Page Type-Aware Data Migration Technique for Read Disturb Management of NAND Flash Memory

Sangwoo Han, Minjung Cho, Gi Lee, and Eui-Young Chung

Abstract— **Read disturb is a phenomenon in which the cells of all unselected wordlines (WLs) within a block are soft programed while reading a selected WL, which causes errors. To solve the problem of read disturb, *read reclaim* (RR) operations, which migrate deteriorated block data to a new free block, are executed when the read count of the deteriorated block exceeds a preset threshold. RR requires additional operations, such as error correction, reprogramming, block erasure, and occupying I/O. So as the number of RR operations increases, the performance and reliability of the memory degrade. We observe that read disturb affects pages within a block differently according to page types. With this perspective, we propose Page Type-aware Data Migration technique (PTDM) technique, which migrates pages within blocks according to the vulnerability to read disturb. Shifting RR from the conventional block level to the page level improves the efficiency of the memory by reducing data migration. Also, we propose distributing read-hot data into several blocks during migration to reduce read disturb impact. Our experimental results show that the proposed method can reduce the amount of RR data migration and the overall erasure counts by 39.53% and 21.69%, respectively.**

Index Terms— **Data storage system, NAND flash memory (NFM), read disturb, read disturb management, read reclaim (RR).**

I. Introduction

NAND flash memory (NFM) is widely used in various computer systems, from mobile to servers. To reduce the costs and increase the capacity of NFM, one approach is to store multiple bits in one cell, such as a multilevel cell (MLC) and a triple-level cell (TLC). This approach increases the number of pages per block, which can increase the memory capacity but degrade its reliability. The main reason for the degraded reliability is *read disturb*, which refers to the unintentional injection of electrons into unrelated cells during a read operation, which causes read errors [1]. Read operations can access the same page many times, meaning that the number of read disturbs occurring within a block is unlimited. Thus, as the number of pages per block increases, the read disturb increases exponentially. In addition, read-intensive workloads on NFM continue to increase, which further increases the impact of read disturb.

To resolve the read disturb issue, the NAND controller executes the *read reclaim* (RR) operation, which migrates the disturbed data to a new free block before the data become unreadable. The conventional method is to count the number of read operations for each block and migrate blocks whose read count exceeds a preset threshold [2]. As noted above, read disturb becomes severe as the number of pages per block increases. Thus, the conventional method causes frequent RR operations, which reduce the system performance by increasing additional operations, such as data migration and block erasure, and degrade reliability by increasing program/erase (P/E) cycles.

Manuscript received 15 September 2022; revised 17 November 2022 and 29 December 2022; accepted 14 January 2023. This work was supported in part by the Ministry of Trade, Industry, and Energy (MOTIE) under Grant 1415181094; in part by the Korea Semiconductor Research Consortium (KSRC) Support Program for the Development of the Future Semiconductor Device under Grant 20019456; and in part by Samsung Electronics Company Ltd., Hwaseong, South Korea. The EDA tool was supported by the IC Design Education Center (IDEC), Daejeon, South Korea. *(Corresponding author: Eui-Young Chung.)*

The authors are with the School of Electrical and Electronic Engineering, Yonsei University, Seoul 03722, South Korea (e-mail: swhan0330@yonsei.ac.kr; rati77@yonsei.ac.kr; leegi01713@yonsei.ac.kr; eychung@yonsei.ac.kr).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TVLSI.2023.3240172.

