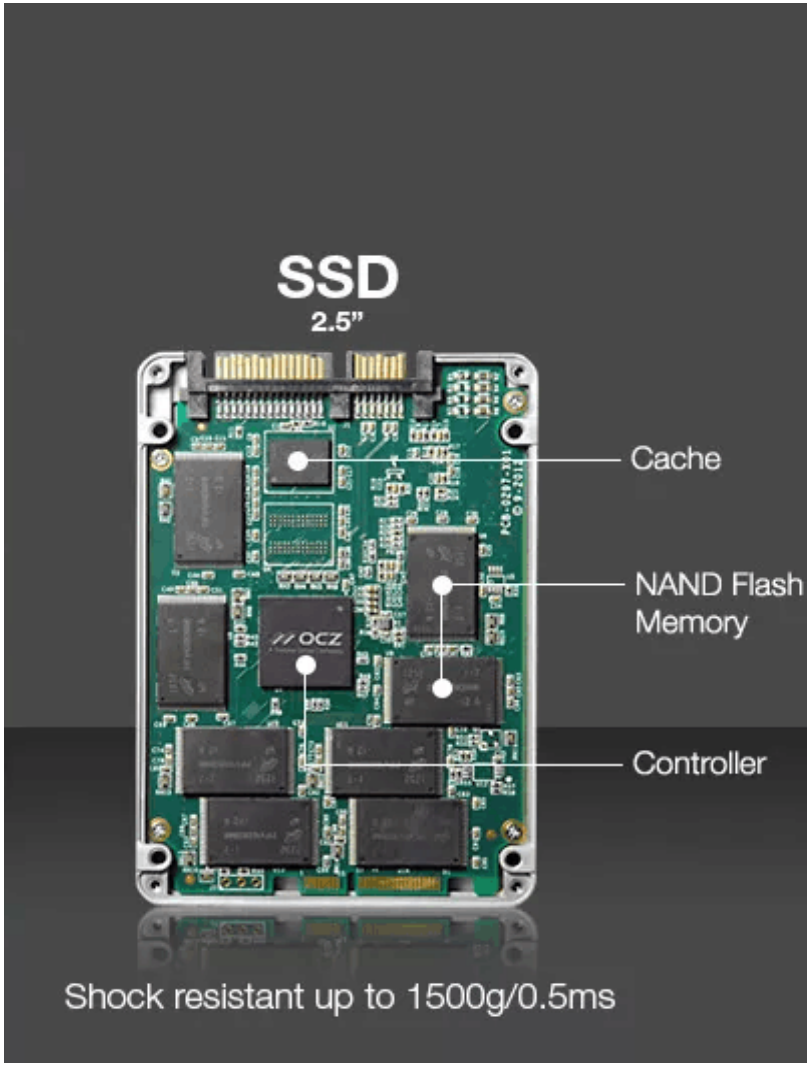


Paper Presentation
on
Are Superpages Super-fast? Distilling Flash Blocks to Unify Flash Pages of a Superpage in an SSD

Authors: Shih-Hung Tseng , Tseng-Yi Chen , and Ming-Chang Yang

Group - 16

CSCE 5610.001 - Computer System Architecture

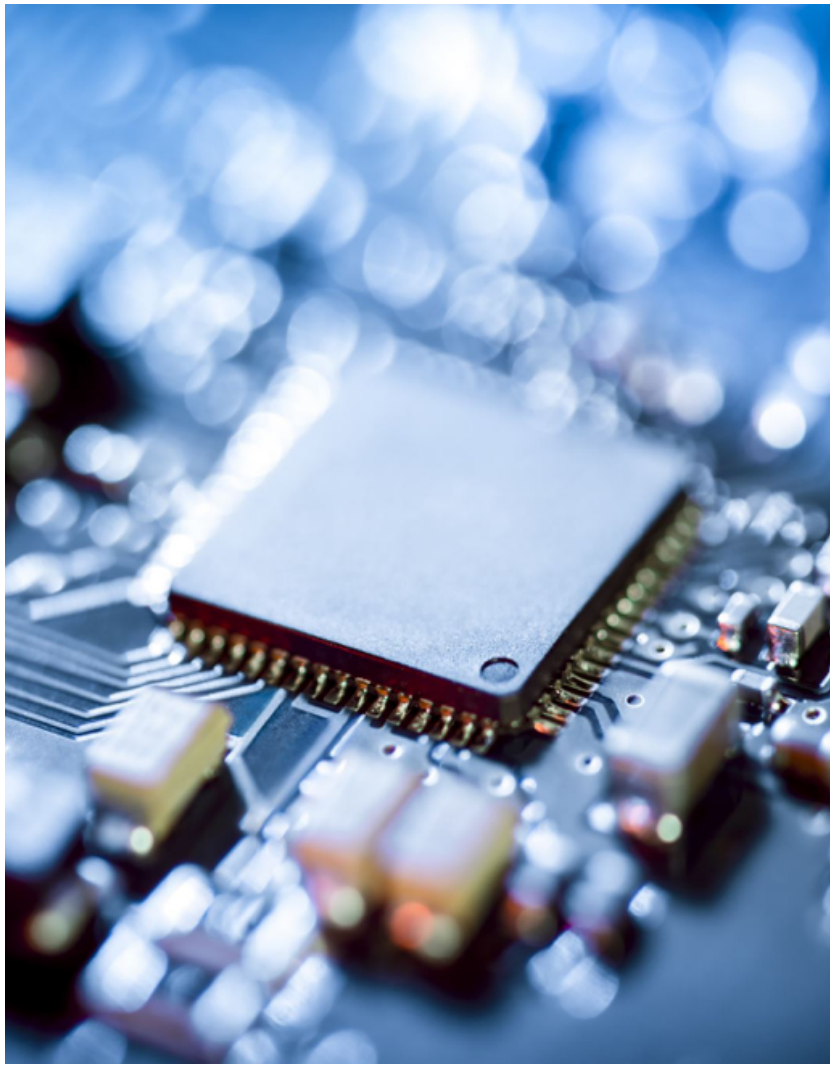
Team Members:

