TailCut: Improving Performance and Lifetime of SSDs Using Pattern-Aware State Encoding

Jaeyong Lee
Seoul National University
jylee@davinci.snu.ac.kr

Myungsunk Kim
Kyungpook National University
ms.kim@knu.ac.kr

Wonil Choi
Hanyang University
wonilchoi@hanyang.ac.kr

Sanggu Lee
Seoul National University
jakesanggulee@davinci.snu.ac.kr

Jihong Kim
Seoul National University
jihong@davinci.snu.ac.kr

ABSTRACT

Although lateral charge spreading is considered as a dominant error source in 3D NAND flash memory, little is known about its detailed characteristics at the storage system level. From a device characterization study, we observed that lateral charge spreading strongly depends on vertically adjacent state patterns and a few specific patterns are responsible for a large portion of bit errors from lateral charge spreading. We propose a new state encoding scheme, called `TailCut`, which removes vulnerable state patterns by modifying encoded states. By removing vulnerable patterns, `TailCut` can improve the SSD lifetime and read latency by 80% and 25%, respectively.

1 INTRODUCTION

3D NAND flash memory has been a key enabler of the continuous growth in the storage capacity of SSDs. By stacking memory cells in a vertical direction, the capacity of recent 3D flash memory has increased up to 1-Tera bits per die. Although an organization of stacked word line (WL) layers in 3D flash memory was very successful for the capacity increase, it introduced new reliability problems that were not present in 2D flash memory. One such a key reliability problem is *lateral charge spreading*. Lateral charge spreading occurs between vertically-adjacent cells that are connected to the same bit line. As charge of a flash cell in a WL leaks and moves to its (vertically) neighboring [1] flash cells, all the neighboring flash cells are vulnerable to bit errors.

Although the lateral charge spreading problem has been well investigated at the device level [1, 2], little is known how it can be effectively managed at the SSD level. In order to develop an efficient scheme for mitigating an impact of the charge loss problem, in this paper, we conducted an extensive experiment using real 3D TLC flash devices so that the error behavior of lateral charge spreading can be better understood at the storage level. (See Section 3 for details of our characterization study.)

[1]We use *neighboring* and *vertically neighboring* interchangeably in this paper.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
DAC '22, July 10–14, 2022, San Francisco, CA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9142-9/22/07…$15.00
https://doi.org/10.1145/3489517.3530471

From our characterization study, we made three key observations. First, lateral charge spreading is strongly *data dependent*. That is, the degree of lateral charge spreading varied significantly depending on specific state patterns among neighboring flash cells. For example, lateral charge spreading from the worst state pattern was higher by up to 80% over the best state pattern. We call a state pattern *weak* when lateral charge spreading from the state pattern is high. Second, weak patterns are a major contributor to retention errors of 3D flash memory because they place V_{th} states of affected flash cells at the long tail end of threshold voltage distributions. Third, retention errors from weak patterns are a main cause of shortening the SSD lifetime and degrading the read latency of an SSD.

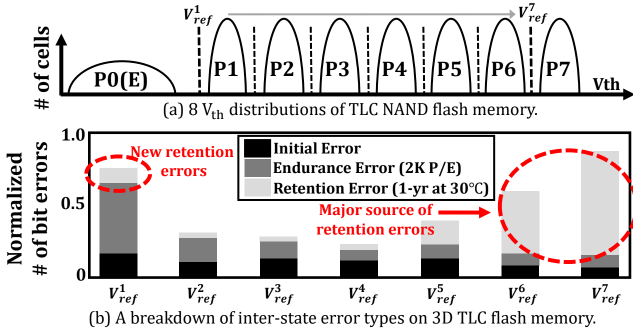
Motivated by three key findings, we propose a new state encoding technique, called `TailCut`, which removes weak patterns by modifying encoded states so that lateral charge spreading can be minimized. `TailCut` examines whether the current write request will generate a weak pattern when the encoded states of neighboring cells are considered. If the current write can introduce a weak pattern, `TailCut` modifies the state encoding of the current write so that a potential weak pattern can be avoided by properly flipping data bits. When such flipped data are read, `TailCut` restores its original data by flipping back the inverted bits.

While `TailCut` can be implemented in firmware as a part of an extended FTL, a direct SW-based implementation can cause a significant performance degradation as well as a capacity reduction. Managing information on flipped bits per WL can increase the latency of normal read/write operations and requires extra space for storing flipped bit positions. For an efficient implementation of `TailCut`, we devised a simple but effective on-chip scheme that implements most steps of `TailCut` by reusing existing flash datapath when they are idle with small control logic modifications. Our implementation incurs little performance/resource overhead over a conventional SSD controller implementation.

