DWR: Differential Wearing for Read Performance Optimization on High-Density NAND Flash Memory

Yunpeng Song[2], Qiao Li[4], Yina Lv[2], Changlong Li[2], ✉Liang Shi[1,2,3]
[1]Software/Hardware Co-design Engineering Research Center, Ministry of Education, Shanghai, China
[2]School of Computer Science and Technology, East China Normal University, Shanghai, China
[3]Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan, Hubei, China
[4]Department of Computer Science, City University of Hong Kong, Hong Kong, China

***Abstract*—With the cost reduction and density optimization, the read performance and lifetime of high-density NAND flash memory have been significantly degraded during the last decade. Previous works proposed to optimize lifetime with wear leveling and optimize read performance with reliability improvement. However, with wearing, the reliability and read performance will be degraded along with the life of the device. To solve this problem, a differential wearing scheme (DWR) is proposed to optimize the read performance. The basic idea of DWR is to partition the flash memory into two areas and wear them at different speeds. For the area with low wearing speed, read operations are scheduled for read performance optimization. For the area with high wearing speed, write operations are scheduled but designed to avoid generating bad blocks early. Through careful design and real workloads evaluation on 3D TLC NAND flash, DWR achieves encouraging read performance optimization with negligible impacts to the lifetime.**

I. Introduction

With cost reduction and density optimization, 3D NAND flash memory has been widely adopted as storage systems in personal computers and data centers. However, two critical issues holding back their further development, which are lifetime and performance degradation. To solve these issues, previous works proposed to use wear leveling for lifetime optimization [1], [10], [15]. Their basic idea is to wear the memory evenly to maximize the lifetime. However, with the device wearing, the reliability is degraded [11]. For example, for the state-of-the-art high-density 3D NAND flash memory, the maximal number of program/erase (P/E) cycles is around 1000 [7]. With the reliability degradation, the read performance will be significantly impacted. Previous work presented that high-density 3D NAND flash adopts low-density-parity-check code (LDPC) as error correction code (ECC) to optimize reliability [16]. One critical characteristic of LDPC is that its access performance is highly correlated with the reliability of data [6]. For data with low reliability, it needs a long latency to retry the data and vise versa. This work will propose a scheme to reduce the read latency induced by device wearing.

The current wear leveling scheme on flash memory is proposed to wear the flash evenly. However, the read performance will be degraded with the wearing. Previous works proposed to optimize the LDPC of flash memory for read performance optimization [4]–[6], [13], [14]. LDPC is designed to correct data based on the sensed information from the memory cells. If the information is out of error correction capability, it needs to retry the information with different voltages. For example, Li et al. [6] proposed to determine the optimal sensing voltages of LDPC to reduce the number of retries. Others have proposed to optimize read performance by designing the data placement method [9], [12]. For example, Lv et al. [9] proposed to identify hot read data and place them at the high reliable pages of high-density flash memory. However, with the reliability degradation of flash memory, the read latency still increases, which cannot be solved by the above schemes.

Different from the above schemes, this work is the first on proposing a differential wearing scheme (DWR) for high-density 3D NAND flash memory. The basic idea is to physically partition the flash memory into two areas with different wearing speeds. First, one area is worn with a low speed and hot-read data are placed in the area, which is called low wearing area (LWA). Second, the other area is used for write operations. This will gather the write operations in this area, avoiding wearing the LWA. This area is called high wearing area (HWA). With this approach, the read performance can be optimized. However, there are several challenges. First, the data with different access characteristics should be identified to determine the right areas for them. Second, the lifetime should be well controlled since the area for write operations may be worn faster than the area for read operations. To solve these issues, this paper makes the following contributions:

- A data placement scheme is proposed to determine the areas for the data. The placement scheme is designed to schedule all the writes to HWA. For LWA, it will serve hot read data for read performance optimization.
- A low-cost data migration scheme is proposed to re-determine the placement of data. The migration scheme includes two parts: migration from HWA to LWA for hot read data and migration from LWA to HWA when the size of LWA is full.
- A lifetime optimization scheme is presented to avoid the prematurely worn-out of flash memory. Three techniques are presented, which cover two scenarios, consumer storage without device replacement, and data center storage with periodic device replacement.
- Through careful design and real workloads evaluation on 3D TLC NAND flash based simulator, experimental results show

This work is supported by the NSFC 62072177 and 61972154, Shanghai Science and Technology Project (20ZR1417200) and the Open Project Program of Wuhan National Laboratory for Optoelectronics NO. 2019WNLOKF008. The corresponding author is Liang Shi (shi.liang.hk@gmail.com).

