Turkish Online Journal of Qualitative Inquiry (TOJQI)
Volume 12, Issue 10, October 2021: 1017-1028

Modified DEC for Short BCH Codes for Parallel Correction of 3-Bit Error with High Decoding Efficiency

[1] Hiba Tabassum, [2] B. Kiran Kumar, [3] Dr. Mohammad Jabirullah
[1]PG Scholar, Department of ECE, Lords Institute of Engineering & Technology, Hyderabad, India
[2]Associate Professor, Department of ECE, Lords Institute of Engineering & Technology, Hyderabad, India
[3]Associate Professor & HOD, Department of ECE, Lords Institute of Engineering & Technology, Hyderabad, India
[1]hibatabassum6@gmail.com, [2]b.kirankumar@lords.ac.in, [3]drjabirullah@lords.ac.in

Abstract*:*

This paper describes a new Bose – Chaudhuri – Hocquenghem (BCH) code decoder for error correction in expanding memory that has a high decoding capacity while using less power. For the DEC-TED BCH code, we propose an adaptive error correction method that identifies the amount of mistakes in a codeword immediately after syndrome generation and uses a different error correction algorithm based on the error situation. This technique improves decoding performance. Adaptive error correction lowers average decoding time and energy consumption substantially by increasing decoding performance. Erroneous transitions in error-finding blocks are caused by syndrome vector glitches, which may be removed to save power. The proposed decoders for the (79, 64, 6) BCH code have an average decoding latency of only 37 percent -48 percent and achieve a power reduction of over 70 percent compared to the conventional fully parallel decoder for the 104–102 raw bit error rate, according to synthesis results using an industry-compatible 65-nm technology library.

Index Terms—Adaptive error correction, Double- error-correcting and triple-error-detecting (DEC- TED), Invalid transition inhibition, Bose– Chaudhuri– Hocquenghem (BCH) code, errorcorrecting code (ECC), Emerging memories

I. INTRODUCTION

Researchers looked into new types of storage class memories to bridge the output and density gaps between DRAM and NAND flash memory, including phase change memory, spin-transfer torque magneto resistive random access memory (STT-MRAM), phase shift random access memory (PRAM), and resistive random access memory (ReRAM) (SCMs). DRAM chips are an excellent complement to a memory system because of their high density, low latency, and non-volatility. STT-MRAM has a number of benefits over other developing memories like SCMs, such as rapid read and write latencies, low power leakage, and compatibility with standard digital logic. Because of the scalability of the technology, ECCs, which utilise encoder/decoder circuits, have been used to solve the reliability issues that modern memory face. A 100-error ECC is required for NAND memory because of recent advancements in storage physics[2]–[8], but most future memories will be able to reach the required chip yield with an ECC that can only repair two or three mistakes at a time. Memory output density and energy consumption may be optimised by using ECC in addition to improving memory yield. With a millisecond delay, combinational logic gates for correcting (DEC) BCH code decoders have been suggested in [13]–[17]. In spite of this, it has a penalty of 50–80 percent delay and uses 6–8 times more power than the combined single and double-error decoder (SEC- DED). Using a decoder like the DEC-TED decoder is problematic because of the increased RBER in nanotechnology. This lowers the efficiency of the DEC-TED decoder. Decoders must use numerous dynamic resources to correct a mistake that was introduced in the error-finding chain. It's necessary to decrease the capacity of fully parallel BCH decoders in low-power applications like smart watches and IoT devices if you want to take full use of memory development advantages. This article's low-power BCH decoder's decoding efficiency and power consumption have been optimised for usage in memory creation that incorporates DEC and triple-error detection (DEC-TED). It is possible that an adaptive error-correction technique will reduce the mean latency and power consumption of the DEC-TED BCH code. consumption. Using flip-flops (FFs) and a particular ECC clock, an incorrect transition inhibition method lowers power usage even more. Dec-TED BCH decoder, when synthesised using 65-nm technology, reduce latency by more than 50% on average, saving 70–75% in power over conventional decoders with small overhead areas, says the proposed DEC-TED BCH decoder. The following parts of this text are structured as follows: Section II introduces BCH codes and their completely parallel construction. Section III examines a more conventional serial-to-parallel BCH decoder. See Section IV for further information on the decoder's low-power design and excellent decoding efficiency.

