Research Article

A Block Classification Method with Monitor and Restriction in NAND Flash memory

Myungsub Lee
Department of Computer Information, Yeungnam University College,Daegu, Republic of Korea
skydream@ync.ac.kr

Article History: Received: 11 January 2021; Accepted: 27 February 2021; Published online: 5 April 2021

Abstract*:*In this paper, we propose a block classification with monitor and restriction (BCMR) method to isolate and reduce the interference of blocks in garbage collection and wear leveling. The proposed method monitors the endurance variation of blocks during garbage collection and detects hot blocks by making a restriction condition based on this information. This method induces block classification by its update frequency for garbage collection and wear leveling, resulting in a prolonged lifespan for NAND flash memory systems. The performance evaluation results show that the BCMR method prolonged the life of NAND flash memory systems by 3.95% and reduced the standard deviation per block by 7.4%, on average.

Keywords: NAND flash memory, garbage collection, wear leveling, lifetime, block classification

1. Introduction

Unlike conventional magnetic disks, NAND flash memory, a type of nonvolatile memory, has advantages such as low power consumption, fast operating speeds, and light weight. Owing to these advantages, NAND flash memory is used in various applications such as solid-state drives (SSD), USB memory, and smartphones. However, unlike magnetic disks, NAND flash memory has problems such as a shorter lifespan, an erase-before-write requirement, and imbalance of operation units [1-4]. To address these problems, many researchers have proposed file systems or data structures considering the inherent characteristics of NAND flash memory [6-8]. However, recently considerable research has been conducted on flash translation layers (FTL), a development that can solve these problems without changing the traditional file systems or data structures.

As shown in Fig. 1, FTL is composed of an address translation table, garbage collection, and wear leveling [7]. Previous researchers have conducted studies on methods to extend the lifespan of NAND flash memory using either garbage collection or wear leveling. However, these researchers have proposed optimization methods to extend life assuming that each component of the FTL layer operates alone. Therefore, problems arise when certain blocks are concurrently selected in the system environment where the proposed methods are simultaneously connected and performed in the FTL layer. This causes shortening of the NAND flash memory lifespan. Therefore, in system environments with FTL layers, in which garbage collection and wear leveling are simultaneously considered, the problem of wear in certain blocks can be solved easily by classifying and allocating separate blocks to the two processes.

This paper introduces a block classification with monitor and restriction (BCMR) methodology to classify frequently selected blocks in a system environment in which garbage collection and wear leveling are simultaneously conducted. The proposed BCMR uses the following procedure in the process of garbage collection.

- Information on changes in the block erase count is monitored.
- The sacrifice block is selected based on the monitored information.
- When the sacrifice block is selected, the constraints are applied.

Based on the above procedure, frequently selected blocks in BCMR are selected according to the algorithm proposed in this paper, thereby helping extend the lifespan of NAND flash memory. This paper is structured as follows. Section 2 discusses the background information and related research on NAND flash memory. Section 3 details the process of performing BCMR. Section 4 describes a performance evaluation of BCMR, and Section 5 concludes the paper.

2. Related Work

As shown in Table 1, single-level cells (SLC), multi-level cells (MLC), or triple-level cells (TLC) are implemented in NAND flash memory depending on the method in which bits are classified in flash cells [8].

Table 1. Comparison of flash cells

Description	SLC	MLC	TLC
Read latency	25 µs	60 µs	100 µs
Program latency	200 µs	800 µs	2.4 ms
Erase latency	700	1.5 ms	3.0 ms

	μs		
Bits per flash cell	1 bit	2 bit	3 bit
Endurance	10^5	10^4	10^3
Price per bit	high	low	very low

As shown in Table 1, as the number of bits per flash cell in NAND flash memory increases, operation latency increases. Because an increase in bits per flash cell has the effect of decreasing price per bit, recently, MLC has mainly been used for its trade-off between performance and price. In addition, NAND flash memory has inherent characteristics different from conventional magnetic disks, which are described as follows [1-2].

- NAND flash memory operations include read, write, and delete. Read and write operations are executed by page units, whereas delete and block operations are executed by block units. Here, a block is a group of multiple pages.
- As shown in Table 1, the latencies of read, write, and delete operations of NAND flash memory are asymmetric.
- Unlike magnetic disks, there is a limited number of write cycles per block in NAND flash memory.