978-3-9819263-6-1/DATE22/©2022 EDAA

that DWR can achieve significantly read performance optimization at most life stages with negligible lifetime impacts.

The rest of the paper is organized as follows. The background and research problem are presented in Section II. Section III presents the proposed DWR. Section IV presents the experimental results. Section V is the conclusion.

II. Background and Problem Statement

A. *Solid State Drives and Wear Leveling*

Figure 1 shows an architecture for flash based solid state drives (SSD). SSD usually consists of two parts, the flash chip array (FCA) and the flash translation layer (FTL). The flash chip is organized with blocks, where each block includes several pages. The read unit is a flash page, the program unit is a flash page or word-line for 3D NAND flash, and the erase unit is a flash block. The flash chip array provides basic read, program, and erase operations for the FTL. The FTL is provided by the vendor to control the underlying flash memory chip. The FTL typically contains three components, allocator, garbage collector and wear leveler, to overcome several limitations inherent to flash memory. At the FTL, the allocator is responsible for converting logical addresses to actual physical addresses. When there are too many invalid pages in flash memory, the garbage collector is triggered. It erases those blocks that contain invalid data and migrates the valid data in the victim blocks to other free blocks to reclaim the occupied space.

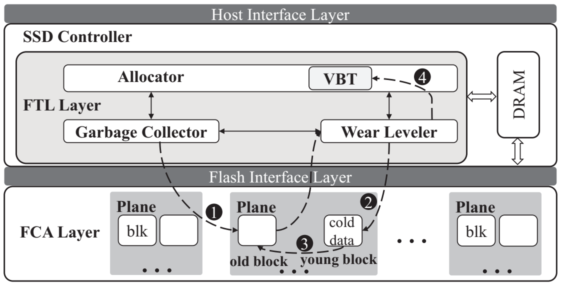


Fig. 1. A typical architecture of SSD.

Two types of wear leveling schemes have been popularly adopted in current devices, including dynamic and static wear leveling. Dynamic wear leveling (DWL) always selects the free block with the least number of P/E cycles for data allocation. Differently, static wear leveling (SWL) will further migrate infrequently updated data, distributing P/E cycles evenly among all blocks. Figure 1 shows the flow of SWL. ❶ SWL is triggered when some flash blocks are excessively erased. ❷ Then, the SWL algorithm selects blocks that have been erased less frequently and hold cold data as erase targets for wear leveling. ❸ Before erasing the target block, the valid data need to be migrated to the previously excessively erased flash block, thus preventing the old block from being used continuously. Meanwhile, after the target block is erased, it will be further used as a free block. ❹ Finally, the wear leveler needs to notify the allocator to update the information related to the virtual block address to the physical block address mapping table (VBT) of the migrated data. With this approach, the whole blocks can be worn evenly to achieve lifetime improvement.

B. *Read Operation on High-Density Flash Memory*

NAND flash memory cell uses floating gate (FG) or charge trap (CT) to store charges. Figure 2 shows the probability density function (PDF) of the threshold voltage distribution for a triple level cell (TLC) based NAND flash. The number of states S depends on the number of bits n stored in a memory cell, where S=2^n. In the TLC flash cell, there are 8 states and 3 bits in a cell, which are the most-significant bit (MSB), the center-significant bit (CSB), and the least-significant bit (LSB).

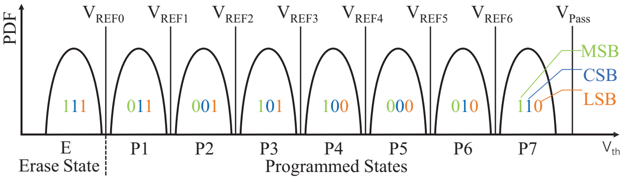


Fig. 2. V_{TH} distribution of 3D TLC flash cell.

