International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 07 | July 2020 www.irjet.net p-ISSN: 2395-0072

IMPLEMENTATION OF TRIPLE BIT ERROR CORRECTING BCH DECODER WITH HIGH DECODING EFFICIENCY FOR EMERGING MEMORIES

K. Sneha[1], P. Krishna Reddy[2]

[1]P.G Scholar. ECE., Dhanekula Institute of Engineering and Technology.
[2]Asst. Professor. ECE., Dhanekula Institute of Engineering and Technology.

Abstract: As the technology scales down emerging memories struggling with reduced reliability. As a solution error-correcting code ECC and its decoder circuits have been applied. To correct two (or) three errors BCH code is widely adopted. We have 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. Here we are implementing triple bit error correcting (TEC) decoder to increase the decoding efficiency, we propose an adaptive error correction technique for the 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.

Keywords: adaptive error correction, bose-Chaudhuri–Hocquenghem (BCH) code, double error correcting (DEC)and triple error detecting & correcting (TED -TEC), emerging memories, error correcting code(ECC),LUT- based decoders, invalid-transition-inhibition technique

1. INTRODUCTION

Emerging memories, such as phase change memory, spin-transfer torque magneto resistive 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 multi bit (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.

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.

2. CODENING THEORY

Background coding theory more detailed accounts of error-correcting codes can be found in: Hill, Pless MacWilliams and Sloane, van Lint, and Assumes and Key. See also Peterson for an early article written from the engineers' point of view. Proofs of all the results quoted here can be found in any of these texts; our summary here follows.

Here a message is first given by the source to the encoder that turns the message into a codeword, i.e. a string of letters from some alphabet, chosen according to the code used. The encoded message is then sent through the channel, where it may be subjected to noise and hence altered. When this message arrives at the decoder belonging to the receiver, it is equated with the most likely codeword, i.e. the one (should that exist) that, in a probabilistic sense depending on the channel, was probably sent, and finally this "most likely" codeword is decoded and the message is passed on to the receiver.

3. BCH CODES AND FULLY PARALLEL BCH DECODERS FOR EMERGING MEMORIES

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]:

Codeword length: $n = 2^m - 1$

Number of information bits: $k \geq 2^m - mt - 1$

Minimum distance: $d \geq 2t + 1$.

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, DECTED 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 H is given by

$$\mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^3 & \alpha^6 & \cdots & \alpha^{3(n-1)} \end{bmatrix} = \begin{bmatrix} \mathbf{1} \\ \mathbf{H_1} \\ \mathbf{H_3} \end{bmatrix}$$

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

TABLE I

RELATION SHIP BETWEEN THE NUMBER OF ERRORS AND SYNDROME VECTORS FOR THE BCH DEC-TED CODE

	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, v, has errors, syndrome vector S is calculated as

$$\mathbf{S = v \cdot H^T = v \cdot 1^T, v \cdot H^T 1, v \cdot H^T 3 = [S_0, S_1, S_3]}$$

where S_0 is a 1-bit vector, and S1 and S3 are m-bit vectors for the code generated in GF(2m). A single-bit error can be corrected using only the S_1 vector because H1 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

$$\sigma(x) = 1 + \sigma 1x + \sigma 2x\ 2$$

Notice that each coefficient of the ELP is an m-bit vector if the codeword is constructed on GF(2m). 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 (σ1 and σ2), the Chien search is performed to find the roots of the ELP by substituting n elements of GF(2m), $\{\alpha^0, \alpha^1, ..., \alpha^{n-1}\}$, into (3).

