IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 12, December 2016

A Study on Efficient FTL Algorithms for Flash Memory Systems

Sunil Kim[1] and Jun-Yong Lee[1]

Department of Computer Engineering, School of Engineering, Hongik University, Seoul, Korea[1]

Abstract: This paper represents a survey on NAND-type flash memory and FTL algorithms which are used to improve the performance of flash memory. Flash memory is widely used as data storage and various embedded devices nowadays. The most attractive feature of flash memory is that the stored data is non-volatile and can be updated erase/write operation. But, every block in flash memory has a limited write/erase cycle. In order to manage these problems, a software module called FTL (Flash Translation Layer) algorithm is used. Various FTL algorithms have been proposed. This paper surveys the characteristics of those algorithms and their features.

Keywords: Computer Architecture, Embedded system, FTL, NAND flash memory.

I. INTRODUCTION

Flash memory is non-volatile memory device which can be erased and reprogrammed electrically. Flash memory was evolved from EEPROM (Electrically Erasable Programmable Read Only Memory). There are two main types of flash memory, which are NAND-type flash memory and NOR-type flash memory. NAND-type flash memory can be written and read in blocks or pages, while NOR-type flash memory allows a single word or byte to be written or read independently.

The NAND-type is primarily used in main memory, USB devices, solid-state drives and other general purpose storage devices. The NOR-type is used as a replacement of EPROM or certain kinds of ROM due to the characteristics of random access capability.

This paper surveys on the various schemes of FTL (Flash Translation Level) for flash file system regarding the NAND-type flash memory, as NAND-type flash memory is mostly used computer memory devices nowadays.

FTL is the driver which works in conjunction with an existing operating system to enable flash memory appear to the system like a general disk drive. FTL creates virtual small blocks of data or sectors out of flash memory's large blocks. Then it manages data on the flash so that it appears to be written in place when it is being stored in the different spots in the flash.

II. NAND MEMORY OVERVIEW

The most popular flash memory cell is based on the Floating Gate (FG) technology. A MOS transistor is built with two overlapping gates rather than a single one. The first one is completely surrounded by oxide, while the second one is contacted to form the gate terminal. The isolated gate constitutes a trap for electrons, which guarantees charge retention for years.

The operation performed to inject and remove electrons from the isolated gate are called program and erase, respectively. These operations modify threshold voltage V_{TH} of the memory cell, which is a special type of MOS transistor. Applying a fixed voltage to cell's terminals, it is then possible to discriminate two storage levels of one and zero

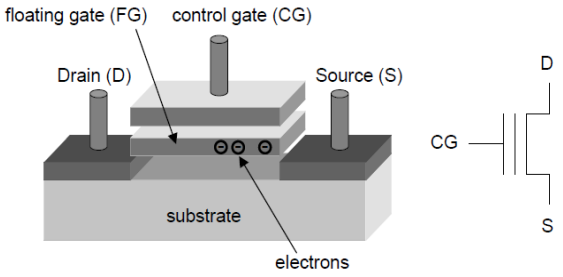

Fig. 1. Structure of NAND memory cell and its symbol

A. NAND memory array

The memory cells are packed to form a matrix in order to optimize silicon area occupation.

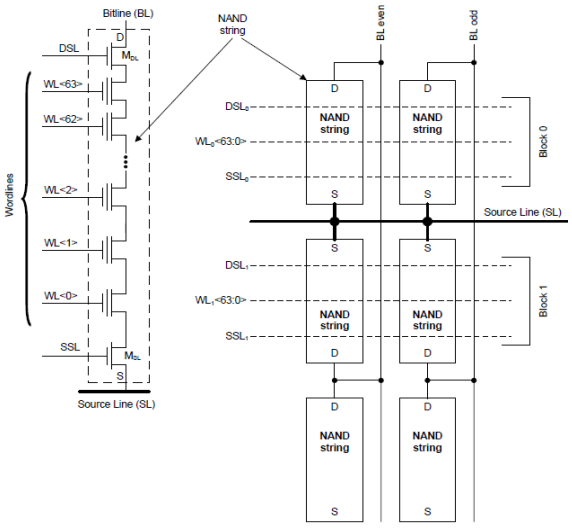

Fig. 2. NAND string and NAND array

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 12, December 2016

In the NAND string, the cells are connected in serried, in groups of 32 or 64. Two selection transistors are placed at the edges of the string, to ensure the connections to the source line (through M_{SL}) and to the bit line (through M_{DL}). Each NAND string shares the bitline contact with another string. Control gates are connected through wordlines (WLs).
Logical pages are made up by cells belonging to the same wordline. The number fo pages per wordline is related to the storage capabilities of the memory cell. Depending on the number of storage levels, flash memories are referred to in different ways: SLC memories store 1 bit per cell, MLC memories store 2 bits per cell, 8LC stores 3 bits per cell and so on.