As the number of P/E cycles increases, the reliability of flash memory cells gradually decreases, and the phenomenon of retention loss occurs as the data retention time increases [11]. To solve the reliability issue, high-density NAND flash uses the low-density-parity-check code (LDPC) as error correction code (ECC). LDPC is a kind of probability based ECC, which needs to acquire accurate information for data error corrections. When reading a page, one bit of information is extracted from the memory cell. The information is acquired by adding sensing voltages to the memory cell. The sensing voltages are different for the above three bits. Take Figure 2 as an example. The read operation needs to acquire the state information multiple times until LDPC can correct the errors. If the reliability is degraded, the number of sensing voltages will be significantly increased for LDPC. In this case, the read latency will be significantly increased when the reliability is degraded.

C. *Problem Statement*

From the above discussions, the read performance is highly correlated with the reliability of flash memory. Due to the rapidly decreasing endurance of 3D TLC NAND flash chips, the design of wear leveling has great improvement on the lifetime of flash. However, many state-of-the-art wear leveling strategies specify a fixed threshold for the number of P/E cycles of the blocks. Wear leveling strategies can achieve uniform block erasure over the endurance of the device, and result in a certain number of P/E cycles for all blocks, which leads to a simultaneous degradation of the reliability of the flash blocks. In this case, the above wear leveling scheme will induce the read performance degradation with the wearing of flash.

To understand the characteristics of the performance degradation with the wearing of flash, the read performance is evaluated under different wearing degrees. LDPC is adopted as the default ECC. Eight widely used workloads are selected to show their read performance along the wearings. Detailed configurations are presented in the experiment section. Figure 3 shows the normalized average read latency by wearing the whole flash with different wearing degrees, from 20% to 80%. With SWL, all the flash blocks have similar reliability. The results show

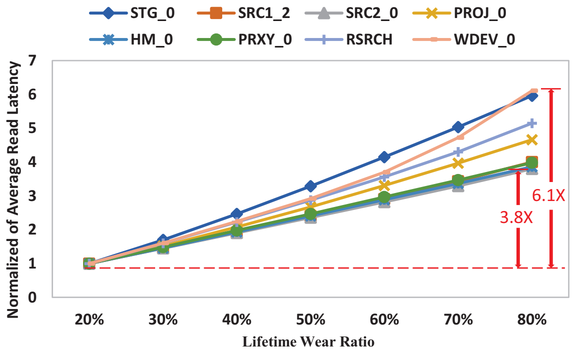


Fig. 3. Motivational results with the real-life workloads.

that the read latency is significantly increased along with the wearing by 3.8x to 6.1x. The results indicate that the reliability induced read performance degradation has been a critical issue. The above results motivate us to propose a new design, where not all the flash blocks are worn uniformly. With the design, we should achieve a similar lifetime to that of wear leveling, but significantly optimize the read performance of flash devices.

III. Differential Wearing Aware Read Performance Optimization

A. Overview

In this section, a differential wearing scheme (DWR) is designed to optimize the read performance of high-density NAND flash memory. Figure 4 shows the architecture of DWR, which is implemented as a module, namely, differential wear leveler, so as to prevent spending lots of effort on integrating the DWR design into different state-of-the-art FTL designs. The differential wear leveler contains three parts, data placement for the differential wearing, data migration for read performance optimization, and lifetime optimization. The data placement scheme is designed to partition the flash memory into a low wearing area (LWA) and a high wearing area (HWA). LWA is used to serve read operations for performance optimization. HWA is used to serve write operations, avoiding the wearing of LWA.

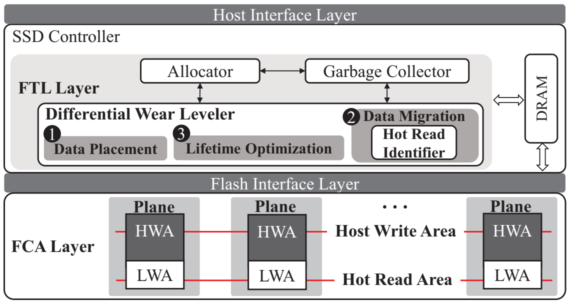


Fig. 4. Overview of differential wearing.

