March 27 – 31, 2023
Tallinn, Estonia

The 38th ACM/SIGAPP Symposium
On Applied Computing, SAC 2023

Efficient Sanitization Design for LSM-based Key-Value Store over 3D NAND Flash

Liang-Chi Chen[1], Shu-Qi Yu[2], Chien-Chung Ho[1],
Wei-Chen Wang[3], Yung-Chun Li[4]

[1] Computer Science and Information Engineering
National Cheng Kung University, Taiwan

[2] Computer Science and Information Engineering
National Taiwan University , Taiwan

[3] Electrical Engineering and Computer Science
Massachusetts Institute of Technology, USA

[4] Emerging System Lab
Macronix International Co., Ltd., Taiwan

Outline

- ➢Motivation
- ➢Background
- ➢Observation
- ➢Design
- ➢Experiment
- ➢Conclusion and Future Work

Outline

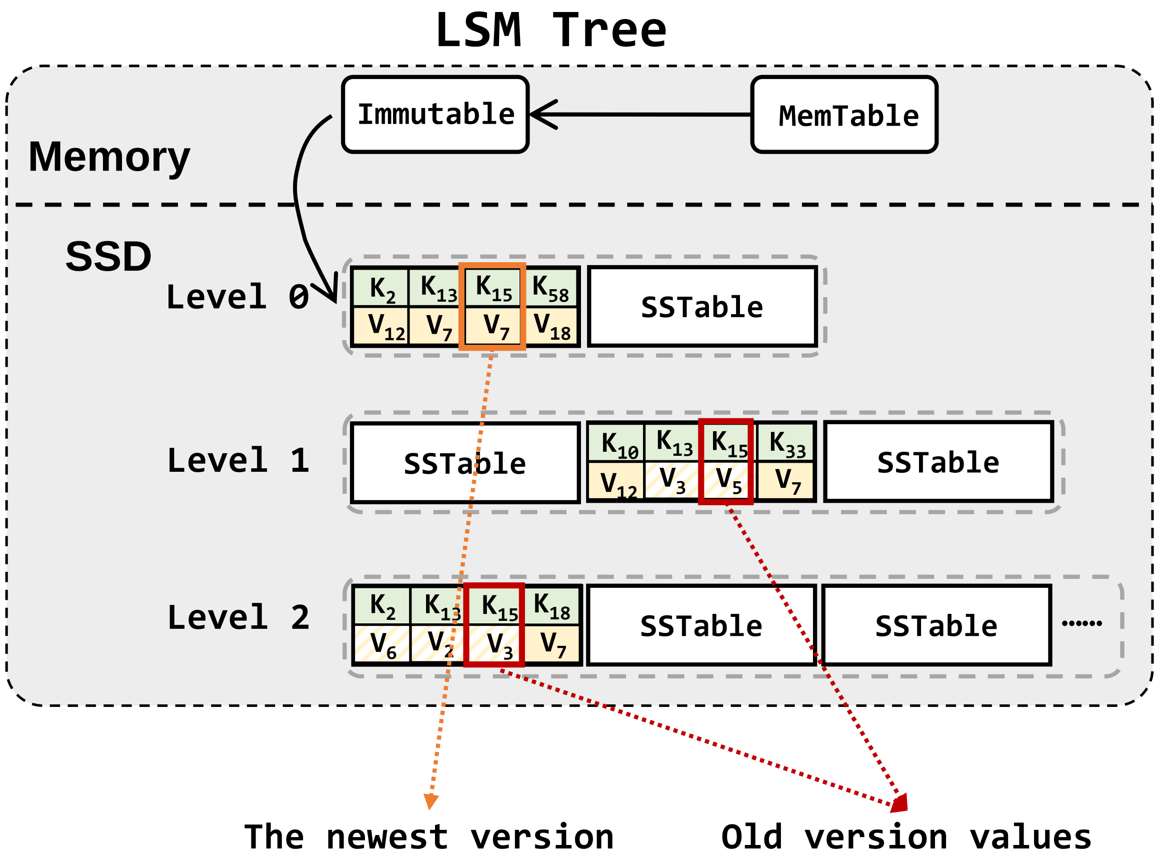
- Motivation
- Background
- Observation
- Design
- Experiment
- Conclusion and Future Work

Outdated data in LSM Tree

- Log-Structured Merge-Tree (***LSM-tree***) are widely adopted to modern ***data-intensive*** applications
  - RocksDB
  - LevelDB
- LSM-tree leaves several out-of-date values to the specified key
  - Those values are removed during compaction

RocksDB

LEVELDB



LSM Tree
Immutable
MemTable
Memory
SSD
Level 0
K_{2} K_{13} K_{15} K_{58}
V_{12} V_{7} V_{7} V_{18}
SSTable
Level 1
SSTable
K_{10} K_{13} K_{15} K_{33}
V_{12} V_{3} V_{5} V_{7}
SSTable
Level 2
K_{2} K_{13} K_{15} K_{18}
V_{6} V_{2} V_{3} V_{7}
SSTable
SSTable
......
The newest version
Old version values

There is a lot of outdated data in the LSM tree !!

Privacy Security on NAND Flash

- ***3D NAND flash-based devices*** are widely used in many situations

People store their personal data in these devices, e.g., image, video, and password

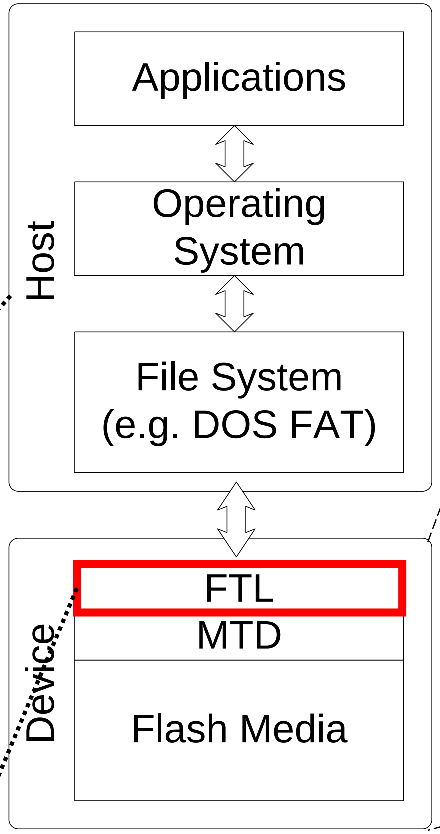
USB flash drive

Memory card

SSD

Privacy Security on NAND Flash

- A **deletion** or **updating** request is sent from host to storage
    - NAND flash only performs ***logical deletion***
    - Out-of-date data ***isn't removed physically*** from device-level immediately
- **The data which user want to delete is still on the device !**



SD,MemoryStick CompactFlash

Delete an image

Logical deletion

After scanning raw data:
Image is still on device

Privacy issue !!