4) Correcting Errors: Through step 3, an error vector, e, is obtained, and a corrected codeword, v^*, can be represented as $v^* = v + e$. 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

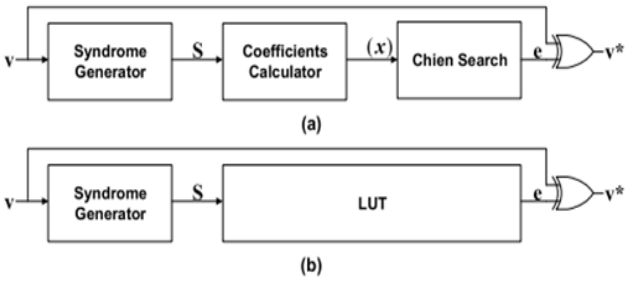


Fig.1.Block diagram a) PA-based decoder b) LUT-based decoder

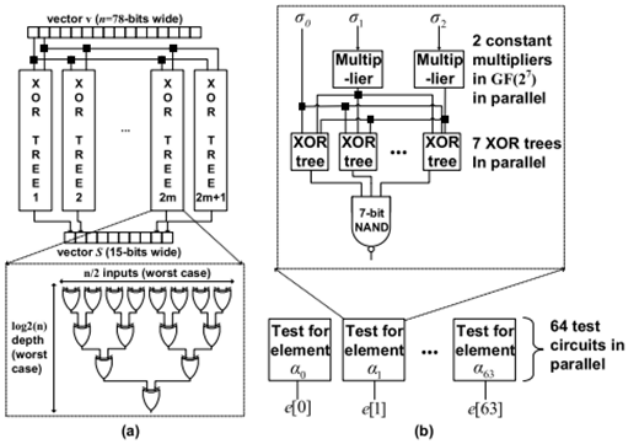


Fig. 2.(a) Syndrome generator and (b) Chien search blocks [31] for 64-bit codeword in the fully parallel DEC-TED BCH decoder.

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

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

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

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

The overall structure of a PA-based decoder is shown in Fig. 1(a). A coefficient calculator determines the bit components of the σ˜0, σ˜1, and σ˜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 σ (α ˜ i) for 0 ≤ i ≤ n −1 are conducted in parallel using simple logic operations [13].

The test circuit for checking whether σ (α ˜ i) is 0 requires a multiplication by a constant (αi), 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.

4. PROBLEMS IN CONVENTIONAL FULLY PARALLEL BCH DECODERS

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. 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 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 Dynamic Power Problem in Fully Parallel BCH Decoder:

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.

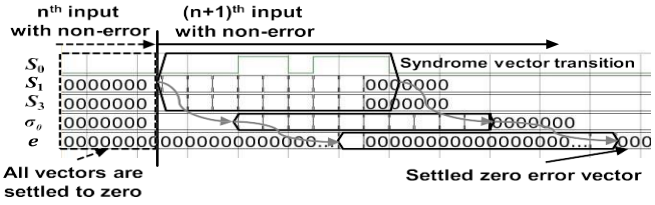


Fig. 4. Invalid transitions in the coefficient calculator and parallel Chien search blocks due to the transitions on syndrome vectors in the case of consecutive non-error input code words

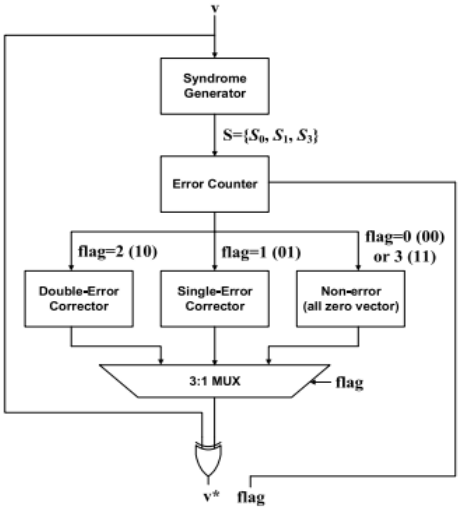


Fig.5. Conceptual block diagram of DEC-TED BCH decoder with adaptive error correction.

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.

The effect of invalid transition on power is severe, especially when consecutive non-error code words 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. This invalid transition problem can be prevented if stable syndrome vectors are delivered to subsequent blocks

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

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

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

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.

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 H1 matrix with the S1 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. 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.