In order to evaluate the effectiveness of `TailCut`, we have constructed a weak pattern-aware SSD, called `TcSSD`, using an open source SSD emulation platform [3]. Evaluation results using various workloads show that `TcSSD` can improve the read latency and SSD lifetime by up to 25% and 80%, respectively, over a baseline SSD that does not consider lateral charge spreading.

This paper makes the following key contributions:

- To our knowledge, this work is the first to quantitatively identify the data dependency of lateral charge spreading in 3D flash memory so that it can be exploited at the SSD level. (See Section 3.)



(a) 8 V_{th} distributions of TLC NAND flash memory.

(b) A breakdown of inter-state error types on 3D TLC flash memory.

Figure 1: V_{th} distributions of TLC flash with its inter-state error breakdown.

- We proposed a new state encoding technique, `TailCut`, that can remove most weak patterns with its efficient implementation within a flash chip with little added overhead. (See Section 4.)
- We demonstrated the efficiency of `TailCut` using `TcSSD`, which confirmed that the read latency and SSD lifetime can be improved by up to 25% and 80%, respectively. (See Section 5.)

2 RELIABILITY CHARACTERIZATION

The flash memory consists of multiple flash cells, which store data, and peripheral circuits, which support flash commands such as read and write. For writes, flash memory changes target flash cell's V_{th} states based on individual bit data by injecting electron charges into the flash cell. Figure 1(a) illustrates V_{th} distributions for the TLC flash memory that stores 3 bits (LSB, CSB, and MSB bits) within a single flash cell by using 2^3 distinct V_{th} states. Eight states are distinguished based on their V_{th} distributions. The higher the state is, the more its V_{th} distribution is skewed to the right. In order to successfully retrieve the stored data, 7 different read reference voltages (V^i_{ref}'s) are used to sense their V_{th} states. The initial V_{th} distribution shown in Figure 1(a) gets progressively distorted under various error conditions, thus causing an increasing number of bit errors from the stored data. For example, if flash memory experience repetitive program/erase (P/E) cycles, a V_{th} distribution of each state gets wider and neighboring V_{th} distributions of two adjacent states may overlap each other. When two V_{th} distributions overlap, we cannot reconstruct the stored data by using the original read reference values.

2.1 Inter-State Error Distribution

In order to understand the impact of different error conditions in 3D TLC flash memory, we performed a characterization study using commercial 3D TLC flash chips. We measured two types of errors, endurance errors and retention errors, between two adjacent states. Figure 1(b), which summarizes a key finding of our study, shows a typical breakdown of different error types for each read reference voltage V^i_{ref}.[2] As shown in Figure 1(b), we observed that a dominant error type as well as an error frequency is strongly *state dependent*. Most of the endurance errors occur between the E and P1 state, whereas most of retention errors are concentrated in higher states, P6 and P7.

[2]Note that a flash cell error occurs when the state of the cell cannot be correctly recognized by its read reference voltage V^i_{ref}. Therefore, we show a breakdown of error types for each inter-state interval, which is indicated by its read reference voltage.

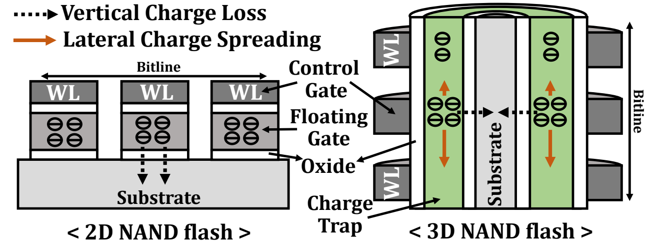


Figure 2: Differences in the charge loss mechanism between 2D and 3D flash memory.

We also observed that retention errors play a bigger role in deciding the lifetime of 3D flash memory compared to 2D flash memory. For example, retention bit errors account for more than 65% of the total flash bit errors under the common retention requirement (e.g., 1 year retention after 2K P/Es at 30°C). A large increase in retention errors is attributed to *lateral charge spreading (LCS)*, which is the unique phenomenon of 3D flash memory. Furthermore, under the influence of LCS, moderate number of retention errors were observed in the E state. No retention errors was observed in the E state of 2D flash memory.