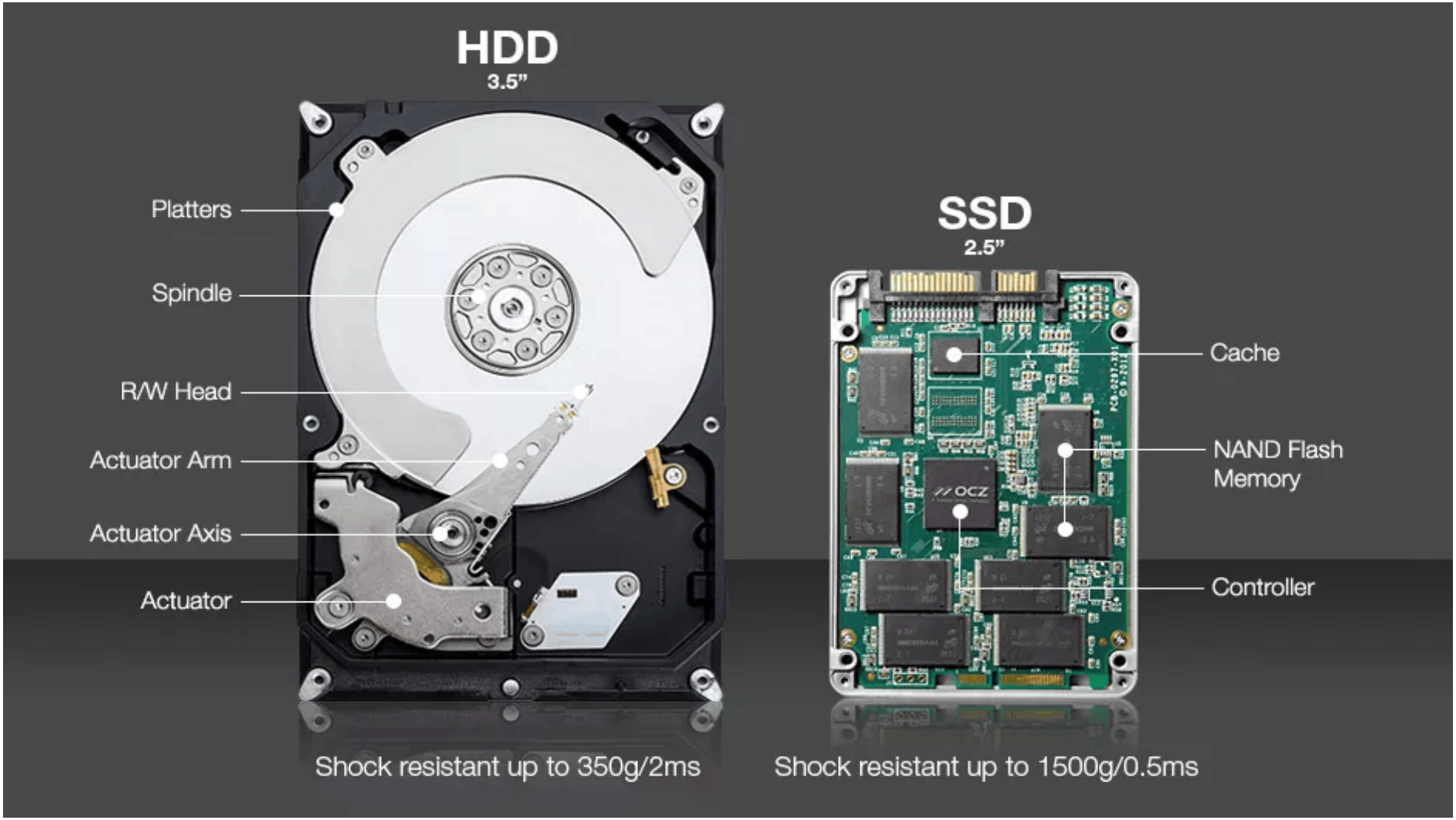
Uday Bhaskar Valapadasu 11696364
Tulasi Sai Pechetti 11663410
Jayakrishna Amathi 11697680

Presentation
Date : 10/17/2024
Time : 12:30pm-12:50pm

Agenda

- **Introduction**
  - **SSD & Its Architecture**
  - **3D NAND Flash Memory**
  - **3D SSD Internal Parallelism**
- **Problem Statement**
- **The Challenge: Process Variation**
- **Process Variation and Similarity in Flash Memory**
- **Objective & Motivation**
- **Characteristics & Findings**
- **Optimization Strategies**
- **Proposed Scheme for QSTR-MED**
- **Perform Evalutation**
- **Comparison with Traditional Methods**
- **Conclusion**
- **Future Work**
- **References**



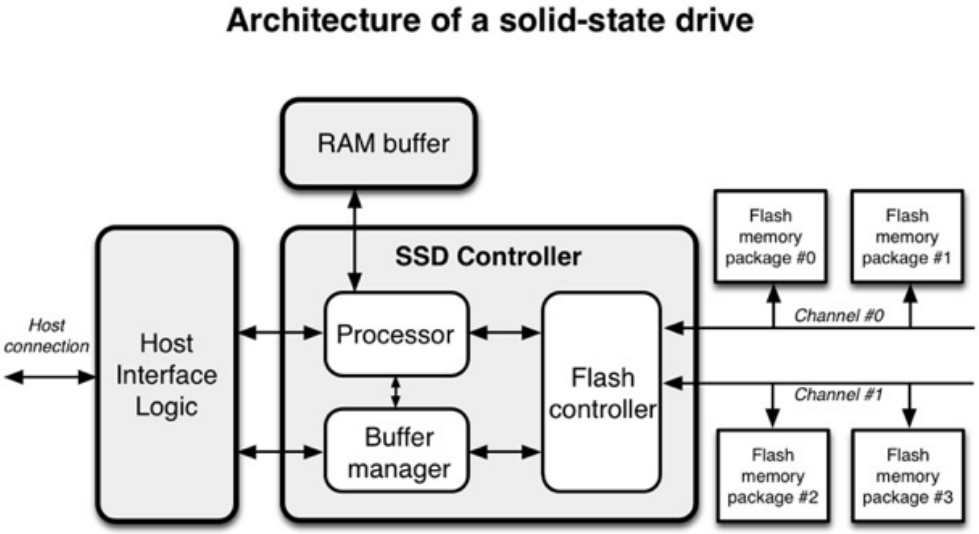

HDD
3.5”
Platters
Spindle
R/W Head
Actuator Arm
Actuator Axis
Actuator
Shock resistant up to 350g/2ms
SSD
2.5”
Cache
NAND Flash Memory
Controller
OCZ
Shock resistant up to 1500g/0.5ms