Outline

- Motivation
- Background
- Observation
- Design
- Experiment
- Conclusion and Future Work

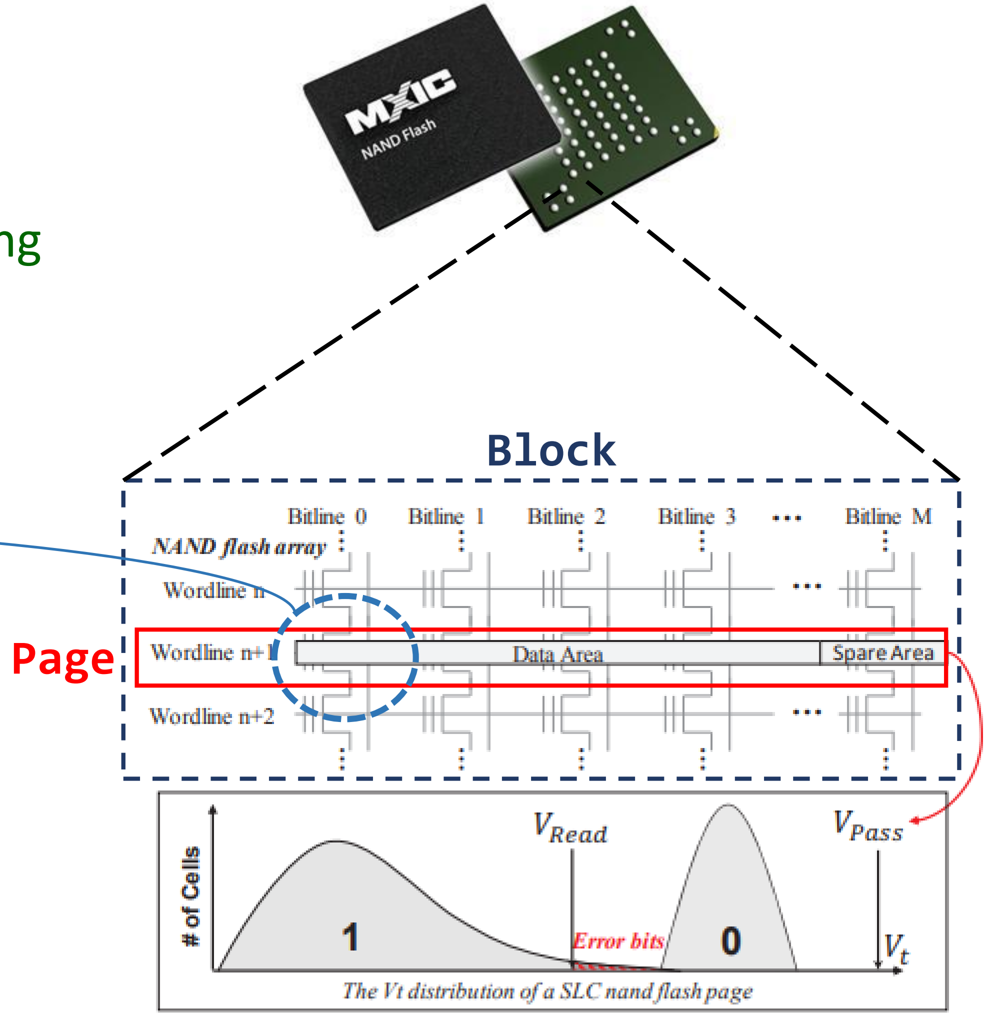
Basics of NAND Flash

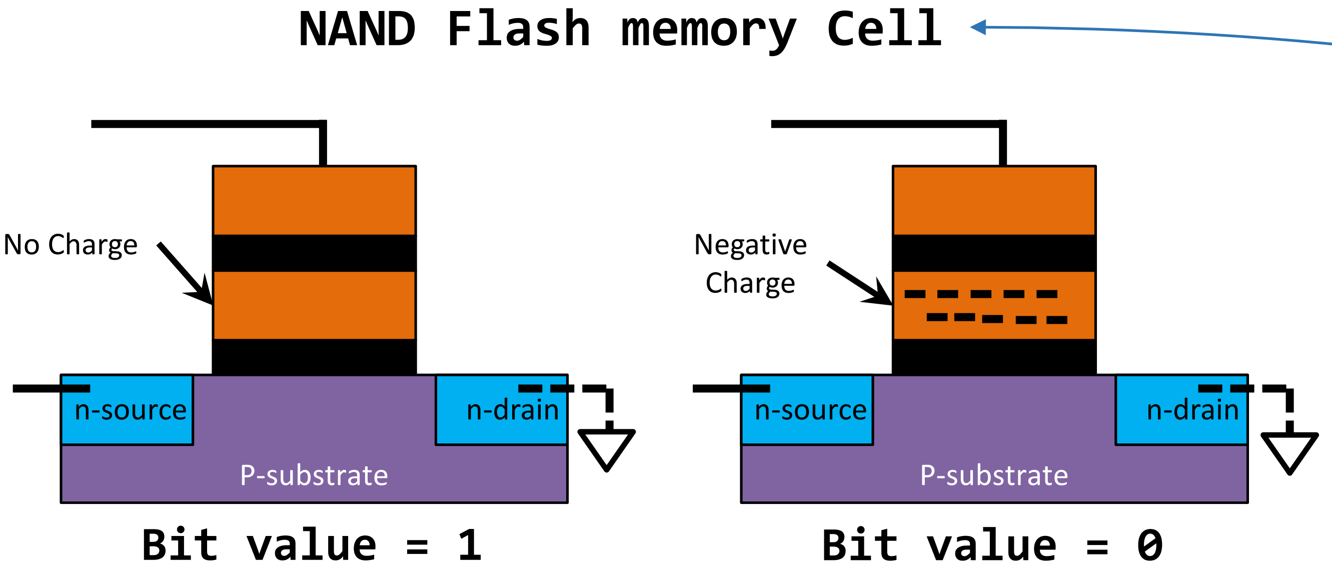
Basic operations

1. Program (write) : page
2. Read : page
3. Erase : block

Features:

1. Write-once
2. Bulk-erasing
   → Out-of-place updating
   → Garbage collection





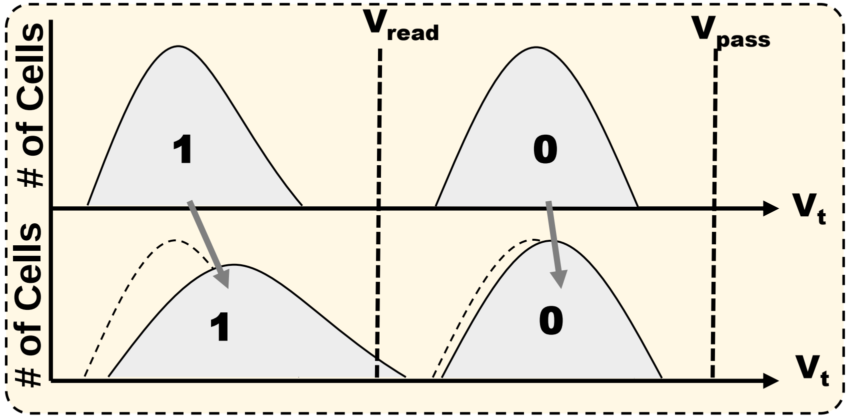
Program Disturbance

- Program Disturbance
  - When enough electrons are forced into a neighboring cell during a program, that cell will appear as ***weakly programmed in the next page*** of that cell's page

Block #0

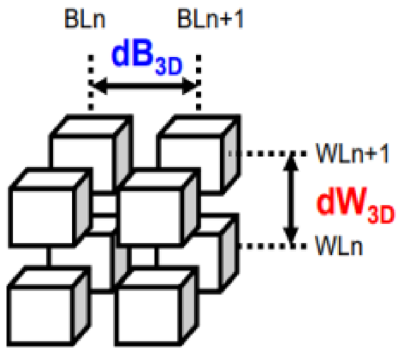
WL 0	Disturbed
WL 1	Programmed
WL 2	Disturbed
WL 3	

