# Enhanced matrix-based error correction coding techniques for embedded memories

**Jammula Shivani, Mandadi Shankar Teja, Manickaraj Vinodhini**
Department of Electronics and Communication Engineering, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Bengaluru, India

**Article Info**

***Article history:***

Received Sep 5, 2023
Revised Apr 29, 2024
Accepted May 26, 2024

***Keywords:***

Correction masking technique
Error correction code
H-matrix
Memories single bit upsets
Multiple bit upsets

**ABSTRACT**

Memories play a very important role in computing systems due to the continuous advancements in technology. They are used to store data that is used for proper system operations. Memory architectures that are more intricately designed are more prone to radiation-induced errors such as single bit upsets (SBU) and multiple bit upsets (MBU). Error correction codes (ECC) are used to recover the corrupted data that are stored in memories. H-matrix-based ECC is the commonly used ECC for memories. On the other side, the correction masking (CM) technique was added to ECC to mask the correctable error patterns. CM technique protects the error-free bits while correcting erroneous bits. In this paper, optimized H-matrices are presented. These matrices are used to design an ECC with the CM technique to correct 2-bit to 7-bit adjacent errors. The result shows there is a reduction in the power of 2, 3, and 4 adjacent bit errors by 19.009%, 4.615%, and 27.934% respectively.

***Corresponding Author:***

Manickaraj Vinodhini
Department of Electronics and Communication Engineering, Amrita School of Engineering
Amrita Vishwa Vidyapeetham
Bengaluru, India
Email: m_vinodhini@blr.amrita.edu

## 1. INTRODUCTION

Memories are referred to as the storage space in computing systems. The encoded data is kept in memory and then retrieved, decoded for error checking, and subjected to additional processing. Memories use different storage spaces for storing single-bit and multi-bit data. In today's modern technology, semiconductor memories are used to store data in the computing system. Different types of memories can be ranked according to their capacity and accessing speed such as read only memory (ROM), static random access memory (SRAM), dynamic random access memory (DRAM), and so on [1].

In today's modern technology, the reliability of the data is more important for the functioning of digital systems. The data in the memories can be corrupted due to various noise sources and also due to the ionizing effect of atmospheric neutrons and alpha particles [2], [3]. In memories, there is also a chance of occurrence of single bit upsets (SBU) or multiple bit upsets (MBU) due to cosmic radiations [4]–[7]. It is important to identify and rectify the error present in the data stored, as the computations are very sensitive to the errors. When the memories are connected to many electronic devices or because of the complementary metal-oxide semiconductor (CMOS) technology's downscaling in terms of architecture, the memories tend to experience soft errors [8].

A method for error detection and correction in computing systems for corrupted data in memories is called error correction codes (ECC). Initially, ECC is introduced for the correction of SBUs later they are implemented for MBUs. The ECC method used in SRAM is power efficient and capable of multi-bit error detection and correction as well [9]–[11]. As technology is scaling down, memories tend to experience errors in numerous bits, and ECC which rectifies those error bits became popular [12]–[17]. According to errors present in the data, different types of ECC are introduced to identify and rectify them [18]. Error detection and correction are accomplished using many ECC techniques, one of which is the H-matrix-based technique. The size of the H-matrix increases as the overall number of errors in the memory-stored data rises.

The correction masking (CM) technique is an existing, effective, and structured technique, which is implemented in syndrome-based error correction decoders. Implementing this H-matrix method with the CM technique can ensure more error protection. The CM technique can be applied to any syndrome-based error correction decoder which comprises many ECCs such as single error correction (SEC), SEC double adjacent error correction (SEC-DAEC), 3-bit adjacent, and 4-bit adjacent errors. Previously for effective protection of data, the triple modular redundancy (TMR) method was used. This method results in more consumption of area and power and provides less protection compared to the CM technique [19].

A coding method that reduces many errors at the time of transmission of data bits in memories is the diagonal hamming method [18]. This method detects and rectifies up to a maximum of 8-bit errors for 32-bit input data. Using this method, the reliability of error correction with the requirement of less area and latency is achieved. By incorporating the CM approach into the syndrome-based error correction decoders, Liu *et al.* [19] increase the error correction capacity and also outperform TMR in terms of logic, latency, area, and power. CM technique refers to the masking of correctable error patterns such that error-free bits may not experience errors while correcting erroneous bits. This technique can be implemented in any ECC for correcting both adjacent and random errors. In this study, a memory-efficient architecture of ternary content addressable memory (TCAM) based on multipumping-enabled multiple port SRAM is proposed. The approach operates the SRAM blocks at a frequency that is several times higher than the system as a whole [20], [21].

