A Decoder for Short BCH Codes With High Decoding Efficiency and Low Power for Emerging Memories

Sara Choi, Hong Keun Ahn, Byung Kyu Song, Jung Pill Kim, Seung H. Kang, and Seong-Ook Jung, *Senior Member, IEEE*

***Abstract*— In this paper, a double-error-correcting and triple-error-detecting (DEC-TED) Bose–Chaudhuri–Hocquenghem (BCH) code decoder with high decoding efficiency and low power for error correction in emerging memories is presented. To increase the decoding efficiency, we propose an adaptive error correction technique for the DEC-TED BCH code that detects the number of errors in a codeword immediately after syndrome generation and applies a different error correction algorithm depending on the error conditions. With the adaptive error correction technique, the average decoding latency and power consumption are significantly reduced owing to the increased decoding efficiency. To further reduce the power consumption, an invalid-transition-inhibition technique is proposed to remove the invalid transitions caused by glitches of syndrome vectors in the error-finding block. Synthesis results with an industry-compatible 65-nm technology library show that the proposed decoders for the (79, 64, 6) BCH code take only 37%–48% average decoding latency and achieve more than 70% power reduction compared to the conventional fully parallel decoder under the 10^{-4}–10^{-2} raw bit-error rate.**

***Index Terms*— Adaptive error correction, Bose–Chaudhuri–Hocquenghem (BCH) code, double-error-correcting and triple-error-detecting (DEC-TED), emerging memories, error-correcting code (ECC), invalid transition inhibition.**

I. Introduction

EMERGING memories, such as phase change memory, spin-transfer torque magnetoresistive random access memory (STT-MRAM), phase change RAM (PRAM), and resistive random access memory (ReRAM) have been investigated to fill the gaps in terms of performance and density between DRAM and NAND flash memory, referred to as storage class memories (SCMs). They are of interest for their flexible and efficient memory hierarchy, owing to their nonvolatile, high-density, and low-latency characteristics [1]. In addition to SCMs, some emerging memories, such as STT-MRAM, are also considered promising candidate embedded memories due to their fast read and write latencies, low leakage power, and logic-friendly compatibility [2], [3].

As technology scales down, these emerging memories are also struggling with reduced reliability, and as a solution, error-correcting code (ECC) and its encoder/decoder circuits have been applied. While NAND flash requires a powerful ECC capable of correcting up to 100 errors, most of the emerging memories can reach the required chip yield using an ECC capable of correcting two or three errors because of new developments in storage physics [2]–[8]. In addition to simply increasing the memory yield, ECC can be used to optimize memory performance regarding density [9], [10] and energy consumption [11], [12]. In this manner, ECC has become an essential part of emerging memories.

To correct two or three errors, the Bose–Chaudhuri–Hocquenghem (BCH) code is widely adopted for emerging memories [2]–[8]. However, the standard iterative and sequential decoding processes, which require multiple cycles, are not compatible with emerging memories. This is because the latency of the BCH code decoder should be a few nanoseconds, considering the short read or write access time in emerging memories. To achieve a double-error-correcting (DEC) BCH code decoder with latency of a few nanoseconds, a fully parallel decoder structure that uses combinatorial logic gates has been proposed in [13]–[17]. However, it continues to have 50%–80% latency penalty and consumes 6–8 times more power than the single-error-correcting and double-error-detecting (SEC-DED) decoder. As non- or single-bit errors are considerably more likely than multibit (double-bit or triple-bit) errors despite the increased raw bit-error rate (RBER) in nanotechnology, it is inefficient to deal with non- or single-bit errors with a DEC-TED decoder in terms of latency and power, which leads to reduced decoding efficiency. Moreover, the fully parallel decoders consume large dynamic power owing to the invalid transitions in the error-finding block. Since most emerging memories have been widely researched for use in low-power applications, such as wearable devices and IoT devices, the power of fully parallel BCH decoders should also be reduced to maximize the benefits of emerging memories.

Manuscript received May 21, 2018; revised August 27, 2018; accepted October 10, 2018. This work was supported by the Graduate School of Yonsei University Research Scholarship Grants in 2017. *(Corresponding author: Seong-Ook Jung.)*

S. Choi, H. K. Ahn, B. K. Song, and S.-O. Jung are with the School of Electrical and Electronic Engineering, Yonsei University, Seoul 120-749, South Korea (e-mail: sjung@yonsei.ac.kr).

J. P. Kim and S. H. Kang are with Qualcomm Inc., San Diego, CA 92121 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TVLSI.2018.2877147

In this paper, we propose a high-decoding-efficiency and low-power BCH decoder with DEC and triple-error-detecting (DEC-TED) capability for emerging memories. To reduce the average delay and power consumption, an adaptive error correction technique for the DEC-TED BCH code is proposed. In addition, an invalid transition inhibition technique using flip-flops (FFs) and a specific ECC clock is applied to reduce the power consumption further. The synthesis results using 65-nm technology show that the proposed DEC-TED BCH decoder with 64-bit data words achieves more than 50% average latency reduction and 70%–75% average power saving in comparison to the conventional decoder with an insignificant area overhead.

The remainder of this paper is organized as follows. In Section II, an overview of BCH codes and fully parallel structure is given. In Section III, the problems in conventional fully parallel BCH decoders are discussed. The proposed decoder with high decoding efficiency and low power is presented in Section IV. Section V presents the synthesis and comparison results of conventional decoders and the proposed decoder. Finally, Section VI concludes this paper.

II. BCH Codes and Fully Parallel BCH Decoders for Emerging Memories

A. Primitive Binary BCH Code and Decoding Algorithm

In general, a primitive binary BCH code is defined over a binary Galois field with degree m, denoted by GF(2^m).

The (n, k, d) BCH code over GF(2^m) is represented as follows [18]:

$$\begin{aligned}\text{Codeword length} &: n = 2^m - 1\\ \text{Number of information bits} &: k \geq 2^m - mt - 1\\ \text{Minimum distance} &: d \geq 2t + 1.\end{aligned}$$

This code is capable of correcting any combination of t or fewer errors in a block of n digits, called a t-error-correcting BCH code. Since the number of information bits is not represented as the power of two, a shortened binary BCH code is applied in a memory system by eliminating information bits (p), such as ($n - p$, $k - p$, d).

The RBER of the memory cell varies widely depending on design goals such as memory density, read or write latency, and energy consumption. For emerging memories, RBERs of STT-MRAM, ReRAM, and PRAM are distributed with a range of 10^{-10}–10^{-3} [5]–[18], [19]–[21]. These RBER can be reduced by appropriate device, circuit, and architecture design techniques [5], [6], [8], [20]. When it is lower than 10^{-5}, the target block failure rate (BFR) can be achieved with an ECC capable of correcting two errors [2]–[8]. If TED option is added to DEC, the BFR can be improved further. Thus, DEC-TED BCH code is adopted in this paper, and the following decoding processes are described based on the primitive binary DET-TED BCH code [22], [23].

1) Computing Syndrome: For (n, k, 6) DEC-TED BCH code, the parity-check matrix $\mathbf{H}$ is given by

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^3 & \alpha^6 & \cdots & \alpha^{3(n-1)} \end{bmatrix} = \begin{bmatrix} \mathbf{1} \\ \mathbf{H_1} \\ \mathbf{H}_3 \end{bmatrix} \quad (1)$$

where α is the primitive element in GF(2^m).

TABLE I

Relationship Between the Number of Errors and Syndrome Vectors for the BCH DEC-TED Code

Number of Errors	Syndrome Condition	
	S_0	S_1 and S_3
Non	0	$S_1 = S_3 = 0$
Single-bit	1	$S_1^3 = S_3$
Double-bit	0	$S_1^3 \neq S_3$
Triple-bit	1	$S_1^3 \neq S_3$

To determine whether the received codeword, $\mathbf{v}$, has errors, syndrome vector $\mathbf{S}$ is calculated as

$$\mathbf{S} = \mathbf{v} \cdot \mathbf{H}^{\mathrm{T}} = \left[\mathbf{v} \cdot \mathbf{1}^{\mathrm{T}}, \mathbf{v} \cdot \mathbf{H}_{\mathbf{1}}^{\mathrm{T}}, \mathbf{v} \cdot \mathbf{H}_{\mathbf{3}}^{\mathrm{T}}\right] = [S_0, S_1, S_3] \quad (2)$$

where S_0 is a 1-bit vector, and S_1 and S_3 are m-bit vectors for the code generated in GF(2^m). A single-bit error can be corrected using only the S_1 vector because $\mathbf{H_1}$ can be used as the parity-check matrix for the Hamming code. For a double-bit error correction, S_1 and S_3 vectors are utilized together. It is worth noticing that the syndrome vector can be used to detect the number of errors in the received codeword using the specific relationships among S_0, S_1, and S_3 vectors, as shown in Table I. This specific relationship is applied in the proposed decoder, as will be explained in Section IV.

