A New Method to Enhance Memory Reliability

Priyabrota Sen[1], Muhammad Sheikh Sadi[2], Md. Manjur Ahmed[3] and Nabil Ashab[3]
[1] Department of Computer Science and Engineering,
Khulna University of Engineering & Technology, Khulna-9203, Bangladesh.
[2] Faculty of Computer Systems and Software Engineering,
Universiti Malaysia Pahang (UMP) 26300 Gambang, Kuantan, Pahang, Malaysia.
priyabrotasenpapon94@gmail.com, manjur@ump.edu.my

Abstract— **Multiple Cell Upsets (MCUs) are getting to be significant issues in the dependability of memory elements. To keep MCUs from impelling corruption of data, some Error Correction Codes (ECCs) are generally cast off to ensure memory reliability, yet the main issue here is existing error correction codes are time consuming. In this paper, a new method is initiated according to ensure memory more reliable compare to other existing technique together with lower time overhead. It enhances the capability of Decimal Matrix Code (DMC) to increase the error detection and correction capability. As long as error tolerance capability is in concern, many existing methods are outperformed by the proposed method which also reviles by experimental studies.**

Index Terms— **Decimal Matrix Code (DMC), Multiple Cells Upsets, Error Correction Codes, Encoder Reuse Technique, Memory Reliability.**

I. Introduction

As many innovations and development emerged and surprised us as the power of computation increased and for satisfying the ever-proliferating computation technologies like Nano and others are now integrated in disparate electronic circuits, the expeditiously extending memory cells soft error rate is all time high in present time [1]. Particularly the risk is becoming high when memories work in space situations because of ionizing impacts of radiation of neutron, alpha-molecule, and astronomical beams and other particles [2]. For memory reliability single bit upset is an already alarming aspect. The state of error occurrence becoming a matter of great concern in several memory reliable approach as Multiple Cell Upsets (MCUs) proliferates [3], [4].

Various memory reliability techniques have been initiated at different system layers to remove soft errors, most of them are particularly rely on full-scale redundancy [2]. Constructing memory units as convincible error-enduring and for reducing the rate of soft error occurred in memory cells by detecting and correcting it, various Error Correction Codes short for ECC are applied with the base systems [5], [6], [7], [8]. Among these coding techniques, Bose-Chaudhuri-Hocquenghem (BCH) codes [9], Reed–Solomon (RS) codes [10], and Punctured Difference Set (PDS) [11] codes etc. are more popular to protect memory against data faults and memory errors. But these has some consequences like the overly complexity of circuits. Because the redundant bits are increasing as the data size is increasing and thus more computation power is needed as more codes are required with the previously existing BCH, RS and PDS code. As a result, the encoding and decoding process require more time and the time overhead increases.

Reshuffle of memory cells are occurring because of the limitation of MCUs in interleaving systems. It means logical words are changing into physical words in physical arrangement of particular bits. Still, in Content-Addressable Memory (CAM), interleaving methods may not be used, due to the strong linkage of hardware structures from both equivalent circuit and memory cells arrangement [13], [14]. To correct single bit error and detect double bits error, Built-in Current Sensors (BICS) are invented to tolerate MCUs [13], [14]. It can accurate at most double errors in a word.

Decimal Matrix Code (DMC) [5] uses decimal calculation (decimal number expansion and decimal number deduction) to distinguish errors. The benefit of utilizing decimal calculation is that the error location ability is augmented so that the memory reliability is upgraded. However, if there are two or more errors in the same column then it is quite unable to detect that error. Hence, an enhancement to this technique is proposed in this paper to overcome this limitation.

The remaining part of the paper is arranged as follows. Section 2 discusses related topics of memory reliable techniques in brief. In Section 3 background of the proposed methodology is presented. Proposed methodology is discussed in section 4. Experimental analysis is shown in Section 5. Finally, Conclusions are drawn in Section 6.