II. LITERATURE SURVEY:

In order to increase decoding efficiency, many ECC structures that make use of multiple error-correction strengths have been extensively investigated. It is the ECC selection procedure that distinguishes between the two types of ECC memory. Adaptive ECCs based on RBER estimates and hierarchical ECCs are also possibilities. There are two types of ECC discussed in this article: the hierarchical ECC and the adaptive ECC. Both are based on RBER estimations. Memory's RBER is estimated using Type 1 ECC's ECC correction capability. A more robust error-correction technique is required to increase the projected RBER. Increasing the anticipated RBER Based on the target memory, the RBER prediction parameters may change. The number of program/erase (P/E) cycles and the retention time are used to identify the ECC type of NAND flash memory. In terms of threshold voltage fluctuations, SRAM's VTH fluctuation reliability should be superb (see [37]). After estimating the resilience, an error-correction code is generated. If a "0" becomes a "1" during a write operation, then the failure rate of the STT-MRAM may be calculated. When going from "0" to "1," it's critical to minimise the number of write failures. Type 1 ECC may be used to estimate the RBER or another error rate in memory. Because the memory controller counts P/E cycles and verifies the retention duration, NAND flash is an excellent option for type 1 ECC. When using SRAM or STT-MRAM, you don't need a second estimate block for RBER. Increased floor area and energy use are both costly. As a consequence of the encoder and decoder having built-in error-correction capabilities, most publications now need a significant amount of additional space.

CREATING NEW MEMORY USING BCH CODING AND COMPLETELY PARALLEL BCH DECODERS

Data Encoding and Decoding Algorithm for Primitive Binary Data GF denotes a binary Galois field of degree m over a basic binary BCH coding (2m). The following is a representation of the (n, k, d) BCH code over GF(2m):

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

Number of information bits: $k \geq 2^m - mt - 1$Minimum distance: $d \geq 2t + 1$.

The t-error is the technical term for this. When there are t or more mistakes in an n-digit sequence, BCH code correction fixes it. Because the memory device's bit count cannot be expressed using a BCH code, certain information bits must be omitted. Depending on the design goals, such as memory capacity, read/write latencies, and energy consumption, this may vary greatly. .... There is a 1010–103 nB difference in the RBERs of STT-MRAM, ReRAM, and PRAM. In order to help the brain generate new memories, below are the numbers 5–18,19–21: Systems, circuits, and architectures must be developed using appropriate methods to reduce RBER. When the block failure rate (BFR) is less than 105 and the ECC can correct two mistakes, the required BFR is achieved. DEC's BFR may rise substantially if he is offered the option of choosing TED. Different methods of decoding the DECTED BCH code have been found.[22] and [23]. When it comes to syndromes, there are two main types. The following is the (n, k, 6) DEC-TED BCH code parity test matrix:

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

Number of Errors	S_0	S_1,S_2
Non	0	00
Single-bit	1	01
Double-bit	0	10
Triple-bit	1	11

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

To determine whether the received codeword, v, has any mistakes, the syndrome vector S is calculated as follows:

$$S=v\cdot H^T=[v\cdot 1^T,v\cdot H^T{}_1,v\cdot H^T{}_3\quad]=[S0,\ S1,\ S3](2)$$

The GF(2 m) code is generated using a 1-bit vector S0 and m-bit vectors S1, S2, and S3. One-bit errors can be handled using the S1 vector alone since H1 may be used as the Hamming code parity check matrix. To fix a double-bit error, we join the S1 and S3 vectors. One thing to bear in mind is that the codeword's error count may be calculated using the different relationships between the S0, S1, and S3 vectors. The proposed decoder makes this link specifically in Section IV. Using the syndrome vectors calculated in step 2, the error location polynomial (ELP) has to be completed. DEC-TED BCH codes may represent the ELP.