There are several challenges for the above design, which are listed as follows: 1) The traditional static wear leveling technology guarantees a simultaneous decrease in the reliability of the blocks. To achieve differential wearing, we need to design a new data placement scheme to form differential wearing. 2) Differential wearing creates HWA and LWA. Hot read data should be identified and migrated from HWA to LWA for read performance optimization. The data migration scheme should be efficient and low cost. 3) Differential wearing may accelerate the wearing of HWA. Under differential wearing, the lifetime of the flash memory needs to be safeguarded from being affected. In the following, three techniques are presented: First, a data placement scheme is presented to construct the differential wearing areas. Second, a data migration scheme is presented to identify hot read data and place them at LWA with low cost. Finally, a set of lifetime optimization schemes are presented to minimize the impact on the lifetime.

B. Data Placement for Differential Wearing

The data placement is designed to physically partition the flash memory into two areas. One is called low wearing area (LWA) and the other one is called high wearing area (HWA). During accesses, write operations are processed in HWA in priority. If data in HWA are frequently read, they will be migrated to LWA for read performance optimization. To avoid parallelism impact, these two areas are partitioned inside the plane, which is the last parallelism level of flash based storage devices. Any state-of-the-art wear leveling schemes can be adopted for these two areas for lifetime optimization. With this approach, LWA can be avoided from a large amount of writes induced wearing and the read performance of LWA can be well protected during the life of the device. For the above design, one critical issue is to determine the sizes of the two areas, S_{HWA} and S_{LWA}. On one hand, if S_{LWA} is too small, there will be little space for frequently read data. Then, the read performance cannot be well optimized. On the other hand, if S_{LWA} is too large, S_{HWA} will be small and the lifetime may be impacted. In the experiment, the sizes of these two areas will be characterized with careful studies.

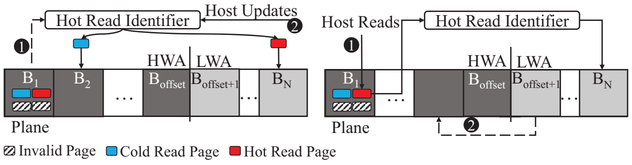
As shown in the bottom of Figure 4, each plane is physically partitioned into HWA and LWA. HWA is designed to process host writes and LWA is designed to process host reads. With this approach, LWA is protected for better read performance.

C. Data Migration for Hot Read Data

To optimize the read performance, one critical step is to place the hot read data at LWA. In this section, we propose a data migration scheme to improve read performance. First, to identify the hot read data, we propose a simple scheme to determine the migration from HWA to LWA. The basic idea is that if the data at HWA are read for N_R times and they are retried due to LDPC, they should be migrated to LWA for read performance optimization. The reason for the above settings is that only the data which are read multiple times and retried in HWA has an impact on the read performance. More importantly, with this approach, the migration cost will be small. For the design, one important parameter is N_R. For a large N_R, it may introduce limited read performance optimization, and with little migrations. But for a small N_R, it may introduce better read performance optimization, but with many migrations. Note that the migration is correlated with the reliability of flash. With the wearing of flash, N_R should be changed to achieve cost and performance trade-off.

When the reliability of HWA is good, N_R should be large to avoid migration. With the wearing of HWA, N_R is reduced to encourage the migration for read performance optimization. In the experiments, N_R will be discussed in detail. Second, the migration process is to reads and writes between HWA and LWA, which is high cost on high-density NAND flash. To optimize the cost, we propose to migrate the data as follows.

The first case is to migrate data with the internal mechanism of SSD and host data updates. As shown in Figure 5 (a), ❶ when garbage collection or wear leveling is activated in HWA, the data enrolled during these processes are identified and migrated to LWA. In addition, ❷ when the data is updated, it will be written to LWA if the data has been identified as hot read data. The second case is to migrate data actively. As shown in Figure 5 (b), ❶ if the data are accessed and satisfy the above conditions, they will be cached in the SSD controller first and migrate to LWA with a one-pass programming process. One case not considered above is that if LWA is full, the migration cannot be proceeded. To solve this issue, we propose to migrate one block of data at LWA back to HWA (❷ in Figure 5 (b)). Considering that there is a temporal locality in the hotness of data, a first-in-first-out scheme is adopted to select the earliest occupied block as the victim block.