II. Related Work

Nowadays, memory reliability is a crucial issue because of the ever-growing storage space in almost all kinds of systems possible. To control the flaws, there are necessities of error detection and the methodologies to correct the detected error. Several approaches are made to increase the reliability of memory such as parity bits, checksum, Cyclic Redundancy Checks (CRC), cryptographic hash functions etc. These methods are efficient in a small memory space but to maintain error in large scale, some methods are recently showing more efficiency like Matrix Code (MC), Decimal Matrix Code (DMC), and Punctured Difference Set (PDS) etc. These methods use less redundant bits and work faster than others. However, there is still a large gap towards obtaining desired reliability.

In the case of PDS systems, errors can be corrected only for four bits. Though it can correct four bits' error its correction coverage rate is not satisfied. MC detects up to nine bits' errors but the correction coverage of this approach decreases after 2 bits [13]. MCUs per word which minimizes decoding time overhead, where a single data word is partitioned into various lines and various segments in a coherent manner. Every segment is constituted with parity code confirms the column bit in Hamming code. To revise the two errors identified in Hamming code the enaction of syndrome bits positioned in vertical order is performed [13]. Consequently, MC is fit for amending just two mistakes in all cases. In a nutshell, almost all of these mentioned methods are more or less easy to implement but their correction coverage is not sufficient for large scale integration where large schemes of erroneous bits must handle.

Jing Guo et al. proposed a method to enhance the reliability of memory using Decimal Matrix Code (DMC) [5]. DMC is a newly invented system to detect errors in bit and correct the erroneous bits. The system is more efficient than the previously mentioned methods. It is considered that it is capable of correcting a data word as long as 16 bits. But, it has some downside too like it is not capable of detecting errors in bit when the erroneous bits are located in the same column. This type of mistake happens for some cases.

III. Background

Since the proposed method tries to remove the limitations of DMC and enhances its capability to tolerate faults, it is necessary to have a brief idea about the working principles of DMC in order to understand the proposed method. Hence, the working principle of DMC is outlined shortly as follows.

In DMC method, initially there are two executions occurred. First, the partition of memory cells in symbol. Second, the arrangement of those divided symbol in matrix order. Then, the data word is divided into some symbols contains some bits and the whole data is arranged into a matrix. Suppose there are N bit data word. It is then divided into symbols of S numbers and each and every symbol contains t number of bits. The organization of these symbols is defined by a matrix S. The matrix S is a 2-Dimentional matrix and the formation is $S = (S_1 \text{ x } S_2)$. The formation of the matrix means the varying of row number S_1 and column number S_2 can influence the correction capability and the number maximum number of bits it can correct. Then after the matrix formation the horizontal redundant bits H is created by treating the symbols as a decimal number and performing the decimal integer addition. Finally, by binary XOR operations the vertical redundant bits V are created. It ought to be noticed that both segregated symbols and arranged matrix are actualized in a logical manner rather than in a physical manner. Hence, altering the physical structure assembly of the memory is not a requirement in DMC.

To clarify the DMC method, we take a 32-bit word for instance. The memory cells are arranged from w_0 to w_{31} and the number of symbols here are eight and each and every symbol contains 4 bits. Simultaneously the value insertion of S_1= 2 and S_2 = 4 the organization of h_0 to h_{19} check bits in horizontal order and v_0-v_{15} check bits in vertical order occurs. The value of S and t influences the error correction capability and the number of maximum correction bits. Along these lines, S and t ought to be accurately changed in accordance with enhancing the adjustment ability and minimization of repetitive bits. The proper formation of matrix along with the proper value of S_1 and S_2 for maximum performance with minimum redundant bits for DMC are shown with example in previous paragraph.

