International Journal of Advanced Information Science and Technology (IJAIST) *ISSN: 2319:268*
Vol.2, No.4, April 2013 *DOI:10.15693/ijaist/2013.v2i4.152-155*

Improved Error Correction Capability in Flash Memory using Input / Output Pins

[1] A M Kiran and [2] J Shafiq Mansoor
[1]PG scholar, [2] Assistant Professor, ECE department, Karpagam University, Coimbatore-641021

***Abstract*-- Error control coding (ECC) is essential for correcting soft errors in Flash memories. In this paper we propose use of product code based schemes to support higher error correction capability. We also introduce input/output pin by which programming cycle reduces which inturn reduces the bit error rate. While these schemes have slightly larger latency and require additional parity bit storage, they provide an easy mechanism to increase the lifetime of the Flash memory devices.**

Index Terms—*Error correction codes (ECCs), flash memories,multi-level cell, product codes.*

1. OVERVIEW

Flash memory has become the dominant technology for non-volatile memories [1]. It is used in memory cards, USB flash drives, and solid-state drives in application platforms such as personal digital assistants, laptop computers, digital audio players, digital cameras and mobile phones. We focus on NAND Flash memories since they have lower erase times, less chip area per cell which allows greater storage density, and lower cost per bit than NOR Flash memories [2]. Specifically, we focus on multi-level cell (MLC) Flash memories which store two or more bits per cell by supporting four or more voltage states. These have even greater storage density and are the dominant Flash memory technology. There are some inherent limitations of NAND Flash memo- ries. These include write/read disturbs, data retention errors, bad block accumulation, limited number of writes [3]–[5], and stress-induced leakage current [6]. In recent years, due to cell size scaling, these issues have become critical . In particular, re-liability of MLC memory significantly degrades due to reduced gap between adjacent threshold levels. To enhance the reliability of NAND Flash memories and sup- port longer lifetimes, combinations of hardware and software techniques are used. These include wear leveling, bad block management and garbage collection. Wear leveling distributes the data to different physical locations so that all memory blocks are used approximately the same number of times [7]. Bad block management marks blocks once they show unrecoverable errors.

While these Flash management techniques increase the life time of Flash memories, they are not good at correcting soft errors. Error correction code (ECC) techniques, which can detect and correct errors by storing and processing extra parity bits, have now become an integral part of Flash memory design[9]. Single error detection/correction codes, such as Hamming codes, used to be sufficient to enhance the reliability of single-level cell (SLC) Flash memory systems [10]. In recent years, long linear block codes with high error correc- tion capability are used because the single error correction capability of Hamming code is no longer sufficient. The Bose-Chaudhuri-Hocquenghem (BCH) code and its subclass Reed-Solomon (RS) code are the best-known linear block codes for memories. Pipelined or bit-parallel BCH code has been used in [11]–[13]. Schemes based on concatenation of BCH codes and trellis coding modulation (TCM) have recently been proposed in [14]. While they reduce the error correction burden of a single BCH code, they require five (instead of four) threshold states per cell. ECC based on RS codes have been used in several commercial MLC Flash memories [15]–[17]. They use plain RS codes and can correct up to 24 errors in 512B, at the cost of larger hardware and coding latency. Clearly, higher error correction capability can be achieved by using stronger BCH or RS codes. However, it is expensive both in terms of area and latency. In this paper we propose use of product codes which use smaller constituent codes along rows and columns and achieve high error correction capability due to cross parity checking. Such codes have lower hardware overhead and have been successfully used in embedded SRAM caches [18] and interconnection networks [19]. An important factor in deciding on the ECC scheme is error characterization in terms of both type as well as distribution of errors. In current Flash memories, the error distribution is considered to be random. However with increased technology scaling, when the number of program/erase cycles is quite high, the probability of multiple bit upset (MBU) errors is likely to increase. This is because of the increased variation in threshold voltage which causes an increase in the probability of the long tailed threshold voltage distribution crossing over to the adjacent voltage states. To take these effects into consideration, we study the performance of the ECC schemes for two error models: fully random error model and hybrid error model with 90% random errors and 10% MBU errors. For these two error models, we present product code schemes that have better BER performance, lower area and smaller latency than single BCH and RS codes with comparable error correction capability.

2. INTRODUCTION

Flash memory devices can be found almost everywhere today. They are lighter, faster and more shock resistant than traditional magnetic hard drives. As this technology scales and the storage density increases, data errors become more prevalent, making error correction coding critical for maintaining data integrity. The storage density of a Flash memory device is dependent on the number of discrete voltage levels the floating gate cell is capable of representing. In early generations, every memory cell could represent two voltage levels and thus store a

single bit (SLC). The demand for increased storage capacity has created the need to store more than a single bit per cell by simply representing more than two voltage levels. In this work, we follow the commonly adopted nomenclature and assume that multiple level cell (MLC) chips store multiple bits per cell, and that in particular triple level cell (TLC) chips store three bits per cell.

