash Memory Background



Flash Memory

Block 0

Read

Pass

Pass

…

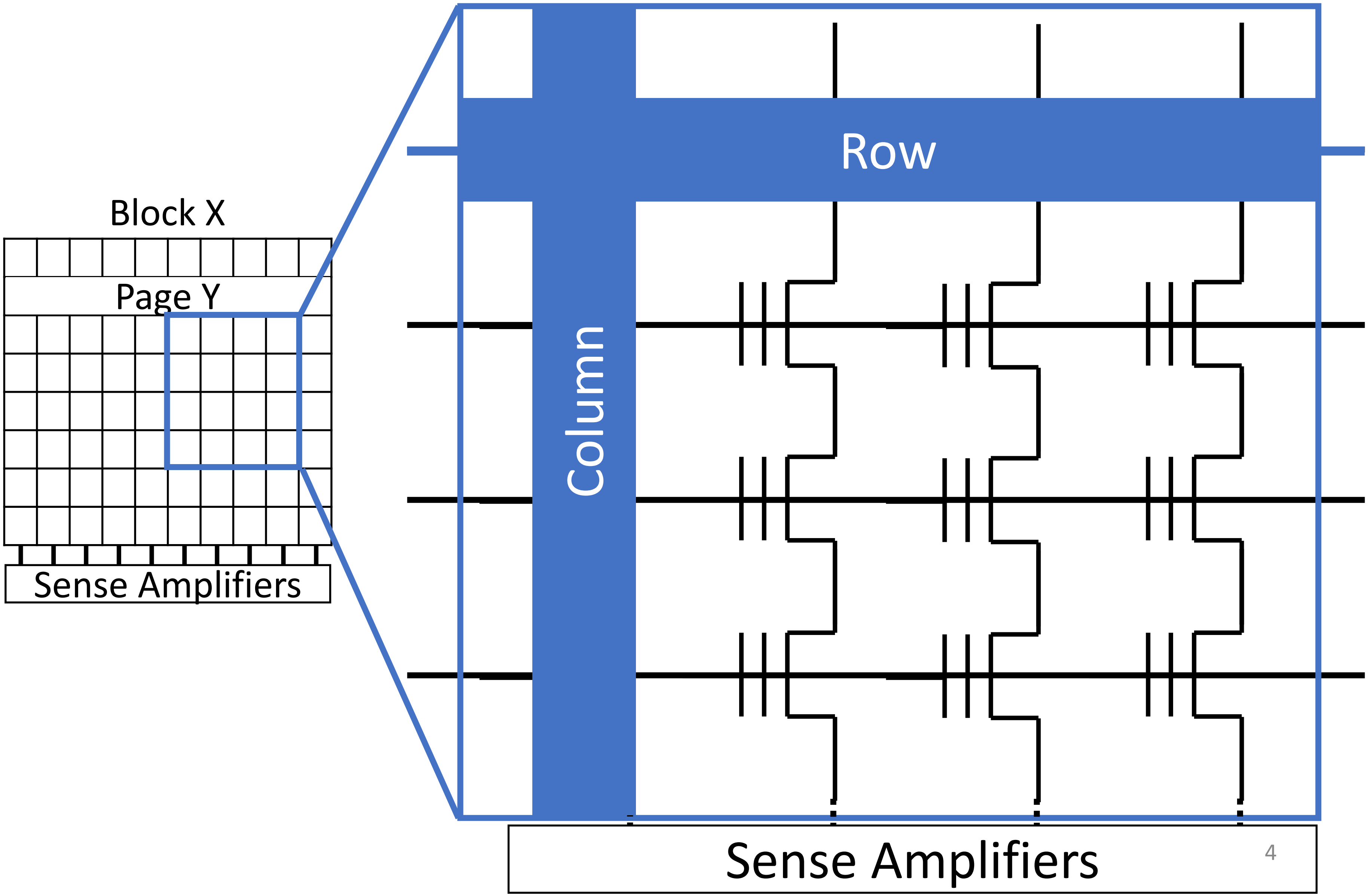
Pass

……

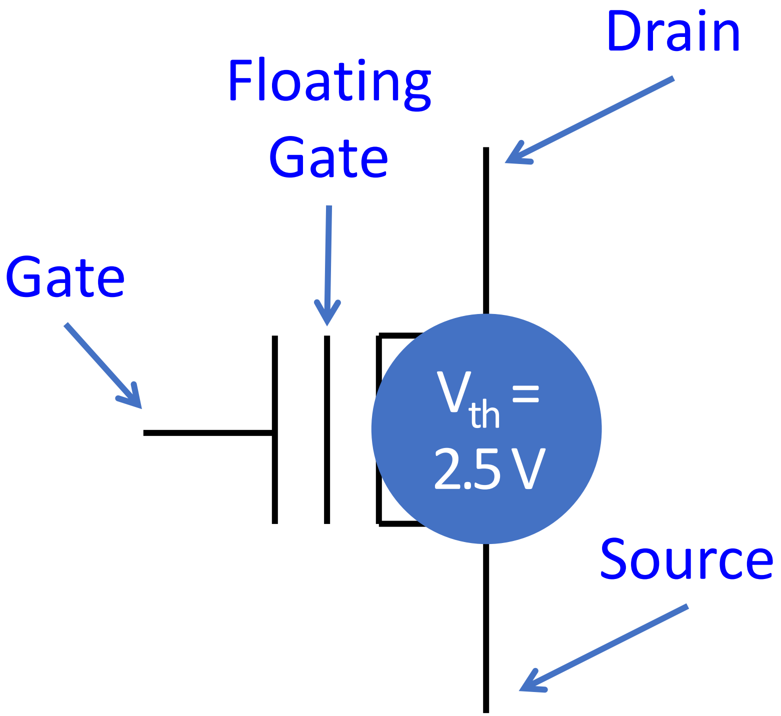
Block N

Flash Controller

Flash Cell Array

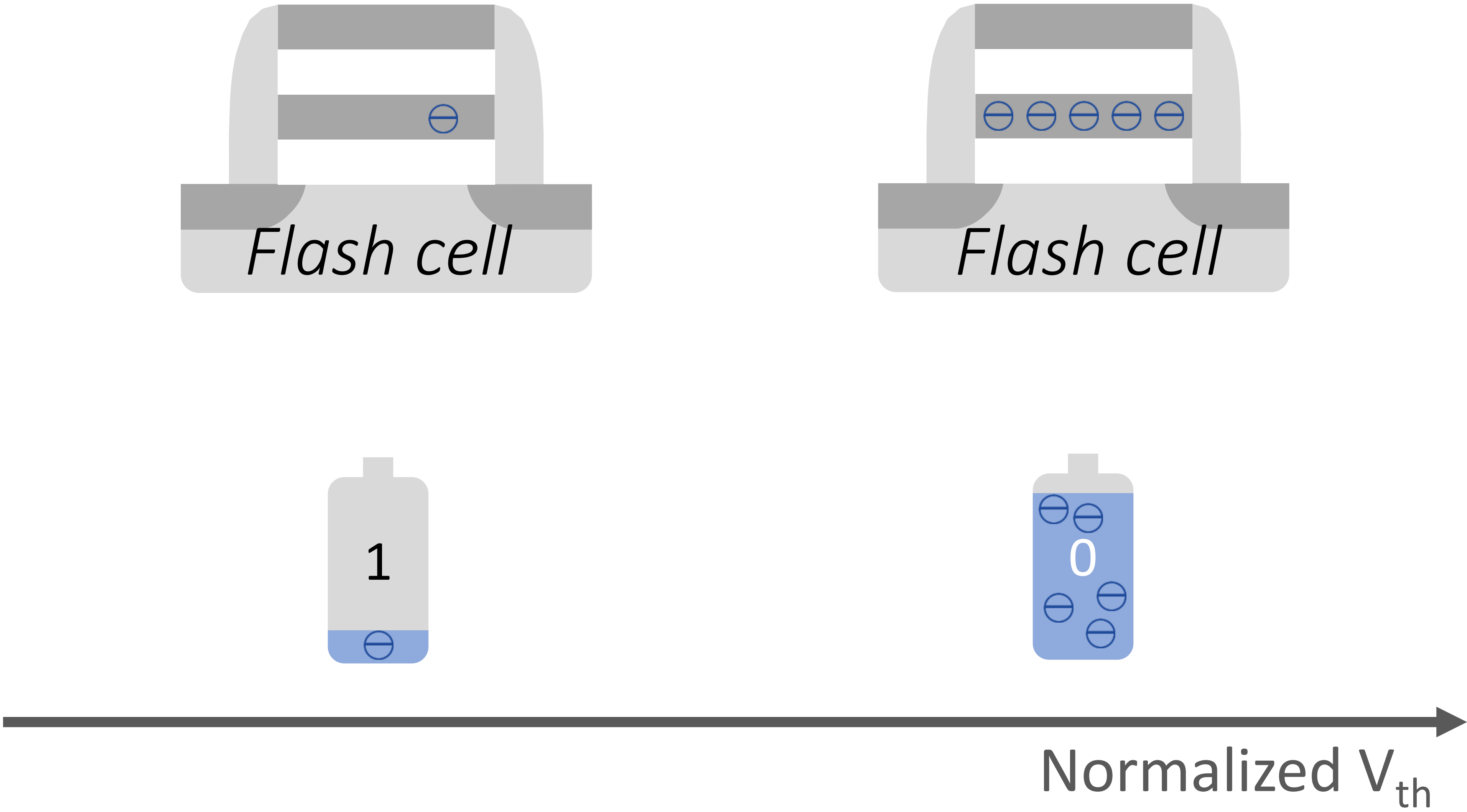


Flash Cell

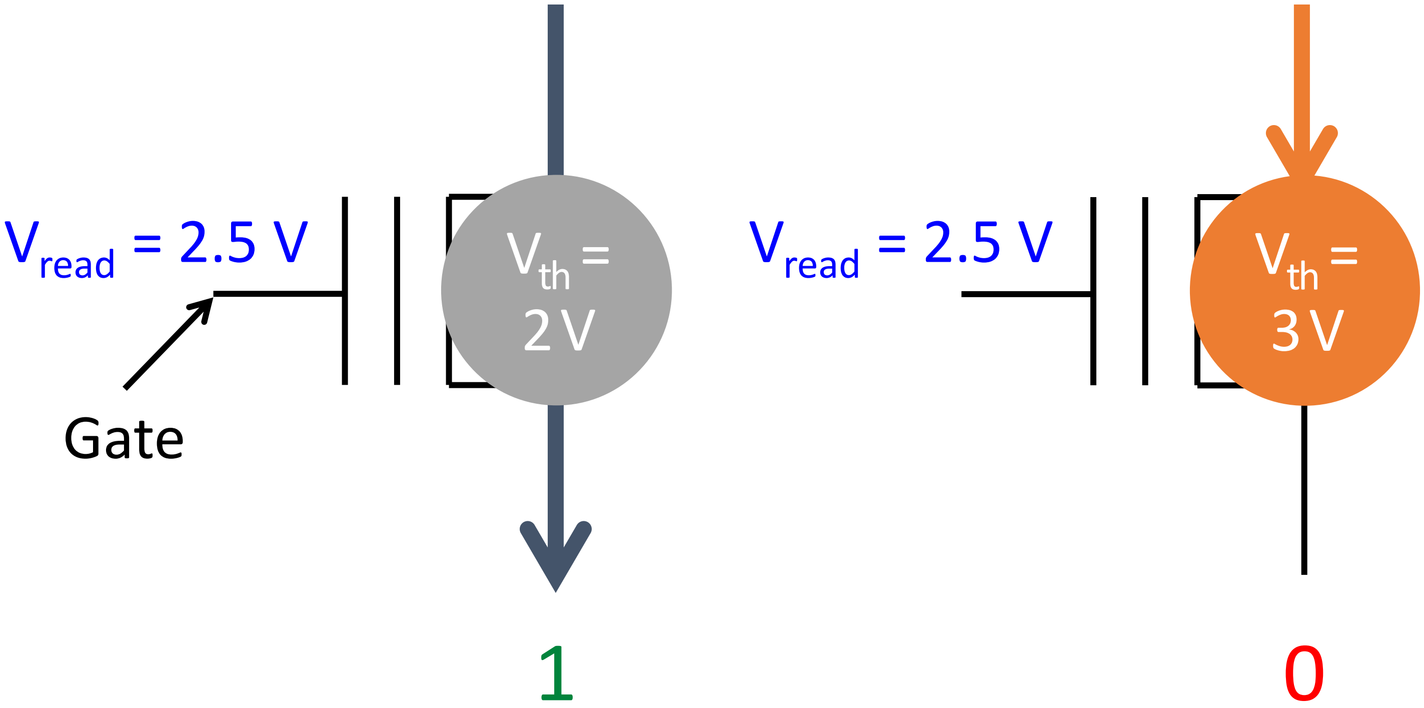


Floating Gate Transistor
(Flash Cell)

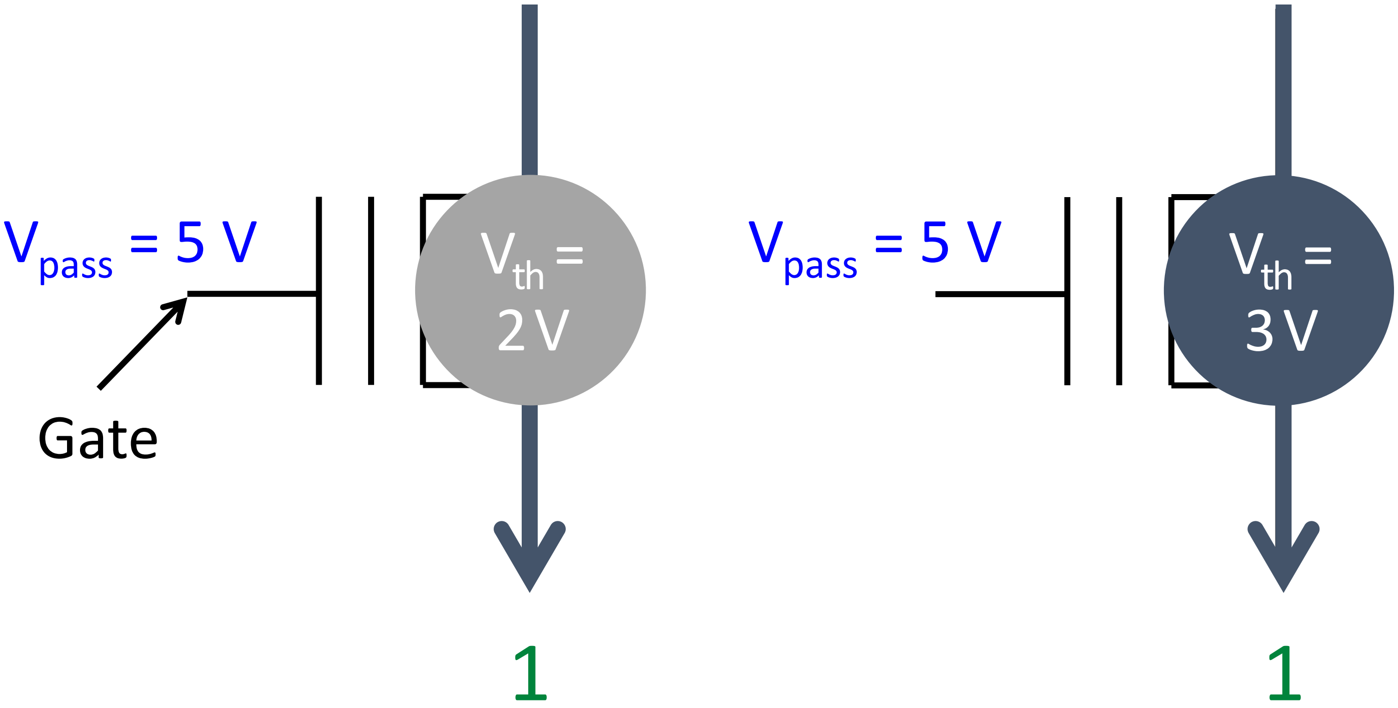
Threshold Voltage (V_{th})



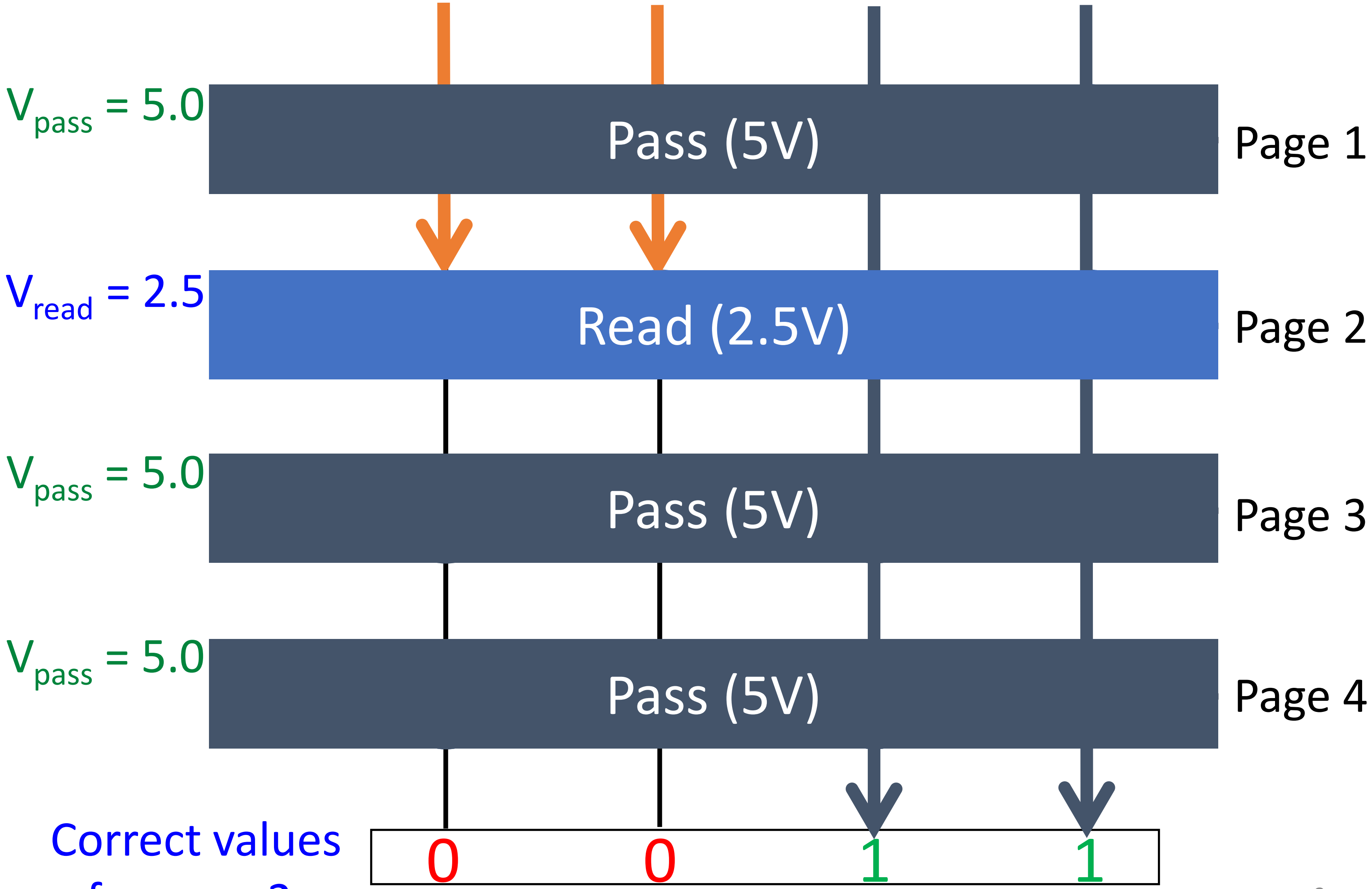
Flash Read



Flash Pass-Through



Read from Flash Cell Array

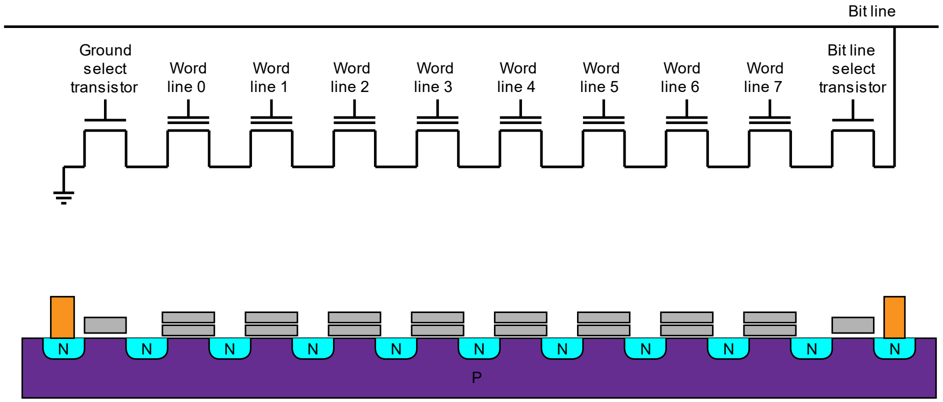


Correct values for page 2:

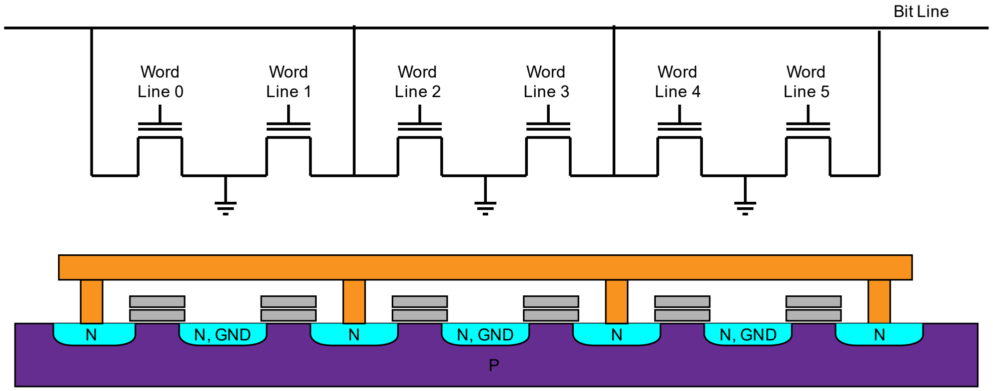
0	0	1	1

Aside: NAND vs. NOR Flash Memory

NAND



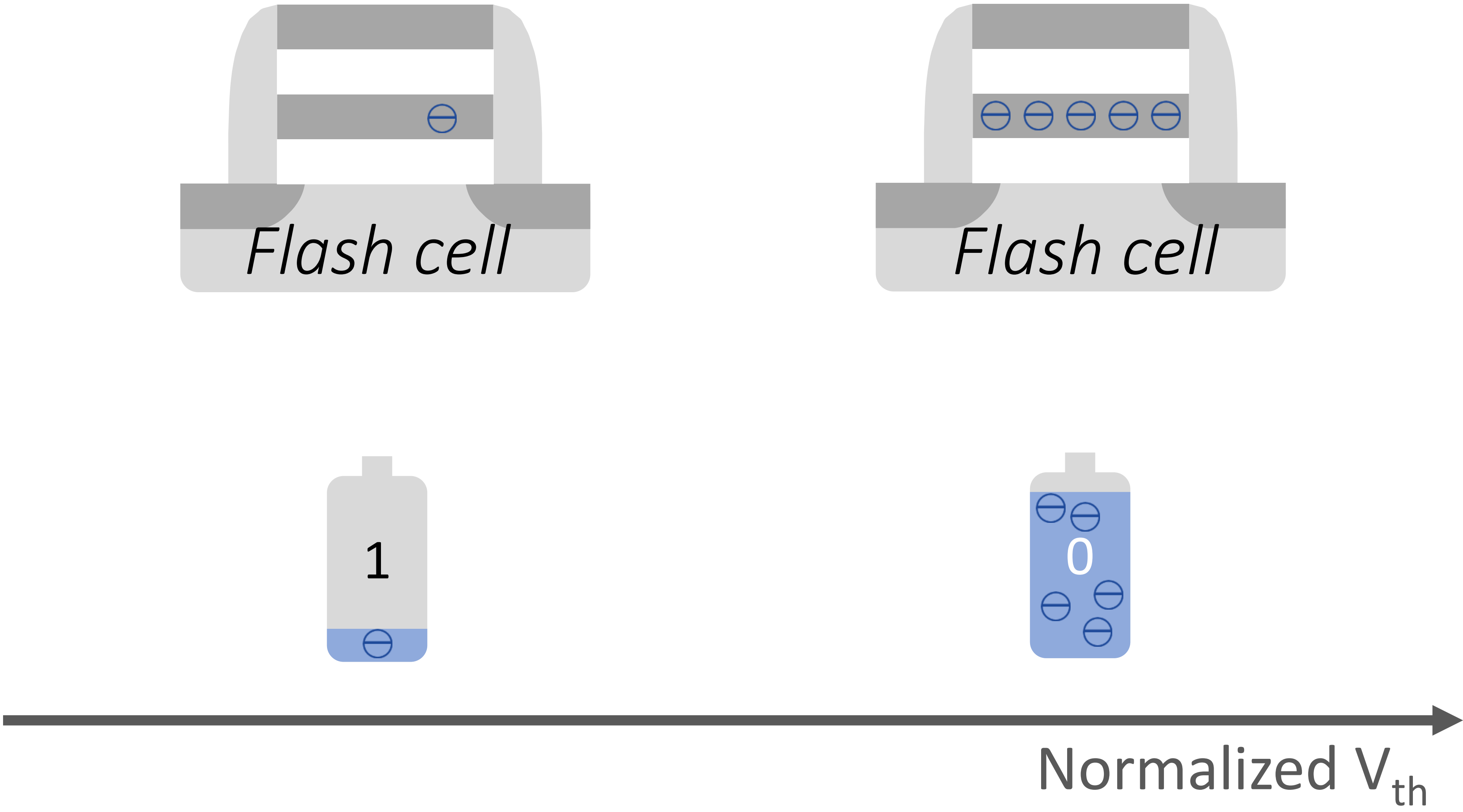
NOR



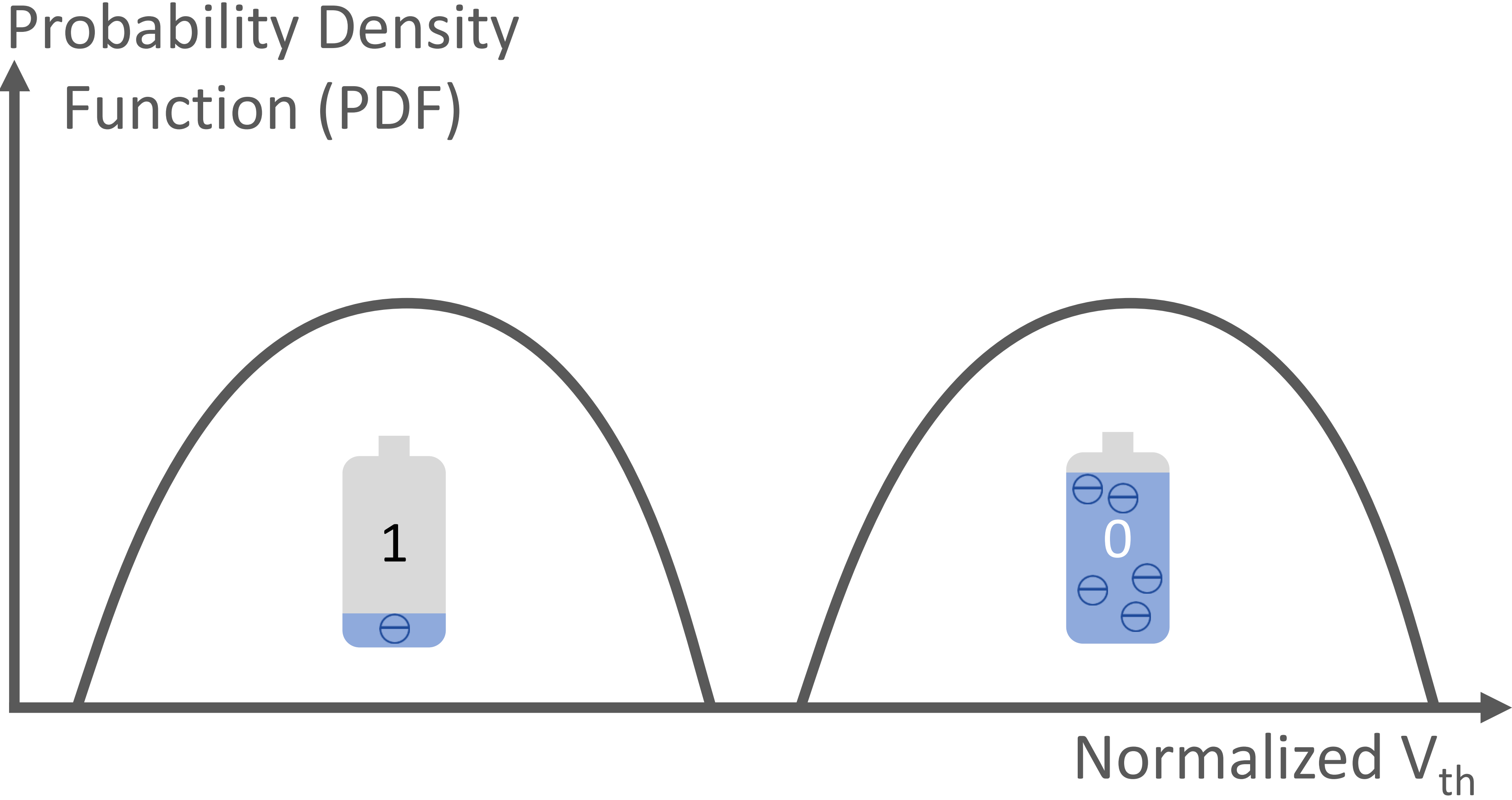
By wikipedia user Cyferz, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=4571194

By wikipedia user Cyferz, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=4571172

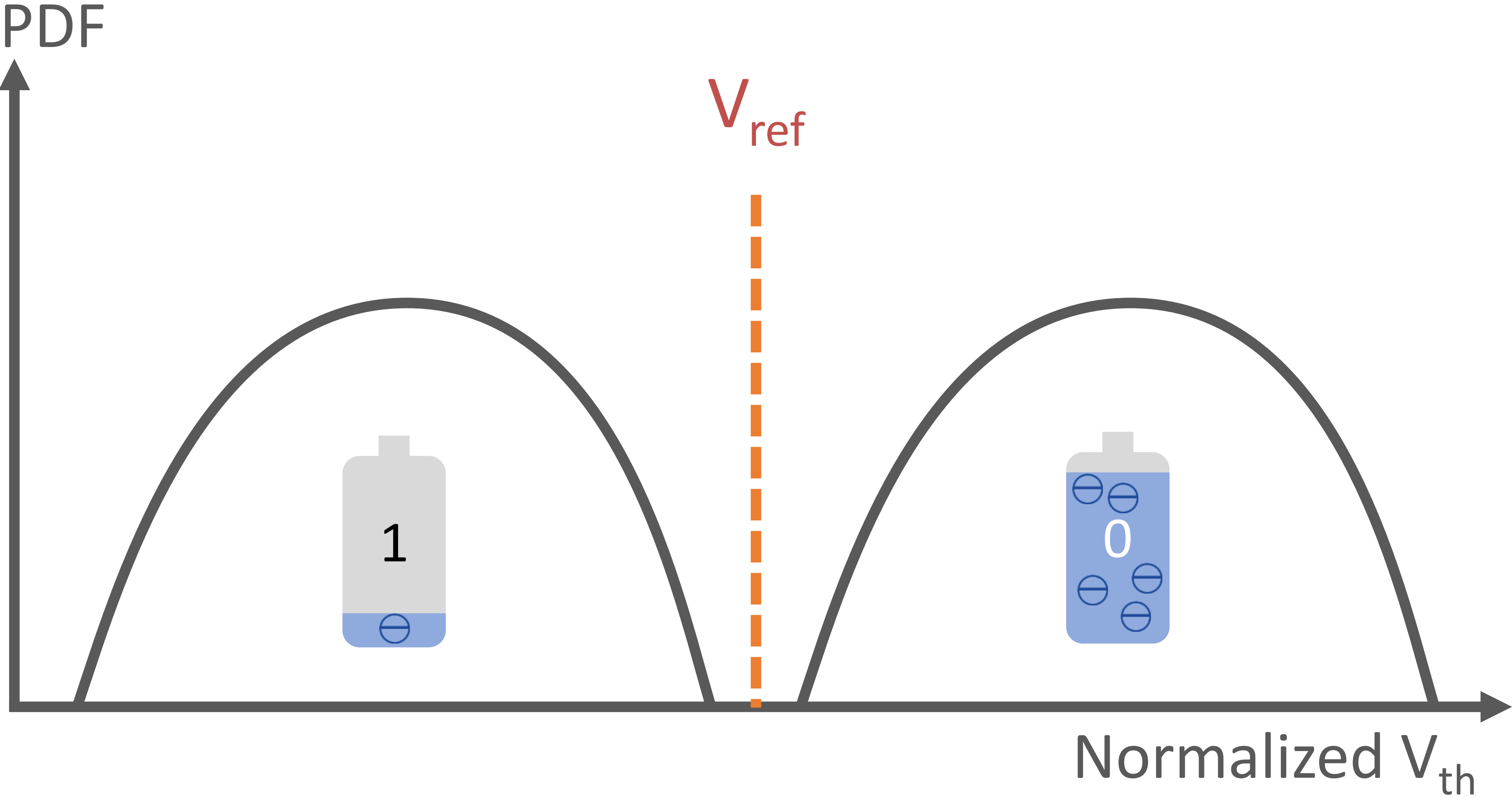
Threshold Voltage (V_{th})



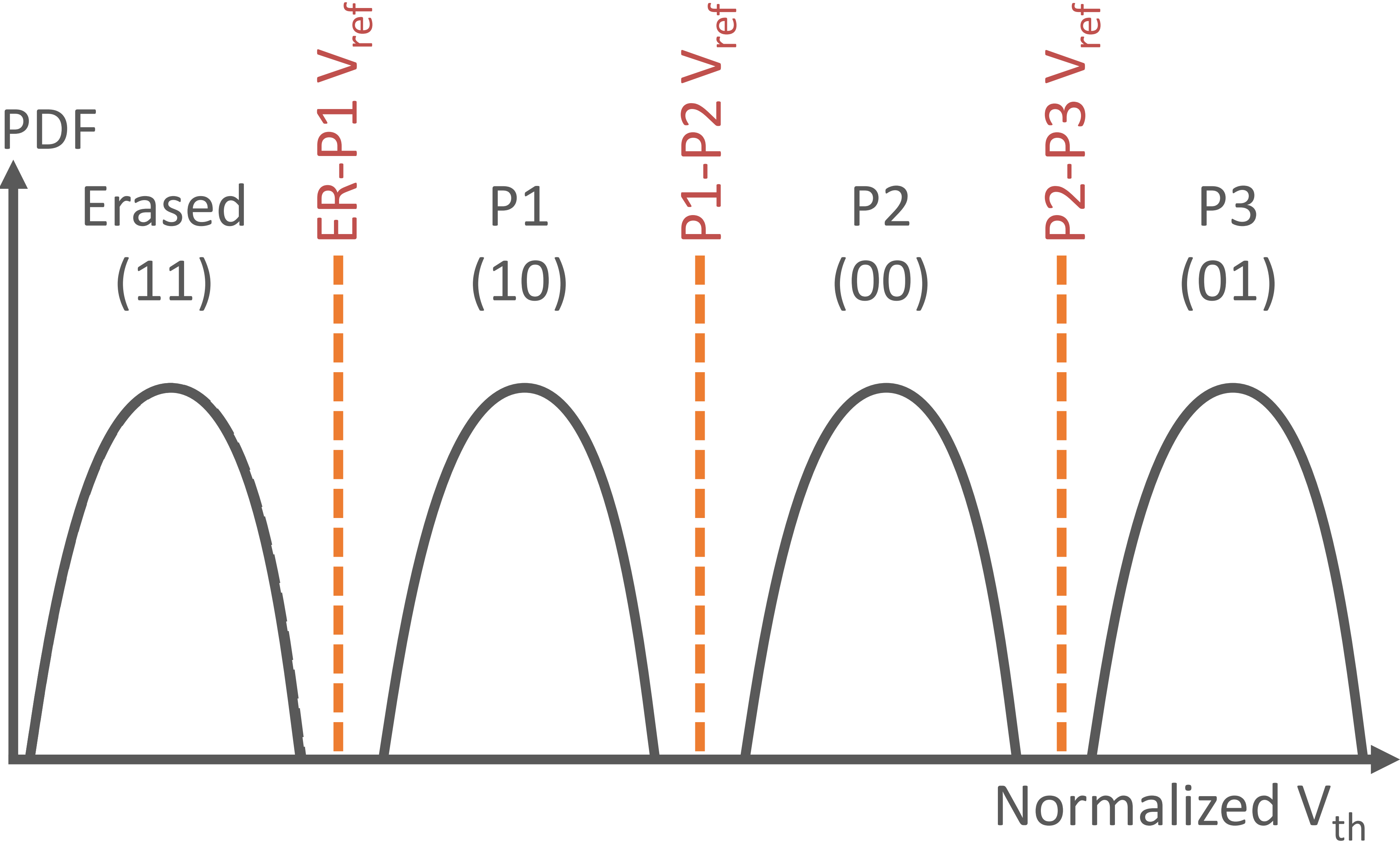
Threshold Voltage (V_{th}) Distribution



Read Reference Voltage (V_{ref})

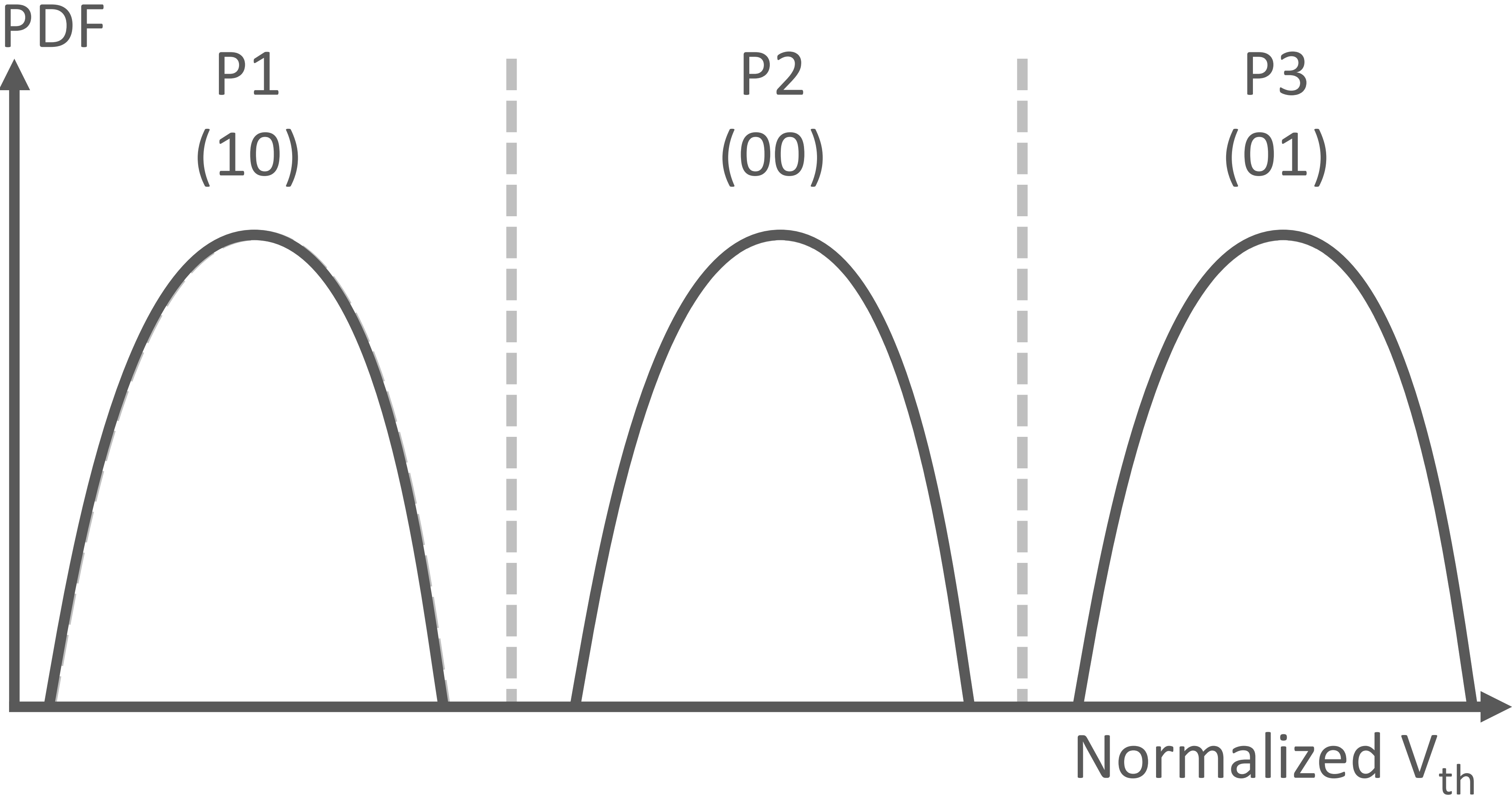


Multi-Level Cell (MLC)