2D NAND flash

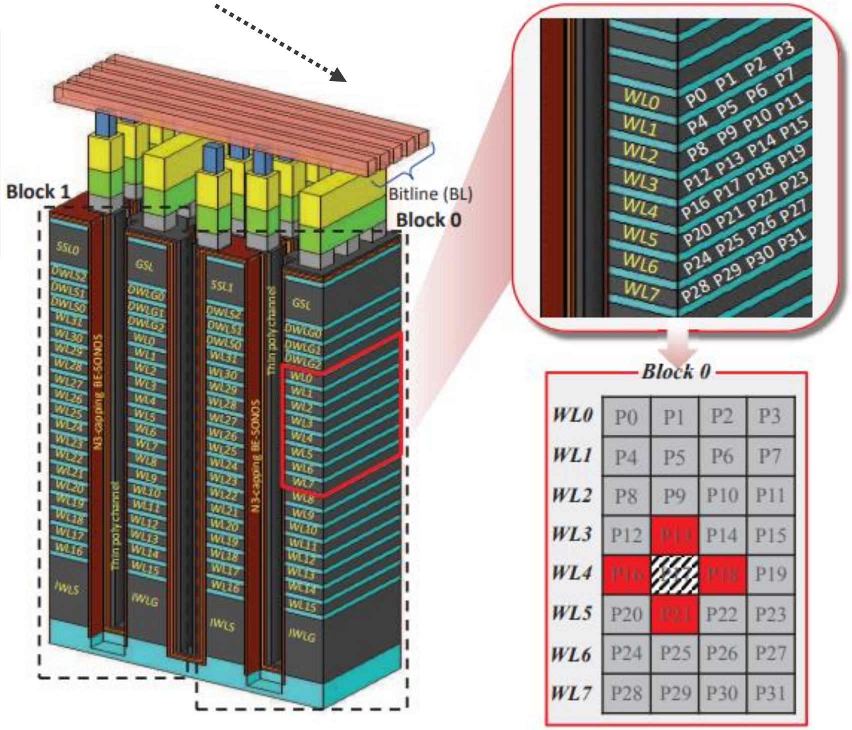


3D flash memory
- Reduce the bit cost ☺
- *Introduce a new sort of disturbance from the vertical direction* ☹
  - x , y , z disturb directions

3D Structure



3D NAND flash



Block 0

WL0	P0	P1	P2	P3
WL1	P4	P5	P6	P7
WL2	P8	P9	P10	P11
WL3	P12	P13	P14	P15
WL4	P16	▨	P18	P19
WL5	P20	P21	P22	P23
WL6	P24	P25	P26	P27
WL7	P28	P29	P30	P31

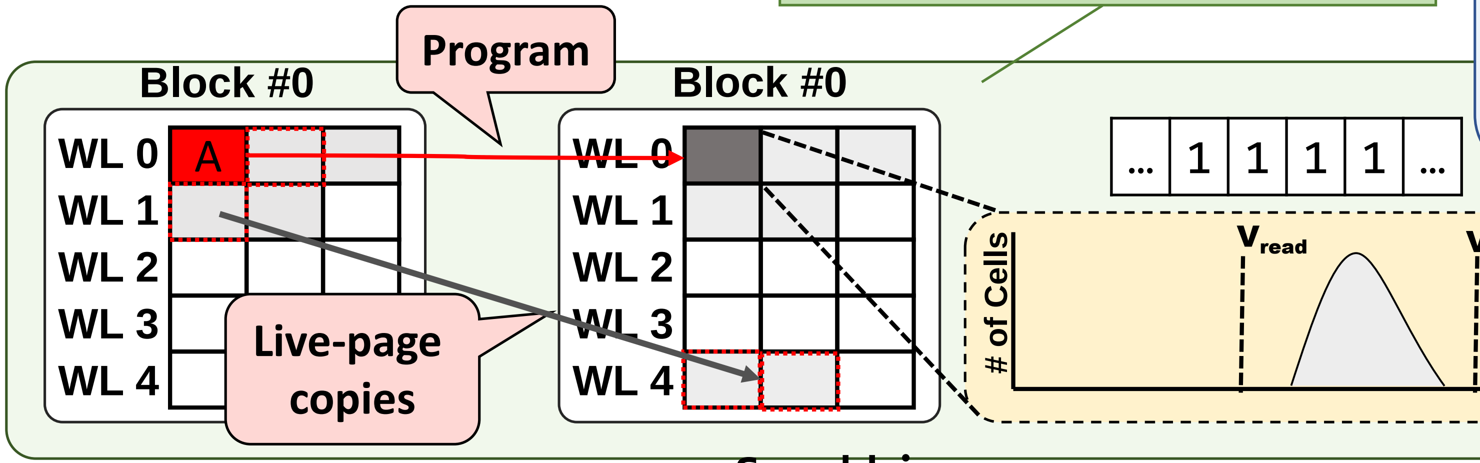
Source : http://cushychicken.github.io/posts/nand-pt5-how-nand-breaks

W. -C. Wang, P. -H. Lin, Y. -C. Li, C. -C. Ho, Y. -M. Chang and Y. -H. Chang, "Toward Instantaneous Sanitization through Disturbance-induced Errors and Recycling Programming over 3D Flash Memory," 2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2019, pp. 1-8.

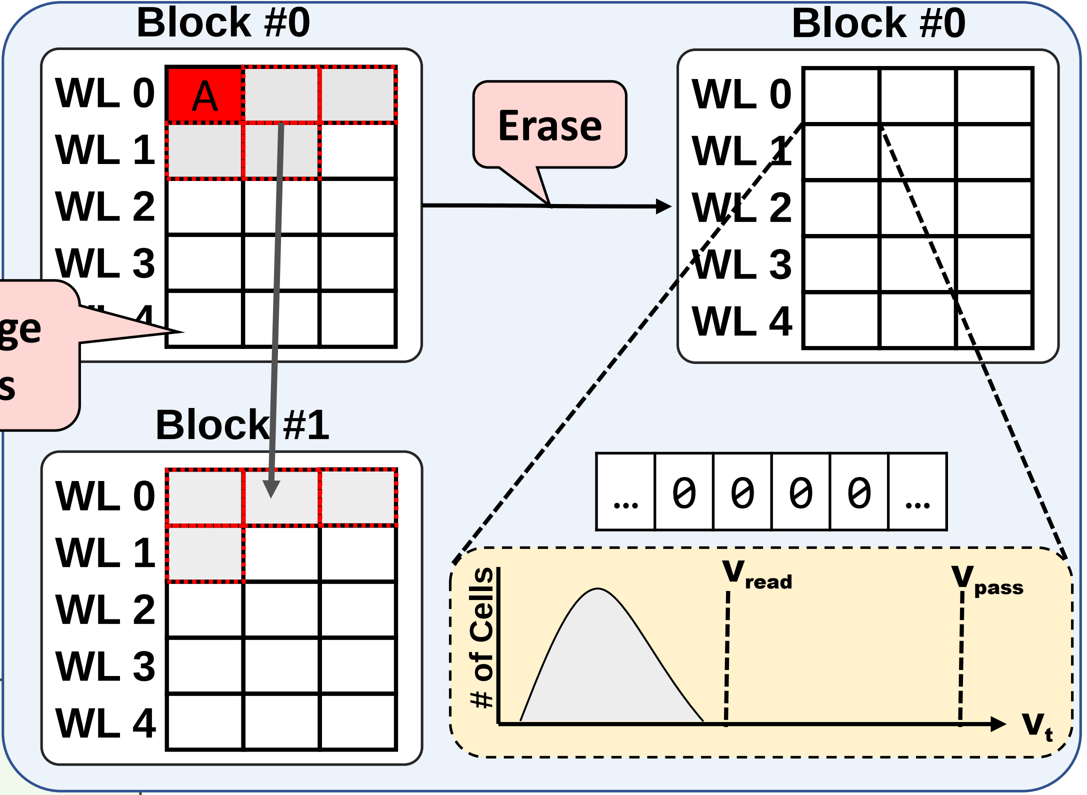
Conventional Sanitization Methods

- ***Erasure-based*** method
  - Issue “**block-erase**” on the block where the victim page resides
- ***Rewriting-based*** method (scrubbing)
  - **Re-program** the victim page
- ***Encryption-based***
  - Additional key **encoding**