2.2 Mechanism of LCS

Before we investigate the impact of LCS on 3D flash memory, we describe why LCS occurs in 3D flash memory. The transition from 2D to 3D flash memory was possible by leveraging a new type of flash cell (i.e., a charge-trapping flash cell) because a conventional floating-gate flash cell was not suitable for a vertically stacked organization of 3D flash memory. However, due to its unique structure, 3D flash memory experiences LCS, a new phenomenon in 3D flash memory. Figure 2 illustrates why lateral charge loss is present in 3D flash memory while it is not an issue in 2D flash memory. In 2D flash memory, since each flash cell is isolated by dielectric layers, stored charges can leak only through the thin oxide layer between floating-gate and substrate by various de-trapping mechanisms. (This phenomenon is commonly called as vertical charge loss.) On the other hand, unlike 2D flash memory, flash cells along the same bitline in 3D flash memory share the same charge trap layer. Therefore, stored charges in a flash cell can be easily moved (i.e., spread) into vertically adjacent neighboring cells whose charge trap layer is directly connected, thus significantly accelerating the speed of charge loss. Although rather contradictory, we call this type of charge loss *lateral charge spreading* so that it can be distinguished from vertical charge loss of 2D flash memory.

Since LCS is a diffusion process, the effect of LCS on retention errors can be intuitively understood using the theory of a diffusion process. For example, the rate of charge mixture is proportional to the concentration gradient among neighboring states. That is, the bigger the difference between the V_{th} state of adjacent cells, the faster the rate of charge change among adjacent cells. As the retention time increases, the rate of charge mixture gets slower but the total amount of charge loss keeps increasing.

3 IMPACT ANALYSIS OF LCS

Since LCS is a new phenomenon in 3D flash memory and it is a key reliability concern, we have conducted an extensive characterization study using 160 3D TLC flash chips so that we can better understand the impact of LCS on the reliability and performance

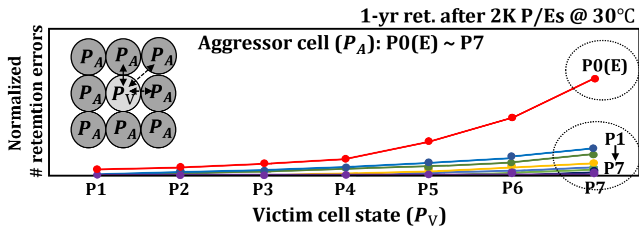
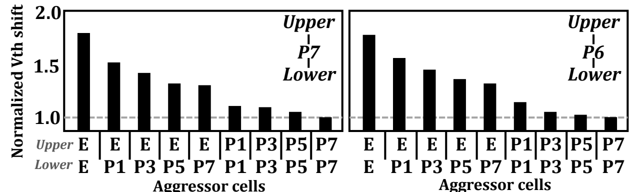


Figure 3: The effect of LCS on retention errors under varying test patterns.



(a) Victim cell state : P7 @ 2K PEs 1-yr ret. (b) Victim cell state : P6 @ 2K PEs 1-yr ret.

Figure 4: Variations on V_{th} shift under different test patterns.

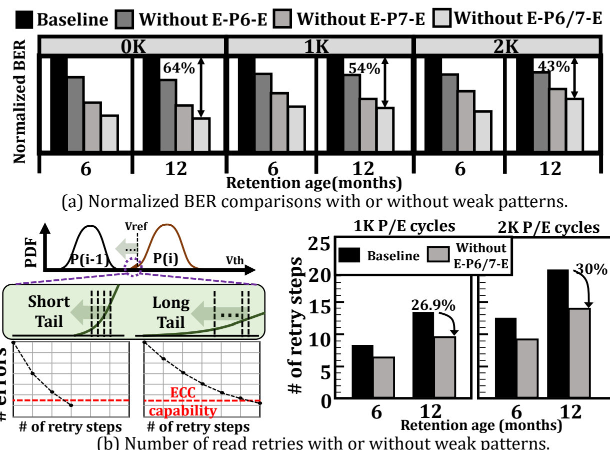
of 3D flash-based SSDs. To minimize the potential distortion of the evaluation results from process variations, we evenly selected 120 test blocks from each chip at different physical locations, and tested a total of 11,520,000 pages (i.e., 3,840,000 WLs) in our study. As a reliability measure of flash cells, we used the number of retention bit errors while changing both P/E cycles and retention times. Our test procedure follows the JEDEC standard that is the common industry practice for flash products.

3.1 Quantitative Evaluation

As described in Section 2.2, charge movements under LCS is a diffusion process. Since a large gradient of charge concentration accelerates the diffusion process, we investigated how the V_{th} difference between adjacent cells affects the speed of lateral charge spreading. The faster the speed of LCS, the higher the number of retention errors. In order to quantitatively measure the effect of the V_{th} difference among cells (i.e., the difference of charge concentration) on the number of bit errors, we devised a large number of test patterns. Each test pattern (P_V, P_A) represents a 3×3 cell matrix where a victim cell with the state P_V is positioned in the center of the 3×3 matrix and 8 aggressor cells with the state P_A surrounds the victim cell[3]. For example, the pattern ($P3$, $P1$) is used to measure the number of bit errors when P3 state is surrounded by 8 P1 state cells.

