P1-9

Spread Programming using Orthogonal Code for Alleviating Bit Errors of NAND Flash Memory

JaeGyu Jung, SangBum Suh and Chuck Yoo

Abstract**-- As flash memory requires higher density, bit error rate becomes more important. In this paper, we propose Direct Sequence Spread Spectrum (DSSS) based spread programming that makes flash based storage to be tolerant against errors.**

I. INTRODUCTION

As mobile embedded systems evolve into data centric, the popularity of high-density flash memory as data storage media has increased steadily for a wide spectrum of mobile embedded systems such as PDAs, MP3 players, mobile phones and digital cameras. To achieve high-density, sharing method of single medium by separating it into several levels is designed. Multi Level Cell(MLC) technology accommodates multiple bits to be stored into a single cell, but the complexity of software that detects and corrects bit errors should be increased to compensate for a larger bit error rate. In this paper, we propose Direct Sequence Spread Spectrum (DSSS) based spreading technology to alleviate bit errors. The parallelism makes it possible to adopt DSSS into flash based storage.

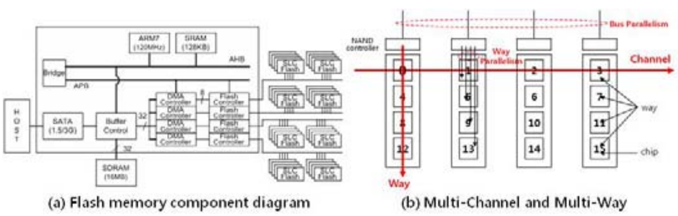

Fig. 1. NAND flash based storage consists of several flash memory chips. Each NAND controller has one external bus to access data and several internal buses connected with chips. The external bus is called channel and the internal bus is called way.

Fig. 1 shows the components and internal architecture of flash based storage. For improving the performance of flash based storage, multi-way and multi-channel interleaving architecture that accesses several flash memory chips simultaneously is explored [1].

Due to the parallelism, the storage should process several physical pages at the same time and we call these logically grouped pages a superpage. This implies that reading and programming operations will be done as the unit of superpage and internal fragmentation cannot be avoidable because flash memory cannot overwrite the same page before erasing. Fig. 2 shows how flash based storage causes internal fragmentation. However, we utilize this parallel property to enable spread programming.

II. SPREAD PROGRAMMING

Spread spectrum technology is used to establish secure communication, increase resistance to natural interference and

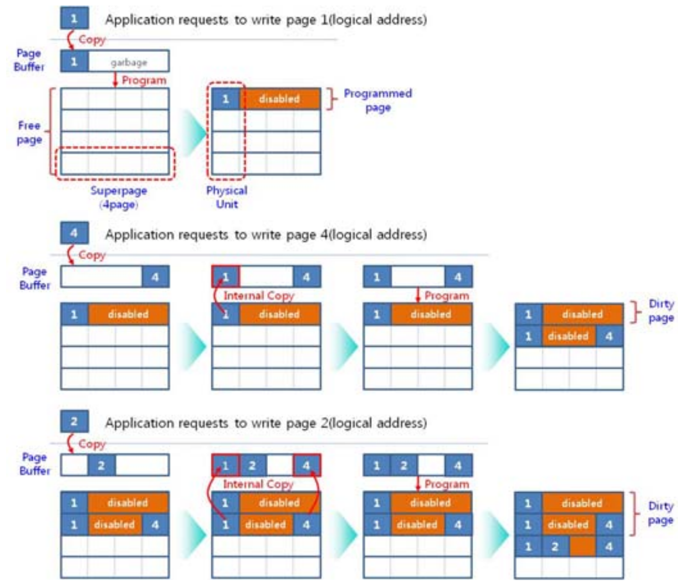

Fig. 2. Even though logical page 1, 2, 3, and 4 can be grouped as a single superpage, they cannot be appended to already programmed page if the writing request is called separately. For space efficiency, flash based storage merges logical pages into newly programmed superpage.

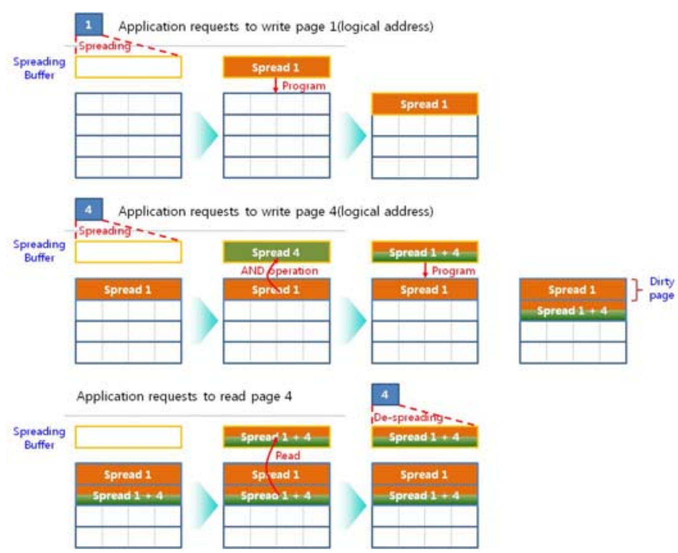

Fig. 3. Logical page 1 should be spread by multiplying it by assigned code and programmed. If logical page 4 is requested to be programmed, it also should be spread and merged with previous spread page 1 by adding it to spread page 4.

so on. In this paper, we adopted DSSS, one of the spread spectrum technologies [2]. DSSS multiplies the data by a pseudo-noise code and generates a noise-like spread data. Spread data can be used to exactly reconstruct the original data later by multiplying it by the same pseudo-noise code. Moreover, DSSS let multiple users can share single data

978-1-4244-4316-1/10/$25.00 ©2010 IEEE

medium, so it is applied to code division multiple access (CDMA). We adopted DSSS due to its two benefits, resistance to external interference and sharing of a single medium among several users. We applied *walsh code*, the representative of orthogonal code, to spreading original data. Orthogonal codes have a cross-correlation equal to zero, in other words, they do not interfere with each other.