This work focused on a method that comprises two blocks of logic for error detection and correction i.e., they are 3 parity check (PC) and horizontal vector hamming code (HVHC) blocks in semiconductor memories [22]. The 3PC block is used to find and fix problems in input data, whereas the HVHC block is used to detect and fix faults in redundant bits. The technique mentioned above can detect and correct up to 3-bit errors in redundant bits and data bits. This method increases error correction capacity and reliability more compared to horizontal vector diagonal (HVD) and multidirectional methods. However, this method introduces increased complexity to the system, and consumption of more circuit area and power, as it uses two different blocks for detecting and correcting data and redundant bits. The aforementioned combination method increases the correction capacity, but there can be a trade-off between identifying and fixing bit faults in redundant bits against those in input data bits. The column-line-code (CLC) method which is an alternative to hamming code, discusses the detection and correction of error bits in memories. The CLC method requires more parity bits to identify and rectify errors in comparison to hamming code [23]. Although the above method has high error correction capacity, the main disadvantage is the requirement of more redundant bits for this method which tends to increase in the circuit area, which in turn, increases power and delay parameters. Silva *et al.* [24] introduced the extended CLC code and altered CLC code in numerous ways. While still using the same CLC code matrix, this process went through more cycles to uncover new flaws. The largest drawback, though, was the amount of power needed to use the procedure. Saiz-Adalid *et al.* [25] explained the technique regarding ultra-fast error detection and correction till 2 adjacent bit errors. The work mainly discusses the extended hamming code technique to detect and correct 2 adjacent bit errors. The H-matrix method is used in this method to find and fix errors. The error detection and correction process's delay is reduced by the H-matrix's design.

This paper proposes a novel optimized H-matrix that can rectify all lower-order adjacent errors and till 7-bit of adjacent errors in 16-bit data which is simulated using Modelsim and synthesized using Cadence Genus 45 nm technology. The recursive search methodology is used to identify the syndrome bits with the lowest hamming weights and produce the optimized H-matrix. Using the recursive search approach, the H-matrices are optimized for 2-bit to 6-bit adjacent errors with fewer number of one's such that it reduces either one of the parameters i.e., area, power or delay. Then these optimized H-matrices are used to design the ECC and implemented with the existing CM technique for MBU correction.

The work is organized as follows: in section 2, which describes the proposed work, a brief discussion of syndrome-based decoding using the CM technique is provided, as the generation of optimized H-matrix, and also about the implementation of optimized H-matrix in syndrome-based decoder with the CM technique. The comparisons and evaluations of the synthesis findings are examined in section 3. The conclusion and upcoming works are given in section 4.

## 2. PROPOSED WORK

### 2.1. Syndrome-based decoder with CM technique

The syndrome-based decoder is utilized in error detection and correction. It is implemented to tackle different errors like SEC double-error detection (SEC-DED), SEC-DAEC, and 3-bit to 7-bit adjacent errors. By using single error detection and CM, it is ensured that SBUs that affect the decoder when it receives a valid word as the output data will not be influenced by input data. The syndrome vector computation is done using PC equations. The comparison of the vector is done with the syndrome of every correctable pattern in the H-matrix to identify the error bit positions. The (1) below demonstrates the computation of syndrome vector $S_k$ by ORing of received vector and transposing PC equations. These PC equations are implemented as individual, independent blocks with no logic sharing. After syndrome computation, if all the bits in the syndrome vector are zero, it means that the input data is error-free. The syndrome vector will be non-zero even if the input data contains a single bit of error, and it is compared with the columns of the H-matrix to determine where the error in the input data bits is located. If more than one-bit error is present in input data, the identification of that error position is done by comparing the syndrome with XORing of the particular columns corresponding to that error position and particular error signals $e_k$ are generated.

$$S_k = rH^T(k) \tag{1}$$

Where: $S_k$ is syndrome vector, r is received vector, and H is PC matrix. After the computation of $S_k$, syndrome bits are ORed with $S_r$ to produce the masking variable signal $Corr_{en}$ as shown in (2) which is used for masking error bits. The $Corr_{en}$ signal is ANDed with error signals $e_k$ to obtain the masked output. The design specifies that the correction will take place if both the $e_k$ and $Corr_{en}$ signals are one. Now the masked output is XORed to the input data $d_k$ to obtain the error-free final output data $d_{kcorrcm}$ as shown in (3):

$$Corr_{en} = S1 + S2 + \; + Sr \tag{2}$$

$$d_{kcorrcm} = d_k \oplus (e_k Corr_{en}) \tag{3}$$

the above syndrome-based decoder with CM Technique will protect the data in the memories from SBUs and MBUs with simpler logic, efficient protection, minimal increase in power, area, and delay compared to only the syndrome-based decoder. In the next subsection, the optimization of H-matrices is discussed.