- Write amplification
- Additional overhead



Scrubbing

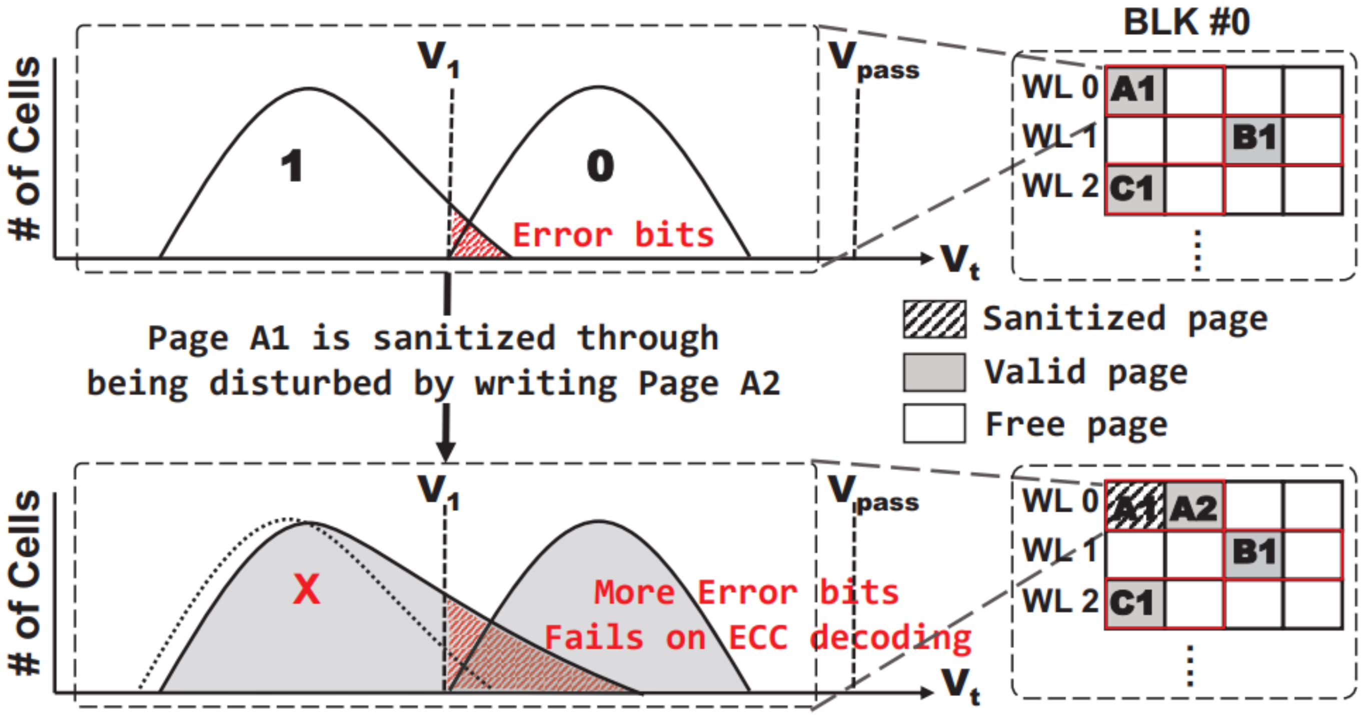


Erasure-based method

- Long latency
- Write amplification
- Endurance

Instantaneous Sanitization

- To delete an outdated data by a write of new version data ***Instantaneously***
- Deliberately create a ***certain number of errors in beginning stage***
- If there are ***too many disturbance-induced errors*** in a flash page, it fails on ECC decoding
  - Original data can’t be recognized
  - Successfully sanitized



W. -C. Wang, P. -H. Lin, Y. -C. Li, C. -C. Ho, Y. -M. Chang and Y. -H. Chang, "Toward Instantaneous Sanitization through Disturbance-induced Errors and Recycling Programming over 3D Flash Memory," 2019 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2019, pp. 1-8.

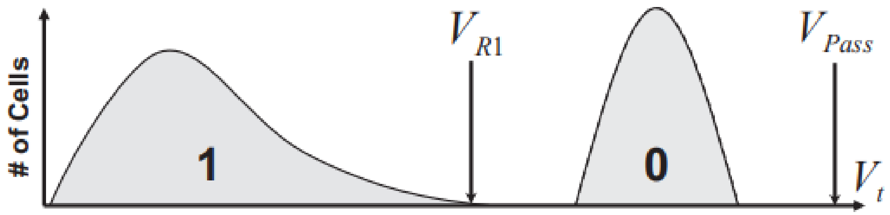
Outline

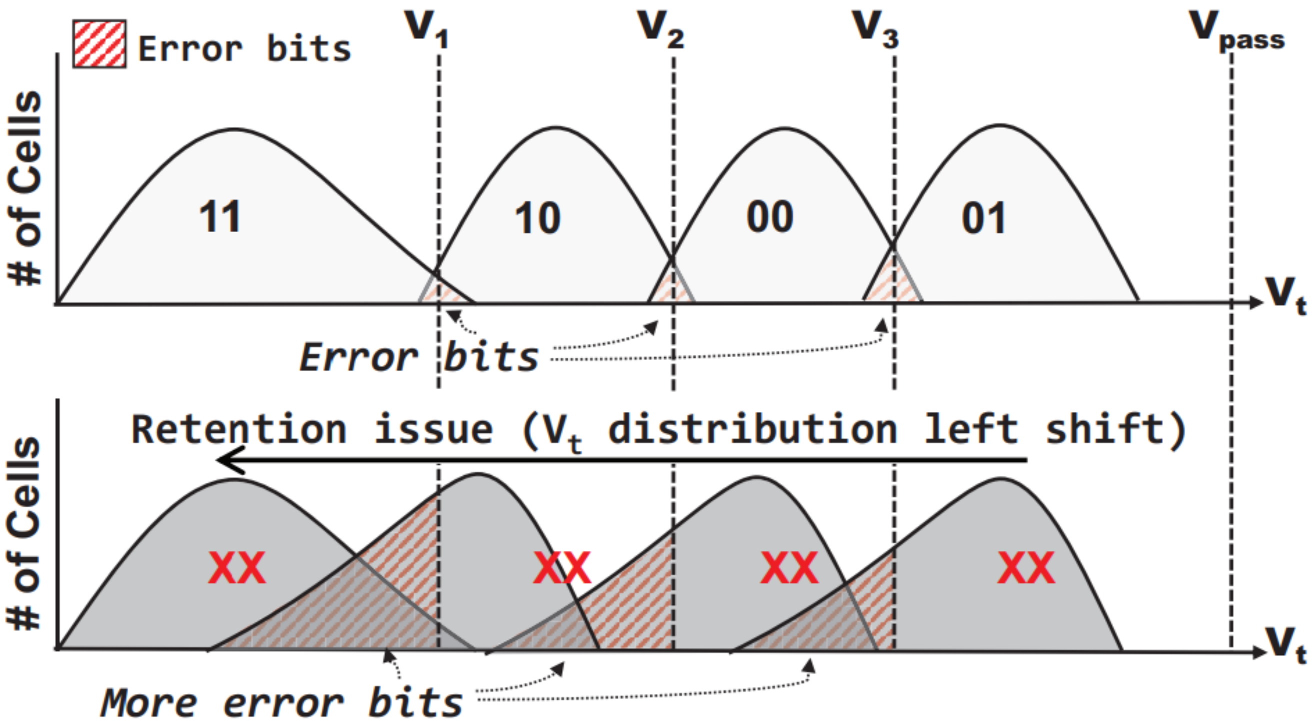
- ➢Motivation
- ➢Background
- ➢**Observation**
- ➢Design
- ➢Experiment
- ➢Conclusion and Future Work