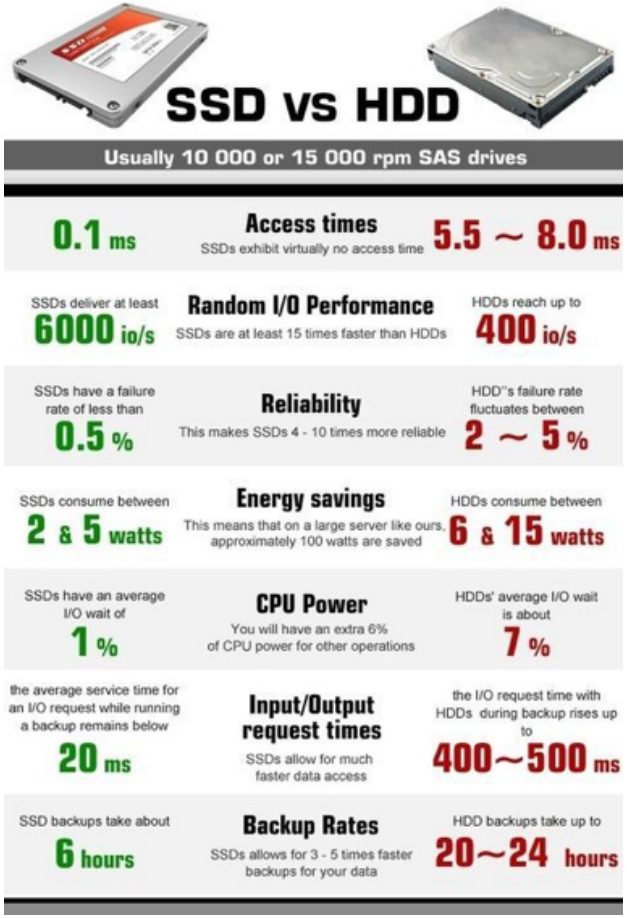
Solid State Drive (SSD)

SSDs uses NAND flash memory chips (i.e. Non-Volatile) to store data.

Additional Benefits:

- **Durability**: Since there are no moving parts, SSDs are less prone to physical damage, offering better shock resistance.
- **Internal Parallelism**

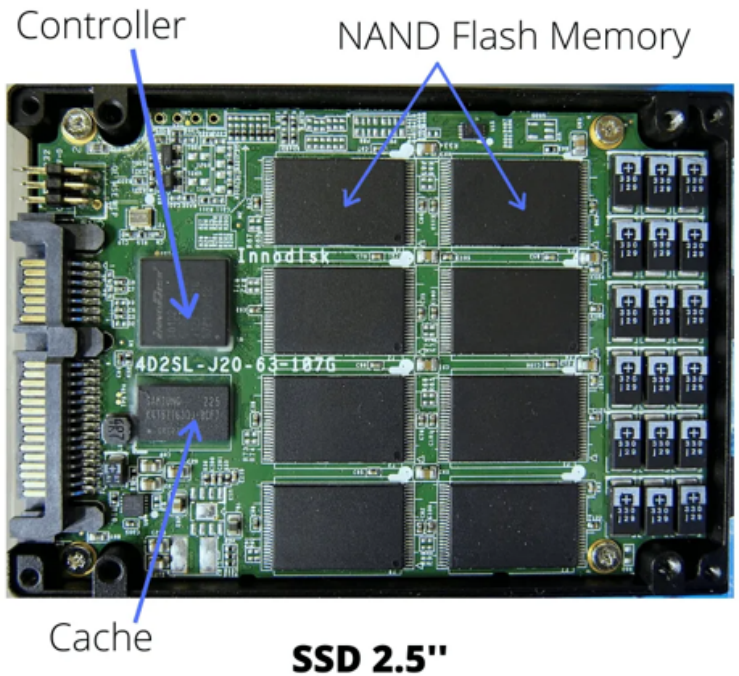




Continued on the Next Slide

SSD Architecture

The architecture of an SSD is designed to maximize performance and reliability.



Cache/DRAM: A small amount of volatile memory used to frequently accessed data to speed up read and write operations.

NAND Flash Memory:
This is the primary storage medium in SSDs. It consists of memory cells that store data in the form of electrical charges.

Channels:
SSDs have multiple independent **channels** that connect the controller to the NAND chips. Data is written to multiple channels in parallel, significantly improving throughput and reducing latency.

Controller:
The controller is the brain of the SSD, responsible for managing data read/write operations, error correction, wear leveling, and garbage collection. It coordinates data flow between the NAND flash and the host system.

3D NAND Flash Memory

•**NAND Flash Memory**: Floating-gate-based non-volatile memory.
•**Function**: Holds electrical charge to store data persistently.
•**Advantages**: High data storage density, fast read/write speeds, and durability.
•**Data Unit**: Transistors connected in series as word-lines to increase storage density.

3D NAND Flash Memory Architecture:
•**Chip Composition**: Composed of several planes.
•**Planes**: Each plane contains many flash blocks (BLK).
•**Blocks**: Basic unit for erasing (ERS).
•**Word-Lines and Strings**: Block consists of many physical word-lines (PWL) and several strings (STR).

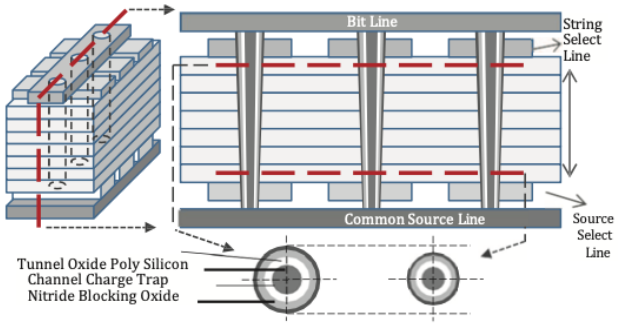


Figure 3. The organization of 3D NAND flash memory

Example:

- 96 physical word-lines (3D NAND with 96 layers).
- 4 strings.
- 384 logical word-lines (WL) (0 to 383).

•**Programming Process**: Applies voltage to one selected physical word-line and activates one string to inject charges into cells.