2) Determining the Error Location Polynomial: The next decoding step is to complete the error location polynomial (ELP) based on the calculated syndrome vectors. For a DEC-TED BCH code, the ELP can be represented by

$$\sigma(x) = 1 + \sigma_1 x + \sigma_2 x^2. \quad (3)$$

Notice that each coefficient of the ELP is an m-bit vector if the codeword is constructed on GF(2^m). Conventionally, the Berlekamp–Massey (BM) [24] algorithm is widely applied to compute the coefficients of the ELP.

3) Finding the Error Locations: After computing the coefficients (σ_1 and σ_2), the Chien search is performed to find the roots of the ELP by substituting n elements of GF(2^m), $\{\alpha^0, \alpha^1, \dots, \alpha^{n-1}\}$, into (3).

4) Correcting Errors: Through step 3, an error vector, $\mathbf{e}$, is obtained, and a corrected codeword, $\mathbf{v}^*$, can be represented as $\mathbf{v}^* = \mathbf{v} + \mathbf{e}$. This can be implemented using XOR gates.

B. Fully Parallel BCH Decoders for Emerging Memories

The long BCH code is already adopted in NAND flash memories to correct tens of errors in thousands of data bits [25]–[27]. For long BCH codes, conventional iterative BCH decoding algorithms are applied, and the decoder is usually implemented by the linear feedback shift register, which takes $2n + 2t$ cycles to finish the error correction. However, this decoding algorithm is not compatible with low latency emerging memories, so a fully parallel decoding architecture has been employed to achieve a decoding latency of a few nanoseconds [13]–[17]. The fully parallel decoding architecture is fully parallelized and implemented using a combinatorial logic, which can significantly reduce the decoding latency at the expense of hardware overhead. However, the hardware overhead caused by the fully parallelized

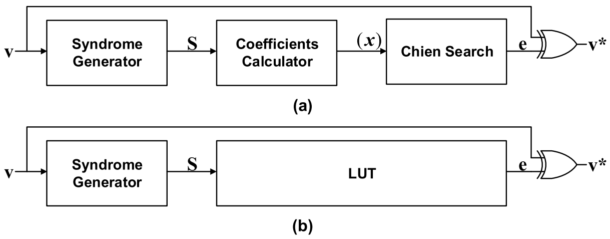


Fig. 1. Block diagrams of (a) PA-based decoder and (b) LUT-based decoder.

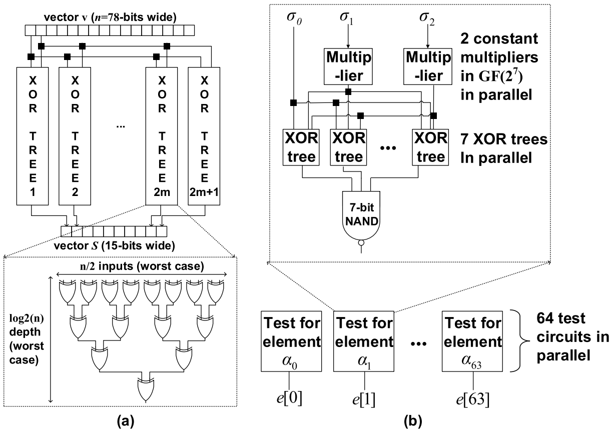


Fig. 2. (a) Syndrome generator and (b) Chien search blocks [31] for 64-bit codeword in the fully parallel DEC-TED BCH decoder.

implementation is not significant in emerging memories. This is because a short BCH code [28]–[30] can be used owing to a low required error-correcting capability and its relatively low size of memory array compared to the NAND flash memories [2]–[8].

In the fully parallel structure, the syndrome vector **S** can be obtained by separate XOR trees with inputs taken from the received code vector, as shown in Fig. 2(a) [31]. According to the decoding algorithm and implementation methods in determining ELP and finding the roots, fully parallel BCH decoders can be divided into two categories, Peterson's algorithm (PA)- and lookup table (LUT)-based decoders.

1) Peterson's Algorithm-Based Decoder: As an alternative to the BM algorithms, PA was proposed in [32] to eliminate the time-consuming iterations. By applying PA, the ELP of DEC-TED BCH code is given by

$$\sigma(x) = 1 + \sigma_1 x + \sigma_2 x^2 = 1 + S_1 x + \left(S_1^2 + \frac{S_3}{S_1}\right) x^2. \quad (4)$$

However, complex finite-field dividers are required to compute the coefficients. Thus, a reverse ELP (RELP) was proposed in [16] to alleviate the complexity of coefficient evaluation and computation during the Chien search, and it is expressed as

$$\tilde{\sigma}(x) = \tilde{\sigma}_0 + \tilde{\sigma}_1 x + \tilde{\sigma}_2 x^2 = \left(S_1^3 + S_3\right) + S_1^2 x + S_1 x^2. \quad (5)$$

The overall structure of a PA-based decoder is shown in Fig. 1(a). A coefficient calculator determines the bit components of the $\tilde{\sigma}_0$, $\tilde{\sigma}_1$, and $\tilde{\sigma}_2$ vectors in (5), and each component of the coefficients is obtained by using the syndrome vector bit components with only modulo-2 addition and multiplication [14]. In the Chien search block, the computations of $\tilde{\sigma}(\alpha^i)$ for $0 \leq i \leq n-1$ are conducted in parallel using simple logic operations [13]. The test circuit for checking whether $\tilde{\sigma}(\alpha^i)$ is 0 requires a multiplication by a constant (α^i), and it can be implemented by XOR-trees, as shown in Fig. 2(b) [31].

2) LUT-Based Decoder: In [17], an LUT-based decoder is proposed by replacing the coefficients calculator and Chien search blocks in the PA-based decoder with an LUT. The LUT contains all the possible pairs of syndromes and their corresponding error patterns. In this decoder, the error positions can be determined directly from the syndromes after a syndrome vector is computed.

3) Comparison of PA-Based and LUT-Based Decoders: In the LUT-based decoder, the error vector can be directly determined immediately after the syndrome vector is computed. Thus, the LUT-based decoder has a shorter decoding latency at the cost of increased area overhead in comparison to the PA-based decoder. However, as the number of correctable errors (t) or the number of information bits (k) increases, the table size exponentially increases, resulting in an inefficient realization in both area and delay. In comparison to the LUT-based decoder, the increased area of the PA-based decoder with increased t or k is much smaller. Therefore, the PA-based decoder is more appropriate for area-constrained applications.

In terms of dynamic power consumption, the PA-based decoder consumes more power than the LUT-based decoder. In the PA-based decoder, the computed syndrome vectors are continuously used in both the coefficient calculator and Chien search blocks until the error vector is determined. Thus, whenever the syndrome vectors are newly computed in response to a newly received codeword, most of the nodes in the blocks following the syndrome generator are toggled, leading to high dynamic power consumption. On the other hand, only one circuit path in the LUT block is activated by the corresponding input syndrome vector due to the inherited LUT characteristic, leading to low dynamic power consumption. Therefore, the LUT-based decoder is favorable in power-constrained applications.

III. Problems in Conventional Fully Parallel BCH Decoders

A. Decoding Efficiency Issue Based on Error Probability

Although RBER for memories increases in nanotechnologies, non- or single-bit errors in a codeword are still much more likely than multibit errors (double- or triple-bit errors), as shown in Fig. 3. In the case of 10-ppm RBER, most codewords (∼99.1%) will have non-errors; thus, error correction is required for only a few codewords (∼0.9%). Moreover, in codewords having errors, 99.6% of them are corrected using single error correction (SEC), and only 0.4% codewords require DEC. Thus, probabilistically, the DEC-TED decoder corrects non- and single-bit errors mostly, and double-bit errors are corrected very infrequently.

