Improving Multi-Level NAND Flash Memory Storage Reliability Using Concatenated BCH-TCM Coding

Shu Li and Tong Zhang, *Senior Member, IEEE*

***Abstract*—By storing more than one bit in each memory cell, multi-level per cell (MLC) NAND flash memories are dominating global flash memory market due to their appealing storage density advantage. However, continuous technology scaling makes MLC NAND flash memories increasingly subject to worse raw storage reliability. This paper presents a memory fault tolerance design solution geared to MLC NAND flash memories. The basic idea is to concatenate trellis coded modulation (TCM) with an outer BCH code, which can greatly improve the error correction performance compared with the current design practice that uses BCH codes only. The key is that TCM can well leverage the multi-level storage characteristic to reduce the memory bit error rate and hence relieve the burden of outer BCH code, at no cost of extra redundant memory cells. The superior performance of such concatenated BCH-TCM coding systems for MLC NAND flash memories has been well demonstrated through computer simulations. A modified TCM demodulation approach is further proposed to improve the tolerance to static memory cell defects. We also address the associated practical implementation issues in case of using either single-page or multi-page programming strategy, and demonstrate the silicon implementation efficiency through application-specific integrated circuit design at 65 nm node.**

***Index Terms*—Error rate, fault tolerance, hardware cost, throughput, trellis-coded modulation (TCM).**

I. Introduction

As ONE OF the fastest growing segments in semiconductor industry, NAND flash memory is being used in increasingly diverse applications to realize high-capacity non-volatile data storage. The continuous growth of NAND flash memory storage density has been mainly driven by aggressive technology scaling, e.g., a NAND flash memory with 43-nm CMOS technology has been recently reported in [1]. Besides technology scaling, multi-level per cell (MLC) technique, i.e., to store more than 1 bit in each memory cell (or floating-gate MOS transistor) by programming the cell threshold voltage into one of $l > 2$ voltage windows, has been widely used to further improve the NAND flash memory storage density. Because of its obvious storage density advantage, MLC NAND flash memory has been increasingly dominating global flash memory market. In current design practice, most MLC NAND flash memories store 2 bits per cell, while 3 and even 4 bits per cell NAND flash memories have been recently reported in the open literature [2], [3].

Because of their obviously smaller noise margin, MLC NAND flash memories increasingly rely on system-level error correction to ensure their storage reliability, and almost all the existing MLC NAND flash memories use Bose–Chaudhuri–Hocquenghem (BCH) codes, a family of linear block error correction code (ECC) being widely used in many data storage and communication systems, to realize error correction. For a t-error-correcting BCH code, its coding redundancy and decoding computational complexity are approximately proportional to t and t^2 [4], respectively. Therefore, as the raw storage reliability of MLC NAND flash memories continues to degrade with the technology scaling, the conventional BCH-only approach has to use increasingly stronger BCH codes (i.e., BCH codes with large values of t) at the cost of more redundant memory cells and higher BCH decoder implementation complexity. This trend may quickly make the BCH-only approach prohibitively expensive, e.g., for 4 kB user data per page MLC NAND flash memories with target page error rate of 10^{-16}, suppose the raw bit error rate degrades from 10^{-4} to 10^{-3}, the t of the BCH code must increase from 26 to 91 (i.e., the redundant bits increases from 416 to 1456 and the decoding computational complexity increases by about 12 times).

This paper proposes to concatenate BCH code with trellis-coded modulation (TCM) [5], as an alternative to the conventional BCH-only approach, in order to improve the system-level error correcting efficiency, e.g., given the same amount of redundant memory cells, it can tolerate much worse raw storage reliability, or under the same raw storage reliability, it can largely reduce the redundant memory cells and decoder implementation complexity. Being widely used in modern digital communication systems [6], [7], TCM cohesively considers *multi-dimensional multi-level* signal modulation and convolutional error correction code construction in order to reduce the bit error rate at very low complexity. With the nature of multi-level storage per cell, MLC flash may also benefit from using TCM design principle. A TCM-only approach for MLC NOR flash memories has been recently presented in [8]. Although TCM itself cannot ensure a very low page error rate (e.g., 10^{-14} and below) for NAND flash memories because of the large page size, it may efficiently reduce the bit error rate and hence enable the use of a weaker BCH code to achieve the target page error rate. This naturally suggests the use of a concatenated BCH-TCM coding strategy as a cost-effective alternative to the conventional BCH-only approach for MLC NAND flash memories.

Manuscript received January 30, 2009; revised April 22, 2009. First published September 22, 2009; current version published September 24, 2010.

The authors are with the Department of Electrical, Computer and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180 USA (e-mail: lis4@rpi.edu; tzhang@ecse.rpi.edu).

Digital Object Identifier 10.1109/TVLSI.2009.2024154

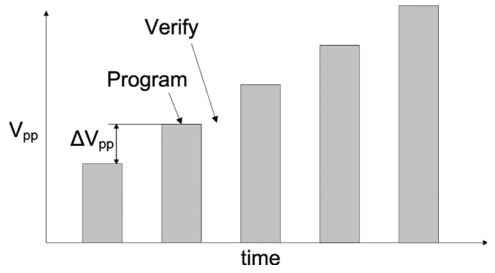

Fig. 1. Control-gate voltage pulses in program-and-verify operation.

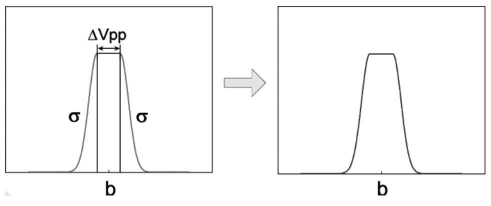

Fig. 2. Threshold voltage distribution model NAND flash memory (except the erase state).

In this paper, we present a concatenated BCH-TCM coding system for 2 bits per cell NAND flash memories to demonstrate the promise of such concatenated coding design solution. A particular 4-D TCM design solution has been developed, which can largely reduce the flash memory bit error rate and hence relieve the burden of the BCH code without incurring any extra redundant memory cells. Simulation results show that, compared with its BCH-only counterpart, a BCH-TCM concatenation can reduce the t of BCH codes by almost one order of magnitude, leading to significant savings of redundant memory cells and BCH decoding complexity. This concatenated coding solution can be applied to single-page programming MLC NAND flash memories in a very straightforward manner, and we present a memory architecture that can further enable the use of this design solution in multi-page programming MLC NAND flash memories. We also investigate how to modify this concatenated coding strategy to handle static memory cell defects. Application-specific integrated circuit (ASIC) design at 65-nm technology node has been carried out to demonstrate the silicon implementation efficiency of this concatenated coding approach. Finally, we should point out that the gain of this BCH-TCM concatenation solution come with two main penalties: 1) it demands the number of storage levels per cell increase from 4 to 5, which will complicate flash memory programming circuits and 2) TCM decoding demands fine-grained sensing quantization, which will increase the memory sensing latency and demand a larger page buffer. Nevertheless, given the significant system performance improvement potential as shown in this paper, we believe that such BCH-TCM concatenation can be a viable alternative to the existing design practice for future MLC NAND flash memories.