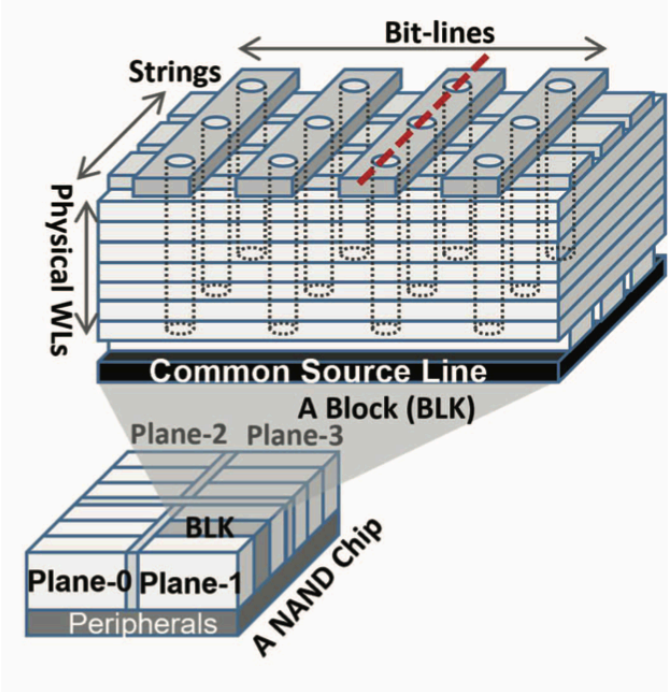


Figure 1. 3D NAND Flash Memory Organization.

- **Wordlines(WLs)**: Imagine a long horizontal wire running through a skyscraper. Applying voltage to a wordline selects a specific "floor" (layer) of memory cells.
- **Bitline**: Think of these as vertical wires passing through all floors. Each bitline connects to a column of memory cells. Data is written or read through these lines. *
- **String**: A single vertical stack of memory cells connected by a bitline, spanning multiple layers (floors).

Essentially, wordlines select the layer, bitlines carry data, and a string is the entire vertical column of cells accessed

Bit Lines and Data Pages

•**Bit Lines**: Each word-line connects many bit lines.

•**Number of Bit Lines**: 18KB page (16KB user data, 2KB spare area) = 147,456-bit lines.

•**Data Pages**: Logical word-line stores one to four pages based on cell types (e.g., TLC).

•**Types of Data Pages:**

•**LSB Page**: Least significant bit of memory cells.

•**CSB Page**: Central significant bit of memory cells.

•**MSB Page**: Most significant bit of memory cells.

Multi-Plane Commands

•**Purpose**: Support high access speed by operating on different planes in parallel.

•**Commands**:

- **Multi-Plane Page Read**: Reads pages on different planes.
- **Multi-Plane Word-Line Program**: Programs word-lines on different planes.
- **Multi-Plane Block Erase**: Erases blocks on different planes.

• **Completion**: Command completes when operation is finished on all targeted pages, word-lines, or blocks.

• **Summary**: Modern 3D NAND flash memory maximizes storage density and performance.

• **Key Components**: Planes, blocks, word-lines, strings, and bit lines.

• **Optimization**: Multi-plane commands enhance access speed and efficiency.

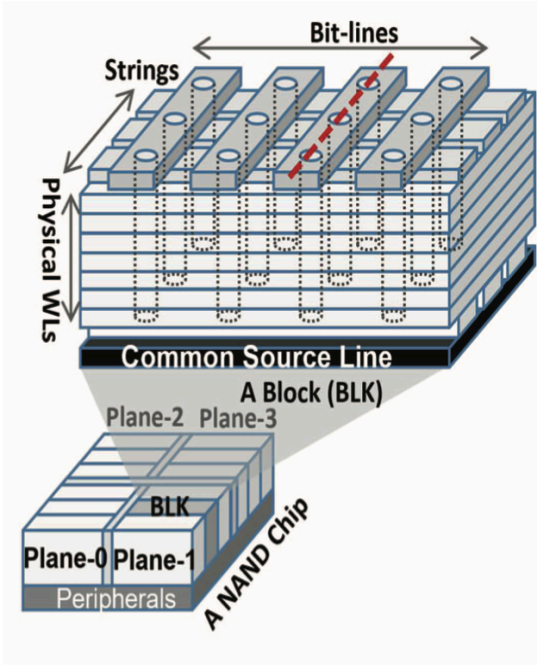


Figure 1. 3D NAND Flash Memory Organization.

SSD Internal Parallelism

All flash memory chips work independently to increase storage throughput.

Superblock Organization

- **Superblock (SB)**: Groups multiple blocks from different planes and chips.
- **Formation**: One block from each plane in different chips of channels is picked to form a superblock.
- **Components**:
- **Physical Word-Lines**: Labeled with the same word-line number are bonded as a super word-line.
- **Pages**: Pages with the same page type (LSB, CSB, MSB) are organized as a superpage.

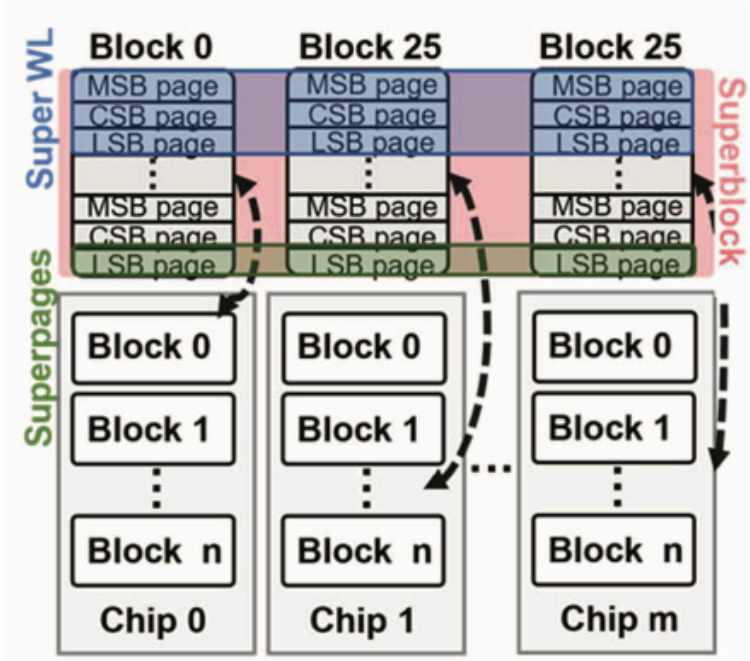


Figure 2. Superblock, super word-line, and superpage.