Figure 3 shows how the number of retention bit errors changes depending on a combination of a victim cell state and an aggressor cell state. Retention bit errors were measured with the 1-year retention time requirement after 2K P/E cycles at 30°C. All the measurements were normalized over the number of retention bit errors when both the victim cell and aggressor cells are in the same state. As expected, the BER significantly increases as the V_{th} difference between P_V and P_A gets larger because of the diffusion nature of LCS. When the V_{th} difference is largest, that is, in the test state pattern ($P7$, $P0$), the number of retention bit errors was 67% higher than the test state pattern ($P7$, $P7$). From our test state

[3] Although not included in the paper, charge loss among adjacent cells along different directions (e.g., horizontal and diagonal directions) were also investigated. The impact of charge loss on retention errors along these directions was negligible because cells along these directions are physically separated without sharing common medium for charge spreading.



(a) Normalized BER comparisons with or without weak patterns.

(b) Number of read retries with or without weak patterns.

Figure 5: Impact of weak patterns on lifetime and performance.

pattern variations, we observed that two test patterns, ($P7$, $P0$) and ($P6$, $P0$), exhibited the largest number of retention errors, which we call *weak patterns*.

Since LCS occurs along the vertical direction only, we denote two weak patterns by a 3x1 cell matrix `X-Y-Z` where `X` is the state of the least-recently programmed cell and `Z` is the state of the most recently programmed cell. Assuming that a WL is programmed from the bottom to the top direction, two weak patterns can be represented by `E-P7-E` and `E-P6-E`, respectively.

Since the amount of charge loss under LCS increases monotonically until there exists no concentration gradient among neighboring cells, we investigated the *quality* of error bits from LCS when the retention time requirement is very long. As the quality measure of an error bit, we compared the amount of a V_{th} shift under various 3×3 cell matrices. The amount of a V_{th} shift of a victim cell is normalized to that of cells located in the center of V_{th} distribution. Figure 4 shows how the amount of a Vth shift is affected by the states of neighboring cells after the 1-year retention time after 2K P/E cycles at 30 °C. When the P7 and P6 states are placed between the upper and lower E states, the amount of a V_{th} shift is the largest. For such a large V_{th} shift, the victim cell is likely to be positioned around the tail end of its V_{th} distribution, which, in turn, significantly increases a possibility of a retention error. Furthermore, since the quality of retention errors from LCS are very bad, these retention errors are likely to decide the SSD lifetime and read tail latency because these metrics are dependent on the worst-case reliability characteristics.

3.2 Impact on Lifetime and Performance

In order to understand the impact of LCS on the lifetime and performance of 3D flash memory, we compared the number of retention errors of four versions of data written to test flash blocks: a baseline data, the same data without the `E-P7-E` pattern, the same data without the `E-P6-E` pattern and the same data without both `E-P7-E` and `E-P6-E` patterns. Figure 5(a) shows a comparison result under varying P/E cycles. When we eliminate both `E-P7-E` and `E-P6-E` patterns, the average number of bit errors was reduced by 43.8%. Assuming that our ECC module can correct up to 60-bit errors per 1 KB, we observed that the lifetime of 3D flash memory can be extended by up to 80% (i.e., from 5K to 9K P/E cycles).

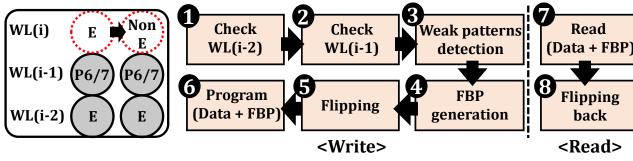
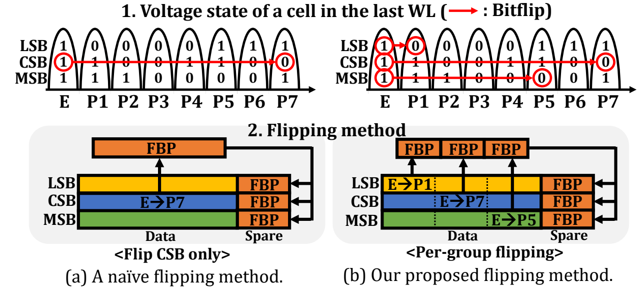


Figure 6: A high-level operational overview of `TailCut`.



(a) A naïve flipping method.

(b) Our proposed flipping method.

Figure 7: Comparisons of two bit flipping approaches.

