ELectronics EXpress

LETTER

IEICE Electronics Express, Vol.9, No.24, 1874-1880

A high-performance wear-leveling algorithm for flash memory system

Ching-Che Chung[1a)], **Duo Sheng**[2], **and Ning-Mi Hsueh**[1]

[1] *Department of Computer Science & Information Engineering, National Chung-Cheng University, 168 University Road, Min-Hsiung, Chia-Yi 621, Taiwan*

[2] *Department of Electrical Engineering, Fu Jen Catholic University, 510 Chung-Cheng Rd. Hsin-Chung, Taipei 24205, Taiwan*

a) *wildwolf@cs.ccu.edu.tw*

Abstract: In this paper, a low-complexity high-performance wear-leveling algorithm which named sequential garbage collection (SGC) for flash memory system design is presented. The proposed SGC outperforms existing designs in terms of wear evenness and low design complexity. The lifetime of the flash memory can be greatly lengthened by the proposed SGC. The proposed SGC doesn't require any tuning threshold parameter, and thus it can be applied to various systems without prior knowledge of the system environment for threshold tuning. Simulation results show that the maximum block erase count and standard deviation of the block erase count compared to the greedy algorithm are decreased by up to 75% and 94%, respectively.

Keywords: NAND flash, wear leveling, garbage collection

Classification: Storage technology

References

[1] A. Kawaguchi, S. Nishioka, and H. Motoda, "A flash-memory based file system," *Proc. USENIX Annual Technical Conference*, pp. 155–164, June 1995.

[2] M. Rosenblum and J. K. Ousterhout, "The design and implementation of a log-structured file system," *ACM Transactions on Computers Systems*, vol. 10, no. 1, pp. 26–52, Feb. 1992.

[3] J.-W. Park, et al., "An integrated mapping table for hybrid FTL with fault-tolerant address cache," *IEICE Electron. Express*, vol. 6, no. 7, pp. 368–374, April 2009.

[4] Y.-H. Chang, J.-W. Hsieh, and T.-W. Kuo, "Improving flash wear-leveling by proactively moving static data," *IEEE Trans. Comput.*, vol. 59, no. 1, pp. 53–65, Jan. 2010.

[5] Y.-H. Chang, J.-W. Hsieh, and T.-W. Kuo, "Endurance enhancement of flash-memory storage systems: an efficient static wear leveling design," *Proc. ACM/IEEE Design Automation Conference*, pp. 212–217, June 2007.

[6] Samsung Electronics, "K9GAG08U0M 2G × 8 bit NAND Flash Memory Data Sheet," 2006.

[7] L.-P. Chang and L.-C. Huang, "A low-cost wear-leveling algorithm for block-mapping solid-state disks," *Proc. ACM SIGPLAN/SIGBED Conference on Languages, Compilers, Tools and Theory for Embedded Systems (LCTES),*

DOI: 10.1587/elex.9.1874
Received October 22, 2012
Accepted November 06, 2012
Published December 28, 2012

pp. 31–40, April 2011.

1 Introduction

In recent years, flash memory had become an important storage system in smartphones and ultra-thin notebooks. Flash memory has lower power consumption, faster random access performance, and better shock resistance than the conventional hard disks. However, it also has some disadvantages, such as the high price cost, writing speed, and the most important is the endurance of the flash memory.

There are some special characteristics in the flash memory, such as the unit of read/program operation is a page, but the unit of erase operation is a block. In addition, the speed in different operations has great difference. The content of the flash memory cannot be overwritten; it must be erased before new data can be stored. Therefore, flash memory requires a flash translation layer (FTL) to translate the logical addresses into physical addresses [1, 2, 3]. When data are written to the flash memory, the amount of free pages becomes low, and thus the garbage collection (GC) operation is performed to recycle the space occupied by invalid pages. The GC operation collects valid pages in the selected block and copies these pages to another free block, and then it erases the selected block. Greedy algorithm (GA) [1] is the most effective algorithm to choose the block with highest number of invalid pages.

2 Motivation

The chip density of the flash memory is growing up but the endurance of flash memory is declining. Typically, each block has the endurance of 500 times (triple-level cells, TLCs) to 100,000 times (single-level cells, SLCs) erase operations. GC operation with greedy algorithm selects blocks that have the highest number of invalid pages. Therefore, GC operation may wear out some blocks of the flash memory due to the localities of data access. For example, if some blocks stored hot data, they will be invoked by GC operation more frequently. These blocks may retire early, and this causes the reliability problem of the flash memory. To solve this problem, many wear leveling techniques have been proposed. The purpose of wear-leveling algorithm is to evenly distribute block erases over the flash memory.

