Fast NAND Flash Memory System for Instruction Code Execution

Bo-Sung Jung, Cheong-Ghil Kim, and Jung-Hoon Lee

The objective of this research is to design a high-performance NAND flash memory system containing a buffer system. The proposed instruction buffer in the NAND flash memory consists of two parts, that is, a fully associative temporal buffer for temporal locality and a fully associative spatial buffer for spatial locality. A spatial buffer with a large fetching size turns out to be effective for serial instructions, and a temporal buffer with a small fetching size is devised for branch instructions. Simulation shows that the average memory access time of the proposed system is better than that of other buffer systems with four times more space. The average miss ratio is improved by about 70% compared with that of other buffer systems.

Keywords: NAND flash memory, instruction characteristics, buffer system, embedded applications.

I. Introduction

Recently, NAND flash memory has been widely used for data storage in various mobile applications. Due to its nonvolatility, stability, economical feasibility, low power usage, durability, fast speed, and high density, NAND flash memory is utilized in storage devices for desktop computers [1].

Nonvolatile memory can be divided into two categories: NAND flash memory and NOR flash memory. They are similar in terms of their cell structure for byte storage, but there are pros and cons to each of them due to the difference in the composition of their cell assembly [2]. The most distinguishing feature between NAND and NOR flash memory is the initial access time. The initial access time of NAND is 25 μs, whereas that of NOR, depending on its reading mode, is approximately 50 ns to 100 ns; this is approximately a 250- to 500-fold difference between the two [3]. After the initial access, both NAND and NOR flash take a similar amount of time for sequential data reading. This means that NAND has a great disadvantage in terms of indirect (random) access time.

Due to these features, NOR flash memory is mainly used to store instruction codes for operation, whereas NAND is used for data storage. However, NAND does have more economical benefits. It is approximately 30% to 40% less expensive than NOR flash memory. Therefore, there is ongoing research into enhancing the access time of NAND flash using static RAM (SRAM) and/or synchronous dynamic RAM (SDRAM).

The most important difference between designing an instruction cache for CPUs and a RAM buffer of NAND is their reading mode. In the case of the former, a read operation may require the identical time for both hit and miss processing. However, in the case of the latter, two different read modes must be considered: random and serial. When a serial page read is executed and the page is not found in the buffer, the page must exist in the 2-KB read/write register. This means that unlike an instruction cache miss, this buffer miss can guarantee fast access time. However, the random read operation is enabled when an address is generated over the previous page boundary. If the random page read is not found in the buffer, the system has a long accessing latency. Therefore, it is a major factor to reduce the number of random access in designing an instruction buffer system of NAND flash.

The objective of this research is to design high-performance NAND flash memory architecture including an instruction

Manuscript received Nov. 9, 2011; revised May 7, 2012; accepted July 9, 2012.

This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (No. 2011-0009368).

Bo-Sung Jung (+82 55 772 1747, blueking80@gnu.ac.kr) and Jung-Hoon Lee (corresponding author, leejh@gsnu.ac.kr) are with the School of Electrical and Electronic Engineering, ERI, Gyeongsang National University, Jinju, Rep. of Korea.

Cheong-Ghil Kim (cgkim@nsu.ac.kr) is with the Department of Computer Science, Namseoul University, Cheonan, Rep. of Korea.

http://dx.doi.org/10.4218/etrij.12.0211.0472

buffer as a replacement for NOR flash. If requested instructions are found in a buffer, then they can be read from fast SRAM/SDRAM. As a result, the execution speed of the flash memory becomes very fast. To increase the buffer hit rate, we propose a new instruction buffer system and mechanism to improve performance.

II. Related Work

Among the studies utilizing the existing buffer system, Jung and Lee [4] and Huang and others [5] enhanced the performance of the overall system by applying a buffer system to the existing flash memory to reduce the number of write operations, a high number of which causes the biggest bottleneck in a flash memory system. A smart buffer [4] consists of a buffer with a large block size fetched from the flash memory and a buffer that stores data discarded from the cache memory. A smart buffer reduces the write operation frequency of the flash memory and guarantees fast memory access time. On the other hand, Huang and others [5] utilized one writing buffer and two reading buffers to decrease the write operation frequency and improve system performance. Data requests from the write operation are stored in the write buffer, and read operation requests are stored in the read buffer. However, these works only focused on data.