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

It is possible to think about ELP coefficients as 2D arrays (2m). ELP coefficients are often calculated using the Berlekamp–Massey (BM) [24] method. How to Track Down the Issue: Once coefficients have been computed, this search is utilised to find ELP roots (2m), 0,1,...,n1 within GF elements using the Chien method (1 and 2). (3). There are a lot of mistakes in Step 3, therefore the codeword v is fixed in Step 4. Using XOR gates may be a possibility in this case. Emergence of Full-Parallel BCH Decoders in Emerging Memory

The time-consuming BCH method has traditionally been used to fix individual bits of data in NAND flash memory[25]–[27]. It is possible to decode long BCH codes using a linear feedback shift register and standard iterative BCH algorithms in 2n + 2t cycles. Increasing memory was not possible with this decoding technique due to the requirement for a highly parallel decoding architecture to achieve low latency decoding of just a few nanoseconds. Using as much parallelism and combinatorial logic as possible, decoding works as fast as possible without incurring additional hardware costs. In contrast, full parallelism costs money in terms of hardware.

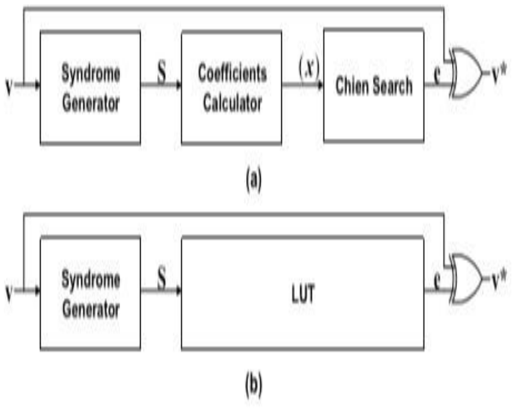


Fig. 1. Block diagrams of (a) PA-based decoder and (b) LUT-based decoder.

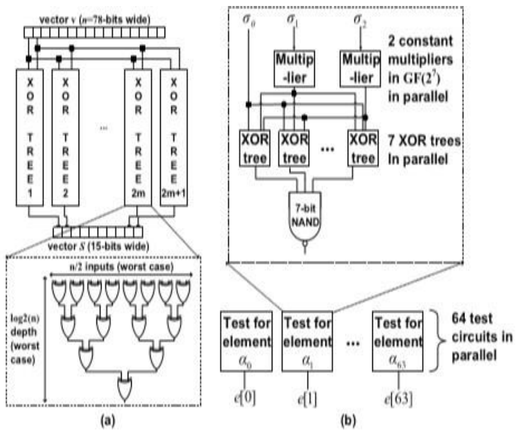


Fig. 2. (a) Syndrome generator and (b) Chien search blocks [31] for 64-bit codeword in the fully parallel DEC-TED BCH decoder.

The usage of dynamic memory is not required in your programme. Due to the absence of error correction and the restricted capacity of the memory array, a short BCH code[28]–[30] may be employed.

However, NAND flash memory is a kind of nonvolatile storage. S may be generated in a parallel structure using inputs from the code vector and the syndrome vector, as shown in Figure [31]. 2(a) There are two types of completely parallel BCH decoders: Peterson's algorithm (PA) and lookup table (LT). This is because the decoding technique and implementation methodology used to evaluate the ELP and locate the source are different (LUT). A Decoder-Based Algorithm Peterson was in charge of creating it. To cut down on the number of time-consuming iterations, an AIn PA was suggested. PA generates the ELP for the DEC-TED BCH code's ELP.

S21 + S3 / S1 x2 = 1+S1x + (S21 + S3 / S1) x2. (4)
Points of Reference (4)

You'll need a strong finite-field division to compute the coefficients. In [16], a RELP was suggested as a potential