In general, the decoder for DEC-TED BCH codes has longer latency, higher area complexity, and much higher power consumption than the decoder for SEC-DED codes. In Table II, the performance of the decoder for SEC-DED Hamming code and fully parallel decoders for DEC-TED BCH code, synthesized with 65-nm technology, are obtained for 64 bits

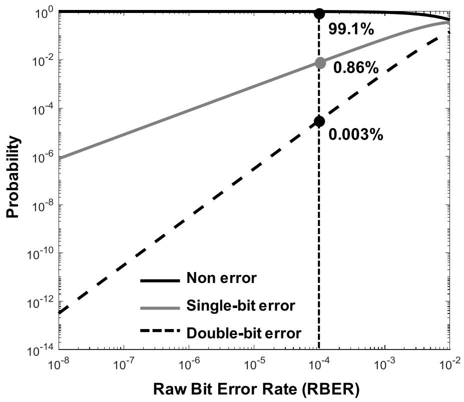


Fig. 3. Probability depending on the RBER with various types of errors (non-, single-bit, and double-bit errors) for 64 bits of codeword.

TABLE II

PERFORMANCE COMPARISON OF SEC-DED AND FULLY PARALLEL DEC-TED DECODERS

Decoder type, ECC configuration	SEC-DED	DEC-TED	
	Hamming (72, 64, 4)	BCH (79, 64, 6)	
		PA-based	LUT-based
Latency (ns)	1.5	3	3
Area (μm^2)	1048.64	2716.8	10347.8
Power (mW)	0.45	3.2	2.2

of codeword when the latency constraints of SEC-DED and DEC-TED decoders are 1.5 and 3 ns, respectively. It should be noted that the PA-based decoder has about four times smaller area than the LUT-based decoder, but it consumes more power under the same latency constraint as discussed in Section II-B. When the latency of the SEC-DED decoder is half that of the DEC-TED decoder, the SEC-DED decoder requires only 40% (10%) area and consumes 14% (20%) power in comparison to the PA-based (LUT-based) DEC-TED decoder, as shown in Table II.

Thus, correction of all non-, single-, and double-bit errors with the DET-TED decoder is inefficient in terms of latency and power consumption, and this reduces the decoding efficiency. If the proper decoder between SEC-DED and DEC-TED decoders can be adaptively selected depending on the error conditions, decoding latency and power consumption can be significantly reduced on average.

B. Dynamic Power Problem in Fully Parallel BCH Decoders

Most of the previous studies on a fully parallel architecture for the BCH decoder have focused on circuit optimization methods to reduce the latency while minimizing the complexity of implementation. However, considering that the read or write power of emerging memories is generally at the microwatt level, much higher power consumption in conventional fully parallel decoders undermines the low-power advantage of emerging memories.

The high dynamic power consumption in the fully parallel BCH decoders can mainly be attributed to the syndrome vector dependence in decoder blocks following the syndrome generator. As discussed in Section II-B, the syndrome vector is a key factor in finding errors in both PA- and LUT-based decoders. Whenever a new input is entered into the decoder, the syndrome generator produces invalid glitches before its outputs settle down. The glitches cause undesired transitions at internal nodes in the blocks that follow the syndrome generator, such as the coefficient calculator, the parallel Chien search (LUT), and the error corrector in the PA-based (LUT-based) decoder, and increase dynamic power consumption.

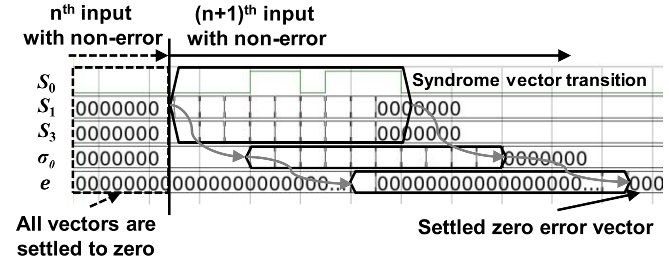

n^th input with non-error
(n+1)^th input with non-error
S_0
S_1
S_3
σ_0
e
0000000
0000000
0000000
0000000000000000000000...
0000000000000000...
000
0000000
0000000
0000000
Syndrome vector transition
All vectors are settled to zero
Settled zero error vector

Fig. 4. Invalid transitions in the coefficient calculator and parallel Chien search blocks due to the transitions on syndrome vectors in the case of consecutive non-error input codewords.

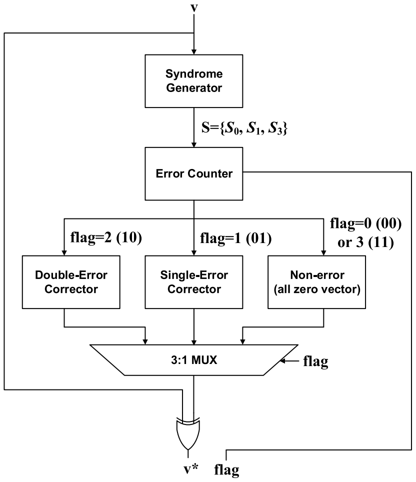


Fig. 5. Conceptual block diagram of DEC-TED BCH decoder with adaptive error correction.

The effect of invalid transition on power is severe, especially when consecutive non-error codewords are received. When a non-error codeword is entered into the decoder, most of the key generated vectors, such as syndromes, coefficients of RELP, and error vectors, are settled to 0. If the next non-error codeword is immediately entered, then the syndrome vector toggles several times until it settles down to 0. This causes invalid transitions in other blocks following the syndrome generator, as shown in Fig. 4. This invalid transition problem can be prevented if stable syndrome vectors are delivered to subsequent blocks. Implementation of stable syndrome vector delivery will be discussed in detail in Section IV.

IV. PROPOSED HIGH-DECODING-EFFICIENCY AND LOW-POWER DEC-TED BCH DECODER

In this section, a DEC-TED BCH decoder using an adaptive error correction and an invalid transition inhibition technique is proposed to achieve the high decoding efficiency and low-power consumption.

TABLE III

FLAG CONDITION FOR THE NUMBER OF ERRORS CLASSIFICATION

Number of Errors	2bit flag condition	
	flag [1] = | (S_1^3+S_3)	**flag [0] = S_0**
Non	0	0
Single-bit	0	1
Double-bit	1	0
Triple-bit	1	1

TABLE IV

ESTIMATED DELAY AND POWER OF THE PA-BASED DECODER WITH THE ADAPTIVE ERROR CORRECTION TECHNIQUE ACCORDING TO THE NUMBER OF ERRORS FOR 64 BITS OF CODEWORD

Number of Errors	Delay	Power
Non/ Triple-bit	T_{synd}+T_{EC}+T_{MUX}+T_{cor} $\approx$ $0.37T_D$	P_{synd}+P_{EC}+P_{MUX}+P_{cor} $\approx$ $0.34P_D$
Single-bit	T_{synd}+T_{EC}+T_{SEC}+T_{MUX}+T_{cor} $\approx$ $0.5T_D$	P_{synd}+P_{EC}+P_{SEC}+P_{MUX}+P_{cor} $\approx$ $0.37P_D$
Double-bit	T_{synd}+T_{EC}+T_{DEC}+T_{MUX}+T_{cor} $\approx$ T_D	P_{synd}+P_{EC}+P_{DEC}+P_{MUX}+P_{cor} $\approx$ P_D

A. DEC-TED BCH Decoder With Adaptive Error Correction

Fig. 5 shows a conceptual block diagram of the proposed adaptive error correction technique. After syndrome vectors are generated, the number of errors caused in the received codeword is classified in an error counter block, and a 2-bit flag signal that represents the number of errors is generated. Then, different error correction algorithms are applied depending on the generated flag signal to improve the decoding efficiency, and a proper error vector is added to the received codeword through the 3:1 MUX.

The 2-bit flag signal can be generated based on the generated syndrome vectors, as shown in Table III. For odd numbers of errors (single- or triple-bit errors), S_0 is "1," whereas for non-error and double-bit errors, S_0 is "0." Multiple-bit error (MBE), a logical OR of all the vector bit components ($\tilde{\sigma}_0[m-1]|\tilde{\sigma}_0[m-2]|\cdots|\tilde{\sigma}_0[0]$) is "0" in the case of non- and single-bit errors because of $S_1^3 + S_3 = 0$. For double- and triple-bit errors, MBE is "1" due to the nonzero vector of $\tilde{\sigma}_0$.