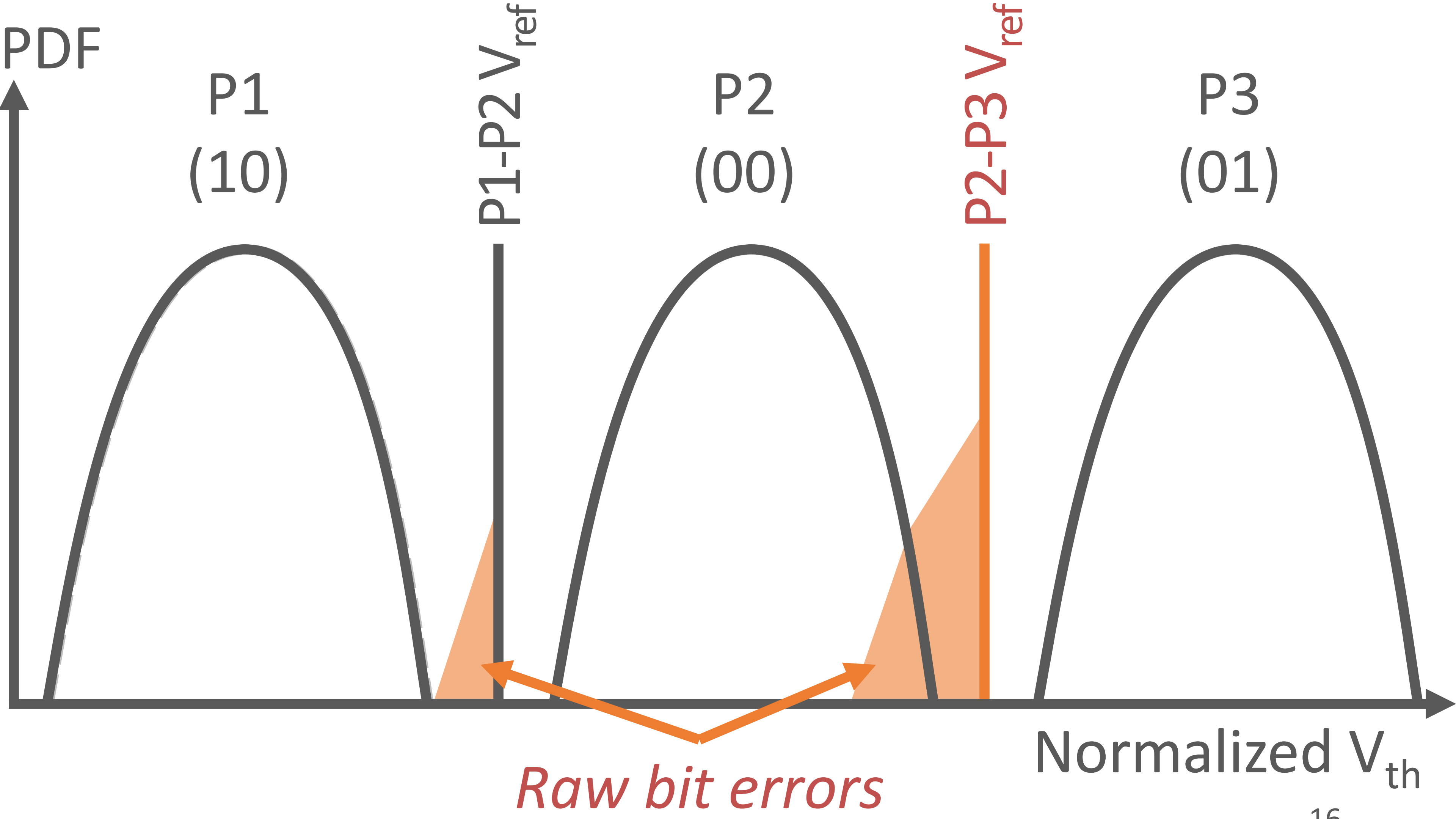
Threshold Voltage Reduces Over Time

After some retention loss:



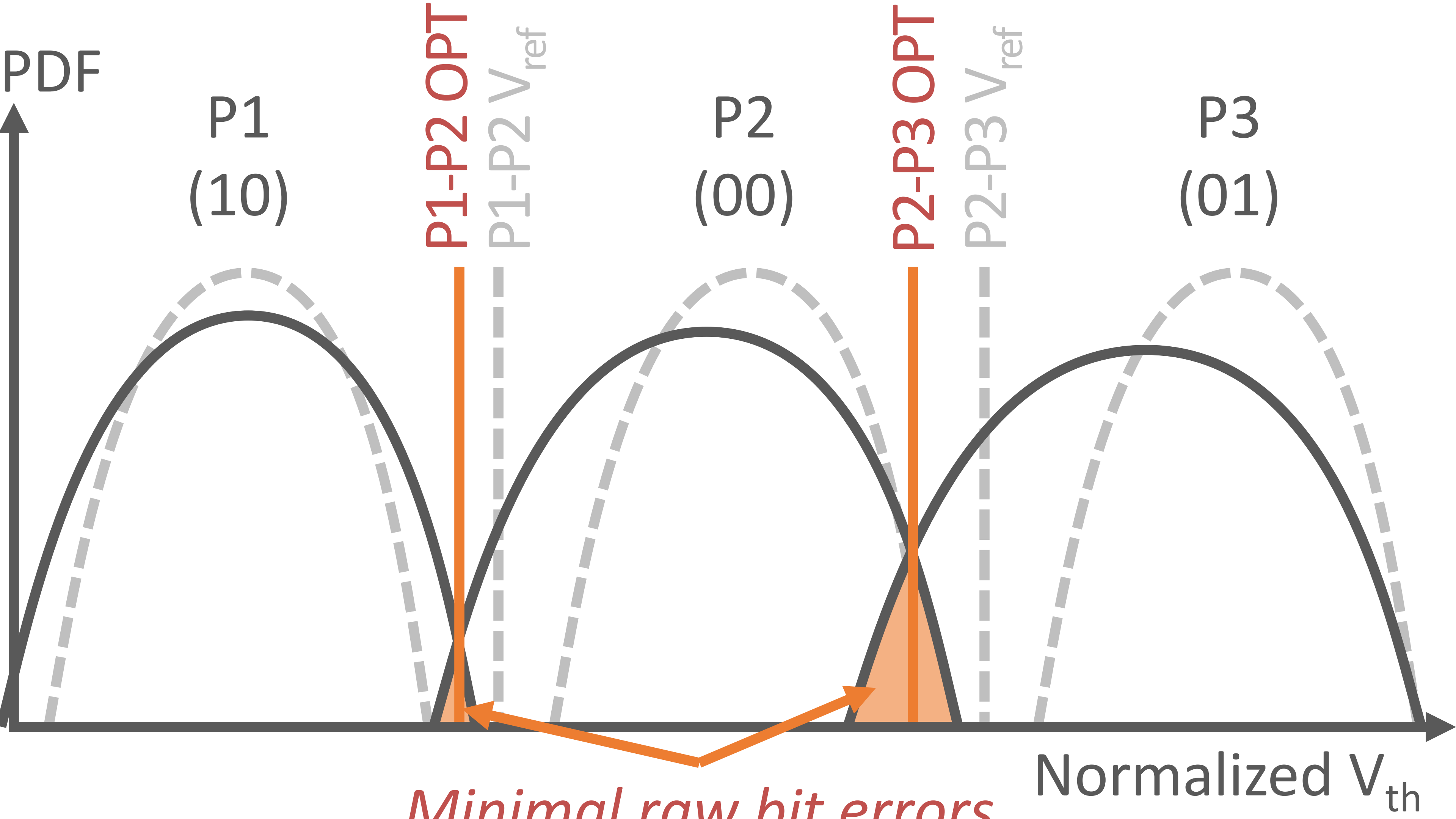
Fixed Read Reference Voltage Becomes

After some retention loss:



Optimal Read Reference Voltage (OPT)

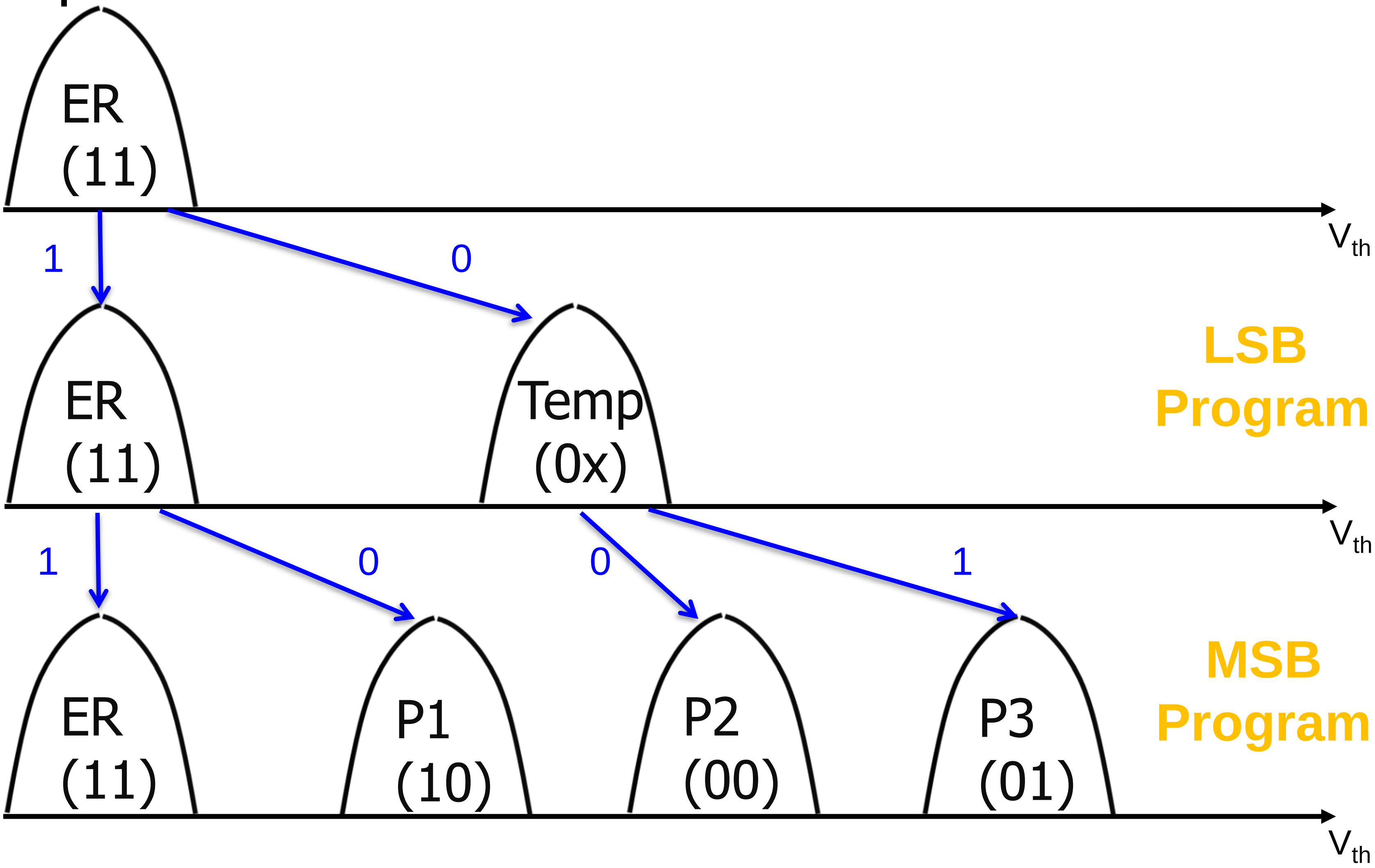
After some retention loss:



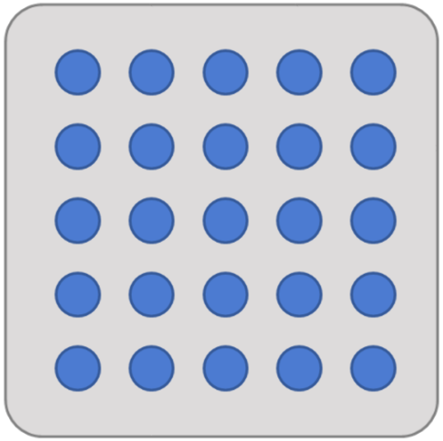
Minimal raw bit errors

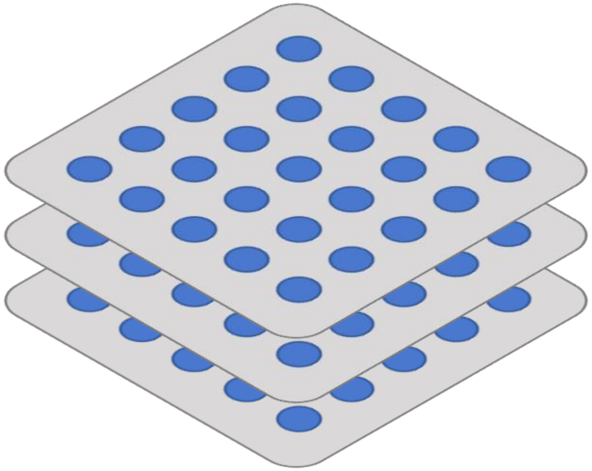
How Current Flash Cells are Programmed

- Programming 2-bit MLC NAND flash memory in two steps



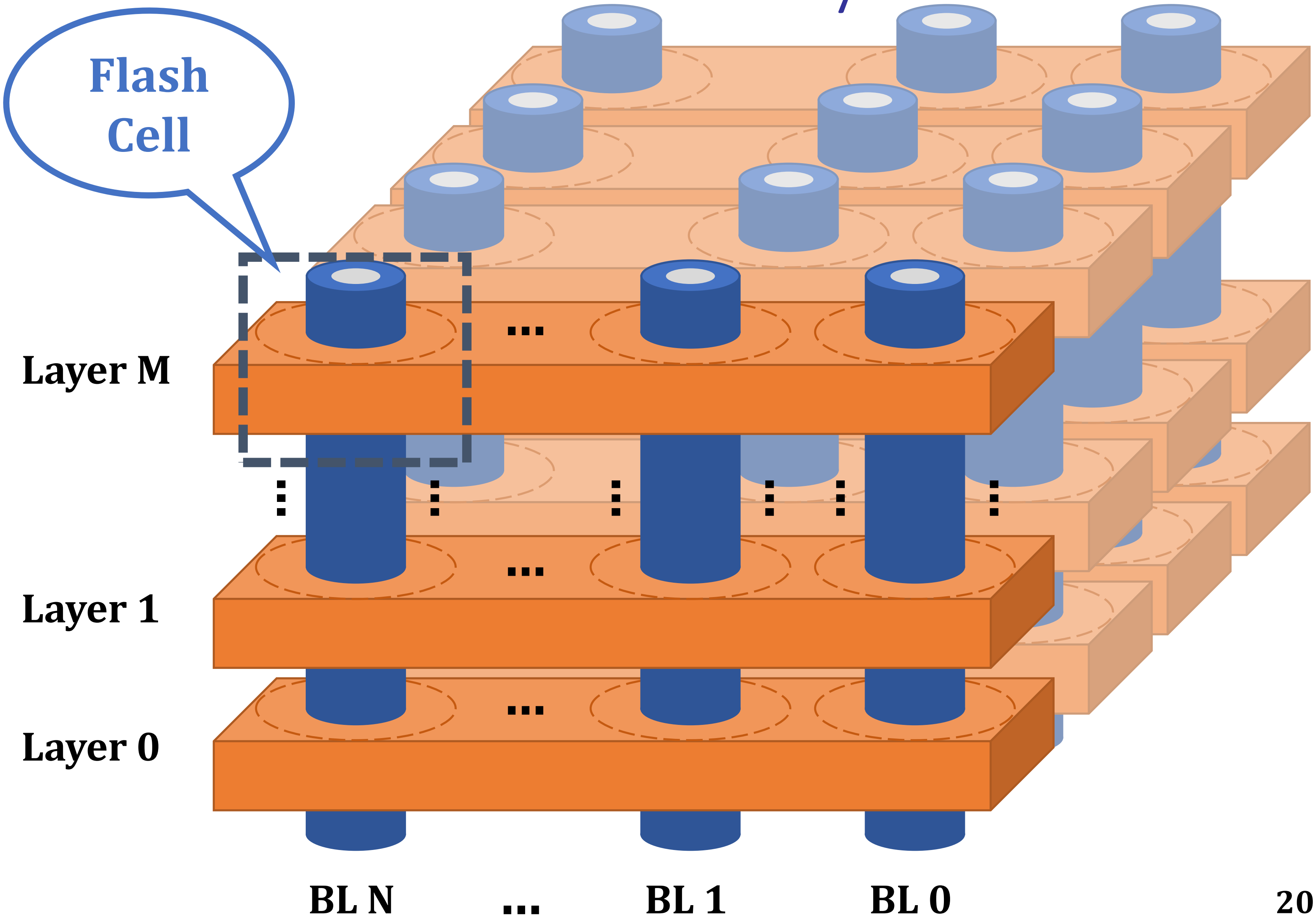
Planar vs. 3D NAND Flash Memory





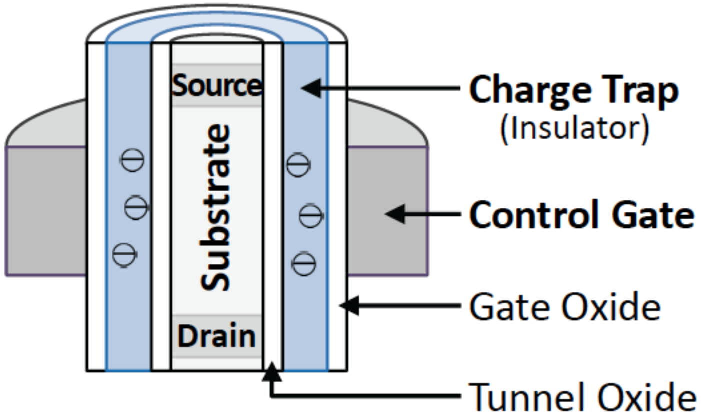
	Planar NAND Flash Memory	3D NAND Flash Memory
Scaling	Reduce flash cell size, Reduce distance b/w cells	Increase # of layers
Reliability	Scaling hurts reliability	**Not well studied!**

3D NAND Flash Memory Structure

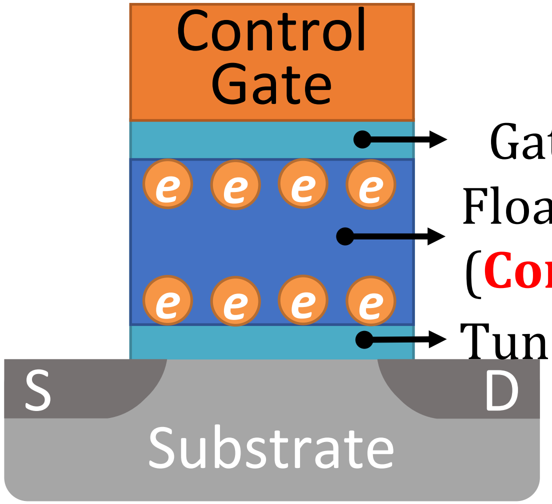


Charge Trap Based 3D Flash Cell

- Cross-section of a charge trap transistor



2D vs. 3D Flash Cell Design

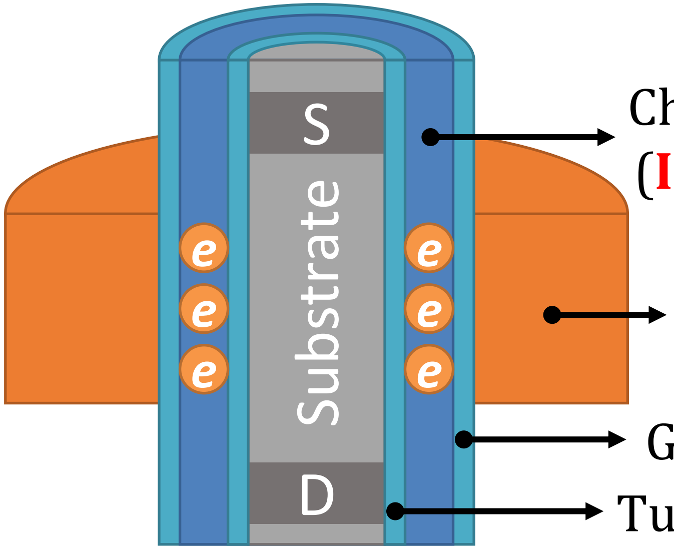


Gate Oxide

Floating Gate (**Conductor**)

Tunnel Oxide

2D Floating-Gate Cell



Charge Trap (**Insulator**)

Control Gate

Gate Oxide

Tunnel Oxide

3D Charge-Trap Cell

3D NAND Flash Memory Organization

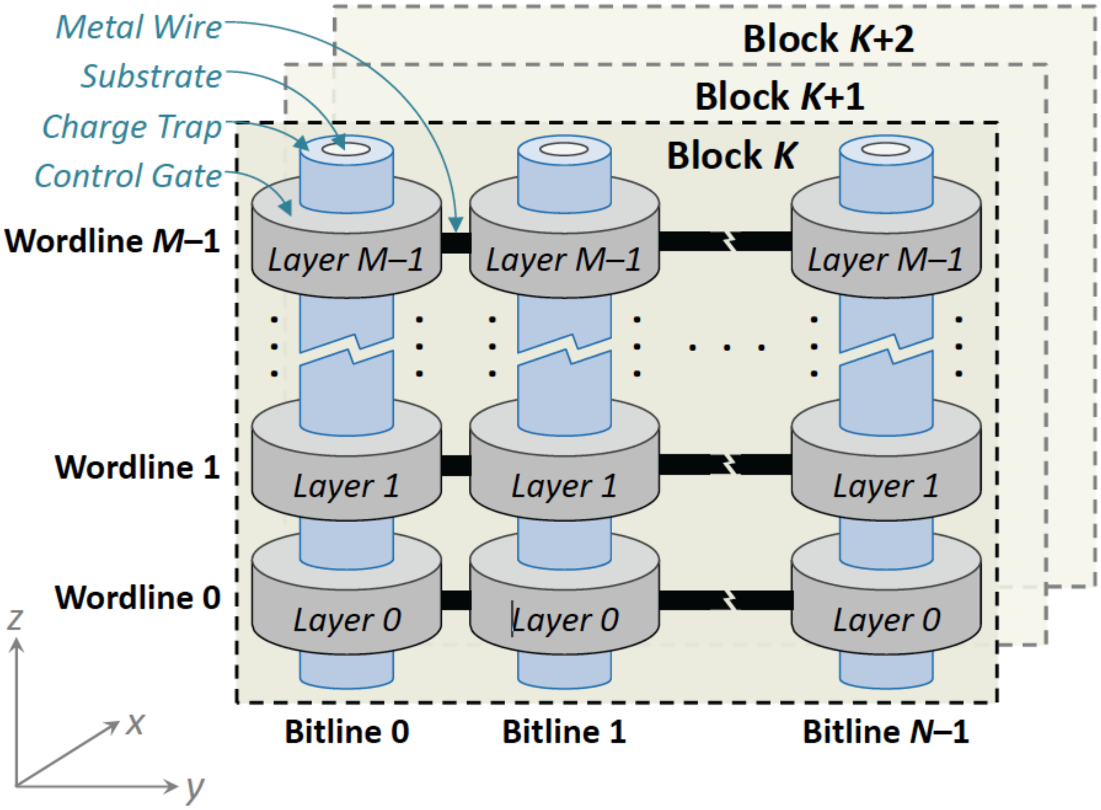


Fig. 43. Organization of flash cells in an *M*-layer 3D charge trap NAND flash memory chip, where each block consists of *M* wordlines and *N* bitlines.

Modern Solid-State Drive (SSD) Architecture

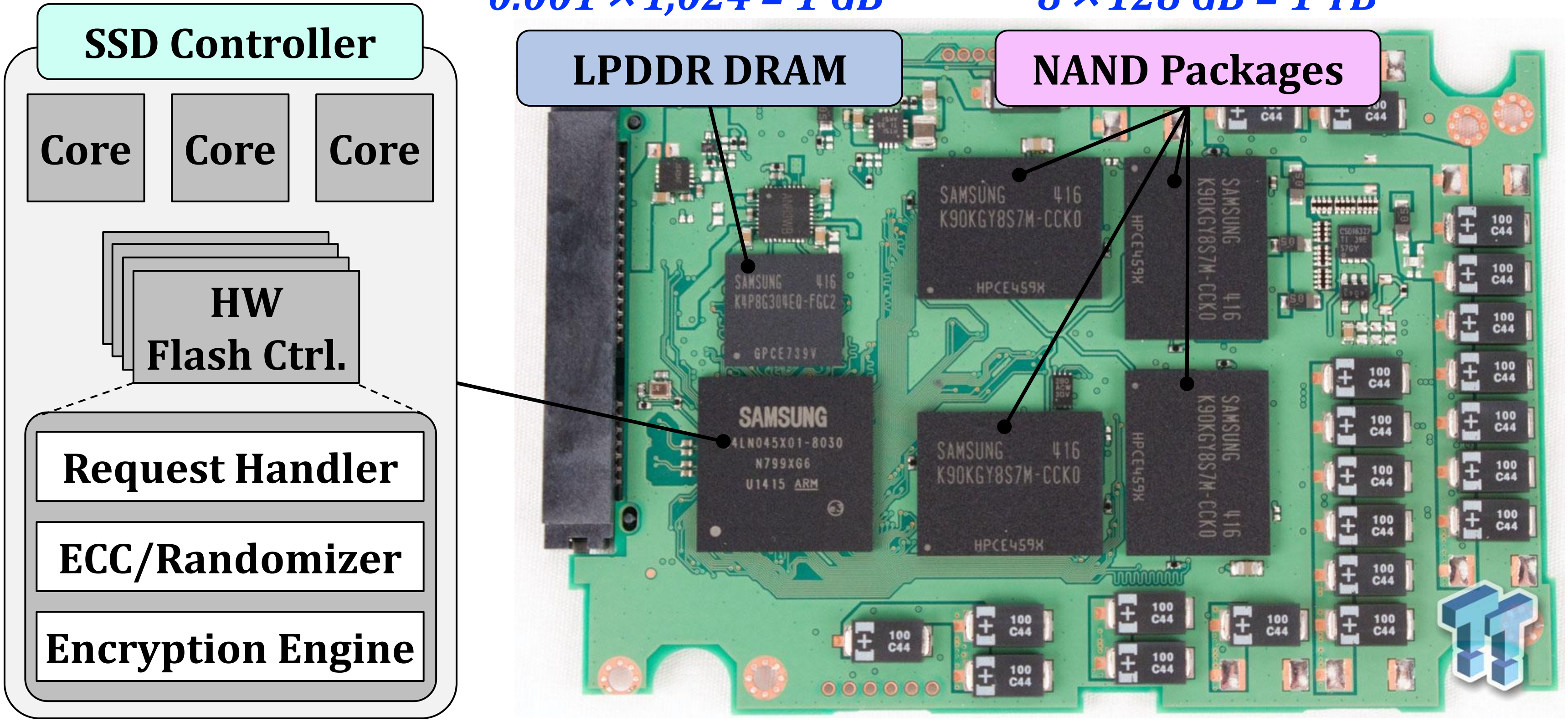
(本節內容改自Prof. Onur Mutlu, “Computer Architecture,” 17th Lecture, Fall 2023 課程講義)

Modern SSD Architecture

- A modern SSD is a complicated system that consists of multiple cores, HW controllers, DRAM, and NAND flash memory packages

0.001 ×1,024 = 1 GB

8 ×128 GB = 1 TB



SSD Controller

Core

Core

Core

HW Flash Ctrl.

Request Handler

ECC/Randomizer

Encryption Engine

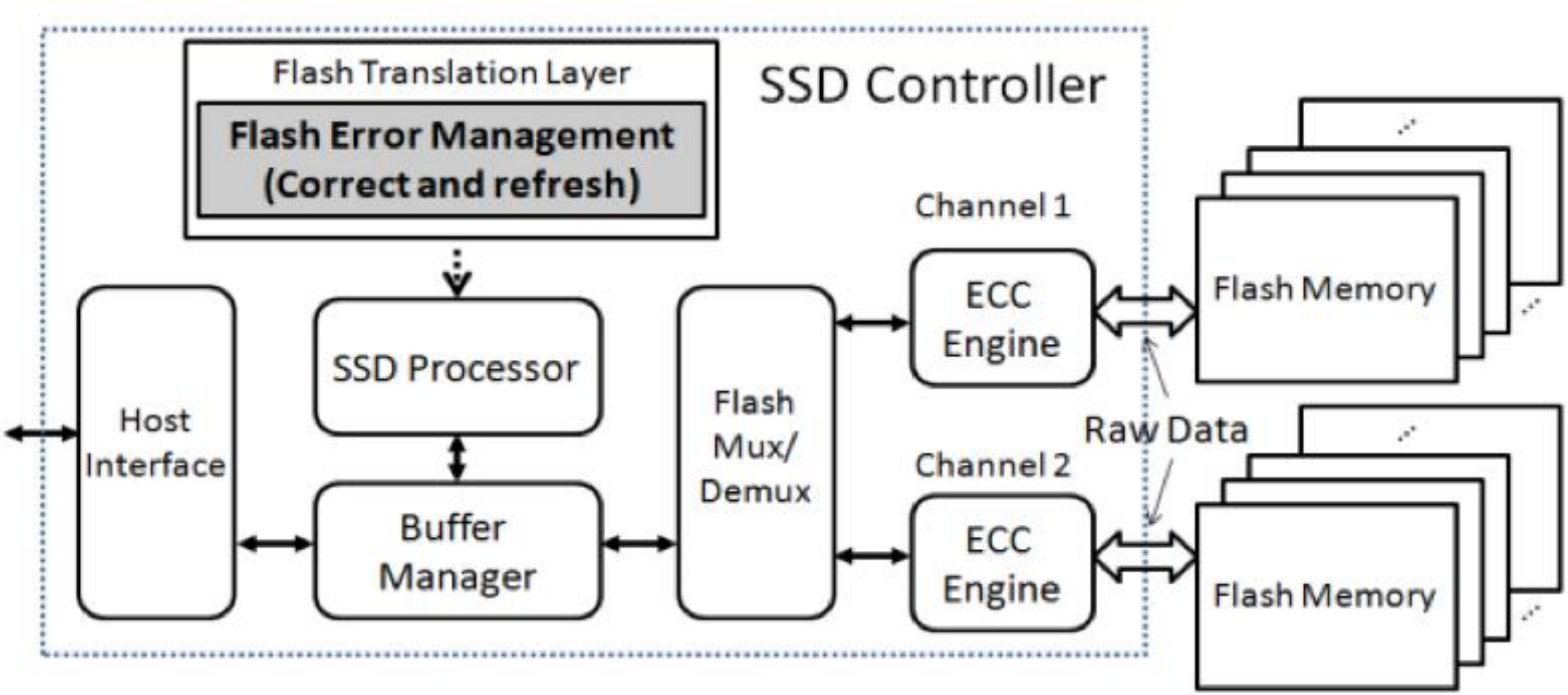
LPDDR DRAM

NAND Packages

Samsung PM853T 960GB Enterprise SSD (from https://www.tweaktown.com/reviews/6695/samsung-pm853t-960gb-enterprise-ssd-review/index.html)

Flash Memory (SSD) Controllers

- Similar to DRAM memory controllers, except:
  - They are flash memory specific
  - They do much more: complex error correction, wear leveling, voltage optimization, garbage collection, page remapping, …



Cai+, "Flash Correct-and-Refresh: Retention-Aware Error Management for Increased Flash Memory Lifetime", ICCD 2012.

[Slides Credit] Prof. Onur Mutlu, "Computer Architecture," 12th Lecture, Fall 2023

Another View of the SSD Controller

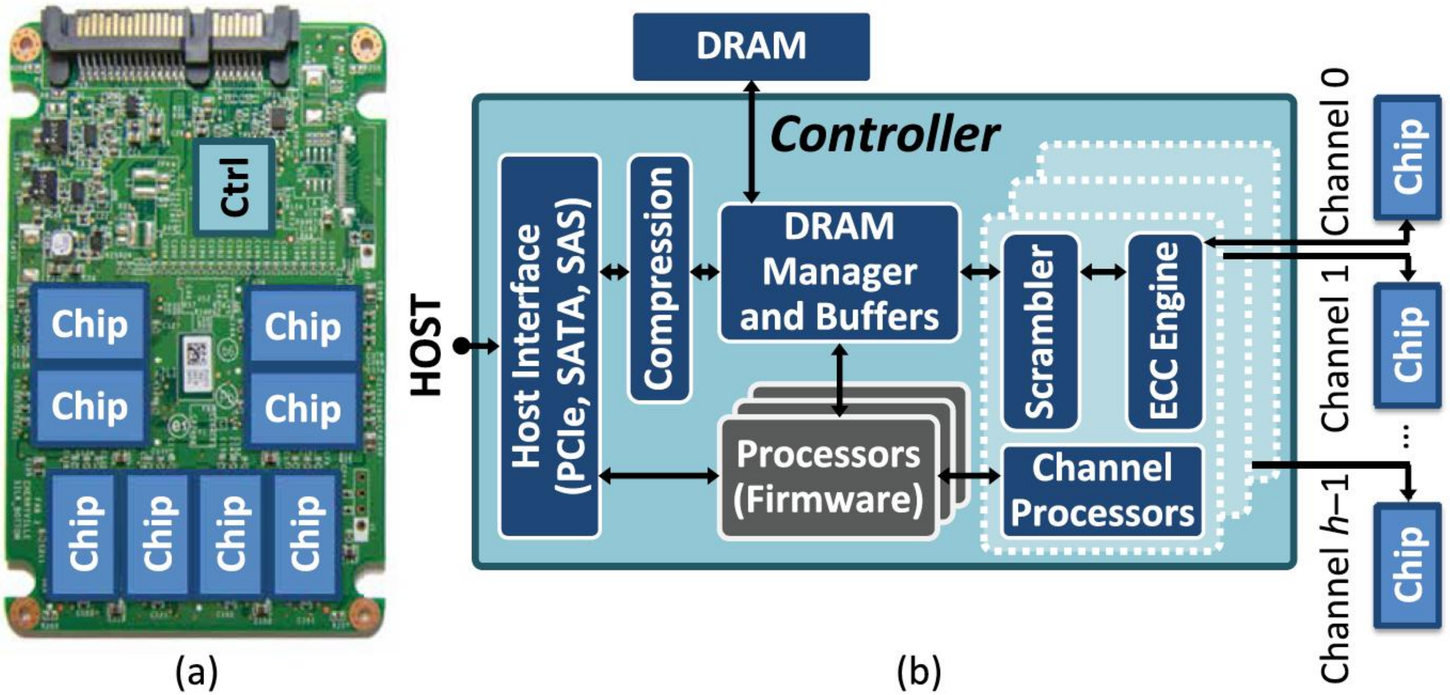
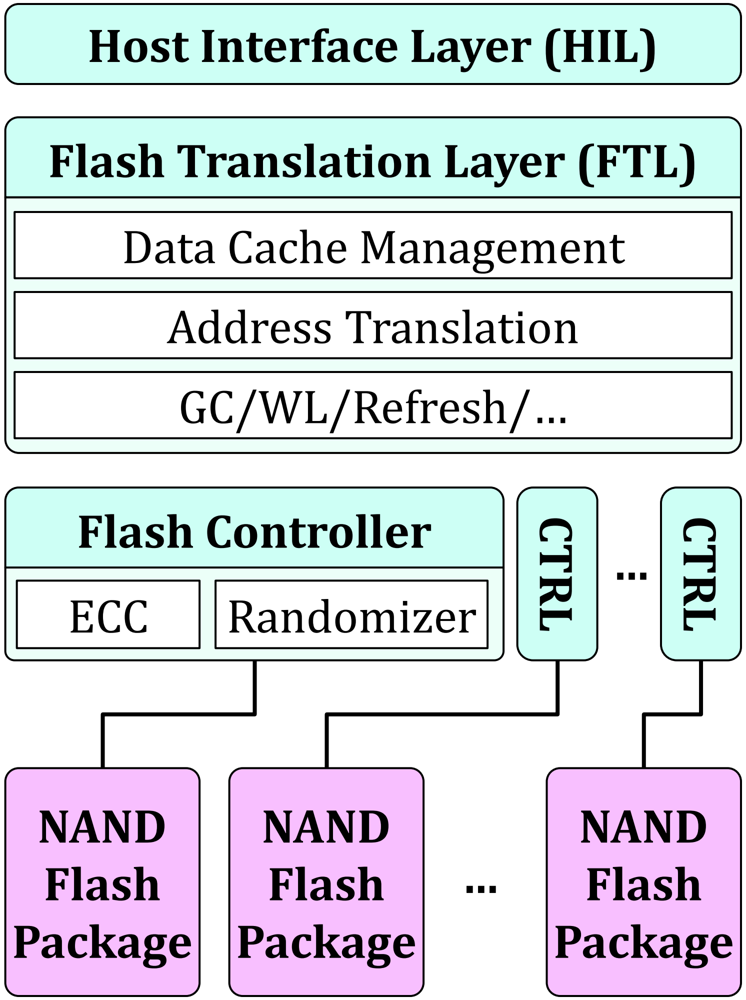


Fig. 1. ***(a) SSD system architecture, showing controller (Ctrl) and chips. (b) Detailed view of connections between controller components and chips.***

Cai+, "Error Characterization, Mitigation, and Recovery in Flash Memory Based Solid State Drives," Proc. IEEE 2017.