(a) Migration with internal or host updates

(b) Migration with host read requests

Fig. 5. Data migration for hot read data.

D. Lifetime Optimization

In Section III-B, the size of HWA is critical to the lifetime of SSD. Because HWA is smaller than the whole storage device, issuing all writes to this area will speed up the wearing of this area and induce lifetime impact. To solve this issue, we propose several approaches to limit the impact on the lifetime. The first scheme is to limit the size of LWA. LWA is designed to store hot-read data. As we know, the percentage of hot-read data is always small. As presented in previous work from Li et al. and Lv et al [8], only 10% of data from real workloads are read over 4 times and only 5% of data are read over 8 times. From this observation, the size of LWA can be very small. The second scheme is to limit the upper bound of wearing to HWA. Even though HWA is worn with a faster speed due to a large amount of writes, the wearing to HWA is limited once it crosses a wearing ratio, such as 80% in this work. Once the limitation is approached, we propose to slow the wearing of HWA by leading more writes to LWA for lifetime optimization. The first write of data will be written to the HWA. When the data is updated, it will be written to the LWA. At this time, LWA can be used for serve more write operations. With this approach, LWA is worn with a faster speed to avoid the prematurely worn out of HWA. The above schemes can be adopted for consumer devices without device replacement. However, for data centers with device replacement, when HWA is worn with a high ratio, the lifetime of the device is mostly achieved. The device should be at its old stage. In this case, we recommend to use the device as a warm or old write device. Host-side writes should be redirected to other SSDs. Since the device still has an area with low wearing, it still can be used as a read device.

E. Implementation and Analysis

To implement DWR, the FTL of flash memory is updated as follows. First, for differential wearing, the flash plane is partitioned into two areas. In this work, we use the block offset to record the partition. As shown in Figure 4, each plane is physically partitioned, where the upper part is HWA and the bottom part is LWA. Then, we only need one variable to differentiate the areas. For data writes, we need to maintain two write points in HWA and LWA, respectively. Second, for hot read data identification, each page needs a counter to record the number of reads. The mapping table is extended with three bits to record them. During accesses, if write operations are issued, they are directed to HWA. If read operations are issued to HWA, the counter of the data page is checked for migration. If the counter is larger than N_R and retry happens, the migration will be activated. For the migration from LWA to HWA, we use a simple scheme to select the victim block. In this work, we record the last victim block index in the LWA. For example, the current index is x and the total number of blocks in a plane is N. The next index is $(x+1)\%(N\textit{-offset})$ and the block number of the next victim block is $(x+\textit{offset})\%N$. Since, only one variable is needed to represent the index for each plane. In conclusion, the memory and computation costs are negligible.

IV. Performance Evaluation

A. Experiment Setup

TABLE I
EVALUATED 3D TLC NAND FLASH CHIP CONFIGURATION.

Parameters	**Value**	**Workloads**	**Footprint (GB)**	**Read (GB)**	**Write (GB)**	**R.Ratio (%)**
# of chips	16	HM_0	0.65	2.76	8.03	25.6
Chip size	16GB	PROJ_0	1.71	4.15	14.71	22.0
Plane size	8GB	PRXY_0	0.08	0.27	5.81	4.4
Block size	8MB	STG_0	6.17	7.39	9.31	44.3
Page size	16KB	SRC1_2	1.11	2.49	35.99	6.5
P/E cycles	1000	RSRCH	0.07	1.36	11.02	10.9
Write latency	700 μs	WDEV_0	0.20	3.18	9.24	25.6
Erase latency	5 ms	SRC2_0	0.40	1.63	9.09	15.2

Read latency								
New	LWA HWA	127 μs (0) 127 μs (0)	**Middle**	LWA HWA	762 μs (5) 1524 μs (11)	**Old**	LWA HWA	1016 μs (7) 3048 μs (23)