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

solution because of the difficulties in assessing and calculating coefficients during the Chien search. As shown in Figure 1, a PA-based decoder has a basic layout (a). All that's required is modulo-2 addition and multiplication for each coefficient component. This may be accomplished by using the syndrome vectors' bit components in (5). The Chien search block uses basic logic operations to do many calculations of I for 0i n1. Determine if an input value, I, is 0 by multiplying it by the constant value illustrated in Figure 2 using a basic XOR-tree test circuit (b). I Rather than using the PA-based decoder's coefficients calculator and Chien search blocks, [17] proposes a new kind of LUT-based decoder. To identify each combination of symptoms and incorrect patterns, a LUT is utilised. Once the syndrome vector is calculated, the error locations may be identified using this decoder. Analyzing the Differences Between LUT and PA Decoders The error vector may be found once the syndrome vector has been calculated using the LUT-based decoder. LUT-based decoding has a higher area overhead than PA-based decoding, thus the latter has a shorter decoding latency. No matter how many mistakes there are to repair (t), or how much information there is to store, the table grows slowly (k) realising at the same time from the standpoints of space and time. The increasing area is considerably less when utilising a PA-based decoder than when using a LUT-based decoder. As a result, for small-scale applications, the PA-based decoder is better suited. A PA-based decoder uses less dynamic power than a LUT-based decoder while decoding audio. The syndrome vectors calculated in the decoder are used by coefficient calculators and Chien search blocks until the error vector is found. Because the syndrome vectors must be recalculated each time a new codeword is received, the dynamic power consumption is elevated. If the matching input syndrome vector is used instead, the LUT block only has to activate one circuit route due to the inherited LUT feature, resulting in reduced dynamic power consumption. Low-power applications benefit greatly from the LUT-based decoder.

INVESTIGATIONS IN CONVENTIONAL FULLY PARALLEL BCH DÉCOR

Error Probability-Based Decoding Issues In spite of advances in nanotechnology, single-bit mistakes in a codeword are still much more common than multibit errors, as seen in Figure 3. (errors of two or three bits). Only 0.9% of codewords need error correction when using 10-ppm RBER since the vast majority (99.1%) are error-free. In 99.6 percent of the situations, codeword mistakes are corrected using single error correction (SEC); decoder error correction (DEC) is needed in the remaining 0.4 percent of cases. Due to these probability variables, the DEC-TED decoder corrects single-bit and double-bit mistakes extremely seldom. The delay, area complexity, and power consumption of the DEC-TED BCH code decoder are all considerably greater than those of the SEC-DED code decoder. However, the results of the decoder are shown in Table II. 64-bit DEC-TED BCH code decoders synthesised on 65-nm technology may be utilised if the latency limitations of the SEC-DED and DEC-TED decoders do not exceed 1.5 and 3 nanoseconds. It uses more power, but it's smaller and has the same latency limit as the PA-based decoder. The SEC-DED decoder's latency will be halved.

According to Table II, as compared to the PA-based (LUT-based) DEC-TED decoder, the SEC-DED decoder uses 40% less space and consumes 20% less power. Inefficient decoding is caused by the DET-TED decoder's latency and power consumption. SEC-DED and DECTED decoders may be adaptively selected from a pool of candidates to decrease decoding delay and power consumption.

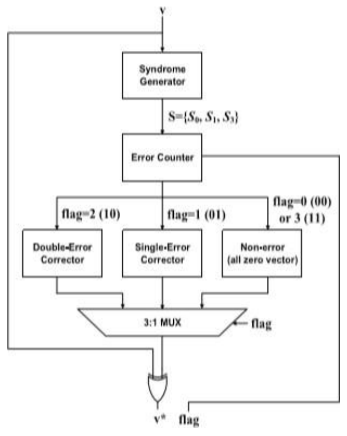


FIG 3 FLOW CHART DIAGRAM FOR PROPOSED SYSTE

PROPOSED HIGH DECODING-EFFICIENCY AND LOW-POWER DECODING ARCHITECTURE

INVESTIGATION INTO THE BCH DEC-TED An adaptive error correction and invalid transition inhibition based DEC-TED BCH decoder with excellent decoding efficiency while utilising minimal power is presented in this part.

i) DEC-TED BCH Decoder With AdaptiveError Correction

Figure 5 depicts a possible adaptive error correcting block design. When an input codeword contains mistakes, an error counter block generates a two-bit error flag signal. The next step is to create syndrome vectors. On the basis of the produced flag signal, several error correction techniques are conducted and an appropriate error vector is introduced to enhance decoding performance.