To address the above-mentioned problems of NAND flash memory, initial studies have proposed specialized file systems or data structures [7]; however, the recent studies have mainly used the flash translation layer (FTL), a method that supports the properties of magnetic disks as-is. The structure of FTL is shown in Fig. 1 [1].

The address translation table records the logical and physical addresses as one item; accessing the physical address in NAND flash memory is supported by using the logical address. Through this method, NAND flash memory overcomes the problems of existing operations that execute delete operations before write operations. In other words, in-place update of NAND flash memory is possible by using the address translation table. However, a number of invalid pages arise when this process is repeated, thereby taking up unnecessary memory space. Therefore, to clean up this unnecessary space, a garbage collection process is needed.

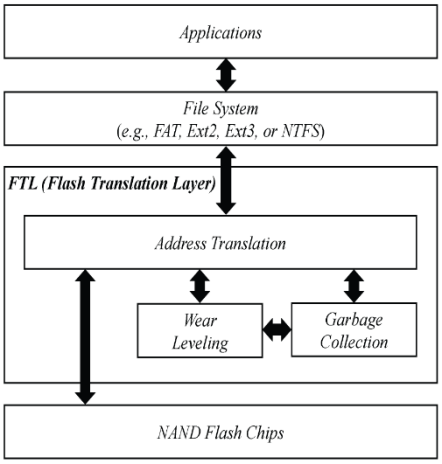

Fig. 1. System architecture of FTL

The address translation table records the logical and physical addresses as one item; accessing the physical address in NAND flash memory is supported by using the logical address. Through this method, NAND flash memory overcomes the problems of existing operations that execute delete operations before write operations. In other words, in-place update of NAND flash memory is possible by using the address translation table. However, a number of invalid pages arise when this process is repeated, thereby taking up unnecessary memory space. Therefore, to clean up this unnecessary space, a garbage collection process is needed.

Garbage collection supports the function of collecting the invalid pages of NAND flash memory, which is carried out during the in-place update support or data deletion. However, if garbage collection is executed repeatedly, a problem arises in which only certain blocks frequently execute delete operations, thus becoming the main cause for NAND flash memory life degradation. To address this issue, the wear leveling method is used.

Wear leveling is a technique to address the problem of imbalanced delete operations between blocks that arise in garbage collection. The number of delete operations on each block is equalized to extend the lifespan of NAND flash memory. In other words, wear leveling plays a role in leveling the wear of memory by selecting blocks that have almost no delete operations to participate in garbage collection. Most recent research on methods to extend the life of NAND flash memory uses the afore-mentioned garbage collection and wear leveling techniques [10, 15].

Initial research on extending the life of NAND flash memory focused on securing space for use. As a result, certain blocks in memory were found to be frequently used. This practice was found to deplete NAND flash memory life faster than expected. To address this problem, the CB, CAT, and SAGC methods have been proposed [11, 16].

Recently, wear leveling has been actively researched to extend the life of NAND flash memory. A representative study is SWL. In SWL, a bit flag table is used to identify the blocks that have not participated in garbage collection and make them to participate in garbage collection, thereby leveling the memory wear and extending the life of NAND flash memory.

However, the proposed methods are efficient when either garbage collection or wear leveling is used and do not consider the system environment that uses both processes simultaneously. For example, to employ wear leveling efficiently, only blocks that contain pages that have almost no write operations should be selected to participate in garbage collection. To achieve this, the pages that participate and those that do not participate in the operation should be classified and transferred to separate blocks. As a result, the migrated pages are situated between the hot block (block that frequently deletes) and the cold block (block that frequently does not delete). In a system that takes into account both garbage collection and wear leveling, there is a risk that the blocks between the hot and cold blocks are frequently referenced. Therefore, in a system environment in which garbage collection and wear leveling are performed simultaneously, it is necessary to separate the hot and cold blocks. To fulfill this requirement, this paper focuses on preventing interference and ensuring isolation of the selected blocks between the two processes.

3. Proposed Method

This paper proposes a block classification with monitor and restriction method to prevent interference of blocks and ensure isolation in the garbage collection and wear leveling process. The procedure we use is shown in Fig. 2.

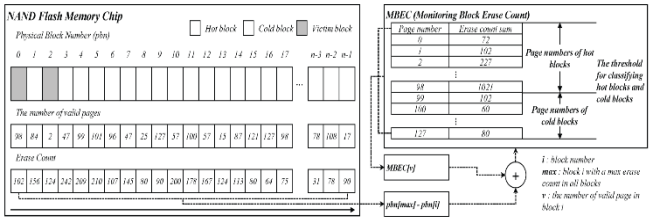

Fig. 2. Procedure of BCMR

As shown in Fig. 2, the main procedure of the BCMR method is executed in the following three steps.