Digital Object Identifier 10.1109/TVLSI.2023.3240172

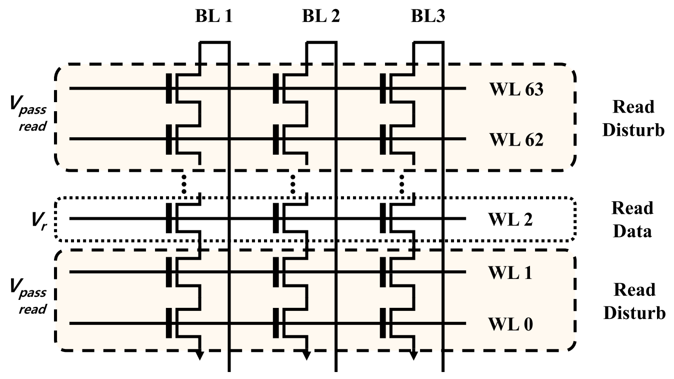

Fig. 1. Voltage settings during the read operation.

The least significant bit (LSB), central significant bit (CSB), and the most significant bit (MSB) pages within a wordline (WL) have different degrees of deterioration caused by read disturb. Some research suggested a method to reduce error by setting reference voltage differently according to the page type when reading a page [3]. However, the read-disturb errors still occur, and RR must be executed. We focus on making RR more efficient. In this brief, we propose a Page Type-aware Data Migration (PTDM) that executes RR more efficiently by migrating only the data of page types that are expected to have severe errors. By shifting RR management from the block level to the page level, the proposed PTDM effectively reduces the number of data migrations. Additionally, we propose distributing read-hot data into several blocks to relieve the read disturb.

By migrating only the data that are considered severe errors, our scheme can reduce the amount of RR data migration by 39.53%. The rest of this brief is organized as follows. Section II introduces the read disturb and techniques for mitigating the read disturb problem and describes the limitations of conventional technologies. Section III explains the proposed PTDM, and Section IV reports the experimental results using benchmark traces. Section V summarizes our findings and discusses future extensions of the proposed method.

II. Background and Related Works

A. Read Disturb in NFM

Fig. 1 shows the voltage settings of each WL during a read operation. The NAND controller applies V_r to the WL of the cell that wants to be read. Because V_r is the median value for the threshold voltage of a cell state, the cell turns on or off depending on the cell

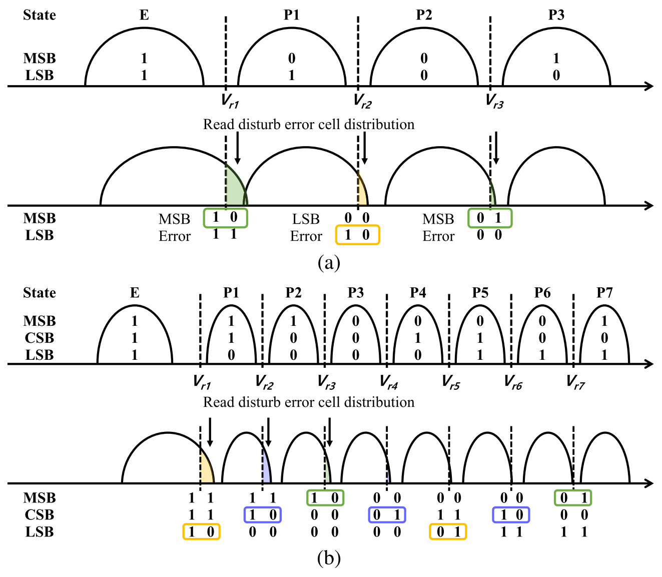

Fig. 2. Threshold voltage distribution of normal cell and deteriorated cell: (a) MLC and (b) TLC.

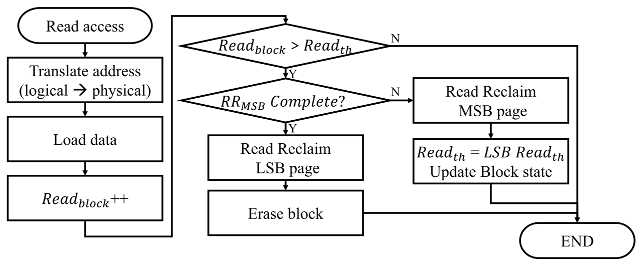

Fig. 3. Flowchart for the read request.

state. By sensing whether the cell turns on, the controller reads the data in the cell. Because the bit lines are connected in series, the cells of the unselected WLs should not affect the sensing. Therefore, the NAND controller applies a pass-through voltage, called V_P, to all unselected WLs in the block. V_P is less than the program voltage but high enough for soft programming. This soft programming injects electrons into a cell, which may later cause a read error. Read disturb affects the entire block.