[Slides Credit] Prof. Onur Mutlu, "Computer Architecture," 12th Lecture, Fall 2023

Another Overview



Host Interface Layer (HIL)

Flash Translation Layer (FTL)

- Data Cache Management
- Address Translation
- GC/WL/Refresh/…

Flash Controller

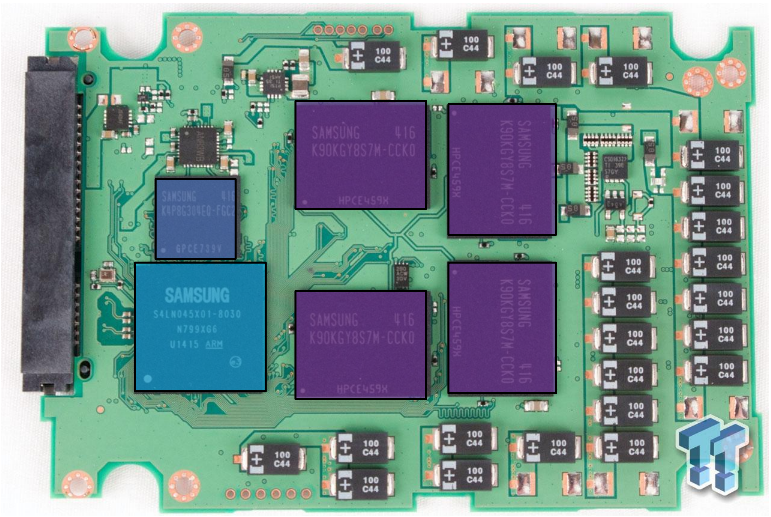
- ECC
- Randomizer

CTRL … CTRL

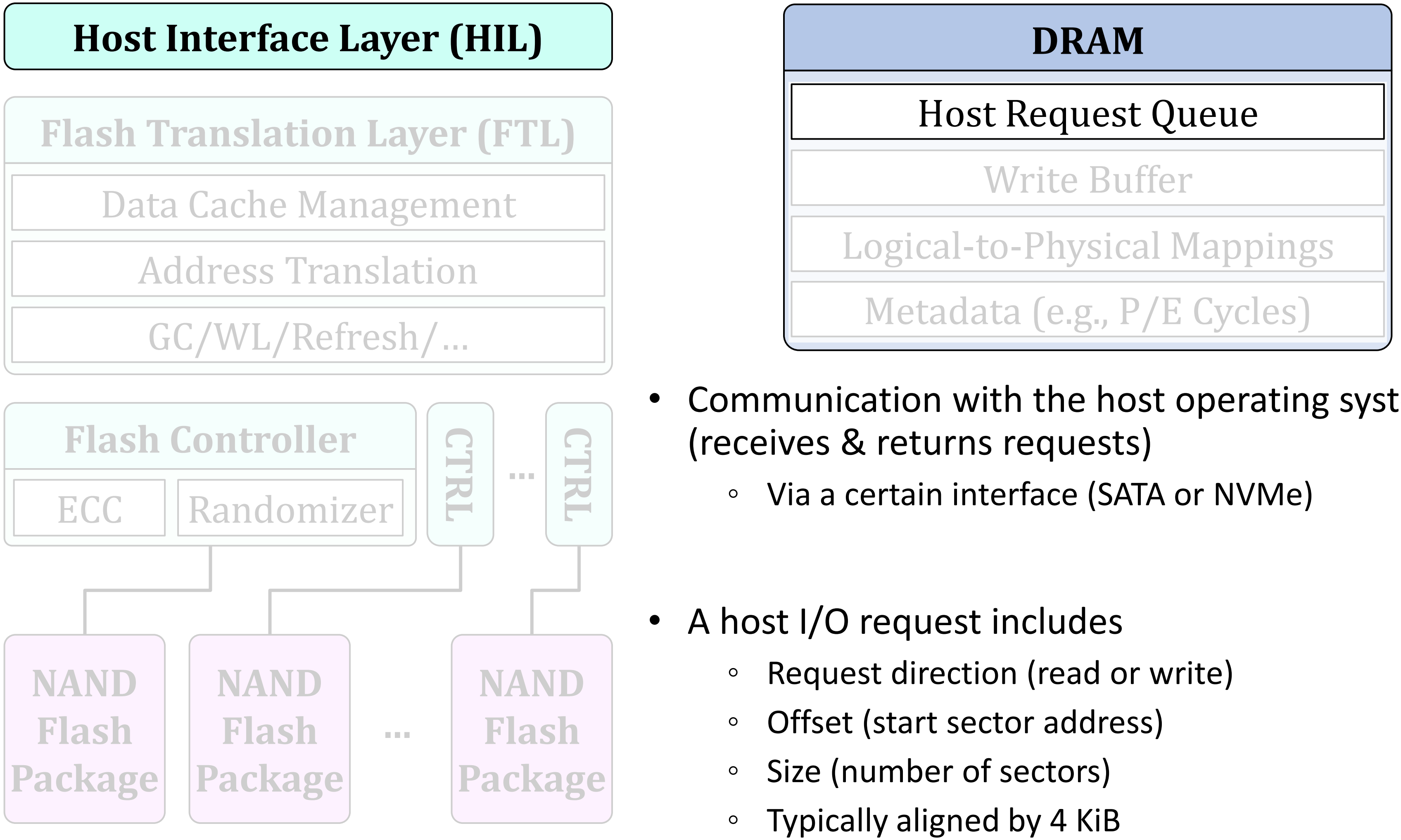
NAND Flash Package | NAND Flash Package | … | NAND Flash Package

DRAM

- Host Request Queue
- Write Buffer
- Logical-to-Physical Mappings
- Metadata (e.g., P/E Cycles)



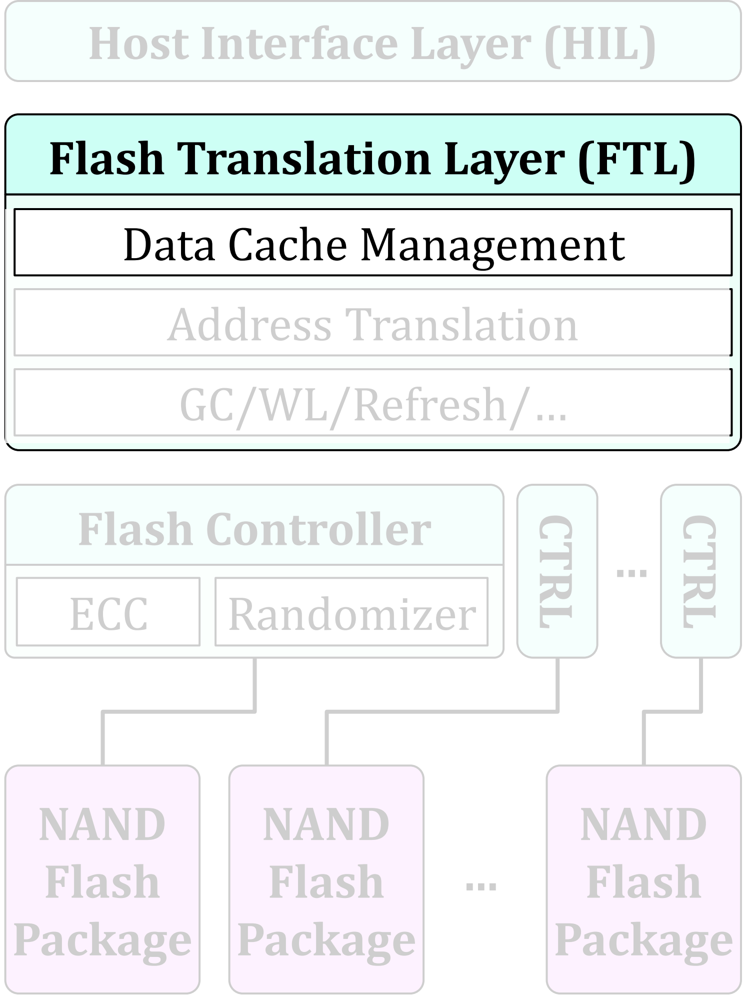
Request Handling: Write

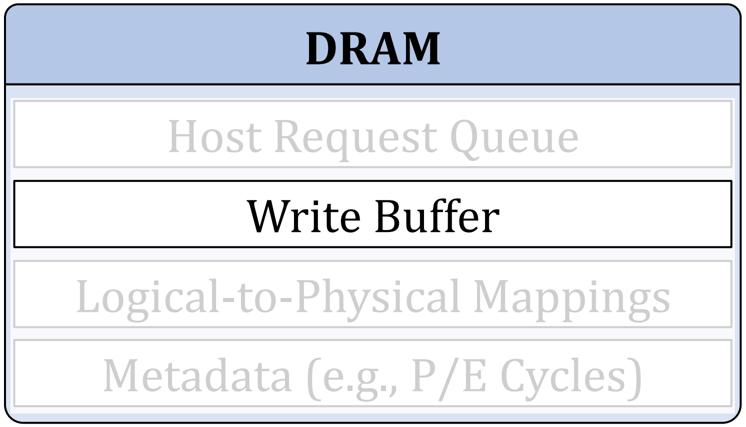


- Communication with the host operating system (receives & returns requests)
    - Via a certain interface (SATA or NVMe)

- A host I/O request includes
    - Request direction (read or write)
    - Offset (start sector address)
    - Size (number of sectors)
    - Typically aligned by 4 KiB

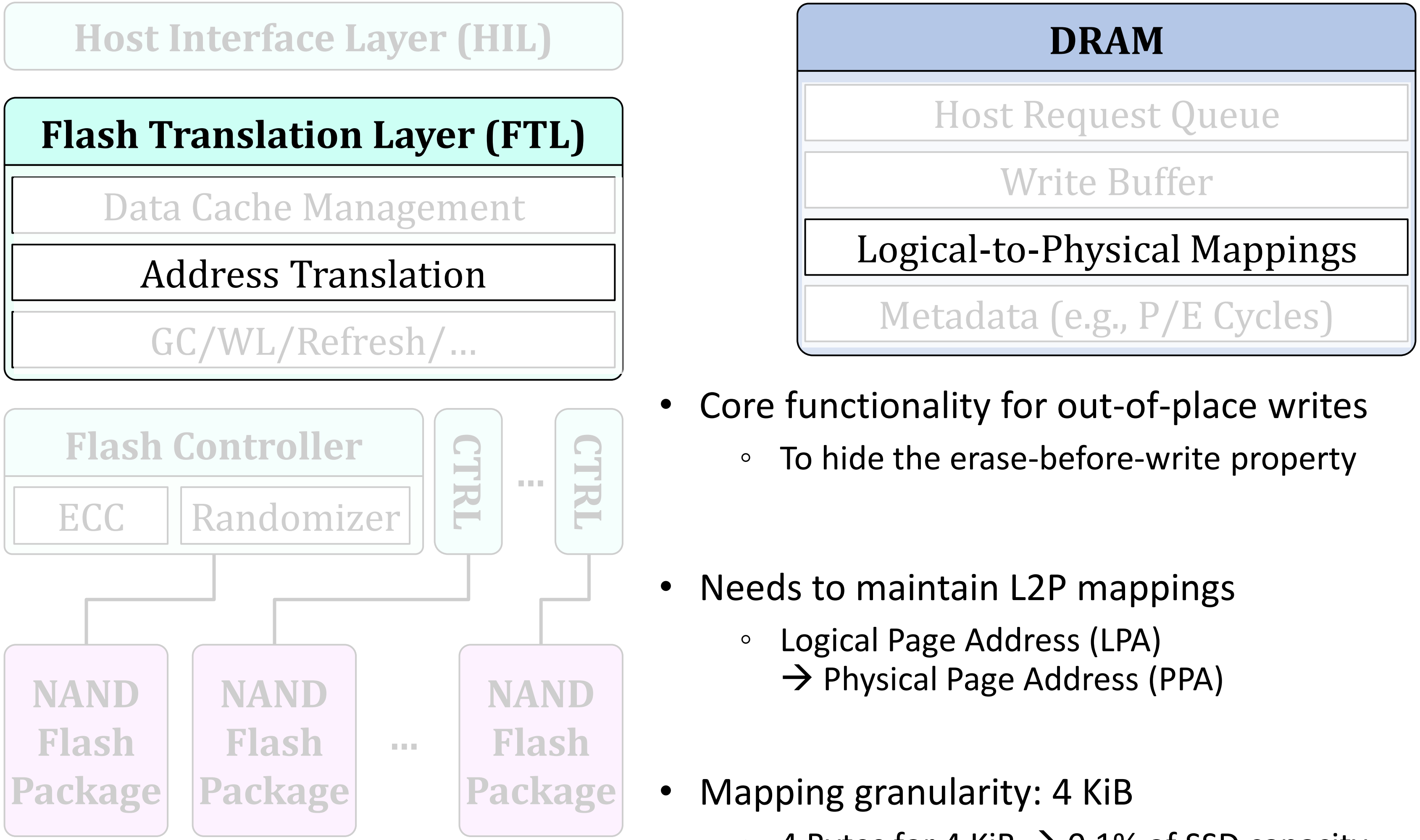
Request Handling: Write





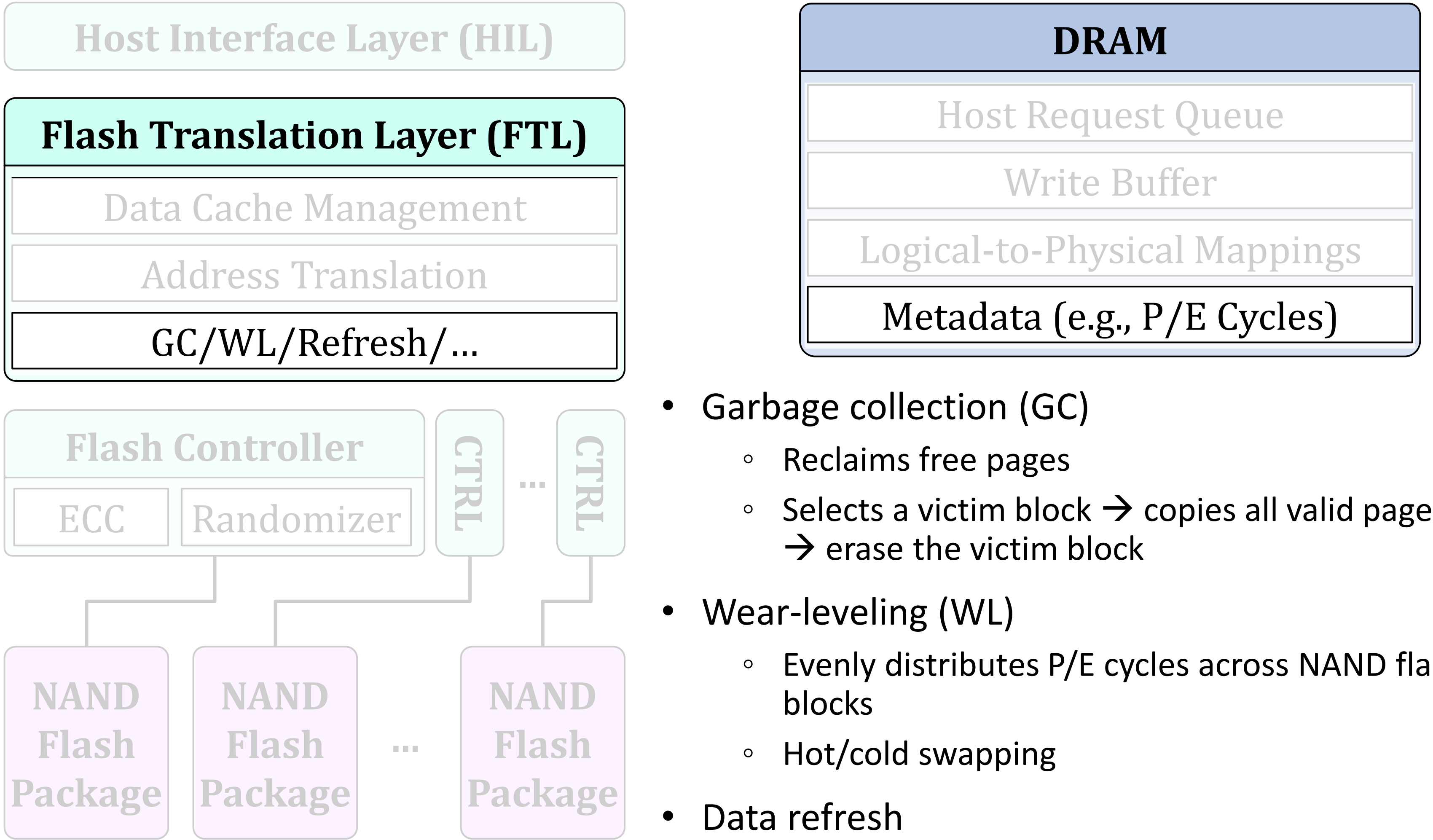
- Buffering data to write (read from NAND flash memory)
  - Essential to reducing write latency
  - Enables flexible I/O scheduling
  - Helpful for improving lifetime (not so likely)
- Limited size (e.g., tens of MBs)
  - Needs to ensure data integrity even under sudden power-off
  - Most DRAM capacity is used for L2P mappings

Request Handling: Write



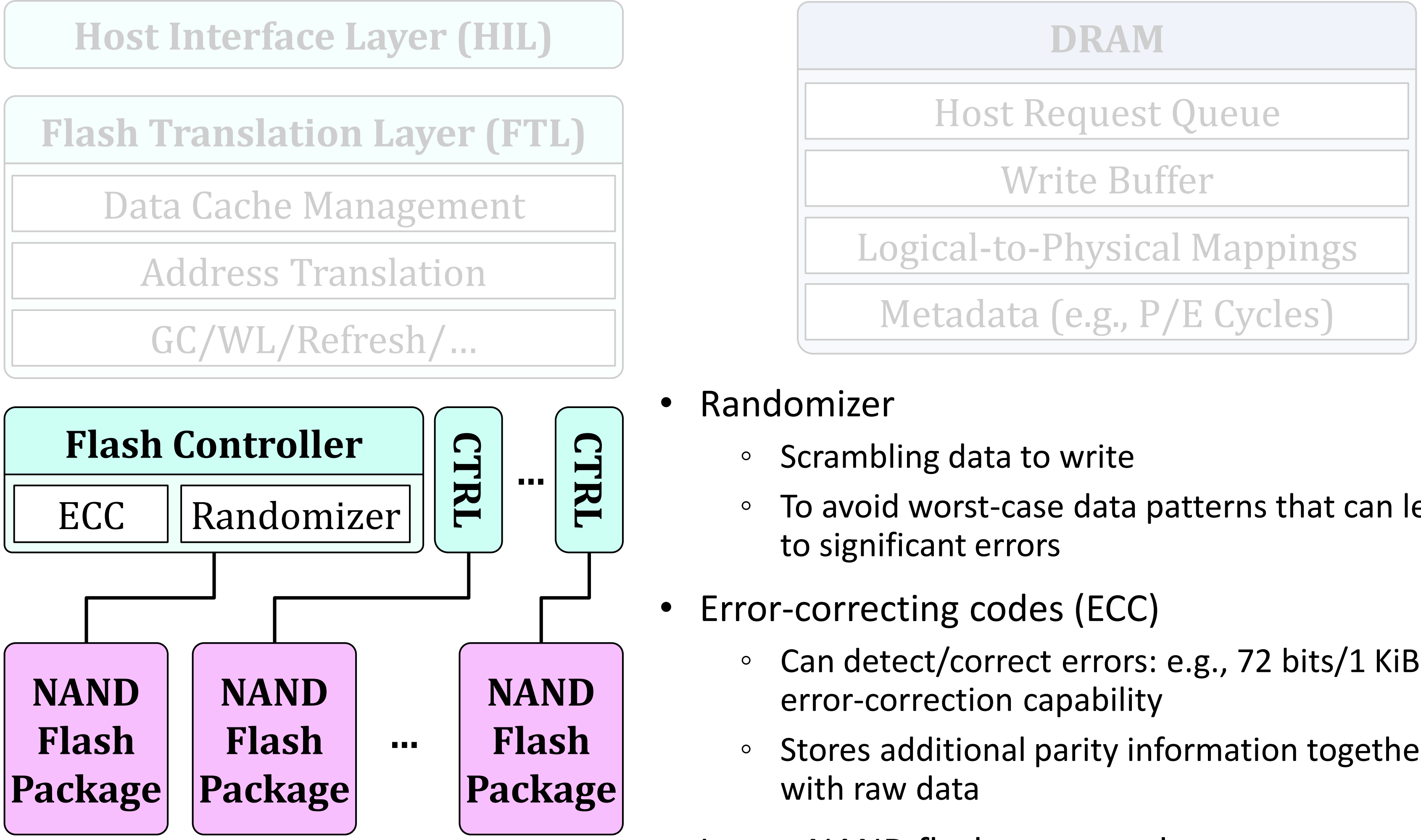
- Core functionality for out-of-place writes
  - To hide the erase-before-write property
- Needs to maintain L2P mappings
  - Logical Page Address (LPA) → Physical Page Address (PPA)
- Mapping granularity: 4 KiB
  - 4 Bytes for 4 KiB → 0.1% of SSD capacity

Request Handling: Write



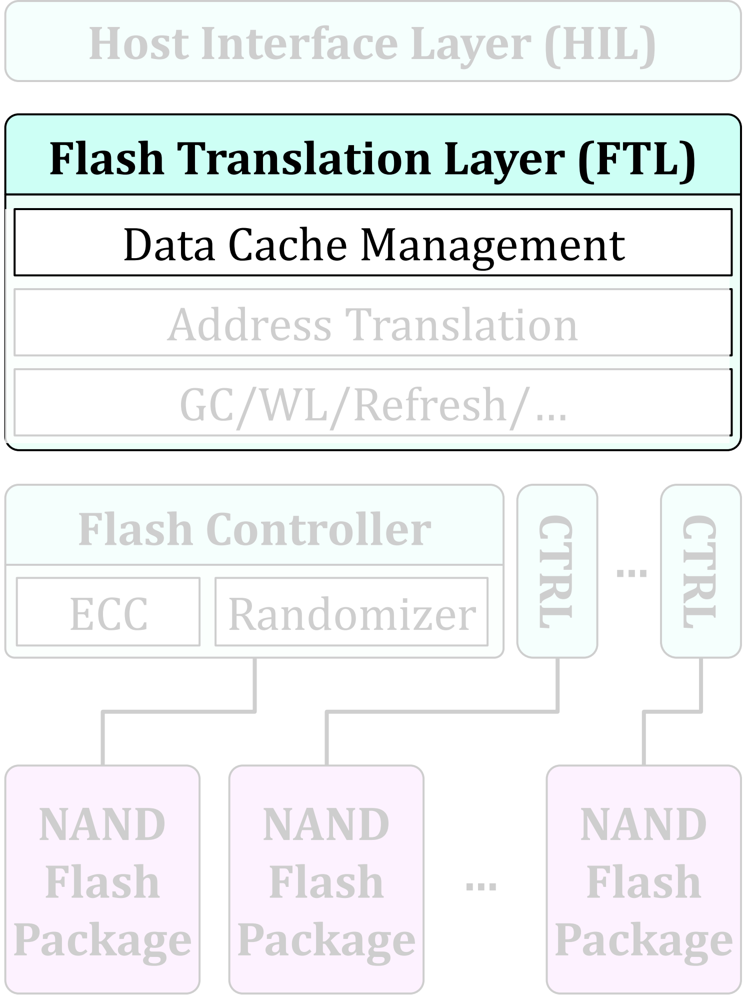
- Garbage collection (GC)
  - Reclaims free pages
  - Selects a victim block → copies all valid pages → erase the victim block
- Wear-leveling (WL)
  - Evenly distributes P/E cycles across NAND flash blocks
  - Hot/cold swapping
- Data refresh
  - Refresh pages with long retention ages

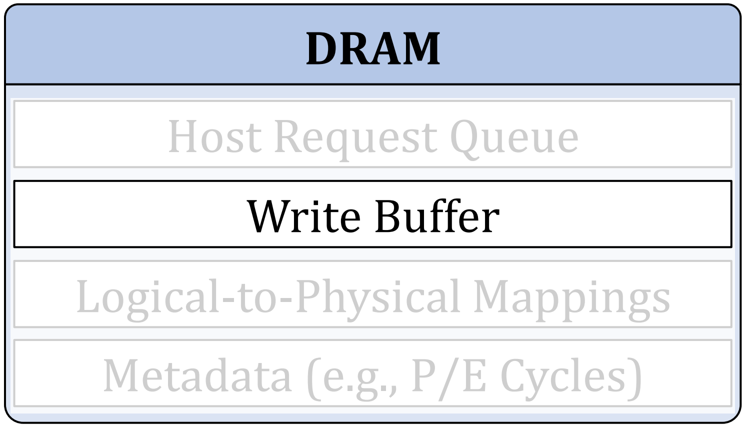
Request Handling: Write



- Randomizer
  - Scrambling data to write
  - To avoid worst-case data patterns that can lead to significant errors
- Error-correcting codes (ECC)
  - Can detect/correct errors: e.g., 72 bits/1 KiB error-correction capability
  - Stores additional parity information together with raw data
- Issues NAND flash commands

Request Handling: Read

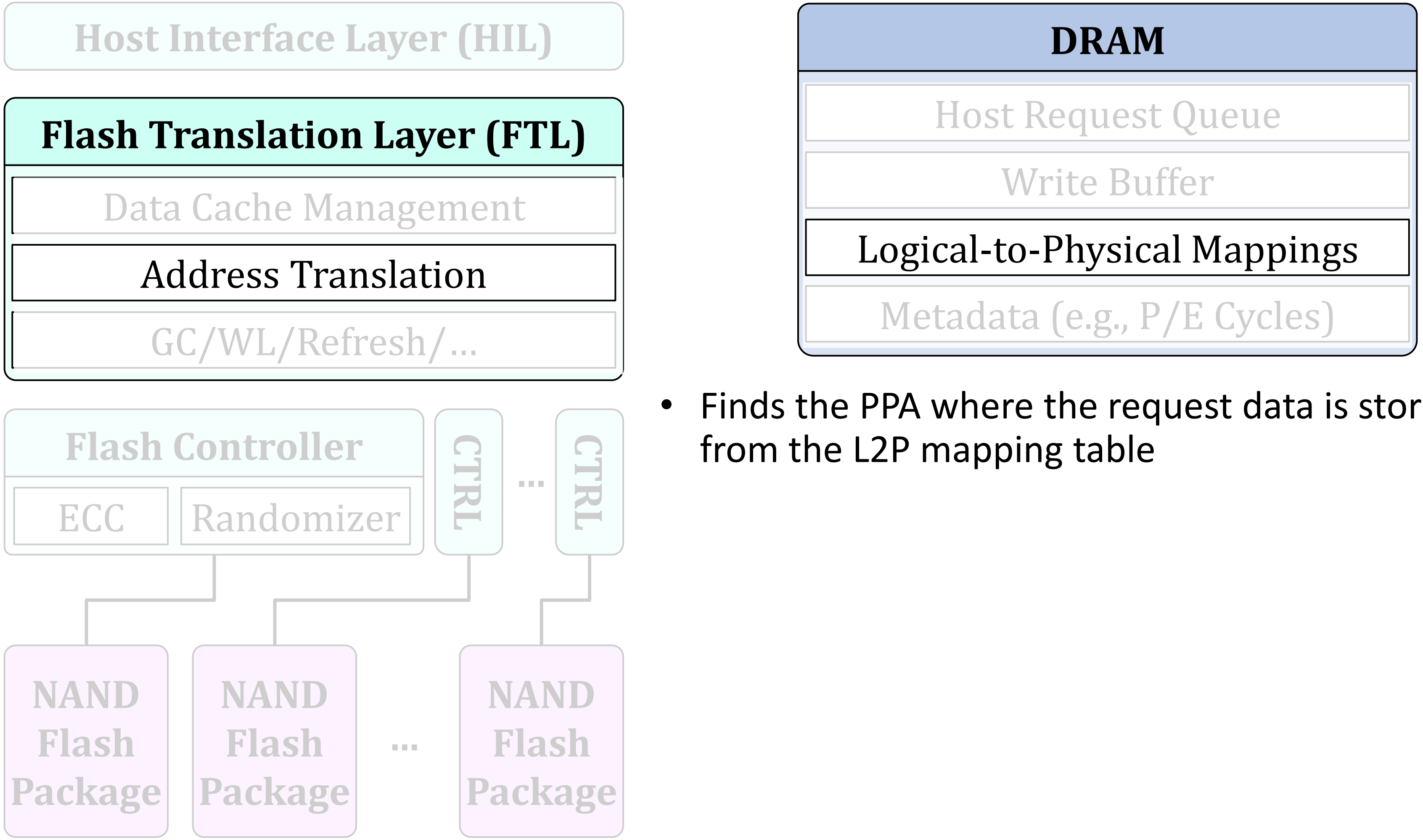




- First checks if the request data exists in the write buffer
  - If so, returns the corresponding request immediately with the data

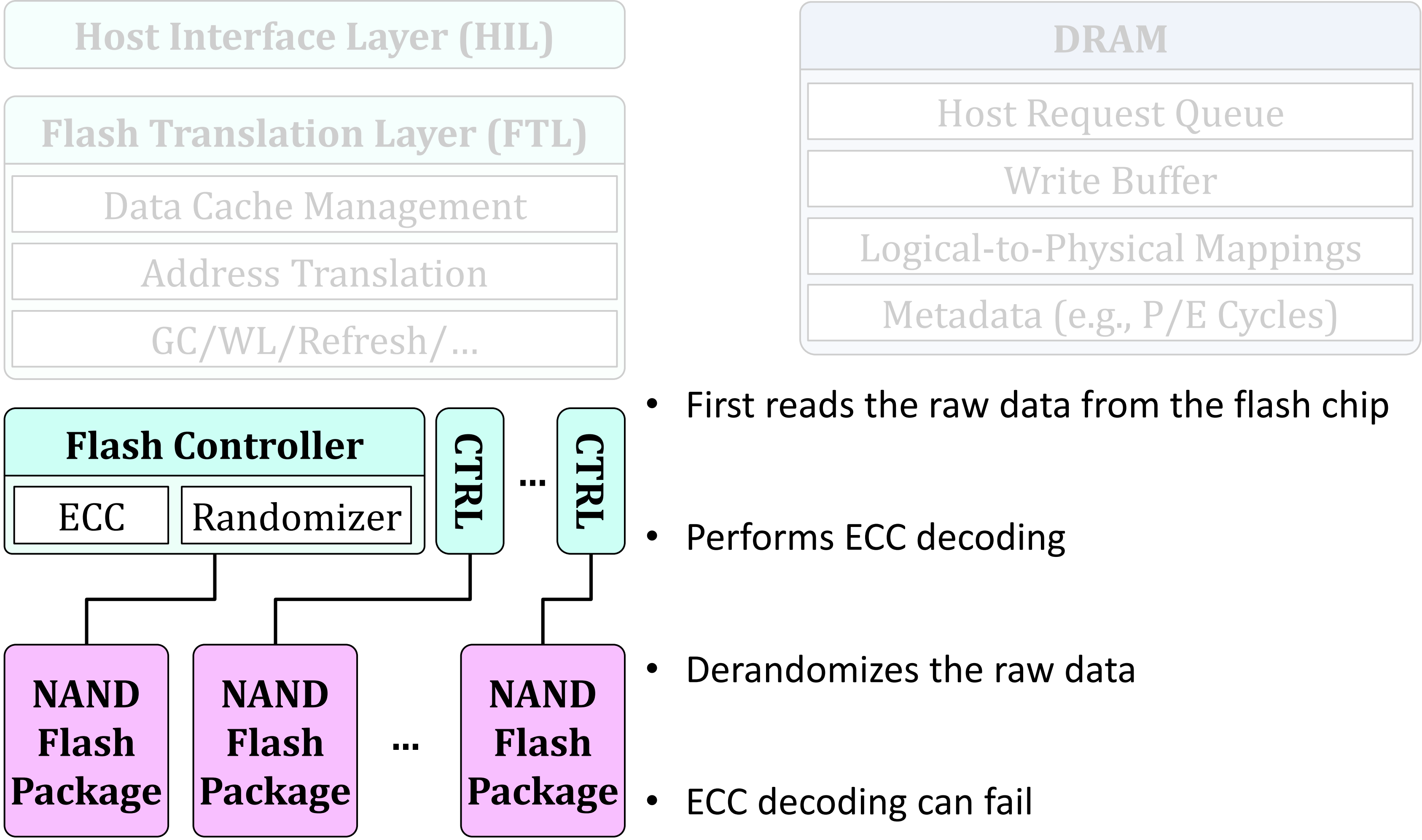
- A host read request can be involved with several pages
  - Such a request can be returned only after all the requested data is ready

Request Handling: Read



- Finds the PPA where the request data is stored from the L2P mapping table

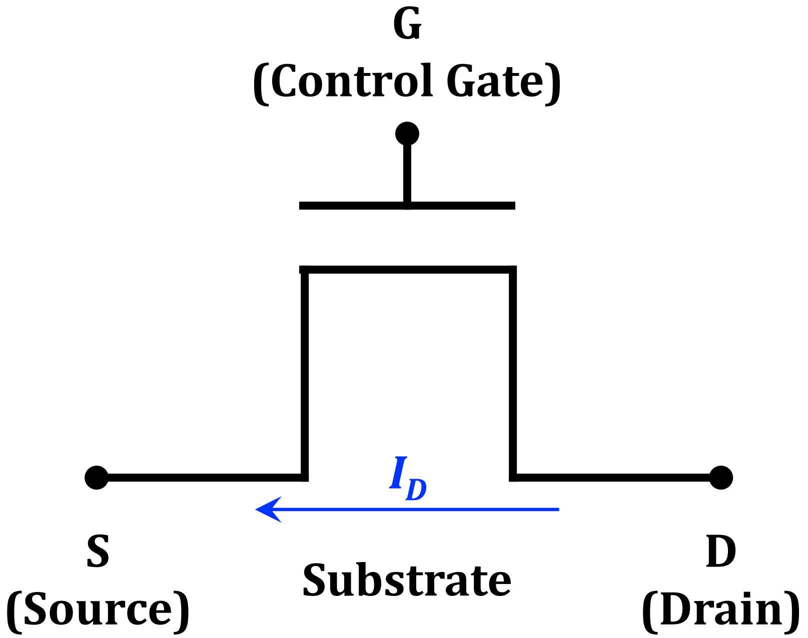
Request Handling: Read

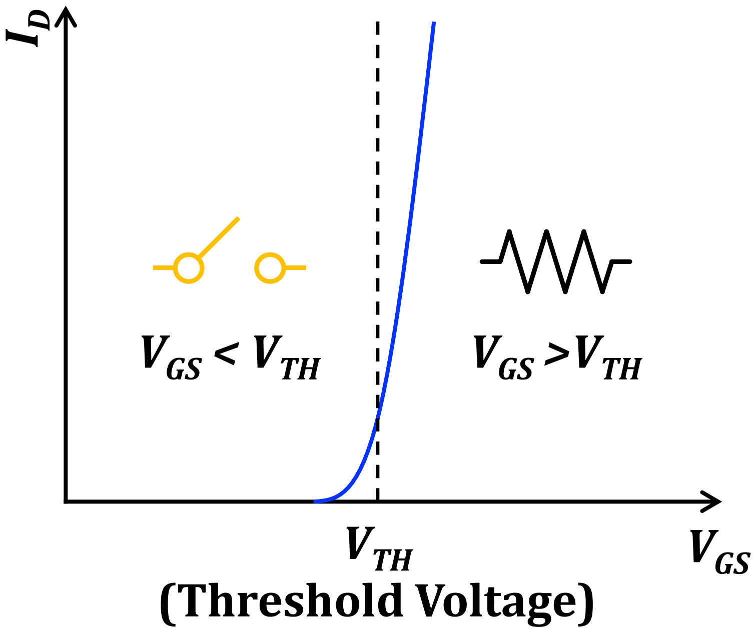


- First reads the raw data from the flash chip
- Performs ECC decoding
- Derandomizes the raw data
- ECC decoding can fail
  - Retries reading of the page w/ adjusted V_{REF}

A Flash Cell

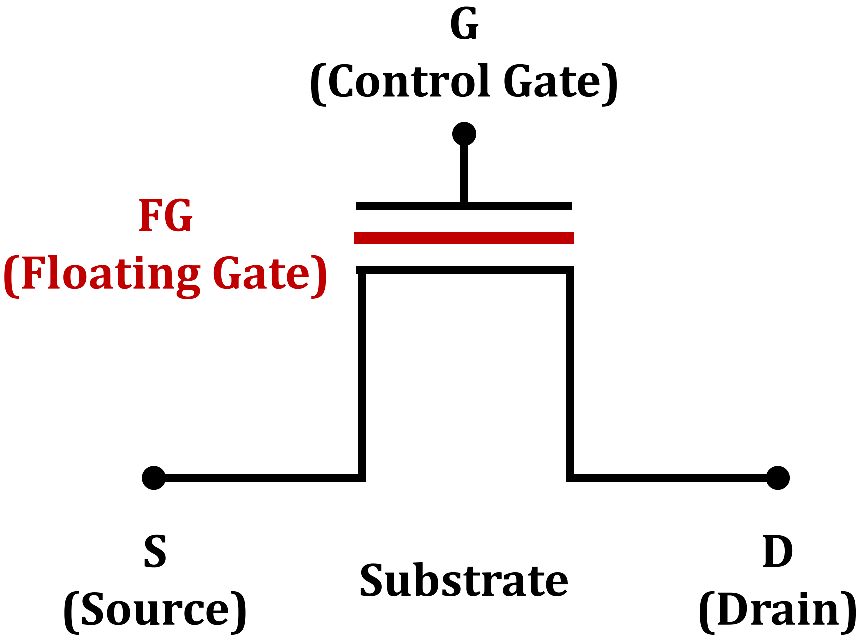
- Basically, it is a transistor

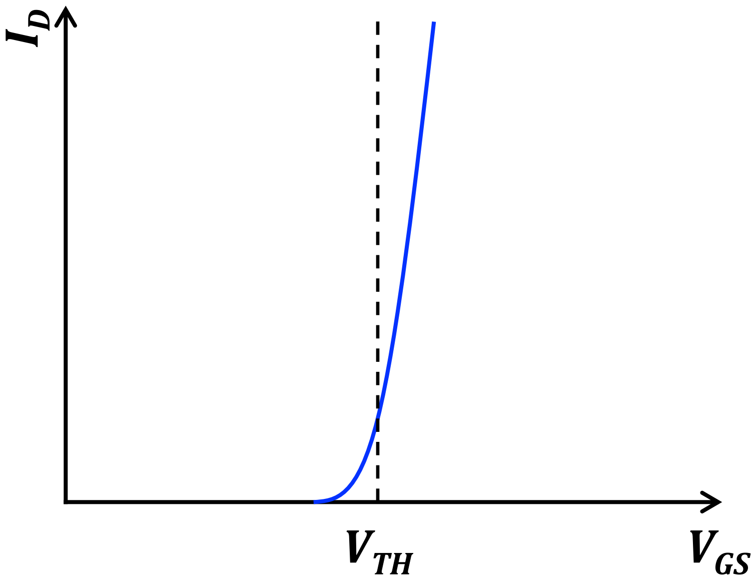




A Flash Cell

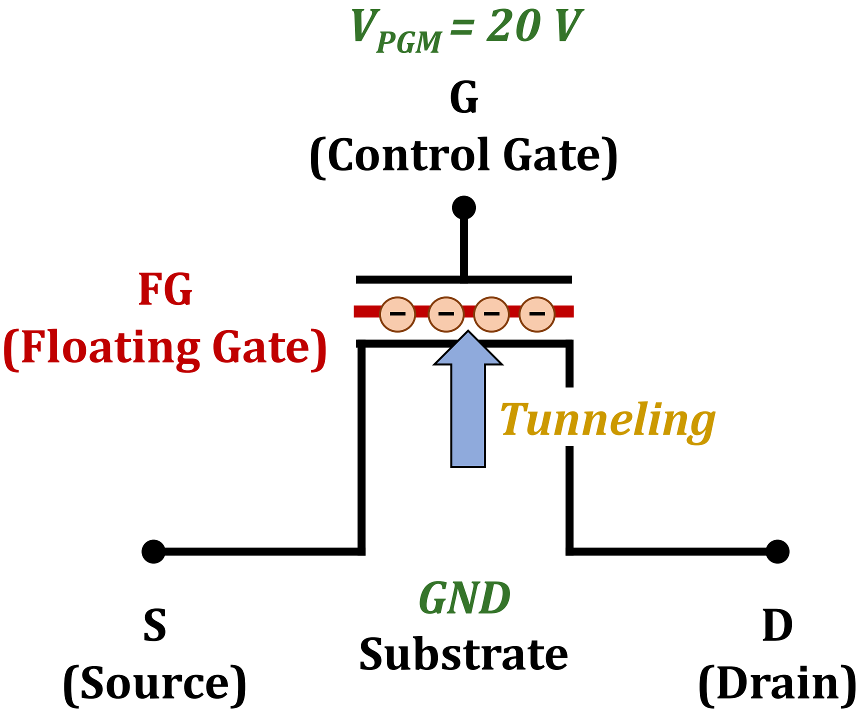
- Basically, it is a transistor
  - w/ a special material: Floating gate (2D) or Charge trap (3D)