### 2.2. Generation of optimized H-matrix

This section discusses how to generate an H-matrix that is optimized for 7-bit adjacent error correction using 16-bit input data. First, it is necessary to compute the parameters of the syndrome vector and the total amount of redundant bits needed. Next, all potential mistake sequences that might happen had to be forecasted. Then to determine the properties of the syndrome vector it is necessary to calculate the number of parity bits.

The possible number of error sequences that can be detected in 7-bit adjacent ECC are 1-bit errors (__1__), 2-bit errors (__11__), 3-bit errors (__111__), (__101__), 4-bit errors (__1111__), (__1011__) (__1101__) (__1001__), 5-bit errors (__11111__), (__10111__) (__11011__) (__11101__) (__10011__) (__11001__) (__10101__) (__10001__) and 6-bit errors (__111111__), (__101111__) (__1101111__) (__111011__) (__111101__) (__1111101__) and so on, 7-bit error (__1111111__). As a result, the calculation of the total redundant bits required to correct every error is provided (4).

$$m + (m-1) + 2(m-2) + 4(m-3) + 8(m-4) + 16(m-5) + (m-6) \leq 2^{m-k} \tag{4}$$

Where: k is the number of input data bits and m is the number of received codeword bits. According to (4), the minimum number of redundant bits that are required to decode 7-bit adjacent error is calculated as 10. Hence, the code length m=(16+10)=26. The column vectors needed to build the optimized H-matrix are now computed. The goal of H-matrix optimization is to reduce the number of ones in the matrix, which equates to fewer gates being implemented, ultimately resulting in a reduction in area, power, or delay parameters.

To generate the H-matrix, the steps presented in Figure 1 are followed. To start this process first, all possible syndrome vectors for the error mentioned in the above paragraph are to be found. The vectors that have a minimal number of ones have to be selected. The next column vector is selected after the first one so that none of the remaining syndrome vectors should match any of the combinations that were made using this column. This process is repeated until all low-weighted column vectors are checked. To ensure that all the vectors and their related syndrome bits are covered, this search traverses through the same weighted vectors. This phase ensures that every vector selected has a reduced order weight, producing an optimized H-matrix. If the syndrome bits do not match after checking every vector of lower order weight, the search proceeds on to vectors of subsequent higher order weights. The vectors with higher order weights are also used in this recursive search. After this process, the H-matrix for 7-bit adjacent error is generated and shown in Figure 2. By following the same procedure mentioned in section 2.2, H-matrices are optimized with a lesser number of ones for 2-bit to 6-bit adjacent errors for 16-bit input data as shown in Figures 2(a) to 4(b).