MLC techniques are to store two or more pages in one wordline, which increases the number of pages per block. Increasing the number of pages per block increases the number of read operations in a block. Thus, the MLC techniques reinforce the impact of read disturb.

Fig. 2(a) and (b) shows the cell distributions before (top) and after (bottom) deterioration by read disturb in an MLC NFM and TLC NFM, respectively. The lowest E state has the worst deterioration due to read disturb, and the highest state P3 has the least deterioration [4]. To read data stored on the LSB page, a cell is sensed by applying V_{r2} to WL, and data stored on the MSB page are read by applying V_{r1} and V_{r3} to WL. Because the deterioration due to read disturb is severe in the E state, numerous errors occur in V_{r1}, and few errors occur in V_{r2} and V_{r3}. Errors in V_{r1} cause MSB data 1 to be misread as 0. Similarly, errors in V_{r2} and V_{r3} cause the LSB and MSB data, respectively, to be misread. Therefore, MSB pages are more vulnerable to read disturb than LSB pages. Some studies reported that LSB pages have 1.6 times more endurance to read disturb than MSB pages [5]. Similarly, according to the TLC bit mapping, it is vulnerable to read disturb in the order of LSB, CSB, and MSB.

B. Techniques for Mitigating the Read Disturb

To mitigate read disturb, the flash translation layer (FTL) executes RR. FTL is an intermediate system that controls the gap between file systems and NFM. RR is a technique that reads data before read failure occurs due to data deterioration, modifies it through error correction codes (ECC), and reprograms it into a new block. ECC operates and corrects errors whenever data is read through FTL due to garbage collection (GC), RR, or user requests. Conventional RR counts the number of reads per block; when the read count exceeds a preset threshold, then all the valid data within the block are migrated to a new free block. After migrating all the valid data within the deteriorated block, FTL erases the block. Vendor preset this threshold value conservatively, considering P/E cycle or physical issues such as process variation. RR can correct errors completely, but the additional processes can degrade the performance. Some studies have shown that frequent RR degrades the performance by adding extra processes, such as data migration and block erasures, and the reliability decreases because of the increased number of P/E cycles [6]. Thus, RR should be more efficient in a situation with increasing read-disturb impacts.

For efficient RR, some studies proposed making strong blocks against read disturb and migrating read-intensive data to strong blocks [5], [7]. Other studies suggested mixing writing data caused by RR and user requests, to reduce second RR operation [8]. As mentioned in Section II-A, pages have different error endurance against read disturb, depending on the page types. However, previous methods did not consider these characteristics and execute RR at the block level. These methods are still inefficient since they migrate the entire block simultaneously, even though some page types can perform more reads. Therefore, this brief proposes a method that divides and migrates blocks according to the page types.

III. Proposed Method

The proposed PTDM management is implemented on FTL. PTDM comprises a block read counter, page classifier, migration manager, and NAND device information table. The block read counter stores the read counts of each block. The page classifier divides a block into several groups according to the page types. The migration manager determines whether each page needs an RR, manages data migration, and processes requests for data during migration. The NAND information table stores information for performing PTDM.

A. Page Type-Aware RR

As mentioned in Section II-A, even with the same amount of read disturb, the error rates of each page are different depending on the page type. Therefore, the threshold read counts of each page type are different. These threshold read counts can be obtained by a formula or measurement [4], [5]. PTDM stores the preset threshold read count of each page type differently. The page classifier distinguishes page types according to the physical page address.