Based on the generated flag signal, we can choose between the single-error (SE) corrector and the double-error (DE) corrector. In the proposed design, the SE corrector uses Hamming SEC code and the DE corrector uses the DEC BCH code. Since error correction algorithms are not required regarding non- or triple-error cases (flag = "00" or "11"), all zero vectors go directly to the MUX without being processed in most delay and power consuming error correction blocks. Thus, the latency and power consumption can be minimized for non- or triple-bit error cases. Since the most common non-error case has minimum latency and power, the average decoding latency and power consumption can be greatly reduced. When a single-bit error occurs (flag = 01), the SE corrector, which compares each column of the $\mathbf{H_1}$ matrix with the S_1 vector, carries out single-bit error correction. Thus, when there is a single-bit error in the received codeword, the proposed decoder has similar latency and slightly larger power consumption in comparison to the conventional SEC-DED code decoder. In the case of double-bit errors (flag = 10), the DE corrector performs error correction, and the latency and power consumption are similar to those of conventional fully parallel DEC-TED BCH decoders.

Thus, the delay and power consumption of the DEC-TED BCH decoder with the adaptive error correction varies according to the types of errors in the codeword. Based on the synthesized results in Table I, Table IV summarizes the estimated delay and power consumption of the PA-based decoder, which employs the proposed adaptive error correction technique for each error cases, where $T_{\text{synd}}(P_{\text{synd}})$, $T_{\text{EC}}(P_{\text{EC}})$, $T_{\text{SEC}}(P_{\text{SEC}})$, $T_{\text{DEC}}(P_{\text{DEC}})$, $T_{\text{MUX}}(P_{\text{MUX}})$, $T_{\text{cor}}(P_{\text{cor}})$, and $T_{\text{D}}(P_{\text{D}})$ represent the delay (power consumption) of the syndrome generator, error counter, SE corrector, DE corrector for PA-based decoder, MUX, error correction block, and total latency (power consumption) of the conventional PA-based DEC-TED decoder, respectively. The total estimated average latency and power for 64-bit data words can be calculated with the probabilities of each error case (shown in Fig. 2). Using a 100-ppm RBER, the average latency (T_{average}) and power (P_{average}) results are calculated as follows:

$$T_{\text{average}} \approx \text{Pr}_0 * T_0 + \text{Pr}_1 * T_1 + \text{Pr}_2 * T_2 = 0.371T_D \quad (6)$$

$$P_{\text{average}} \approx \text{Pr}_0 * P_0 + \text{Pr}_1 * P_1 + \text{Pr}_2 * P_2 = 0.341P_D \quad (7)$$

where $\text{Pr}_0(T_0)$, $\text{Pr}_1(T_1)$, and $\text{Pr}_2(T_2)$ are the probabilities (latencies) of non-, single-, and double-bit error cases, respectively. These results show that the average latency and power consumption are highly determined by those in non-error cases that most frequently occur. The total average latency of the proposed decoder is only 37% that of the conventional PA-based decoder, while it requires only 34% of the total average power consumed by the conventional PA-based decoder. Note that, when the adaptive error correction technique is applied for the LUT-based decoder, the average latency is the same as that of the PA-based decoder, and the average power consumption is reduced to half that of the conventional LUT-based DEC-TED decoder. Thus, we can conclude that the proposed DEC-TED decoder with adaptive error correction improves the decoding efficiency in terms of latency and power consumption. The average power consumption can be reduced more by solving the invalid transition problem. This will be discussed in Section IV-B in detail.

To realize the adaptive error correction technique, additional error counter, SE corrector, and MUX blocks are added to the conventional fully parallel DEC-TED decoder. Regarding the PA-based DEC-TED decoder, the coefficient calculator in the conventional decoder can be a part of the error counter because it originally generates the $\tilde{\sigma}_0$ vector. The error counter additionally requires m-bit OR gate for the BCH code in GF(2^m) to evaluate the flag [1] value. Thus, the costs of area, delay, and power in the error counter for the PA-based DEC-TED decoder with adaptive error correction are minor. In addition, the cost of area in an additional SE corrector is insignificant because the area of the error location block in the SEC decoder is much smaller than that of the coefficient calculator and parallel Chien search blocks in the conventional DEC-TED

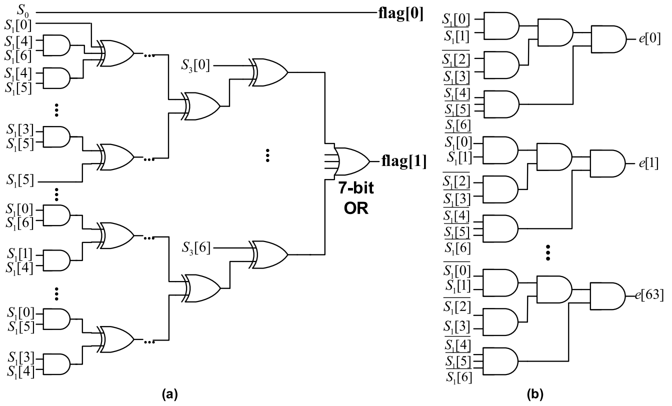


Fig. 6. (a) Error counter and (b) SE corrector blocks in the fully parallel DEC-TED BCH decoder for the 64-bit codeword.

decoder. The hardware structures of the error counter and the SE corrector blocks are shown in Fig. 6.

For the LUT-based DEC-TED decoder, the area and power costs of the SE corrector are negligible. The pair of syndrome vectors and corresponding error patterns in the LUT for the conventional DEC-TED decoder can be divided into two parts. One is for single-bit error cases, and the other is for double-bit error cases. Thus, each part can be replaced by the SE corrector and the DE corrector in the proposed LUT-based decoder, respectively. Since the number of errors is already classified in the error counter block, the S_0 vector is no longer necessary in both SE and DE correctors in our proposed LUT-based decoder. Also, especially for the SE corrector, only the m-bits S_1 vector is required to determine the corresponding error vector. Thus, the sum of the total area of the SE and DE correctors is smaller than that of the LUT block in the conventional LUT-based decoder. Unlike the PA-based decoder, increased area, delay, and power consumption due to the additional error counter are inevitable. However, it would be insignificant or compensated by the reduced area of the LUT block owing to the smaller required size of the syndrome vector. For the hardware implementation of the LUT-based decoder, only the DE corrector block differs from the PA-based decoder. The DE corrector for the LUT-based decoder is implemented with AND gates similar to the SE corrector block.

B. Invalid Transition Inhibition Technique for DEC-TED Decoder

As mentioned in Section III-B, settled syndrome vectors should be transferred to the SE or DE corrector to prevent invalid transitions. Furthermore, the SE and DE correctors should not operate simultaneously in the proposed decoder to ensure lower power consumption. FFs are used between the syndrome generator and the SE and DE correctors to satisfy these two constraints. A block diagram of the proposed DEC-TED decoder with adaptive error correction and invalid transition inhibition techniques is shown in Fig. 7. Note that positive-edge-triggered FFs are used in this design. FFs connected to the SE corrector (DE corrector) are called SEC-FFs (DEC-FFs) for easy representation.

To make sure that both FFs transfer the settled syndrome vectors, the control signals of both FFs should be activated after the syndrome vector and flag bits become stable. To achieve this, a specific clock for the decoder (called the ECC clock) is used to generate the control signal of the FFs, as shown in Fig. 8. For positive-edge-triggered FFs, an inverting ECC clock (FF clock in Fig. 8) is used, and the pulsewidth of the ECC clock should be larger than the summation of the worst delay of syndrome generator (T_{synd}) and that of error counter (T_{EC}). In addition, to prevent the simultaneous operation of SEC and DEC-FFs, a clock-gating technique is applied to the FF clock signal and flag bits using simple INV and AND gates. Note that for non- and triple-error bits cases, both FFs do not transfer the vectors to the following blocks.

The SEC-FFs convey the settled S_1 vector to the SE corrector only when a single-bit error occurs. DEC-FFs do not transfer the syndrome vectors to the DE corrector; thus, the power consumption is significantly reduced in the single-bit error case. Similarly, when a double-bit error occurs, only DEC-FFs transfer the S_1 and $\tilde{\sigma}_0$ vectors (S_1 and S_3 vectors) to the DE corrector in the PA-based (LUT-based) decoder.