II. Basics of MLC NAND Flash Memories

Each NAND flash memory cell is a floating gate transistor whose threshold voltage can be configured (or programmed) by injecting certain amount of charges into the floating gate. Hence, data storage in an l-level per cell NAND flash memory is realized by setting the threshold voltage of each memory cell into one of l non-overlapping voltage windows. Before one memory cell can be programmed, it must be erased (i.e., its threshold voltage is set to the lowest voltage window). A tight threshold voltage control is typically realized by using program-and-verify approach with a stair case program voltage V_{pp} [9] as illustrated in Fig. 1.

Ideally, threshold voltage distributions of adjacent storage states should be sufficiently far away from each other to ensure a high raw storage reliability. In practice, due to various effects such as background pattern dependency, noises, and cell-to-cell interference [10], threshold voltage distributions may be very close to each other or even overlap, leading to non-negligible raw bit error rates. In the following, we present an MLC cell threshold voltage distribution model that will be used to evaluate and compare the error correction performance of different design solutions throughout this paper. The erase state tends to have a wide white Gaussian-like distribution [11], i.e., the probability density function (PDF) of the threshold voltage distribution can be approximated as

$$p_0(x) = \frac{1}{\sigma_0\sqrt{2\pi}} \cdot e^{-(x-\mu)^2/2\sigma_0^2} \tag{1}$$

where σ_0 is the standard deviation and μ is the mean threshold voltage of the erase state. All the other states tend to have the same threshold voltage distribution, as illustrated in Fig. 2. The model consists of two parts, including an uniform distribution in the middle and Gaussian distribution tail on both sides [11]. The width of the uniform distribution equals to the program step voltage ΔV_{pp}, and the standard deviation of the white Gaussian distribution is denoted as σ. The Gaussian distribution on both sides models the overall effect of background pattern dependency, noises, and cell-to-cell interference. Let P_0 and P_1 denote the probabilities of the uniform distribution and Gaussian distribution, respectively. We have the overall PDF $f_{pr}(x)$ as shown in the following:

$$f_{pr}(x) = \begin{cases} \frac{c}{\sigma\sqrt{2\pi}}, \; b-0.5\Delta V_{pp} \le x \le b+0.5\Delta V_{pp} \\ \frac{c}{\sigma\sqrt{2\pi}} e^{-(x-b-0.5\Delta V_{pp})^2/2\sigma^2}, \; x > b+0.5\Delta V_{pp} \\ \frac{c}{\sigma\sqrt{2\pi}} e^{-(x-b+0.5\Delta V_{pp})^2/2\sigma^2}, \; x < b-0.5\Delta V_{pp} \end{cases} \tag{2}$$

where b is the mean of the threshold voltage (i.e., the center of the distribution as shown in Fig. 2), and the constant c can be solved based on $P_0 + P_1 = \int_{-\infty}^{+\infty} f_{pr}(x)dx = 1$.

NAND flash memory cells are organized in an array $\rightarrow$ block $\rightarrow$ page page hierarchy, as illustrated in Fig. 3, where one NAND flash memory array is partitioned into blocks, and each block contains a number of pages. Within one block, each memory cell string typically contains 16 to 64 memory cells, and all the memory cells driven by the same word-line are programmed and sensed at the same time. All the memory cells within the same block must be erased at the same time. Data are programmed and fetched in the unit of page, where the page size ranges from 512 B to 8 kB user data. All the memory cell blocks share the bit-lines and an on-chip page buffer that hold the data being programmed or fetched. For MLC NAND flash memories, there are two options for programming each page, referred to as single-page and multi-page programming. These two different programming options have different operation

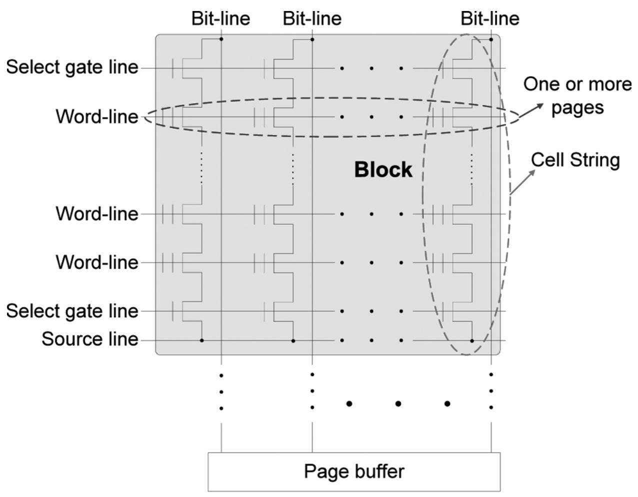

Fig. 3. NAND flash memory structure.

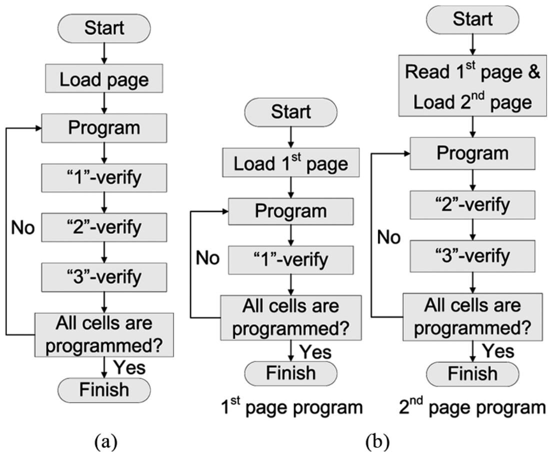

Fig. 4. Operation flow of (a) single-page programming and (b) multi-page programming for 2 bits per cell NAND flash memories.

flows as illustrated in Fig. 4 for 2 bits per cell NAND flash memories.

In the single-page programming, all the bits stored in each MLC memory cell belong to the same page and hence programmed at the same time, e.g., for 2 bits per cell memory, the programming of all the 3 states (i.e., "1", "2", and "3") are accomplished during the same memory program-and-verify process, as shown in Fig. 4(a). On the other hand, in the context of multi-page programming, all the bits stored in each memory cell belong to different pages and are programmed at different time. For example, the two bits in each cell are stored separately: to store the first bit, only state "1" should be programmed; to store the second bit, state "2" and state "3" can be further programmed based upon the combination of the first bit and second bit. Interested readers are referred to [12] and [13] for detailed discussions on single-page programming and multi-page programming. Most existing MLC NAND flash memories use the multi-page programming due to its advantage of higher programming speed. Nevertheless, because single-page programming is able to achieve a tighter threshold voltage window control, it may be preferable as the technology continues to scale down and/or a more aggressive use of MLC technique is being pursued [2], [3].