Observation: Vt distribution

- MLC allows each memory cell to store multiple bits of information
  - Higher density
  - Lower cost
  - TLC and QLC are gradually being used
- *Data retention decreases as memory density increases*
  - SLC is better than MLC
- Instantaneous sanitization creates errors deliberately
  - *Causes retention issue*
  - The storage system cannot have poor reliability

- SLC has better reliability
- Wide Vt window per state





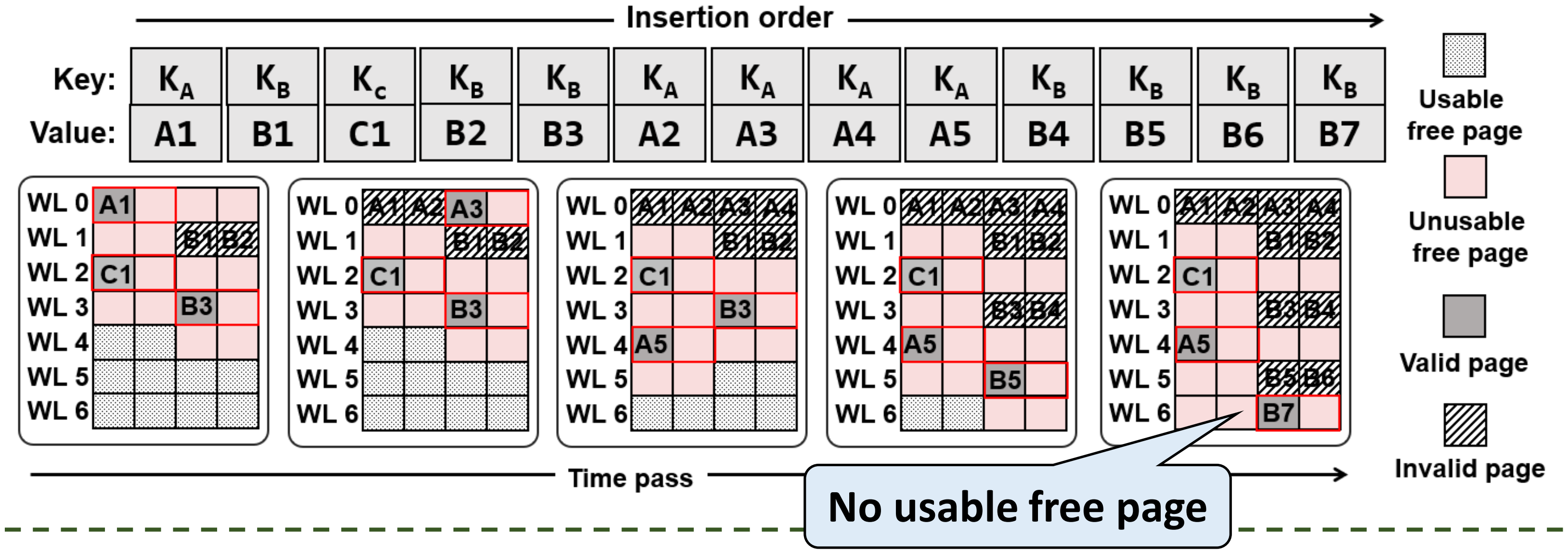
- MLC (or TLLC, QC) has retention issue
- Narrow Vt window per state

Observation: Space Utilization

- The impact of inconsistent data update patterns to space utilization in a block

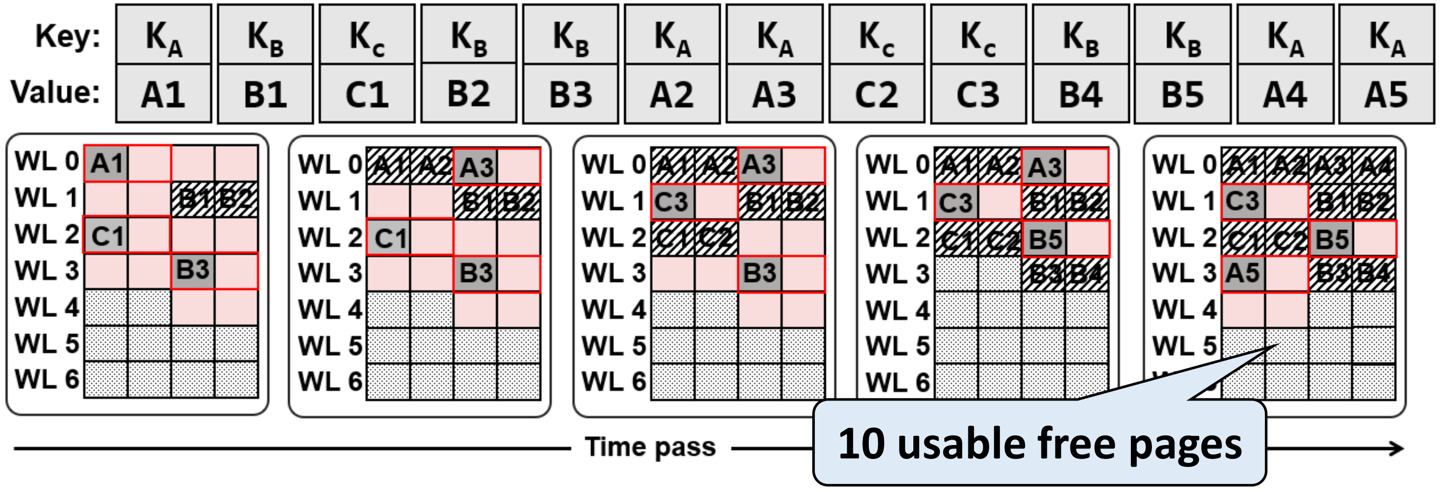
Infrequent data updates result in more wasted space

Regular data updates has optimal space utilization



Key:	K_A	K_B	K_C	K_B	K_B	K_A	K_A	K_A	K_A	K_B	K_B	K_B	K_B
Value:	A1	B1	C1	B2	B3	A2	A3	A4	A5	B4	B5	B6	B7

No usable free page



Key:	K_A	K_B	K_C	K_B	K_B	K_A	K_A	K_C	K_C	K_B	K_B	K_A	K_A
Value:	A1	B1	C1	B2	B3	A2	A3	C2	C3	B4	B5	A4	A5

10 usable free pages

Observation Summary

- ***Vt distribution***

☹
- MLC has retention issue
- Narrow Vt window per state

→ How to re-design the MLC Vt distribution ?