When the read count of a block reaches the MSB threshold read count, the PTDM migrates only MSB pages within the block. When migrating the MSB pages is complete, the PTDM RR manager updates the threshold read count of that block to the LSB threshold read count. Afterward, when the read count of the block reaches the LSB threshold read count, the PTDM migrates the remaining LSB pages of the block and then executes block erasure.

Fig. 3 shows the flowchart for PTDM when read operations are accessed. $\text{Read}_{\text{block}}$ is the read count of a block. And Read_{th} is the threshold read count, whose initial value is the MSB threshold read count. In the case of TLC, RR is performed according to the page types in the same way.

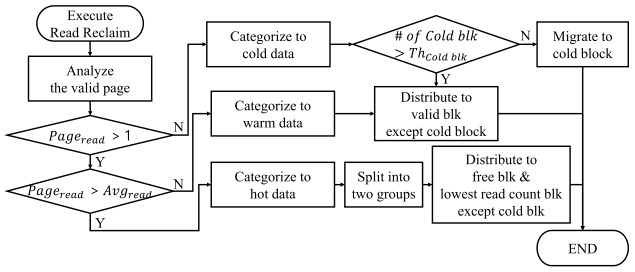

Fig. 4. Flowchart for the data distribution technique.

B. Data Distribution Technique for Mitigating Read Disturb

The fewer read counts within a block, the less impact the read disturb has. To mitigate the read disturb, we propose distributing read-intensive data in several blocks when migrating the data. We classified the data to be migrated into three categories: cold, warm, and hot. The cold data whose read count is zero or one until execute RR are migrated to a cold block. RR will hardly be executed on these cold blocks. The hot data whose read count is more than the average read count is split into two groups and migrated to a free block and a block with the lowest read count among valid blocks, except for the cold blocks. The remaining warm data are distributed to a valid block, except for the cold blocks. When distributing the data, the RR manager considers the data's read count and the target block's total read count.

If the number of cold blocks increases due to a large amount of cold data, hot data may be concentrated into a few blocks. Then the RR operation may increase. Therefore, we limit the number of cold blocks, and if the number of cold blocks exceeds this threshold value, cold data are also distributed to the valid block. Also, GC may occur frequently due to insufficient free blocks while distributing the data to several blocks. Therefore, this technique operates when free blocks are sufficient.

Fig. 4 shows our distribution technique. $\text{Page}_{\text{read}}$ and Avg_{read} are the read count of the page and the average read count of the block, respectively. $\text{Th}_{\text{cold blk}}$ is the threshold value of the number of cold blocks, which we set as 10% of total blocks according to our experiment [9].

C. Overheads

PTDM has two major overheads: side effects resulting from granular data migration and storage overheads.

Block erasure can only be executed when all data within the block are migrated. For example, in MLC NAND, block erasure can be executed after the migration of LSB pages. If RR is executed only on MSB pages in several blocks when there are not enough free blocks, GC may occur frequently to create a free block. However, RR also executes erase on a block. Since the PTDM reduces the amount of RR significantly, the erasure caused by RR is much decreased when applying PTDM. So, the overall erasure count was reduced. Also, the number of migrated pages per GC is decreased, which can dissipate the data migration overhead. This overhead is explained in Section IV-C through experiments.

The storage overhead for PTDM is not expected to be significant. Even with conventional RR, the read counts are stored for each block. Thus, additional values to be stored are the RR status of each block and the cold block information. In the NAND configuration in our experiments, the total storage overhead per chip is 85 kB for 3-D and 54 kB for 2-D. This storage overhead is less than 1% of conventional solid disk drive (SSD) metadata.