Eliminating the weak patterns can also improve the flash performance by reducing the number of read retries. Figure 5(b) shows how the number of read retries changes when the weak patterns are removed. Our result indicates that the average number of read retries can be reduced by 30% from 20.3 to 14.2 at the worst condition (i.e., after the 1-year retention time at 2K P/E cycles (30°C)) when both `E-P7-E` and `E-P6-E` patterns are removed. Since the weak patterns cause the state of victim cells to be positioned at the tail end of their V_{th} distributions, we expect a large improvement in the read latency if charge loss from LCS can be effectively mitigated.

4 PATTERN-AWARE STATE ENCODING

4.1 Overview of `TailCut`

As described in Section 3, two patterns, E-P7-E and E-P6-E, are dominant sources of retention errors from LCS. Furthermore, retention errors from these patterns experience a large V_{th} shift so that they tend to be *worst-case* errors. As a simple but effective solution to mitigate the impact of weak patterns, we propose a pattern-aware state encoding scheme, called `TailCut`, which removes the two weak patterns in the data layout. `TailCut` deliberately changes the **E** state of `E-P7/P6-`**`E`** patterns to one of non-E states. Such a state re-encoding can be realized by *flipping* a bit (from "1" to "0") out of three 1's of the E state encoding. When the bit is flipped, we need to keep the position of the flipped bit so that the original data (before bit flipping) can be correctly read. We call this position information the flipped bit position (FBP). FBPs are stored into a WL alongside user data. Figure 6 presents a high-level operational overview of `TailCut`. Whenever three bits (MSB, CSB, and LSB) are programmed to a cell of a WL (WL_i), the state of two neighboring cells of the previously-programmed WLs (WL_{i-2} and WL_{i-1}) are checked if they meet the state configuration of weak patterns. WL_{i-2} is first checked whether in the E state (❶ in Figure 6) and WL_{i-1} is examined whether it is in P6/P7 (❷). When two neighboring cells of WL_{i-2} and WL_{i-1} meet the weak pattern requirement, the cell state of WL_i is checked if it is the E state (❸). For an identified pattern, its FBP data are computed (❹). The FBP data should be generated to restore the original data when it is read. Finally, after cell states of WL_i are properly flipped (❺), flipped data are programmed into WL_i with its FBP information (❻). For a read operation, both the user and FBP data are read (❼) and the original data is flipped back using the FBP data (❽).

4.2 Design Consideration of `TailCut`

Although it is rather straightforward to implement `TailCut` based on Figure 6, a naive implementation can incur significant capacity and performance overheads. In this section, we present our design choices to meet two key implementation requirements of `TailCut`.

4.2.1 Minimizing the size of FBP. One simple way to remove the weak patterns is to flip the CSB bits of the data to be programmed to WL_i. As shown in Figure 7(a), doing so can convert the E state of a cell to the P7 state without making any change to the LSB and MSB bits. Unfortunately, this approach generates FBP data only for the CSB page. Since the FBP data of the CSB page should be stored on the spare area of the CSB page, this requires a very large spare area for the CSB page, which is infeasible in practice. As shown in Figure 7(a), a more practical solution is to use the spare area of the LSB/MSB page for storing the FBP data of the CSB page. However, this approach suffers from extra page reads when the CSB page is read. Since the FBP data of the CSB page must be read from the LSB page and MSB page, the read latency of the CSB page becomes three times slower over that of the LSB/MSB page.

To avoid such a drawback, we devise a novel flipping strategy that uses all three pages in the bit flipping process. Figure 7(b) illustrates our proposed strategy. Unlike the naive approach of Figure 7(a) that converts bits of the CSB page only, our scheme divides the WL into three groups and a different page is used in flipping bits of each group. For the leftmost WL group, we flip bits from the LSB page, converting E to P1. For the middle WL group, we flip bits from the CSB page, converting E to P7. For the rightmost WL group, we flip bits from the MSB page, converting E to P5. In our per-group flipping strategy, FBP data of each page is stored into its own spare area, thus requiring no extra page read as in the naive approach. Furthermore, all three pages require an spare area of the same size for its FBP data.

Although our per-group flipping scheme reduced the maximum FBP size by one third by assigning a different group to a different page, the FBP size is still quite big. For example, when the WL size is 18 KB (i.e., 16 KB page + 2 KB spare), we still require a large FBP size to cover a 6-KB group. To further reduce the size of FBP data per group, we make a *per-chunk* flip decision instead of a bit-level flip decision. We divide a WL into a sequence of 16-bit chunks and flip all the 16 bits of a chunk when any weak pattern is identified within the chunk. Since weak patterns are not commonly observed, when a weak pattern is identified within a 16-bit chunk, it is unlikely that flipping a 16-bit chunk introduces new weak patterns into the chunk.[4] By applying one flipping decision to every 16 bits, the size of FBP data is reduced by $\frac{1}{16}$. For a 18-KB WL, we need 144 B only for per-page FBP data, requiring 0.78% more space for the 18-KB WL.