C. Comparison With the Previous Works

As a way of improving the decoding efficiency, several ECC structures that utilize more than one error-correcting strength have been well researched. These ECC structures for memories can be categorized into two types based on the ECC selection mechanism. The first type is an “adaptive ECC based on RBER estimation,” and the other type is a “hierarchical ECC.” In this paper, the adaptive ECC based on RBER estimation and hierarchical ECC are called “type 1 ECC” and “type 2 ECC,” respectively.

In the case of the type 1 ECC, the ECC correction ability is determined based on the memory RBER estimation. If the estimated RBER increases, then the stronger error-correcting algorithm is used [12], [33]–[37]. According to the target memory, the parameters for RBER prediction are different. In the case of NAND flash, ECC types are usually determined based on the number of program and erase (P/E) cycles and retention time [33]–[36]. In [37], the reliability of SRAM is predicted by the threshold voltage (V_{TH}) variation. Then, based on the estimated reliability, ECC with appropriate error-correcting ability is performed. For the STT-MRAM, the number of bits flipping from “0” to “1” in a write operation is used to estimate the write failure rate [12]. Then, the code rate of SEC-DED is changed to reduce the write error rate, especially for writing “1” from “0.” In fact, type 1 ECC can be applied to the memory that can predict RBER or some target error rate. That is why NAND flash is a good target memory for applying type 1 ECC because the memory controller counts the P/E cycles and measures the retention time [33]–[35]. On the other hand, for other memories such as SRAM and STT-MRAM, the additional RBER estimation block is required, which causes area and power overhead [12], [37]. In addition, in most papers, encoder and decoder must be implemented separately according to the error-correcting ability, which leads to the significant area overhead [12], [33]–[35], [37]. Moreover, type

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

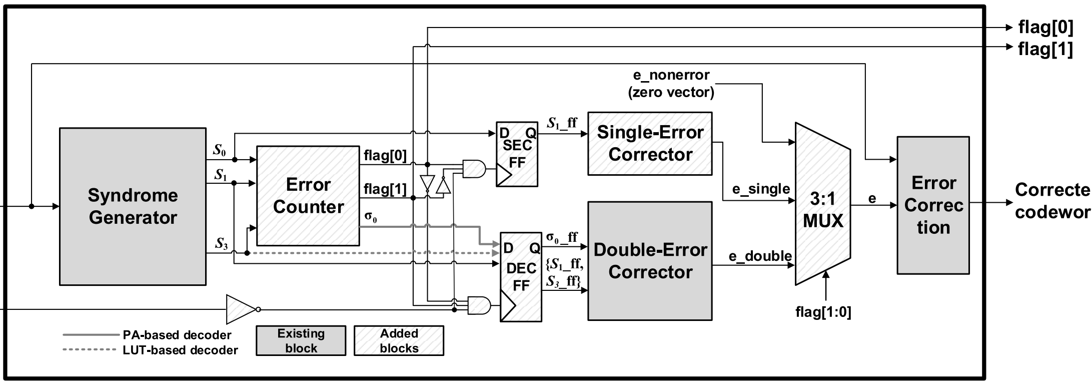


Fig. 7. Block diagram of the proposed high-decoding-efficiency and low-power DEC-TED decoder.

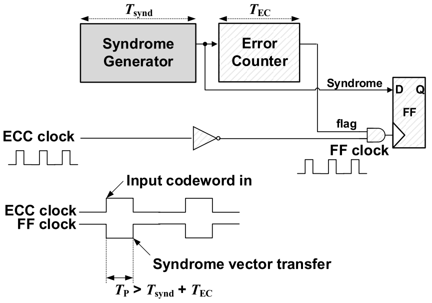


Fig. 8. Timing requirement of ECC clock.

1 ECC cannot fundamentally prevent the situation that single-bit errors are corrected by the multibit error decoder because only one error-correcting algorithm is applied to each decoding process.

Compared to type 1 ECC, the proposed ECC does not require the additional RBER estimation block since the proposed decoder can detect the number of errors in codeword using the error counter. Thus, theoretically, the proposed ECC can be applied to all the memory types. In addition, the same encoder and syndrome generator are used regardless of the error-correcting strength, minimizing area overhead. It can also eliminate all cases of correcting single-bit errors with multibit error decoder, regardless of reliability. Therefore, the proposed ECC maximizes the decoding efficiency than type 1 ECC.

For type 2 ECC, SEC-DED is first performed using Hamming decoder to figure out whether the number of errors is 0, 1, or more than 1. In [39] and [40], they decode the code hierarchically such that SEC-DED is always conducted first, and then, DEC is performed for correcting two errors whenever double-bit errors are detected. Thus, for a non- or single-bit error case, the latency and power can be reduced in comparison to using only the DEC-TED decoder. However, this decoder uses more time and power, especially for the double-bit error cases, because two error correction processes (SEC and DEC) are performed. On the other hand, in [38], to avoid the latency overhead in the double-bit error case, SEC and DEC decoders are concurrently performed, and the output is determined based on the detected number of errors. However, since both SEC and DEC decoders are simultaneously consuming the power regardless of the number of errors, the average power is highly increased compared to the conventional DEC decoders.

TABLE V

COMPARISON OF THE CONVENTIONAL PA-BASED AND LUT-BASED DECODERS AND THE PROPOSED PA-BASED AND LUT-BASED REALIZATIONS

		Conv. PA-based decoder	Proposed PA-based decoder	Conv. LUT-based decoder	Proposed LUT-based decoder
Code		DEC-TED BCH			
Technology		65nm			
Data Width		64			
Algorithm		Peterson		LUT	
Supply Voltage		1.2			
Latency (ns)	Non/ triple	3	1.1	3	1.1
	Single		1.5		1.5
	Double		3		3
Area (μm^2)		2716.8	3264.8	10347.8	10490.1
Power (μW)	Non	3203.9	789.5	2218.8	681.2
	Single		884.7		777.3
	Double		1295.8		1171.4
	Triple		827.7		738.1

In addition, most type 2 ECC requires separate SEC and DEC encoder and decoder circuits.

Contrary to type 2 ECC, the proposed decoder can be implemented with similar latency and power consumption compared to the conventional SEC-DED or DEC-TED decoders in single-bit and double-bit error cases, respectively. This is because the number of errors is detected before the actual error-correcting algorithm is applied, and an appropriate error-correcting algorithm is performed. Also, the decoding latencies for non- and triple-error cases are shorter than that of the SE case because the error correcting algorithm is not performed in the proposed decoder. Furthermore, only one of the error correction algorithms operates depending on the detected number of errors, thus eliminating the power overhead.

V. SYNTHESIS RESULTS AND COMPARISON

This section presents the synthesis results for the proposed DEC-TED BCH decoder. The proposed adaptive error correction and invalid transition inhibition techniques are applied

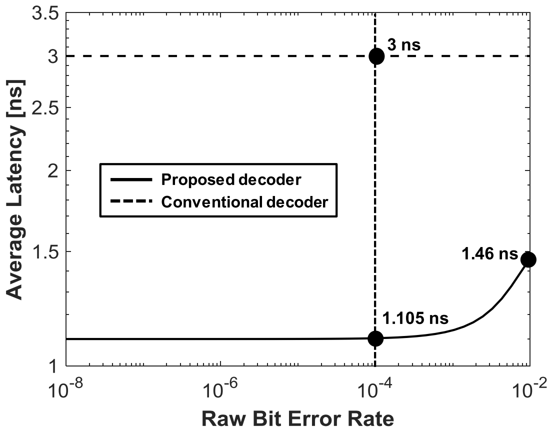


Fig. 9. Average decoding latency for the proposed decoder and conventional decoder according to the RBER.

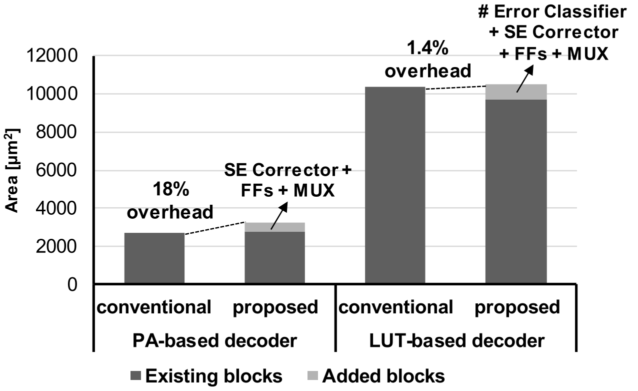


Fig. 10. Area comparison of conventional decoders and the proposed decoders.