In static wear-leveling (SW) [4, 5], a block erasing table (BET) is created to identify which block has been erased during a given time period. When the difference between the maximum block erase count and the minimum block erase count is larger than the specified threshold, it will move the static data to the hot block to balance the erase count in each block. This algorithm prevent cold data from staying at any block for a long time, and then the maximum block erase count difference between any two blocks can be minimized.

The implementation complexity of the wear-leveling algorithms determines the applicability of the algorithm. Existing wear-leveling algorithms require prior knowledge of the system environment for threshold

DOI: 10.1587/elex.9.1874
Received October 22, 2012
Accepted November 06, 2012
Published December 28, 2012

tuning, and thus increasing their design complexity. However, the threshold for various systems can be very different. Using inadequately tuned parameters can cause unexpectedly high wear-leveling overhead or worse performance of wear-leveling. Therefore, an on-line threshold tuning method is proposed in [7] to reach good balance between wear evenness and overhead. However, it makes the wear-leveling algorithm becomes more complex and can only be implemented with a software approach.

In this paper, a low-complexity high-performance wear-leveling algorithm which named sequential garbage collection (SGC) for flash memory system design is presented. The proposed algorithm combines GC operation and wear-leveling into a single process. In addition, the proposed SGC doesn't require any tuning threshold parameter, and thus it can be applied to various systems without prior knowledge of the system environment. The low-complexity low-cost SGC algorithm makes it is easy to be implemented by firmware-based or hardware-based approaches.

3 The proposed SGC algorithm

In the flash memory system, the embedded micro-controller has very limited memory space, and the hardware and computing resources are constricted. Thus the proposed sequential garbage collection (SGC) algorithm requires a low memory space and demands limited hardware resources. The basic idea to combine GC operation and wear-leveling into a single process is very simple that blocks should be erased in a sequential manner.

Fig. 1 shows the proposed SGC algorithm. A bit vector which named invalid page flag (IPF) is added to reduce living page copy overhead since we didn't use the greedy algorithm to search for the block with maximum invalid pages. The IPF records whether a block contains more than 75% invalid pages. For example, if IPF[*index*] is equal to 1, then it means the block with index number: *index* contains more than 75% invalid pages. Oppositely, if IPF[*index*] is equal to 0, then it means the block has less than 75% invalid pages. The IPF vector can be updated during the write operation, and each block requires only one-bit flag. The GC operation will erase the blocks with IPF=1 with a higher priority.

```
-------------------------------------------------------------------------------
Algorithm : SGC (Sequential Garbage Collection)
-------------------------------------------------------------------------------
Input: free_block, fcnt, seq, index, and IPF
Output: null
 1: if free_block = 1 then
 2:     if fcnt = 0 then
 3:         seq ← (seq+1) MOD BlockSize
 4:         index ← seq;
 5:         GarbageCollection(index);
 6:     end else
 7:         while IPF[index]=0 do
 8:             index ← (index+1) MOD BlockSize;
 9:         end
10:         GarbageCollection(index);
11:         index ← (index+1) MOD BlockSize;
12:     end
13: end
```

Fig. 1. The proposed SGC algorithm.

DOI: 10.1587/elex.9.1874
Received October 22, 2012
Accepted November 06, 2012
Published December 28, 2012

In the proposed SGC, *free_block* means the number of free blocks in the flash memory, and *fcnt* is the number of "1" in the IPF vector. If *fcnt* is not zero, there are blocks having more than 75% invalid pages. In addition, *seq* and *index* are the index numbers for searching for the target block for current GC operation. If the number of free blocks becomes one, the controller executes SGC algorithm and performs GC operation on the selected block. In SGC algorithm, if *fcnt* is zero, there has no block with a higher priority. Then the blocks are erased in a sequential manner (step 2-6). Otherwise, if *fcnt* is not zero, there are blocks having more than 75% invalid pages. The SGC algorithm searches for the block with IPF=1 and then performs GC operation on the selected block (step 7-11).

The proposed SGC algorithm evenly distributes block erases over the entire flash memory, and thus the lifetime of the flash memory can be greatly lengthened. The proposed SGC algorithm requires a low memory space and demands limited hardware resources, and thus it is easy to be implemented in the flash memory system.

4 Experimental results

The proposed SGC algorithm had been verified by the Socle Technology Corporation MDK-3D development board. The CPU is ARM1176JZF and the frequency is up to 1 GHz. The AHB frequency is up to 200 MHz and it supports the NOR-flash/NAND-flash/DDR2. We implement the SGC algorithm in the flash controller with the field programmable gate array (FPGA). Fig. 2 shows the verification platform. The write addresses and flash memory commands are sent to the system controller and memory controller with the AHB bus. After testing, the erase counts of the blocks are sent out by the APB bus. The 2 GB flash memory module has 4096 blocks, and each block has 128 pages.