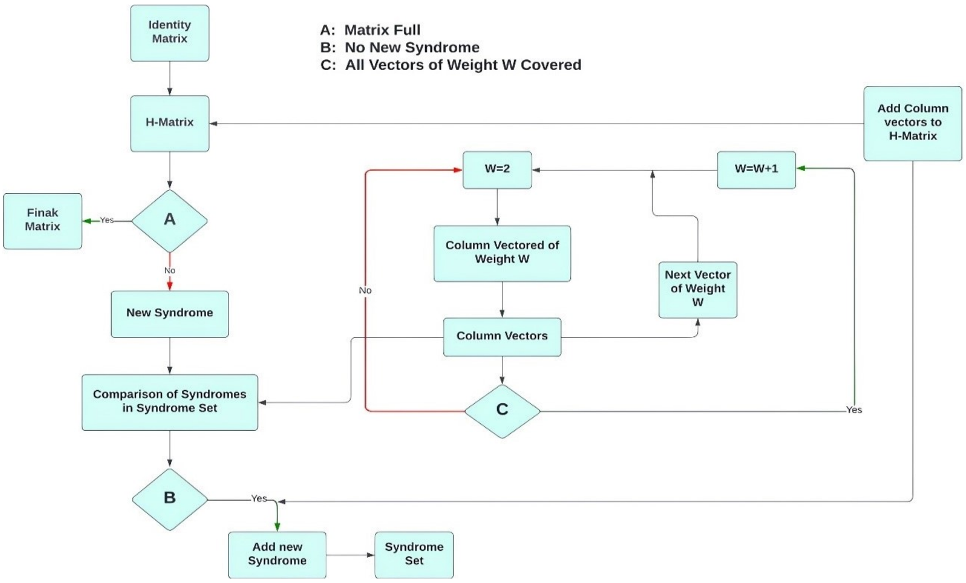


Figure 1. lowchart for generation of optimized H-matrix

$$H = \begin{bmatrix}
1&0&0&0&0&0&0&1&1&1&0&0&0&1&1&0&0&0&1&1&1&0&0\\
0&1&0&0&0&0&0&0&0&0&1&1&0&1&0&1&0&0&1&0&1&1&1\\
0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&1&0&0&0&0&1&0\\
0&0&0&1&0&0&0&0&0&0&1&0&0&1&0&0&0&1&0&1&0&0&0\\
0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&1&0&0&1&0&0&1&0\\
0&0&0&0&0&1&0&0&0&0&0&0&1&0&0&1&0&1&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&1&0&0&1&0&0&0&0&1&0&0&1&0&1
\end{bmatrix}$$

(a)

$$H = \begin{bmatrix}
1&0&0&0&0&0&0&1&1&1&0&0&0&1&1&0&0&0&1&0&1&0&1\\
0&1&0&0&0&0&0&0&0&0&1&1&0&1&0&1&1&1&0&0&1&0&1\\
0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&1&0&0&1&0&0&0\\
0&0&0&1&0&0&0&0&0&0&1&0&0&1&0&0&0&0&1&0&0&1&0\\
0&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&1&0&0\\
0&0&0&0&0&1&0&0&0&0&0&1&0&0&1&0&1&0&0&1&0&0&1\\
0&0&0&0&0&0&1&0&0&1&0&0&1&0&0&1&0&0&0&0&0&1&0
\end{bmatrix}$$

(b)

Figure 2. Optimized H-matrix for error correction; (a) 2-bit adjacent error and (b) 3-bit adjacent error

$$H = \begin{bmatrix}
1&0&0&0&0&0&0&0&1&1&1&0&0&0&0&1&1&0&1&0&0&0&1&0\\
0&1&0&0&0&0&0&0&0&0&0&1&1&0&0&0&1&0&0&0&0&1&1&1\\
0&0&1&0&0&0&0&0&1&0&0&0&0&1&1&0&0&0&1&0&0&0&0&1\\
0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&1&1&0&0&1&0&0&0&0\\
0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&1&0&1&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&0&1\\
0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&1&0&0&1&0&0&1&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&1&0&1&0&0&0&0&0&1&0&0&0
\end{bmatrix}$$

(a)

$$H = \begin{bmatrix}
1&0&0&0&0&0&0&0&1&1&1&0&0&0&0&0&0&1&1&0&1&1&0&0\\
0&1&0&0&0&0&0&0&0&0&0&1&1&1&0&0&0&1&0&0&1&0&0&1\\
0&0&1&0&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&1&0&0&1&0\\
0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&1&0&1&0&0&0&1&0&0\\
0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&0&1&0&0&0&1&0&0&1\\
0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&1&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&1&0&0&0&0&1&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&1&0&0&1&0&0&0&0&0&1&0
\end{bmatrix}$$

(b)

Figure 3. Optimized H-matrix for error correction; (a) 4-bit adjacent error and (b) 5-bit adjacent error

$$\mathbf{H} = \begin{bmatrix} 1000000001100111010100000 \\ 0100000000010000001011000 \\ 0010000001000000100000010 \\ 0001000000001010001000001 \\ 0000100000101000000010000 \\ 0000010000010001000000000 \\ 0000001000000100100000100 \\ 0000000100000000010001010 \\ 0000000010000000000100101 \end{bmatrix}$$

(a)

$$\mathbf{H} = \begin{bmatrix} 10000000001111000000001000 \\ 01000000000000111100000100 \\ 00100000001000000011000011 \\ 00010000000000100000001000 \\ 00001000000100000000100100 \\ 00000100000000010010010000 \\ 00000010000010000000000010 \\ 00000001000000001000100000 \\ 00000000100001000001000000 \\ 00000000010000000100010001 \end{bmatrix}$$

(b)

Figure 4. Optimized H-matrix for error correction; (a) 6-bit adjacent error and (b) 7-bit adjacent error

### 2.3. Implementation of optimized H-matrix in syndrome-based decoder with CM technique

In this section, the implementation of the optimized H-matrix in the syndrome-based decoder with the CM technique is presented with an example. Let us consider a 16-bit data c=0011001000011010 first this data is sent through the encoder and the redundant bits are calculated and attached to the original data. So, the data stored in the memories is r=01011010110011001000011010, in that 1st 10 bits are the parity bits which will help in error detection and correction. Now let us introduce errors in the data bits at the $2^{nd}$ to $8^{th}$ position. Now the codeword will change to r=01011010110011001111100110.