4.2.2 Hiding the overhead of detecting the weak patterns: In order to detect a weak pattern, as shown in Figure 6, a total of nine

[4]Since an SSD randomizes user data before write, state patterns are believed to be uniformly distributed. The probability of a weak pattern is $\frac{1}{256}$. Therefore, for a 16-bit chunk, it is unlikely for a weak pattern is introduced after a chunk flipping.

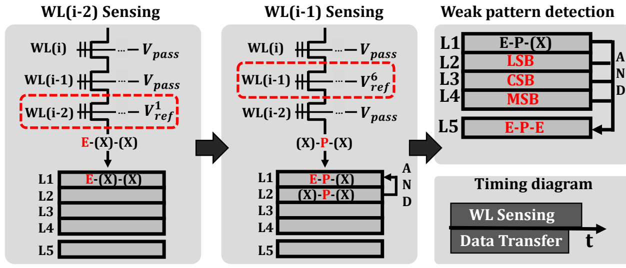
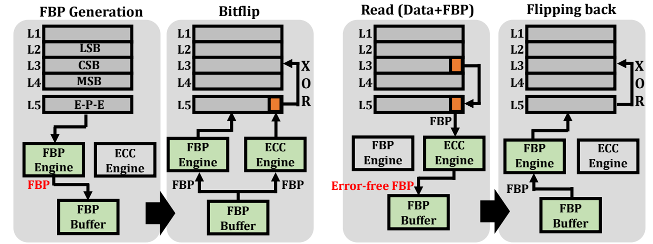


Figure 8: An operational illustration of the on-chip weak pattern detector.



(a) The process of write.

(b) The process of a page read.

Figure 9: Illustration of on-chip modules for TailCut.

pages from WL_{i-2}, WL_{i-1} and WL_i are needed. Since both WL_{i-2} and WL_{i-1} are available only inside a chip, a direct implementation requires at least six off-chip page reads, which take a significant amount of time. To hide a performance overhead, we devised a simple but effective on-chip weak pattern detector which does not need expensive off-chip read operations. Figure 8 illustrates our on-chip weak pattern detector. Our detector exploits existing on-chip page latches and associated bitwise logical operation units that modern flash chips commonly support internally. For example, a TLC flash chip includes five page latches (i.e., four data buffers and one cache buffer) [4, 5]. To identify whether WL_{i-2} and WL_{i-1} is E state and P6/7 state respectively, two consecutive WL sensing operations with V^1_{ref} and V^6_{ref} are conducted. Since V^1_{ref} can distinguish whether E state or not (See Figure 1(a)), the output of sensing operation tells the positions of E state cells in WL_{i-2}. Similarly, the output of V^6_{ref} sensing indicates the positions of P6/7 state cells in WL_{i-1}. Finally, the positions of weak patterns can be computed by bitwise AND operations among three pages of WL_i and the final output of two sensing operations. Note that by overlapping the data transfer time of WL_i with the sensing times of WL_{i-2} and WL_{i-1}, we can hide most of the weak pattern detection time as shown in the timing diagram of Figure 8.

4.3 Overhead Analysis for On-Chip Processing

By implementing the key steps of `TailCut` inside a flash chip, as discussed in Section 4.2.1, we were able to hide most of performance and storage overheads of `TailCut`. However, in order to support on-chip `TailCut` processing, we need a few extra modules that are not present in conventional flash chips. A high-level datapath diagram (i.e., a shaded diagram) of Figure 9 shows the main additional modules needed for `TailCut`. The current implementation requires 1) the FBP engine module (that handles FBP data), 2) an FBP buffer (that works as a temporary buffer), and 3) an on-chip ECC module (that is used to reliably write and read FBP data.)

Figure 9 illustrates how these modules interact to service write and read operations. For a write operation, as shown in Figure 9(a), the FBP engine module generates FBP by accessing latches that stores the position of weak patterns. After FBP is generated, FBP is encoded by an ECC engine and stored in latches to be programmed. To flip the pages, the FBP engine module sets L5 as bit flip flags where 1 enables a bit flip operation on the corresponding bit position. Actual bit flips are implemented by bitwise XOR operation between latches. For a read operation, as shown in Figure 9(b), the stored FBP data is read through the ECC engine and moved to the FBP buffer. The FBP engine module, then, configures L5 as bit flip flags. Similarly to write, bitwise XOR operation will restore the flipped bits to their original bits.