DWR is evaluated with a popular SSD simulator, SSDsim [3]. The storage is simulated with a 3D TLC NAND flash. The details are as follows: The SSD is 256GB with 8 channels. For each channel, there are 2 chips, and each chip includes 2 planes. Each plane includes 1024 blocks, each block includes 512 pages and each page is 16KB. The maximal number of P/E cycles is set to 1000. Eight workloads from MSR traces [2] are selected for evaluation. Table I shows the configuration of the 3D TLC NAND flash based high-density SSD and the workload characteristics, including footprints, reads and writes. The size of SSD used by each workload is configured based on their footprints. The latency configuration is presented in the bottom of Table I, which complies to the work in [11]. The number in ()

is the average number of retries. To evaluate the benefit of DWR during the whole lifetime, the life of the storage is partitioned into three stages based on the reliability characteristics: new, middle and old. The three stages are defined as follows: For *the new stage*, both HWA and LWA have good reliability, where both of them are worn with only a little number of P/E cycles; For *the middle stage*, HWA is worn with several hundreds of P/E cycles and LWA is still worn with only a little number of P/E cycles; For *the old stage*, HWA is worn approaching the wear rate limit and LWA is also worn with hundreds of P/E cycles. These three stages are simulated based on the retry information from [11]. The number of retries is presented in the table. Since N_R is highly correlated with reliability, it is set with different values at these three stages. In this work, N_R is set to 5, 4, 3 for the above three life stages for simplicity. Detail sensitive studies will be presented in the following section.

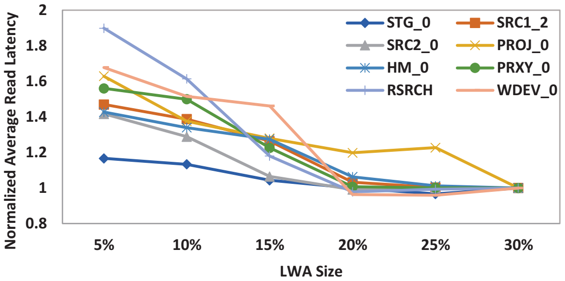


Fig. 6. Normalized average read latency with different LWA sizes.

One of the most important configurations is the size ratio between HWA and LWA. It not only impacts the read performance, but also is critical to the lifetime. Figure 6 shows the read latency by varying the percentages of LWA from 5% to 30%. The results show that when LWA exceeds 20%, the improvement on read performance is diminishing. Based on this observation, LWA is set with 20% in the following experiments. The wearing upper bound is set to 80%. By implementing SWL at HWA, the lifetime impact is further reduced.

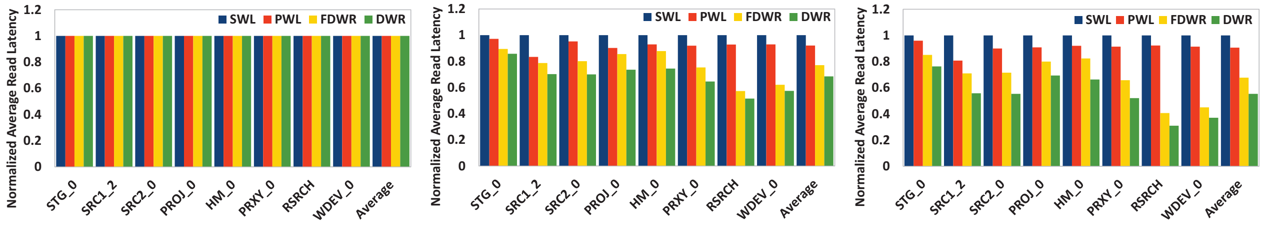
B. Experimental Results

To evaluate DWR, three schemes are evaluated and compared in this section: *Static wear leveling (SWL)*: SWL represents the traditional static wear leveling [10] method with a fixed P/E threshold for blocks. *Progressive wear leveling (PWL)*: PWL represents the identification of hot read data with a fixed threshold and migration them with GC and WL. This scheme refers to [15] that gradually accelerates the triggering of static wear leveling by dynamically reducing the P/E threshold. *Fixed hotness threshold based DWR (FDWR)*: FDWR represents the identification of hot read data with a fixed threshold under differential wearing. For FDWR, N_R is set to 5.