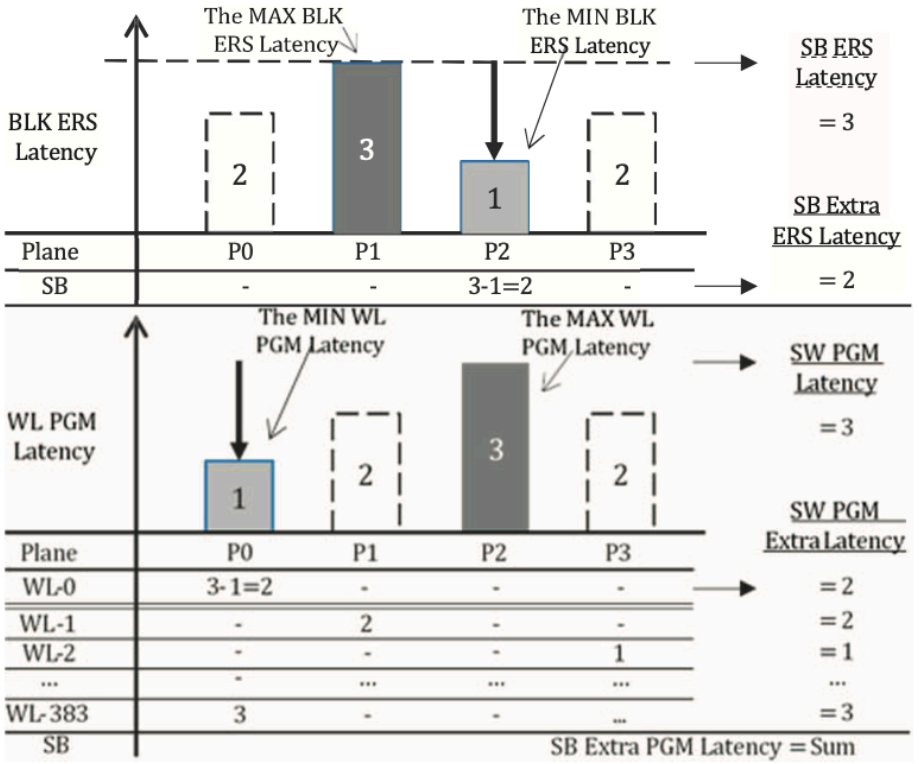
Parallel Operations

- **Super Word-Line**: Word-lines in the same super word-line are programmed simultaneously.
- **Superpage**: Pages in the same superpage can be read in parallel.
- **Example**:
- **Superblock**: Formed by aggregating Block 0 on Chip 0, Block 25 on Chip 1, and Block 25 on Chip m.
- **Multi-Plane Command**: Reported as completed when the issued command is done on all planes.

Problem Statement

SSD performance degrades when slow flash blocks are grouped with fast blocks in super pages.

- Process variations in flash memory manufacturing cause performance discrepancies.
- Misaligned grouping in superblocks increases latency, reducing overall system efficiency.



The Challenge: Process Variations

- **Problem:** Process variations during manufacturing cause some flash blocks/word-lines to be slower than others.
- **Impact:** When slow and fast blocks are grouped together in superblocks or super pages, the slowest block drags down overall performance, causing **extra latency**.

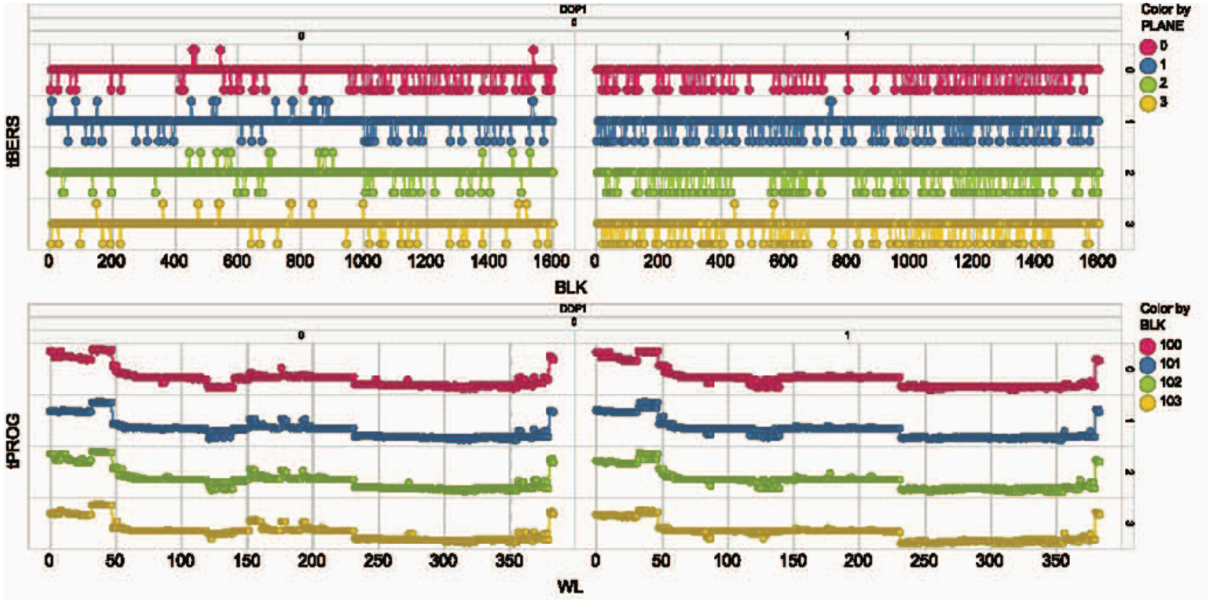


Figure 5. Collected latency of erasing block (top) and programming word-line (bottom) of different blocks. tBERs represents the erase latency for each block. Collected from two different chips on a double die NAND flash memory package. tPROG denotes the program latency for each word-line.

Top Section (Block Erase Latency): Shows how different flash blocks have varying erase times, with slower blocks causing delays.

Bottom Section (Word-line Programming Latency): Illustrates variability in word-line programming times, where slower word-lines increase latency.

Challenge: Minimizing the performance bottleneck caused by these slow blocks and word-lines.

Process Variation and Similarity in Flash Memory

- **Process Variation:**
  + Natural variation in transistor attributes (width, length, and oxide thickness) during fabrication
  + Results in varied performance and reliability among transistors
  + More pronounced in small process node sizes (<65nm)
- **Effects of Process Variation:**
  + Impacts flash memory performance, reliability, and lifetime
  + Can lead to uncorrectable error bits and reduced storage performance
- **Previous Studies on Process Variation:**
  + Proposed schemes to improve read/write performance
  + Enhance flash memory reliability and extend lifetime
  + Alleviate the impact of process variation on flash memory performance

Process Similarity:
+ Cells in the same word-line layer have similar size and characteristics
+ Different apertures of cells between word-line layers influence performance and reliability
+ NAND vendors group WL layers and apply different operating parameters to each group
+ Results in process similarity between WL layers

- **Benefits of Process Similarity:**
  + Can be exploited to improve read/write performance and prolong flash memory lifetime
  + Previous studies have proposed schemes to reuse parameters and minimize uncorrectable error occurrence
- **Implications for Superblock Organization:**
  + Flash memory controller must organize superblocks carefully to minimize extra program latency
  + Storage performance degrades if superblock organization is not optimal

Objective & Motivation

To minimize the extra latency in superblock organization, which is caused by the process variation among blocks and word-lines in a flash memory.