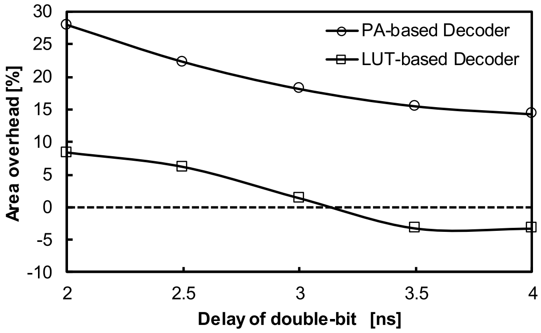


Fig. 11. Area overhead of the proposed decoders according to the delay constraint of double-bit error.

to both conventional PA- and LUT-based (79, 64, 6) DEC-TED BCH decoders synthesized using a 65-nm technology cell library. Both proposed PA- and LUT-based decoders are designed with the same delay constraint and are compared in terms of area and power consumption. The delay for the double-bit error case (maximum) is constrained as the maximum delay for the conventional DEC-TED BCH decoder (3 ns), and for the single-bit error case, the delay constraint is half that for the double-bit error case. The synthesis results for the conventional PA- and LUT-based DEC-TED BCH decoders and the proposed PA- and LUT-based DEC-TED decoders double-bit error case are summarized in Table V.

A. Latency Comparison

Due to the adaptive-error-correction technique, the latency of the proposed decoder varies according to the error cases in the received codeword. In the case of non- and triple- error cases, both SE and DE correctors do not operate because conducting error correction algorithm is not required. Thus, the non- and triple-bit error cases have the shortest delay (1.1 ns), which is almost one-third that of the conventional decoder. For the single-bit error case, the decoding latency is only half that of the conventional decoder.

Fig. 9 shows the average decoding latency under various RBERs. When the RBER is very small, the proposed decoder takes only 1.1 ns to finish decoding on average, which is similar to the estimated latency in (6). As the RBER increases, the latency increases, but it is still less than half the latency of the conventional decoder, even when the RBER increases up to 0.01. In this regard, the proposed decoder significantly improves the decoding efficiency in terms of delay.

B. Area Comparison

Compared to the conventional PA-based decoder, the area of the proposed PA-based decoder increases by 18%, as shown in Fig. 10. As previously mentioned, the area overhead in the error counter for the proposed PA-based decoder is almost negligible. The increased area in the proposed PA-based decoder is mainly due to the added blocks, such as the SE corrector, FFs, and MUX. Although the increased area is the same for both of the proposed decoders, the area overhead of the PA-based decoder is much larger than that of the LUT-based decoder because the PA-based decoder is originally implemented with a much smaller area than the LUT-based decoder. However, the proposed PA-based decoder is still implemented with three times smaller area in comparison to the proposed LUT-based decoder.

For the proposed LUT-based decoder, the area of the DE corrector is reduced because of the reduced contents of the LUT as discussed in Section IV-A. The increased area due to the added blocks can be compensated; thus, the total area of the proposed decoder is very similar to that of the conventional one.

The area overhead of the proposed decoders can be reduced as the latency constraint for the double-bit error case of the decoder increases, as shown in Fig. 11. The area overhead in the proposed PA-based decoder is reduced to 14% when the latency constraint increases to 4 ns. Especially for the proposed LUT-based decoder, the area is smaller than that of the conventional one as the latency increases by more than 3 ns. This is because the area reduction of the LUT in the proposed decoder is greater than the area increase due to the added blocks.

C. Power Comparison

1) Power Measurement Method: In our proposed decoder design, the signal path differs according to the number of errors in the received codeword, leading to power consumption variation. Thus, we evaluated the average power consumption

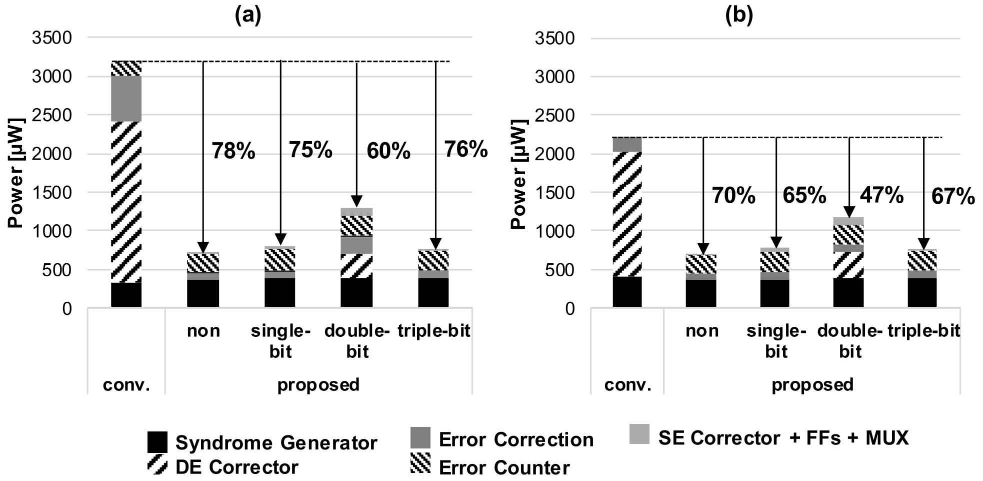


Fig. 12. Power consumption comparison of the conventional decoders and the proposed decoders. (a) PA-based decoder. (b) LUT-based decoder.

for the specific input vectors in relation to the error cases. Input codewords were entered into the decoder for every 10 ns because we assumed that the decoder inputs are changed with 10-ns cycles based on the required read or write operation time and ECC decoder operating time in most emerging memories for real applications. Then, the average power was measured during several thousands of cycles (20k cycles). Note that the input codeword vectors were generated based on the generator matrix of the BCH code.

To measure the power consumption of non-error cases, successive input vectors that had no errors were used. Unlike non-error cases, the probability of continuous error occurrence is quite low. Thus, to measure single-, double-, and triple-bit error cases, erroneous and non-error input vectors were applied alternately to the decoder. This sequence is practical because it can maximize the transitions in operating blocks.

2) Power Comparison: Fig. 12 shows the power reduction of the proposed PA-based and LUT-based decoders in relation to the number of errors. Similar to latency, the power consumption for non-error cases was the lowest because the most power consuming DE corrector and error correction blocks did not operate. Moreover, the power consumption for the error correction block was greatly reduced (almost three times lower) because invalid transitions are eliminated. Thus, the power of the proposed PA- and LUT-based decoders was reduced by 75% and 70% compared to the conventional decoders, respectively. Note that the power reduction rate in the proposed LUT-based decoder was lower than that of the PA-based decoder in all error cases, due to the added power consumption in the error counter. For the single-bit error case, the power consumption was very similar as that of the non-error case in both proposed decoders because the power consumption of additional blocks (SE-FFs and SE corrector) was relatively small. Even with the double-bit error case, the total power consumption was greatly reduced. The significant reduction in power consumption occurred in the DE corrector because settled syndrome vectors were transferred, preventing invalid transitions. In addition, the power of the error correction block was also reduced due to the elimination of invalid transitions in the DE corrector. Finally, the power consumption in the added blocks, such as the error counter in the LUT-based decoders, DE-FFs, and MUX, was not significant. In the case of triple-bit error, even though the operating blocks were the same as in the non-error case, the power consumption was slightly higher than that of the non-error cases because of nonzero syndrome vectors in the error counter.

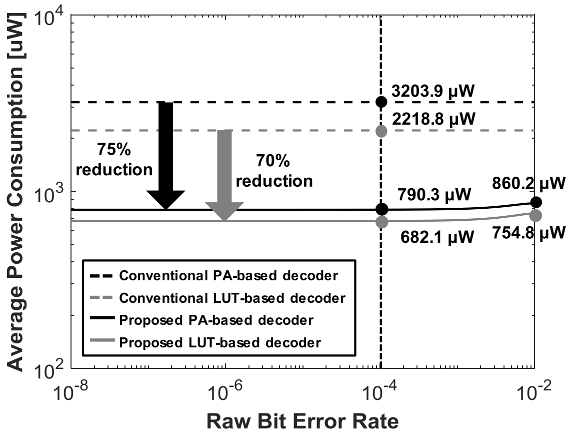


Fig. 13. Average power consumption for the proposed PA- and LUT-based decoders and conventional decoders according to the RBER.