Codeword received from the 3:1 MUX and sent to the receiver. As indicated in Table III, a 2-bit flag signal may be constructed from the syndromevectors. Errors must be distinguished from non-errors by setting S0 to "1," whereas errors must be set to "0" when separating two bits. S3 1 plus S3 equals zero, thus the MBE for non- and single-bit errors is zero. An OR is formed by combining the vector bit components logically. Due to the lack of a zero in the double- and triple-bit error nonzero vector, MBE is "1." The SE and DE correctors may be used in conjunction with each other, or they can be used independently. In the design, SE correction uses Hamming SEC code and DE correction DEC BCH code. All zero vectors travel straight to the MUX without being processed in the majority of latency and power expensive error correction blocks since error correction methods for non-or triple-error situations are not needed (flag="00" or "11"). Nonor triple-bit error situations reduce latency and battery usage. The average decoding time and power consumption may be substantially decreased if only non-error situations are taken into account. As soon as a single-bit error is detected (flag = 01), the SE corrector conducts single-bit error correction on each H1 column. A single bit mistake in the received codeword makes it impossible to compare the proposed decoder to the standard SEC-DED codedecoder. Double-bit mistakes (flag = 10) may be corrected using the DE corrector, but the cost is latency and power consumption comparable to that of more traditional DEC-TED BCH decoders. Adaptive error correction in DEC-TED BCH decoders results in variable delay and power consumption. A PA-based decoder uses the suggested adaptive error correction method for each error scenario as a consequence of the synthesis findings in Table I, where denotes.

MUX, error correction block, and the typical delay of the PA-based DEC-TED decoder are all included in this sequence, as is the latency of the syndrome generator (which consumes power). The probability of each mistake situation may be used to determine the overall anticipated delay and power for 64-bit data words (shown in Fig. 2). The average latency and average power may be calculated using a 100ppm RBER (Paverage). The standard deviation (SD) is equal to 0.375 standard deviations. As a result of this, the formula is as follows: (6) No, single, and double-bit mistakes all have three probabilities (delay values) that may be calculated (T2). Paverage is defined as Pr0 = P0 + Pr1 = P1 + Pr2 = P2. These findings show that non-errorcases that occur the most often have a significant impact on average delay and power usage. Compared to a conventional PA-based decoder, the suggested decoder consumes 37% less power and has 37% lower latency, making it a viable option. Because of this, the average delay of a LUT-based decoder is equal to that of a PA-based decoder, with just half the power usage. It follows that DEC-TED decoder's adaptive error correction helps with latency and power consumption. The average power usage may be further lowered if the invalid transition issue can be resolved. After that, in Section IV-B, we'll go further into the topic. This may be done using a standard DEC-TED decoder that uses multiple concurrent error counters, SE correctors, and MUX blocks. The traditional decoder's coefficient calculator may be used as part of the error counter when using the PA-based DEC-TED decoder. In order to assess the flag [1] value, the error counter utilises a GF m-bit OR gate (2m). If you want to put it another way, the expenses related to

For adaptive error correction, the DEC-TED AEC decoder uses a PA-based error counter (AEC). Due to the SEC decoder error location block's modest size compared to the traditional TED's calculator and parallel Chien search blocks for the DEC coefficient, the area cost of a SE corrector is similarly low.

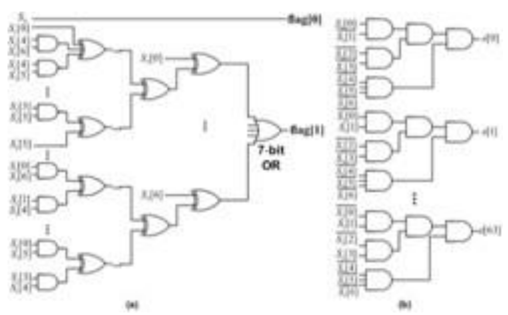


Fig4.(a)Error correctorand (b) SE correctorblocks in the fully parallel DEC-TED BCH decoder for the 64-bit codeword.

Decoder: Figure 6 shows the SE corrector and error counter hardware configurations seen in Figure 7. Using the SE corrector instead of the LUT-based DEC-TED decoder saves just a little amount of space and electricity. These error patterns in the LUT may be split in half for typical DEC-TED decoding systems. Single-bit and double-bit mistakes are the two kinds of errors that may occur. The proposed LUT-based decoder may make use of both the SE and DE correctors. The S0 vector is not needed when using our suggested LU since the error counter section already tells us how many mistakes there are. For the SE corrector, obtaining the error vector value is as simple as looking up m-bits in the m-bit S1 vector. A conventional LUT-based decoder uses SE and DE correcters, which take up less space than the LUT stack. Increased area, latency, and power consumption are unavoidable because of a double-bit mistake in the PA-based decoder. All other vectors (S1 and S3) are decoded by the PA-based (LUT-based) encoder using the DEC-FFs.