TABLE I
CONFIGURATION OF NFM

Parameter	MLC	TLC
t_{PROG} (LSB / MSB)	330 us / 950 us	700 us
t_R(LSB / MSB)	5 us / 42 us	60 us
t_{BERS}	3.5 ms	3.5 ms
Chip size	128 Gb	512 Gb
Page size	16 KB	16 KB
Number of Pages / block	300	768
Preset read count (1^{st} / 2^{nd} /3^{rd})	(100k / 160k / -)	(10k / 12k / 13k)

TABLE II
TRACE INFORMATION

Trace	Num. of Req. (k)	Read ratio (%)	Avg. Req. length (sector)		Random ratio (%)	
			Read	Write	Read	Write
websearch1	1055	99.9	30.29	17.2	0	0
websearch3	4262	99.9	30.81	52.96	0	0
TPC_C	349	62.9	16.24	18.28	0.1	0.1
financial	3699	82.3	4.55	5.83	92.85	87.45
ads	1532	90.5	63	14.07	6.04	0.1

IV. EVALUATION

A. Experimental Setup

We evaluated the proposed PTDM scheme by implementing a trace-driven simulator [10], [11], which includes a page-level FTL [12] and NAND flash memories. Table I presents the timing parameters and configuration of the NFM [13], [14]. An MLC and TLC block typically endure 100k and 10k page read operations, respectively [5], [7]. Block endurance is determined by the worst page group. Thus, we set the first preset threshold read counts for MLC and TLC as 100k and 10k, respectively. The threshold read counts for other page types were also set by referring to [5] and [15]. As mentioned in Section II-B, vendors provide fixed threshold read count. Thus, we experimented with a fixed threshold read count. If vendors provide data about the deterioration due to read disturb according to the P/E cycle, we can easily apply changing threshold values to our method. Some research studies report different endurance disparities between strong and weak page types [16], [17], so we also evaluated PTDM with various threshold read counts. Table II presents the trace files used in the experiment and their characteristics [18], [19], [20].

To evaluate the effectiveness of PTDM, we compared it with three methods: conventional RR (baseline) [2], PTDM, and D-PTDM. The baseline migrates an entire block when the read count of the block reaches the threshold read count. PTDM is a technique explained in Section III-A, and D-PTDM is a technique in Sections III-B and III-A.

B. Efficiency of RR

Figs. 5 and 6 show the number of migrated pages during RR execution. On average, in MLC, PTDM reduced the migration data size by 41% compared to the baseline because PTDM only migrated pages that were expected to have a severe error rate, not the entire block. In TLC, PTDM reduced the migration data size by 37% compared to the baseline. Furthermore, by distributing read-intensive data, D-PTDM performs less RR than PTDM.

In Fig. 5, the number of migrations of LSB pages is particularly small in the "financial" benchmark. This is because the NAND controller executes GC before RR. "Financial" has a small request size and high randomness, and thus, GC occurs frequently. During

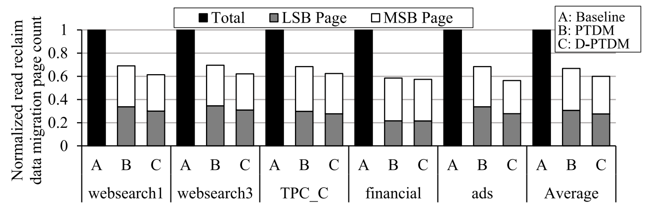

Fig. 5. Normalized amount of data migration while RR at MLC.

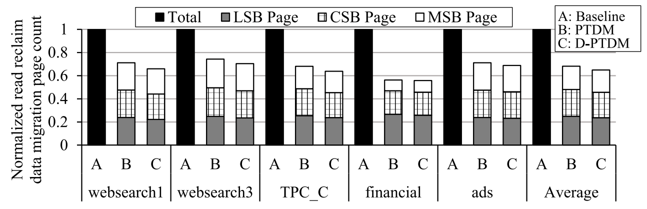

Fig. 6. Normalized amount of data migration while RR at TLC.