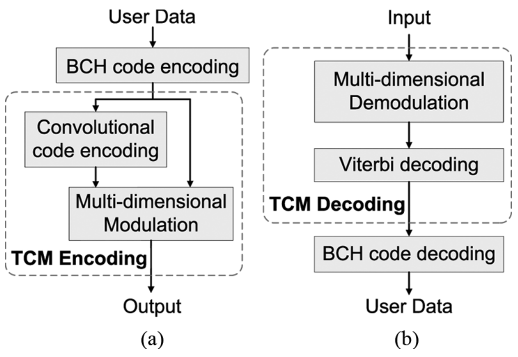

Fig. 5. Operation flow of concatenated TCM-BCH (a) encoding and (b) decoding.

III. CONCATENATED BCH-TCM CODING SYSTEM

As pointed out earlier, this work proposes to use concatenated BCH-TCM coding as an alternative to the conventional BCH-only approach. A concatenated BCH-TCM coding system uses TCM as an inner code and BCH as an outer code. As illustrated in Fig. 5, the operation flow of the concatenated BCH-TCM coding can be briefly described as follows: For encoding, the user data are first encoded by a BCH code encoder, and the BCH codeword is further encoded by a TCM encoder that consists of a convolutional code encoder and a multi-dimensional modulator. For decoding, the data are first processed by a TCM decoder that consists of a multi-dimensional demodulator and a Viterbi decoder, and the output is sent to a BCH decoder to recover the user data.

In concatenated BCH-TCM coding systems, the key issue is the design of TCM sub-system. Assume we want to apply an m-dimensional (m-D) TCM to an n bits per cell NAND flash memory. As a general guideline, the convolutional code encoder only introduces 1 redundant bit for every $n \cdot m$ bits of input data. In order to obviate any redundant memory cells, the m-D modulator in the TCM encoder should map the total $n \cdot m + 1$ bits onto $m \cdot n$ memory cells. As a result, the number of storage levels per cell must increase from 2^n to l, where l is the minimal integer satisfying the requirement $2^{n \cdot m+1} \leq l^m$. Once we determine the value of l, we can apply the basic principle of multi-dimensional space constellation partitioning as presented in [5], [14], [15] to derive the overall modulation strategy. Assuming the modulation strategy partitions the m-D space into 2^s subspaces (or subsets), we further construct a convolutional code with the code rate of $(s-1)/s$ to finish the TCM sub-system design.

In the following, we present a specific TCM design solution for 2 bits per cell NAND flash memory. We choose 4-D modulation (i.e., $m = 4$) and have $l = 5$ (i.e., we need to program each cell with five levels instead of four levels). Since the convolutional code encoder only introduces 1 redundant bit for every 8 bits of input data, the 4-D modulator maps 9 bits onto 4 memory cells. Following the principle presented in [5], [14], and [15], we derive a multi-dimensional space constellation partitioning strategy that partitions the 4-D space formed by 4 memory cells into $2^3 = 8$ subsets and each subset contains

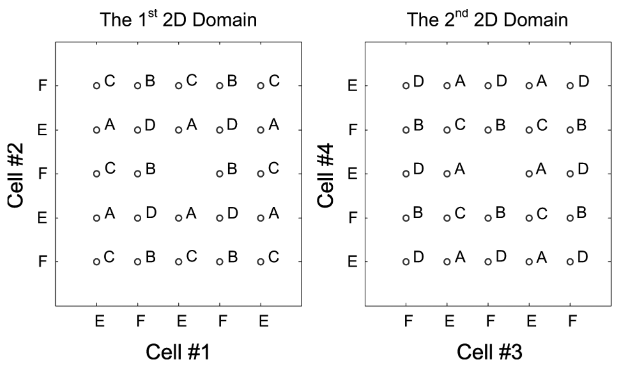

Fig. 6. 1-D and 2-D signal constellation partitioning.

TABLE I
CONSTRUCTION OF THE EIGHT 4-D SUBSETS

4-D subset	Concatenation form	4-D subset	Concatenation form
$\mathcal{P}_1$	$(A,A)\cup(B,B)$	$\mathcal{P}_5$	$(A,C)\cup(B,D)$
$\mathcal{P}_2$	$(C,C)\cup(D,D)$	$\mathcal{P}_6$	$(C,B)\cup(D,A)$
$\mathcal{P}_3$	$(A,B)\cup(B,A)$	$\mathcal{P}_7$	$(A,D)\cup(B,C)$
$\mathcal{P}_4$	$(C,D)\cup(D,C)$	$\mathcal{P}_8$	$(C,A)\cup(D,B)$

64 4-D points. To enable an efficient implementation of the 4-D modulation and demodulation, the 4-D space constellation partitioning is realized in a hierarchical manner as described below in the following.

1) We first partition each 1-D signal constellation (corresponding to each five-level memory cell) into two subsets E and F, as shown in Fig. 6.
2) Next, we partition each 2-D signal constellation into four subsets $A = (E,E)$, $B = (F,F)$, $C = (E,F)$, and $D = (F,E)$, as shown in Fig. 6.
3) Finally, we partition the 4-D signal constellation into eight 4-D subsets shown in Table I.

In summary, the modulator maps 2^9 points onto a 4-D constellation. Since each dimension has 5 levels, this 4-D constellation contains 5^4 points from which we choose 2^9 points. We hierarchically construct the 4-D modulation by combining two 2-D modulations, and each 2-D modulation is formed by two 1-D modulations. Each 4-D subset is the union of two 4-D types, and each 4-D type is constructed by two 2-D subsets, as listed in Table I.

Based on the above 4-D modulation strategy, we further construct a rate-2/3 convolutional code. Hence, the TCM encoder receives 8-bit input at once, out of which 6 bits are directly feed to the 4-D modulator and 2 bits are feed to the convolutional code encoder that generates a 3-bit output. Given the total 9-bit input, the 4-D modulation is realized as follows.

1) The 3-bit input from the convolutional code encoder determines which 4-D subset is selected out of the 8 4-D subsets $\mathcal{P}_1,\ldots,\mathcal{P}_8$.
2) The remaining 6 bits further determine which one out of the 64 4-D points within the selected 4-D sub-set is chosen as the modulation output.

In the context of TCM decoding, the operation of the 4-D demodulation is described as follows. The 4-D demodulator receives $\hat{Z} = \{\hat{z}_1, \hat{z}_2, \hat{z}_3, \hat{z}_4\}$ from four memory cells, where each $\hat{z}_i$ is the quantized data sensed from one memory cell. Given the input $\hat{Z}$, the 4-D demodulator attempts to find the most likely point (i.e., the point that has the minimal Euclidean distance from $\hat{Z}$) within each 4-D subset. Leveraging the hierarchical structure of the 4-D modulation, the 4-D demodulation can be decomposed into 1-D demodulations and 2-D demodulations. First, we use four 1-D demodulators, and each one receives the data read from one cell and searches for

$$\text{Metric\_E} = \min(d_E^2),\ \text{Metric\_F} = \min(d_F^2) \qquad (3)$$

where d_E and d_F denote the 1-D Euclidean distance from the received data to the storage levels in 1-D subset E and F, respectively. Since each 2-D subset is formed by combining two 1-D subsets, each 2-D demodulator can simply combine the 1-D metrics to generate 2-D metrics and locate the most likely 2-D point within each one of the four 2-D subsets. Similarly, based on the output of the two 2-D demodulators, 4-D demodulator locates the most likely point within each one of eight 4-D subsets and the corresponding metrics. Following the TCM design principle [5], we use the output from the 4-D demodulator as branch metrics in the succeeding Viterbi decoder, which carries out a maximum-likelihood search to find the overall most likely 4-D point and generate the 8-bit output for every group of 4 memory cells.