The specific motivations are:

- **Process variation**
- **Extra latency**
- **Storage performance**

The motivation is driven by the need to:

- **Improve storage performance**: By minimizing extra latency and unifying blocks and word-line performance.
- **Increase efficiency**: By reducing the time wasted due to process variation and extra latency.
- **Enhance reliability**: By minimizing the impact of process variation and extra latency on storage reliability.

Character And Findings

Experimental Results:

•**Tested 24 real flash memory chips** to analyze performance under varying conditions:

- Conducted experiments across **P/E cycles** ranging from **0 to 3,000**.
- Evaluated **high-temperature data retention** effects on SSD performance.

Key Features Tested:

•**Block Erase Latency:** Measured time taken to erase each flash block.

•**Word-line Programming Latency:** Measured time taken to program each word-line.

•**Superblock Organization:** Compared latency impacts of random vs. optimized block grouping.

TABLE I. THE RESULTS OF THESE EIGHT DIRECTIONS.

Method	PGM LTN ↓ (Avg.)	Imp. %
SEQUENTIAL	1,367.57 μs	10.45%
ERS-LTN	1,118.35 μs	8.55%
PGM-LTN	1,356.38 μs	10.37%
OPTIMAL (8)	**2,550.73 μs**	**19.49%**
LWL-RANK (8)	1,845.64 μs	14.11%
PWL-RANK (8)	2,036.86 μs	15.57%
STR-RANK (8)	**2,390.05 μs**	**18.27%**
STR-MED (4)	**2,189.94 μs**	**16.74%**

Findings:

•**Latency Observations:**

- Mixing slow and fast blocks in superblocks resulted in **extra latency** during operations.

•**Results from Optimized Grouping:**

- Achieved a **16.61% reduction in extra program latency**.
- Achieved a **34.55% reduction in extra erase latency** through optimal grouping.

Action Plan:

•**Optimization Strategy:**

- Refine the **QSTR-MED** method to enhance grouping of flash blocks based on performance characteristics.

•**Implementation:**

- Integrate findings into SSD firmware for improved flash memory management.

Optimization Strategies

A total of 8 methods were proposed to optimize the grouping of flash blocks into superblocks, each aiming to minimize latency and improve performance.:

1. **Sequential Assembly**: Group blocks with the same sequence number from different chips as a superblock.
2. **Erase Latency Assembly**: Group blocks with similar erase latency from different chips as a superblock.
3. **Program Latency Assembly**: Group blocks with similar program latency from different chips as a superblock.
4. **Local Optimal Assembly**: Use a brute force method to find the optimal combination of blocks from different chips to minimize program latency.
5. **LWL-Rank Assembly**: Rank logical word-lines in each block by program latency and compare the ranks between blocks to calculate the distance.
6. **PWL-Rank Assembly**: Rank physical word-lines in each block by program latency and compare the ranks between blocks to calculate the distance.
7. **STR-Rank Assembly**: Rank strings in each block by program latency and compare the ranks between blocks to calculate the distance.
8. **STR-Median Assembly**: Assign a rank of 0 or 1 to each string in a block based on its program latency and compare the ranks between blocks to calculate the distance.

Continued on the Next Slide

Methodology: Each logical word-line in a block is ranked based on its **program latency**.

•The ranking helps identify which blocks have similar performance characteristics.

•The goal is to group blocks with lower latency together to create superblocks that minimize overall latency during operations.

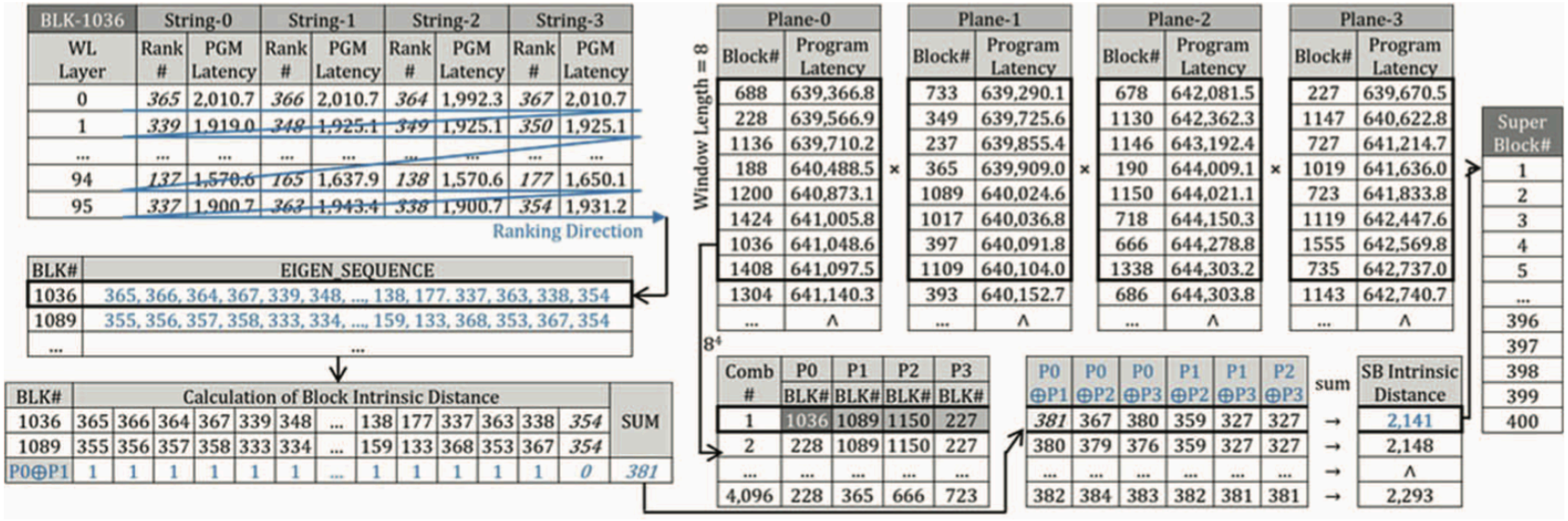


Figure 7. The LWL-rank assembly.