Park and others [6] created a flash memory package for managing instructions by replacing the NOR flash memory with a NAND flash memory and SRAM based on the XIP (execute in place) method. This architecture consists of SDRAM and a buffer with a victim cache structure. Either the victim buffer or SDRAM is chosen for storage, based on the reference pattern of the flash memory cell. These reference patterns are analyzed through a profiling process, and they are stored in a spare area in the flash memory. That is, the pages with high priority are stored in the victim buffer, and the pages with low priority are stored in SDRAM. The difference between our mechanism and the approach in [6] can be summarized as two distinct points. One is the way of utilizing localities. For this purpose, we propose a dual buffer with two different block sizes, whereas [6] proposed the profiler to detect the locality (priority). The other is the architectural difference. Our architecture is only constructed with a small fully associative buffer (for example, 8 KB), but the system of [6] consists of three parts, that is, a 32-MB SDRAM, a 64-KB victim buffer, and flash spare areas with priority information.

As previously mentioned, although much has been achieved in enhancing flash memory performance through the use of a buffer in NAND flash data, there is a great lack of research on NAND flash instruction.

III. Proposed NAND Flash Memory System

Existing typical NAND flash memory is composed of a flash memory cell and a read/write register for a page. Specifically, in NAND flash memory, read and write operations are executed by page unit, and the serial access of such operations takes 20 ns. This access time is 1,000 times faster than the random access time of approximately 20 μs. Unlike data flash, instruction flash carries out read operations only; therefore, it can effectively improve system performance by reducing the random access time. Basically, instructions have a greater spatial locality due to sequential reading while the program is running. What is special about spatial locality is that it can take advantage of a large fetching size very effectively [7]. Existing NAND flash memory can use spatial locality effectively because it has one read/write register. However, because there is a single register, the branch instruction must reaccess the flash memory, and this is a shortcoming.

Figure 1 shows the average memory access time (AMAT) of flash memory when various types of buffers are applied to NAND flash. The buffer size in each structure is only 4-KB SRAM, and the fetching size from a read/write register to the buffer is 32 B, which is the minimum size in each buffer. As found in Fig. 1, the main advantage of NAND flash memory with a small conventional buffer is that a small conventional buffer can reduce the AMAT of the existing NAND flash memory without a buffer system, but it shows a large performance gap compared with the performance of the existing NOR flash. The drawback of the conventional buffer structure is that a unified buffer with one block size cannot effectively take advantage of the locality inherent in the reference stream of a specific application. In general, two primary types of locality are available in programs, and the degree of efficiency on utilizing them mainly depends on the execution characteristics of programs. However, most unified buffer systems have shown a tendency to exploit only one or

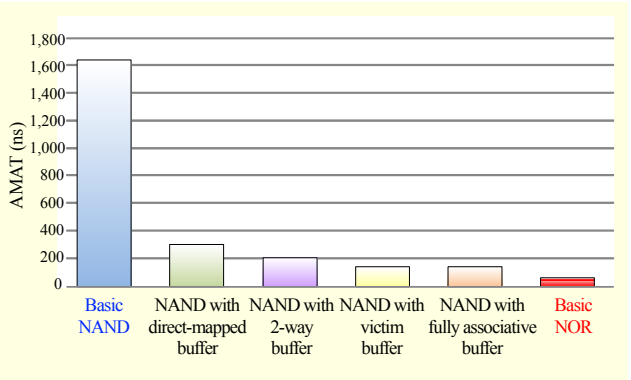

Fig. 1. Performance improvements of NAND flash memory with various buffer structures.

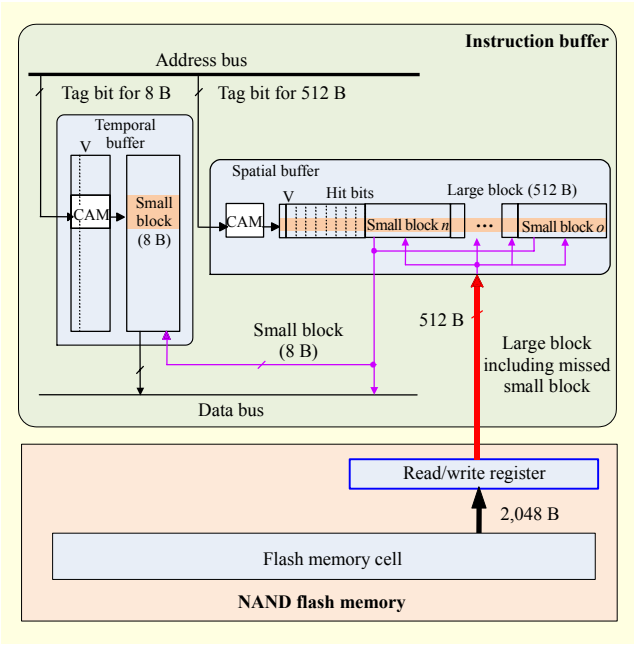

Fig. 2. Proposed NAND flash memory system with instruction buffer.

the other of them because each of them may require contradictory requirements on the structure of the hardware. Therefore, we propose a NAND flash memory system consisting of two separate fully associative buffers for temporal and spatial locality.