1) Performance: Figure 7 shows the normalized average read latency in different life stages. For the new stage, the performance among the four evaluated schemes is similar. This is because the reliability of flash in this stage is good. For the middle stage, PWL achieves 10% read performance optimization due to that it proposed to migration hot read data with internal mechanisms. FDWR further reduces the read latency by 23% due to that it further proposed to migration hot read data with host reads. DWR further reduces the read latency by 10% due to that it sets a smaller migration threshold for performance optimization. For the old stage, the read performance is further improved compared with SWL and PWL. Compared with SWL and PWL, DWR reduces the read latency by 45% and 33%, respectively. However, we also notice that some workloads have little performance optimization, such as STG_0. The reason is that this workload has little locality. In this case, it is hard to construct differential wearing. For this kind of workload, we propose to place data in one area based on their access characteristics, such as read dominate or write dominate, for simplicity.

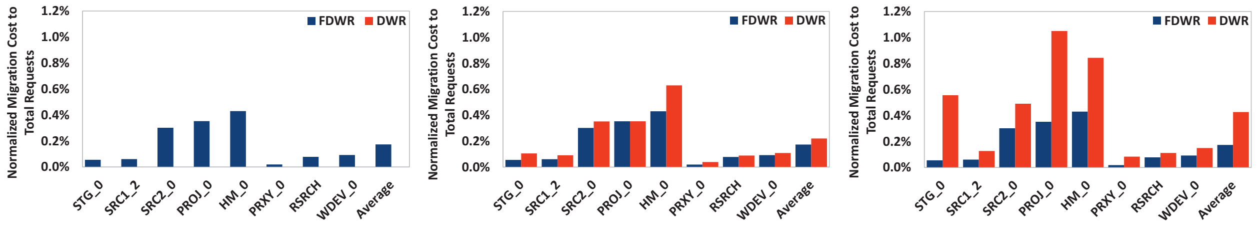
To understand the migration cost for the proposed scheme, the migration cost for FDWR and DWR are collected respectively. Figure 8 shows the results compared with total host requests. The results show several observations: 1) the migration cost for FDWR is similar at different life stages. Even at the new stage with strong reliability, it still introduces migration cost. 2) the overall migration cost is negligible. On average, the migration overhead for DWR is 0.22% in the middle stage and 0.42% in the old stage. 3) the migration cost of DWR is slightly increased at the old stage compared with FDWR. This is because N_R is 3 for DWR, which encourages migration for read performance optimization.

2) Lifetime: To evaluate the lifetime, we use normalized terabyte writes (TBW) is used as the metric. During the implementation of DWR, an existing wear leveling scheme can be adopted at HWA and LWA. In this part, two existing wear leveling schemes, SWL and PWL [15], are adopted to show the characteristics of DWR. For the first case, SWL is adopted for both HWA and LWA, respectively. Once the wearing limitation is approached, the proposed lifetime optimization scheme is activated to avoid the wearing of HWA. For the second case, PWL is adopted for both HWA and LWA, respectively. PWL



(a) New stage (b) Middle stage (c) Old stage

Fig. 7. Normalized average read latency in different stages.



(a) New stage (b) Middle stage (c) Old stage

Fig. 8. Normalized migration cost in different stages.

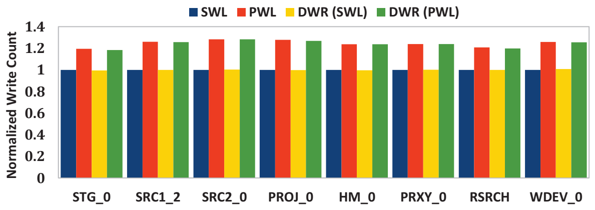
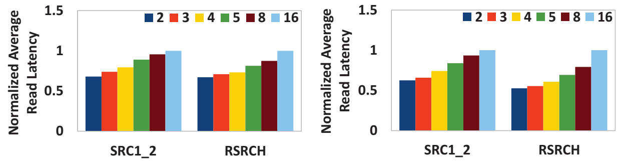


Fig. 9. Endurance achieved by different wear leveling policies.

also has a wearing limitation before wear leveling is activated, which should be smaller than that of the wearing limitation of HWA. During the evaluation, the maximal number of P/E cycles is set to 1000 and corresponding workloads are cycled until there are 2% of bad blocks generated. Figure 9 shows the normalized TBW for the evaluated schemes. The results show that TBW of DWR with SWL or PWL is similar to that of SWL or PWL. It means that DWR has a negligible lifetime impact. The reason for the above results comes from that most workloads have access locality. By leading more write requests to LWA at the old stage, the wearing between these two areas becomes small and similar to the effects of wear leveling.