IV. PERFORMANCE EVALUATION

Based upon the above specific 4-D TCM design approach, this section presents simulation results to demonstrate the superior error correction performance of BCH-TCM concatenation over the conventional BCH-only approach. Recall that, in order to enable the use of TCM without incurring extra redundant memory cells, each memory cell needs to store five levels, instead of four levels when the BCH-only approach is being used. Hence, to ensure a fair comparison between these two scenarios, they should have the same memory programming configurations, i.e., the program step voltage ΔV_{pp} should be the same, and the overall cell threshold voltage range should be the same (i.e., the distance between the erase state and the highest threshold voltage state should be the same). As a result, the adjacent threshold voltage windows will be closer and more easily overlap when using the BCH-TCM concatenation. Finally, we note that the demodulation in TCM decoding demands a fine-grained memory sensing quantization, and the more quantization levels are used, the better TCM decoding performance can be achieved. In this work, we assume a 16-level uniform quantization scheme as illustrated in Fig. 7(b), where the quantization thresholds are represented by the dot lines. In case of BCH-only approach, the threshold voltage of each memory cell can be programmed into one of 4 levels, where the distance between adjacent levels is 4/3 times larger and we only need 4-level sensing quantization as shown in Fig. 7(a).

In computer simulations, we normalize the distance between two adjacent levels as 1 when BCH-TCM concatenation is being used, hence the distance between two adjacent levels increases to 4/3 when BCH-only approach is being used. For the modeling of threshold voltage distribution, we fix the width of the uniform part as 1/3 and the deviation of the erase state distribution σ_0 as 0.3, and the value of σ (i.e., the deviation of the Gaussian part of

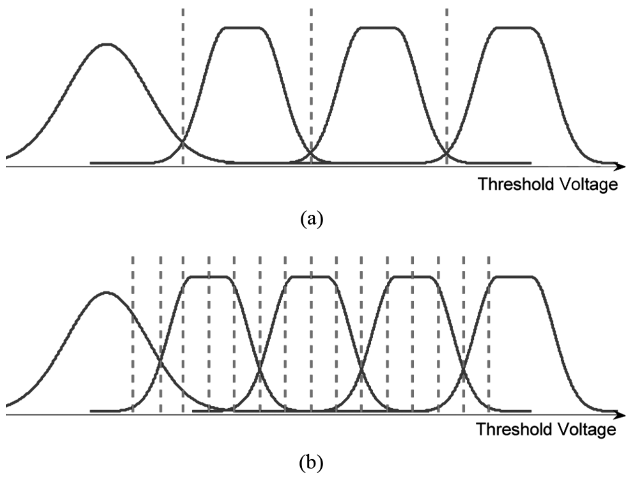

Fig. 7. Sensing quantization schemes for (a) BCH-only approach and (b) BCH-TCM concatenation.

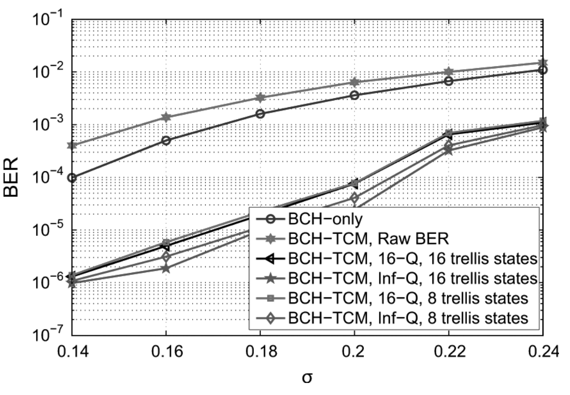

Fig. 8. BER to be handled by BCH code in both BCH-only and BCH-TCM schemes, and the raw BER before TCM-BCH system.

the other threshold voltage states) varies from 0.14 to 0.24 with a step of 0.02. In terms of page size, we consider both 512 B and 4 kB user data per page scenarios. Fig. 8 shows the bit error rate (BER) to be handled by BCH code in both BCH-TCM and BCH-only schemes. Because of the shorter distance between adjacent levels when using BCH-TCM concatenation, its raw BER is worse than that in case of using BCH-only approach, as shown in Fig. 8. In case of BCH-TCM, we consider both 16-level quantization (16-Q) and ideal infinite quantization (Inf-Q). In terms of the trellis size of the convolutional code in TCM, we consider both 8-state and 16-state trellises. The results clearly show that, under the same value of σ, the BCH code in BCH-TCM concatenation only needs to handle a BER that is $1 \sim 2$ orders of magnitude less than that in case of using the BCH-only approach. Therefore, we can use a much weaker BCH code in BCH-TCM concatenation, leading to much less redundant memory cells and BCH decoder implementation complexity. Equivalently, given the same number of redundant memory cells, this BCH-TCM concatenation can tolerate a larger σ and hence can accommodate worse raw memory storage reliability compared with the BCH-only approach.

To achieve a target page error rate of 10^{-16} for NAND flash memories, BCH-only and BCH-TCM approaches will require different BCH codes under the same value of σ. Recall that t

TABLE II
USING BCH-TCM TO REDUCE PROGRAMMING TIME UNDER DIFFERENT NOISE DEVIATIONS

σ		0.14	0.16	0.18	0.20	0.22
ΔV_{pp}	BCH-Only	0.33	0.33	0.33	0.33	0.33
	BCH-TCM	0.59	0.57	0.56	0.54	0.51
t of BCH		13	22	32	60	92

denotes the error correction capability of BCH codes, Fig. 9 shows t versus σ curves of both BCH-TCM concatenation and BCH-only approaches in case of 512 B user data per page and 4 kB user data per page, respectively. The results clearly demonstrate a great potential of using the BCH-TCM concatenation to improve the error tolerance of MLC NAND flash memories. As shown in Fig. 9, TCM systems with 8-state trellis and 16-state trellis achieve almost the same performance, while the implementation complexity of the Viterbi decoder almost doubles when we increase the number of trellis states from 8 to 16. This suggests that, in this case study, using a 8-state trellis could be a good choice considering the tradeoff between error correction performance and implementation cost.

Finally, we note that the superior error correction performance of the BCH-TCM concatenation can be directly traded for improving some other memory system performance metrics. For example, leveraging its stronger error correction performance, we could program MLC NAND flash memories with a larger program voltage step ΔV_{pp}, which directly shortens the memory programming time at the penalty of widened threshold voltage distribution and hence degraded raw storage reliability. The following example could further illustrate this design tradeoff option, where the TCM uses a convolutional code with 8 trellis states and the 16-level sensing quantization as described above.