This data is now sent to the decoder side, and the decoder will now calculate the syndrome for the codeword. If there is no error in the input data the syndrome will be zero based on the retrieved data, the identification of that error position is done by comparing the syndrome with XORing of the particular columns corresponding to that error position as shown in Table 1 at the positions at which the bits are flipped and the resulting s=0011111110 at this time the ORing of each syndrome bits is done and stored in variable $Corr_{en}$ which will be used in the masking technique after the columns of the H-matrix will be compared with the syndrome vector when it has been calculated. The generated syndrome vector is used to identify the column vectors. The column vector identifies the location of the erroneous bits. As the error bits are identified now the error bits are ANDed with variable $Corr_{en}$ which is part of the masking process this will provide additional protection to the error bits.

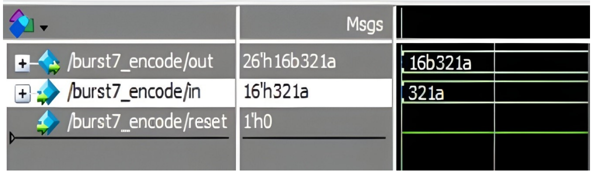
Table 1. Calculation of syndrome vector using column vectors

| H[2] | H[3] | H[4] | H[5] | H[6] | H[7] | H[8] | H[2] ⊕ H[3] ⊕ H[4] |
|---|---|---|---|---|---|---|---|
| | | | | | | | H[5] ⊕ H[6] ⊕ H[7] |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 1 | 0 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

## 3. EVALUATIONS AND COMPARISONS

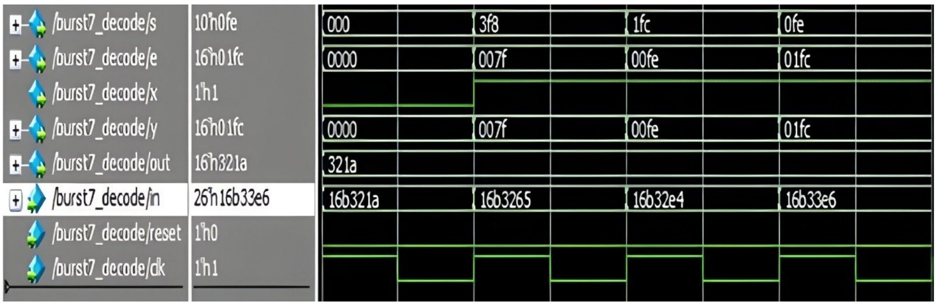
The proposed work has been compared with existing H-matrices of ECC and the benefits of the proposed work have been discussed in this section. The encoders and decoders are designed in Modelsim using Verilog as this tool simulates behavioural, return to learn (RTL), and gate-level code, resulting in improved design quality and debug productivity and synthesized in Cadence 45 nm technology as this technology is the latest technology that gives accurate results which is 10 times faster than other technology. Figure 5 shows the encoding of the input data i.e., in=321a and the output of the encoder will be input data appended with parity bits i.e., out=16b321a and this will as given as input to the decoder.

Figure 6 shows the syndrome-based decoder with the CM technique in which the output of the encoder is provided with and without introducing error. Firstly, the data is considered as 16b321a which does not have an error in it. Hence, syndrome bits are obtained as s=000, and the final output of the decoder after correction will be out=321a which is the original data. Now, let's consider a 7-bit adjacent error in the input data i.e., in= 16b3265. Since the error is introduced, syndrome bits as non-zero i.e., s= 007f, and the final output of the decoder after correction will be out=321a and similarly, simulations are carried out for the other two cases.



| Msgs | |
|---|---|
| /burst7_encode/out | 26'h16b321a |
| /burst7_encode/in | 16'h321a |
| /burst7_encode/reset | 1'h0 |