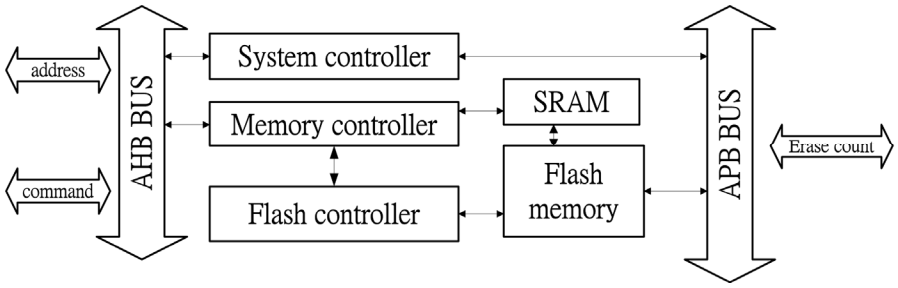
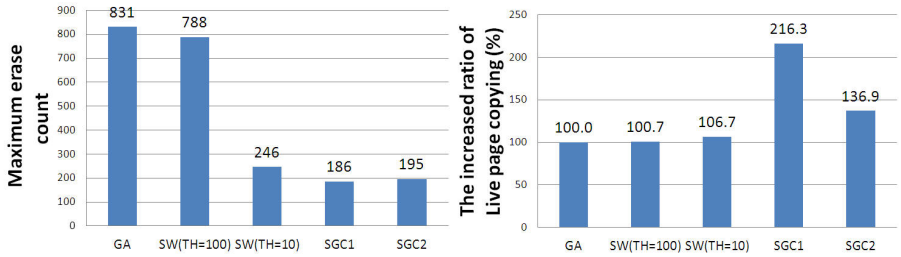


Fig. 2. The verification platform.

For comparisons with existing wear-leveling algorithms, we rebuild the GC operation with greedy algorithm [1] and named as **GA**. In GA, GC operation will select blocks that have the highest number of invalid pages, and thus GA doesn't consider the wear-leveling problem. We also rebuild the static wear-leveling algorithms [4, 5] (with different threshold values) and named as **SW**. In the simulation, totally 200 GB data are written to verify the performance of wear-leveling algorithms. To show the effectiveness of the IPF vector, we disable the IPF vector in the simulation for comparison and named as **SGC1**. The SGC algorithm with IPF vector is named as **SGC2**.

DOI: 10.1587/elex.9.1874
Received October 22, 2012
Accepted November 06, 2012
Published December 28, 2012

Lifetime is an important issue of the flash memory. In Fig. 3 (a), the algorithm which results in a higher maximum erase count means that the lifetime of the flash memory with that algorithm is shorter. In GA, it performs GC operation on the blocks with highest number of invalid pages. Therefore it may often erase on the same blocks which stored the hot data. In SW, when the difference between the maximum block erase count and minimum block erase count is larger than the specified threshold, SW moves the static data to the hot block to balance the erase count in each block. On the other hand, different threshold value will cause different results. For example, the maximum block erase count for SW with threshold=100 and with threshold=10 are 788 and 246, respectively. SGC1 always erases blocks in a sequential manner, and therefore it can achieve the lowest maximum block erase count but the extra living page copy overhead is too large. Thus SGC2 uses the IPF vector to reduce the overhead in SGC1.



(a) **(b)**

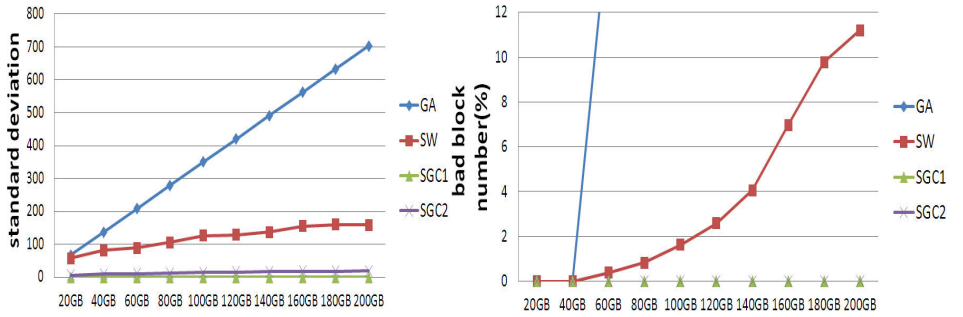
Fig. 3. (a) Maximum block erase count. **(b)**The increased ratio of live page copying (%)

The overhead of the wear-leveling algorithm includes extra block erase operations and extra living page copy operations. GA has smallest living page copy overhead as shown in Fig. 3(b). SW still uses the greedy algorithm to perform GC operation, but it requires extra GC operations to balance the erase count in each block, thus it has a higher overhead than GA. In SGC1, the efficiency of recycling the space occupied by invalid pages is not as well as greedy algorithm, thus it causes many extra erase operations. Therefore, SGC2 uses the IPF vector to erase the block that has more than 75% invalid pages with a higher priority, so it can reduce the overhead in SGC1, and then the increased ratio of the extra living page copy is greatly reduced.