Figure 2 shows the proposed NAND flash memory system containing the instruction buffer system. This architecture is divided into three main parts: standard NAND flash memory with a 2-KB read/write register, spatial buffer, and temporal buffer. NAND flash memory is the same as the standard flash structure, and each page is set at 2 KB in accordance with the current trend of mass storage. The dual-buffer architecture for instruction consists of spatial and temporal fully associative buffers. A spatial buffer can have a large fetching size to effectively utilize spatial locality, the basic feature of instruction, while a temporal buffer selectively saves referenced blocks. Such a temporal buffer can enhance system performance considerably, especially in terms of branch instructions. The fetching size of the spatial buffer is set to be approximately n times greater than that of the temporal buffer.

The fetching size of the temporal buffer is 8 B and the spatial buffer is made up of n×8 B for more efficient spatial locality. In addition, the spatial buffer has n hit bits per each entry for saving referenced instructions onto the temporal buffer selectively. Here, we consider a large fetching size of the spatial buffer to be 512 B. Therefore, in a fetching entry of the spatial buffer, there will be a total of 64 hit bits for the temporal buffer.

When an instruction is demanded from the flash memory, the dual instruction buffer is accessed first. If access to the spatial buffer is a hit, a hit bit of a corresponding small block (for example, 8 B) is renewed; that is, the hit bit is set as 1, and the requested instructions are sent to the CPU. In addition, upon a temporal buffer access hit, the requested instructions are directly sent to the CPU without any further actions.

If an access misses both buffers, the 512-B page including the requested instruction is stored in the spatial buffer from the flash memory. Here, the hit bit of the referenced small block within the fetching size of the spatial buffer is renewed as 1. If the spatial buffer is not full, the page is stored in the spatial buffer without removing an old page and the corresponding hit bit is set to 1. However, if all the entries in the spatial buffer are valid, one entry is chosen, based on the FIFO (first-in, first-out) algorithm. The referenced small blocks in one replaced large block are selectively transferred to the temporal buffer; only the small blocks whose hit bit is set as 1 in the spatial buffer are selectively transferred to the temporal buffer.

IV. Performance Evaluation

The media benchmark [8] is used to evaluate the system performance for multimedia devices. To create traces of these benchmarks, Simplescalar 3.0 [9] is used to monitor the referenced address while hundreds of millions of instructions are processed. NAND flash memories with various buffer systems are evaluated in this simulation. Simulation parameters are shown in Table 1.

For the performance test of the NAND flash memory, a hybrid mapping algorithm is used. The proposed flash memory page size is 2 KB and the block size is 128 KB, resulting in a total of 64 pages. Using the buffer access miss ratio and AMAT, we analyze and compare the proposed buffer system to other buffer systems with NAND flash memory.

We select a direct-mapped buffer, 2-way associative buffer, victim buffer, and fully associative buffer for comparison in the

Table 1. Simulation parameters.

System parameter	Value
Page size	2 KB
Block size	128 KB (64 pages)
NOR random read time	70 ns/B
NOR page read time	25 ns/B
NAND random read time	25 μs/page
NAND serial read time	25 ns/B
SRAM buffer read time	20 ns/B

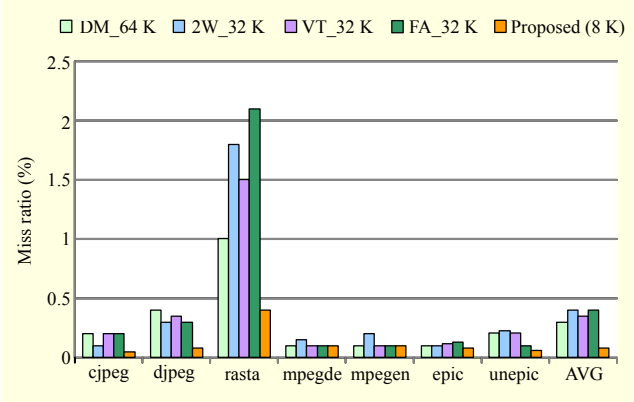

Fig. 3. Buffer miss ratios.