GC, the garbage collector selects blocks with many invalid pages and migrates the data in them. When the RR manager migrates some page types within the blocks, the blocks have many invalid pages and, thus, are more likely to be selected by the garbage collector. Therefore, in such a benchmark, page types that are more resistant to read disturb are migrated more with GC, not RR. This is why the number of LSB pages in MLC and MSB pages in TLC migrated by RR is small, as shown in Figs. 5 and 6. Likewise, in the "financial" benchmark, the free blocks are insufficient, so the effect of distributing was not significant.

Among the experimental benchmarks, "websearch1" and "websearch3" had almost no write operations; thus, GC was hardly executed. So, the amount of data migration due to RR is similarly reduced in LSB and MSB pages. PTDM RR reduces the amount of data migration not only for read disturb tolerant pages (LSB in MLC, CSB, and MSB in TLC) but also for weak pages. This is because the data migrated by RR are distributed to several blocks. This reduces the number of read operations per block, thereby reducing the overall number of RR operations and overall migrated pages. Since "ads" and "TPC_C" had less random write access, less GC was executed, resulting in similar results.

Figs. 7 and 8 show the number of migrated pages during RR execution while changing the second and third threshold read count. The number in parentheses on the x-axis of the graph indicates the second or third preset threshold read count. For example, (1, 1.3) in Fig. 7 means the first preset threshold read count and second preset threshold read count are 100k and 130k, respectively. Similarly, (1, 1.5, 2.0) in Fig. 8 means that the first, second, and third preset threshold read count in TLC is 10k, 15k, and 20k, respectively.

The results show that the larger the endurance disparities between each page type, the less amount of data migration during RR execution. For a fair experiment, we experimented with various second or third threshold read counts while keeping these values from exceeding twice the first threshold value. In an extreme case, such as (1, 4, 8) in TLC, the execution of GCs increases, but the amount of data migration is slightly reduced. The detailed results are provided in [9].

In Fig. 7, in the case of "TPC_C" benchmark, the migration amount of the LSB pages was significantly reduced in (1, 1.8) and (1, 2.0). Due to the characteristics of "TPC_C," RR was not executed on the LSB page because the data was no longer accessed after a specific read count.

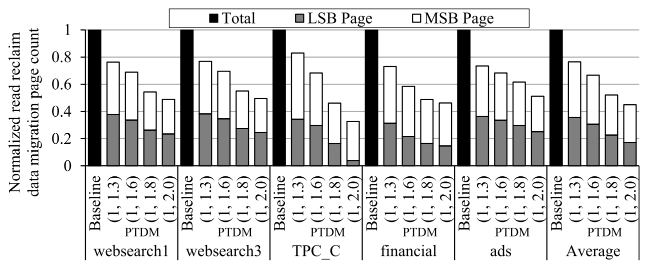

Fig. 7. Normalized amount of data migration while varying preset threshold read count at MLC.

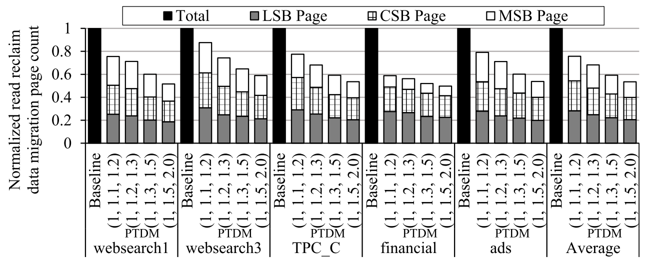

Fig. 8. Normalized amount of data migration while varying preset threshold read count at TLC.

C. Overhead Analysis

As stated earlier, PTDM has an overhead in increasing the number of GCs. Fig. 9 shows the overall erase counts, including erasure induced by GC and RR, while executing the benchmark in MLC. In the case of "financial," where free blocks are insufficient, the number of GCs increased compared to the baseline. However, the amount of reduction in RR executions is bigger than the amount of increase in GCs when applying PTDM. As a result, the overall number of erasures is reduced. In conclusion, when applying PTDM, even though free blocks are insufficient, and GC increases as a result, the efficiency of reducing RRs overcomes the overhead induced by GCs. The result shows that PTDM decreases overall erasure count by 21.7%, compared to the baseline. Similar results can be obtained for TLC. The result was similar when the trace was repeated until the NAND was worn out [9]. Furthermore, the number of migrating pages per GC is reduced, which can dissipate the data migration overhead [9].