3) Sensitive study: One of the most important parameters is the setting of N_R. It not only affects migration overhead, but is also critical to read performance. Figure 10 shows read performance by WDR by varying N_R from 2 to 16. The results show that with the increases of N_R, the read performance is degraded due to that more requests happen in HWA. In this work, to avoid high migration cost and achieve read performance optimization, N_R was set to 4 in the middle stage to reduce migration overhead, and to 3 in the old stage to further optimize read performance.



(a) Middle stage (b) Old stage

Fig. 10. Normalized average read latency with different N_R.

V. Conclusion

This work proposes a differential wearing design, the DWR design, which retains a portion of blocks with lower P/E cycles to store hot read data, improving the read performance of flash memory. Read performance is further optimized by storing hot read data on reliable blocks through a data migration scheme. The experimental results show that DWR achieves 45% read performance optimization with negligible impacts on the lifetime. In future work, we will further perform relevant tests on 3D QLC NAND flash to verify the performance of DWR. At the same time, other wear leveling designs will be performed to further optimize the read performance.

References

[1] Y. H. Chang, J. W. Hsieh, and T. W. Kuo. Improving flash wear-leveling by proactively moving static data. *IEEE Transactions on Computers*, 59(1):53–65, 2009.

[2] Narayanan Dushyanth and Thereska Eno et al. Migrating server storage to SSDs: analysis of tradeoffs. In *Proceedings of ACM European conference on Computer systems*, pages 145–158, 2009.

[3] Yang Hu and Hong Jiang et al. Exploring and exploiting the multilevel parallelism inside SSDs for improved performance and endurance. *IEEE Transactions on Computers (TC)*, 62(6):1141–1155, 2013.

[4] Qiao Li and Liang Shi et al. Improving ldpc performance via asymmetric sensing level placement on flash memory. In *2017 22nd Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 560–565, 2017.

[5] Qiao Li and Liang Shi et al. Process variation aware read performance improvement for ldpc-based nand flash memory. *IEEE Transactions on Reliability*, 69(1):310–321, 2020.

[6] Qiao Li and Min Ye et al. Shaving retries with sentinels for fast read over high-density 3d flash. In *2020 53rd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*, pages 483–495, 2020.

[7] Shuwen Liang and Zhi Qiao et al. An empirical study of quad-level cell (qlc) nand flash ssds for big data applications. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 3676–3685, 2019.

[8] Yina Lv and Liang Shi et al. Access characteristic guided partition for read performance improvement on solid state drives. In *2020 57th ACM/IEEE Design Automation Conference (DAC)*, pages 1–6, 2020.

[9] Yina Lv and Liang Shi et al. *Latency Variation Aware Read Performance Optimization on 3D High Density NAND Flash Memory*, page 411–414. 2020.

[10] M. Murugan and Dhc Du. Rejuvenator: A static wear leveling algorithm for nand flash memory with minimized overhead. In *IEEE Symposium on Mass Storage Systems and Technologies*, 2011.

[11] J. Park and M. Kim et al. Reducing solid-state drive read latency by optimizing read-retry. *ACM*, 2021.

[12] Roman Pletka and Nikolaos Papandreou et al. Improving nand flash performance with read heat separation. In *2020 28th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pages 1–8, 2020.

[13] Shigui Qi and Dan Feng et al. A new solution based on multi-rate ldpc for flash memory to reduce ecc redundancy. In *2015 IEEE Trustcom/BigDataSE/ISPA*, volume 1, pages 918–923, 2015.

[14] Shigui Qi and Dan Feng et al. Cdf-ldpc: A new error correction method for ssd to improve the read performance. *ACM Trans. Storage*, 13(1), February 2017.

[15] Mingchang Yang and Yuanhao Chang et al. Reducing data migration overheads of flash wear leveling in a progressive way. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 24(5):1808–1820, 2016.

[16] Kai Zhao and Wenzhe Zhao et al. Ldpc-in-ssd: Making advanced error correction codes work effectively in solid state drives. In *11th USENIX Conference on File and Storage Technologies (FAST 13)*, pages 243–256, San Jose, CA, February 2013. USENIX Association.