Recently, the subject of error-correction coding for Flash memory has received significant attention. In [18], trellis coded modulation techniques were applied to Flash memory. In [13], the use of LDPC codes was investigated, and in [19] it was found that using soft information from multiple reads in the LDPC decoder lowered the error rate. In [9], algebraic error-correction codes were used for rewriting as well as for correcting errors. In [2], [5], [6], [12], codes that correct limited magnitude asymmetric errors were constructed. In [23], this model was extended to correct graded error patterns. In [17], constructions were given for single error-correcting codes that can correct limited magnitude errors in 2 directions. In [25], a different error model was considered where the likelihood of an error occurring was directly related to the value of the cell being programmed. The problem was to construct codes that maximized the size of a codebook given some fixed tolerable error probability.

In [11] a novel method of encoding information was introduced that reduced the occurrence of errors during programming. The error model in this work is motivated by data collected from a TLC Flash device. As observed in [24], if the information from each Flash cell is interpreted as a triple-bit word, then the errors (referred to as graded bit-errors) largely but not exclusively cause only a single bit in each word to change. From this observation, we suggest the use of a class of codes derived from tensor product codes [22] in the context of Flash memory. We refer to this class of codes as graded bit-error-correcting codes. The contribution of this work is to generalize the result of [24] to produce code constructions that correct errors that mostly have only a small number of bits in error for each cell-error. In fact, some of the proposed codes indeed end up having the same algebraic structure as generalized tensor product codes (cf. [10]). The novelty of this work is to show that for certain parameters of the constituent codes, such constructions can correct graded bit-errors.

Tensor product codes were first introduced in [22] and were generalized to produce efficient binary codes in [10]. In [20], these constructions were revisited and an efficient method of encoding was provided. More recently, tensor product codes were used in the context of magnetic recording [3], [4]. In a concatenated coding scheme, the use of a tensor product parity code as the inner code was shown to offer the performance advantages of a short length parity code but without the associated rate penalty. In [1], tensor product codes were used in conjunction with soft iterative decoding methods to manage the size of the syndrome table. In this work, a new type of generalized tensor product codes, the graded bit-error-correcting codes, is developed. These codes are demonstrated to correct the errors that occur within a TLC Flash device. In particular, generalized tensor product codes are shown to delay the onset of errors longer than conventional coding schemes. Delaying the onset of errors is significant since the device can potentially be used for a longer period of time.

3. ERROR MODELS

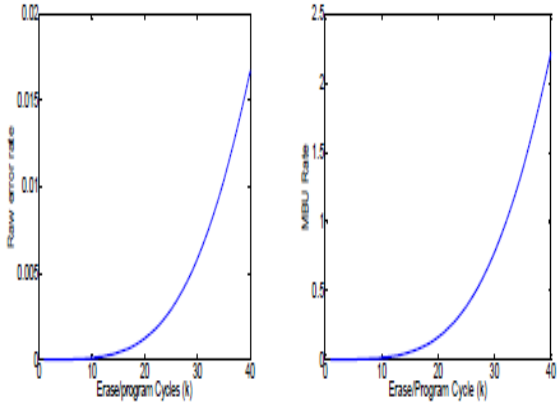
3.1 Error sources

There are many sources of errors in MLC Flash memories. Single event upset can be caused by charged particles due to sun activity or other ionization mechanisms [14]. Multi-bit upsets can occur due to a high-energy particle hitting at a low incident angle and striking many cells in a row. Furthermore, in MLC, the voltage window for threshold of each data state is smaller. Since all the programmed levels must be allocated in a predetermined sized voltage window, there is reduced spacing between adjacent programmed levels, making the MLC memories less reliable. Also, read/write operations in MLC memory can cause threshold voltage fluctuations, which inadvertently result in errors in consecutive bits [2]-[4].

Another important source of error is due to gradual charge leakage from the floating gate resulting in voltage shift in memory cells, ultimately resulting in a flip in the data stored in these cells. Blocks that have been erased many times have a shorter data retention life than blocks with lower erase/program cycles [2]-[4].

The number of errors due to program/erase wear out increases from 1* at 9000 cycles to 8* after 15000 cycles for MLC Flash [2].

With increased number of program/erase cycles, the number of MBU errors also increase as demonstrated through these simulations. First, using the results in [3][17], we model the distribution with a continuous Rayleigh distribution. The variance of the distribution is assumed to be a function of number of program/erase cycles and increases when the number of program/erase cycles increases. Thus for even Gray coded data, larger variance would result in MBU errors.