(1) In the garbage collection execution process of BCMR, all blocks are monitored to measure changes in the block erase count. Based on this information, when the hot block is being wear leveled in garbage collection, the cold block is selected. Here, a hot block refers to the block in which write operations occur frequently, whereas a cold block refers to the block in which write operations occur rarely.

(2) Based on the measured changes in the erase count, cold blocks are excluded from selection in the garbage collection process.

(3) Finally, in the wear leveling process, the hot blocks and cold blocks are classified based on the monitored changes in the erase count in garbage collection and a new threshold is set. Through this process, interference is prevented, and isolation is ensured between the two processes.

As shown in Fig. 2, the monitoring block erase count (MBEC) table is used to measure changes in the block erase count during the execution of garbage collection. The valid page numbers of the frequently selected blocks can be determined by using the information recorded in the MBEC table. Thus, BCMR can measure the block erase imbalance by using the MBEC table to track differences in the changes in the erase count of the blocks preferred by garbage collection. In addition, BCMR also clearly classifies the status of the corresponding block as a hot or cold block to determine the number of valid pages. The details of the BCMR procedure shown in Fig. 2 are given in Algorithm 1.

Algorithm 1: BCMR procedure

input: *max_ec, max_blk, ec, mbec, v_pages, cost, i,*

and v_th

output: *null*

// Get the maximum number of erase count

1 *max_ec* ← get_max_erase_count();

2 **for** *i* = 0 **to** *max_blk***do**

// Measure a difference between a maximum erase count and a corresponding erase count.

3 *mbec[v_pages[i]] = max_ec-ec[i]*;

// Ignore an corresponding block *i* when it satisfies this condition.

4 **if** *v_pages[i]>v_th***then**

5 continue;

6 **end**

// Estimate a cost of block *i* on each technique.

7 **end**

// Reclaim victim blocks by garbage collection.

8 **if** *wear_leveling_condition = true* **do**

// Get a bcmr cost by equation (1).

9 *cost* ← get_bcmr_cost();

10 **if** *cost*> 0 **then**

11 *v_th = v_th*- 1;

12 **else**

13 *v_th = v_th*+ 1;

14 **end**

// Reset the mbec table and perform procedures in each technique.

15 **end**

Through Algorithm 1, BCMR monitors changes in the erase count for all blocks and uses this information to classify hot blocks and cold blocks. In wear leveling, Eq. 1 is used to reset the threshold to prevent the blocks not the target of garbage collection from participating in garbage collection.

$$\text{cost} = \sum_{i=0}^{T} BCMR(i) - \sum_{i=T}^{max} (BCMR(i)\ x\ r) \qquad (1)$$

In this equation, *i*indicates page number, *T* indicates the threshold that classifies the existing hot blocks and cold blocks, *max* indicates the maximum number of pages, and *r* indicates the weight to be assigned to the cold blocks. Based on the cost obtained from Eq. 1, the new threshold of BCMR is calculated as follows.

- If cost exceeds 0, then the change in the erase count of the hot block is large and threshold value is accordingly reduced.
- If cost is less than 0, then the change in the erase count of the cold block is large and the threshold is accordingly increased.

As a result, the threshold of BCMR classifies hot blocks and cold blocks on the basis of the difference between the block erase counts, which is the information required during the garbage collection process to prevent interference and ensure isolation of the selected blocks between the two processes.

4. Experiment and Evaluation

To evaluate the performance of the proposed BCMR method, we used the SSD Extension for DiskSim Simulation Environment [13-14]. The workload used to evaluate the performance of BCMR is as shown in Table 2; in the workload, SPC was set to include the characteristics of the application performed in OLTP and FIU was set to include operational characteristics such as terminal users and web services.