Decimal addition is required for acquiring the horizontal bits the equation of which is shown below:

$$h_4h_3h_2h_1h_0 = w_3w_2w_1w_0 + w_{11}w_{10}w_9w_8 \tag{1}$$

$$h_9h_8h_7h_6h_5 = w_7w_6w_5w_4 + w_{15}w_{14}w_{13}w_{12} \tag{2}$$

Horizontal redundant bits $h_{14}\,h_{13}\,h_{12}\,h_{11}\,h_{10}$ and $h_{19}\,h_{18}\,h_{17}\,h_{16}\,h_{15}$ can be calculated in this same process. Vertical redundant bits are calculated by below equations:

$$v_0 = w_0 \oplus w_{16} \tag{3}$$

$$v_1 = w_1 \oplus w_{17} \tag{4}$$

Remaining vertical redundant bits are calculated by the same process.

Decimal and binary adding operation are used for the encoding process. The encoder that figures the repetitive bits utilizing two components. The first component is a multi-bit adder and XOR gates, here horizontal bits are h_0 to h_{19} and vertical bits are v_0 to v_{15}, and remaining bits are data bits which are u_0 to u_{31}. These are directly simulated starting w_0 to w_{31}.

Translating procedure is essential for obtaining the corrected word. As to begin with, w is accountable for the makeover of horizontal and vertical data bits h_4 to h_0 and v_0 to v_3. Additional, h_4 to h_0 and s_0 to s_3 which are horizontal data bits and vertical disorder bits are calculated using the equations below:

$$\Delta h_4h_3h_2h_1h_0 = h_4h_3h_2h_1h_0' - h_4h_3h_2h_1h_0 \tag{5}$$

$$s_0 = v_0' \oplus v_0 \tag{6}$$

Again, the remaining vertical bits are computed, the sign used here is subtraction sign which is "−" and it signifies whole number deduction which is called decimal subtraction.

If source data bits are obtained then it is stored in the code word and it means no error. Then the case is the value of the following $\Delta h_4h_3h_2h_1h_0$ and s_3 - s_0 bits are zero. Another matter can happen that there are prompted errors are detected and occurred in the position of symbol 0. The value of $\Delta h_4h_3h_2h_1h_0$ and s_3 - s_0 will be non-zero. Correction can be made by below equation:

$$w_{0correct} = w_0 \oplus s_0 \tag{7}$$

The same procedure can be adopted to correct the errors for other information bits as well. However, if there is an even number of errors in the same column then DMC fails to detect that error. Hence, there is scope to enhance the capability of DMC by removing this limitation. This paper tries to propose a new method in this direction. The proposed method is outlined shortly in the next section.

IV. The Proposed Methodology to Enhance Memory Reliability

The proposed method can detect and correct error using the principle of successive XOR and addition. This method processes erroneous bits by successively xoring the data bits and comparing the original redundancy bits with erroneous redundant bits.

This method is proposed to conform reliability to increase performance and less overhead ensure correction erroneous bits but it lacks in performance and in some cases, it not even can correct but also cannot detect a huge portion of the erroneous combination on same error prone data.

A. *Encoding Technique*

In this proposed method, first, the partition of symbol and arrangement of those symbols in matrix order are implemented, i.e., the data word of N bits is divided into S number of symbols of t bits ($N = S \times t$), and these symbols are organized in a 2-D matrix of $S_1 \times S_2$ to maximize correction capability. The maximum correction capability and the number of redundant bit are different when the different values for S and t are chosen. Next, the horizontally organized information bits r is generated which done by the XOR operation of each row for particular symbols. For the acknowledgement every symbol is considered as a 4-bit result of XOR operation. Third, the vertically organized redundant bits p is obtained in two steps. Among columns the bits are participate in the binary addition which is occurs by binary XOR operation between successive bits per symbol. Physical implementation is not occurred and in replace the logically implementation occurs when it comes to the techniques of both divide-symbols and arranged-matrix. Thus, any physical rearrangement of the memory cell is not required in this proposed method.