- ***Space Utilization***

☹ Infrequent data updates result in more wasted space

→ How to maintain the space utilization ?

Outline

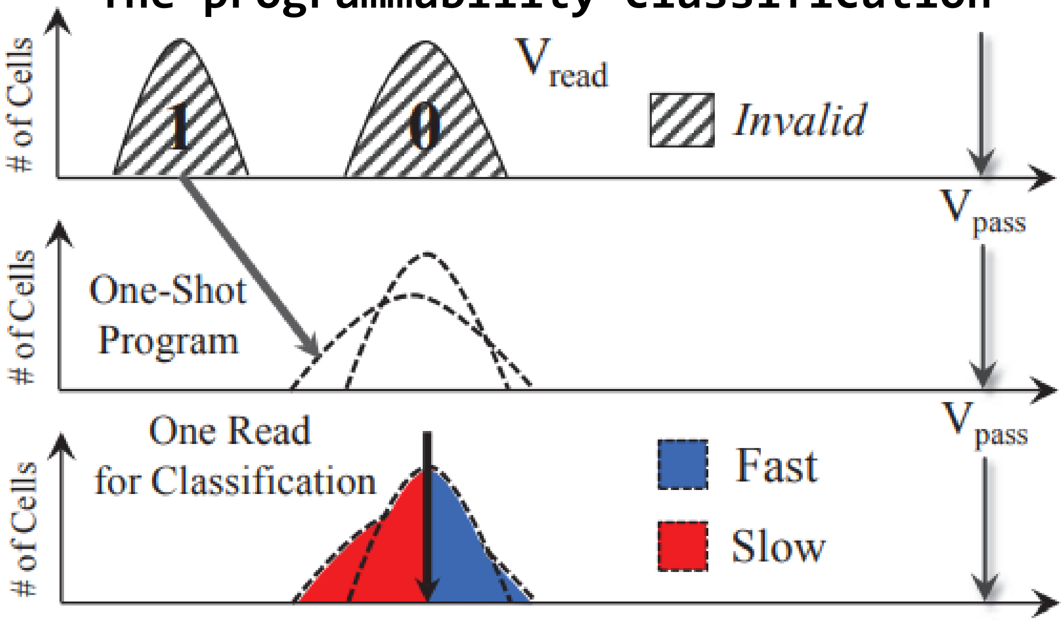
- ➢Motivation
- ➢Background
- ➢Observation
- ➢**Design**
- ➢Experiment
- ➢Conclusion and Future Work

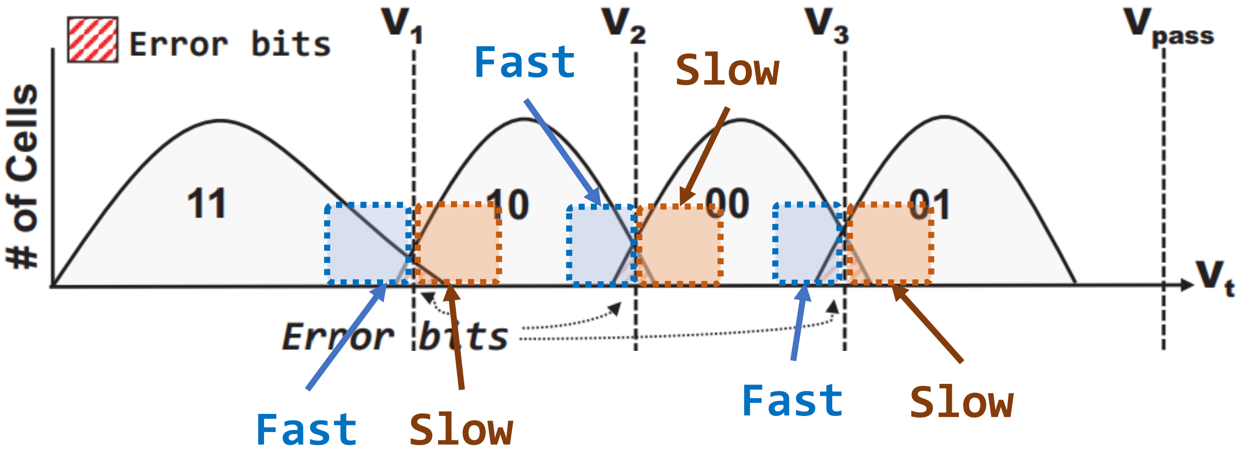
Influence-Classification Programming

- Influence-classification programming (ICP)
  - By considering cells' **programmability** and error properties
    - ***Fast cells: susceptible to interference and charge leakage***
    - ***Slow cells: easily to keep in the current Vt state as long as possible***

- ***Fast cells help sanitization***
  - More errors to fail ECC
- ***Slow cells help improving reliability***
  - No charge loss