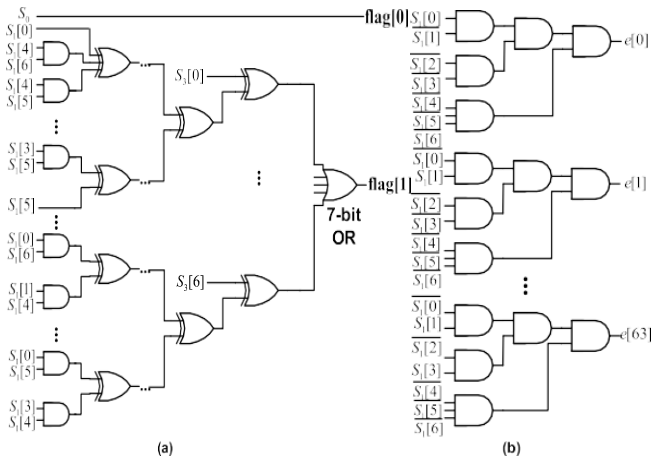


Fig. 6. (a) Error counter and (b) SE corrector blocks in the fully parallel DEC-TED BCH decoder for the 64-bit codeword.

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 σ_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 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 S0 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 S1 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.

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.

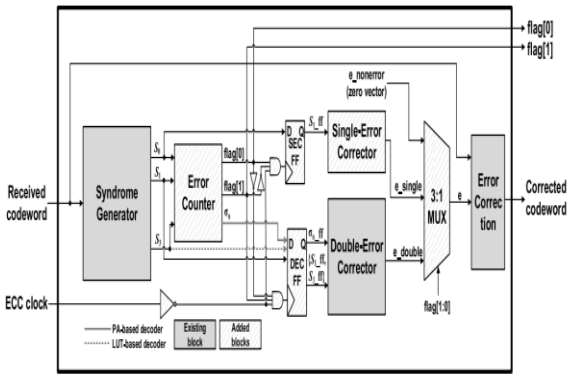


Fig: block diagram of adaptive bch decoder

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 pulse width 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 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 σ_0 vectors (S_1 and S_3 vectors) to the DE corrector in the PA-based (LUT-based) decoder.

6. TRIPLE BIT ERROR CORRECTING BCH DECODER

Here we are implementing the triple bit error correcting 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. The 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 4:1 MUX.

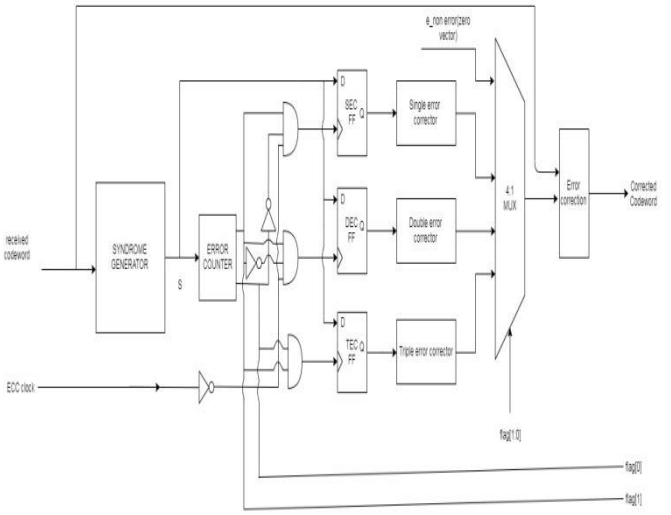
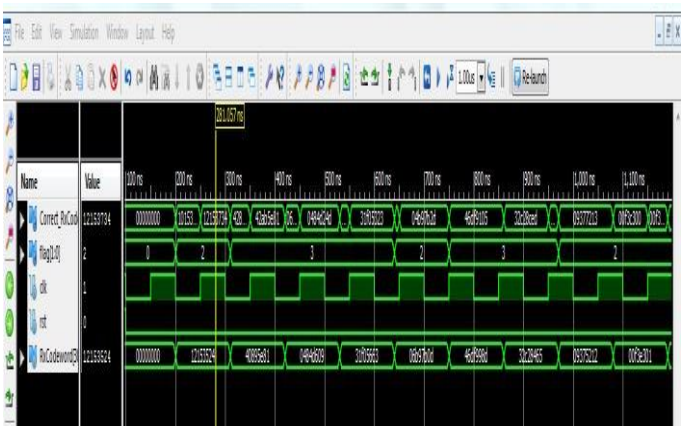