Figure 5. The output of a 7-bit error correction encoder using the CM technique in memories



| | |
|---|---|
| /burst7_decode/s | 10'h0fe |
| /burst7_decode/e | 16'h01fc |
| /burst7_decode/x | 1'h1 |
| /burst7_decode/y | 16'h01fc |
| /burst7_decode/out | 16'h321a |
| /burst7_decode/in | 26'h16b33e6 |
| /burst7_decode/reset | 1'h0 |
| /burst7_decode/clk | 1'h1 |

Figure 6. The output of a 7-bit error correction encoder using the CM technique in memories with and without error

Table 2 shows the synthesized results of the ECC with the existing and optimized H-matrix for 2 to 4 adjacent bits error with the CM technique. The synthesized results provided the performance parameters such as area, power, and delay. From the synthesis results it is inferred that for 2-bit adjacent error, there is a decrease in parameters in terms of area, power, and delay by 11.44%, 19.009%, and 3.314% respectively compared to the existing matrix, for 3-bit adjacent error there is a decrease in the parameter in terms of power by 4.615% and minimal overhead in terms of delay and area, for 4-bit adjacent error there is a decrease in the parameter in terms of delay and power by 13.521% and 27.934% and minimal overhead in terms of area. Table 3 shows the synthesized results of the ECC with the proposed H-matrix for 5 to 7 adjacent bits error with the CM technique.

Table 2. Area, power, and delay parameters for 2 to 4 bits adjacent ECC with existing and optimized H-matrices

| No. of error | No. of parity bits used | | Area (µm$^2$) | | Power (µW) | | Delay (ns) | |
|---|---|---|---|---|---|---|---|---|
| bits corrected | Existing | Optimized | Existing | Optimized | Existing | Optimized | Existing | Optimized |
| 2 | 8 | 7 | 612.625 | 542.520 | 0.2504 | 0.2028 | 0.3349 | 0.3238 |
| 3 | 7 | 7 | 535.906 | 542.168 | 0.2521 | 0.2419 | 0.4202 | 0.4594 |
| 4 | 7 | 8 | 542.376 | 547.472 | 0.2667 | 0.1922 | 0.3831 | 0.3313 |

Table 3. Area, power, and delay parameters for 5 to 7 bits adjacent error ECC with proposed H-matrix

| No. of error bits corrected | No. of parity bits used | Area (µm$^2$) | Power (µW) | Delay (ns) |
|---|---|---|---|---|
| 5 | 8 | 537.648 | 0.1791 | 0.4392 |
| 6 | 9 | 536.221 | 0.1795 | 0.4382 |
| 7 | 10 | 683.746 | 0.2647 | 0.4401 |

In a nutshell, the results from this work's use of optimized H-matrices state that the complexity of encoding and decoding operations decreased as a result of the optimization technique's reduction of the number of ones in the H-matrix. It also asserts that employing optimized H-matrices instead of conventional H-matrices will result in a reduction of at least one parameter, such as area, power, or latency.

## 4. CONCLUSION AND FUTURE SCOPE

The work in this paper has discussed the implementation of optimized H-matrices in syndrome-based decoders along with the CM technique. It is one of the effective methods for safeguarding syndrome-based decoders from SBUs and MBUs. For the above work, we simulated it using the Modelsim tool, synthesized the code and analyzed the results of Cadence Genus in 45 nm technology. Using an optimized PC matrix, the CM technique was implemented, evaluated, and analyzed for a variety of codes, including 2-bit to 7-bit adjacent codes for memories. The result shows there is a reduction in the power of 2, 3, and 4 adjacent bit errors by 19.009%, 4.615%, and 27.934% respectively and erroneous bits are efficiently corrected using optimized H-matrices. The outcomes show that using an optimized PC matrix can deliver effective security while reducing area, power, and latency. As a result, CM appears to be an appealing technique that can be used in memory to provide good error protection while fixing erroneous bits. The future scope is, that protection is only provided to the input data bits, not to the redundant bits so by protecting the redundant bits the syndrome vector can also be protected. Only adjacent bit errors are taken into consideration so we can increase the efficiency of the technique by applying it to the random error bits.

## REFERENCES

[1] S. Vijayakumaran and D. Pal, “On the minimum redundancy of SEC-DAEC-TAEC binary linear block codes,” *IEEE Communications Letters*, vol. 20, no. 4, pp. 652-655, Apr. 2016, doi: 10.1109/LCOMM.2016.2532884.