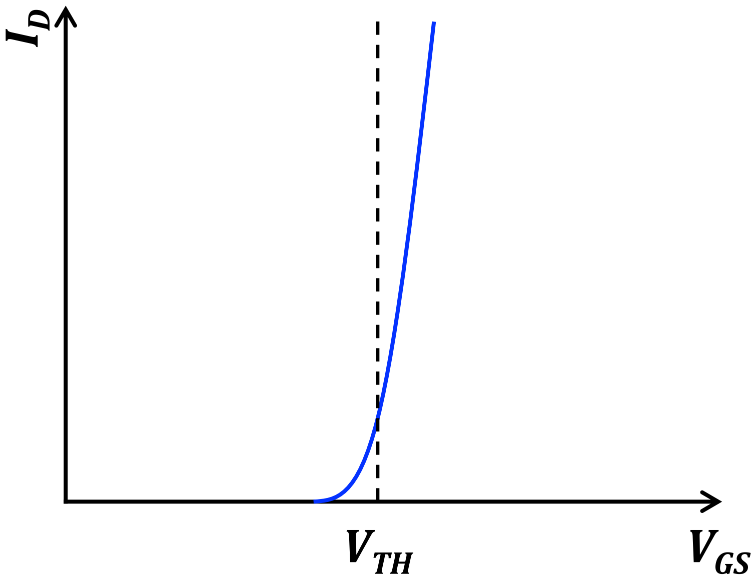




A Flash Cell

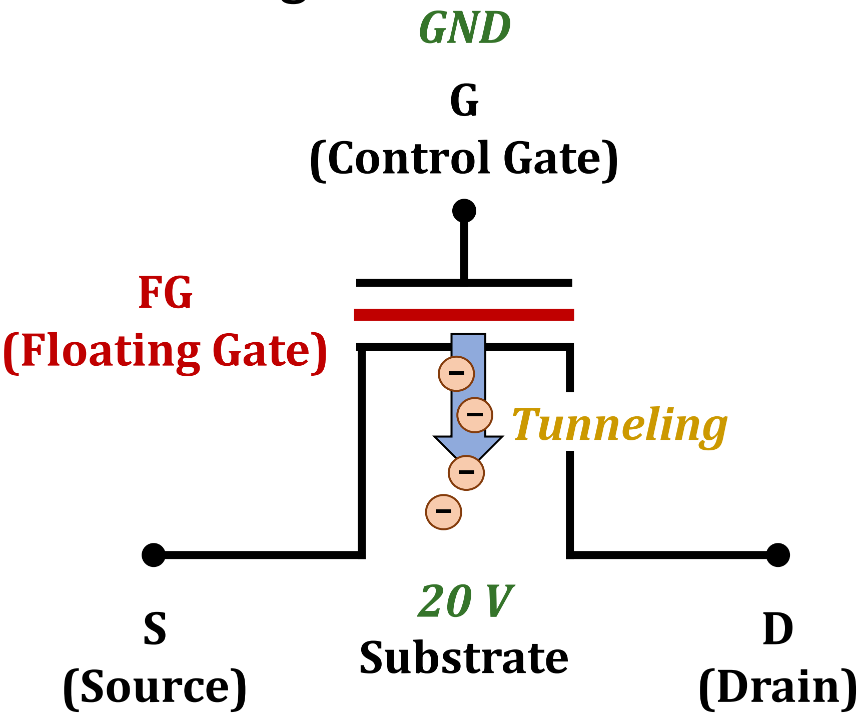
- Basically, it is a transistor
  - w/ a special material: Floating gate (2D) or Charge trap (3D)
  - Can hold electrons in a non-volatile manner

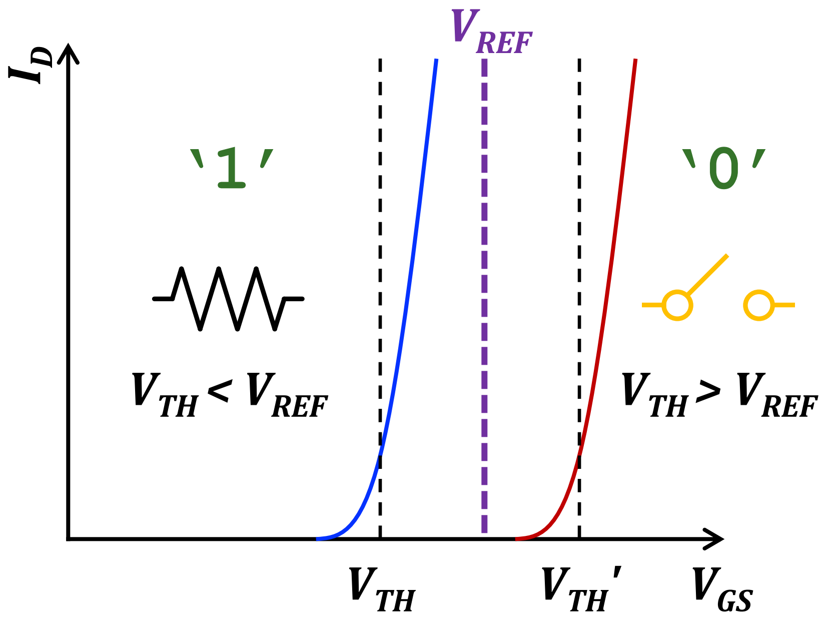




A Flash Cell

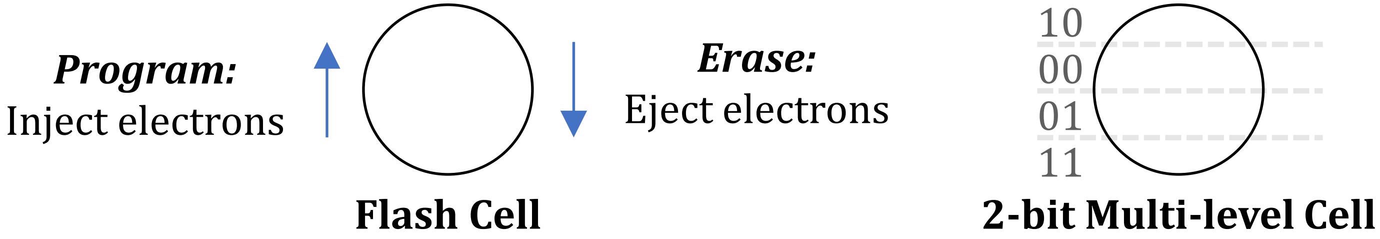
- Basically, it is a transistor
  - w/ a special material: Floating gate (2D) or Charge trap (3D)
  - Can hold electrons in a non-volatile manner
  - Changes the cell’s threshold voltage (V_{TH})





Flash Cell Characteristics

- Multi-leveling: A flash cell can store multiple bits



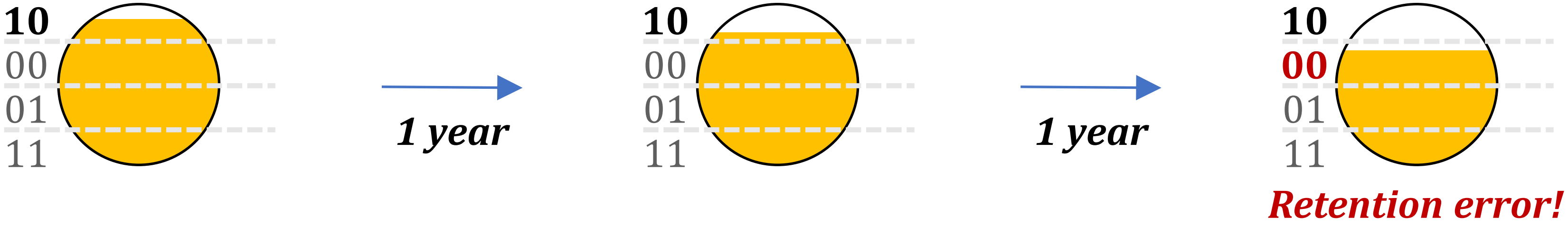
Program:
Inject electrons

Erase:
Eject electrons

Flash Cell

2-bit Multi-level Cell

- Retention loss: A cell leaks electrons over time

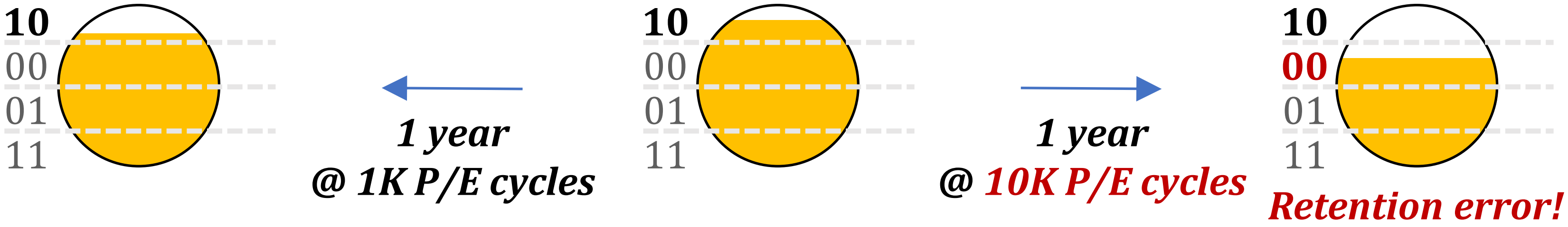


1 year

1 year

Retention error!

- Limited lifetime: A cell wears out after P/E cycling



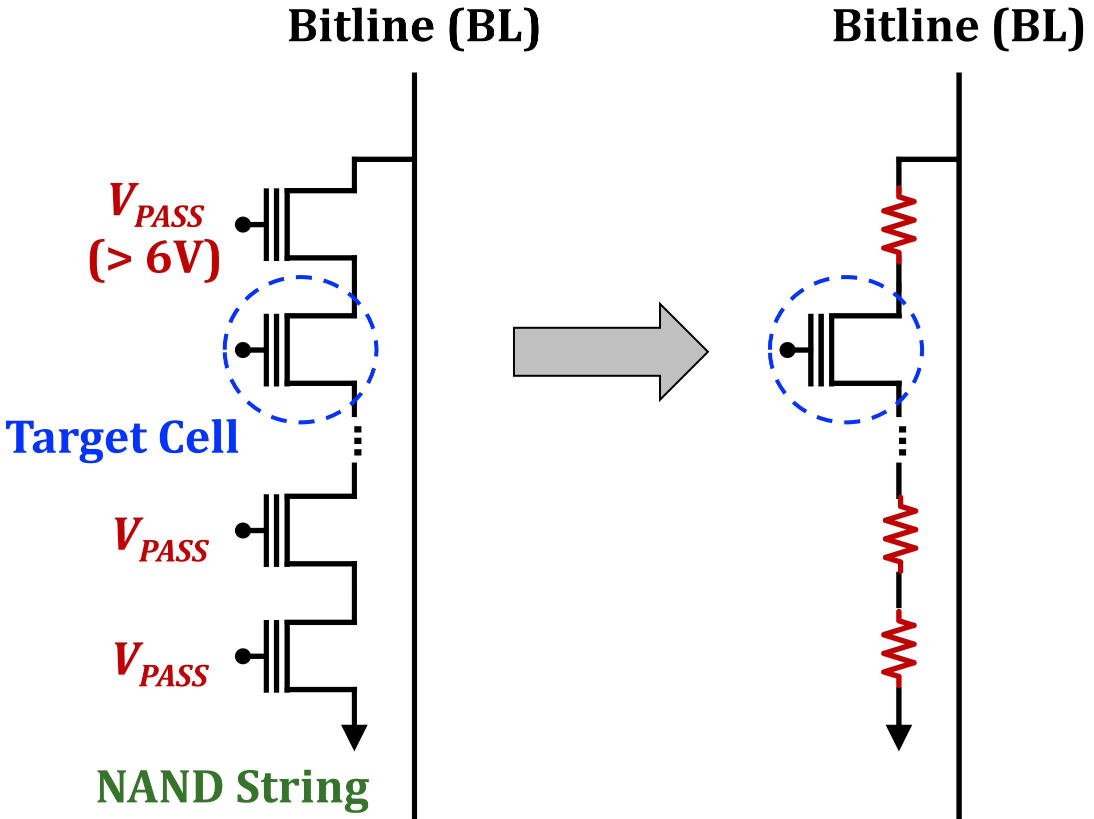
1 year
@ 1K P/E cycles

1 year
@ 10K P/E cycles

Retention error!

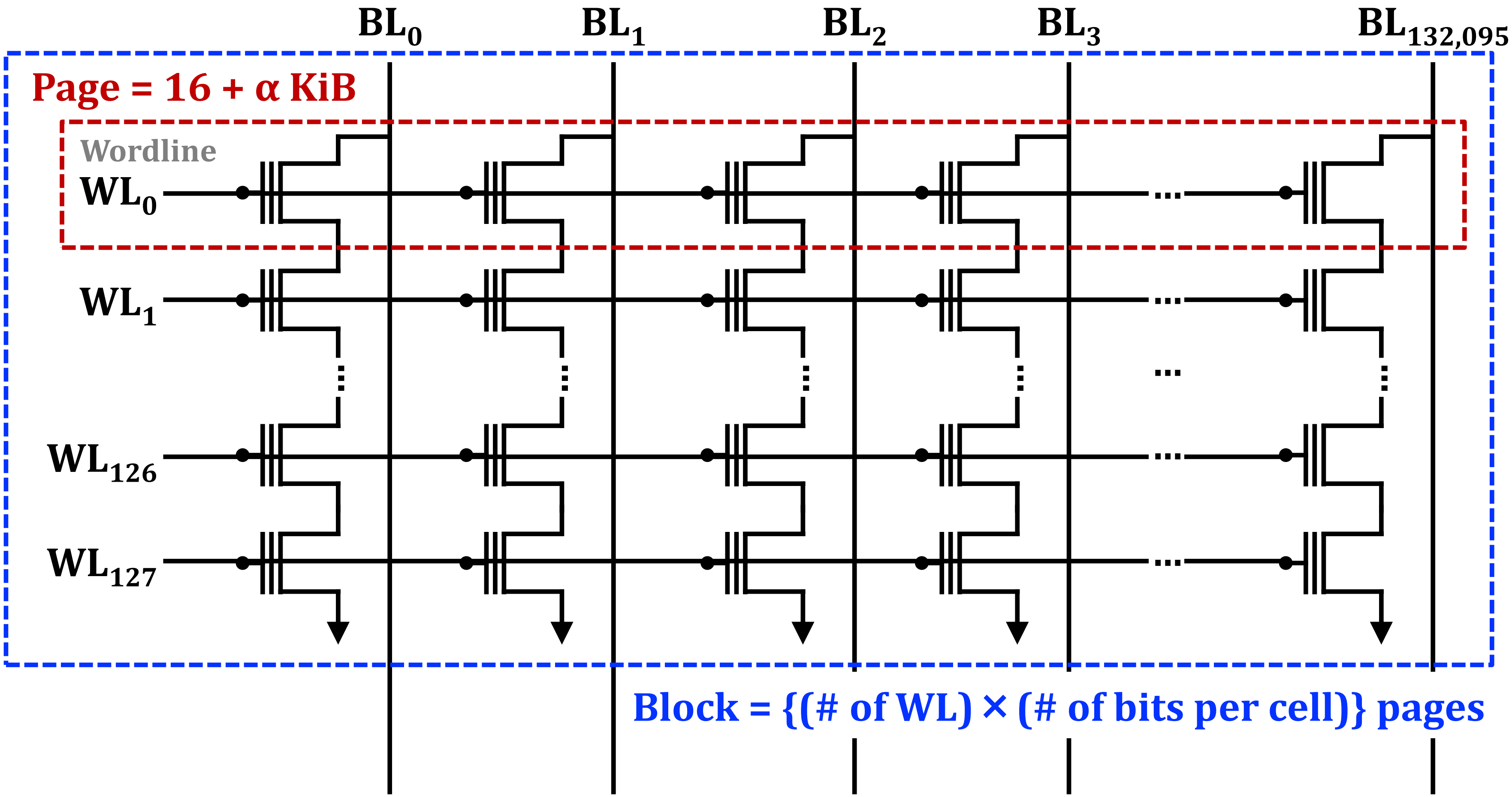
A NAND String

- Multiple (e.g., 128) flash cells are serially connected



Pages and Blocks

- A large number (> 100,000) of cells operate concurrently

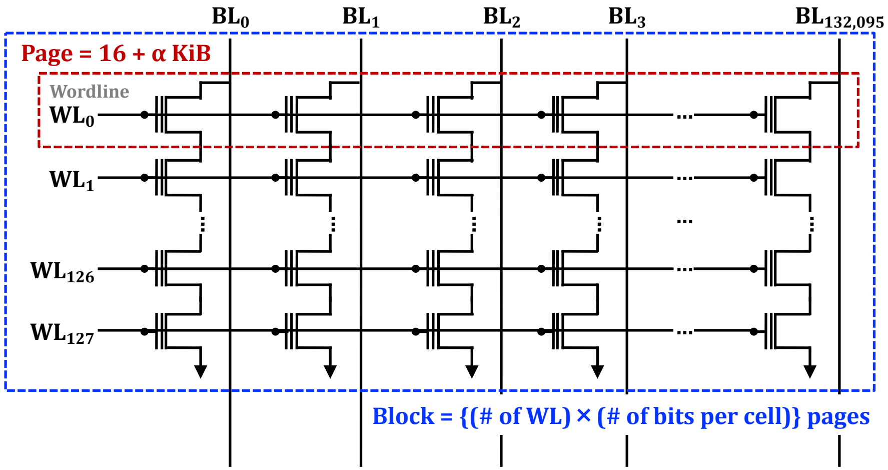


Pages and Blocks (Continued)

- Program and erase: Unidirectional
  - Programming a cell → Increasing the cell's V_{TH}
  - Eraseing a cell → Decreasing the cell's V_{TH}

- Programming a page cannot change '0' cells to '1' cells
  → Erase-before-write property

- Erase unit: Block
  - Increase erase bandwidth
  - Makes in-place write on a page very inefficient
    → Out-of-place write

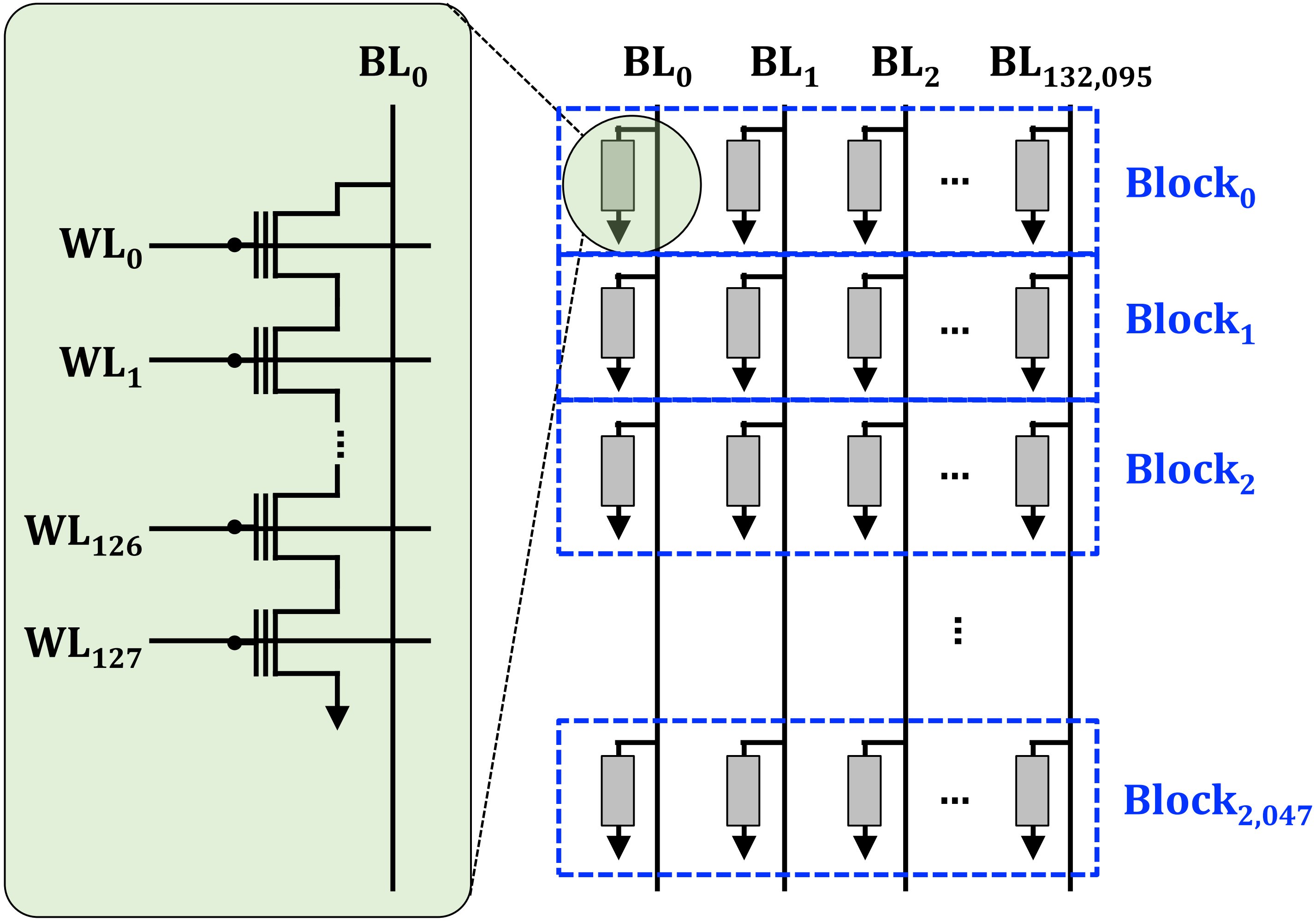


Page = 16 + α KiB

Block = {(# of WL) × (# of bits per cell)} pages

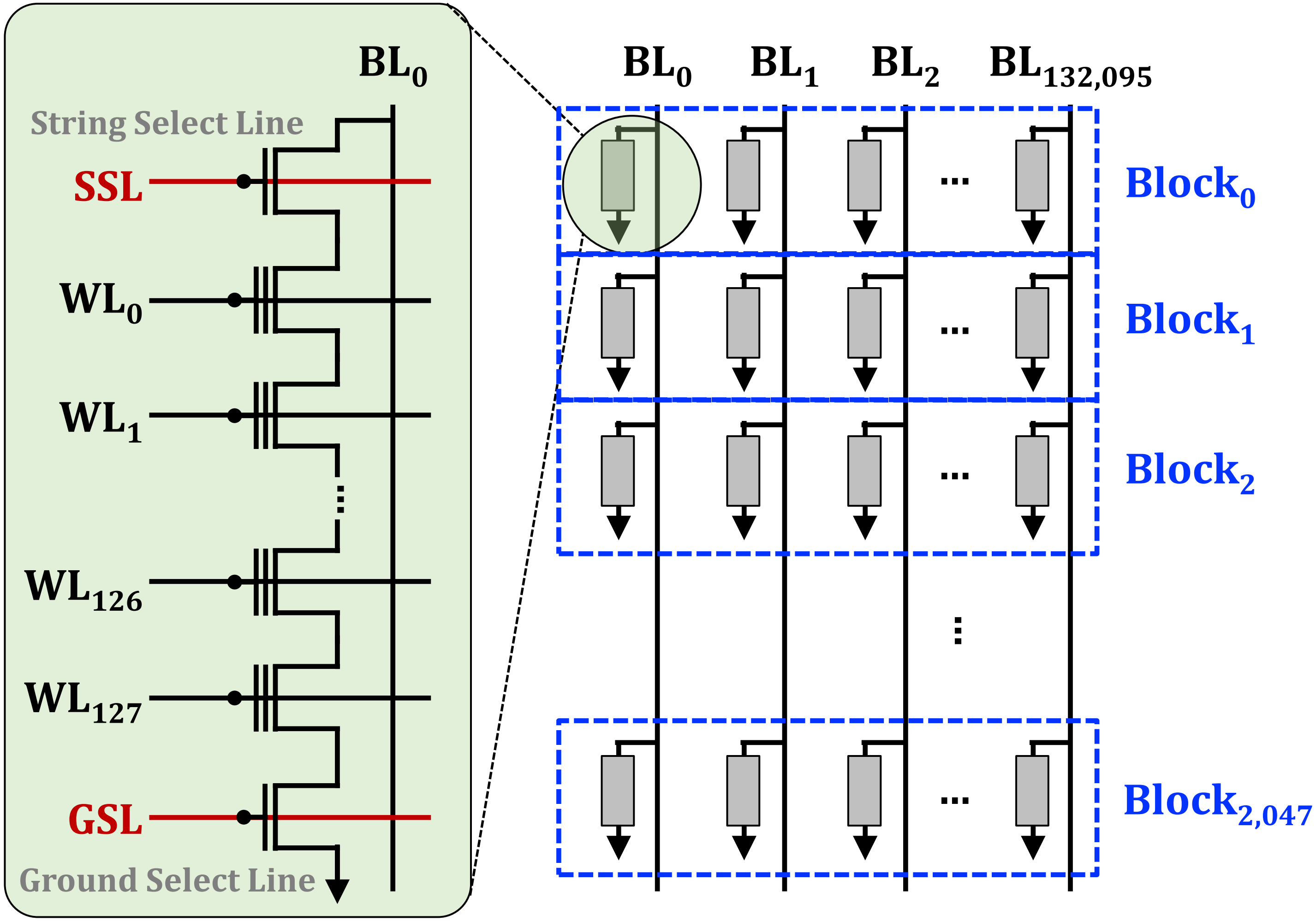
Planes

- A large number (> 1,000) of blocks share bitlines in a plane



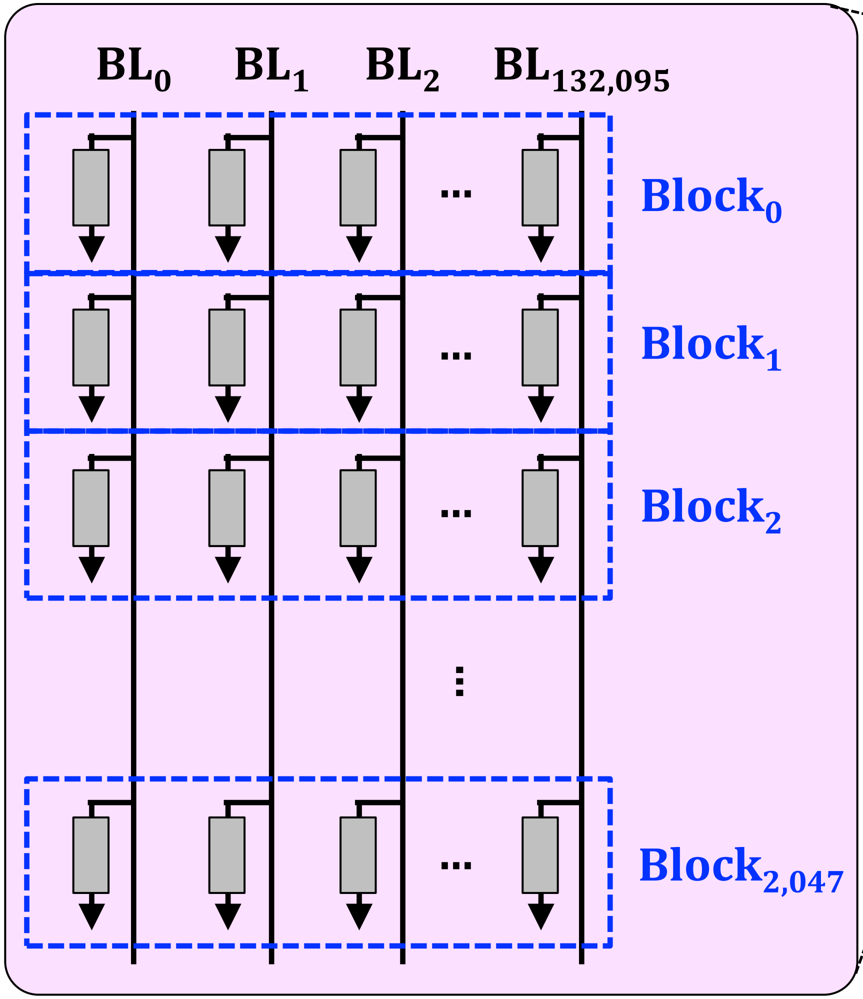
Planes

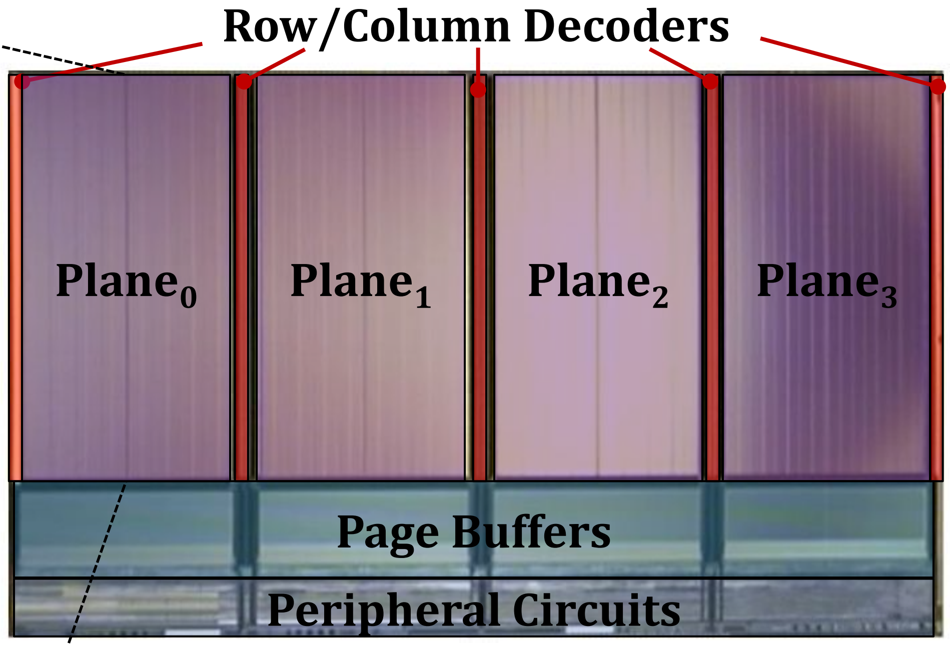
- A large number (> 1,000) of blocks share bitlines in a plane



Planes and Dies

- A die contains multiple (e.g., 2 – 4) planes



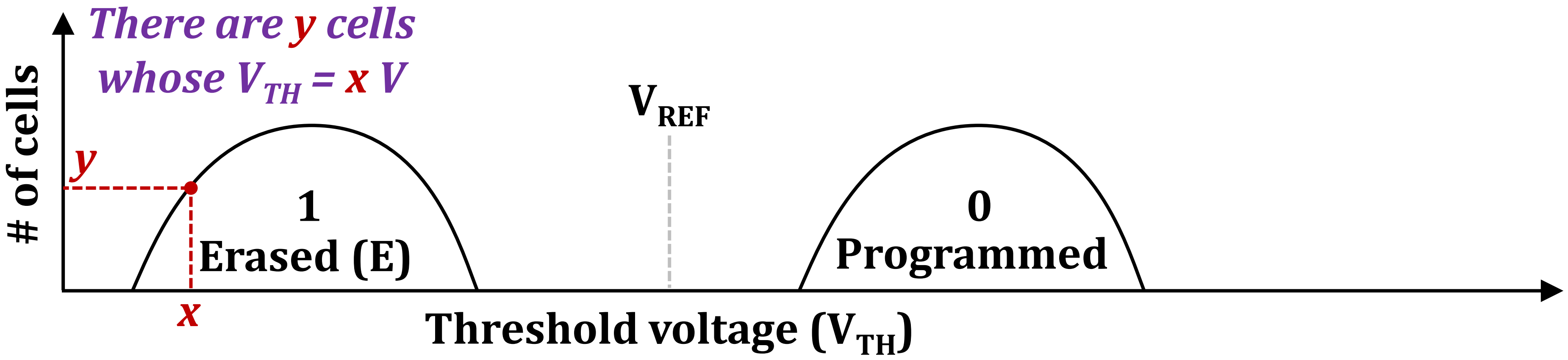


A 21-nm 2D NAND Flash Die

- Planes share decoders: limits internal parallelism (only operations @ the same WL offset)

Threshold Voltage Distribution

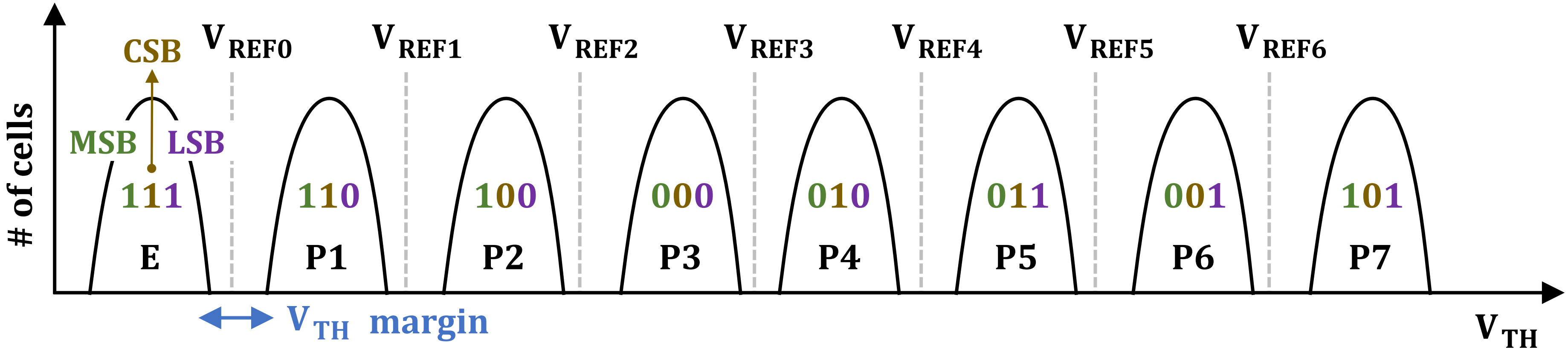
- V_{TH} distribution of cells in a programmed page/block/chip



- Why distribution? Variations across the cells
  - Some cells are more easily programmed or erased

V_{TH} Distribution of MLC NAND Flash

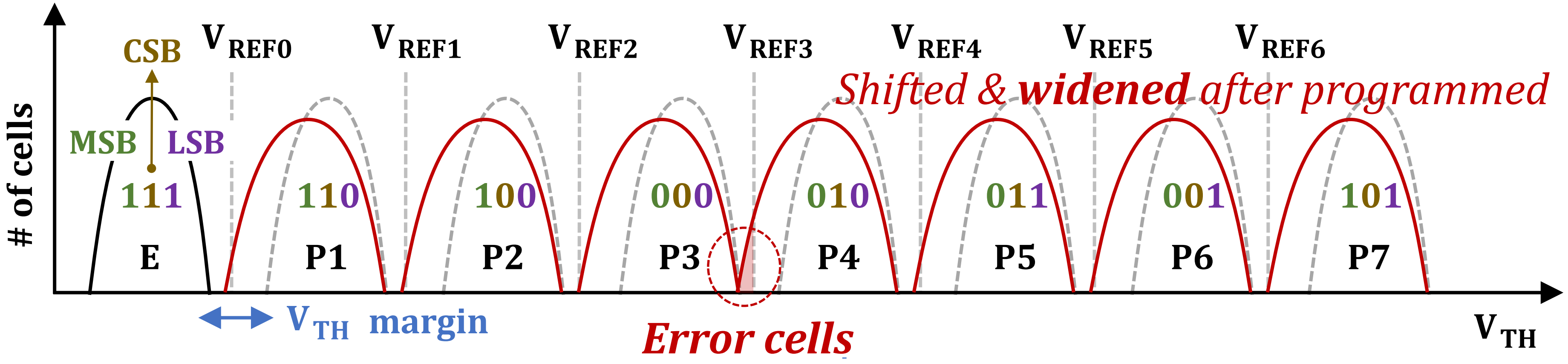
- Multi-level cell (MLC) technique
  - 2^m V_{TH} states required to store *m* bits in a single flash cell



- Limited width of the V_{TH} window: Need to
  - Make each V_{TH} state narrow
  - Guarantee sufficient margins b/w adjacent V_{TH} states

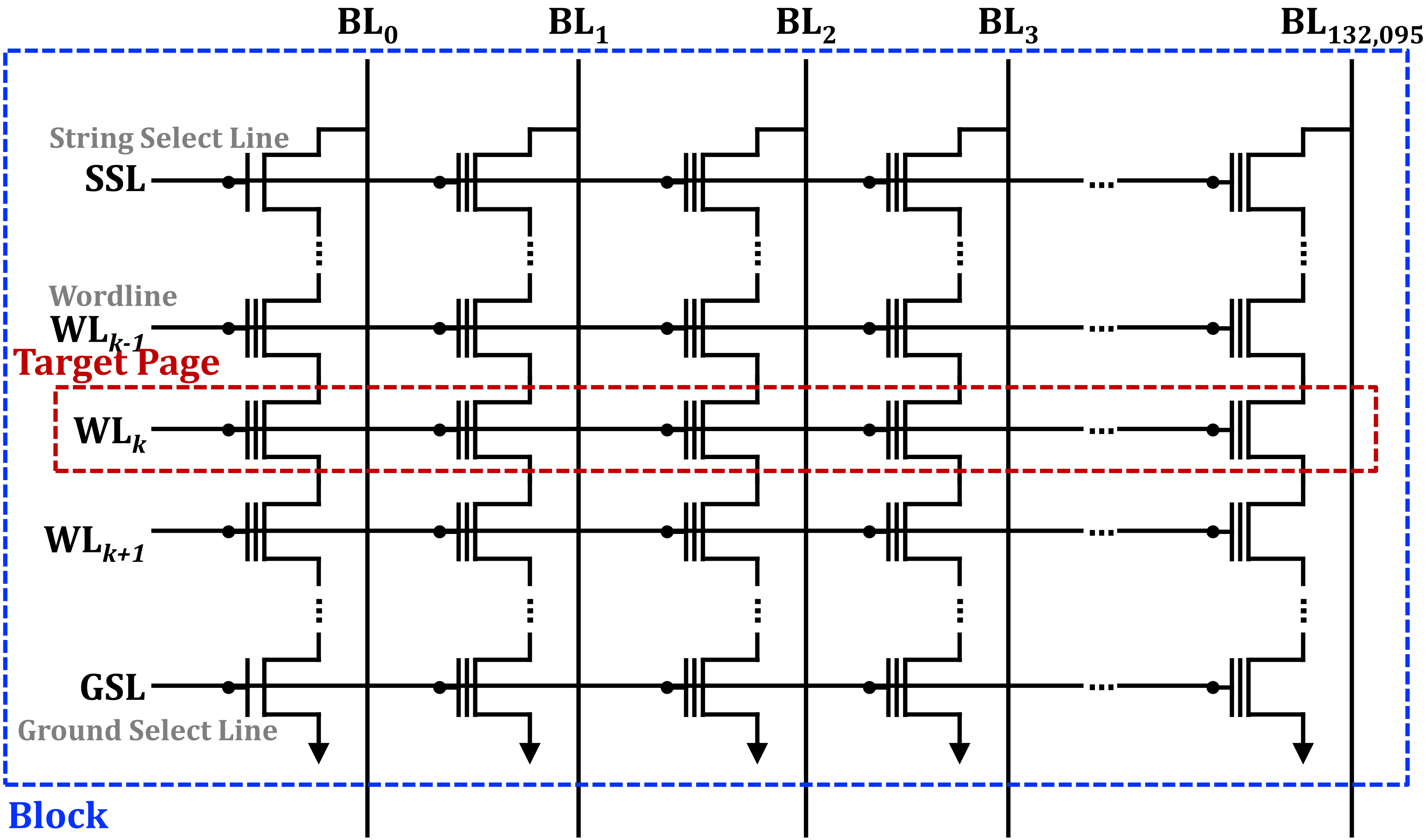
V_{TH} Distribution of MLC NAND Flash

- Multi-level cell (MLC) technique
  - 2^m V_{TH} states required to store *m* bits in a single flash cell