The programmability classification



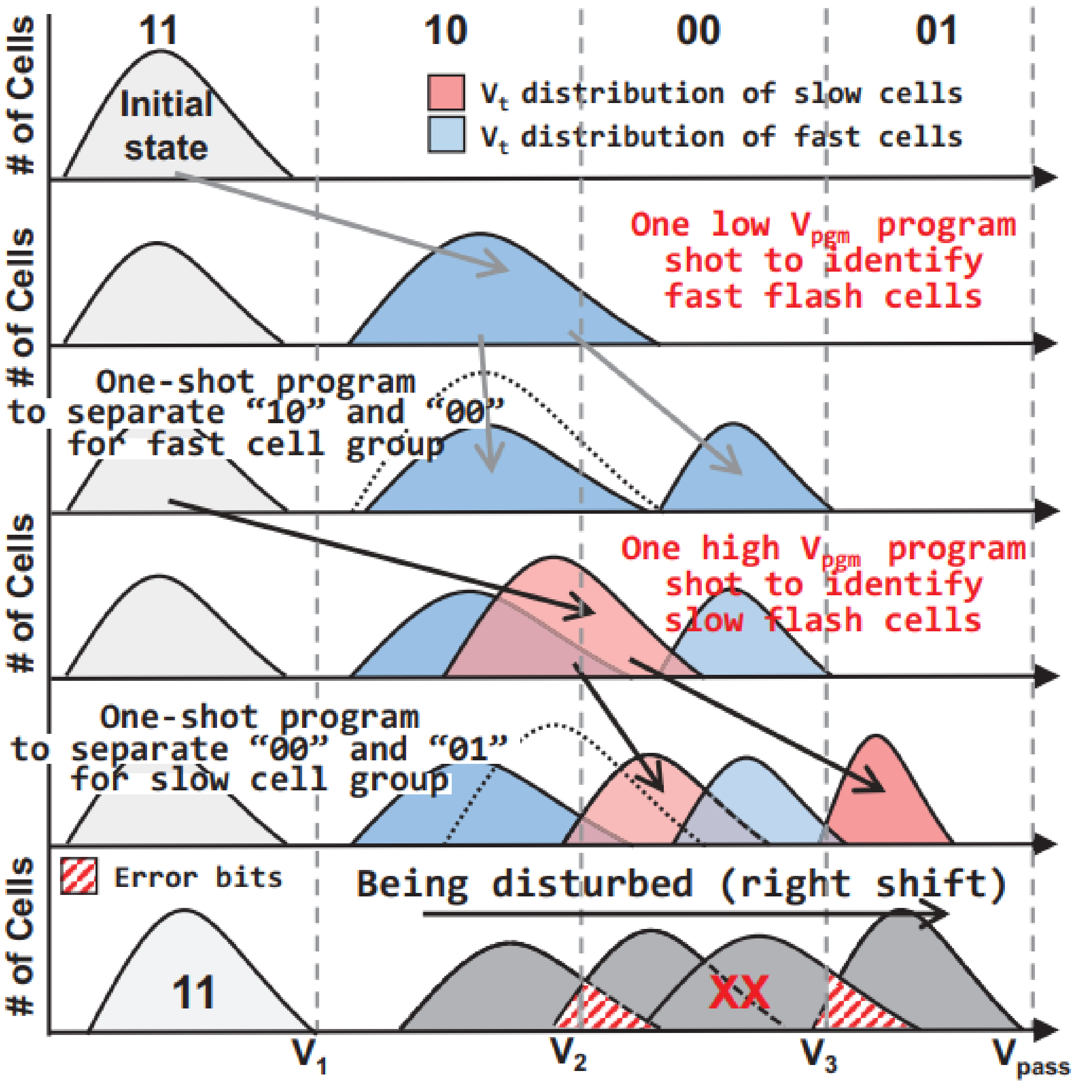


Ho, Chien-Chung, et al. "Achieving defect-free multilevel 3D flash memories with one-shot program design." 2018 55th ACM/ESDA/IEEE Design Automation Conference (DAC). IEEE, 2018.

Influence-Classification Programming

- ***Influence-classification programming (ICP)***
  1. One-shot programming to classify programmability
  2. Another one-shot programming to charge the cells to certain voltage
  3. Redo 1-2 until the MLC page programming is complete

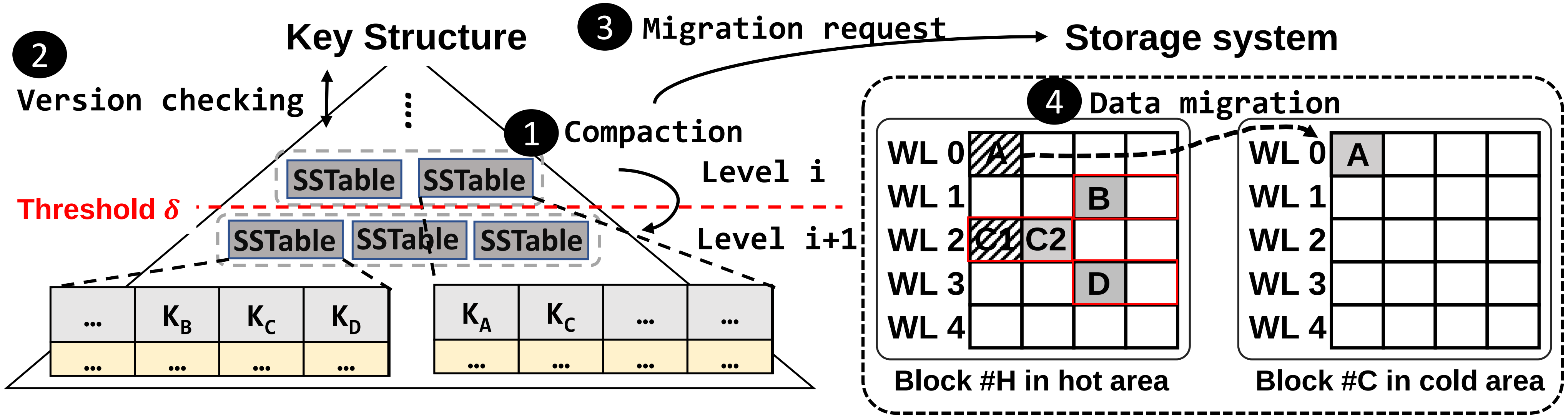
ICP can realize the same effect of instantaneous sanitization on 3D MLC NAND flash



11
10
00
01
Initial state
V_t distribution of slow cells
V_t distribution of fast cells
# of Cells
One low V_{pgm} program shot to identify fast flash cells
One-shot program to separate “10” and “00” for fast cell group
One high V_{pgm} program shot to identify slow flash cells
One-shot program to separate “00” and “01” for slow cell group
Error bits
Being disturbed (right shift)
11
XX
V_1
V_2
V_3
V_{pass}

Hot-Cold Separation Allocation Policy

- We exploit the inherent characteristics of ***LSM-based*** key-value store
  - Data locates in only ***deep layers*** means that it is ***cold***



- Only hot data leaves many out-of-date data that needs efficient sanitization
- Cold data usually does not change versions frequently

Outline

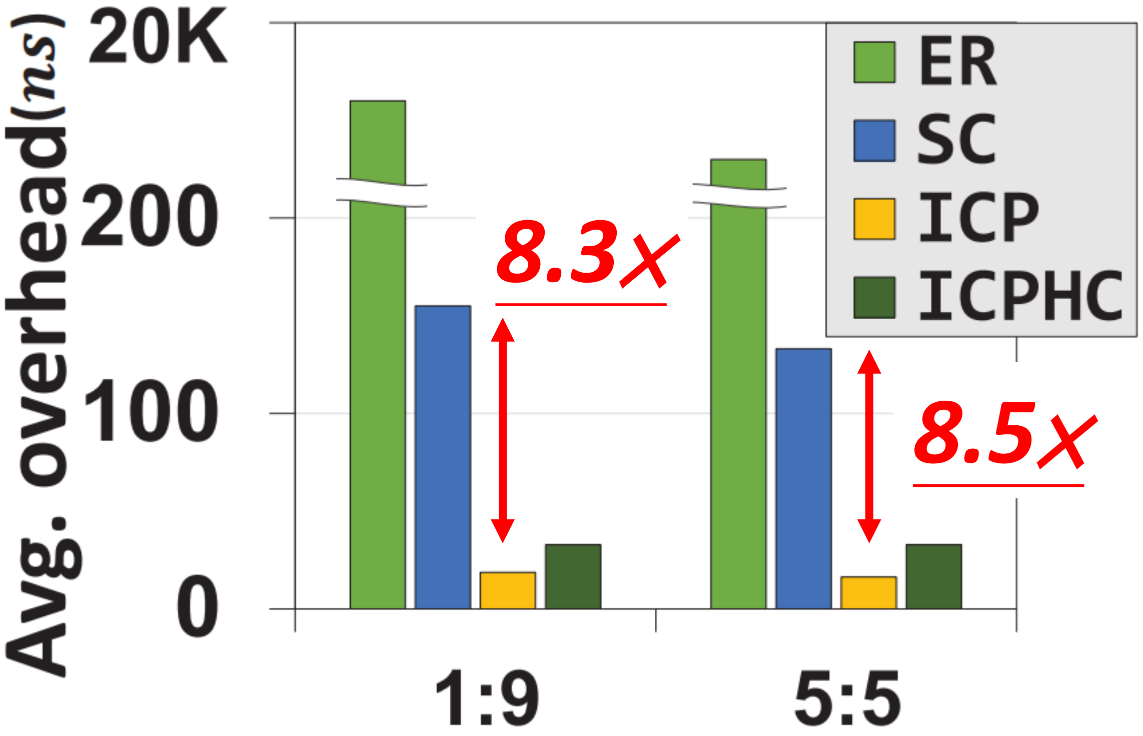
- Motivation
- Background
- Observation
- Design
- **Experiment**
- Conclusion and Future Work