[2] K. Dang and X. Tran., “Parity-based ECC and mechanism for detecting and correcting soft errors in on-chip communication,” in *2018 IEEE 12th International Symposium on Embedded Multicore/Many-core Systems-on-Chip*, Hanoi, Vietnam, pp. 154-161, 2018, doi: 10.1109/MCSoC2018.2018.00035.

[3] G. Tsiligiannis *et al.*, “Multiple cells upset classification in commercial SRAMS,” *IEEE Transactions on Nuclear Science*, vol. 61, no. 4, pp. 1747–1754, 2014, doi: 10.1109/TNS.2014.2313742.

[4] J. Gracia-Mor´an, L. J. Saiz-Adalid, D. Gil-Tom’as, and P. J. Gil-Vicente, “Improving error correction codes for multiple-cell upsets in space applications,” in *IEEE Transactions on Very Large Scale Integration Systems*, vol. 26, no. 10, pp. 2132-2142, Oct. 2018.

[5] J. Guo, L. Xiao, Z. Mao, and Q. Zhao, “Enhanced memory reliability against multiple cell upsets using decimal matrix code,” in *IEEE Transactions on Very Large Scale Integration Systems*, vol. 22,no. 1, pp. 127-135, Jan. 2014, doi: 10.1109/TVLSI.2018.2837220.

[6] D. C. C. Freitas, D. F. M. Mota, C. Marcon, J. A. N. Silveira, and J. C. M. Mota, “LPC: an error correction code for mitigating faults in 3D memories,” *IEEE Transactions on Computers*, vol. 70, no. 11, pp. 2001-2012, Nov. 2021, doi: 10.1109/TC.2020.3034400.

[7] S. Liu, P. Reviriego, and F. Lombardi, “Codes for limited magnitude error correction in multilevel cell memories,” *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 67, no. 5, pp. 1615-1626, May. 2020, doi: 10.1109/TCSI.2019.2961847.

[8] C. Bhargavi, D. V. R. Nishanth, P. Nikhita, and M. Vinodhini, “H-matrix based error correction codes for memory applications,” *2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies*, Bhilai, India, 2021, 1-5, doi: 10.1109/ICAECT49130.2021.9392574.

[9] A. Varada and S. Agrawal, “An efficient SRAM-based ternary content addressable memory (TCAM) with soft error correction,” in *2021 5th International Conference on Electronics, Materials Engineering and Nano-Technology*, Kolkata, India, 2021, pp. 1-6, doi: 10.1109/IEMENTech53263.2021.9614696.

[10] I. Ullah, J. Yang, and J. Chung, “ER-TCAM: a soft-error-resilient SRAM based ternary content-addressable memory for FPGAs,” in *IEEE Transactions on Very Large-Scale Integration Systems*, vol. 28, no. 4, pp. 1084-1088, Apr. 2020, doi: 10.1109/TVLSI.2020.2968365.

[11] P. Reviriego, S. Pontarelli, and A. Ullah, “Error detection and correction in SRAM emulated TCAMs,” in *EEE Transactions on Very Large Scale Integration Systems*, vol. 27, no. 2, pp.486–490, Feb. 2019, doi: 10.1109/TVLSI.2018.2877131.

[12] J. Li, L. Xiao, P. Reviriego, and R. Zhang, “Efficient implementations of 4-bit burst error correction for memories,” *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 65, no. 12, 2018, pp. 2037-2041, doi: 10.1109/TCSII.2018.2817390.

[13] N. Farheen and K. S. Pande, “Error detection and correction using RP SEC-DED,” *2020 4th International Conference on Electronics, Materials Engineering, and Nano-Technology*, Kolkata, India, 2020, pp.1-6, doi: 10.1109/IEMENTech51367.2020.9270062.

[14] S. Cha and H. Yoon, “Efficient implementation of single error correction and double error detection code with check bit precomputation for memories,” *Journal of Semiconductor Technology and Science*, vol. 2, no. 4, pp. 418-425, 2012, doi: 10.5573/JSTS.2012.12.4.418.

[15] J. Li, P. Reviriego, L. Xiao, C. Argyrides, and J. Li, “Extending 3-bit burst error-correction codes with quadruple adjacent error correction,” in *IEEE Transactions on Very Large Scale Integration Systems*, vol. 26, no. 2, pp. 221-229, Feb. 2018, doi: 10.1109/TVLSI.2017.2766361.

[16] N. Sridevi, K. Jamal, and K. Mannem, “Implementation of error correction techniques in memory applications,” *2021 5th International Conference on Computing Methodologies and Communication*, Erode, 2021, pp. 586-595, doi: 10.1109/ICCMC51019.2021.9418432.