B. Basic operations
When we read a cell, its gate is driven at V_{READ} (0 V), while the other cells are biased at $V_{PASS,R}$ (usually 4~5 V), so that they can act as pass-transistors, regardless the value of their threshold voltages. In fact, an erased flash cell has a V_{TH} smaller than 0 V; vice versa, a written cell has a positive V_{TH} but, however smaller than 4 V. In practice, biasing the gate of the selected cell with a voltage equal to 0 V, the series of all the cells will conduct current only if the addressed cell is erased.
Programming of NAND memories exploits the quantum-effect of electron tunneling in the presence of a strong electric field. In particular, depending on the polarity of the electric field applied, program or erase take place. During the programming, the number of electrons crossing the oxide is a function of the electric field: in fact, the grater such field is, the greater the injection probability is. Thus , in order to improve the program performances, it is essential to have high electric fields available and therefore high voltages. This requirement is one of the main drawbacks of this program method, since the oxide degradation is impacted by these voltages.
NAND memory is placed in a triple-well structure. Usually, each plane has its own triple-well. The source terminal is shared by all the blocks. In this way the matrix is more compact and the multiplicity of the structures which bias the iP-well is drastically reduced. The electrical erase is achieved by biasing the iP-well with a high voltage and keeping grounded the wordlines of the sector to be erased. Therefore, NAND technologies do not need negative voltages.

C. Logic organization
NAND memory contains information organized in a specific way: pages and blocks. A block is the smallest erasable unit. Generally, there are a power of two blocks within any device. Each block contains multiple pages. The number of pages within a block is typically multiple of 16. A page is the smallest addressable unit for reading and writing. Each page is composed of main area and spare area. Main area can range from 4 to 8 kB or even 16kB. Spare area can be used for CC and system pointers and it is in the order of a couple of hundreds bytes every 4k of main area.

III. THE FLASH TRANSLATION LAYER

Among the total capacity of NAND flash memory, less than four percent of memory is hidden from the consumers for implementing the FTL algorithm. This hidden capacity includes the spare area and some spare blocks given to the FTL programmers to use accordingly. Thus, the number of spare blocks needed for the FTL algorithm is different, and the performance evaluation can also be dramatically affected on the number of spare blocks.

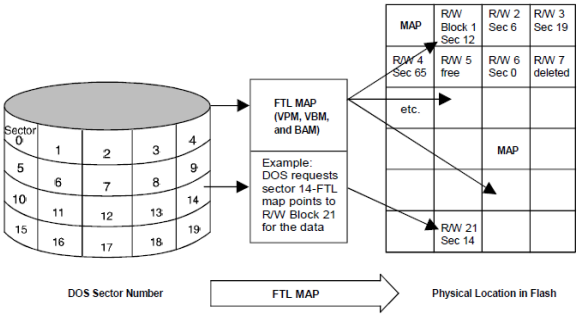

Fig. 3. Mapping process of flash memory

The methods of utilizing the spare blocks for enhancing the performance and durability of flash memory is decided by a software layer called FTL. The basic role of FTL is explained by the general organization of ND flash memory system.

The host system views the flash memory as a hard disk-like block device, and thus the file system issues read or write commands with logical addresses to read from, or to write data to, specific addresses of flash memory. The physical addresses corresponding to the logical addressesare determined by a mapping algorithm of FTL. During the address translation, FTL looks up the address mapping table. The mapping table is stored in SRAM, a fast but expensive memory, which is used for mapping logical addresses to physical addresses in the unit of sector or block.

When issuing overwrites, FTL redirects the physical address to an empty location, thus avoiding the erase operations. After managing an overwrite, FTL changes the address mapping information in SRAM. The outdated block can be erased later. The techniques of adding functions for flash memory in the file system have been used in the past. However, for compatibility with the existing file system, the tendency is moving in the direction of having a device driver as a separate layer.

IV. PERFORMANCE IMPROVEMENT USING FTL ALGORITHMS

The FTL algorithm has three functions coordinating with each other. They are basic mapping, performance enhancement and durability enhancement. Implementing each FTL algorithm can be slightly different depending on the small block and large block flash memory. In this

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 12, December 2016

section, the FTL algorithms based on the small block flash memory are discussed in detail.