Fig. 13 shows the average power consumption of the proposed and conventional PA- and LUT-based decoders according to RBER. When the RBER was very small, the power consumption was reduced by 75% and 70% in the PA- and LUT-based decoders, respectively. Since the invalid transition issue is not considered in (7), the synthesis result of the average power consumption was much lower than the value estimated by (7). Even though the RBER increased to 10 000 ppm, the power consumption of both proposed decoders was three times lower than that of the conventional decoders. In this regard, it is concluded that the proposed decoders operate with much lower power even though they are implemented with fully parallel structures.

The power consumption in the two proposed decoders is similar because the power of the DE corrector and error correction blocks in the PA-based decoder is significantly reduced by the elimination of invalid transitions. Note that the proposed LUT-based decoder consumes slightly less power than the proposed PA-based decoder because fewer changes occur in the internal nodes by only one activated path corresponding to the input syndrome vector in the LUT. Based on the area and power results of the proposed decoders, the proposed PA-based decoder can be implemented with three times less area and similar power consumption in comparison to the proposed LUT-based decoder. Therefore, the proposed PA-based decoder is a more favorable option in any applications than the LUT-based decoder, unlike the conventional decoders.

VI. Conclusion

This paper presented a (79, 64, 6) BCH DEC-TED decoder with high decoding efficiency and low power for emerging memories. We proposed an adaptive error correction technique that chooses a different decoding algorithm depending on the error conditions in a codeword, to improve the decoding efficiency regarding delay and power consumption. Also, to avoid the high dynamic power consumption problem in conventional fully parallel BCH decoders, the invalid transition inhibition technique was adopted by using FFs and a specific ECC clock. In comparison to the conventional PA- and LUT-based DEC-TED decoders, the average decoding latency of the proposed decoders is less than half that of the conventional decoders with an RBER of 100 ppm. Due to the added blocks, the area increases by 18% and 1.4% in the PA- and LUT-based decoders, respectively. However, this overhead can be decreased with an increased latency constraint of the decoder. The proposed PA- and LUT-based decoders achieve 75% and 70% power reduction on average in comparison to the conventional decoders, respectively. This paper provides a promising ECC decoder solution to achieve the target yield even with a high RBER, especially for high-performance and low-power applications using emerging memories.

References

[1] P. Amato, S. Bellini, M. Ferrari, C. Laurent, M. Sforzin, and A. Tomasoni, "Fast decoding ECC for future memories," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 9, pp. 2486–2497, Sep. 2016.

[2] S. H. Kang, "Embedded STT-MRAM for energy-efficient and cost-effective mobile systems," in *IEEE Symp. VLSI Circuits Dig. Tech. Papers*, Jun. 2014, pp. 36–37.

[3] H. Noguchi, K. Ikegami, N. Shimomura, T. Tetsufumi, J. Ito, and S. Fujita, "Highly reliable and low-power nonvolatile cache memory with advanced perpendicular STT-MRAM for high-performance CPU," in *IEEE Symp. VLSI Circuits Dig. Tech. Papers*, Jun. 2014, pp. 1–2.

[4] P. Amato, C. Laurent, M. Sforzin, S. Bellini, M. Ferrari, and A. Tomasoni, "Ultra fast, two-bit ECC for emerging memories," in *Proc. 6th IEEE Int. Memory Workshop (IMW)*, May 2014, pp. 79–82.

[5] Y. Emre, C. Yang, K. Sutaria, Y. Cao, and C. Chakrabarti, "Enhancing the reliability of STT-RAM through circuit and system level techniques," in *Proc. IEEE Workshop Signal Process. Syst.*, Oct. 2012, pp. 125–130.

[6] D. Niu, Y. Xiao, and Y. Xie, "Low power memristor-based ReRAM design with error correcting code," in *Proc. 17th Asia South Pacific Design Autom. Conf.*, Jan./Feb. 2012, pp. 79–84.

[7] M. Mao, Y. Cao, S. Yu, and C. Chakrabarti, "Optimizing latency, energy, and reliability of 1T1R ReRAM through cross-layer techniques," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 6, no. 3, pp. 352–363, Sep. 2016.

[8] C. Yang, M. Mao, Y. Cao, and C. Chakrabarti, "Cost-effective design solutions for enhancing pram reliability and performance," *IEEE Trans. Multi-Scale Comput. Syst.*, vol. 3, no. 1, pp. 1–11, Jan./Mar. 2017.

[9] B. Del Bel, J. Kim, C. H. Kim, and S. S. Sapatnekar, "Improving STT-MRAM density through multibit error correction," in *Proc. IEEE/ACM Conf. Design, Autom. Test Eur. (DATE)*, Mar. 2014, pp. 1–6.

[10] Z. Pajouhi, X. Fong, and K. Roy, "Device/circuit/architecture co-design of reliable STT-MRAM," in *Proc. IEEE/ACM Conf. Design, Autom. Test Eur. (DATE)*, Mar. 2015, pp. 1437–1442.

[11] Y. Alkabani, Z. Koopmans, H. Xu, A. K. Jones, and R. Melhem, "Write pulse scaling for energy efficient STT-MRAM," in *Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI)*, Jul. 2016, pp. 248–253.

[12] X. Wang, M. Mao, E. Eken, W. Wen, H. Li, and Y. Chen, "Sliding basket: An adaptive ECC scheme for runtime write failure suppression of STT-RAM cache," in *Proc. IEEE/ACM Conf. Design, Autom. Test Eur. (DATE)*, Mar. 2016, pp. 762–767.

[13] X. Wang, D. Wu, C. Hu, L. Pan, and R. Zhou, "Embedded high-speed BCH decoder for new-generation NOR flash memories," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, Sep. 2009, pp. 195–198.

[14] W. Xueqiang, P. Liyang, W. Dong, H. Chaohong, and Z. Runde, "A high-speed two-cell BCH decoder for error correcting in MLC nor flash memories," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 56, no. 11, pp. 865–869, Nov. 2009.

[15] Y. Yoo and I.-C. Park, "A search-less DEC BCH decoder for low-complexity fault-tolerant systems," in *Proc. IEEE Workshop Signal Process. Syst. (SiPS)*, Oct. 2014, pp. 1–6.

[16] C.-C. Chu, Y.-M. Lin, C.-H. Yang, and H.-C. Chang, "A fully parallel BCH codec with double error correcting capability for NOR flash applications," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Mar. 2012, pp. 1605–1608.

[17] R. Naseer and J. Draper, "Parallel double error correcting code design to mitigate multi-bit upsets in SRAMs," in *Proc. Eur. Solid-State Circuits Conf. (ESSCIRC)*, Sep. 2008, pp. 222–225.

[18] S. Lin and D. J. Costello, "BCH codes," in *Error Control Coding: Fundamentals and Applications*, 2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 2004, pp. 141–177.

[19] D. H. Yoon, J. Chang, R. S. Schreiber, and N. P. Jouppi, "Practical nonvolatile multilevel-cell phase change memory," in *Proc. Int. Conf. High Perform. Comput., Netw., Storage Anal.*, Nov. 2013, pp. 1–12.

[20] B. L. Ji *et al.*, "In-line-test of variability and bit-error-rate of HfO$_x$-based resistive memory," in *Proc. IEEE Int. Memory Workshop (IMW)*, May 2015, pp. 1–4.

[21] S. Sills *et al.*, "Challenges for high-density 16 Gb ReRAM with 27 nm technology," in *IEEE Symp. VLSI Circuits Dig. Tech. Papers*, Jun. 2015, pp. 106–107.

[22] J. V. D. Horst and T. Berger, "Complete decoding of triple-error-correcting binary BCH codes," *IEEE Trans. Inf. Theory*, vol. 22, no. 2, pp. 138–147, Mar. 1976.

[23] F. Gulliang, "An algebraic complete decoding for double-error-correcting binary BCH codes," *J. Electron., China*, vol. 1, no. 1, pp. 12–17, 1984.