(a) (b)

Figure 1. (a) Raw BER and (b) MBU probability as a function of number of erase/program cycles.

In order to determine the variance as a function of the number of program/erase cycles, we match the error rate of

our model with experimental results for MLC Flash memory in [2]. Then, we use curve fitting to extrapolate the results for higher number of erase/program cycles. Figure 1(a) shows the BER curve versus number of erase/program cycles. Note that when the number of erase/program cycles increases from 23K to 27K, the raw BER increases from 2.2* to 4.0* . Figure 1(b) shows the MBU probability as a function of the number of program/erase cycles. This is approximately 2.3% at 40K erase/program cycles. Since the required endurance life time of NAND Flash memories is expected at least cycles [2], it is reasonable to expect that the burst error probability in MLC Flash will cross 10% towards the end of its rated lifetime.

3.2 Error models

We consider two error models: fully random error model and a model based on a mixture of random and MBU (or burst) errors. For burst errors, we assume that the probability of MBU decreases exponentially as the MBU size increases.

3.3 Performance metrics

We compare the different ECC schemes with respect to the following performance metrics:
Redundancy rate: In an (n, k) linear block code, redundancy rate is (n-k)/n. Hardware area: Area of encoder and decoder in ECC block.Encoding/decoding latency: Time for encoding/decoding data in one page. Bit error rate (BER): Number of received bits that have been altered due to errors, divided by the total number of bits.

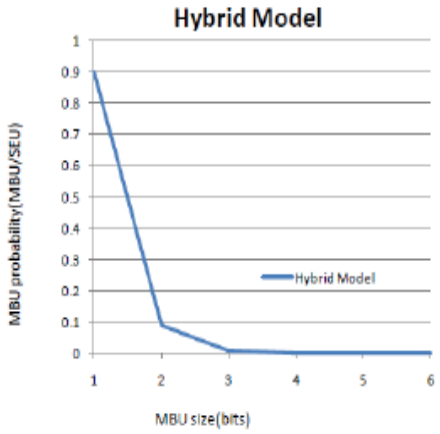


Figure 2. MBU probability as a function of MBU size

4. PRODUCT CODE ECC SCHEMES FOR FLASH MEMORY

A. Product Code Scheme:

Basics Product code is a technique to form a long length code with higher ECC capabilities using small length constituent codes. Compared to plain long length codes, it has high performance from cross parity check [25], and low circuitry overhead since the constituent codewords are of low error correction capability.

If code c1 has Hamming distance d1 and code c2 has Hamming distance d2, the minimum weight of the product code is exactly d1d2. Thus increasing the minimum weight of each code enhances the number of error patterns which can be corrected in the code array. Product code using single-error-correction codes in each dimension has been used in [18] and [19]. In [18], 8-bit even-parity code in both dimensions with bit interleaving has been used for SRAM caches of size 256 256 bits. In [19], 8-bit even-parity code has been used in interconnection networks. Both cases demonstrated the use of product codes for enhanced error correction performance.

In order to provide for high error correction capability in Flash memories, we propose to use a strong code with multiple error correction capability along at least one of the dimensions. Since data is stored along rows in memory, we propose to use stronger ECC along rows so that both random and burst errors can be dealt with efficiently. Furthermore, we choose a long codeword along this dimension to provide good coding performance.

We studied the performance of product codes based on BCH and RS codes. When long BCH/RS codes are used along the rows for high coding performance, for fixed page size, the length of the codeword along the rows is much shorter. Use of cyclic or linear block codes with multiple error correction capability along columns is an overkill and results in unnecessary hardware and latency overhead. So we choose Hamming codes along the columns; they have low overhead and provide enough coding gain for the product code based scheme.

B. Product Code Scheme: Encoding and decoding

Fig. 7(a) shows the encoding flow of the product code scheme, and Fig. 7(b) gives an example of the physical address mapping of RS (255, 247) +Hamming (72, 64) product code when the page buffer size is 16 kB. Note that the physical mapping is different for different product codes. We assume that the Flash controller has the capability to reallocate the storage space to support the different product codes. For the RS (255, 247)+ Hamming (72, 64) product code, during encoding, the RS encoder reads 247 information bytes at a time and generates 8 bytes or 64 bits corresponding to row parity. The row parity bits are stored in the pre-allocated region in the page buffer. Next, the Hamming encoder operates on the information and row parity bits, and generates the column and cross parity bits. The information bits are read with a stride of 247* 8, and the row parity bits are read with a stride of 8 * 8. After column encoding, the column&cross parity bits are stored in the corresponding section of the page buffer. In the allocation shown in Fig. 7(b), there is 64B unused space which can be used to store the beginning address of the different data regions for the Flash controller.