To explain the proposed method, consider a data word of 32 bits, which is divided into memory cells and syndrome bits. The information goes to the memory cells called w_0 -w_{31} bits. The data word of 32 bits produces eight symbols and every symbol contains 4 bits. Then $S_1 = 2$, $S_2 = 4$ is selected at the same time. The horizontally organized redundant bits and vertically organized redundant bits r_0–r_{15} and p_0 -p_{15} are generated. The different number of redundant bits influences the correction capability differently like it determines when correction capability is maximum and the chosen value of s and t also differs. Like this, the value of S and t also influences the redundant bits and correction capability. For different values of S and t the both value of redundant bits and correction capability may differ. This can be best described by this example. Here $t= 8$ and $S= 2\times2$ when we choose this the result is correction capability of 1 bit and number of horizontal and vertical bits combined is 40. But the correction capability is up to 3 bits and the number of redundant bits is decreased when we choose $S= 4 \times 4$ and $t= 2$. The maximum number of bitts can be corrected when the combination is $S= 2 \times 4$ and $t= 4$. But then redundant bits increased to 36 in the DMC method. But in the proposed method the redundant bit is reduced to 32-bit and it ensures more than 5 bits when it's come to correction capability and this is far better than other proposed systems like Hamming, MC, PDS and of course DMC.

r means horizontally organized redundant bits, can be attained as follows:

$$r_3r_2r_1r_0 = d_3d_2d_1d_0 \oplus d_7d_6d_5d_4 \tag{8}$$

$$r_7r_6r_5r_4 = d_{11}d_{10}d_9d_8 \oplus d_{15}d_{14}d_{13}d_{12} \tag{9}$$

And similarly, for the horizontal redundant bits $r_{11}r_{10}r_9r_8$ and $r_{15}r_{14}r_{13}r_{12}$, the representation here is decimal integer operation. Aimed at$P$ the vertical redundant bits and for decimal bits, following equations are required.

$$a_i = d_i \oplus d_{i+1} \tag{10}$$

$$p_3p_2p_1p_0 = a_2a_1a_0 + a_{14}a_{13}a_{12} \tag{11}$$

Now repeat for $p_7p_6p_5p_4$, $p_{11}p_{10}p_9p_8$, and $p_{15}p_{14}p_{13}p_{12}$. This method is also applied for vertical redundant bits.

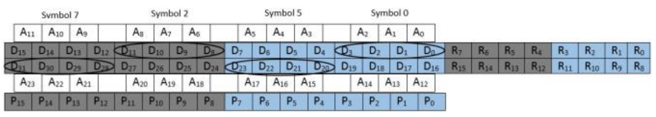

Symbol 7
Symbol 2
Symbol 5
Symbol 0
A11 A10 A9 A8 A7 A6 A5 A4 A3 A2 A1 A0
D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0 R7 R6 R5 R4 R3 R2 R1 R0
D31 D30 D29 D28 D27 D26 D25 D24 D23 D22 D21 D20 D19 D18 D17 D16 R15 R14 R13 R12 R11 R10 R9 R8
A23 A22 A21 A20 A19 A18 A17 A16 A15 A14 A13 A12
P15 P14 P13 P12 P11 P10 P9 P8 P7 P6 P5 P4 P3 P2 P1 P0

Figure 1: Logical organization of 32-bits ($S= 2 \times 4$ and $t= 4$).