D. Complementary to Other Techniques

Most previous research studies proposed granularity RRs, which divide blocks with hot/cold data and migrate data during RR to make RR more efficient [5], [7]. These methods create strong blocks against read disturb and migrate read-hot data to these strong blocks. They create the strong block by changing V_r [5] or not using the MSB page [7]. However, these strong blocks still have different endurance according to the page type, so threshold read counts should differ for each page type. Therefore, PTDM can be easily applied.

Fig. 10 shows the overall erasure count and the amount of overall data migration when applying PTDM to other techniques. We averaged the five traces and showed them. Results show that overall data migration and erase counts are reduced by 5.9% and 6.2%, respectively, compared to the case when PTDM is not applied. It shows PTDM and other methods can be complementary. The detailed results are provided in [9].

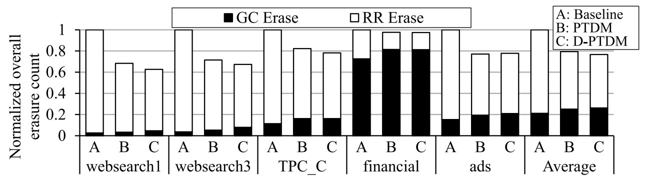

Fig. 9. Normalized overall erase count at MLC.

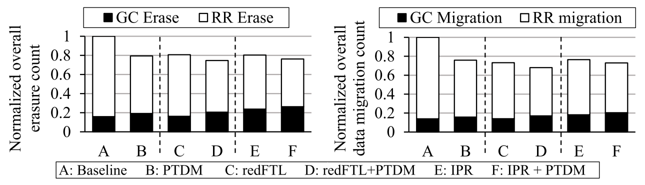

Fig. 10. Normalized number of overall erasure counts and overall data migration when applying `PTDM` to other techniques.

V. Conclusion

Recently, increasing read-intensive workload and the number of pages per block have increased the impact of read disturb in NFM. Conventional RR is inefficient in these circumstances as it reduces the performance and reliability of the memory. Most RR methods do not consider that the impact of read disturb differs based on the page types, even when they are in the same block. Therefore, we propose a new management method that classifies pages within one block into several groups and migrates each based on page types. In the experimental results, `PTDM` reduced the overall number of data migrations by 39.53% compared to that of the `baseline`. In addition, the erasure count was reduced by 21.7%.

References

[1] M. Kang et al., "Improving read disturb characteristics by self-boosting read scheme for multilevel NAND flash memories," *Jpn. J. Appl. Phys.*, vol. 48, no. 4, Apr. 2009, Art. no. 04C062.

[2] H. H. Frost, C. J. Camp, T. J. Fisher, J. A. Fuxa, and L. W. Shelton, "Efficient reduction of read disturb errors in nand flash memory," U.S. Patent 7 818 525, Oct. 19, 2010.

[3] P. Kumari, U. Surendranathan, M. Wasiolek, K. Hattar, N. P. Bhat, and B. Ray, "Radiation-induced error mitigation by read-retry technique for MLC 3-D NAND flash memory," *IEEE Trans. Nucl. Sci.*, vol. 68, no. 5, pp. 1032–1039, May 2021.

[4] Y. Cai, Y. Luo, S. Ghose, and O. Mutlu, "Read disturb errors in MLC NAND flash memory: Characterization, mitigation, and recovery," in *Proc. 45th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw.*, Jun. 2015, pp. 438–449.

[5] K. Ha, J. Jeong, and J. Kim, "An integrated approach for managing read disturbs in high-density NAND flash memory," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 35, no. 7, pp. 1079–1091, Jul. 2016.