C. I/O pins:

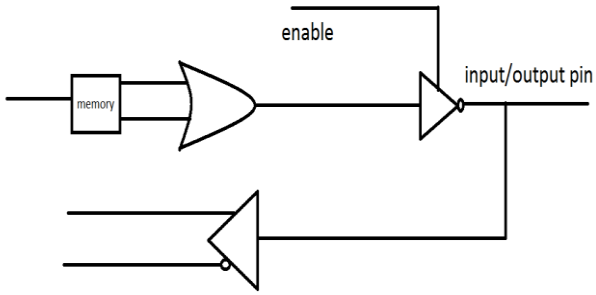


Fig.3 Input /output pins in flash memory

By using I/O pins we can reduce the program cycles. We can also disable the I/O pins using enable option leaving the output as high impedance state.

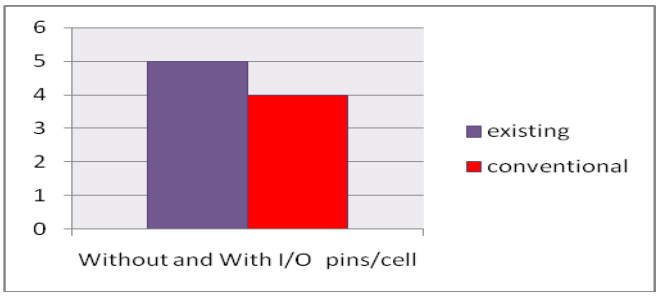
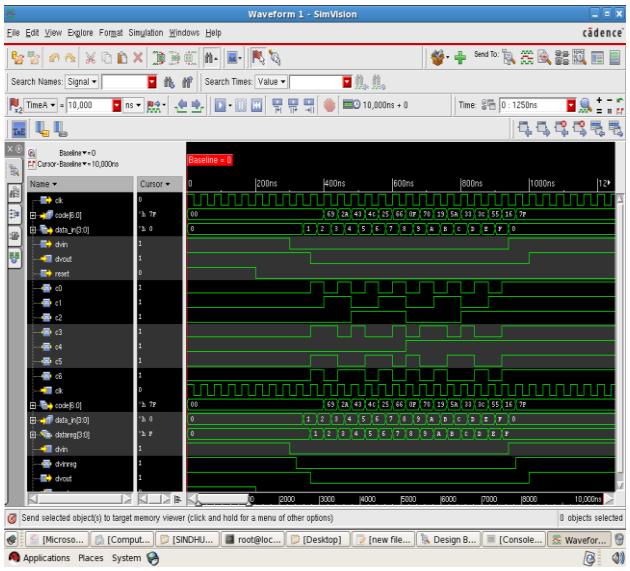


Fig.4. Programming cycles.

	Hamming code		RS code	
	Encoder	**Decoder**	**Encoder**	**Decoder**
Area	4939	4573	16710	15539
Power	225.32	226.32	224.47	225.86

Output Waveform Of Our Work:



5. CONCLUSION:

Thus the method is found to be useful and efficient in error correcting in the product codes, We show that for 8 and 16 kB page sized memories, regular product schemes achieve one decade lower BER when raw BER ranges from to compared to plain RS codes or BCH code with similar code length. A comparison of the area, latency, additional storage also show that product schemes have lower hardware and latency than plain RS codes. The bit error rate is further reduced by introducing I/O pins by which we can reduce the programming cycles.

REFERENCES

[1] R. Bez, E. Camerlenghi, A. Modelli, and A. Visconti, "Introduction to flash memory," Proc. IEEE, vol. 91, no. 4, pp. 489–502, Apr. 2003.

[2] L. Pantisano and K. Cheung, "Stress-induced leakage current (SILC) and oxide breakdown: Are they from the same oxide traps?," IEEE Trans. Device Mater. Reliab., vol. 1, no. 2, pp. 109–112, Jun. 2001.

[3] H. Choi, W. Liu, and W. Sung, "VLSI implementation of BCH error correction for multilevel cell NAND flash memory," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 18, no. 5, pp. 843–847, May 2010. 2011.

[4] S. Li and T. Zhang, "Improving multi-level NAND flash memory storage reliability using concatenated BCH-TCM coding," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 18, no. 10, pp. 1412–1420, Oct. 2010.

[5] C. Yang, Y. Emre, and C. Chaitali, "Flexible product code-based ECC schemes for MLC NAND flash memories," in Proc. IEEE Workshop Signal Process. Syst. (SiPS), 2011.

[6] H. Lee, "High-speed VLSI architecture for parallel Reed-Solomon decoder, IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 11, no. 2, pp. 288–295, Apr. 2003.