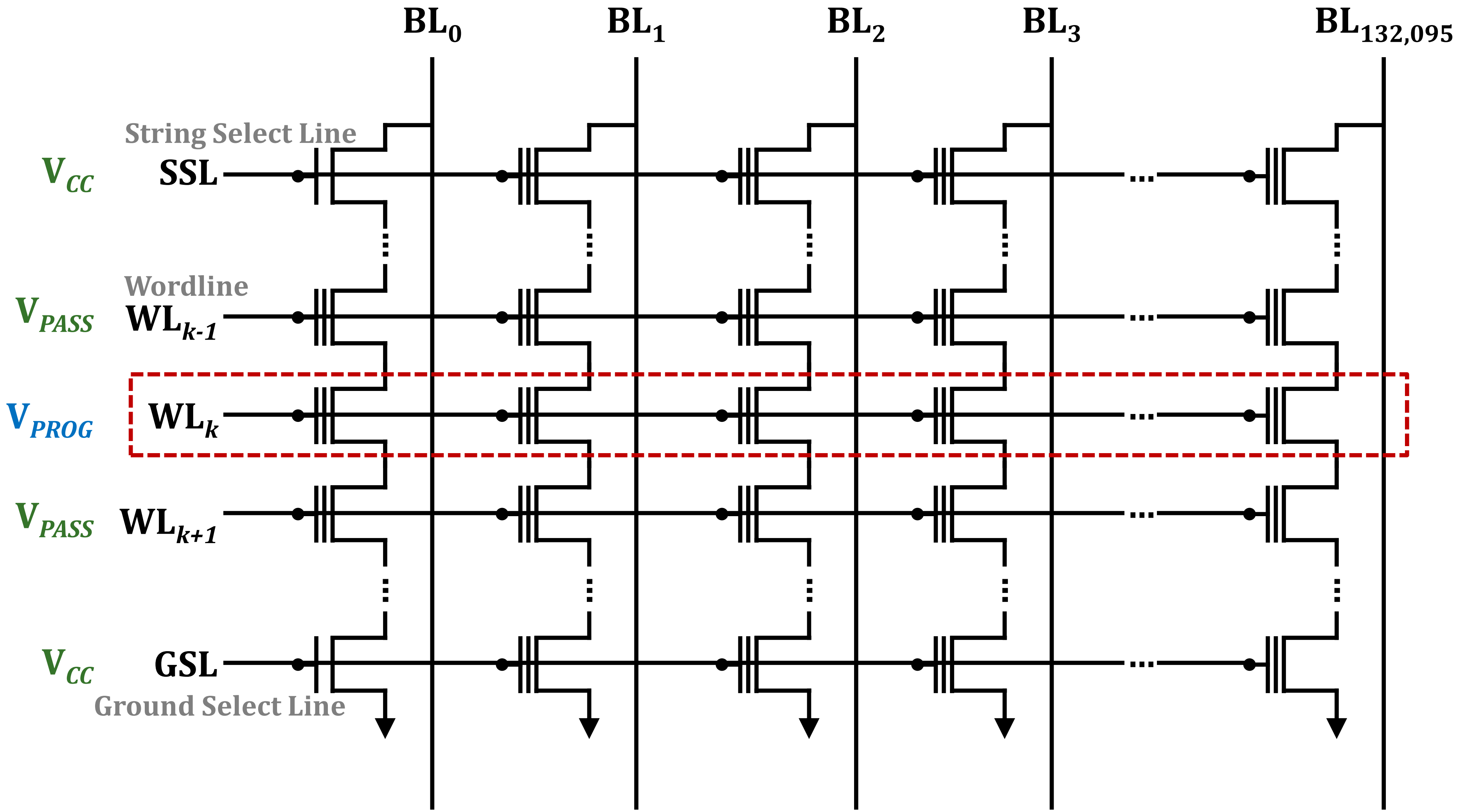
- Limited width of the V_{TH} window: Need to
  - Make each V_{TH} state narrow
  - Guarantee sufficient margins b/w adjacent V_{TH} states
    - V_{TH} changes over time after programmed
    - Narrower margins → Lower reliability
    - More bits per cell → higher density but lower reliability

Basic Operation: Page Program



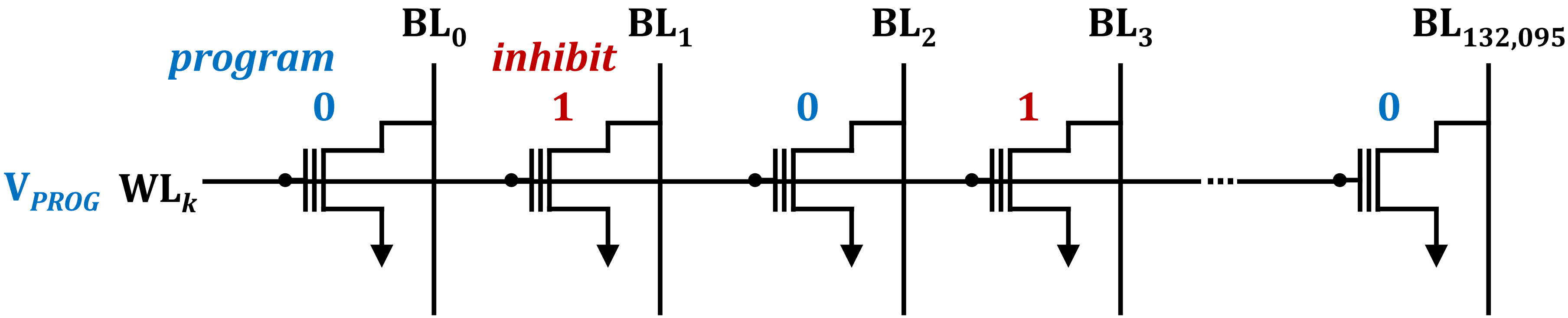
Basic Operation: Page Program

- WL control – All other cells operate as a resistance



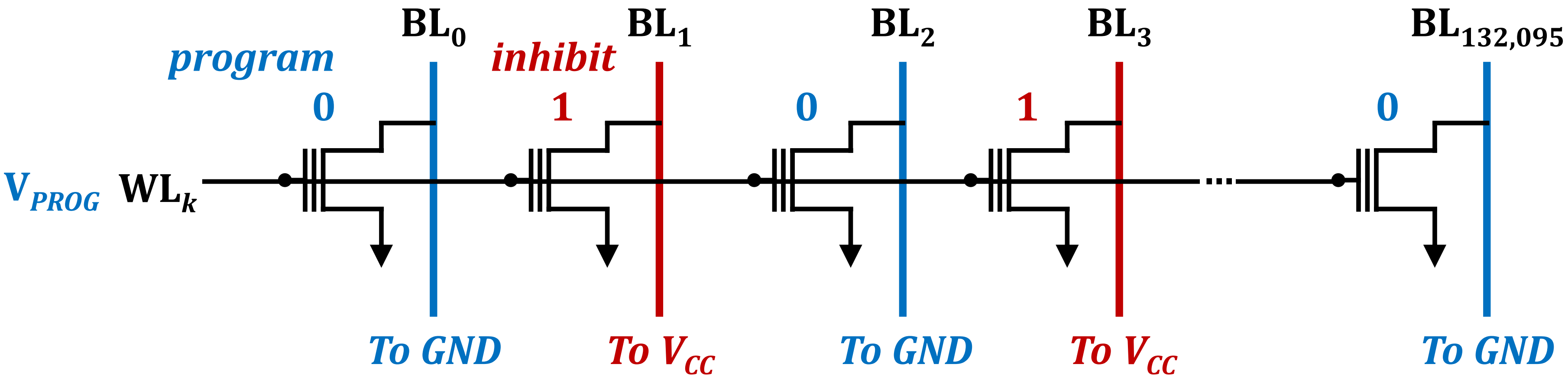
Basic Operation: Page Program

- BL control – Inhibits cells to not be programmed

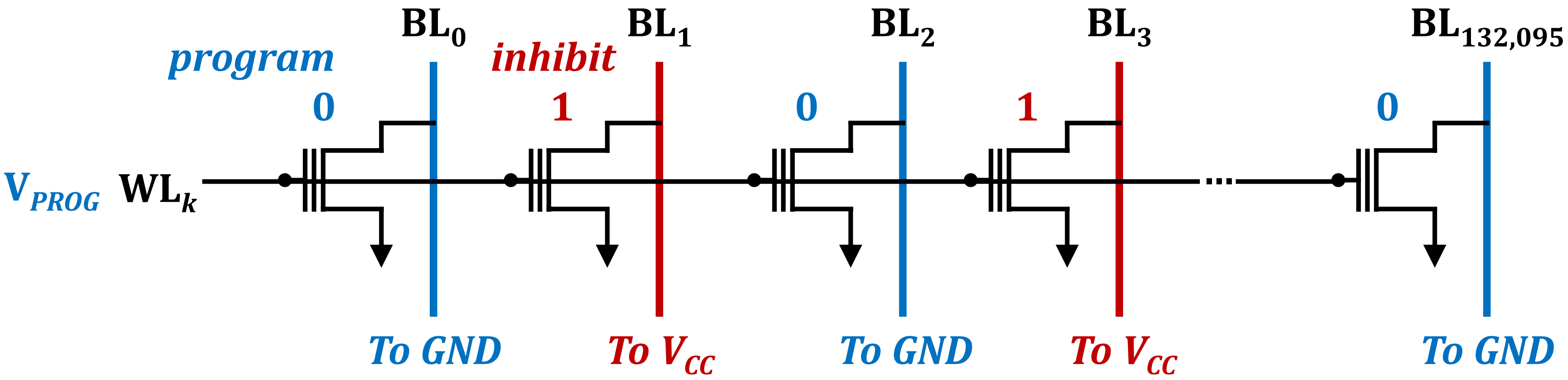


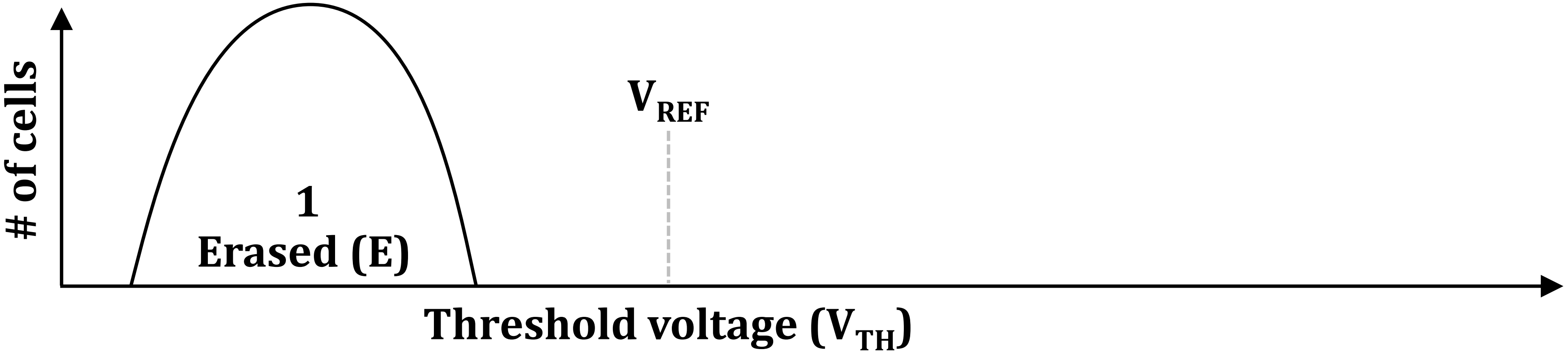
Basic Operation: Page Program

- BL control – Inhibits cells to not be programmed

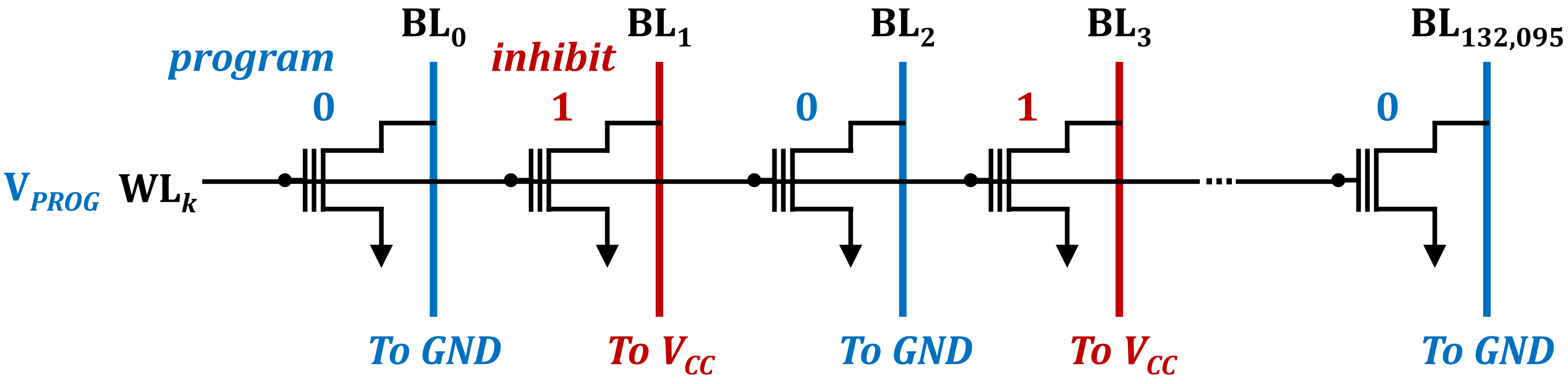


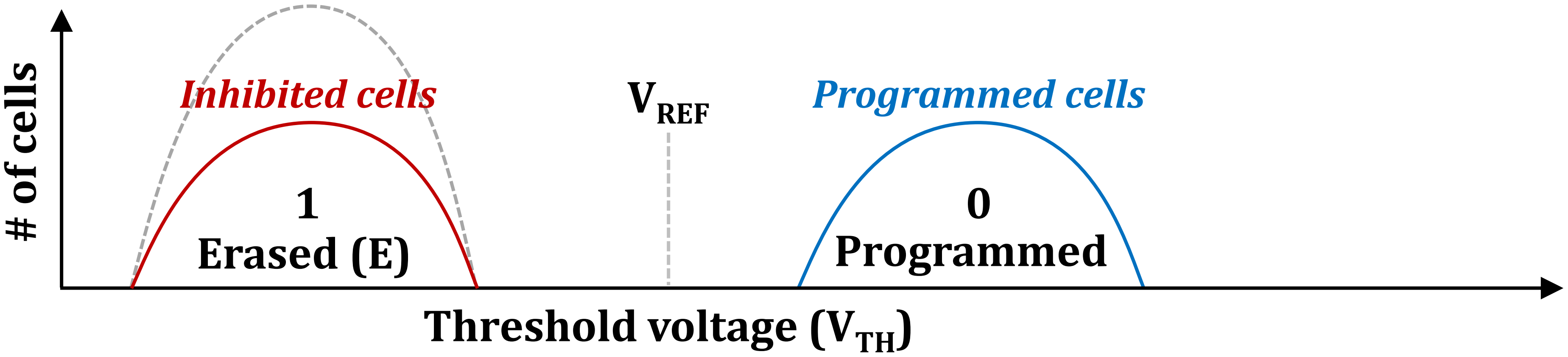
Basic Operation: Page Program



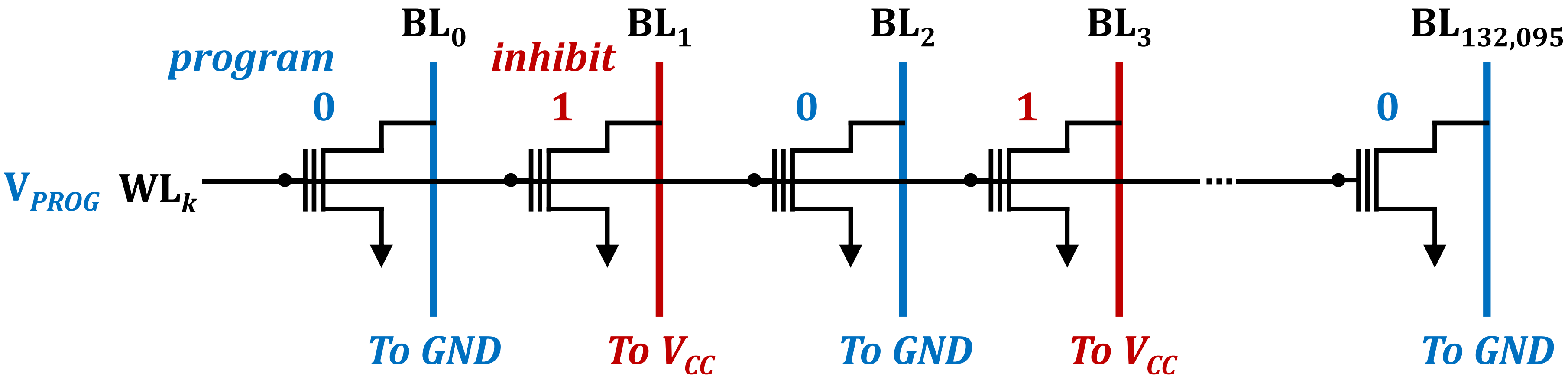


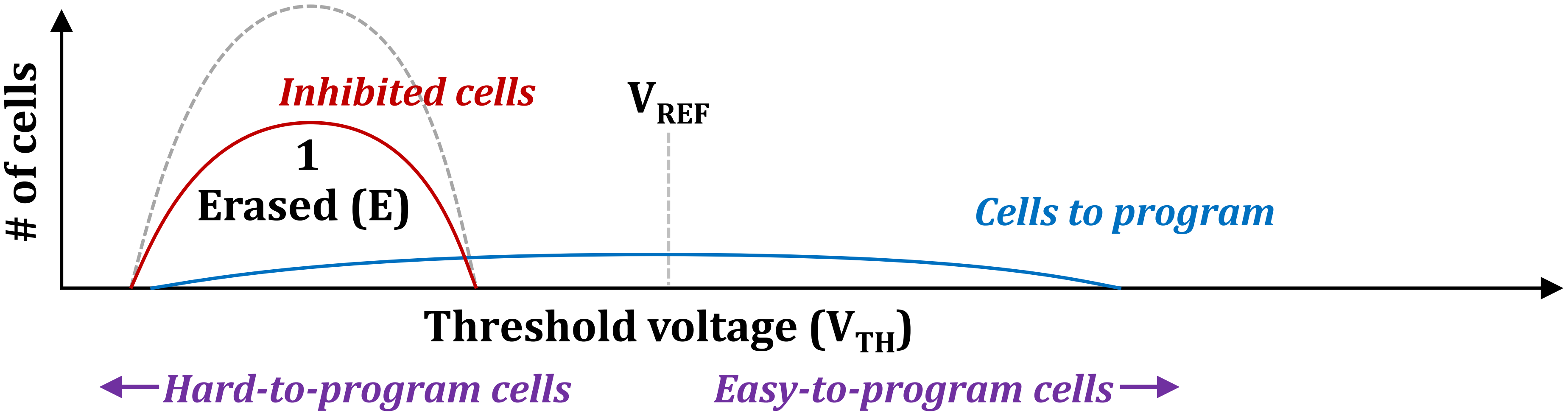
Basic Operation: Page Program





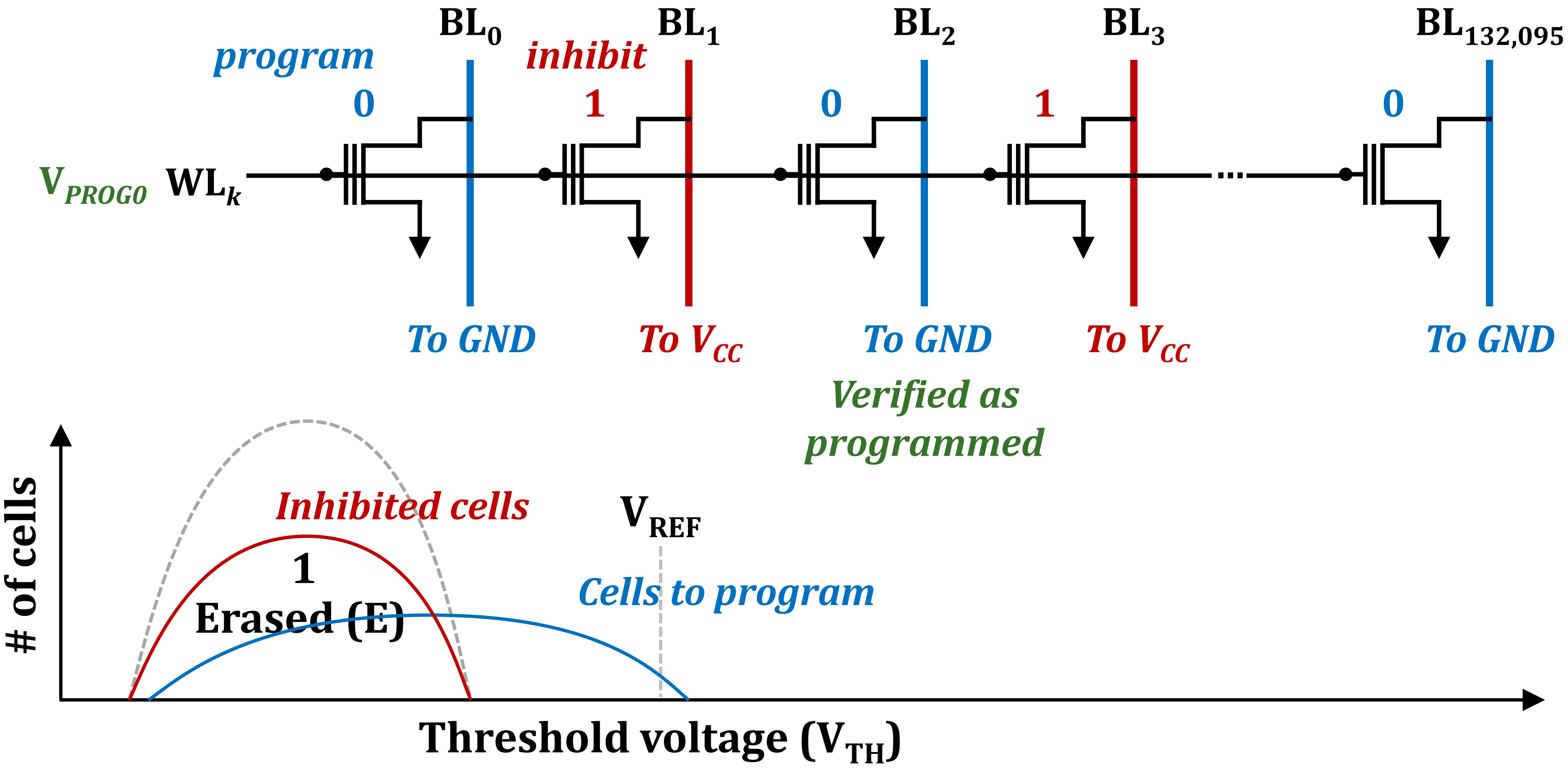
Basic Operation: Page Program





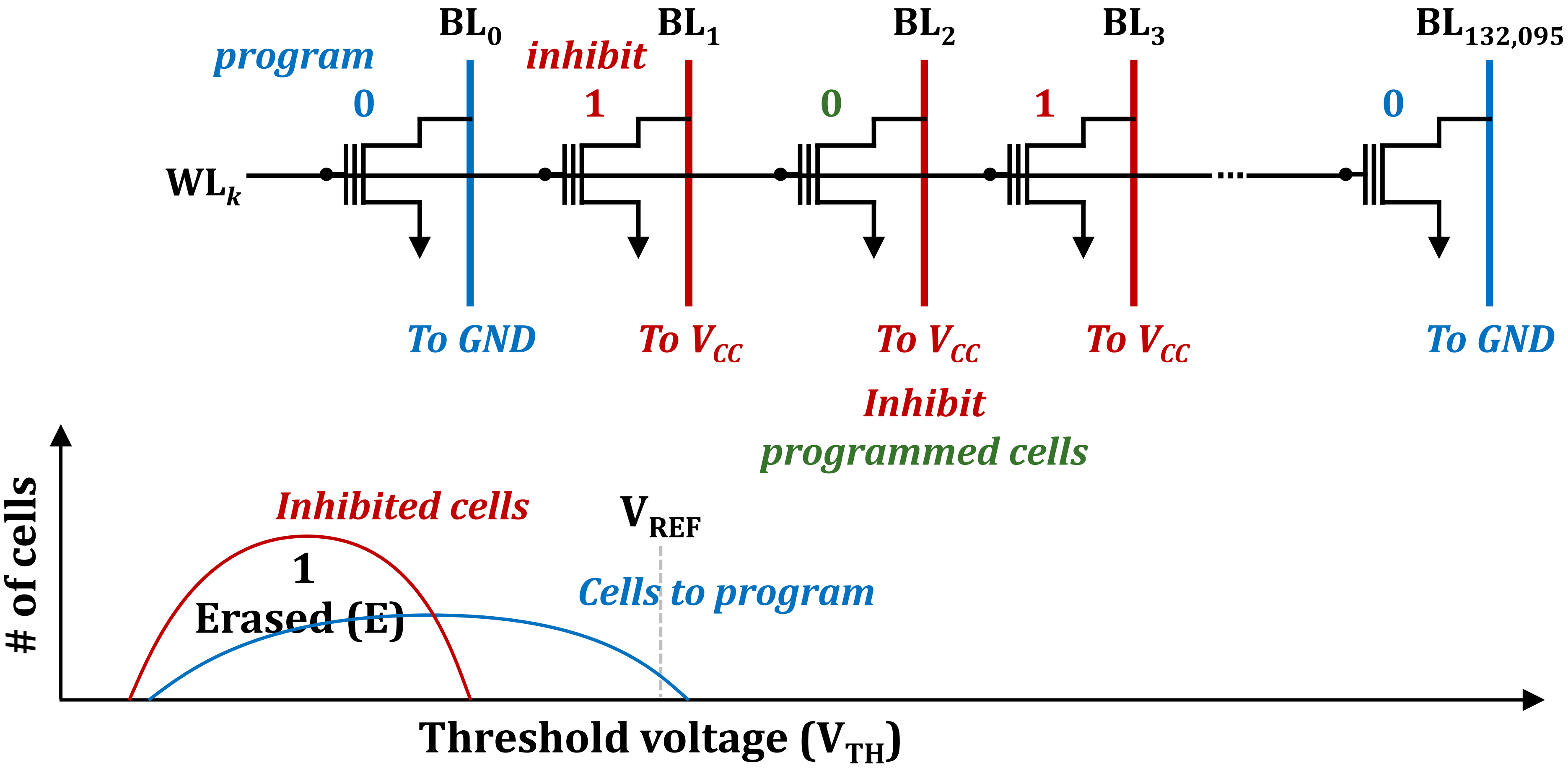
Basic Operation: Page Program

- Incremental Step-Pulse Programming (ISPP)



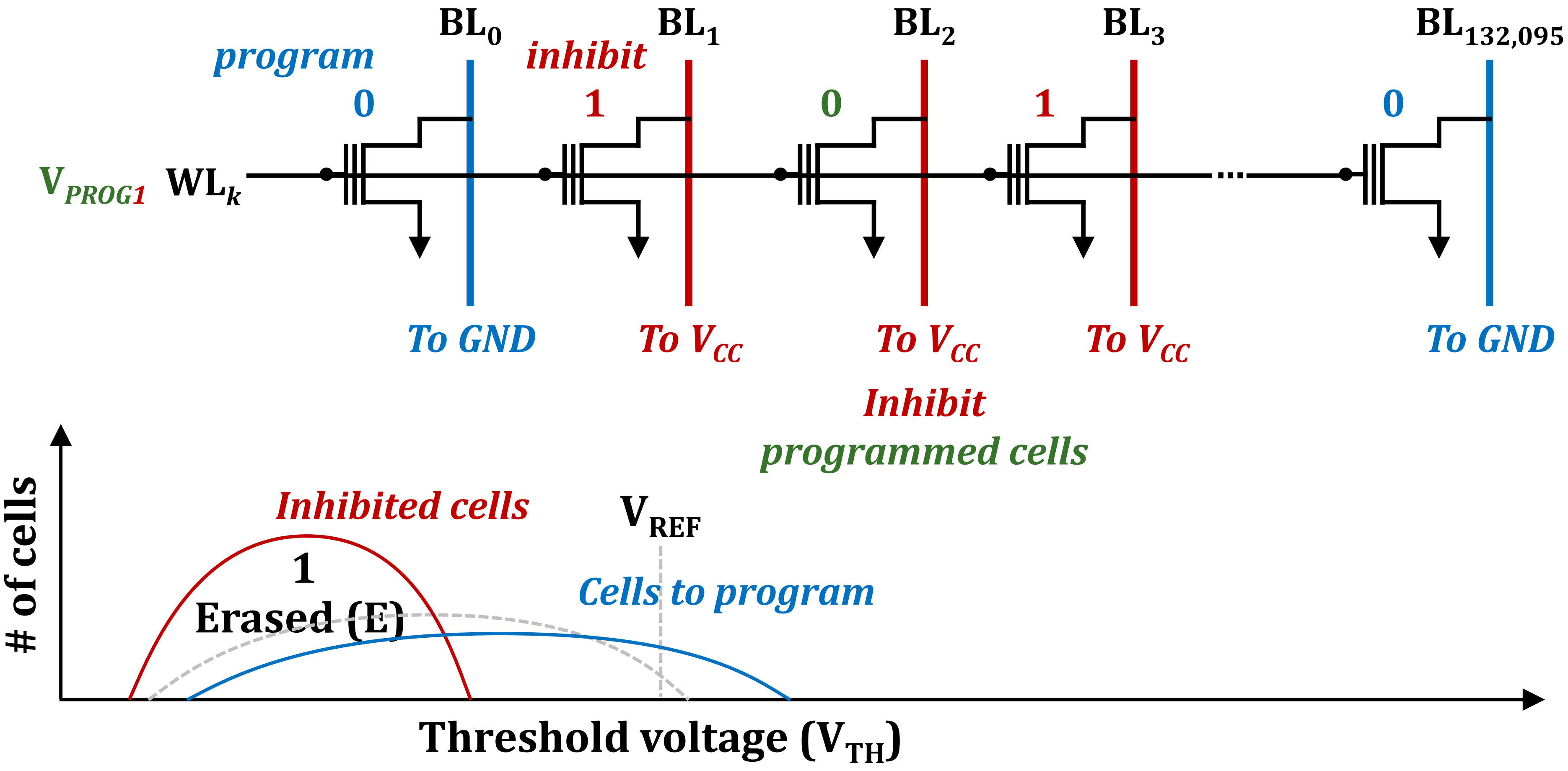
Basic Operation: Page Program

- Incremental Step-Pulse Programming (ISPP)



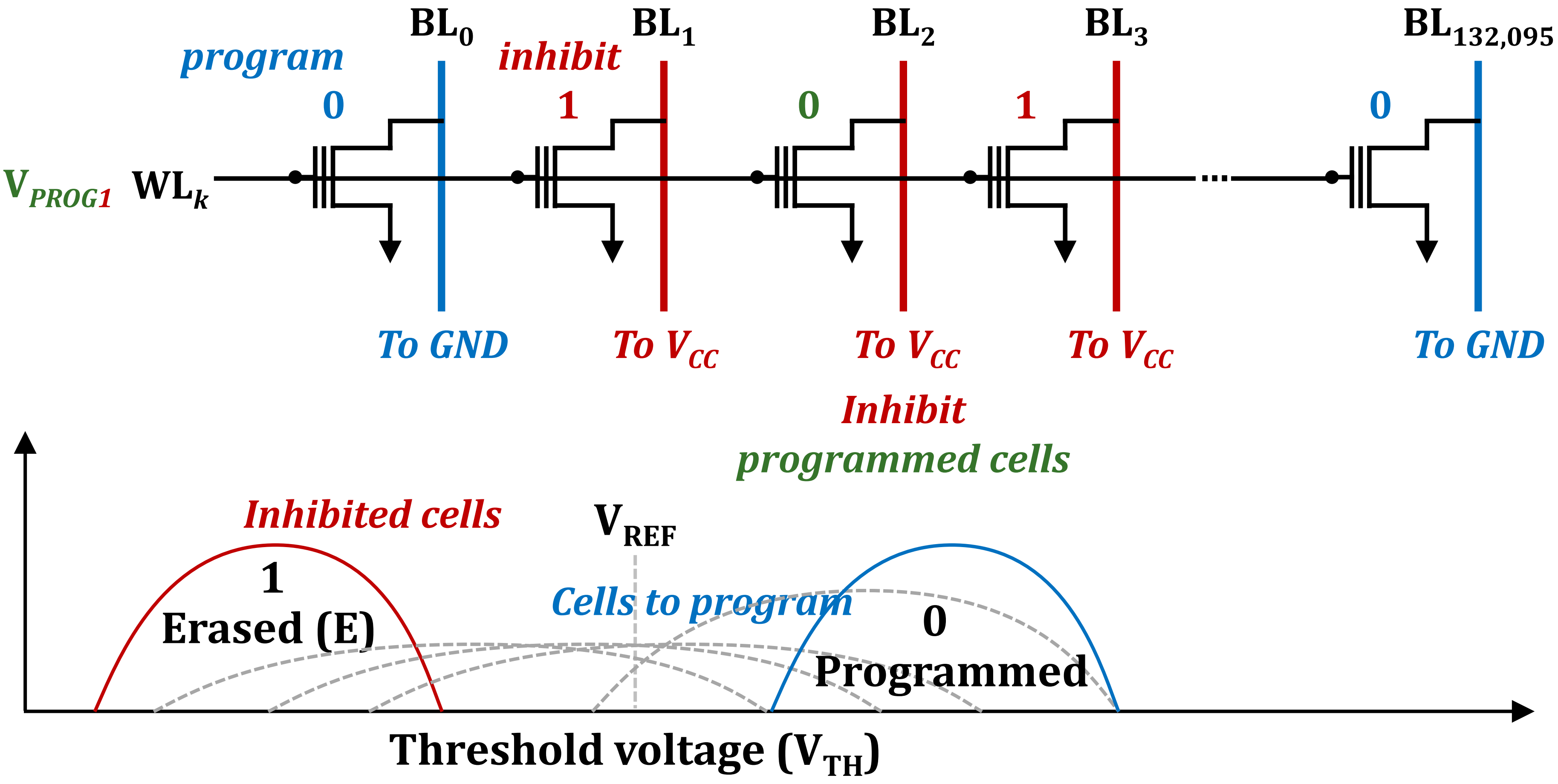
Basic Operation: Page Program

- Incremental Step-Pulse Programming (ISPP)



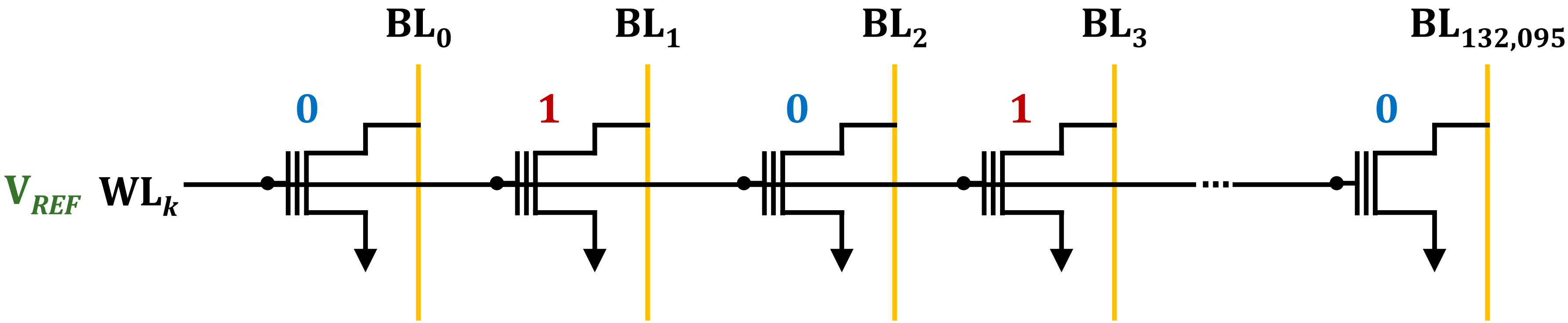
Basic Operation: Page Program

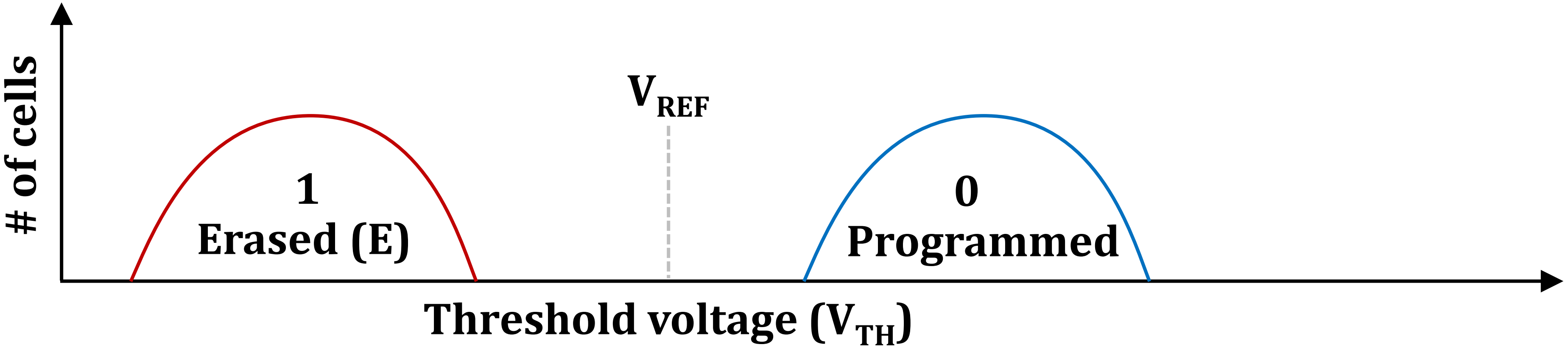
- Incremental Step-Pulse Programming (ISPP)



Basic Operation: Page Read

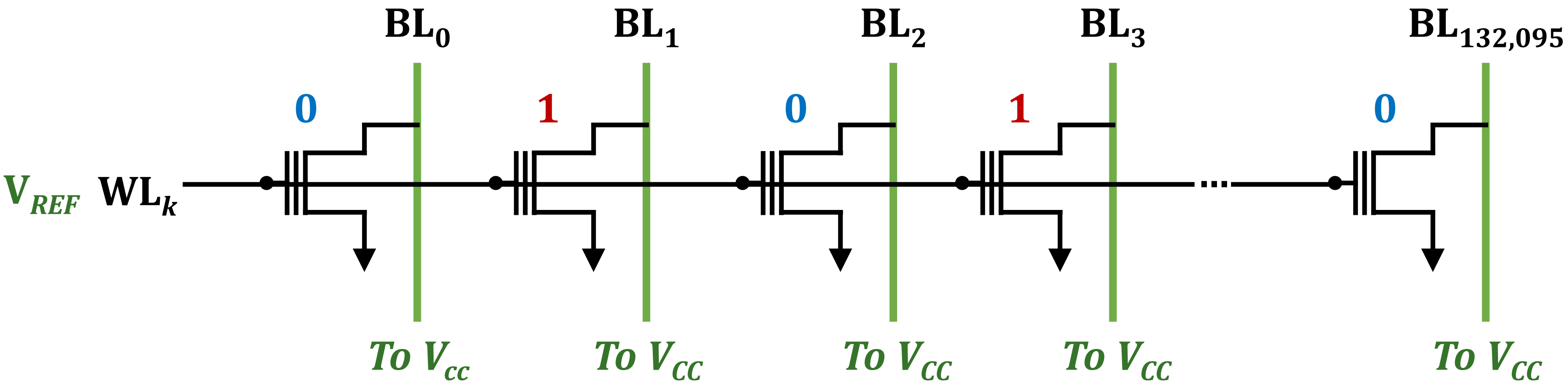
- WL control – All other cells operate as a resistance