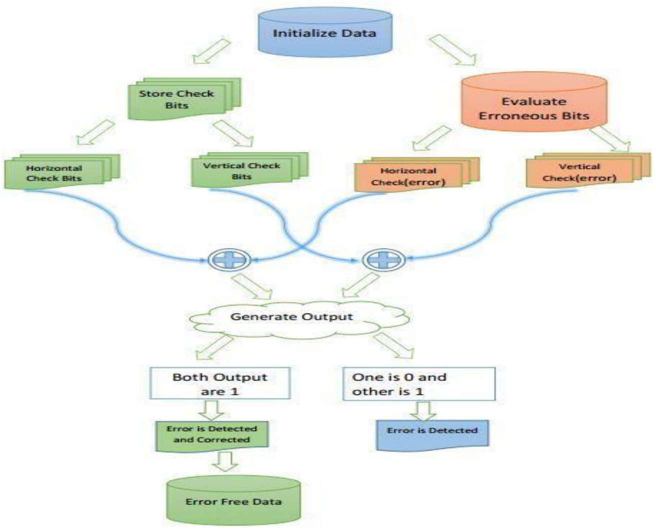


Figure 2: Flow Diagram of Encoding Technique

The encoding system is performed by XOR operation. The addition operation which is binary is also executed here. For redundant bits calculation, encoder is used. Figure 3 shows the whole operation. The multibit adders are assisted the encoding combined with XOR gates. The horizontally organized redundant bits are $r_{15} - r_0$ and vertically organized redundant bits are $p_{15} - p_0$. The remaining bits are information bits symbolized by d_{31} -d_0.

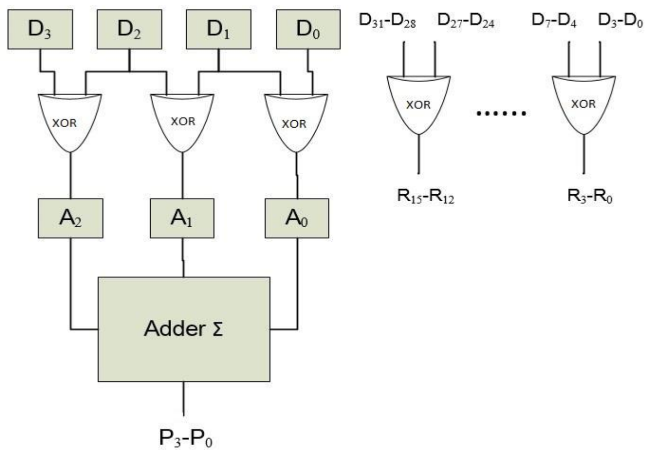


Figure 3: XOR gates and multi-bit adders are used in the32-bit encoder.

B. Decoding Technique

To get the correction of erroneous bit which was detected in the encoding process, there must be a viable process. In this case the decoding process is responsible for the job. First of all, *d* (the received information bits) generates both vertical and horizontal bits. This symbols $h_4h_3h_2h_1h_0$ and v_0–v_3 represents horizontal and vertical received bits. Next, syndrome bits horizontally organized $h_4h_3h_2h_1h_0$ and syndrome bits vertically organized $s_3 - s_0$ are obtained by the next equations:

$$\Delta r_3r_2r_1r_0 = r_3r_2r_1r_0' \oplus r_3r_2r_1r_0 \quad (12)$$

$$s_0 = r_0 \oplus r_0' \quad (13)$$

and like this, the remaining vertically organized syndrome bits, wherever "⊕" signifies the difference in symbols.

When $\Delta r_3r_2r_1r_0$ and $s_0 - s_3$ are equal to zero, the stored code word has no erroneous bits rather than contains original information bits in the position of symbol 0. When $\Delta r_3r_2r_1r_0$ and $s_0 - s_3$ provides non-zero value it means that symbol zero is the position of where the included error occurred and detected and in this particular case 4 is the error number, and for that case these errors can be corrected by