A superpage consists of several pages and each page is assigned different walsh codes and will be multiplied by assigned code before programmed. DSSS can accumulate all spread pages belonged to the same superpage into a single superpage, even though the size of each spread page is same as that of the superpage. The multiple access property of DSSS makes it possible to distinguish each page from the accumulated superpage. When application requests a page, the storage loads the superpage it belonged, and de-spread it by multiplying with assigned code. Fig. 3 shows how we spread and merge logical pages and extract the original data from the spread data.

III. PLUS AND MINUS

A. Advantages

Two properties of DSSS make it to be tolerant against interference. One is spreading code itself. Fig. 4 shows how spreading code overcomes bit errors in spread data. Because DSSS represents each data bit with spreading code, if bit errors do not exceed the half of spreading code, data bit can be recovered during de-spreading.

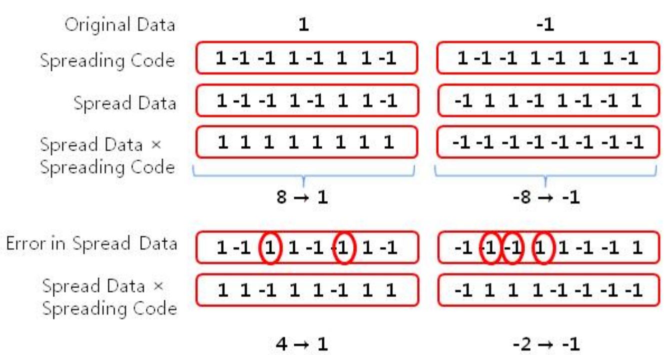

Fig. 4. 8bit spreading code can endure 3bit error in spread data. As the size of superpage is increased, the tolerance against bit error is also raised because the size of spreading code is also increased.

Scattering effect due to spreading is the other assistance for alleviating bit errors. In general, bit errors tend to concentrate on specific bit arrays or specific physical units. Without spreading, concentrated bit errors may corrupt the data of specific page entirely, but we can disperse the influence of errors in all pages belonged to the superpage and we have additional chance to correct them by using data correction algorithm at upper layer.

The tolerance against bit error can extend the durability of flash based storage, because it allows the longer utilization of bad blocks rather than replaces them.

B. Disadvantages and their solutions

There are three weak points in our approach. First is the performance issue. As Fig. 3 shows, spread programming requires additional steps, spreading and de-spreading, before programming and after reading. Even if continuous pages are requested, all pages should be processed separately. Since DSSS requires relatively simple XOR and addition operation, however, we can minimize the overhead by adopting logic based H/W unit and parallelizing the process for all pages. Additionally, we can supplement performance degradation caused by complex error detection and correction, because DSSS decreases bit error rate.

Second issue is induced by partial update. With DFSS, update operation requires more steps than reading and programming, because we should subtract previous data from the superpage. We need to extract the previous data and spread it again for subtracting. To tackle this problem, we designed a novel Flash Translation Layer(FTL) that separates updated pages from the superpage and postpones the merge operation until other pages are also updated. This mechanism also improves the performance by preventing internal copies shown in Fig. 2.

Last problem is the breakdown of flash memory chips themselves because the durability of each chip is different. If some physical pages are broken, our approach may lose whole data, whereas traditional mechanism can restore data partially in live chips. Currently, we do not have the countermeasure against this problem except backup the whole data.

IV. CONCLUSION AND FUTURE WORK

This paper proposes a new programming method for NAND flash based storage. We adopt DSSS due to its error tolerant property and sharing of a single medium among several users. Even though the spreading increases the amount of data to be programmed, each logical page can be accumulated and extracted by multiplying assigned spreading code. Moreover, spreading code makes data to be robust against bit errors and scattering effect disperses the locality of error in all merged pages.

However, spread programming also has disadvantages. It requires additional steps to spread data and it is hard to update some portions of merged pages. Additionally, it may lose all data entirely, when some flash chips are broken. In the future, we will develop logic based operation unit that parallelizes all pages to be spread and de-spread simultaneously. And, we will develop a novel flash translation layer that supports partially updated pages and optimal merging mechanism.

REFERENCES

[1] J. Kang, J. Kim, C. Park, H. Park, and J. Lee, “A multi-channel architecture for high-performance NAND flash-based storage system,” Journal of System Architecture, vol. 53, no. 9, pp. 644-658, Feb. 2007

[2] R. Pickholtz, D. Schilling, and L. Milstein, “Theory of spread spectrum communications – a tutorial,” IEEE Trans. Commun., vol. 30, no. 5, pp.855-884, May. 1982