[24] S. B. Wicker, *Error Control Systems for Digital Communication and Storage*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1995.

[25] X. Zhang and K. K. Parhi, "High-speed architectures for parallel long BCH encoders," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 13, no. 7, pp. 872–877, Jul. 2005.

[26] W. Liu, J. Rho, and W. Sung, "Low-power high-throughput BCH error correction VLSI design for multi-level cell NAND flash memories," in *Proc. IEEE Workshop Signal Process. Syst. Design Implement.*, Oct. 2006, pp. 303–308.

[27] J. Freudenberger and J. Spinner, "A configurable Bose–Chaudhuri–Hocquenghem codec architecture for flash controller applications," *J. Circuits, Syst. Comput.*, vol. 23, no. 2, p. 1450019, 2014.

[28] A. Fahrner, H. Griesser, R. Klarer, and V. V. Zyablov, "Low-complexity GEL codes for digital magnetic storage systems," *IEEE Trans. Magn.*, vol. 40, no. 4, pp. 3093–3095, Jul. 2004.

[29] J. Spinner, M. Rajab, and J. Freudenberger, "Construction of high-rate generalized concatenated codes for applications in non-volatile flash memories," in *Proc. IEEE 8th Int. Memory Workshop (IMW)*, May 2016, pp. 1–4.

[30] I. Zhilin and A. Kreschuk, "Generalized concatenated code constructions with low overhead for optical channels and NAND-flash memory," in *Proc. 15th Int. Symp. Problems Redundancy Inf. Control Syst. (REDUNDANCY)*, Sep. 2016, pp. 177–180.

[31] D. Strukov, "The area and latency tradeoffs of binary bit-parallel BCH decoders for prospective nanoelectronic memories," in *Proc. Asilomar Conf. Signals, Syst. Comput.*, Oct./Nov. 2006, pp. 1183–1187.

[32] W. W. Peterson, *Error-Correcting Codes*. Cambridge, MA, USA: MIT Press, 1972.

[33] S. Tanakamaru, Y. Yanagihara, and K. Takeuchi, "Error-prediction LDPC and error-recovery schemes for highly reliable solid-state drives (SSDs)," *IEEE J. Solid-State Circuits*, vol. 48, no. 11, pp. 2920–2933, Nov. 2013.

[34] J. Xiao-Bo, T. Xue-Qing, and H. Wei-Pei, "Novel ECC structure and evaluation method for NAND flash memory," in *Proc. IEEE Int. Syst.-Chip Conf. (SOCC)*, Sep. 2015, pp. 100–104.

[35] L. Yuan, H. Liu, P. Jia, and Y. Yang, "Reliability-based ECC system for adaptive protection of NAND flash memories," in *Proc. Int. Conf. Commun. Syst. Netw. Technol.*, Apr. 2015, pp. 897–902.

[36] L. Yuan, H. Liu, P. Jia, and Y. Yang, "An adaptive ECC scheme for dynamic protection of NAND Flash memories," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, Apr. 2015, pp. 1052–1055.

[37] A. Basak, S. Paul, J. Park, J. Park, and S. Bhunia, "Reconfigurable ECC for adaptive protection of memory," in *Proc. IEEE Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Aug. 2013, pp. 1085–1088.

[38] P. P. Ankolekar, R. Isaac, and J. W. Bredow, "Multibit error-correction methods for latency-constrained flash memory systems," *IEEE Trans. Device Mater. Rel.*, vol. 10, no. 1, pp. 33–39, Mar. 2010.

[39] R. Ramaraju, R. Rock, E. J. Gieske, C. Park, and D. F. Greenberg, "Hierarchical error correction for large memories," U.S. Patent 8 677 205 B2, Mar. 18, 2014.

[40] Z. Wang, "Hierarchical decoding of double error correcting codes for high speed reliable memories," in *Proc. 50th ACM/EDAV/IEEE Design Autom. Conf. (DAC)*, May/Jun. 2013, pp. 1–7.

Sara Choi was born in Gwangju, South Korea, in 1990. She received the B.S. degree in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 2013, where she is currently working toward the combined Ph.D. degree.

Her current research interests include the peripheral circuit and memory architecture design for STT-RAM.

Hong Keun Ahn was born in Seoul, South Korea, in 1993. He received the B.S. degree in electrical and electronic engineering from Yonsei University, Seoul, in 2017, where he is currently working toward the combined Ph.D. degree.

His current research interests include the memory architecture design for STT-RAM.

Byung Kyu Song was born in Seoul, South Korea, in 1988. He received the B.S. degree in electrical and electronic engineering from Dankook University, Yongin, South Korea, in 2013. He is currently working toward the combined Ph.D. degree at Yonsei University, Seoul.

His current research interests include the bit-cell structure and peripheral circuit design for STT-RAM.

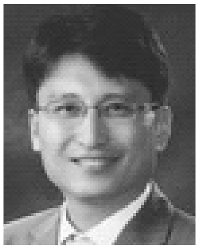
Jung Pill Kim received the B.S. degree in electronic engineering from Hanyang University, Seoul, South Korea, in 1988, and the M.S. and Ph.D. degrees in computer science and electrical engineering from Harvard University, Cambridge, MA, USA, in 2000 and 2003, respectively.

From 1988 to 1998, he was with Hynix Semiconductor, Icheon, South Korea, where he was involved in the development and research of many DRAM products from 1-MB DRAM to 64-MB EDO and SDR DRAM. From 2001 to 2008, he was a Design Team Leader and a Principle Design Engineer with Qimonda, Research Triangle Park, NC, USA, focusing on high-density 1-GB SDR DRAM products, low-power mobile DRAM and LPDDR1 products, and high-performance 1-GB GDDR3 products. Since 2008, he has been with Qualcomm Incorporated, San Diego, CA, USA, where he is involved in the development of STT-MRAM-related IPs and macros. He has authored or coauthored several technical papers and holds over 20 U.S. patents. His current research interests include low-power circuit design, high-speed memory design, and advanced future memory design and technologies.

Seung H. Kang received the B.S. and M.S. degrees from Seoul National University, Seoul, South Korea, in 1989 and 1991, respectively, and the Ph.D. degree in materials science and engineering from the University of California at Berkeley, Berkeley, CA, USA, in 1996.

He was with the Lawrence Berkeley National Laboratory, Berkeley, CA, USA, where he was involved in the fields of SQUID sensors and VLSI interconnects. From 1998 to 2005, he was a Distinguished Member of Technical Staff with the Lucent Technologies Bell Laboratories, Murray Hill, NJ, USA, and led advanced device reliability projects. In 2006, he joined Qualcomm Inc., San Diego, CA, USA, and has pioneered embedded STT-MRAM and spintronic devices for mobile systems. He is currently the Director of Engineering, Corporate Research and Development, Qualcomm Inc., and leads an emerging memory technology group for mobile systems, including Internet of Things and wearables. He has authored or coauthored over 70 papers and delivered over 35 keynotes and invited speeches at international conferences. He holds over 350 patents granted globally.

Dr. Kang currently serves as an IEEE Electron Device Society Distinguished Lecturer. He has served at numerous technical committees.



Seong-Ook Jung (M'00–SM'03) received the B.S. and M.S. degrees in electrical and electronic engineering from Yonsei University, Seoul, South Korea, in 1987 and 1989, respectively, and the Ph.D. degree in the electrical engineering from the University of Illinois at Urbana–Champaign, Urbana, IL, USA, in 2002.

From 1989 to 1998, he was with Samsung Electronics Co., Ltd., Hwaseong, South Korea, where he was involved in the specialty memories, such as video RAM, graphic RAM, window RAM, and merged memory logic. From 2001 to 2003, he was with T-RAM Inc., Mountain View, CA, USA, where he was the Leader of the Thyristor-Based Memory Circuit Design Team. From 2003 to 2006, he was with Qualcomm Inc., San Diego, CA, USA, where he focused on high-performance low-power embedded memories, process variation tolerant circuit design, and low-power circuit techniques. Since 2006, he has been a Professor with Yonsei University. His current research interests include process variation tolerant circuit design, low-power circuit design, mixed-mode circuit design, and future generation memory and technology.

Dr. Jung is currently a Board Member of the IEEE SSCS Seoul Chapter.