Illustrates the process of ranking logical word-lines (LWL) within flash memory blocks to optimize the grouping of superblocks.

Proposed Scheme for QSTR-MED

1. **Method Integration:** Combines features from eight proposed superblock organization strategies to enhance SSD performance.
2. **Superblock Organization:** Groups flash memory blocks into superblocks based on program latency and similarity.
3. **Selective Comparison:** Reduces latency by comparing only blocks with similar performance characteristics.
4. **Performance Optimization:** Minimizes extra latency during read/write operations.
5. **Memory Efficiency:** Requires minimal additional memory overhead.
6. **Low Computational Complexity:** Designed for quick processing with low resource demands.
7. **Adaptability:** Dynamically adjusts to changes in performance for real-time optimization.

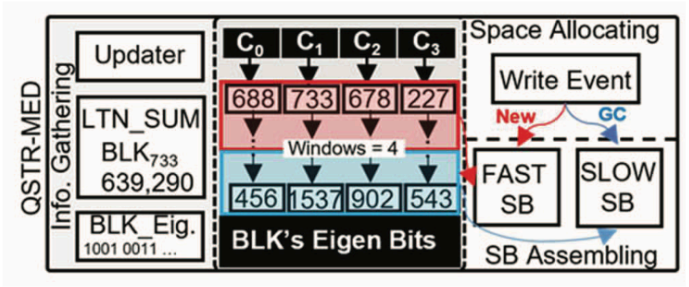


Figure 8. The proposed QSTR-MED scheme.

How QSTR-MED Works:

1. **Block Organization:**

1. **Superblock Formation:** Organizes flash memory blocks into superblocks by evaluating their program latency and performing similarity checks.

2. **Targeted Comparisons:**

1. **Reduced Latency:** Focuses on comparing only selected blocks with similar performance characteristics, minimizing unnecessary comparisons and computational effort.

Performance Evaluation

TABLE V. THE RESULTS OF EXTRA PROGRAM AND ERASE LATENCY

Methods	Extra PGM LTN	Extra ERS LTN
Random	13,084.17 μs	41.71 μs
Sequential	11,716.60 μs	40.12 μs
Optimal	10,533.44 μs	22.65 μs
QSTR-MED(4)	10,911.53 μs	25.10 μs
STR-MED(4)	10,894.23 μs	24.97 μs

- **Key Performance Improvements:**
  - QSTR-MED achieves comparable performance to the optimal method with minimal overhead.
  - Reduces program latency by 16.61% and erase latency by 59.82% compared to random groupings.
- **Computing Overhead Reduction:**
  - QSTR-MED reduces computing overhead by 99.22% compared to other methods like STR-MED.

Comparison with Traditional Methods

Traditional Methods:

•Random Block Organization:

- Often leads to significant extra latency due to lack of structure in data grouping, causing inefficient read/write operations.

QSTR-MED:

•Minimized Latency:

- Effectively reduces latency by organizing flash memory blocks based on similar performance characteristics.

•Practical Feasibility:

- Achieves improvements without excessive computational overhead, making it suitable for real-world applications.

QSTR-MED significantly outperforms traditional methods by reducing latency while maintaining practical efficiency, as illustrated by the latency distributions in Figure 13.

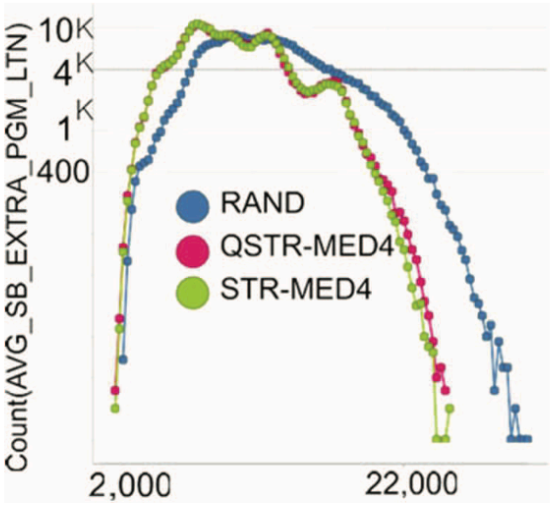


Figure 13. The distributions of the extra PGM LTN.

Distribution of Extra Program Latency:

•**Figure 13** shows that traditional methods have a higher and broader latency distribution, indicating greater extra latencies.

•In contrast, **QSTR-MED** shifts the latency distribution to the left, demonstrating lower average latency and improved performance.

Conclusion

Summary:

- **Performance Challenges:**
- **Process Variations:** The inherent variations in flash pages during manufacturing lead to significant performance challenges in SSDs, resulting in inconsistent latency and overall inefficiency.
- **QSTR-MED Solution:**
- The **QSTR-MED** method offers a practical solution by efficiently organizing flash pages into superblocks based on similar latency characteristics. This organization minimizes extra latency and improves SSD performance.

Key Findings:

1. **Arbitrary Superblock Organization**: Arbitrarily organizing superblocks results in long extra latency.
2. **Eight Directions**: Eight directions were proposed to minimize extra latency, including a local optimal solution.
3. **QSTR-MED**: The QSTR-MED was developed as a practical solution to minimize extra latency in superblock organization.
4. **Reduced Computing Overhead**: QSTR-MED significantly reduces computing overhead, making it a feasible solution.
5. **Minimal Extra Latency**: QSTR-MED can minimize extra program and erase latency in all superblocks.

Continued on the Next Slide

All Superblocks Improvement:

•**Figure 14** visually emphasizes the overall improvements achieved through the QSTR-MED method.

•The figure likely illustrates enhancements in program and erase latency, showcasing how the proposed solution leads to better computational efficiency across all superblocks.