1) Example 4.1: We fix the normalized program voltage step ΔV_{pp} (i.e., the width of the uniform distribution) as 0.33 when using the BCH-only approach. With a target page error rate of 10^{-16} and 512 B user data per page, we calculate the required t value of BCH codes under different values of noise deviation σ, as listed in Table II. With the same BCH code and σ, we further estimate the maximal value of ΔV_{pp} that can be tolerated when using the above presented BCH-TCM concatenation approach, as listed in Table II. The results quantitatively demonstrate a potential to leverage the BCH-TCM concatenation to largely reduce the memory programming time.

V. REDUCED-DIMENSION DEMODULATION TO TOLERATE DEFECTIVE CELLS

This section discusses the use of TCM in the presence of static memory cell defects. In particular, we consider defective floating gates that cannot be turned on, which will make the sensing results for this cell and all the other cells on the same string always stay as the highest sensing quantization level regardless to the data programmed into these cells. As described above, TCM decoding follows the principle of maximum likelihood decoding based on the Euclidean distance. Occurrence of such defective cells will clearly violate the basic assumption of maximum likelihood decoding based on Euclidean distance, hence tends to incur certain decoding performance degradation.

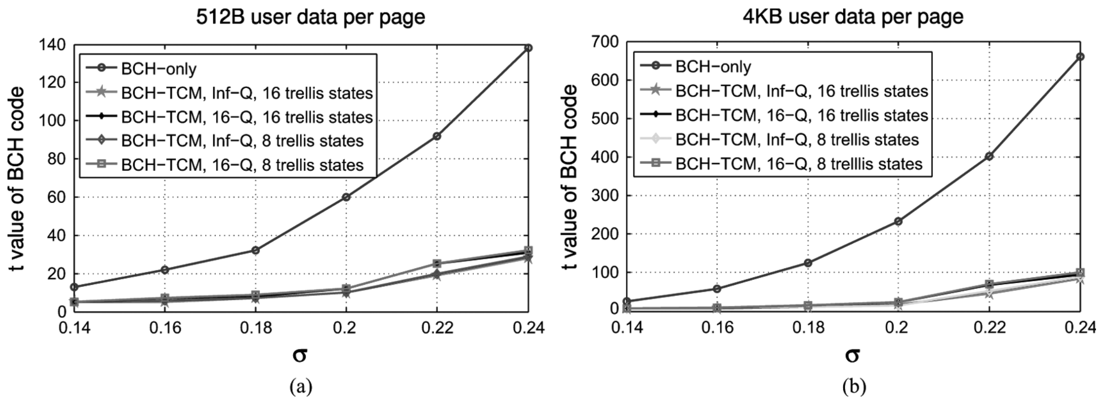

Fig. 9. t of BCH code versus σ for (a) 512 B user data per page and (b) 4 kB user data per page.

In this work, we propose a reduced-dimension TCM demodulation to reduce the performance degradation induced by cell defects that cannot be repaired by redundant word-line or bit-line. The basic idea is simple: suppose we know which cell(s) are defective, we can simply *erase* the defective cell(s) from the constellation space and hence the multi-dimensional demodulation, i.e., we do not count the defective cell(s) when calculating the Euclidean distance during demodulation. Therefore, this method is referred to as reduced-dimension TCM demodulation. In spite of this simple basic idea, it is nontrivial to easily identify those defective cell(s). A straightforward solution is that we may carry out memory testing *a priori* and store the locations of all the unrepairable defective cells in one dedicated on-chip memory block. Such a dedicated memory block should be able to support a fast table lookup operation like content addressable memory (CAM). Nevertheless, since the number of defective cells varies from one page to the next, how to efficiently implement such a dedicated memory block may not be easy. Moreover, embedding such an extra dedicated memory block into flash memory die may greatly complicate the NAND flash memory floorplan.

To address this issue, leveraging the sequential sensing strategy being used in NAND flash memories, we propose a solution to identify static memory cell defects on-the-fly. This is described as follows. During NAND flash memory sensing, the voltage of the word-line of the selected page sweeps from low to high, while the word-lines of the other unselected pages within the same block keep a very high voltage, denoted as V_{unsel}, which is high enough to turn on all the unselected floating gates regardless to their contents. We propose to add voltage V_{unsel} as the last step of the voltage sweep of the selected world-line. Hence, during this last world-line voltage sweep step, all the non-defective cells should be turned on and the bit-line can be successfully discharged. Therefore, the cells associated with the bit-lines that fail to be discharged during this last voltage sweep step are claimed to be defective and hence should be erased from the demodulation in the succeeding TCM decoding.

For the purpose of demonstration, let us consider the 4-D TCM presented in the above. We assume that there is one defective cell among every 1024 cells. Hence, we need to reduce the demodulation from 4-D to 3-D for the group consisting of one defective cell. Let $(\hat{z}_1, \hat{z}_2, \hat{z}_3, \hat{z}_4)$ represent the sensing results of the four cells within each group, and denote the 64 points within each 4-D subset as $(p_1^l, p_2^l, p_3^l, p_4^l),\ l = 1, 2, \ldots, 64$. Suppose the second cell is detected to be defective using the above modified world-line voltage sweep strategy, the metric of each 4-D subset becomes $\min((\hat{z}_1 - p_1^l)^2 + (\hat{z}_3 - p_3^l)^2 + (\hat{z}_4 - p_4^l)^2)$, i.e., the second cell is erased from the Euclidean distance computation. For the erased cell, we assign its state to a fixed one which is randomly determined in the design phase. Using an 8-state trellis convolutional code and 16-level quantization, we carried out simulations to estimate the BER after TCM decoding, as shown in Fig. 10, when randomly injecting one defective cell every 1024 cells. The results clearly show that, in the presence of cell defects, the proposed reduced-dimension TCM decoding can modestly improve the error correction performance compared with using the original TCM decoding.

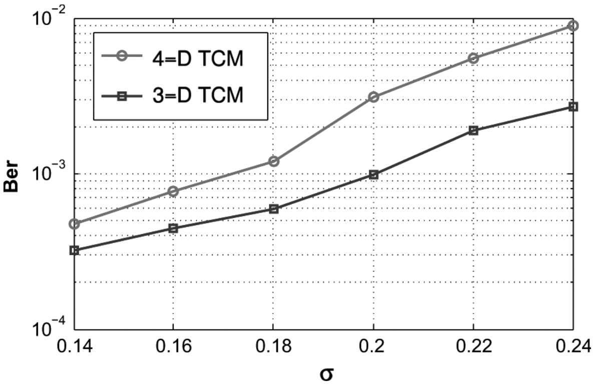

Fig. 10. Performance comparison of 4-D TCM and 3-D TCM with randomly injecting one defective cell every 1024 cells.

VI. Practical Implementation of BCH-TCM Concatenation

For the practical implementation of the proposed BCH-TCM concatenation, we propose to embed the TCM sub-system on the flash memory chip and keep the BCH coding functions off chip, as shown in Fig. 11. This is for the following two main reasons.