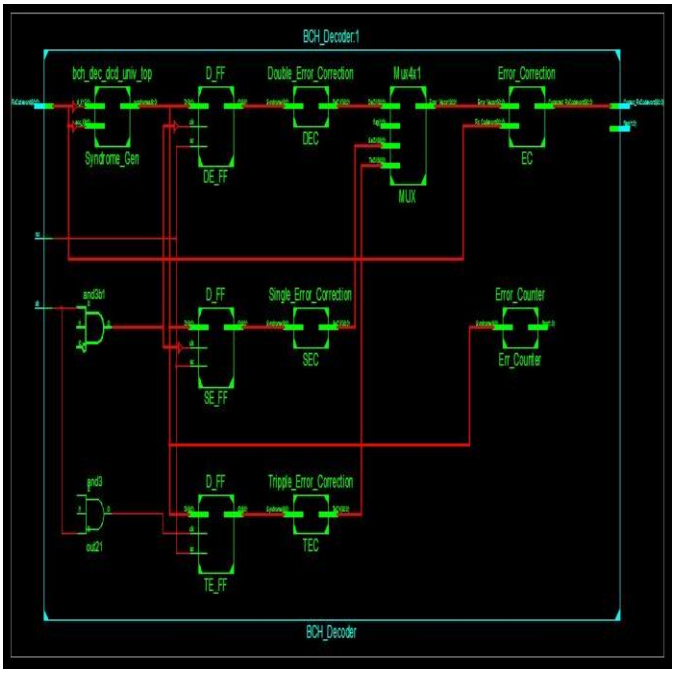
Fig: block diagram of bch decoder with triple error correction capability

7. RESULTS

Simulation results:



RTL Design:



Device Summary:

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice LUTs	136	2400	5%
Number of fully used LUT-FF pairs	0	136	0%
Number of bonded IOBs	66	102	64%
Number of Block RAM/FIFO	6	12	50%

8. CONCLUSION:

In this brief, we proposed an adaptive triple bit error correcting (TEC) BCH decoder with high decoding efficiency for emerging memories. In this an adaptive error correction technique that chooses a different decoding algorithm depending on the error conditions in a code word, to improve the decoding efficiency regarding delay and power consumption. The area overhead due to the added blocks compensated by increased latency constraint of a decoder. Totally this is highly useful for high performance emerging memories.

REFERENCES:

1.P. Amato, S. Bellini, M. Ferrari, C. Laurent, M. Sfozin, 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. King, "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.Fujitha,"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 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.17 th Asia South Pacific Design Autom. Conf., jan./Feb. 2012,pp. 79-84.

7. M.Mao, Y.Cao,S. Yu, and C. Chakrabarthi," Optimizing latency, energy, and reliable 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. 1-11, jan./Mar. 2017.

9. B. Del Bel, J. Kim, C. H. Kim, and S. S Sapatnekar," Improving STT-MRAM desity through multibit error correction", in Proc. IEEE/AC 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-RAM,"in Proc,IEEE/ACM

12. X.Wang,M.Mao,E.Eken, W.wen, H. Li, and Y.Chen,"Sliding basket:An adaptive ECC Schemefor 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 generation NOR flash memories",in Proc. IEEE custom Integer. 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. 15,no. 11, pp, 865-869,Nov.2009.
15. Y. Yoo and L-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 codes 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. Constello, "BCH codes," in Error Control Coding:Fundametals and Applications,2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall,2004,pp. 222-225.
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.