A. Basic mapping algorithm

The mapping technique is the process of mapping the physical blocks to the logical blocks for achieving easy and efficient data access to the block. The algorithms are categorized as sector mapping algorithm, block mapping algorithm and hybrid mapping algorithm.

The sector mapping algorithm is a naive and intuitive FTL algorithm. In the sector mapping, every logical sector is mapped to a corresponding physical sector. Therefore, if there are n logical sectors seen by the file system, then the row size of the logical to physical mapping table is n.

As the sector mapping algorithm requires a large amount of RAM, it is hardly suitable for small embedded system. To overcome this memory issue, Takayuki proposed the block mapping algorithm. The basic idea of block mapping is to contain the mapping table in the unit of block.

In the block mapping algorithm, the row size of the logical to physical mapping table corresponds to the number of blocks in flash memory. Many performance enhancing algorithm use the block mapping technique, since it requires small size mapping information space. However, if the file system issues many write commands with identical LSNs, then this will lead to severe performance degradation due to write and erase operations.

The hybrid mapping is introduced due to the disadvantages of both sector and block mapping algorithms. The hybrid mapping uses both block mapping and sector mapping. First, it uses block mapping to achieve the corresponding physical block, and then it uses a sector mapping to locate an available empty sector within the physical block.

B. Performance enhancement

The unit of mapping algorithm can be defined in terms of sectors, pages and blocks. As the granularity of the mapping table gets finer, the performance increases while the cost of RAM also increases as mentioned earlier. On the other hand, the coarser granularity gives lower cost of RAM, but degraded performance. It is the duty of performance enhancing algorithms to identify the methods to increase the performance while decreasing the cost of RAM. Therefore, the basic mapping algorithms are modified to efficiently utilize the spare blocks in order to reduce overall number of read/write/erase operations.

Previous performance enhancing algorithms are log-sector scheme, log-block scheme, and state-transition scheme. They are based on the partial programming, basic mapping algorithms, and static/dynamic allocation, which were discussed before.

C. Durability enhancement

Every block in flash memory has a program/erase cycle limit. When a block reaches the program/erase cycle limit, the block wears out and the data on that block cannot be reliable. Therefore FTL helps to extend the life of flash memory by reducing the total number of erase operations and by evenly distributing the erase operations to all the blocks.

Previous works have not clearly demarked the differences between wear-leveling algorithms and cleaning policies. Some have mixed the terms together while others separated each other with very unclear demarcation. The two terms can be separated depending upon their level of monitoring.

The wear-leveling algorithms are characterized by the block-level monitoring whereas the cleaning policies with sector/page-level monitoring. If the level of monitoring gets finer, the extra read/write operations increases thus degrading the performance and durability of flash memory.

V. CONCLUSION

In this study, we surveyed the characteristics of NAND-type flash memory and various features of FTL algorithms. There have been many studies on flash memories, especially on NAND-type flash memory due to its low electric power, non-volatile storage, high performance, physical stability and portability.

We have explained the hardware organization of small-block and large-block NAND-type flash memory and classified several functions of FTL algorithms. Further study on FTL algorithms to improve the performance of flash memory can be achieved on the basis of this survey.

ACKNOWLEDGMENT

This work was supported by 2014 Hongik University Research Fund.

REFERENCES

[1] G. Campardo, R. Micheloni, D. Novosel, “VLSI-Design of Non-Volatile Memories,” Springer-Verlag, 2005.
[2] A. Kawaguchi, S. Nishioka, and H. Motoda, “A Flash-Memory Based File System,” Proceedings of the USENIX Winter Technical Conference, pp 15-164, 1995.
[3] J. Kim, J. M. Kim, S. Noh, S. L. Min, and Y. Cho, “ A Space-Efficient Flash Translation Layer for Compact Flash System,” IEEE Transactions on Consumer Electronics, Vol. 48, No. 2, p. 366-375, May 2002.
[4] S.-W. Lee, D.-J.Park, T.-S.Chung, D.-H. Lee, .-W Park, and H.-J. Song, “FAST: A Log-Buffer Based FTL Scheme with Fully Associative Sector Translation,” 2005 US-Korea Conf. on Science, Technology and Entrepreneurship, August 2005.
[5] T. Tanzawaet. al, “A Compact On-Chip ECC for Low Cost Flash Memories, IEEE Journal of Solid-state Circuits Vol. 35, No. 11 pp. 662-669, May 1997.
[6] www.sdcard.com
[7] Samsung Electronics (2010) Nand flash memory.K9GAG08U0M data book.