1) The demodulator in the TCM decoder demands fine-grained sensing quantization to achieve high error correction performance. Therefore, if the TCM decoder

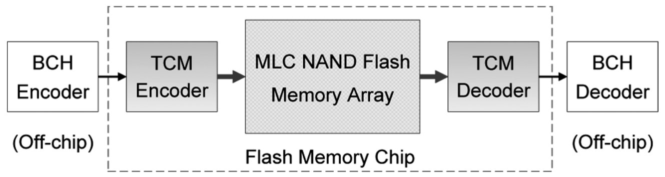

Fig. 11. Overall TCM system framework.

TABLE III
SYNTHESIZED OFF-CHIP BCH DECODER SILICON AREA (mm^2) AT 65-NM NODE

	4KB user data per page		512B user data per page	
σ	BCH-TCM	BCH-only	BCH-TCM	BCH-only
0.20	0.10	10.17	0.02	0.54
0.22	0.94	30.11	0.10	1.31
0.24	1.91	81.63	0.16	3.00

is off chip, a much higher off-chip communication bandwidth is required to send data from the memory chip to the TCM decoder.

2) As demonstrated later, the implementation complexity of TCM coding functions is very small and can well fit into the flash memory die. Moreover, such a function partitioning can ensure a good backward compatibility with the current BCH-only design practice (i.e., the use of on-chip TCM can be transparent to the off-chip BCH).

We first evaluate the off-chip BCH decoder silicon implementation. By fixing the width of uniform distribution as 0.33, we consider three scenarios of σ, i.e., 0.20, 0.22, and 0.24, based on which we obtain the corresponding BCH codes for both BCH-TCM and BCH-only approaches. High-speed BCH decoder silicon implementation has been well studied in the open literature [16]. In this work, with a target decoding throughput of 2 Gb/s, we designed decoders for those BCH codes while assuming the Berlekamp–Massey algorithm is being used. Using Synopsys synthesis tool set with a 65-nm standard cell library, we obtain the synthesized silicon areas that are listed in Table III. The results clearly demonstrate the potential of applying the BCH-TCM concatenation to greatly reduce the implementation complexity of off-chip BCH decoders.

In the remainder of this section, we address the implementation aspects of the on-chip TCM sub-system. As pointed out earlier, MLC NAND flash memories may use single-page or multi-page programming strategy, where either one has its own advantages and disadvantages. Which programming strategy is being used can largely impact the design of on-chip TCM sub-system, hence we discuss the TCM sub-system implementation separately for these two different programming strategies.

A. TCM Sub-System in Case of Single-Page Programming

When MLC NAND flash memories use single-page programming, the data flow of TCM encoding and decoding is essentially the same as conventional TCM, as illustrated in Fig. 5. Hence, we can simply insert the TCM sub-system between the memory I/O and internal data bus, as illustrated in Fig. 12, where we may need to partition the internal bus into several local buses and each local bus associates with one set of TCM encoder/decoder to meet memory programming and read throughput requirements.

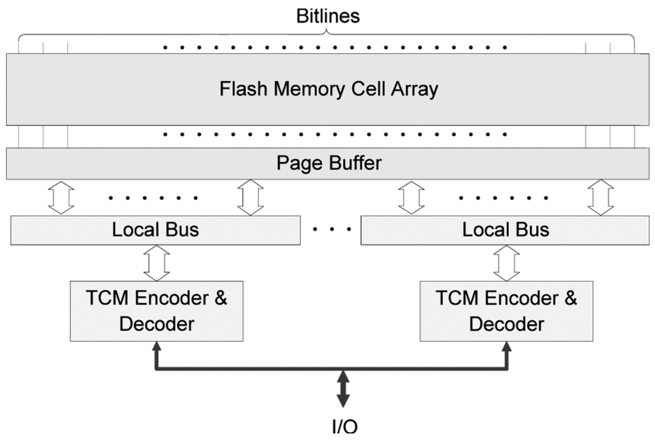

Fig. 12. Implementation of TCM sub-system in case single-page programming is being used.

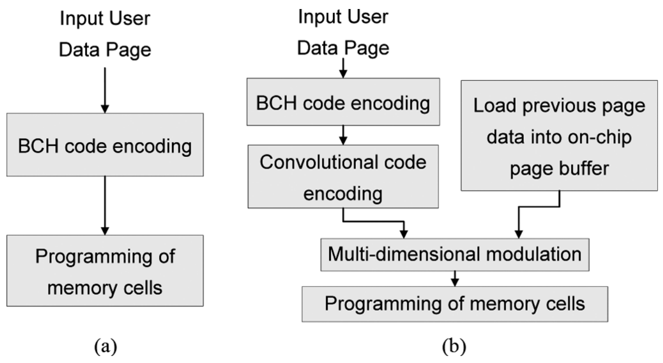
Input User Data Page
BCH code encoding
Programming of memory cells
(a)
Input User Data Page
BCH code encoding
Convolutional code encoding
Load previous page data into on-chip page buffer
Multi-dimensional modulation
Programming of memory cells
(b)

Fig. 13. Programming data flow of (a) the first $N_p - 1$ pages and (b) the last page for an N_p-page programming strategy.

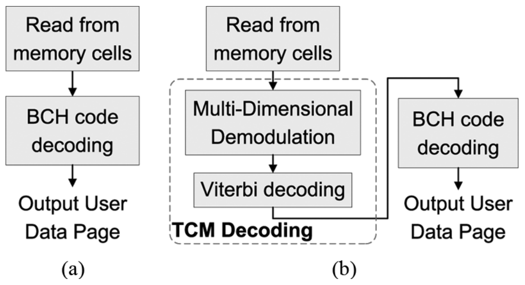

Fig. 14. Read data flow when (a) the last page has not been programmed and (b) the last page has already been programmed.

To evaluate the hardware implementation cost, we use the above 4-D TCM design solution as a test vehicle, which uses a convolutional code with 8 trellis states and uses 16-level quantization sensing scheme. The 4-D modulator and demodulator are designed directly based on the hierarchical modulation and demodulation schemes as described above, and we use the register-exchange state-parallel Viterbi decoder architecture in order to achieve a high decoding throughput. Interested readers are referred to [17]–[19] for detailed discussions on high-speed Viterbi decoder design. Using Synopsys synthesis tool set with a 65-nm standard cell library, we have that one TCM decoder can achieve 1 Gb/s throughput at 0.02 mm^2 silicon

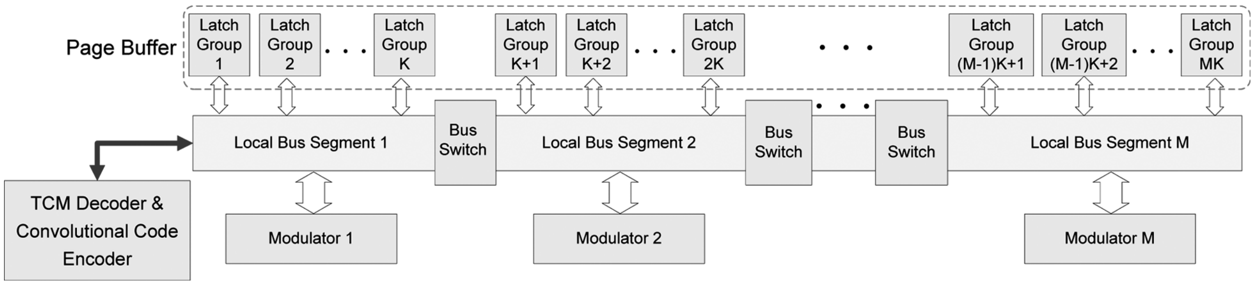

Fig. 15. Structure of one local bus in case of multi-page programming.

area. Hence, to meet the target 2 Gbps, we need two sets of TCM encoder/decoder, which occupy about 0.04 mm^2 in total. The results clearly suggest that, when using the single-page programming strategy, implementation of the on-chip TCM sub-system may only incur very small silicon cost.