4.3.1 *Analysis on timing overhead.*

Since `TailCut` introduces extra steps to a normal write path, there are small timing overhead. First, the total WL sensing time of WL_{i-2} and WL_{i-1} can take more time than the data transfer time of WL_i. For example, if we assume that the data transfer rate of 1 Gbps and the total WL sensing time takes 40 us, there is a timing overhead of 26 us. Second, there is a timing overhead of accessing a latch to store the generated FBP data. Since the page size is 18 KB, if we assume that 1 Gbps data transfer rate, this will take 18 us. Third, the ECC engine can be implemented in a fully parallel fashion, so its latency is fully hidden. Overall, the timing overhead of 44 us is introduced, which can be considered as negligible over a typical program time of 1980 us.

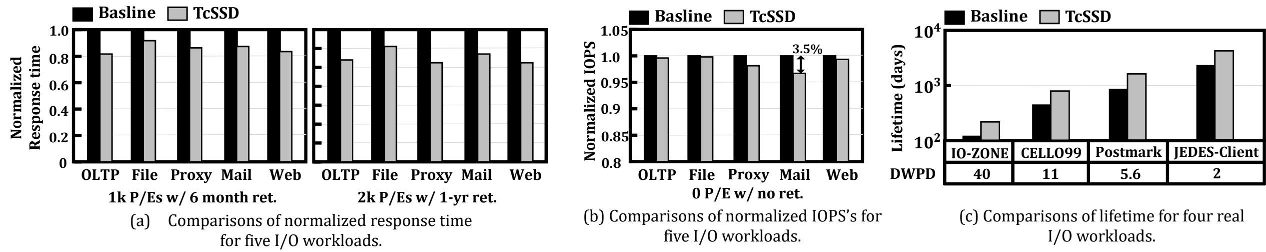
4.3.2 *Analysis on area overhead.*

FBP data need 144 B per page (as described in Section 4.2.). However, in order to support reliable in-flash reads of FBP data, FBP data should be encoded by an on-chip ECC module before they are stored. In our implementation, we divide 144 B FBP data into three parts of 48 B, and each divided 48 B is encoded into 474-bit codeword so that up to 10 bit errors can be corrected by the ECC module. The total overhead of FBP data, therefore, is 0.96% of a 18-KB WL. Three on-chip modules of `TailCut` also incur an extra overhead on the die area. However, their area overhead is almost negligible. For example, the FBP engine module is a simple finite state machine, which can be implemented a few counters. For the FBP buffer, the total of 432 B is needed because each page requires 144 B for its FBP. Roughly, 432 B can take about 346um^2 in the 22-nm technology [6]. For an ECC engine, we employ a fully parallel BCH engine with 474-bit codewords which can correct up to 10-bit errors. This ECC engine is estimated to take about 0.08 mm^2 in the 22-nm technology [7]. Overall, the total area overhead of `TailCut` is not significant over a typical die size for the on-chip peripheral circuit and core logic of a 3D flash chip, which is about 30 mm^2.

5 EXPERIMENTAL RESULT

Using a widely-used SSD emulation platform [3], we construct a weak pattern-aware SSD, called `TcSSD`. Specifically, we model it to capture (i) the behavior of lateral charge spreading, (ii) the probabilistic read-retry operations based on our characterization results. We evaluate our `TcSSD` against a conventional SSD, called `Baseline`, where all the possible patterns are almost uniformly observed due to its data randomization.

For evaluation of contemporary 3D TLC-based SSDs, we configured 128 GiB SSDs for `Baseline` and `TcSSD`. The flash timing values are as follows: its program time and read time are set to (1980+44)



(a) Comparisons of normalized response time for five I/O workloads.

(b) Comparisons of normalized IOPS's for five I/O workloads.

(c) Comparisons of lifetime for four real I/O workloads.

Figure 10: Comparisons of the performance and lifetime under different workloads.

us and 90 us, respectively, while the data transfer rate is set to 1 Gbps. The employed ECC engine (Off-chip ECC engine) is assumed to have a capability of correcting 60 bit error per 1 KiB codeword. We employ two groups of workloads: five (*OLTP*, *Mail*, *File*, *Proxy*, and *Web*) from Filebench benchmark tool and the other four real workloads (*IO-ZONE*, *CELLO99*, *Postmark*, and *JEDES-Client*).

Figure 10(a) shows the average response times of `TcSSD`, which are normalized to those of `Baseline`, under two different device conditions: 6 months retention after 1K P/E cycles and 12 months retention after 2K P/E cycles. Compared to `Baseline`, `TcSSD` can improve the average response times by 13% and 21%, on average. We made the following observations: (i) such enhancements originate from the reduced number of read retries; (ii) `TailCut` becomes more effective as data retention time increases and thus the lateral charge spreading becomes more problematic.