machine that keeps track of mistakes. Because the LUT block's needed size is less, there's a possibility it'll be negligible or compensated for. Only the DE corrector block separates the LUT-based and PA-based decoders in terms of hardware implementation. The LUT-based decoder's DE corrector utilises AND gates, much as the SE corrector. The DEC-TED Decoder's transition inhibition method was shown to be ineffective. As stated in Section III-B, it is essential to transfer the settled syndrome vectors to the SE or DE corrector in order to avoid incorrect transition. This decoder's SE and DE correctors should not be used at the same time to further reduce power usage. In order to meet these requirements, FFs are employed between the syndrome generator and the SE and DE rectifiers. Adaptive error correction and incorrect transition inhibition are shown in Figure 7 of a DEC-TED decoder. Also, bear in mind that this function makes use of FFs that are activated on the positive edge. This is an important consideration. SEC-FFs are FFs connected to the SE device that set them apart from other correction devices (DEC-FFs). Once the vector and flag of the syndrome are stable, all FF control signals will be triggered to ensure that all FFs pass the test. Figure 8 shows how the FFs control signal is generated by using a separate decoder clock (called the ECC clock). This uses an inverted ECC clock and the pulse length of the ECC clock should be larger than the sum of worst syndrome generator delay plus syndrome generator delay for positive-edge triggered FFs (FF clock in Fig. 8). (Tsynd). Basic INV and AND gates are used with an additional clock gating method to handle SEC and DEC-FF clock signals simultaneously. When a non- or triple-error bit is present, the vectors cannot be sent to the next FF block and must remain in memory. Only when a single bit error occurs is the resolved S1 vector sent to the SE corrector through the SEC-FFs. Since the error condition vectors aren't sent to the DE correction device when there's a single bit mistake, the power consumption is significantly reduced (DEC-FF). When, to use another expression, someone

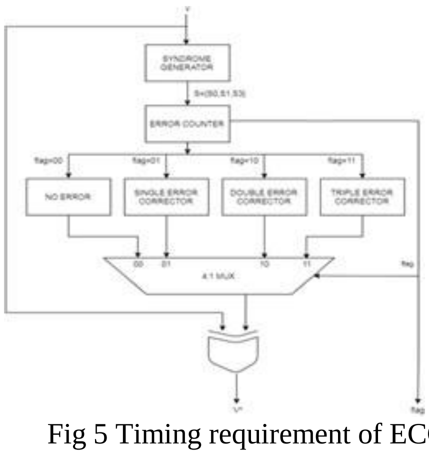


Fig 5 Timing requirement of ECC clock

A 65-nm cell library allowed the development of traditional PA and LUT-based DECTED BCH decoders. Both of these decoders have a delay limitation: one makes use of PAs, the other utilises LUTs. For double-bit errors, the standard DEC-TED BCH decoder has a maximum delay of 3 ns. For single-bit errors, the maximum delay is now 1 ns. Both the standard DEC-TED BCH decoders using PA and LUT and the suggested DEC-TED decoders using PA and LUT were synthesised.

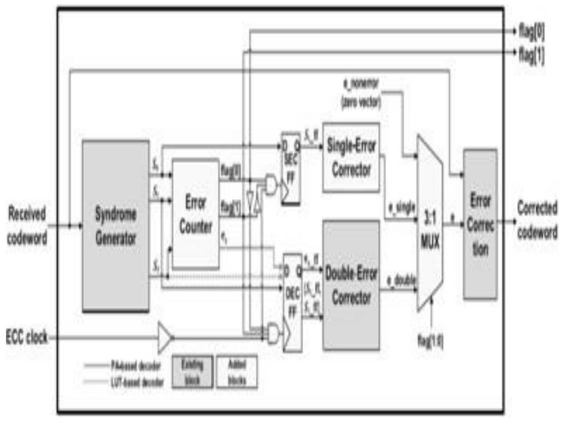


Fig 6 Block diagram of the proposed high decoding efficiency and low power DEC-TED decoder

III. COMPETENCE ADAPTIVE BCH DECODER:

These techniques are discussed in this part in order to improve the TEC BCH decoder's decoding efficiency while using less power. These calculations are presented in block diagram form using the adaptive error correcting technique. As a result, bug counter blocks are built using syndrome vectors, and a 2-bit error flag signal is produced for each defect in the acquired codeword that was caused by the vector. Depending on the flag signals produced and the correct error, several error correction techniques are employed to enhance decoding speed.