Table 2. Workload characteristics of SPC and FIU

Workload		Ratio of request (%)		Avg. request (byte)
		read	write	
SPC	financial 1	23.16	76.84	4754.3
	financial 2	82.28	17.72	4394.0
FIU	home1	1.12	98.88	7246.9
	home2	12.40	87.60	13079.9
	online	35.55	64.45	17566.8
	webmail	26.03	73.97	17930.5

The performance evaluation of BCMR was based on data such as NAND flash memory life, valid page migration, and block standard deviation. First, for the memory life and block standard deviation, the erase count for each block was set to 10^3, and the block standard deviation was set based on the block time that first exhausted all erase counts among all blocks. A comparative analysis was conducted with the valid page migration value based on the number of page migrations after 100 million write operations for overheads. The parameters used in the experimental environment were based on the MLC values in Table 1, while the detailed parameters are shown in Table 3.

Table 3. Simulation parameters

Description	Value
Reserved free blocks	15%
Garbage collection condition	# of free blocks < 5%
# of total blocks	4096
Pages per block	128
Page size	4 KB
Page read latency	60 us
Page program latency	800 us
Block erase latency	1.5 ms
Blocks per lifetime	10^4

Fig. 3 shows the NAND flash memory life. As shown in Fig. 4, the average lifespan of NAND flash memory was extended by 3.58%, 2.13%, 7.06%, and 3.06% when BCMR was activated. As to the reason for the extended lifespan shown in the experimental values, it is thought that the lifespan can be extended by lowering the selection frequency of the corresponding blocks between the hot and cold blocks through selecting hot blocks in garbage collection and cold blocks in wear leveling. In addition, because BCMR takes advantage of the existing garbage collection and wear leveling techniques, we were able to confirm further performance improvements through experimental results.

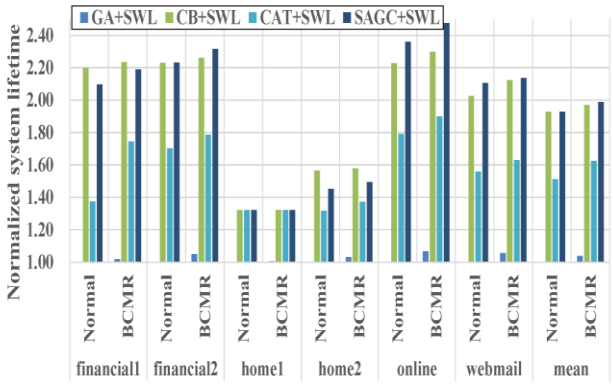

Fig. 3. The lifespan of NAND flash memory system

Fig. 4 shows the block standard deviation, which was reduced by up to 6.52%, 4.16%, 11.44%, and 7.51% when BCMR was implemented. This reduction of the block standard deviation can be considered as a meaningful indicator because it directly affects the lifespan. Although reducing the block standard deviation can extend the NAND flash memory life, there is a risk of NAND flash memory page migration overhead increasing during processing. Therefore, the algorithm that is used must consider this risk.

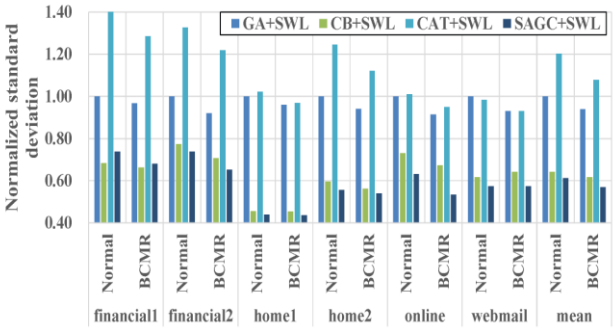

Fig. 4. The normalized standard deviation based on GA and SWL

Fig. 5 shows the average number of page migrations during garbage collection and wear leveling, which was reduced by up to 0.01%, 0.77%, 1.65%, and 0.33% when BCMR was activated.

The overhead required to perform BCMR proposed in this paper is the use of separate memory. The amount of memory used to operate BCMR depends on the number of pages of the NAND flash memory used. Memory usage was measured based on the NAND flash memory used in the experimental environment; about 512 bytes (4 bytes × 128 pages) were used.

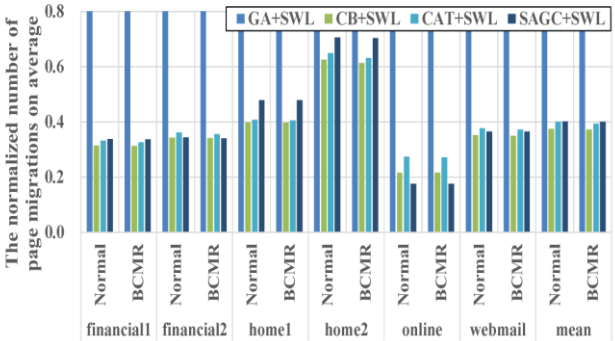

Fig. 5. The normalized number of page migrations on average

Considering the total amount of memory used in the computer, this can be judged to have almost no effect on the operation of the system because of its small size. Accordingly, the method proposed in this paper is proven to extend the life of NAND flash memory.