To investigate the performance overhead brought by `TailCut`, we measured IOPS of `TcSSD` under a pristine device state. Figure 10(b) plots the IOPS values of `TcSSD`, which are normalized to those of `Baseline`, under such a state. The IOPS of `TcSSD` decreases at most by 3.5%, compared to `Baseline`.

We also measured the days for which a device can endure, as a proxy of the SSD lifetime, when each of the workloads is repeatedly executed. A device comes to the end of its life when the number of bit errors exceeds the error correction capability of the employed ECC engine. Figure 10(c) compares the expected device lifetime of the two tested SSDs; `TcSSD` can extend the device lifetime by 80%, on average, compared to `Baseline`. Specifically, when executing *IO-ZONE*, `TcSSD` can last for up to 225 days while `Baseline` comes to the end of its life in 125 days.

6 RELATED WORK

In fact, a group of works attempted to characterize the lateral charge spreading [1, 2]. For example, Mizoguchi *et al.* [1] revealed the consequence of LCS in 3D flash devices by investigating the amount of V_{th} shift. Luo *et al.* [2] unveiled that the key cause of the large amount of early retention loss in 3D flash devices is the LCS. However, most of them mainly focused on the device level characterization and did not quantitatively analyze the impact of LCS on the system performance/reliability.

A large body of works proposed to optimize the read retry. For example, Park *et al.* [8] achieved such a goal by compromising the reliability of each retry operation. Shim *et al.* [9] leveraged the process similarity in 3D NAND devices and proposed an accurate read reference voltage tracking technique. In contrast, our `TailCut` eliminates long tails of V_{th} distributions originated from LCS, and thus, substantially reduces the number of read retries. Since `TailCut` is orthogonal to the aforementioned techniques, it can be combined with them to further improve the performance and reliability of SSDs.

7 CONCLUSION

We have presented a novel state encoding technique, `TailCut`, that efficiently resolves the lateral charge spreading problem in 3D flash memory. Motivated by an extensive flash characterization study, `TailCut` exploits our key observation that a large portion of bit errors from lateral charge spreading are caused by a small number of weak state patterns. By removing these weak patterns by a pattern-aware state encoding scheme, `TailCut` can significantly improve both the SSD lifetime and read latency. Our experimental results show that `TcSSD` can reduce the read latency by up to 25% while increasing the SSD lifetime by 80%. Furthermore, we presented an efficient implementation of `TailCut` in a conventional SSD architecture so that little overhead in the performance and capacity is incurred by `TailCut`.

ACKNOWLEDGMENTS

This work was supported by Samsung Research Funding Incubation Center of Samsung Electronics, Republic of Korea, under Project Number SRFC-IT2002-06, and by Samsung Electronics Co., Ltd. (IO201207-07809-01). The ICT at Seoul National University provided research facilities for this study. This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (NRF-2021R1G1A1094835). (Corresponding Author: Jihong Kim)

REFERENCES

[1] K. Mizoguchi *et al.*, "Lateral charge migration suppression of 3d-nand flash by vth nearing for near data computing," in *Proceedings of the IEEE International Electron Devices Meeting (IEDM)*, 2017.

[2] Y. Luo *et al.*, "Improving 3d nand flash memory lifetime by tolerating early retention loss and process variation," *ACM on Measurement and Analysis of Computing Systems*, vol. 2, no. 3, pp. 1–48, 2018.

[3] H. Li *et al.*, "The case of femu: Cheap, accurate, scalable and extensible flash emulator.," in *Proceedings of the USENIX Conference on File and Storage Technologies (FAST)*, 2018.

[4] R. Micheloni *et al.*, "Inside nand flash memories," Springer Science & Business Media, 2010.

[5] C. Kim *et al.*, "A 21 nm High Performance 64 Gb MLC NAND Flash Memory With 400 MB/s Asynchronous Toggle DDR Interface," *IEEE J. Solid-State Circuits*, vol. 47, no. 4, pp. 981–989, 2012.

[6] B. Haran *et al.*, "22 nm technology compatible fully functional 0.1 m2 6t-sram cell," in *Proceedings of the IEEE International Electron Devices Meeting (IEDM)*, 2008.

[7] D. Strukov, "The area and latency tradeoffs of binary bit-parallel bch decoders for prospective nanoelectronic memories," in *Proceedings of the Asilomar Conference on Signals, Systems Computers (ASILOMAR)*, 2006.

[8] J. Park *et al.*, "Reducing solid-state drive read latency by optimizing read-retry," in *Proceedings of the ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2021.

[9] Y. Shim *et al.*, "Exploiting process similarity of 3d flash memory for high performance ssds," in *Proceedings of the IEEE/ACM International Symposium on Microarchitecture (MICRO)*, 2019.