Fig 4 (a) shows the changes of the standard deviation of block erase count versus the amount of written data, and the threshold for SW is 100. As we can expect, the standard deviation in SGC1 will not have any changes, and SGC2 also achieves a very small standard deviation. However, in SW, the standard deviation is growing up when the amount of written data is increased.

If the triple-level cell (TLC) is used in the flash memory design, the endurance of each block is about 500 times. Thus the block with more than 500 times erase count can be treated as a bad block. Fig. 4(b) shows the changes of the bad block number versus the amount of written data, and the threshold for SW is 100. In GA, the bad block number is quickly over 10% at 50 GB data written because GA doesn't consider the wear-leveling

DOI: 10.1587/elex.9.1874
Received October 22, 2012
Accepted November 06, 2012
Published December 28, 2012



(a) **(b)**

Fig. 4. **(a)** changes in standard deviation. **(b)** changes in bad block number.

problem. In SW, it can balance the erase counts in each block thus the bad block number reaches 10% until 180 GB data written. The proposed SGC algorithm evenly distributes block erases over the entire flash memory, and thus the bad block number is still zero after 200 GB data written.

Table I summarizes the performance comparisons for four methods. GA with the greedy algorithm has the highest efficiency on recycling invalid pages, and thus it has lowest living page copy overhead. However, it causes the flash memory to be retired earlier due to the unbalanced block utilization. In addition, the greedy algorithm needs to record the number of invalid pages of the block, and therefore it also requires a lot of memory space to keep the necessary information. SW also uses the greedy algorithm, and it needs another block erasing table (BET) to identify which block has been erased during a given time period. Thus SW requires more memory space than GA. In addition, SW requires tuning the threshold parameter very carefully. For example, in Fig. 4(a), if the threshold of SW is changed to 10, the growth of the standard deviation will be ended at 28.3, which is much smaller than the case of that threshold equals to 100. In Fig. 4(b), if the threshold of SW is changed to 10, since the maximum block erase count is 246, as shown in Table I, the bad block number will be zero after 200 GB data written. As a result, SW demands more design complexity and memory spaces than the proposed SGC algorithm.

Table I. Performance Comparisons.

	GA	SW(TH=100)	SW(TH=10)	SW(TH=5)	SGC1	SGC2
Maximum erase count	831	788	246	212	186	195
Average erase count	125.7	126	129.1	132.6	185.1	148.5
Standard deviation	279.2	106.5	28.3	17.5	0.22	12.7
Living page copy(%)	100	100.7	106.7	113.7	216.3	136.9
RAM space(bit)	block number*17	block number*(17+1)	block number*(17+1)	block number*(17+1)	0	block number*1

The proposed SGC algorithm uses the IPF vector to records whether a block contains more than 75% invalid pages, and thus it has the lowest memory space requirement than the other approaches. In addition, the proposed SGC algorithm can achieve the lowest maximum erase count and

DOI: 10.1587/elex.9.1874
Received October 22, 2012
Accepted November 06, 2012
Published December 28, 2012

standard deviation as compared with other approaches. As a result, the lifetime of the flash memory can be greatly lengthened with the proposed SGC algorithm.

5 Conclusion

In this paper, a low-complexity high-performance wear-leveling algorithm for flash memory system design is presented. The proposed SGC algorithm combines GC operation and wear-leveling into a single process, and it doesn't require any tuning threshold parameter. Simulation results show that the proposed SGC algorithm has the lowest maximum block erase count and smallest standard deviation. In addition, the low-complexity low-cost SGC algorithm makes it is easy to be implemented by firmware-based or hardware-based approaches. Thus the lifetime of the flash memory can be greatly lengthened by the proposed SGC algorithm.

Acknowledgments

This work was supported in part by the National Science Council of Taiwan, under Grant NSC100-2221-E-194-051. The EDA tools supported by the National Chip Implementation Center are acknowledged as well.

DOI: 10.1587/elex.9.1874
Received October 22, 2012
Accepted November 06, 2012
Published December 28, 2012