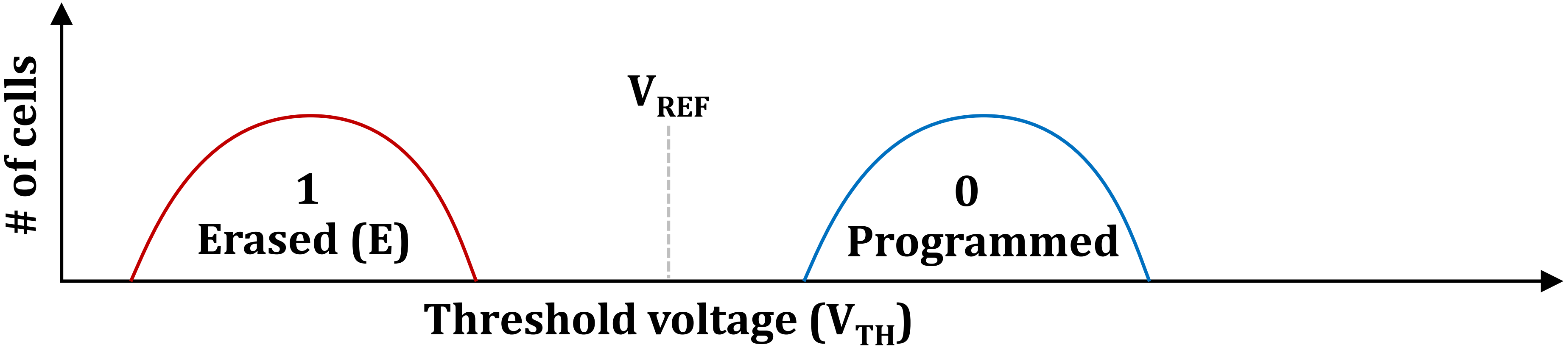




Basic Operation: Page Read

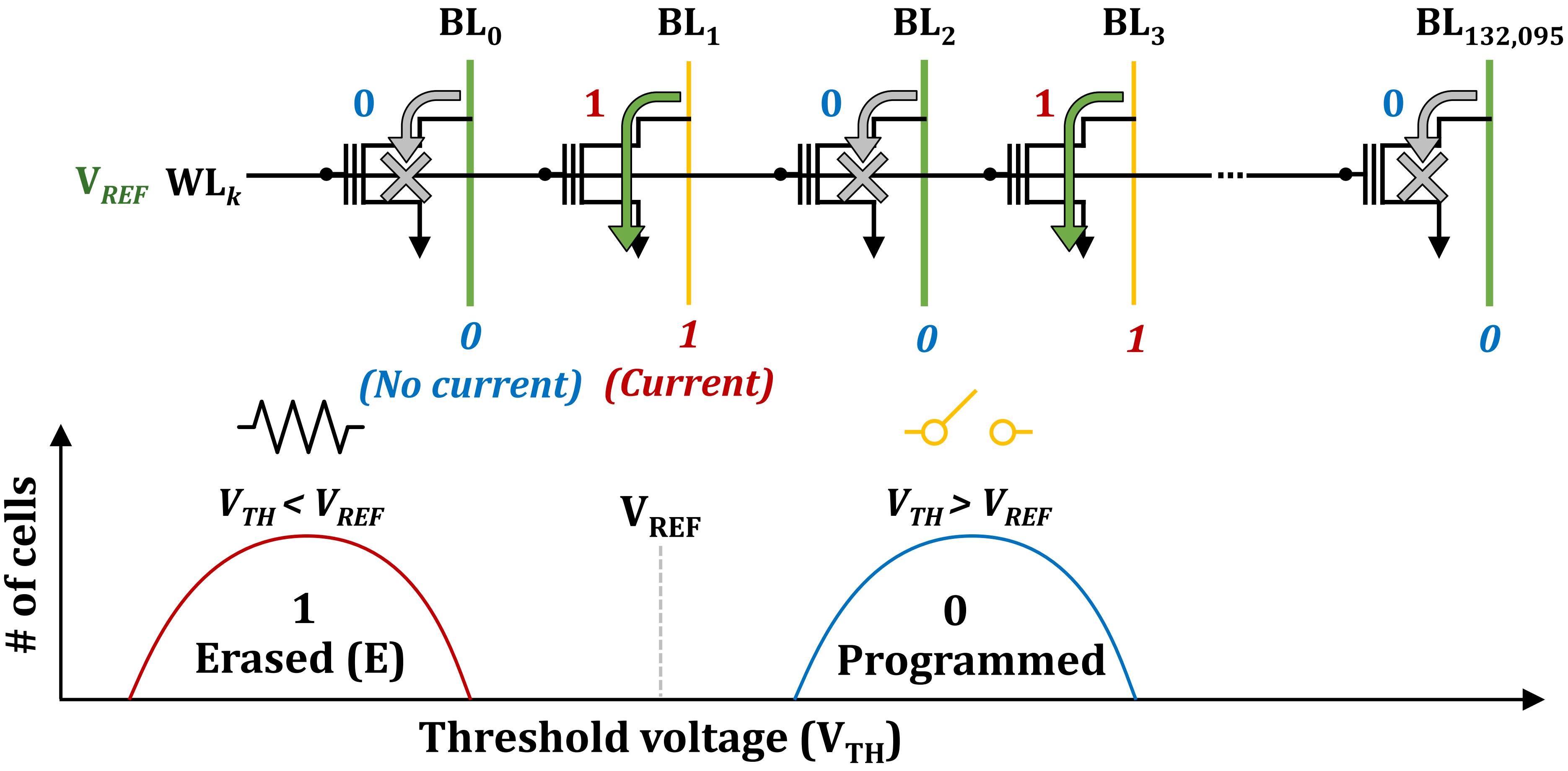
- BL control – Charge all BLs





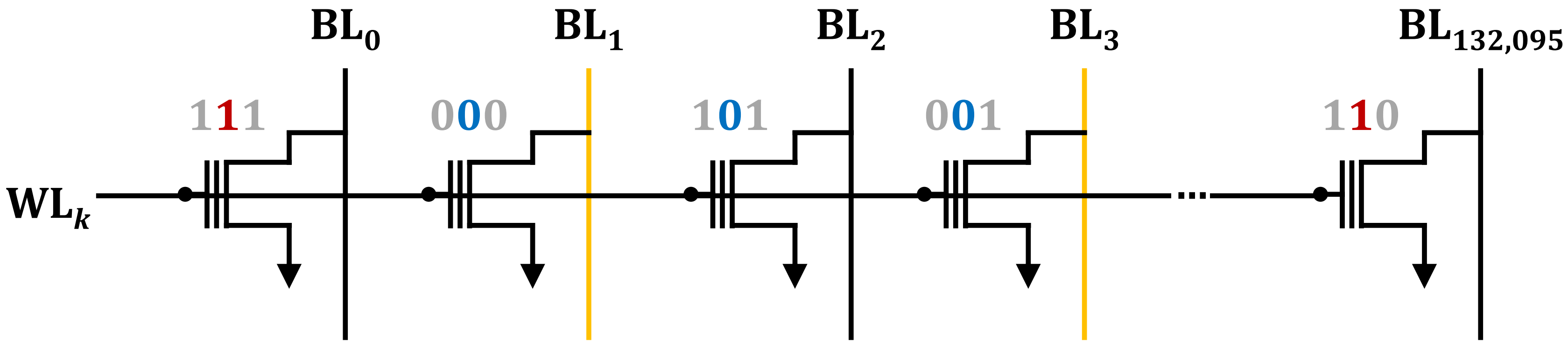
Basic Operation: Page Read

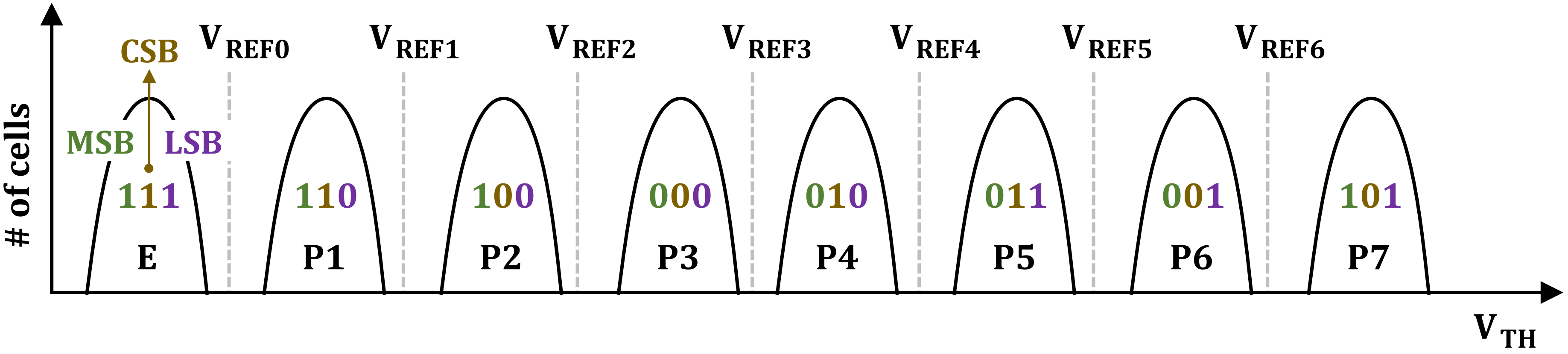
- Sensing the current through BLs



Basic Operation: Page Read - MLC

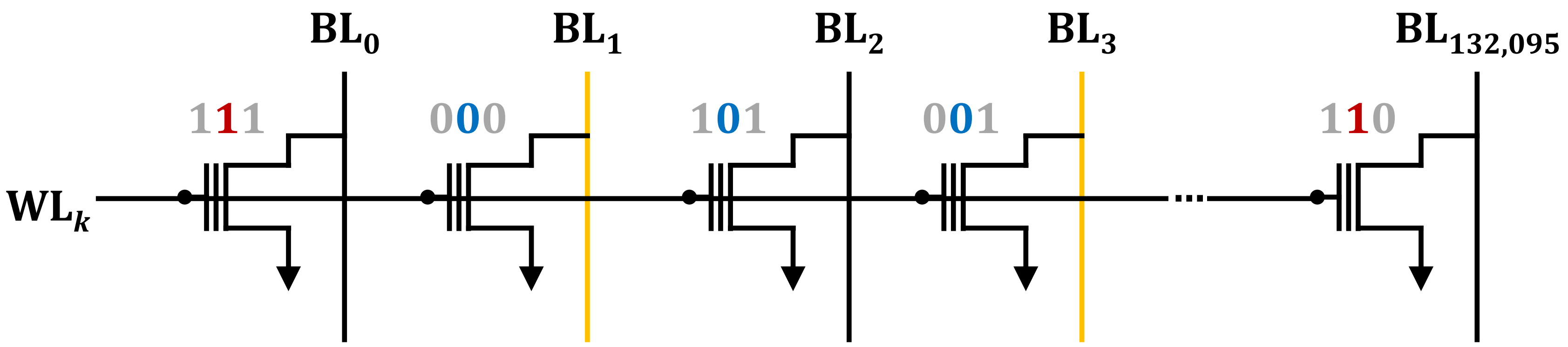
- Sensing the current through BLs

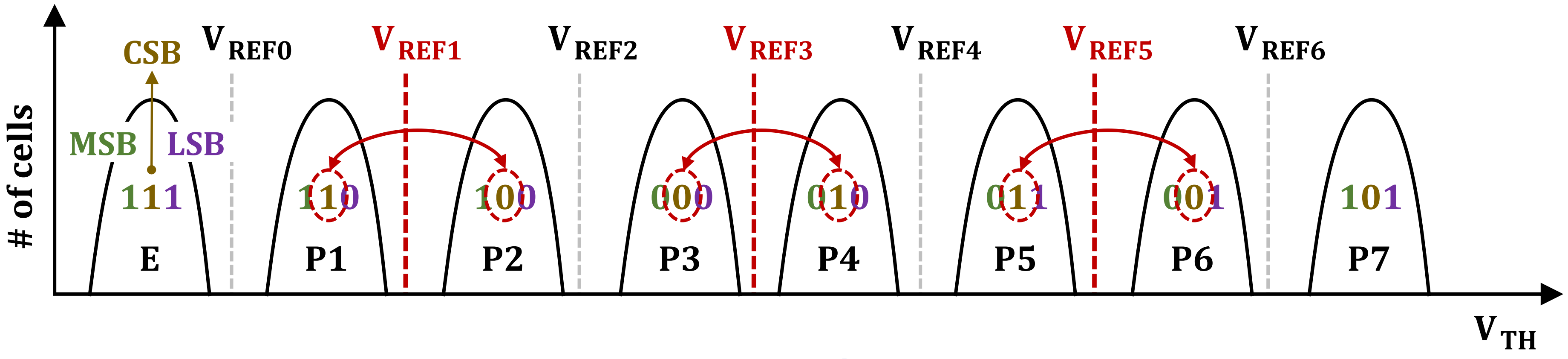




Basic Operation: Page Read - MLC

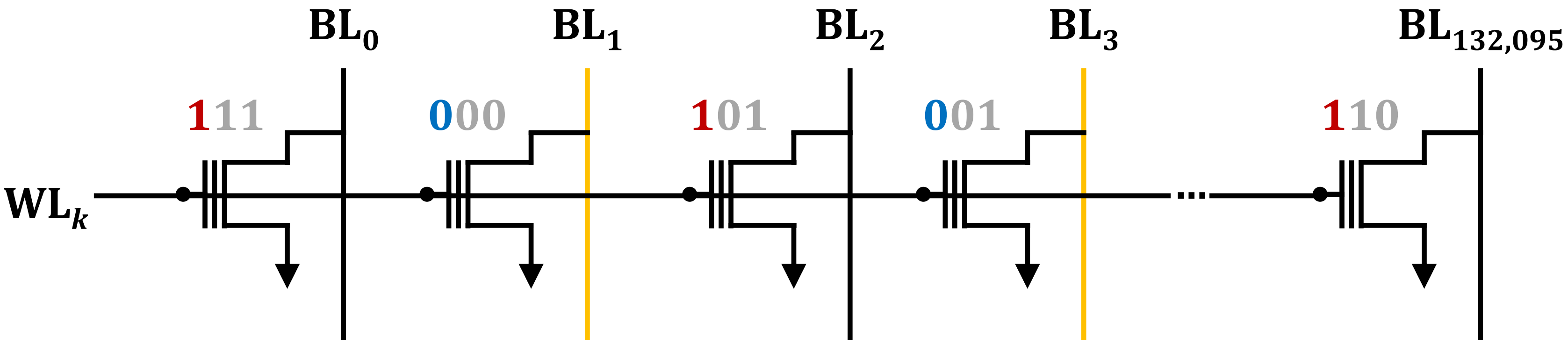
- Sensing the current through BLs

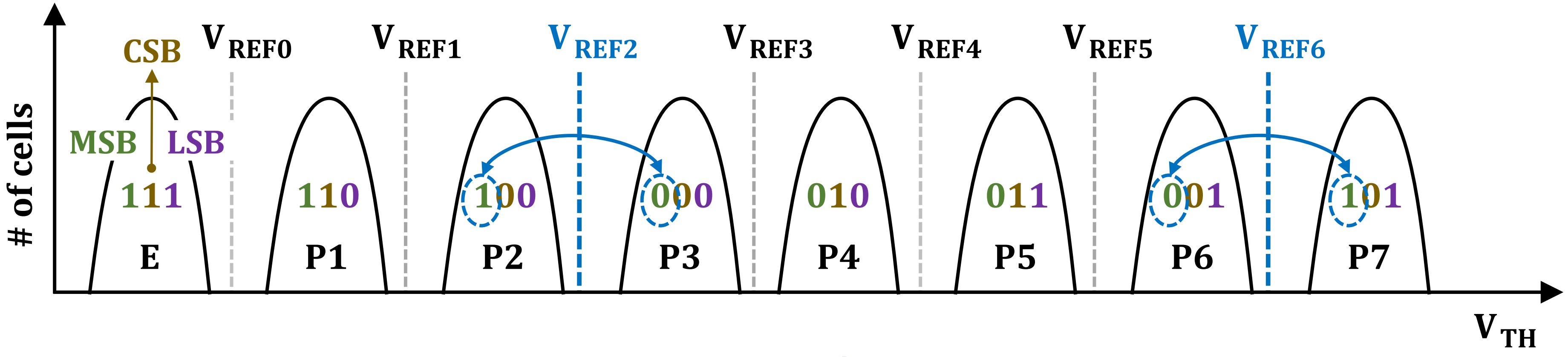




Basic Operation: Page Read - MLC

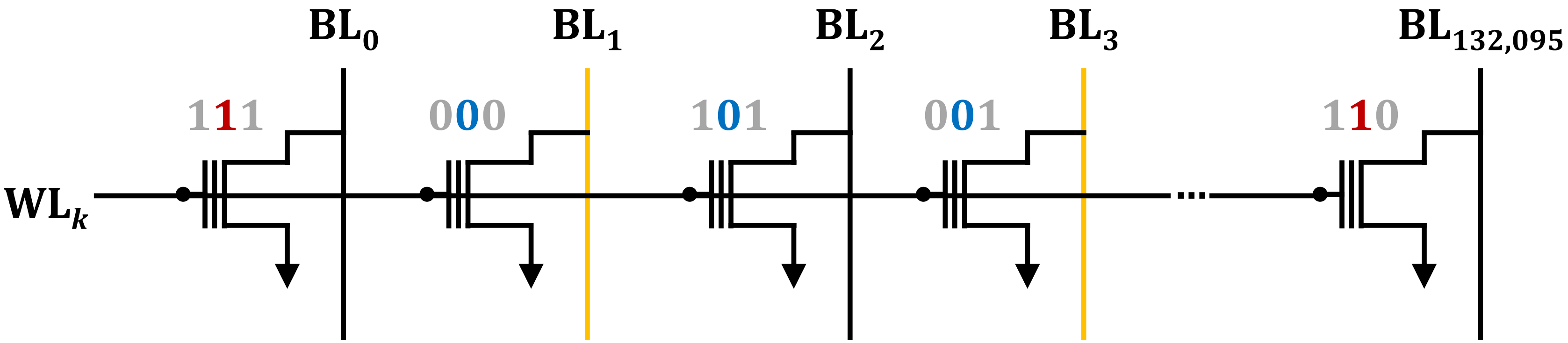
- Sensing the current through BLs

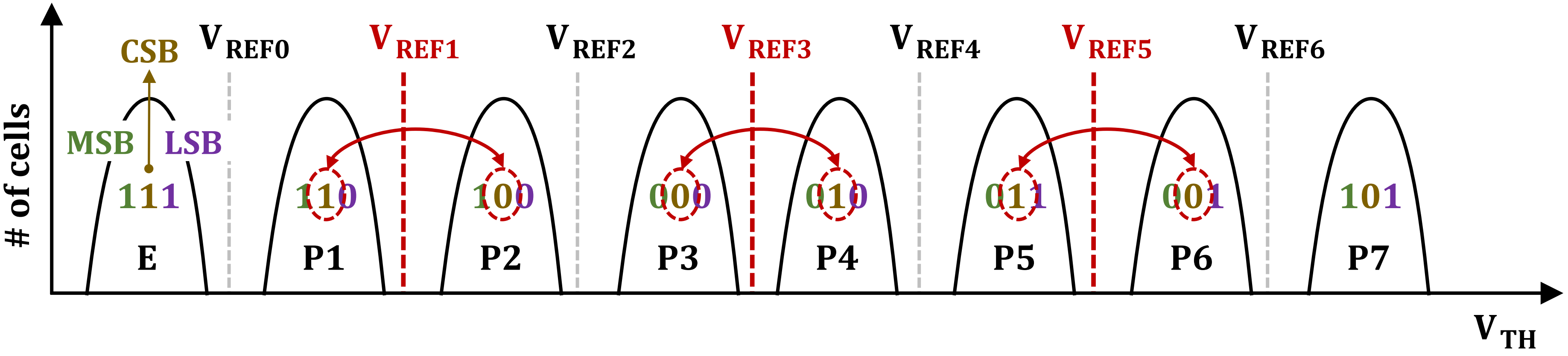




Basic Operation: Page Read - MLC

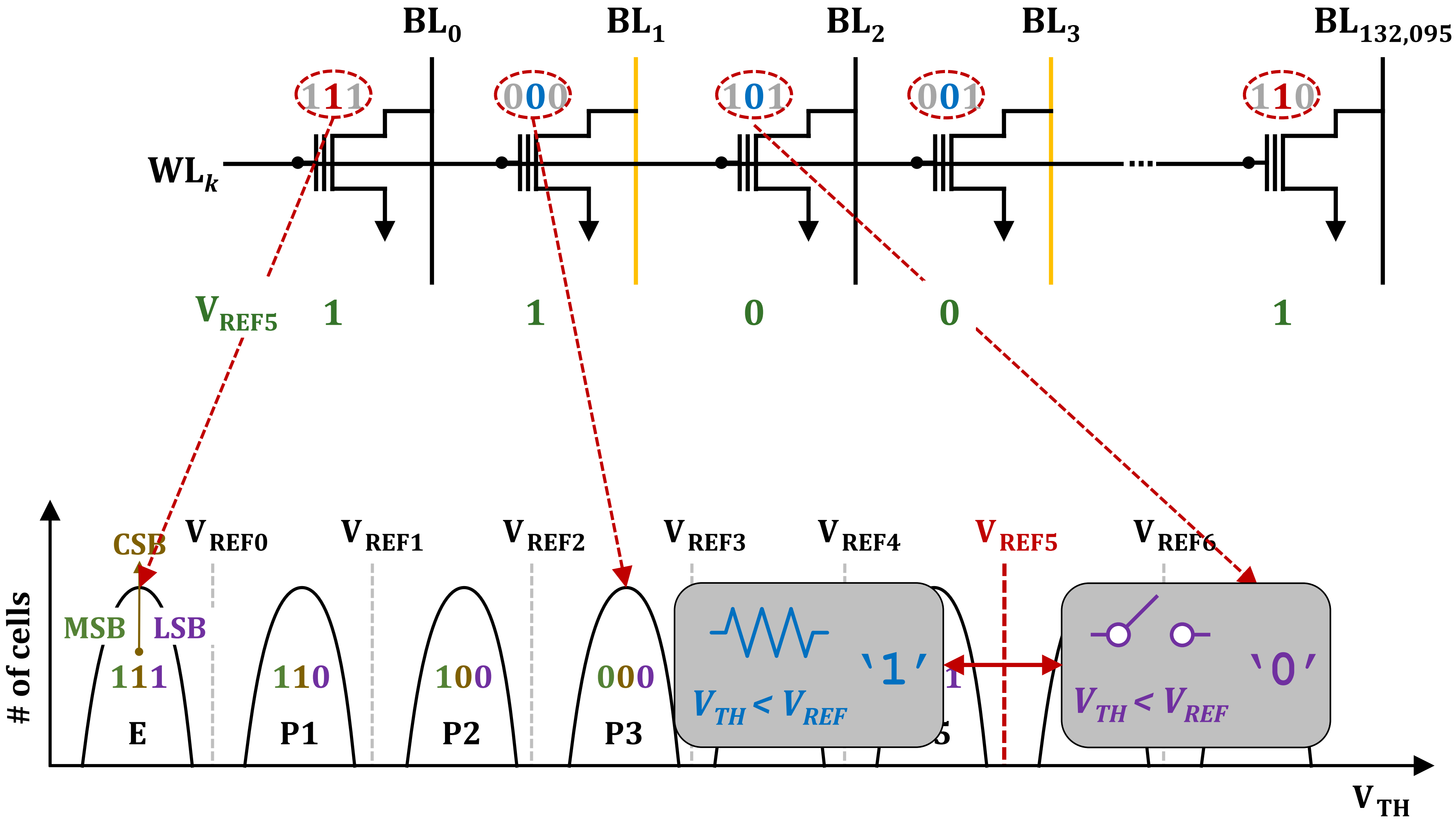
- Sensing the current through BLs





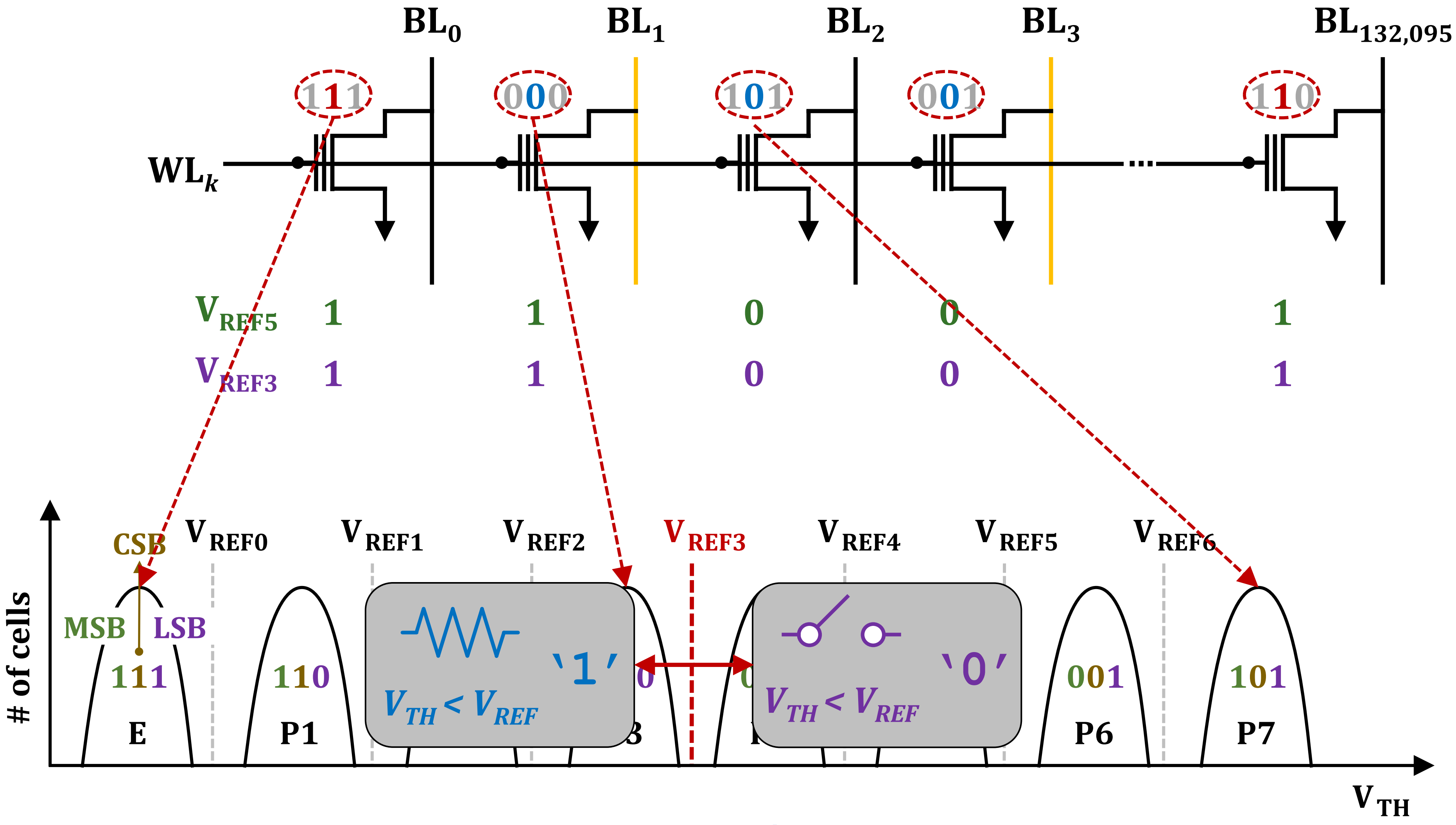
Basic Operation: Page Read - MLC

- Sensing the current through BLs



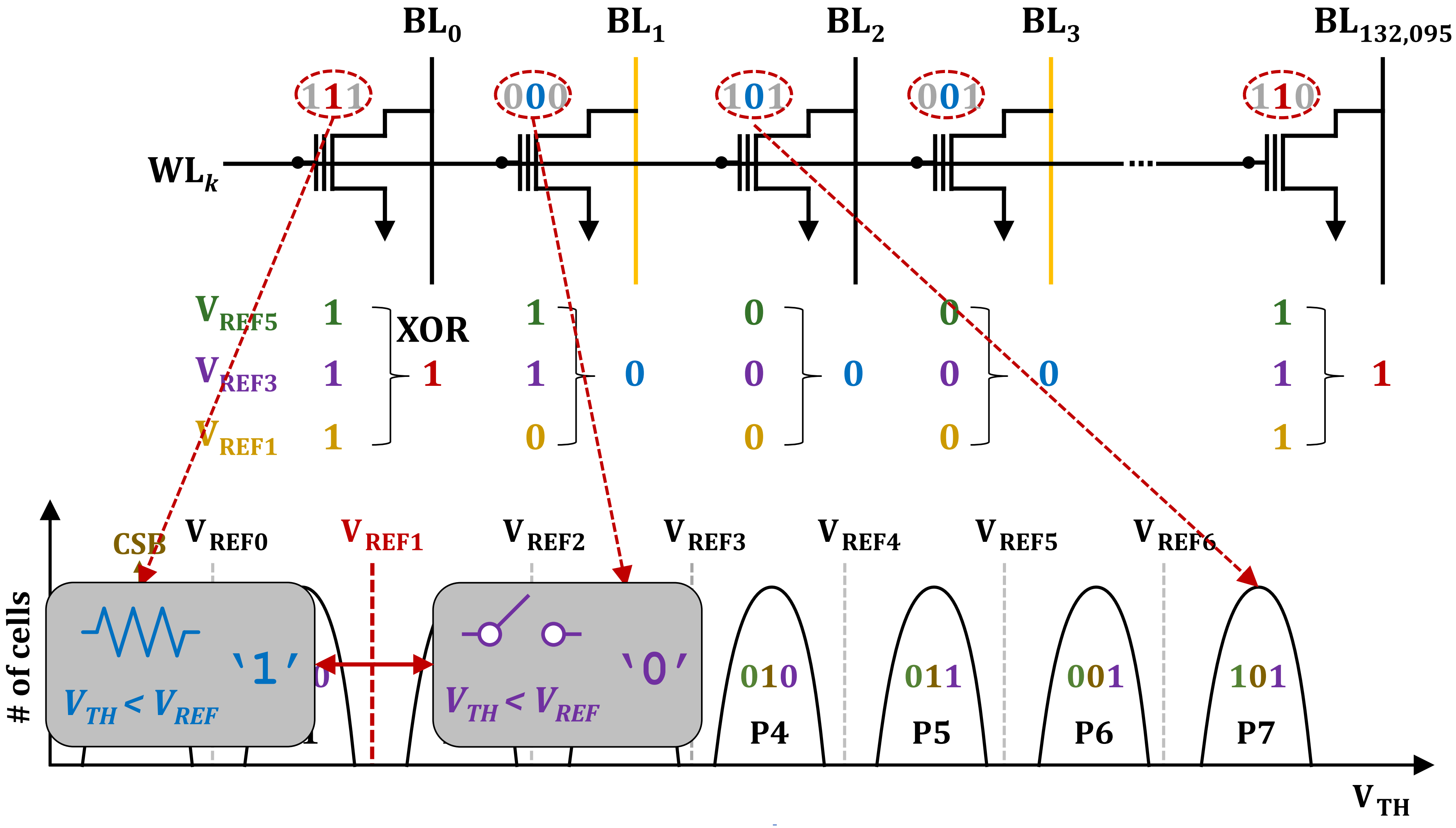
Basic Operation: Page Read - MLC

- Sensing the current through BLs



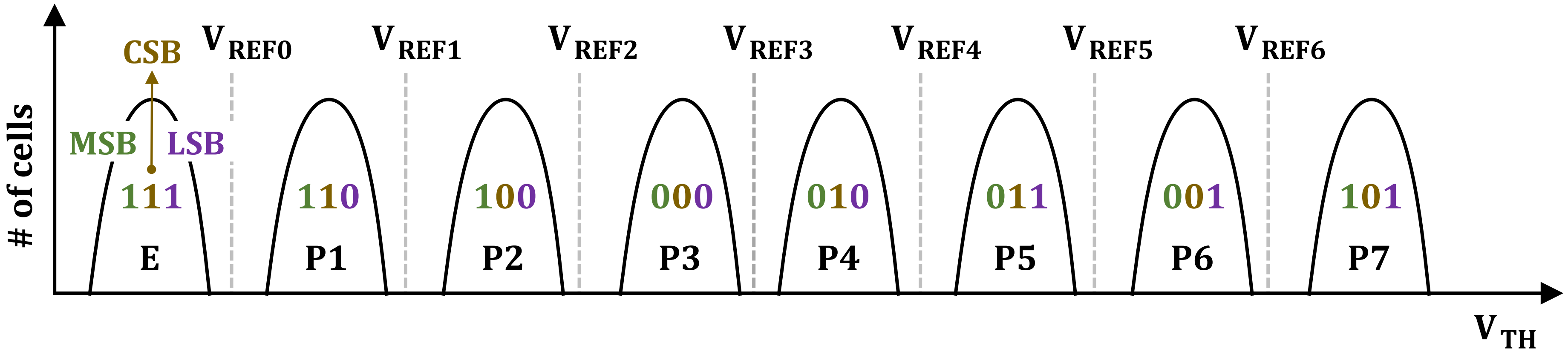
Basic Operation: Page Read - MLC

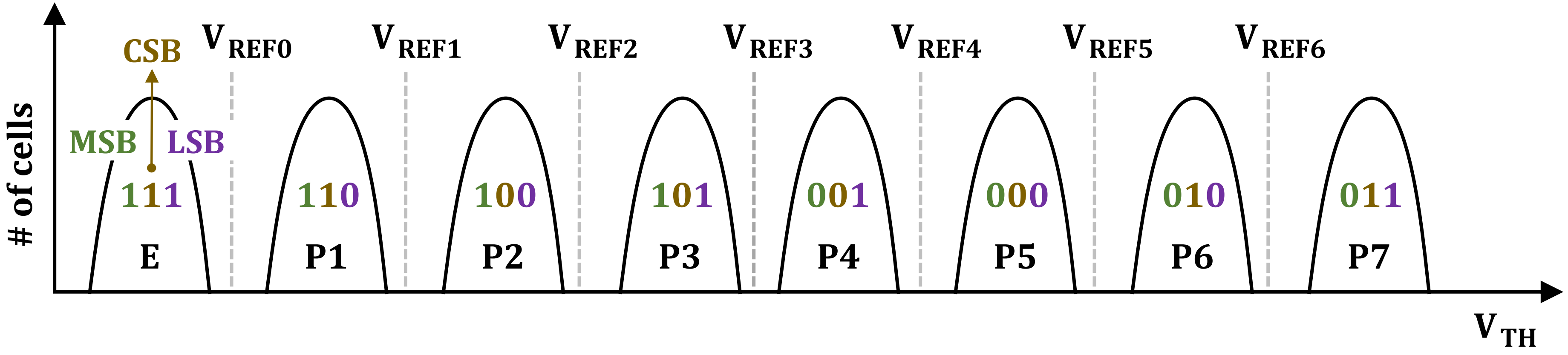
- Sensing the current through BLs



Basic Operation: Page Read – Takeaways

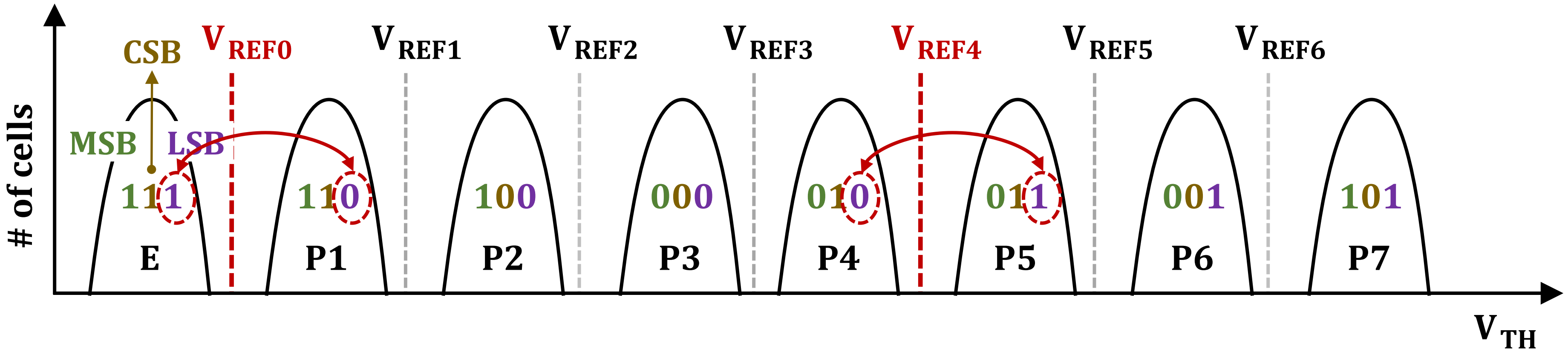
- MLC NAND flash memory requires an on-chip XOR logic
- Bit-encoding affects the read latency!
  - Compare # of sensing for LSB

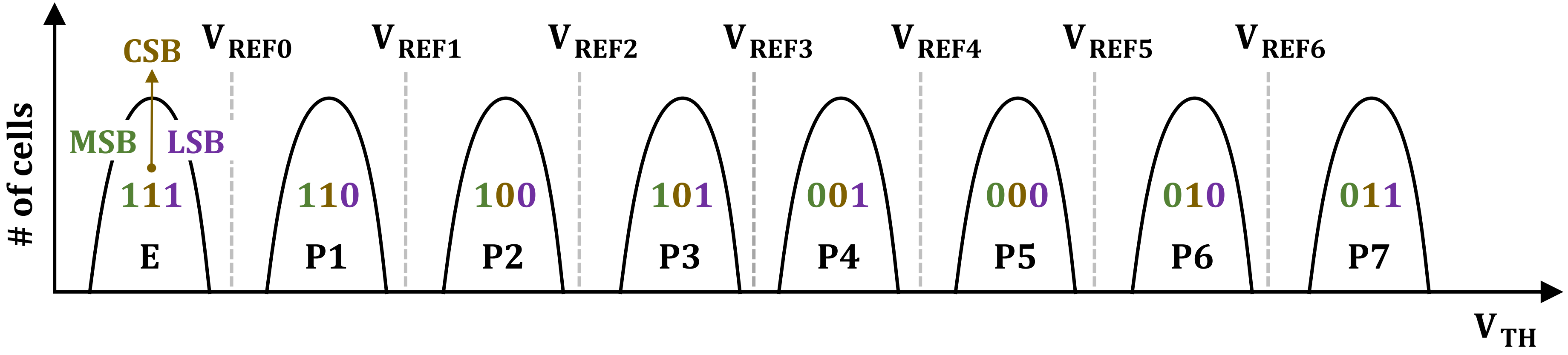




Basic Operation: Page Read – Takeaways

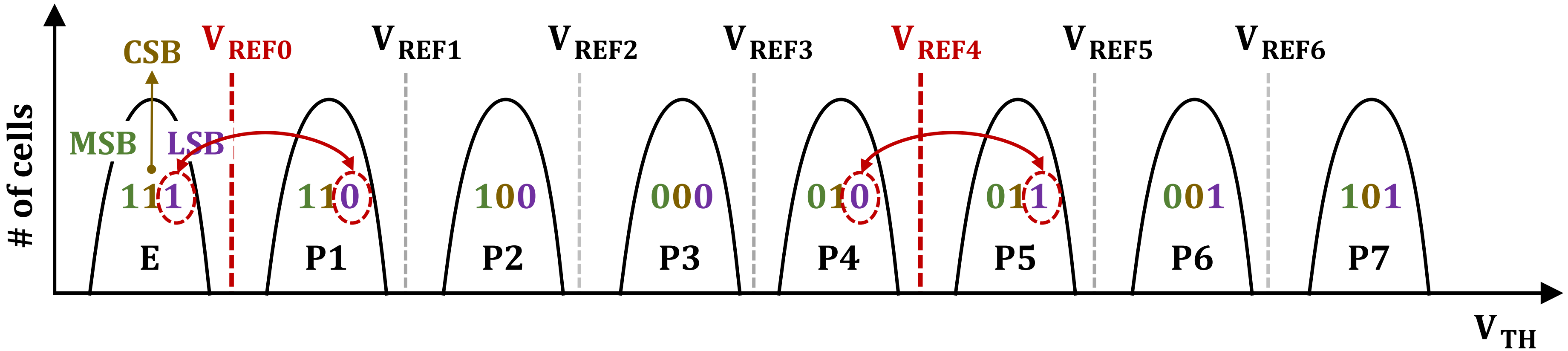
- MLC NAND flash memory requires an on-chip XOR logic
- Bit-encoding affects the read latency!
  - Compare # of sensing for LSB