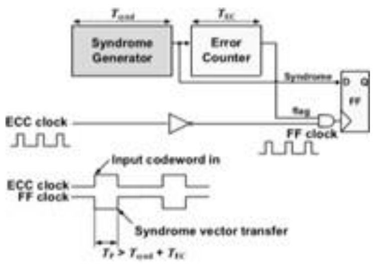


Fig 7 flowchart for extension method

The SE Corrector corrects the proposed layout using Hamming's SEC code, whereas the DE Corrector corrects the layout using DEC BCH code. There is no need to do double- or triple error case operations (flags = "00" or "11") for fixing most late and power-consuming errors, thus MUX may access a zero vector immediately without processing it. Delay and power consumption should be kept to a minimum while dealing with single- and triple-bit errors.

ALGORITHM:

Step-1) after syndrome generation we will gets0,s1,s3.

Step-2) send s0,s1,s3 to error counter t generateerror flag bits.

Step-3) by using error flag bits conditionally weselect path error correction bit.

Step-4) here if flag bits are 00 then it is send to no error block.

Step-5)if it is 01 then it is send to single error correction block ,if it is 10 it means double erroris present in information then it is send to doubleerror corrector.

Step-6) if it is 11 then triple bit error detected then it information send to triple error correctionblock .

Step-7) after error correction blocks it will selectfinal output from mux.

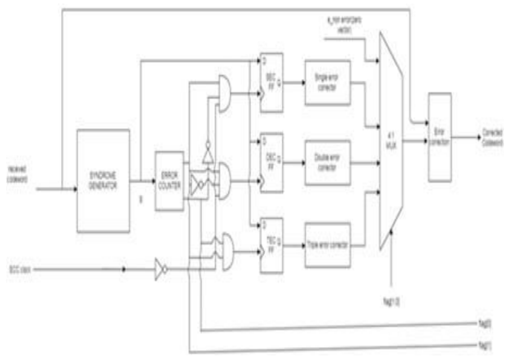
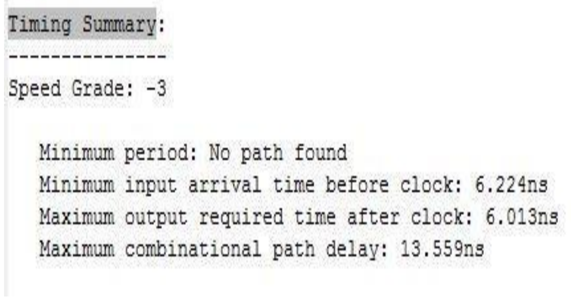
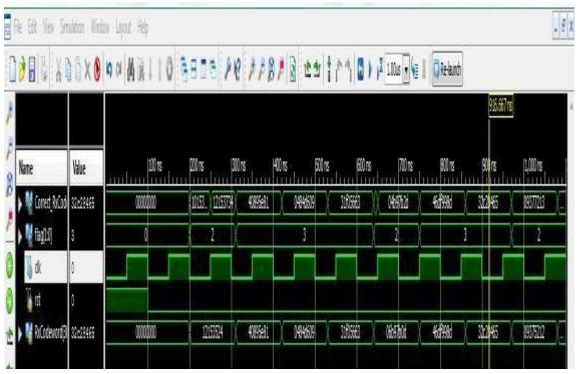


Fig 8 Block Diagram of extension/proposed method with triple error correction.