[17] K. S. Karan, N. Srikanth, and S. Agrawal, “A robust code for MBU correction till 5-bit error,” *2019 International Conference on Communication and Electronics Systems*, Coimbatore, India, 2019, pp. 1524-1529, doi: 10.1109/ICCES45898.2019.9002099.

[18] G. M. Sai, K. M. Avinash, L. S. G. Naidu, M. S. Rohith, and M. Vinodhini, “Diagonal hamming based multi-bit error detection and correction technique for memories,” *2020 International Conference on Communication and Signal Processing*, Chennai, India, 2020, pp. 0746-0750, doi: 10.1109/ICCSP48568.2020.9182249.

[19] H. Liu, P. Reviriego, C. Argyrides, and L. Xiao, “Correction masking: a technique to implement efficient SET tolerant error correction decoders,” in *IEEE Transactions on Device and Materials Reliability*, Vol. 22, no. 1, pp. 36-41, Mar. 2022, doi:

10.1109/TDMR.2021.3132045.

[20] I. Ullah, Z. Ullah, and J. A. Lee, “Efficient TCAM design based on multi pumping-enabled multi-ported SRAM on FPGAA,” *IEEE Access*, vol. 6, pp. 19940–19947, 2018, doi: 10.1109/ACCESS.2018.2822311.

[21] P. Reviriego, A. Ullah, and S. Pontarelli, “PR-TCAM: efficient TCAM emulation on xilinx FPGAs using partial reconfiguration,” in *IEEE Transactions on Very Large-Scale Integration Systems*, vol. 27, no. 8, pp. 1952-1956, Aug. 2019, doi: 10.1109/TVLSI.2019.2903980.

[22] S. Tambatkar, S. N. Menon, V. Sudarshan, M. Vinodhini, and N. S. Murty, “Error detection and correction in semiconductor memories using 3D parity check code with hamming code,” *2017 International Conference on Communication and Signal Processing*, Chennai, India, 2017, pp. 0974-0978, doi: 10.1109/ICCSP.2017.8286516.

[23] H. d. S. Castro, J. A. N. da Silveira, A. A. P. Coelho, F. G. A. e Silva, P. d. S. Magalhães, and O. A. de Lima, “A correction code for multiple cells upsets in memory devices for space applications,” *2016 14th IEEE International New Circuits and Systems Conference*, Vancouver, Canada, 2016, pp. 1-4, doi: 10.1109/NEWCAS.2016.7604783.

[24] F. Silva, J. Silveira, J. Silveira, C. Marcon, F. Vargas, and O. L. Jr, “An extensible code for correcting multiple cell upset in memory arrays,” *Journal of Electronic Testing*, vol. 34, pp. 417–433, 2018, doi: 10.1007/s10836-018-5738-5.

[25] L. J. Saiz-Adalid, P. Gil, J. C. Ruiz, J. Gracia-Morán, D. Gil-Tomás, and J. C. Baraza-Calvo, “Ultrafast error correction codes for double error detection/correction,” *2016 12th European Dependable Computing Conference*, Gothenburg, Sweden, 2016, pp. 108-119, doi: 10.1109/EDCC.2016.28.

## BIOGRAPHIES OF AUTHORS

**Jammula Shivani** is a student who completed B.Tech degree in Amrita Vishwa Vidyapeetham in the field of Electronics and Communication Engineering. Her key research areas are VLSI, IoT, and embedded systems. She can be contacted at email: shivanijammula2001@gmail.com.

**Mandadi Shankar Teja** is a student who completed B.Tech degree in Amrita Vishwa Vidyapeetham in the field of Electronics and Communication Engineering. His key research areas are VLSI, communication, and networking. He can be contacted et email: mandadishankarteja@gmail.com.

**Manickaraj Vinodhini** received the B.E degree in Electronics and Communication Engineering from Madras University, India, in 2000, and the M.Tech degree in VLSI Design from Amrita Viswa Vidyapeetham Deemed-to-be-University, India, in 2010. She completed her Ph.D degree with the Amrita Vishwa Vidyapeetham (University), Bengaluru Campus. She is currently an Assistant Professor at the Department of Electronics and Communication Engineering, Amrita Vishwa Vidyapeetham (University), Bengaluru Campus. She has 22 years of teaching experience. Her current research interests include the design of network-on-chip architecture and the design of error control coding techniques for reliable data transfers. She has 40 international journal/conference publications. She can be contacted at email: m_vinodhini@blr.amrita.edu.