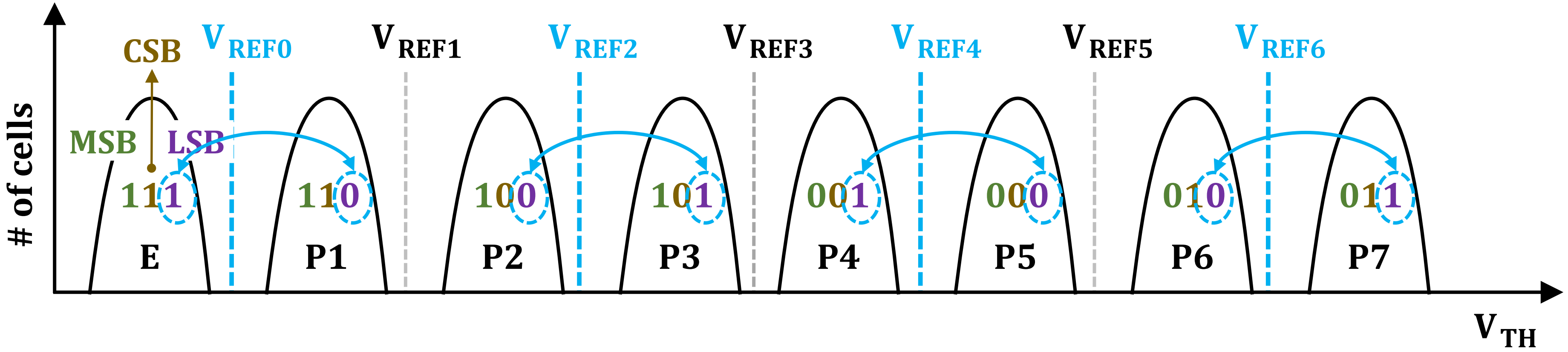




Basic Operation: Page Read – Takeaways

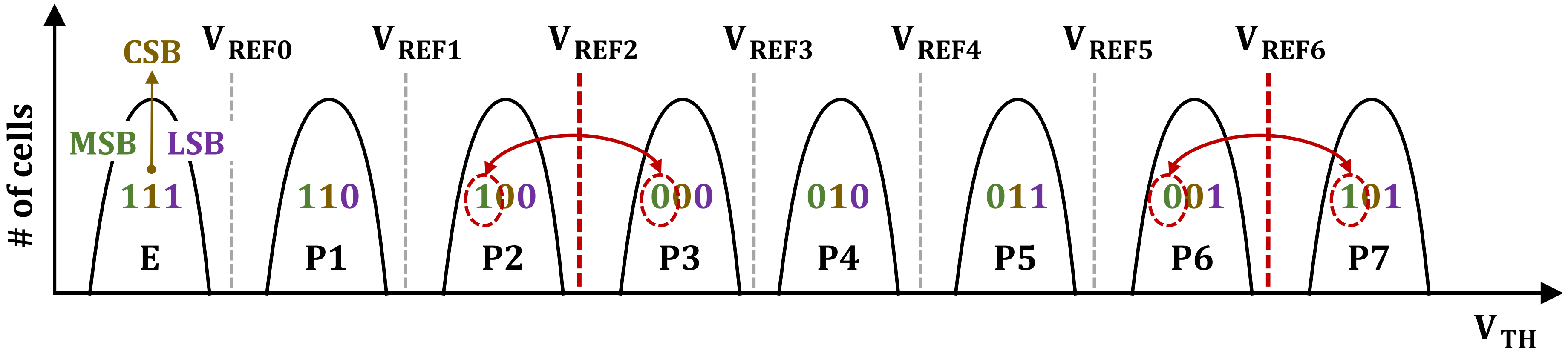
- MLC NAND flash memory requires an on-chip XOR logic
- Bit-encoding affects the read latency!
  - Compare # of sensing for LSB

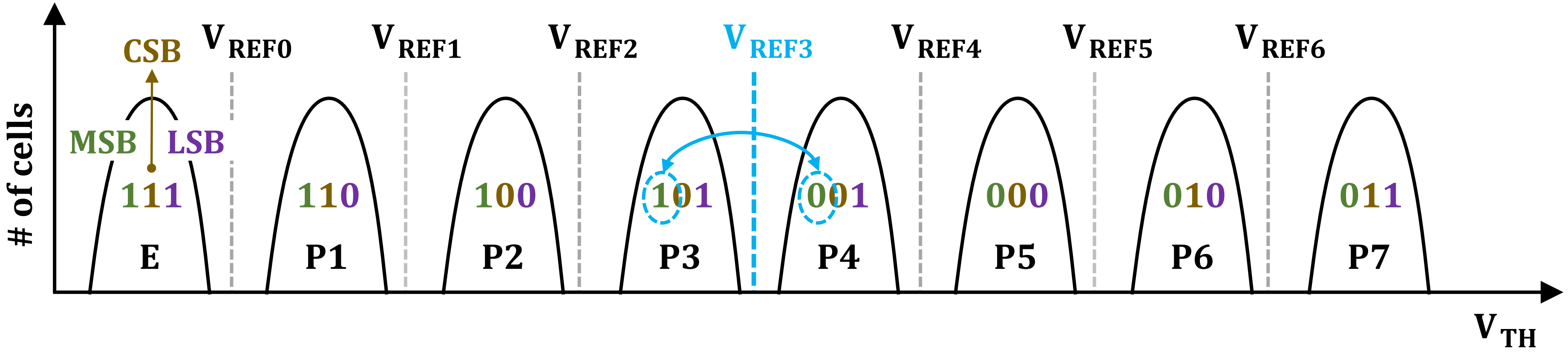




Basic Operation: Page Read – Takeaways

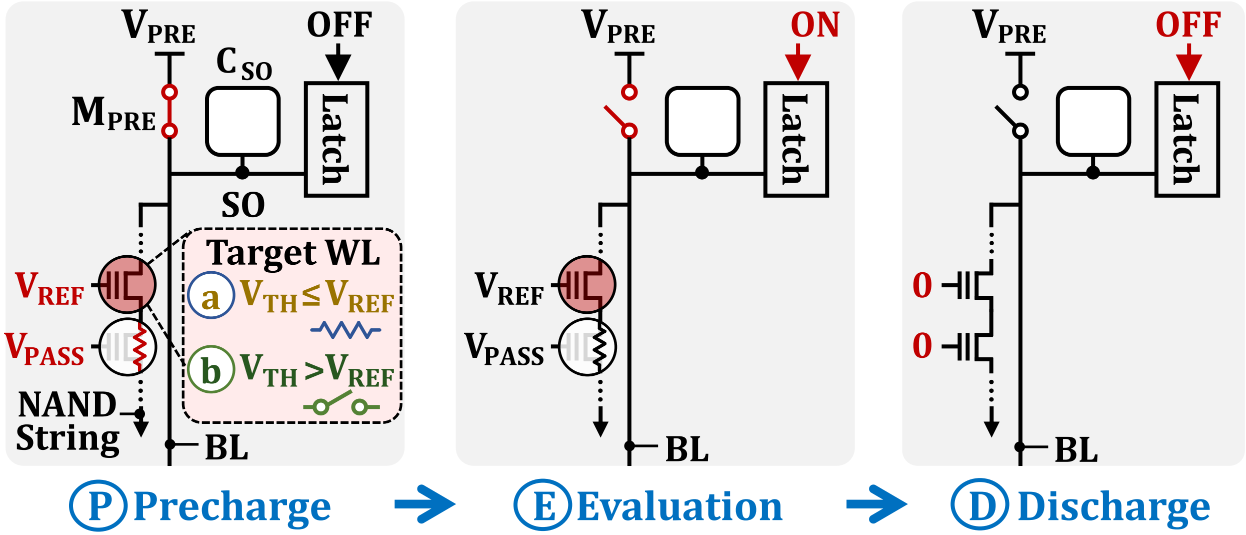
- MLC NAND flash memory requires an on-chip XOR logic
- Bit-encoding affects the read latency!
    - Compare # of sensing for LSB



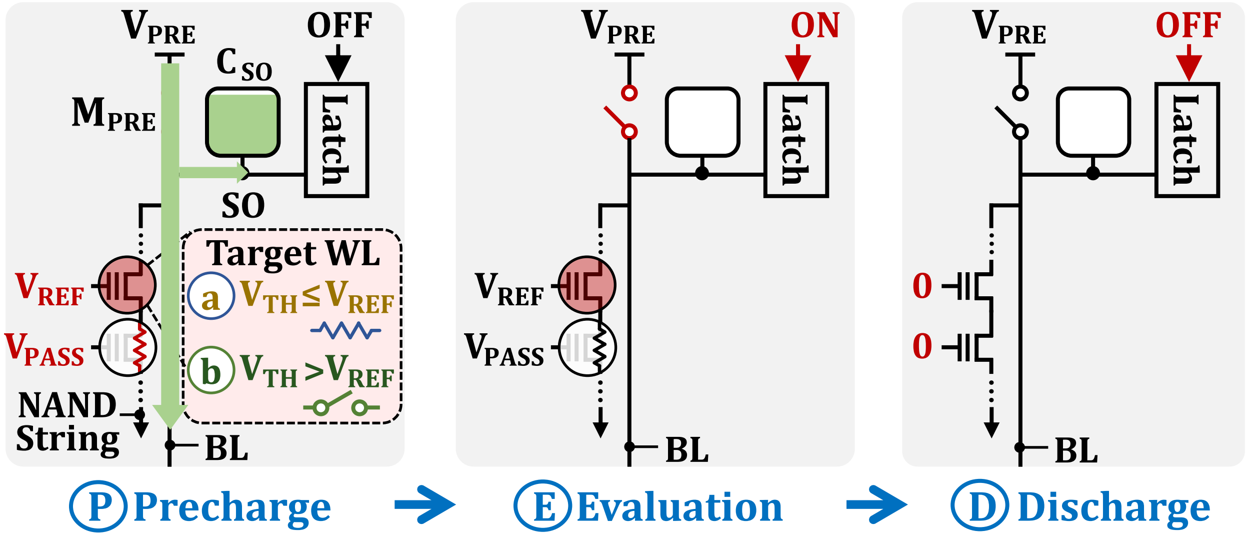


Read Mechanism

- NAND flash read mechanism consists of three steps:
  - 1) Precharge
  - 2) Evaluation
  - 3) Discharge

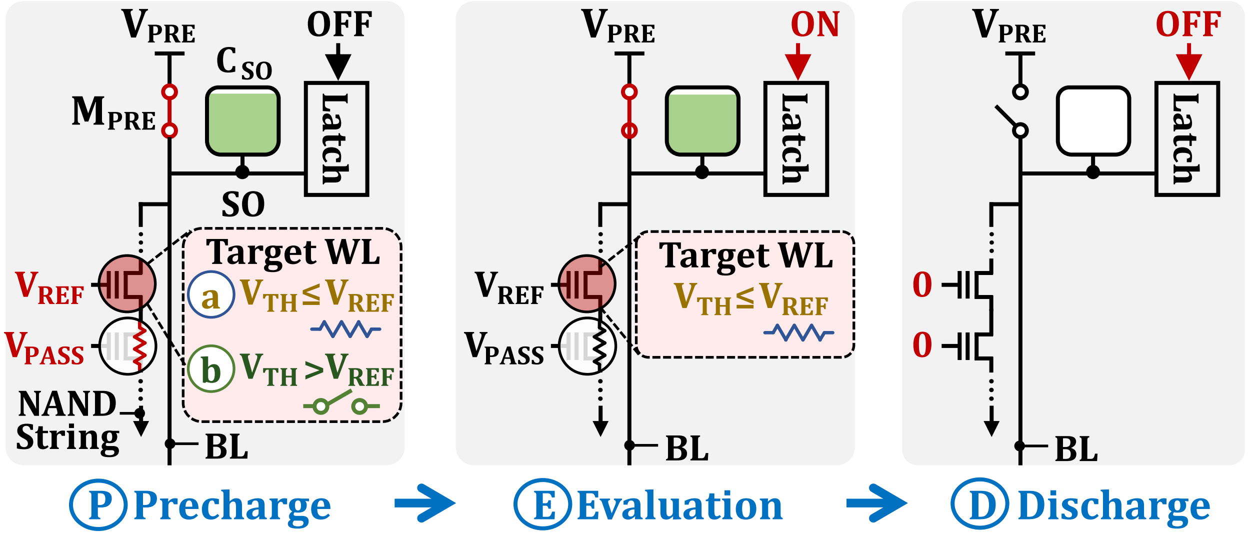


Read Mechanism: Precharge

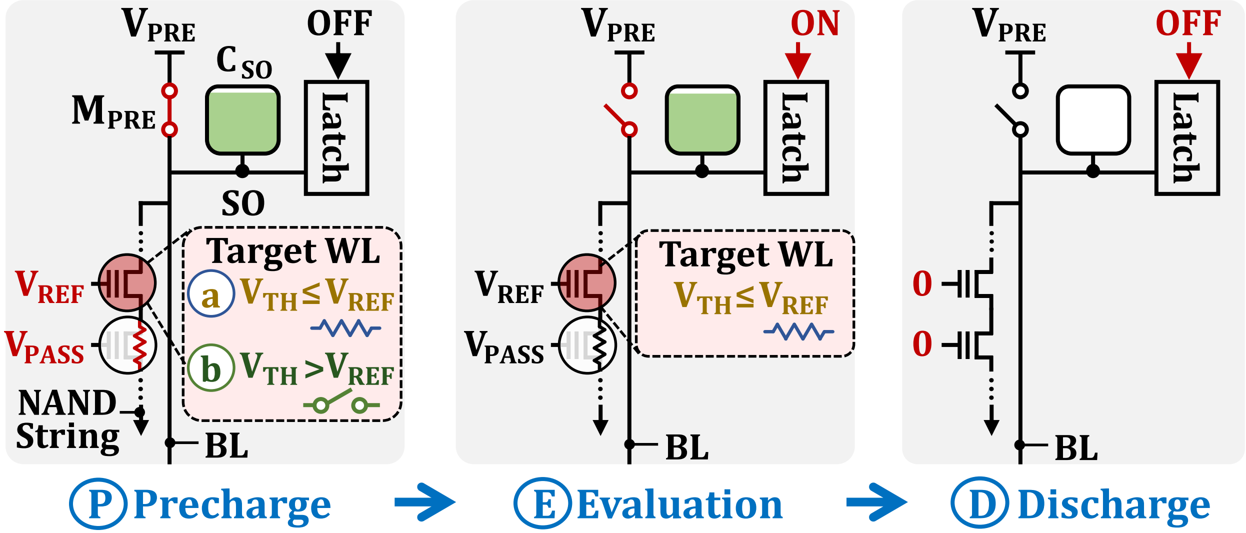


Enable precharge transistor M_{PRE} to charge all target BLs and their sense-out capacitors (C_{SO}) to V_{PRE}

Read Mechanism: Evaluation

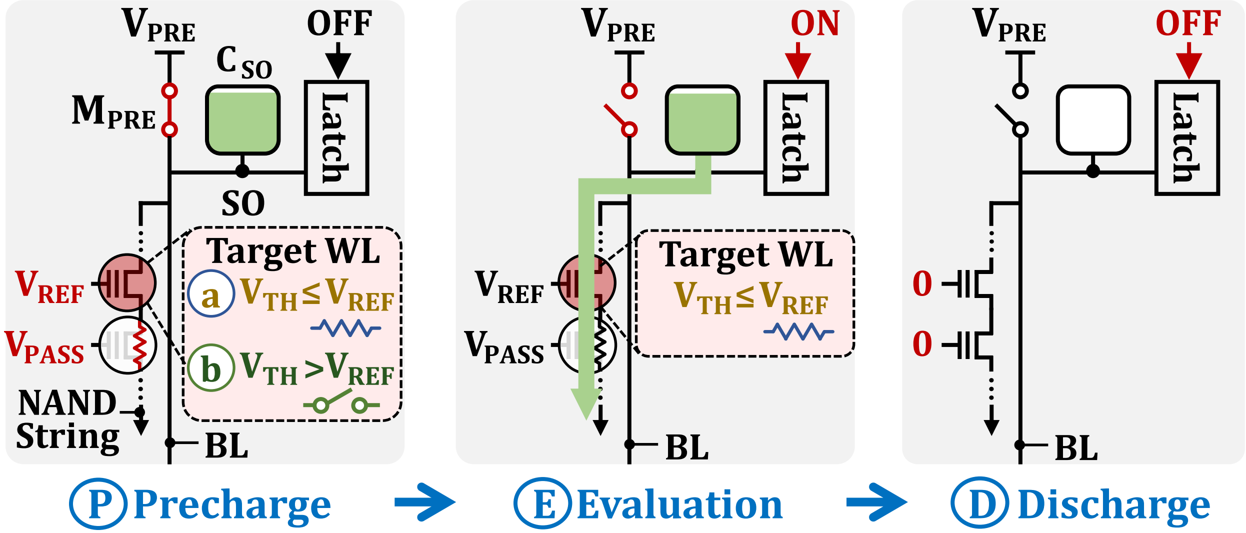


Read Mechanism: Evaluation



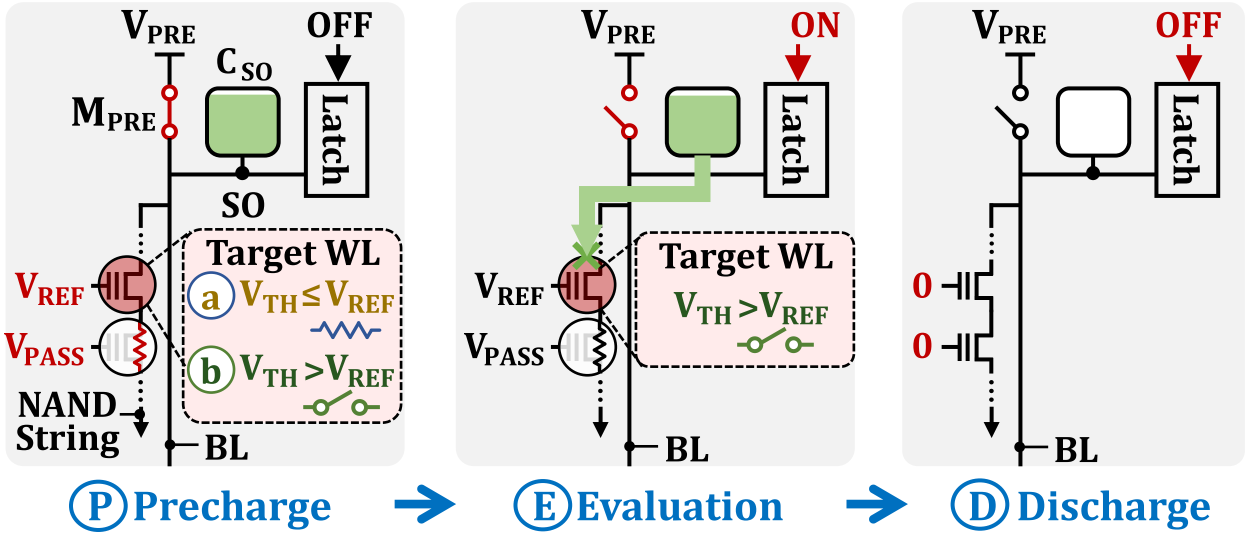
Disconnect the BLs from V_{PRE} and enable the latching circuit

Read Mechanism: Evaluation



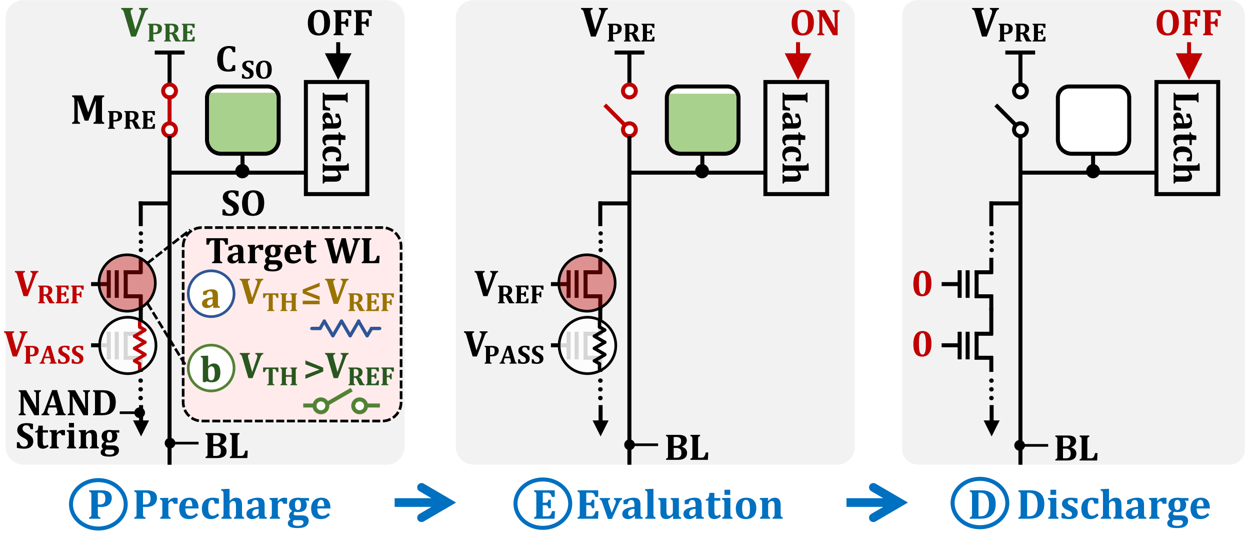
If $V_{TH} \leq V_{REF}$, the charge in C_{SO} quickly flows through the NAND string (Sensed as 1)

Read Mechanism: Evaluation



If $V_{TH} > V_{REF}$, the target cell blocks the BL discharge current (Sensed as 0)

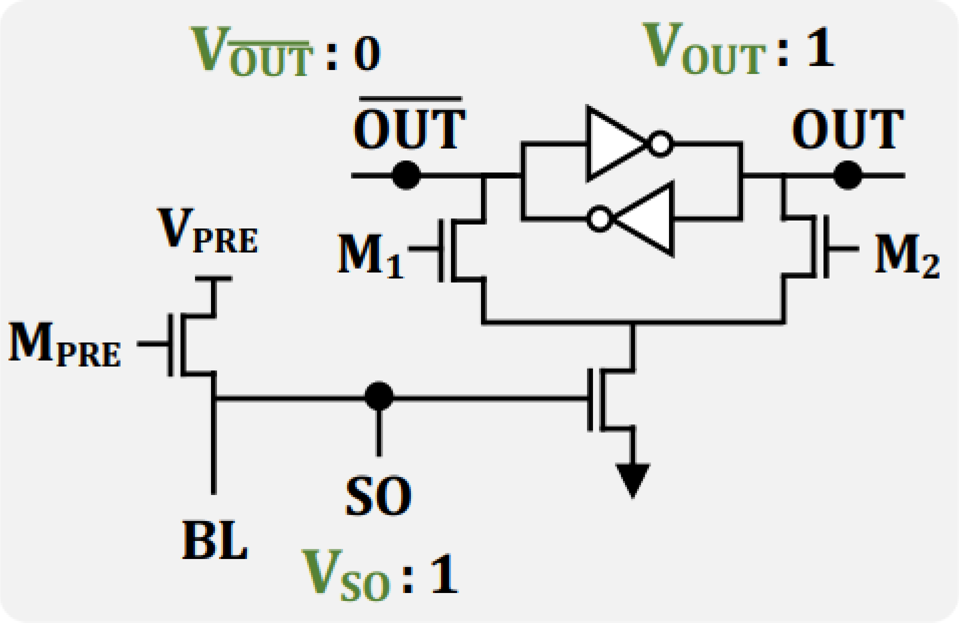
Read Mechanism: Discharge



Bitlines are discharged to return the NAND string to its initial state for future operations

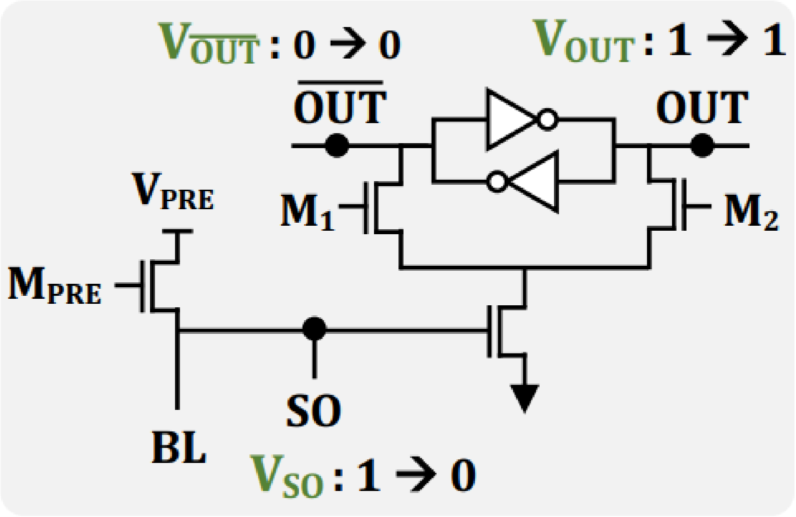
Latching Circuit

- Before the evaluation step, the chip initializes the latching circuit
  - Activating transistor M_1
  - $V_{\overline{OUT}} = 0$
  - $V_{OUT} = 1$

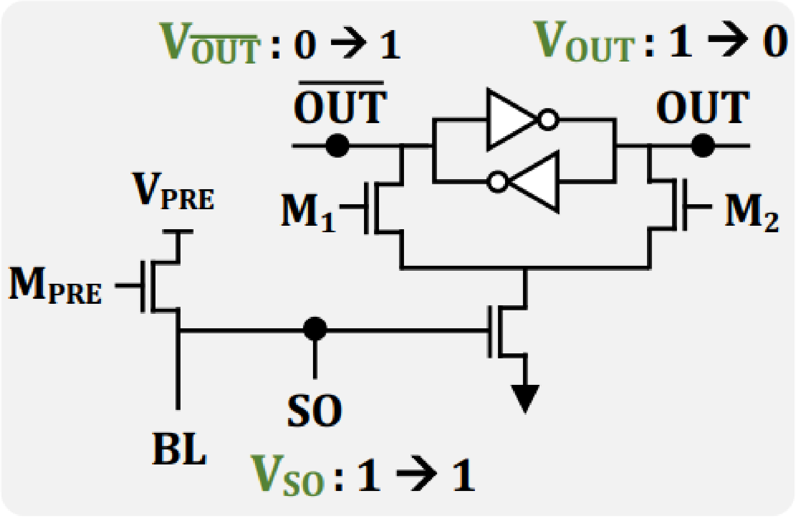


Latching Circuit

- The evaluation step
  - Disables M_{PRE} and M_1
  - Enables M_2



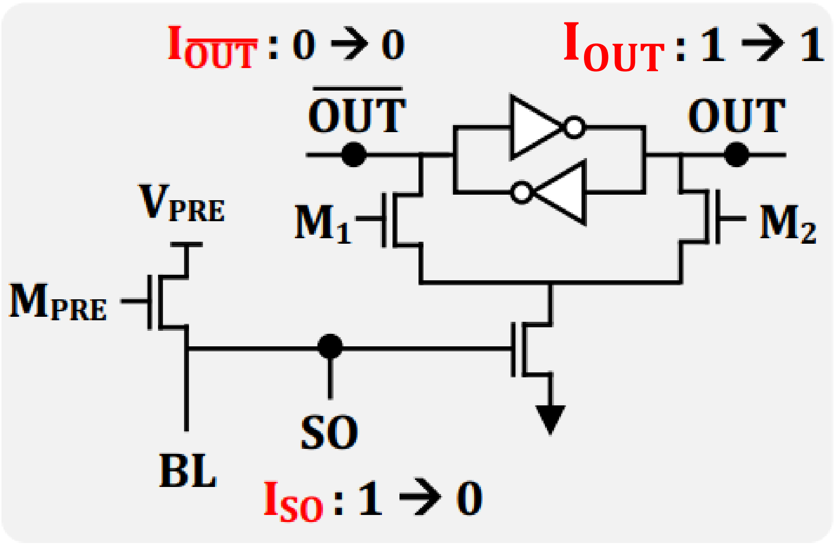
(a) $V_{TH} \leq V_{REF}$



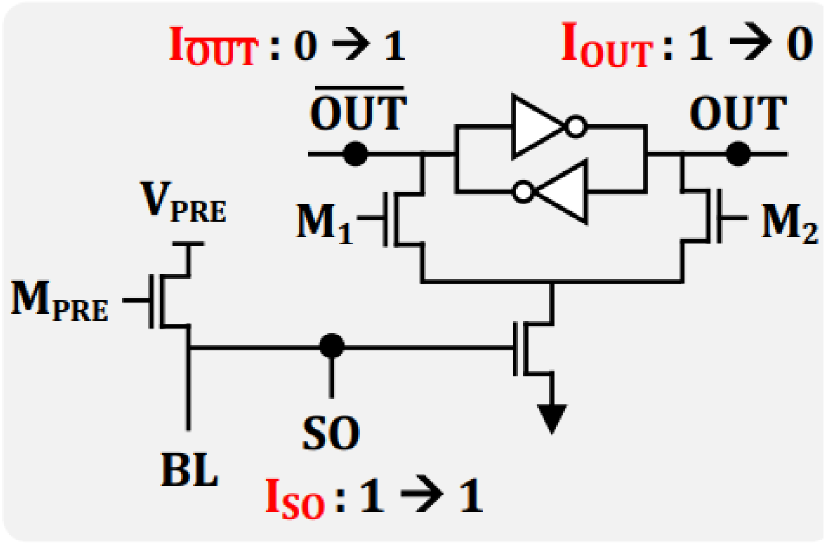
(b) $V_{TH} > V_{REF}$

Inverse Read

- Performing an inverse read by simply changing the activation sequence of M_1 and M_2
  - The precharge step activates M_2
  - The evaluation step disables M_2 and activates M_1



(a) $V_{TH} \leq V_{REF}$



(b) $V_{TH} > V_{REF}$

SSD Performance

- Latency (or response time)
  - The time delay until the request is returned
  - Average read latency (4 KiB): 67 us
  - Average write latency (4 KiB): 47 us

HDD:
5~8 ms

- Throughput
  - The number of requests that can be serviced per unit time
    - IOPS: Input/output Operations Per Second
  - Random read throughput: up to 500K IOPS
  - Random write throughput: up to 480K IOPS

HDD:
> 1K IOPS

- Bandwidth
  - The amount of data that can be accessed per unit time
  - Sequential read bandwidth: up to 3,500 MB/s
  - Sequential write bandwidth: up to 3,000 MB/s

HDD:
~100 MB/s

Source: https://www.anandtech.com/show/16504/the-samsung-ssd-980-500gb-1tb-review

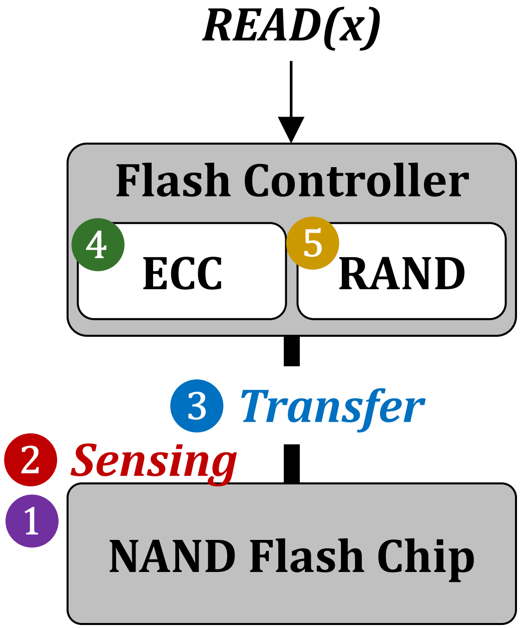
NAND Flash Chip Performance

- Chip operation latency
  - tR: Latency of reading (sensing) data from the cells into the on-chip page buffer
  - tPROG: Latency of programming the cells with data in the page buffer
  - tBERS: Latency of erasing the cells (block)
  - Varies depending on the MLC technology, processing node, and microarchitecture
    - In 3D TLC NAND flash, tR/tPROG/tBERS ≈ 100us/700us/3ms
- I/O rate
  - Number of bits transferred via a single I/O pin per unit time
  - A typical flash chip transfers data in a byte granularity (i.e., via 8 I/O pins)
  - e.g., 1-Gb I/O rate & 16-KiB page size → tDMA = 16 us

NAND Flash Chip Performance (Cont.)

- tR, tPROG, and tBERS
  - Latencies for chip-level read/program/erase operations
  - tR: 50~100 us
  - tPROG: 700us~1000 us
  - tBERS: 3ms~5ms

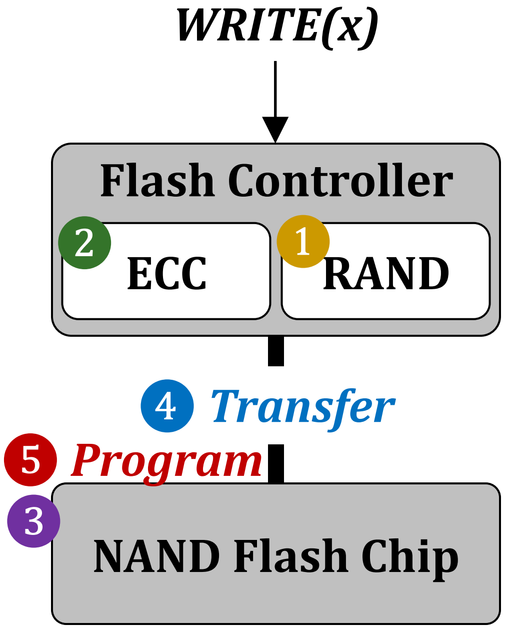
- Flash-controller level latency
  - 1-Gb I/O rate and 16-KiB page size
  - Read
    - (tCMD) + tR + tDMA + $tECC_{DEC}$ + (tRND)
    - e.g., 100 + 16 + 20 = 136 us



NAND Flash Chip Performance (Cont.)

- tR, tPROG, and tBERS
  - Latencies for chip-level read/program/erase operations
  - tR: 50~100 us
  - tPROG: 700us~1000 us
  - tBERS: 3ms~5ms

- Flash-controller level latency
  - 1-Gb I/O rate and 16-KiB page size
  - Read
    - (tCMD) + tR + tDMA + $\mathrm{tECC}_{\mathrm{DEC}}$ + (tRND)
    - e.g., 100 + 16 + 20 = 136 us
  - Program
    - (tRND) + $\mathrm{tECC}_{\mathrm{ENC}}$ + (tCMD) + tDMA + tPROG
    - e.g., 20 + 16 + 700 = 736 us



NAND Flash Chip Performance (Cont.)

- How about bandwidth?
  - Read
    - 16 KiB / 136 us ≈ 120 MB/s
  - Write
    - 16 KiB / 736 us ≈ 22 MB/s

Optimizations w/ advanced commands

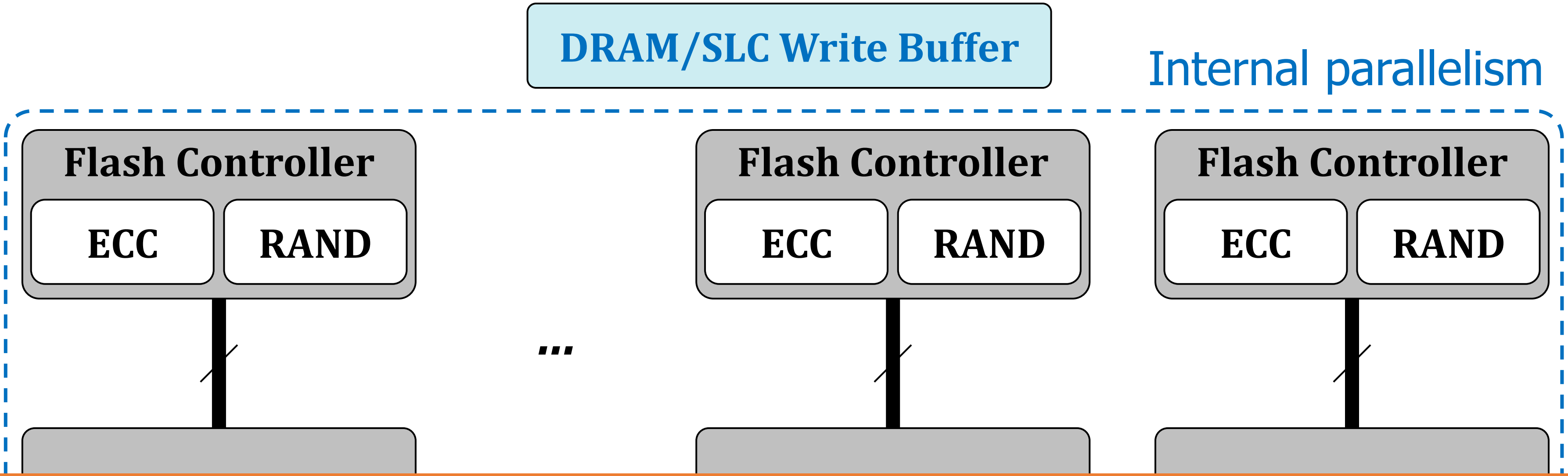
WAIT!

SSD read latency: 67 us

SSD read bandwidth: 3.5 GB/s

SSD write latency: 47 us

SSD write bandwidth: 3 GB/s



Summary: The NAND Flash chip bandwidth is higher than your estimation due to some parallelism enhancement techniques and optimizations!

Flash Error Correct and Refresh

(本節內容改自Yu Cai, …, Onur Mutlu et al., “Flash Correct-and-Refresh: Retention-Aware Error Management for Increased Flash Memory Lifetime, ” ICCD’12 slides.)