5. Conclusions

In this paper, we proposed the BCMR method to prevent interference and ensure isolation in garbage collection and wear leveling. The proposed BCMR monitors changes in the block erase count during garbage

collection and uses this information to select hot blocks by constraining the selection of sacrifice blocks. The performance evaluation results show that the BCMR method extended NAND flash memory lifespan by 3.95% on average, whereas the standard deviation between blocks was reduced by 7.4%

6. Acknowledgment

This research was supported by the Yeungnam University College Research Grants in 2019.

References

1. Myungsub Lee, “A Monitoring Methodology based on Block Erase Count for Classifying Target Blocks”, Smart Convergence of Culture Technology Letters, SCCTL06, pp.265-270, 2020.
2. Myungsub Lee, “Sampling-Based Block Erase Table Method in Wear Leveling Technique for Flash Memory”, Journal of Engineering and Applied Sciences, 13(4), pp. 1023-1028, 2018.
3. Myungsub Lee, “A Wear Leveling Technique based on SBET for Flash Memory”, Advanced and Applied Convergence Letters, AACL10, pp. 113-115, 2017.
4. Myungsub Lee, “A Tracking Cold Blocks for Static Wear Leveling in FTL-based NAND Flash Memory”, The 6th International Symposium on Advanced and Applied Convergence, ISAAC2018, pp. 50-52, 2018.
5. Ma. D, Feng. J, & Li. G, “A survey of address translation technologies for flash memories”, ACM Computing Surveys (CSUR), Vol. 46, No. 3, 2014. (1)
6. Chang. B, Wang. Z, Chen. B, & Zhang. F, “Mobipluto: File system friendly deniable storage for mobile devices”, Proceedings of the 31st Annual Computer Security Applications Conference, pp. 381-390, 2015. (4)
7. Athanassoulis. M, &Ailamaki. A, “BF-tree: approximate tree indexing”, Proceedings of the VLDB Endowment, Vol. 7, No. 14, pp. 1881-1892, 2014.
8. Chung. T. S, Park. D. J, Park. S, Lee. D. H, Lee. S. W, & Song. H. J, “A survey of flash translation layer”, Journal of Systems Architecture, Vol. 55, No. 5, pp. 332-343, 2009.
9. Hachiya. S, Johguchi. K., Miyaji. K, & Takeuchi. K, “TLC/MLC NAND flash mix-and-match design with exchangeable storage array”, 2013 International Conference on Solid State Devices and Mater, pp. 894-895, 2013.
10. Yang. M. C, Chang. Y. M, Tsao. C. W, Huang. P. C, Chang. Y. H, &Kuo. T. W, “Garbage collection and wear leveling for flash memory: past and future”, SACM Computing Surveys (CSUR), pp. 66-73, 2014.
11. Kwon. O & Koh. K, “Swap space management technique for portable consumer electronics with NAND flash memory”, IEEE Transactions on Consumer Electronics, Vol. 56, No. 3, 2010.
12. Chang. Y. H, Hsieh. J. W &Kuo. T. W, “Improving flash wear-leveling by proactively moving static data”, IEEE Transactions on Computers, Vol. 59, No. 1, pp. 53-65, 2010.
13. Bucy. J. S, Schindler. J, Schlosser. S. W, & Ganger. G. R, “The DiskSim simulation environment version 4.0 reference manual (cmu-pdl-08-101)”, Parallel Data Laboratory, 2008.
14. Prabhakaran. V, &Wobber. T, “SSD extension for DiskSim simulation environment”, Microsoft Research, 2009.
15. Csonka, A., Bareith, T., Gál, V. A., &Fertő, I. (2018). Spatial Pattern of CAP Measures Promoting Agroforestry in Hungary. AgBioForum, 21(2), 127-134.
16. Helmar, M., Johnson, S. R., Myers, R. J., Whistance, J., &Baumes, H. (2018). The Economic Impacts of US Tariffs for Fuel Ethanol and Biodiesel. AgBioForum, 21(1), 25-34.