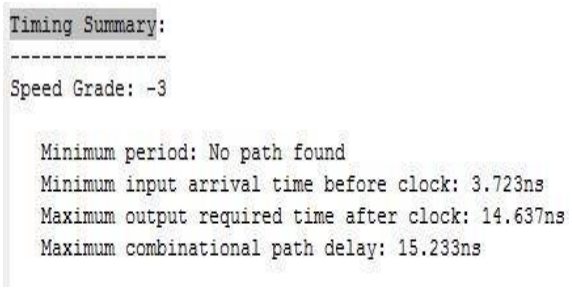
IV. SIMULATION RESULTSPROJECTED SYSTEM:



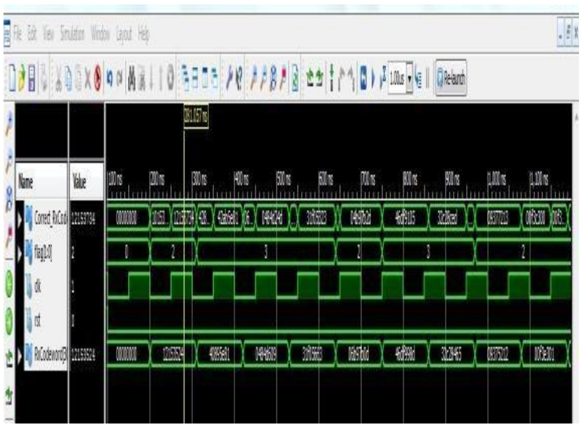
```
Timing Summary:
---------------
Speed Grade: -3

   Minimum period: No path found
   Minimum input arrival time before clock: 6.224ns
   Maximum output required time after clock: 6.013ns
   Maximum combinational path delay: 13.559ns
```



RTL SCHEMATIC

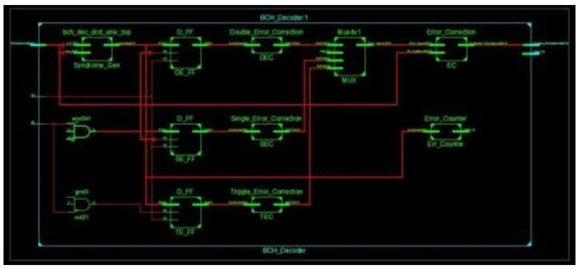


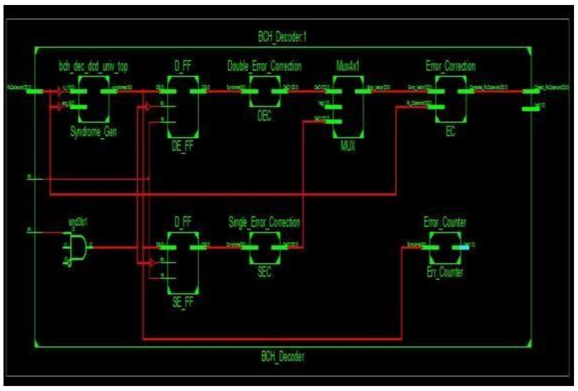
```
Timing Summary:
---------------
Speed Grade: -3

   Minimum period: No path found
   Minimum input arrival time before clock: 3.723ns
   Maximum output required time after clock: 14.637ns
   Maximum combinational path delay: 15.233ns
```



ED EXTENSION SYSTEM RESULTS:

RTL SCHEMATIC





COMPARISON TABLE:

S.No	SYSTEM	delay
1	ADAPTIVE BCH DECODER	15.232ns
2	ADAPTIVE BCH DECODER WITH TRIPLE ERROR CORRECTION	13.559ns

V. CONCLUSION

To provide memory with excellent decoding performance and low power consumption, these researchers created a high-performance BCH DEC-TED decoder (79, 64, 6). We developed an adaptive error correction technique to improve decoding performance in terms of latency and power consumption. This method chooses an algorithm based on the error conditions in a codeword. Modern fully parallel BCH decoders use an excessive amount of dynamic energy during decoding, thus an incorrect transfer inhibition technique with FF and a separate ECC clock was selected to solve this problem.

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 costeffective 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-performanceCPU,” 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. MemoryWorkshop (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 correctingcode,” 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-ScaleComput. 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 BCHdecoder for lowcomplexity fault-tolerant systems,” inProc. 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, SignalProcess. (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 HfOxbased resistive memory," in Proc. IEEE Int. Memory Workshop (IMW), May 2015, pp.1–4.

[21] S. Sills et al., "Challenges for high-density 16 GbReRAM 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-errorcorrecting 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.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.

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