Problem: Limited Endurance of Flash Memory

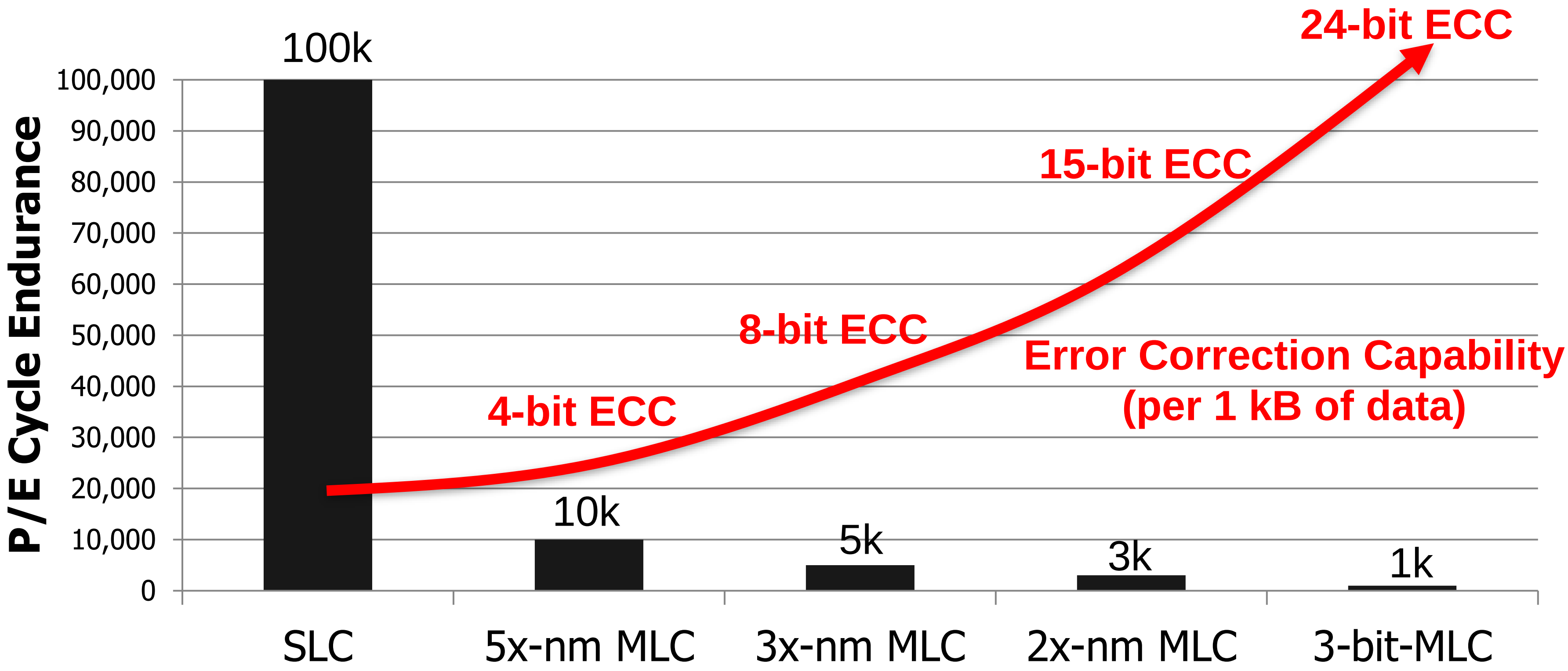
- NAND flash has limited endurance
  - A cell can tolerate a small number of Program/Erase (P/E) cycles
  - 3x-nm flash with 2 bits/cell → 3K P/E cycles

- Enterprise data storage requirements demand very high endurance
  - >50K P/E cycles (10 full disk writes per day for 3-5 years)

- Continued process scaling and more bits per cell will reduce flash endurance

- One potential solution: stronger error correction codes (ECC)
  - Stronger ECC not effective enough and inefficient

Decreasing Endurance with Flash Scaling



Ariel Maislos, “A New Era in Embedded Flash Memory”, Flash Summit 2011 (Anobit)

- Endurance of flash memory decreasing with scaling and multi-level cells
- Error correction capability required to guarantee storage-class reliability (UBER $< 10^{-15}$) is increasing exponentially to reach *less* endurance

UBER: Uncorrectable bit error rate. Fraction of erroneous bits after error correction.

The Problem with Stronger Error Correction

- Stronger ECC detects and corrects more raw bit errors → increases P/E cycles endured

- Two shortcomings of stronger ECC:

  1. High implementation complexity

     → Power and area overheads increase super-linearly, but correction capability increases sub-linearly with ECC strength

  2. Diminishing returns on flash lifetime improvement

     → Raw bit error rate increases exponentially with P/E cycles, but correction capability increases sub-linearly with ECC strength

Methodology: Error and ECC Analysis

- Characterized errors and error rates of 3x-nm MLC NAND flash using an experimental FPGA-based flash platform
  - Cai et al., "Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis," DATE 2012.

- Quantified Raw Bit Error Rate (RBER) at a given P/E cycle
  - Raw Bit Error Rate: Fraction of erroneous bits without any correction

- Quantified error correction capability (and area and power consumption) of various BCH-code implementations
  - Identified how much RBER each code can tolerate

    → how many P/E cycles (flash lifetime) each code can sustain

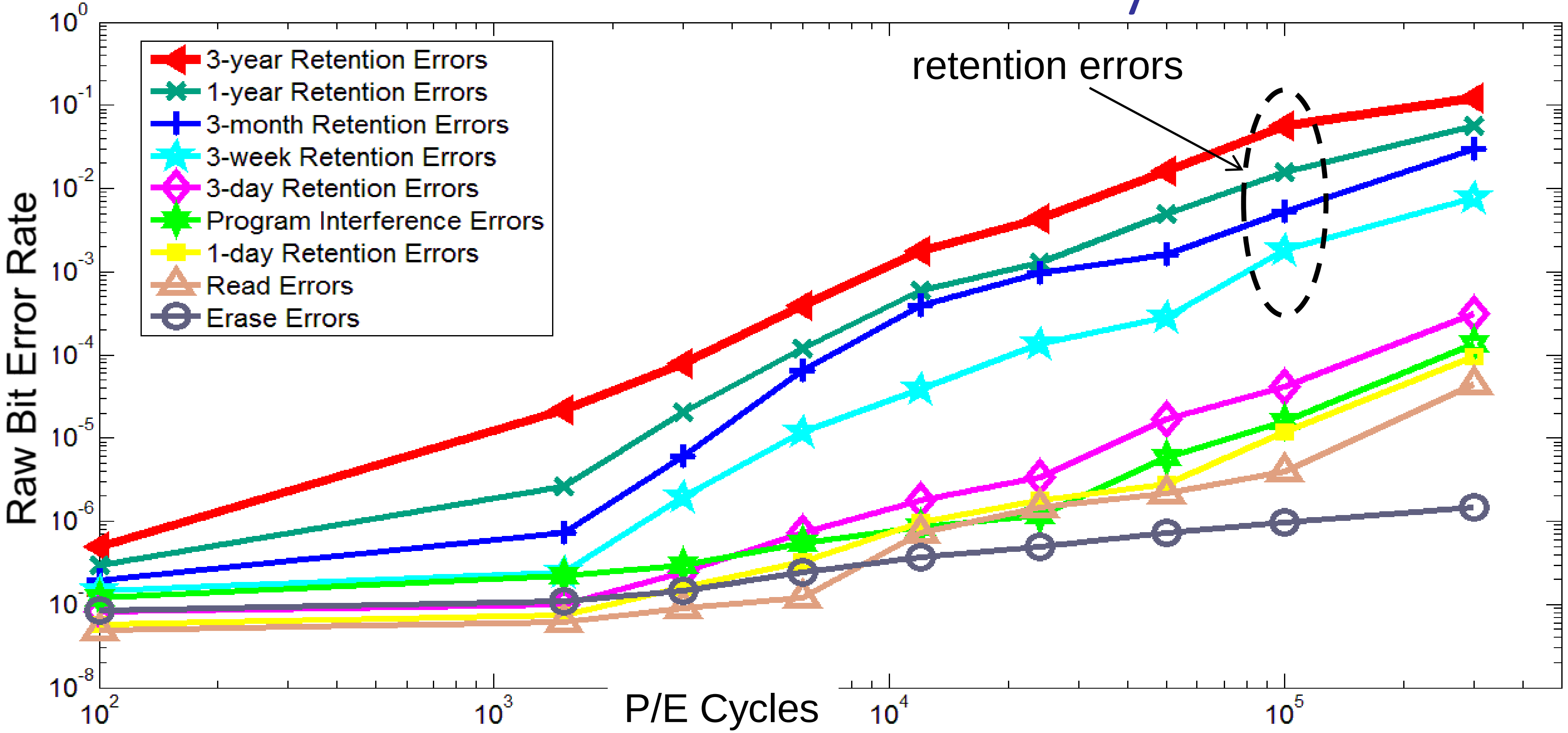
NAND Flash Error Types

- Four types of errors [Cai+, DATE 2012]

- Caused by common flash operations
  - Read errors
  - Erase errors
  - Program (interference) errors

- Caused by flash cell losing charge over time
  - Retention errors
    - Whether an error happens depends on required retention time
    - Especially problematic in MLC flash because voltage threshold window to determine stored value is smaller

Observations: Flash Error Analysis



- Raw bit error rate increases exponentially with P/E cycles
- Retention errors are dominant (>99% for 1-year ret. time)
- Retention errors increase with retention time requirement

Methodology: Error and ECC Analysis

- Characterized errors and error rates of 3x-nm MLC NAND flash using an experimental FPGA-based flash platform
  - Cai et al., "Error Patterns in MLC NAND Flash Memory: Measurement, Characterization, and Analysis," DATE 2012.

- Quantified Raw Bit Error Rate (RBER) at a given P/E cycle
  - Raw Bit Error Rate: Fraction of erroneous bits without any correction

- Quantified error correction capability (and area and power consumption) of various BCH-code implementations
  - Identified how much RBER each code can tolerate

    → how many P/E cycles (flash lifetime) each code can sustain

ECC Strength Analysis

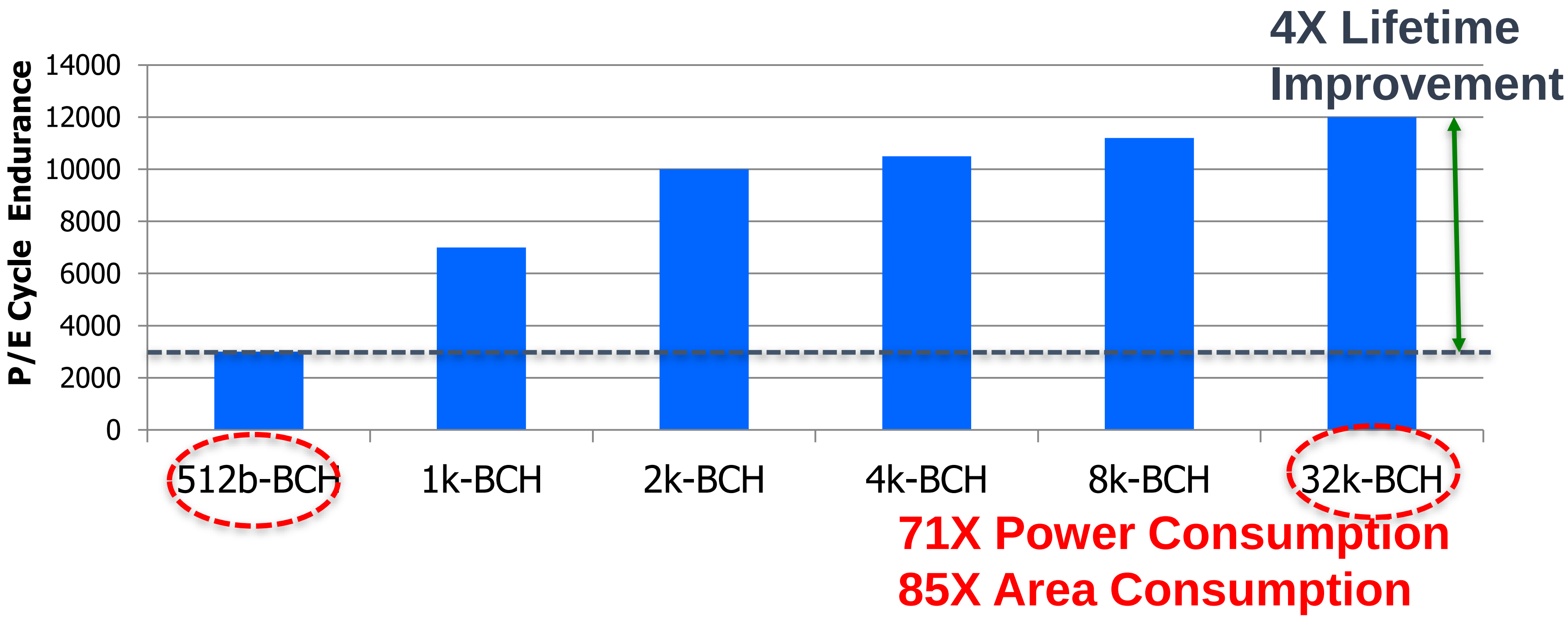
Error correction capability increases sub-linearly

Power and area overheads increase super-linearly

Code length (n)	Correctable Errors (t)	Acceptable Raw BER	Norm. Power	Norm. Area
512	7	$1.0\text{x}10^{-4}$ (1x)	1	1
1024	12	$4.0\text{x}10^{-4}$ (4x)	2	2.1
2048	22	$1.0\text{x}10^{-3}$ (10x)	4.1	3.9
4096	40	$1.7\text{x}10^{-3}$ (17x)	8.6	10.3
8192	74	$2.2\text{x}10^{-3}$ (22x)	17.8	21.3
32768	259	$2.6\text{x}10^{-3}$ (26x)	71	85

Resulting Flash Lifetime with Strong ECC

- Lifetime improvement comparison of various BCH codes



Strong ECC is very inefficient at improving lifetime

Flash Correct-and-Refresh (FCR)

- Key Observations:
  - Retention errors are the dominant source of errors in flash memory [Cai+ DATE 2012][Tanakamaru+ ISSCC 2011]

    → limit flash lifetime as they increase over time
  - Retention errors can be corrected by "refreshing" each flash page periodically

- Key Idea:
  - Periodically read each flash page,
  - Correct its errors using "weak" ECC, and
  - Either remap it to a new physical page or reprogram it in-place,
  - Before the page accumulates more errors than ECC-correctable
  - Optimization: Adapt refresh rate to endured P/E cycles

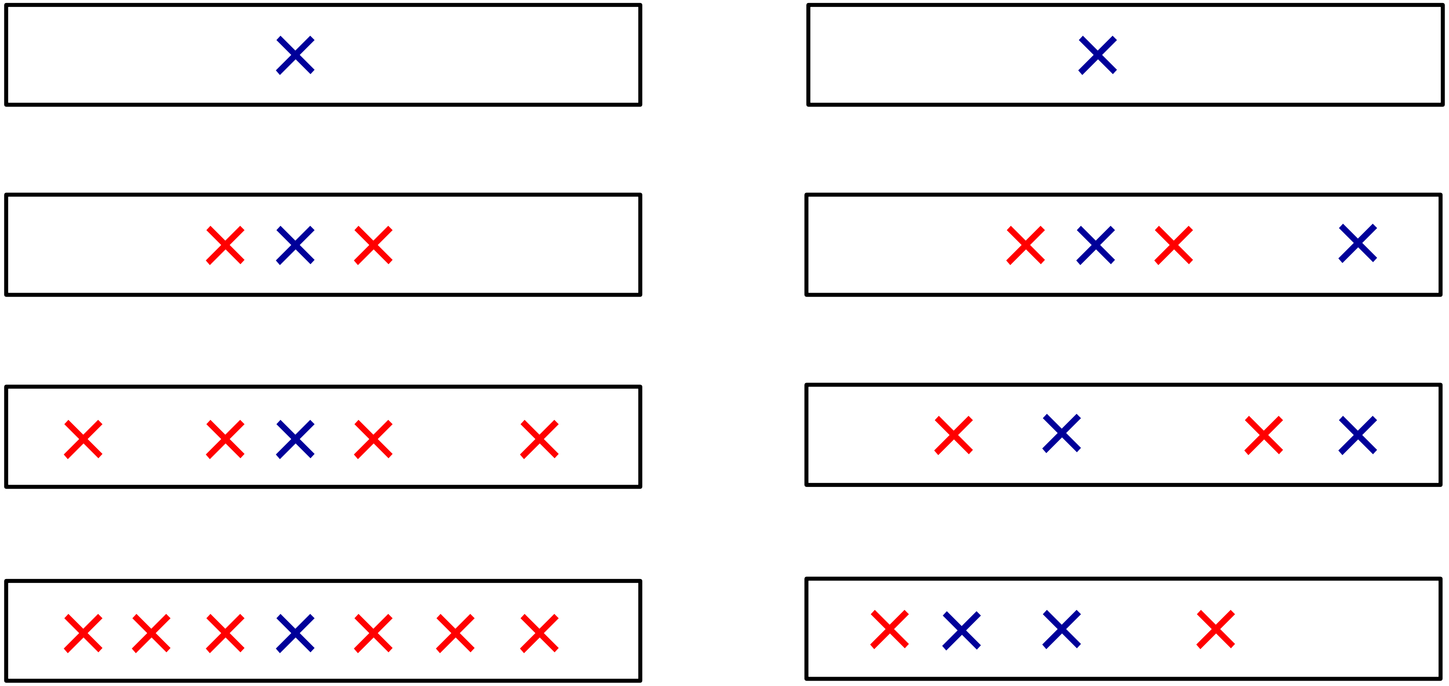
FCR Intuition

Errors with

No refresh

Errors with

Periodic refresh



Program Page

After time T

After time 2T

After time 3T

✕Retention Error ✕Program Error

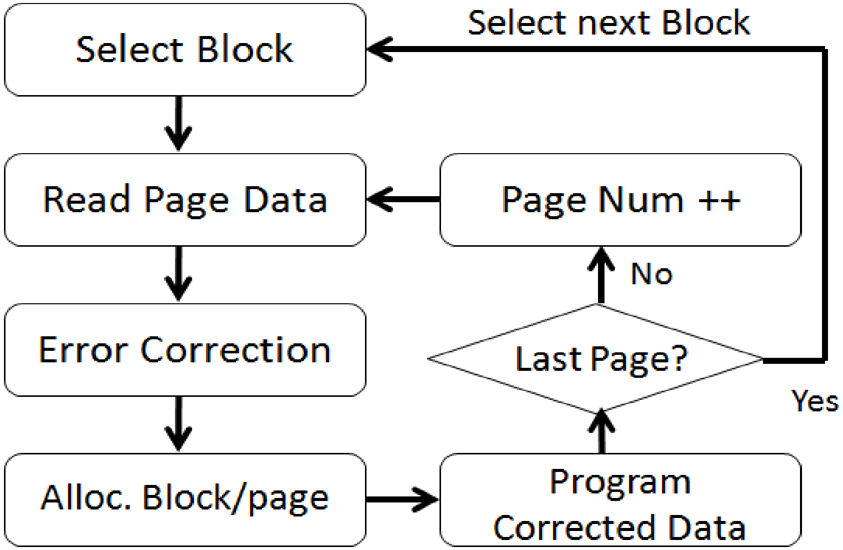
FCR: Two Key Questions

- How to refresh?
  - Remap a page to another one
  - Reprogram a page (in-place)
  - Hybrid of remap and reprogram

- When to refresh?
  - Fixed period
  - Adapt the period to retention error severity

Remapping Based FCR

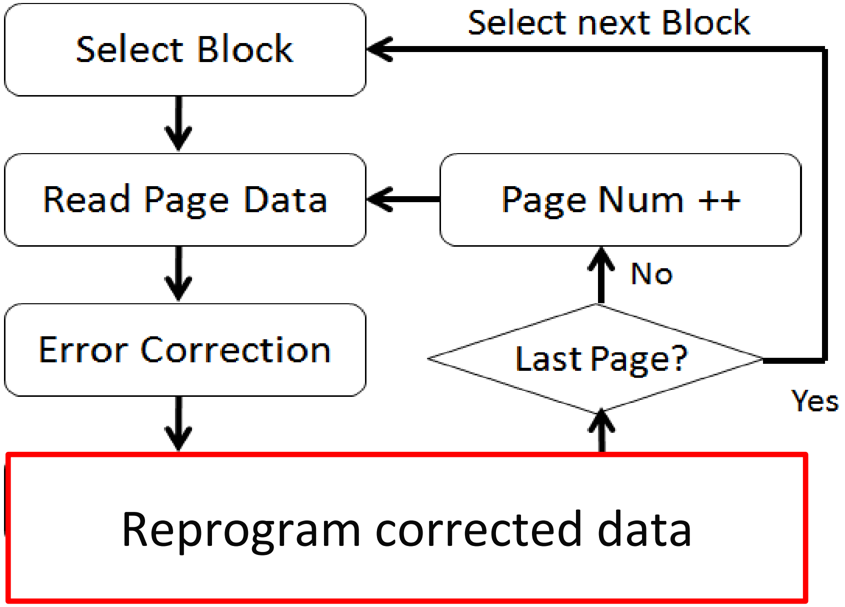
- Idea: Periodically remap each page to a different physical page (after correcting errors)
  - ❑Also [Pan et al., HPCA 2012]
  - ❑FTL already has support for changing logical → physical flash block/page mappings
  - ❑Deallocated block is erased by garbage collector



- Problem: Causes additional erase operations → more wearout
  - ❑Bad for read-intensive workloads (few erases really needed)
  - ❑Lifetime degrades for such workloads (see paper)

In-Place Reprogramming Based FCR

- Idea: Periodically reprogram (in-place) each physical page (after correcting errors)
    - Flash programming techniques (ISPP) can correct retention errors in-place by recharging flash cells

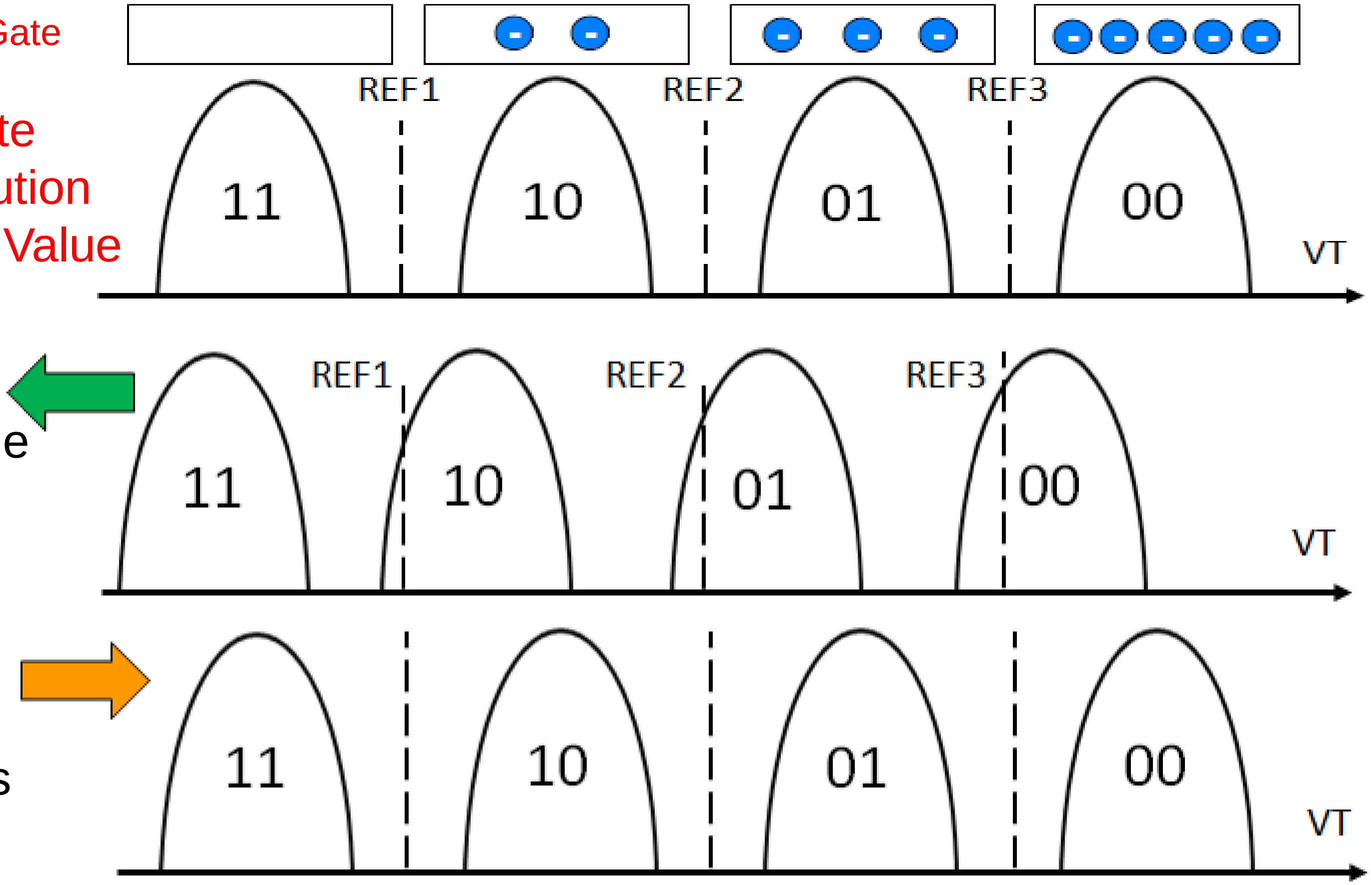


- Problem: Program errors accumulate on the same page → may not be correctable by ECC after some time

In-Place Reprogramming of Flash Cells

Floating Gate

Floating Gate Voltage Distribution for each Stored Value



Retention errors are caused by cell voltage shifting to the left

ISPP moves cell voltage to the right; fixes retention errors

- Pro: No remapping needed → no additional erase operations
- Con: Increases the occurrence of program errors

Program Errors in Flash Memory

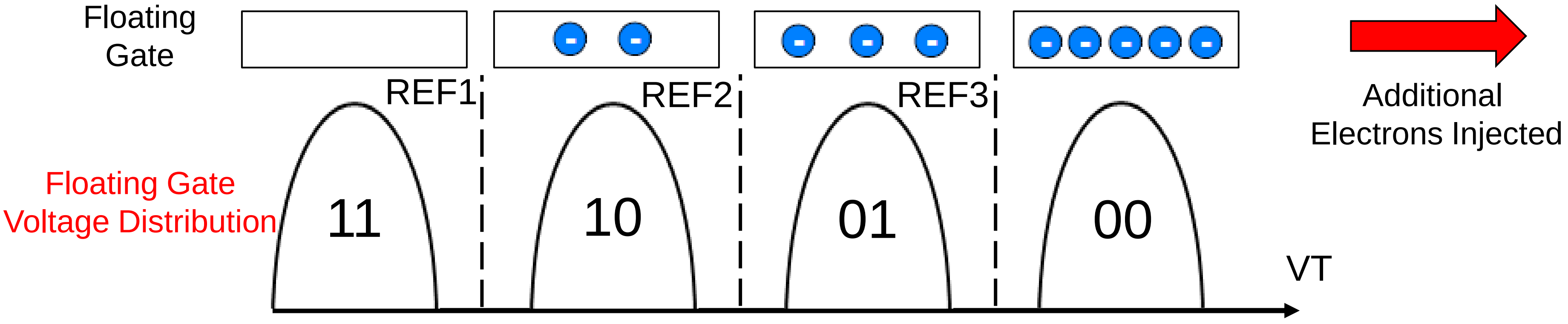
- When a cell is being programmed, voltage level of a neighboring cell changes (unintentionally) due to parasitic capacitance coupling

  → can change the data value stored

- Also called program interference error

- Program interference causes neighboring cell voltage to shift to the right

Problem with In-Place Reprogramming


Floating Gate
REF1
REF2
REF3
Additional Electrons Injected
Floating Gate Voltage Distribution
11
10
01
00
VT

Original data to be programmed	…	00	11	01	00	10	11	00	…
Program errors after initial programming	…	00	10	01	00	10	11	00	…
Retention errors after some time	…	01	10	10	00	11	11	01	…
Errors after in-place reprogramming	…	00	10	01	00	10	10	00	…

1. Read data
2. Correct errors
3. Reprogram back

Problem: Program errors can accumulate over time

Hybrid Reprogramming/Remapping Based FCR

- Idea:
    - Monitor the count of right-shift errors (after error correction)
    - If count < threshold, in-place reprogram the page
    - Else, remap the page to a new page

- Observation:
    - Program errors much less frequent than retention errors → Remapping happens only infrequently

- Benefit:
    - Hybrid FCR greatly reduces erase operations due to remapping

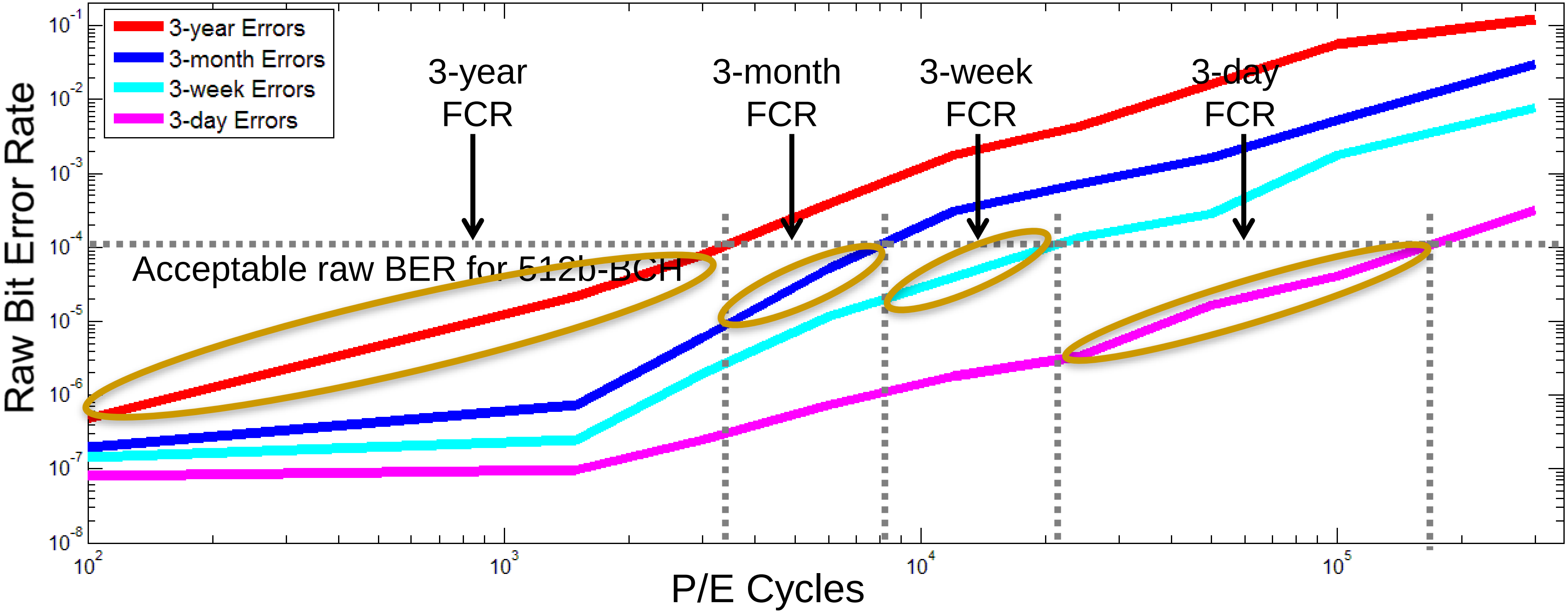
Adaptive-Rate FCR

- Observation:
  - Retention error rate strongly depends on the P/E cycles a flash page endured so far
  - No need to refresh frequently (at all) early in flash lifetime

- Idea:
  - Adapt the refresh rate to the P/E cycles endured by each page
  - Increase refresh rate gradually with increasing P/E cycles

- Benefits:
  - Reduces overhead of refresh operations
  - Can use existing FTL mechanisms that keep track of P/E cycles

Adaptive-Rate FCR (Example)



Select refresh frequency such that error rate is below acceptable rate

FCR: Other Considerations

- Implementation cost
  - ❑No hardware changes
  - ❑FTL software/firmware needs modification
- Response time impact
  - ❑FCR not as frequent as DRAM refresh; low impact
- Adaptation to variations in retention error rate
  - ❑Adapt refresh rate based on, e.g., temperature **[Liu+ ISCA 2012]**
- FCR requires power
  - ❑Enterprise storage systems typically powered on

Evaluation Methodology

- Experimental flash platform to obtain error rates at different P/E cycles [Cai+ DATE 2012]
- Simulation framework to obtain P/E cycles of real workloads: DiskSim with SSD extensions
- Simulated system: 256GB flash, 4 channels, 8 chips/channel, 8K blocks/chip, 128 pages/block, 8KB pages
- Workloads
  - File system applications, databases, web search
  - Categories: Write-heavy, read-heavy, balanced
- Evaluation metrics
  - Lifetime (extrapolated)
  - Energy overhead, P/E cycle overhead

Extrapolated Lifetime

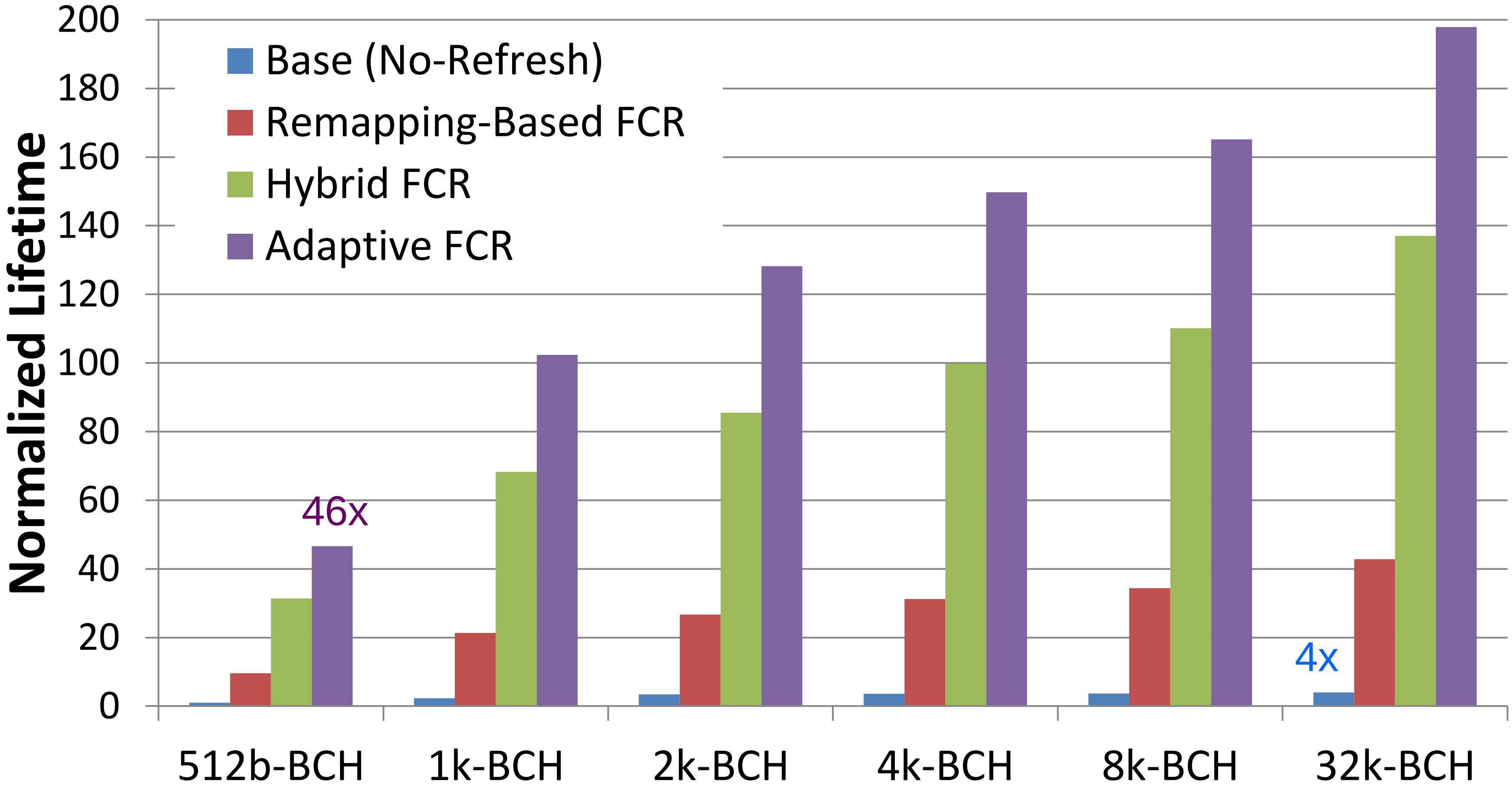
Obtained from Experimental Platform Data

$$\frac{\text{Maximum full disk P/E Cycles for a Technique}}{\text{Total full disk P/E Cycles for a Workload}} \times \text{\# of Days of Given Application}$$

Obtained from Workload Simulation

Real length (in time) of each workload trace

Normalized Flash Memory Lifetime



Lifetime of FCR much higher than lifetime of stronger ECC

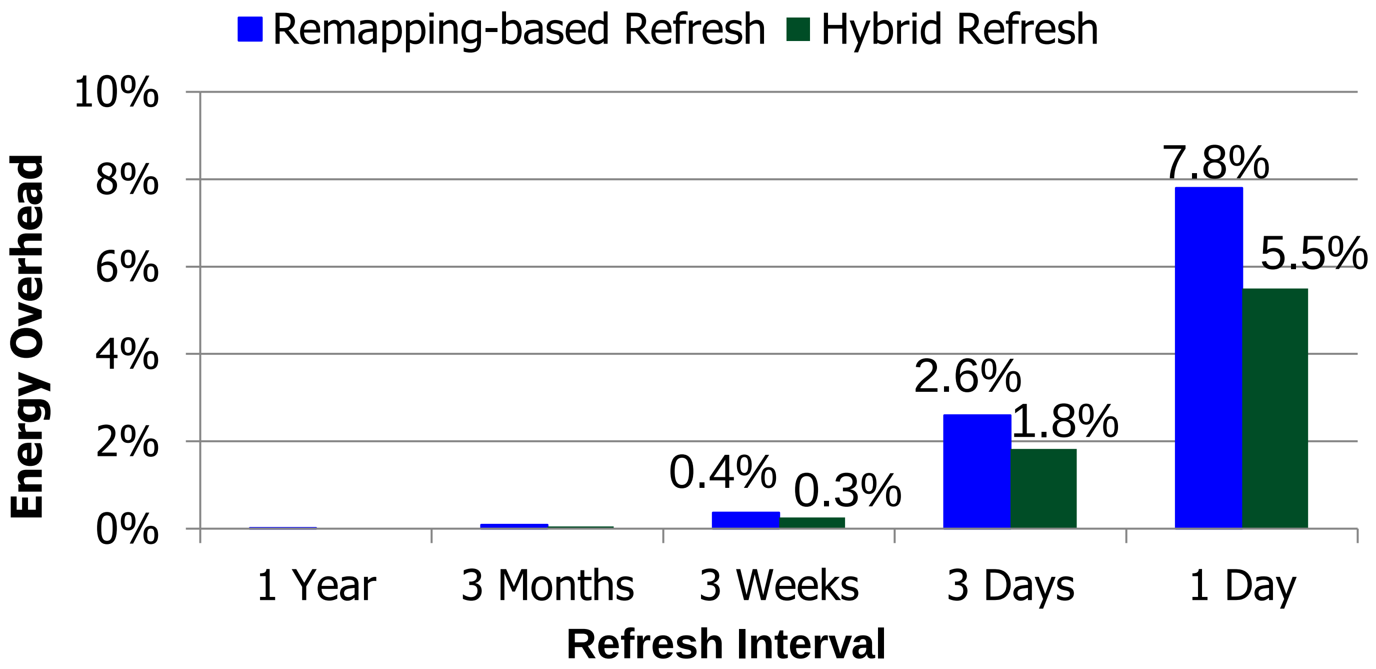
Lifetime Evaluation Takeaways

- Significant average lifetime improvement over no refresh
  - Adaptive-rate FCR: 46X
  - Hybrid reprogramming/remapping based FCR: 31X
  - Remapping based FCR: 9X

- FCR lifetime improvement larger than that of stronger ECC
  - 46X vs. 4X with 32-kbit ECC (over 512-bit ECC)
  - FCR is less complex and less costly than stronger ECC

- Lifetime on all workloads improves with Hybrid FCR
  - Remapping based FCR can degrade lifetime on read-heavy WL
  - Lifetime improvement highest in write-heavy workloads

Energy Overhead



- Adaptive-rate refresh: <1.8% energy increase until daily refresh is triggered

Overhead of Additional Erases

- Additional erases happen due to remapping of pages

- Low (2%-20%) for write intensive workloads
- High (up to 10X) for read-intensive workloads

- Improved P/E cycle lifetime of all workloads largely outweighs the additional P/E cycles due to remapping

More Results in the Paper

- Detailed workload analysis

- Effect of refresh rate