B. TCM Sub-System in Case of Multi-Page Programming

When the multi-page programming strategy is being used, each MLC memory cell stores data that belong to different pages and hence are programmed at different time. For an N_p-page programming, since the worse-case threshold voltage window noise margin (hence the worse-case raw storage reliability) occurs only after the last page of data has been programmed, we can use the BCH-only approach to protect the first $N_p - 1$ pages separately, as in current design practice, and use the BCH-TCM concatenation when the last page is being programmed. Because the convolutional code encoding in TCM encoding is done in serial, a straightforward realization of TCM encoding may have to first read all the previously programmed $N_p - 1$ pages and then follow the standard TCM encoding data flow as illustrated in Fig. 5. However, this will incur a significant overhead on the programming latency of the last page and hence may not be feasible in practice.

Since not all the data participate in the convolutional code encoding (e.g., in the 4-D TCM solution presented above, within each group of 8 bits, only 2 bits participate in the convolutional code encoding and the other 6 bits are only involved in the 4-D modulation), the above problem can be solved if we only make the last page of data participate in the convolutional code encoding while use all the previously programmed pages as uncoded data for multi-dimensional modulation. Therefore, in the context of multi-page programming, we must design the TCM sub-system in such a way that the convolutional code encoding only involves the data in the last page. For an N_p-page programming, Figs. 13(a) and (b) illustrate the corresponding data flows when the first $N_p - 1$ pages and last page are being programmed, respectively.

During the read operation, if the last page has not been programmed, each page can be read and decoded as in the current design practice using BCH-only approach; if the last page has already been programmed, we must carry out the on-chip TCM decoding that provides the input to the off-chip BCH decoder to recover the user data. The corresponding read data flows are illustrated in Fig. 14.

When we program the last page, the multi-dimensional modulation cannot start until we read all the previous pages and load them into the page buffer that may take as long as 50 μs [20]. Hence, it is desirable to minimize the latency induced by multi-dimensional modulation, for which we propose to use a two-level hierarchical on-chip bus partitioning approach. On the first level, we partition the on-chip bus into several local buses and each local bus associates with one set of TCM encoder/decoder, similar to the structure shown in Fig. 12 for the single-page programming. On the second level, in order to improve the multi-dimensional modulation parallelism, each local bus is further partitioned into several segments, and each segment associates with one multi-dimensional modulator. This is illustrated in Fig. 15, which shows that one local bus is partitioned into M segments. Within the page buffer, the latches that contains the data of the memory cells participating in one multi-dimensional modulation are grouped, and all the local bus segments associate with the same number of latch groups. Modulations are carried out in serial group-by-group within the same local bus segment and parallel among all the segments. Modulation for one latch group consists of three steps: 1) load the data from the page buffer into the modulator through the local bus segment; 2) modulate the data; and 3) send the modulated data back to the latch group through the local bus segment. The bus switches between adjacent local bus segments are switched off during modulation process to enable the parallel operations of all the modulators, and otherwise are switched on to form a single local bus to support the TCM decoding and convolutional code encoding within each local bus.

We evaluate the overall silicon cost of the above design strategy using the 4-D TCM design solution for 2 bits per cell NAND flash memories as a test vehicle. Because TCM decoding is carried out on the data in two pages and we only need to feed the TCM decoding output associated with the page being read to the off-chip BCH decoder, the on-chip TCM decoder must achieve 4 Gb/s throughput in order to support the target 2 Gb/s read throughput. Hence, based on the design results at 65-nm node presented above for single-page programming, we need to partition the on-chip bus into four local buses, each local bus associates with one TCM decoder and one convolutional code encoder. All the four sets of TCM decoder and convolutional code encoder occupy about 0.08 mm^2. The silicon area of all the modulators depends on the page size and the tolerable latency incurred by modulation. In case of 4 kB user data per page, we assume that 128 modulators are being used in total. Since each latch group contains the data of four memory cells, each modulator carries modulation for 64 latch groups. We set the clock frequency as 125 MHz (or clock period of 8 ns). Since each modulation takes three clock cycles, the latency overhead incurred by modulation is estimated as $3 \times 64 \times 0.008 = 1.536\ \mu$s, which is negligible compared with the normal flash memory read latency. Based on the synthesis results, the 128 modulators occupy about 0.07 mm^2 in total. Therefore, in case of 4 kB user data per page with about 1.5 μs modulation latency, the total silicon area overhead is $0.08 + 0.07 = 0.15$ mm^2. In case of 512 B user data per page, if we keep the same modulation latency overhead, we only need 16 modulators, leading to a total silicon area cost of $0.08 + 0.01 = 0.09$ mm^2. The above results suggest that, even in case of multi-page programming, implementation of on-chip TCM sub-system may not incur significant implementation overhead.

VII. Conclusion

This paper demonstrates that concatenated BCH-TCM coding system holds a great promise to improve storage reliability of MLC NAND flash memories. The key is to leverage TCM to reduce the bit error rate, which can largely relieve the burden of BCH codes, at no cost of extra redundant memory cells. In particular, we designed a BCH-TCM concatenation approach geared to 2 bits per cell NAND flash memories, and its superior error correction performance over current design practice has been demonstrated. We also present a simple method that can make TCM better handle memory cell defects. We further discuss the practical implementation issues of such concatenated BCH-TCM coding schemes, and demonstrate the silicon implementation efficiency through ASIC design at 65-nm technology node.

References

[1] K. Kanda, M. Koyanagi, T. Yamamura, K. Hosono, M. Yoshihara, T. Miwa, Y. Kato, A. Mak, S. L. Chan, F. Tsai, R. Cernea, B. Le, E. Makino, T. Taira, H. Otake, N. Kajimura, S. Fujimura, Y. Takeuchi, M. Itoh, M. Shirakawa, D. Nakamura, Y. Suzuki, Y. Okukawa, M. Kojima, K. Yoneya, T. Arizono, T. Hisada, S. Miyamoto, M. Noguchi, T. Yaegashi, M. Higashitani, F. Ito, T. Kamei, G. Hemink, T. Maruyama, K. Ino, and S. Ohshima, "A 120 mm^2 16 Gb 4-MLC NAND flash memory with 43 nm CMOS technology," in *Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC)*, 2008, vol. 625, pp. 430–431.