•This graphical representation supports the conclusion by quantifying the positive impact of QSTR-MED on SSD performance

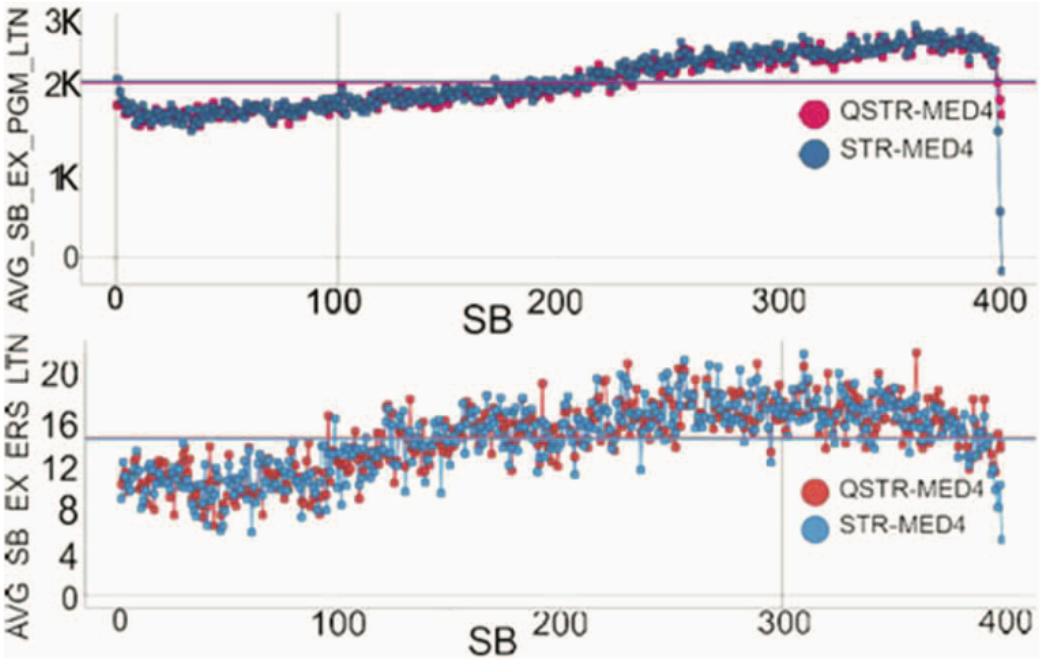


Figure 14. All superblocks improvement.

Future Work

- **Refinement of Strategies:**
- There is potential for further refinement of superpage grouping strategies to enhance the effectiveness of QSTR-MED and adapt to evolving data workloads.
- **Hardware Enhancements:**
- Exploring hardware-based solutions could provide additional optimization opportunities, potentially leading to even greater performance improvements in SSDs

References

Research Paper : https://ieeexplore.ieee.org/document/10476406

E. K. Ardestani et al., "Supporting Massive DLRM Inference through Software Defined Memory," 2022 IEEE 42nd International Conference on Distributed Computing Systems (ICDCS), Bologna, Italy, 2022, pp. 302-312, doi: 10.1109/ICDCS54860.2022.00037. keywords: {Deep learning;Measurement;Power demand;Computational modeling;Memory management;Software;Hardware;DLRM;Hierarchical Memory;Software Defined Memory;Recommendation Models;Inference},

Y. Cai, E. F. Haratsch, O. Mutlu and K. Mai, "Threshold voltage distribution in MLC NAND flash memory: Characterization, analysis, and modeling," 2013 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 2013, pp. 1285-1290, doi: 10.7873/DATE.2013.266. keywords: {Threshold voltage;Noise;Flash memory cells;Correlation;Kernel;Vectors;NAND Flash;Memory Reliability;Memory Signal Processing;Threshold Voltage Distribution;Read Retry},

Y. Cai, O. Mutlu, E. F. Haratsch and K. Mai, "Program interference in MLC NAND flash memory: Characterization, modeling, and mitigation," 2013 IEEE 31st International Conference on Computer Design (ICCD), Asheville, NC, USA, 2013, pp. 123-130, doi: 10.1109/ICCD.2013.6657034. keywords: {Decision support systems;NAND flash;program interference;reliability;read retry;error correction;error model},

M. -F. Chang and S. -J. Shen, "A Process Variation Tolerant Embedded Split-Gate Flash Memory Using Pre-Stable Current Sensing Scheme," in IEEE Journal of Solid-State Circuits, vol. 44, no. 3, pp. 987-994, March 2009, doi: 10.1109/JSSC.2009.2013763.

keywords: {Split gate flash memory cells;Flash memory;Fluctuations;Circuits;Temperature sensors;Foundries;Threshold voltage;Tail;Nonvolatile memory;Manufacturing processes;Flash;process variation;split-gate},

J. Chen, Y. Wang, A. C. Zhou, R. Mao and T. Li, "PATCH: Process-Variation-Resilient Space Allocation for Open-Channel SSD with 3D Flash," 2019 Design, Automation & Test in Europe Conference & Exhibition (DATE), Florence, Italy, 2019, pp. 216-221, doi: 10.23919/DATE.2019.8715197. keywords: {Automation;Europe;Integrated circuits;Electroencephalography;Gallium nitride;Three-dimensional flash memory;process variation;open-channel SSD;charge-trap flash;space allocation},

T. -Y. Chen, Y. -H. Chang, Y. -H. Kuan and Y. -M. Chang, "VirtualGC: Enabling erase-free garbage collection to upgrade the performance of rewritable SLC NAND flash memory," 2017 54th ACM/EDAC/IEEE Design Automation Conference (DAC), Austin, TX, USA, 2017, pp. 1-6, doi: 10.1145/3061639.3062339. keywords: {Three-dimensional displays;Programming;Memory management;Microprocessors;Reliability engineering;Rewritable SLC;erase-free scheme;garbage collection},

Y. Di, L. Shi, C. Gao, Q. Li, C. J. Xue and K. Wu, "Minimizing Retention Induced Refresh Through Exploiting Process Variation of Flash Memory," in IEEE Transactions on Computers, vol. 68, no. 1, pp. 83-98, 1 Jan. 2019, doi: 10.1109/TC.2018.2858771.
keywords: {Time-frequency analysis;Error correction codes;Reliability;Memory management;Flash memories;Minimization;Retention time;refresh optimization;process variation;flash memory},

A. P. Ferreira, S. Bock, B. Childers, R. Melhem and D. Mossé, "Impact of process variation on endurance algorithms for wear-prone memories," 2011 Design, Automation & Test in Europe, Grenoble, France, 2011, pp. 1-6, doi: 10.1109/DATE.2011.5763156. keywords: {Equations;Mathematical model;Phase change materials;Algorithm design and analysis;Approximation methods;Random access memory;Analytical models},

J. Guo, D. Wang, Z. Shao and Y. Chen, "Data-Pattern-Aware Error Prevention Technique to Improve System Reliability," in IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, vol. 25, no. 4, pp. 1433-1443, April 2017, doi: 10.1109/TVLSI.2016.2642055.
keywords: {Encoding;Programming;Bit error rate;Error correction codes;Interference;Reliability;Logic gates;Flash memories;reliability;storage management;solid state disk (SSD)},

Thank You