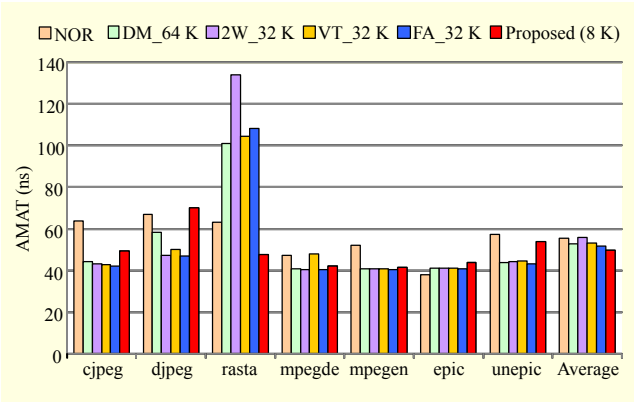

Fig. 4. AMAT.

performance analysis. For the selected buffers, the most effective buffer size and fetching size are selected based on previous simulations carried out on a buffer size ranging from 8 KB to 256 KB and a fetching size ranging from 32 B to 2,048 B. In conclusion, in the case of flash memory, a buffer with a small block performs better than a buffer with a large block. This result differs from the characteristics of conventional instruction caches. Therefore, 32 B is selected as the optimal fetching size for having the best AMAT analysis results. The buffer miss ratios for the conventional buffers and the proposed buffer are shown in Fig. 3. The block size of the conventional buffers is set as 32 B and the optimal buffer sizes are chosen as follows: 64 KB for the direct-mapped buffer (DM_64 K) and 32 KB for the 2-way associative buffer (2W_32 K), victim buffer (VT_32 K), and fully associative buffer (FA_32 K).

The proposed buffer is constructed using a 4-KB temporal buffer with a block size of 8 B and a 4-KB spatial buffer with a block size of 512 B. Notice that the proposed buffer for a given size performs better than the conventional buffer with a buffer size of four or eight times as much space. Figure 4 shows the simulation results regarding AMAT.

In summary, the proposed buffer system shows a 70% lower buffer access miss ratio compared with that of other buffer systems. Additionally, in terms of AMAT, the proposed system performs better than others with four times more space.

V. Conclusion

The goal of this research was to design a fast NAND flash memory system to store instruction codes for operations. To achieve this goal, a new buffering mechanism for exploiting two types of locality effectively and adaptively was designed, including a temporal buffer with a small block size for exploiting temporal locality and a fully associative spatial buffer with a large block size for exploiting spatial locality. A selective mechanism was used to maximize the effect of temporal locality. Our results show that the NAND system enables the buffer size to be reduced by a factor of four to eight relative to that of conventional buffers while performing similarly. The use of a small buffer results in a significant reduction in power consumption. Furthermore, buffer miss may lead to higher power consumption as a result of accessing the flash memory cell.

References

[1] J. Park et al., "A Hybrid Flash Translation Layer for SLC-MLC Flash Memory Based Multibank Solid State Disk," *Microprocessors Microsyst.*, vol. 35, no. 1, Feb. 2011, pp. 48-59.

[2] Wikipedia, "Flash memory 2010." http://en.wikipedia.org/wiki/Flash_memory

[3] Samsung Elec., NAND Flash Memory & Smart Media Data Book, 2010.

[4] B. Jung and J. Lee, "Flash Memory System with Spatial Smart Buffer for the Substitution of a Hard-Disk," *J. Korea Soc. Computer Inf.*, vol. 14, no. 3, Mar. 2009, pp. 41-49.

[5] W. Huang et al., "Energy-Efficient Buffer Architecture of Flash Memory," *Proc. Multimedia Ubiquitous Engineering*, 2008, pp. 543-546.

[6] C. Park et al., "A Low-Cost Memory Architecture with NAND XIP for Mobile Embedded Systems," *Proc. CODES-ISSS*, 2003, pp.138-143.

[7] G. Park et al., "A Way Enabling Mechanism Based on the Branch Prediction Information for Low Power Instruction Cache," *IEICE Trans. Fundamentals Electron., Commun., Computer Sci.*, vol. E92-C, no. 4, Apr. 2009, pp. 517-521.

[8] J. Rucker, *MediaBench II: A Tool for Evaluating Multimedia Workloads on Embedded Systems Mediabench*, University of California, Los Angeles, CA, USA, 2001. http://euler.slu.edu/~fritts/mediabench/

[9] Simplescalar. http://www.simplescalar.com