Experiment Setup

- The proposed ICP is compared to two existing methods
  - **Erasure-based method (ER)**
    - *Overhead : block-erase + copy all live pages in the block*
  - **Scrubbing, or called rewriting-based method (SC)**
    - *Overhead : program (write) + copy the neighboring pages*
- We use different hot-cold ratio workloads to test feasibility on MQsim
  - **1 : 9 (close to real scenario)**
  - **5 : 5 (update very frequently)**
- Metrics
  1. Sanitization overhead
  2. Overall I/O performance
  3. Space utilization analysis

Experiment: Sanitization Performance

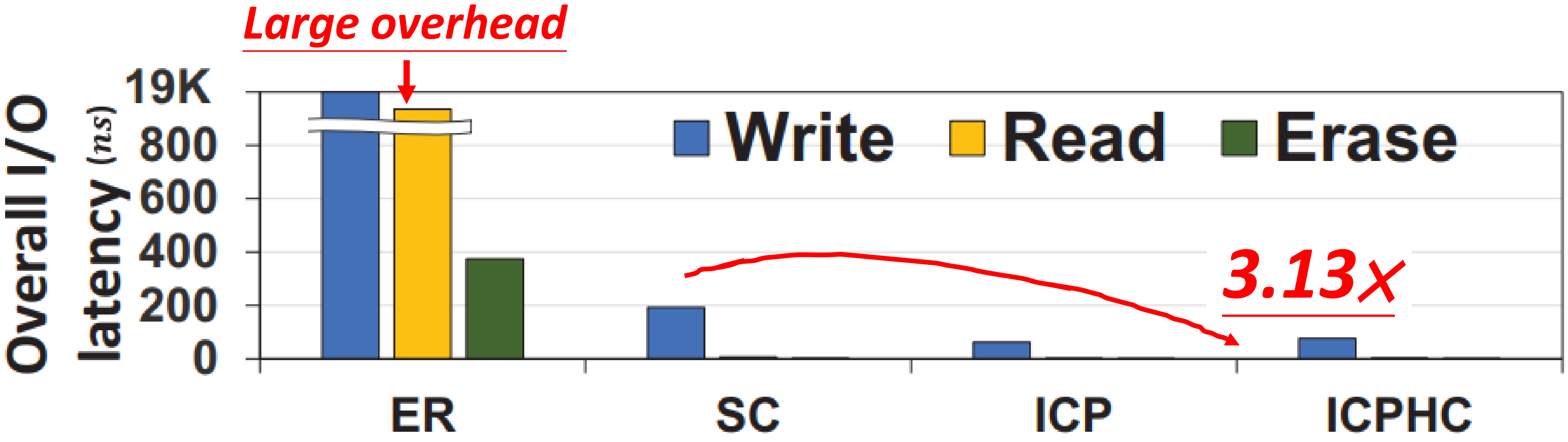
- The sanitization performance of the ***erasure-based method*** is poor in every situations
- Compared to the ***scrubbing***
  - ICP improves the performance by *8.3x* and *8.2x* under the hot-cold ratio of 1:9 and 5:5



Proposed ICP sanitizes data by exploiting programming disturbance and would not incur any additional process

Experiment: Overall I/O Performance

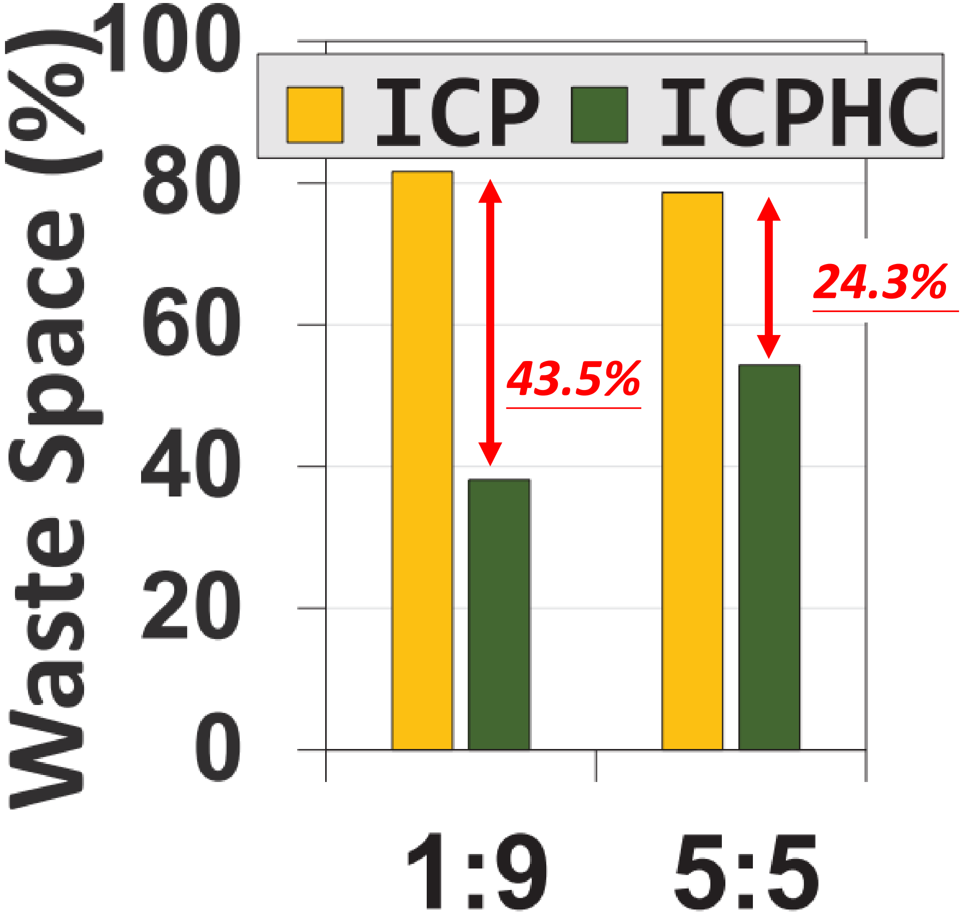
- The trend of overall performance is consistent with that of sanitization performance
  - Performance of the ***erasure-based method*** is poor
  - ***ICP outperforms scrubbing***



Experiment: Space Utilization

- ***Proposed ICP with the hot-cold separation allocation policy***
  - Significantly reduce the wasted space by ***43.5%*** and ***24.3%*** respectively

Proposed hot-cold separation allocation policy **improves space utilization without large overhead**



Outline

- ➢Motivation
- ➢Background
- ➢Observation
- ➢Design
- ➢Experiment
- ➢**Conclusion and Future Work**

Conclusion and Future work

- This paper aims at resolving the ***data security for LSM-based key-value separating storage*** on 3D MLC NAND flash memories
- The proposed ***ICP*** is used to write the new version value and ***sanitize out-of-date values simultaneously***
- The ***hot-cold separation allocation policy*** focuses on ***improving space utilization***
- We will focus on ***device-level observations*** in the future, our idea may incur additional issues on flash with higher density (e.g., TLC or QLC)

Thanks for your attention

Question & Answer