[6] M. Jung and M. Kandemir, "Revisiting widely held SSD expectations and rethinking system-level implications," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 41, no. 1, pp. 203–216, Jun. 2013.

[7] T.-C. Wu, Y.-P. Ma, and L.-P. Chang, "Flash read disturb management using adaptive cell bit-density with in-place reprogramming," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2018, pp. 325–330.

[8] G. Zhang, Y. Deng, Y. Zhou, S. Pang, J. Yue, and Y. Zhu, "Cocktail: Mixing data with different characteristics to reduce read reclaims for NAND flash memory," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, early access, Oct. 14, 2022, doi: 10.1109/TCAD.2022.3214679.

[9] (2022). *Supplementary Data*. [Online]. Available: https://dtl.yonsei.ac.kr/docs/PTDMsuppl.pdf

[10] S.-H. Park, D.-G. Kim, K. Bang, H.-J. Lee, S. Yoo, and E.-Y. Chung, "An adaptive idle-time exploiting method for low latency NAND flash-based storage devices," *IEEE Trans. Comput.*, vol. 63, no. 5, pp. 1085–1096, May 2014.

[11] N. Mi, A. Riska, Q. Zhang, E. Smirni, and E. Riedel, "Efficient management of idleness in storage systems," *ACM Trans. Storage*, vol. 5, no. 2, pp. 1–25, Jun. 2009.

[12] H.-J. Kim and S.-G. Lee, "A new flash memory management for flash storage system," in *Proc. 23rd Annu. Int. Comput. Softw. Appl. Conf.*, Oct. 1999, p. 284.

[13] S. Lee et al., "7.5 A 128Gb 2b/cell NAND flash memory in 14nm technology with tPROG= 640μs and 800MB/s I/O rate," in *IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers*, Jan. 2016, pp. 138–139.

[14] C. Kim et al., "A 512-GB 3-b/cell 64-stacked WL 3-d-NAND flash memory," *IEEE J. Solid-State Circuits*, vol. 53, no. 1, pp. 124–133, Jan. 2018.

[15] W. Liu et al., "Characterization summary of performance, reliability, and threshold voltage distribution of 3D charge-trap NAND flash memory," *ACM Trans. Storage*, vol. 18, no. 2, pp. 1–25, May 2022.

[16] Y. Cai, S. Ghose, Y. Luo, K. Mai, O. Mutlu, and E. F. Haratsch, "Vulnerabilities in MLC NAND flash memory programming: Experimental analysis, exploits, and mitigation techniques," in *Proc. IEEE Int. Symp. High Perform. Comput. Archit. (HPCA)*, Feb. 2017, pp. 49–60.

[17] A. Kobayashi, H. Watanabe, Y. Sakaki, S. Aritome, and K. Takeuchi, "Investigation of read disturb error in 1Ynm NAND flash memories for system level solution," in *Proc. IEEE Int. Rel. Phys. Symp. (IRPS)*, Apr. 2017, p. 6.

[18] O. Application, "Umass trace repository," Tech. Rep., 2007. [Online]. Available: http://traces.cs.umass.edu/index.php/Storage/Storage

[19] S. Kavalanekar and B. Worthington, "Microsoft enterprise traces (SNIA IOTTA trace 146)," in *SNIA IOTTA Trace Repository*, G. Kuenning, Ed. Storage Networking Industry Association, Oct. 2007. [Online]. Available: http://iotta.snia.org/traces/block-io/131?only=146

[20] V. Sharda, S. Kavalanekar, and B. Worthington, "Microsoft production server traces (SNIA IOTTA trace 176)," in *SNIA IOTTA Trace Repository*, G. Kuenning, Ed. Storage Networking Industry Association, Mar. 2008. [Online]. Available: http://iotta.snia.org/traces/block-io/158?only=176