[2] Y. Li, S. Lee, Y. Fong, F. Pan, T.-C. Kuo, J. Park, T. Samaddar, H. Nguyen, M. Mui, K. Htoo, T. Kamei, M. Higashitani, E. Yero, G. Kwon, P. Kliza, J. Wan, T. Kaneko, H. Maejima, H. Shiga, M. Hamada, N. Fujita, K. Kanebako, E. Tam, A. Koh, I. Lu, C. Kuo, T. Pham, J. Huynh, Q. Nguyen, H. Chibvongodze, M. Watanabe, K. Oowada, G. Shah, B. Woo, R. Gao, J. Chan, J. Lan, P. Hong, L. Peng, D. Das, D. Ghosh, V. Kalluru, S. Kulkarni, R. Cernea, S. Huynh, D. Pantelakis, C.-M. Wang, and K. Quader, "A 16 Gb 3b/cell NAND flash memory in 56 nm with 8 MB/s write rate," in *Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC)*, Feb. 2008, pp. 506–632.

[3] N. Shibata, H. Maejima, K. Isobe, K. Iwasa, M. Nakagawa, M. Fujiu, T. Shimizu, M. Honma, S. Hoshi, T. Kawaai, K. Kanebako, S. Yoshikawa, H. Tabata, A. Inoue, T. Takahashi, T. Shano, Y. Komatsu, K. Nagaba, M. Kosakai, N. Motohashi, K. Kanazawa, K. Imamiya, H. Nakai, M. Lasser, M. Murin, A. Meir (Poza), A. Eyal, and M. Shlick, "A 70 nm 16 Gb 16-level-cell NAND flash memory," *IEEE J. Solid-State Circuits*, vol. 43, no. 4, pp. 929–937, Apr. 2008.

[4] S. Lin and D. J. Costello, *Error Control Coding: Fundamentals and Applications*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2004.

[5] G. Ungerboeck, "Trellis-coded modulation with redundant signal sets part I, II," *IEEE Commun. Mag.*, vol. 25, no. 2, pp. 5–21, Feb. 1987.

[6] J. G. Proakis, *Digital Communications*. New York: McGraw-Hill, 2001.

[7] J. B. Anderson and A. Svensson, *Coded Modulation Systems*. Norwell, MA: Kluwer, 2003.

[8] F. Sun, S. Devarajan, K. Rose, and T. Zhang, "Multilevel flash memory on-chip error correction based on trellis coded modulation," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2006, pp. 1443–1446.

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

[10] K.-T. Park *et al.*, "A zeroing cell-to-cell interference page architecture with temporary LSB storing and parallel MSB program scheme for MLC NAND flash memories," *IEEE J. Solid-State Circuits*, vol. 43, no. 4, pp. 919–928, Apr. 2008.

[11] K. Takeuchi, T. Tanaka, and H. Nakamura, "A double-level-$\mathrm{V_{th}}$ select gate array architecture for multilevel NAND flash memories," *IEEE J. Solid-State Circuits*, vol. 31, no. 4, pp. 602–609, Apr. 1996.

[12] M. Ohkawa, M. Kang, D. Kim, S.-W. Hwang, B. Y. Choi, Y.-T. Lee, C. Kim, and K. Kim, "A 98 mm^2 die size 3.3 V 64 Mb flash memory with FN-NOR type four-level cell," *IEEE J. Solid-State Circuits*, vol. 31, no. 11, pp. 1584–1589, Nov. 1996.

[13] T. Tanaka, K. Takeuchi, and T. Tanzawa, "A multipage cell architecture for high-speed programming multilevel NAND flash memories," *IEEE J. Solid-State Circuits*, vol. 33, no. 8, pp. 1228–1238, Aug. 1998.

[14] L. F. Wei, "Trellis-coded modulation with multidimensional constellations," *IEEE Trans. Inf. Theory*, vol. 33, no. 4, pp. 483–501, Jul. 1987.

[15] G. D. Forney, Jr, "Coset codes—I. Introduction and geometrical classification," *IEEE Trans. Inf. Theory*, vol. 34, no. 5, pp. 1123–1151, Sep. 1988.

[16] R. E. Blahut, *Algebraic Codes for Data Transmission*. Cambridge, U.K.: Cambridge University Press, 2003.

[17] G. Fettweis and H. Meyr, "High-speed parallel Viterbi decoding: Algorithm and VLSI-architecture," *IEEE Commun. Mag.*, vol. 29, no. 5, pp. 46–55, May 1991.

[18] H.-L. Lou, "Implementing the Viterbi algorithm," *IEEE Signal Process. Mag.*, vol. 12, no. 5, pp. 42–52, Sep. 1995.

[19] H. Dawid, G. Fettweis, and H. Meyr, "A CMOS IC for Gb/s Viterbi decoding: System design and VLSI implementation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 4, no. 3, pp. 17–31, Mar. 1996.

[20] K. Takeuchi, Y. Kameda, S. Fujimura, H. Otake, K. Hosono, H. Shiga, Y. Watanabe, T. Futatsuyama, Y. Shindo, M. Kojima, M. Iwai, M. Shirakawa, M. Ichige, K. Hatakeyama, S. Tanaka, T. Kamei, J.-Y. Fu, A. Cernea, Y. Li, M. Higashitani, G. Hemink, S. Sato, K. Oowada, S.-C. Lee, N. Hayashida, J. Wan, J. Lutze, S. Tsao, M. Mofidi, K. Sakurai, N. Tokiwa, H. Waki, Y. Nozawa, K. Kanazawa, and S. Ohshima, "A 56-nm CMOS 99-mm^2 8-Gb multi-level NAND flash memory with 10-MB/s program throughput," *IEEE J. Solid-State Circuits*, vol. 42, no. 1, pp. 219–232, Jan. 2007.

Shu Li received the B.S. and M.S. degrees in electrical engineering from Tsinghua University, Tsinghua, China, in 2003 and 2005, respectively, and the Ph.D. degree in electrical engineering from Rensselaer Polytechnic Institute, Troy, NY, in 2009.

His research interests include VLSI architecture and circuit design for communication and storage systems.

Tong Zhang (M'02–SM'08) received the B.S. and M.S. degrees in electrical engineering from the Xian Jiaotong University, Xian, China, in 1995 and 1998, respectively, and the Ph.D. degree in electrical engineering from the University of Minnesota, Minneapolis, in 2002.

Currently, he is an Associate Professor with the Electrical, Computer and Systems Engineering Department, Rensselaer Polytechnic Institute, Troy, NY. His current research interests include algorithm and architecture codesign for communication and data storage systems, variation-tolerant signal processing IC design, fault-tolerant system design for digital memory, and interconnect system design for hybrid CMOS/nanodevice electronic systems.

Dr. Zhang currently serves as an Associate Editor for the IEEE Transactions on Circuits and Systems—II: Brief Papers and the IEEE Transactions on Signal Processing.