$$d_{0_{correct}} = d_0 \oplus s_0 \quad (14)$$

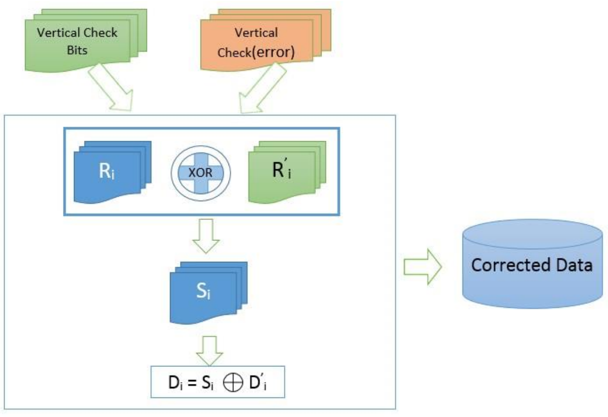


Figure 4: Flow Diagram of Decoding Technique

Figure 5 represents a 32-bit decoder structure. The whole structure is consisted with some component or sub modules. Each component of this module participates into different executions and undertaking particular procedures. The name of the particular executions are error pointer and error correcting device, a calculator calculates syndrome bits etc. The procedure of redundant data bit generation can be described from the figure. First d (the actual received data bit) which contains the actual information is needed. Then the creation of syndrome bits r and s executes. The procedure for creation of syndrome bits required the original set of redundant bits which is contrasted with d (the actual received data bits). Then the part occurs which contains the function of error locator and the function of this component is distinguish the bits where the error occurred. The operation is held by utilizing r and s and find the locate the error bits. At last, the part come where the error collector performs which means is revises the error. It does so by taking the error value and altering them.

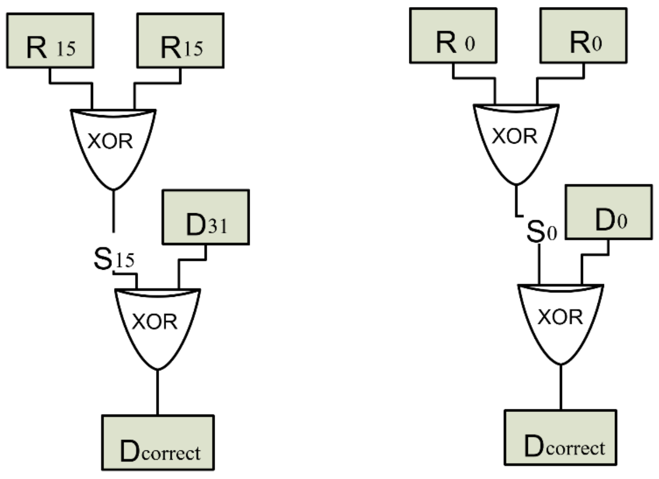
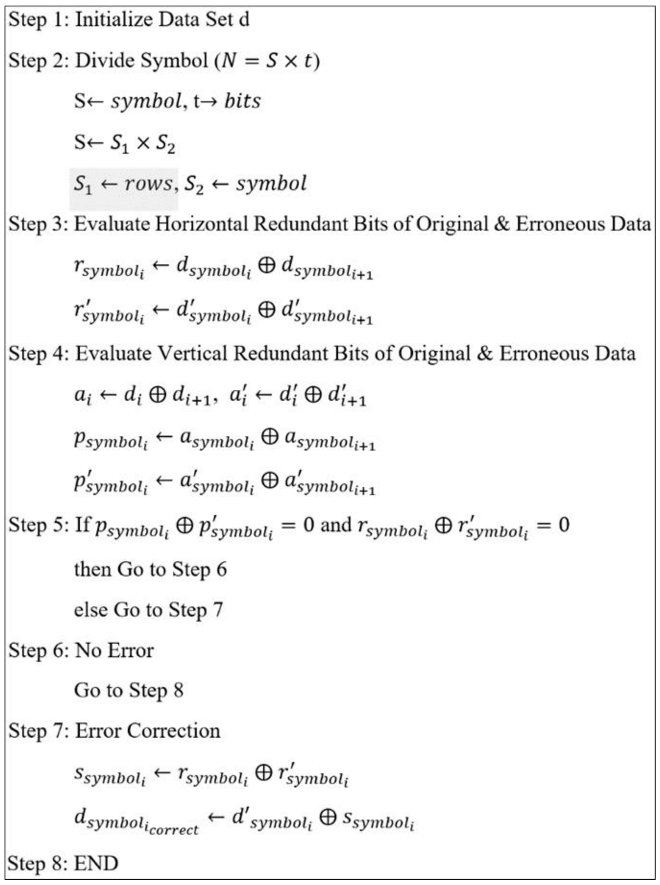


Figure 5: 32-bit decoder structure

C. Detection and Correction Algorithm

Error detection and correction algorithm for proposed method which was discussed before are below:



```
Step 1: Initialize Data Set d
Step 2: Divide Symbol ($N = S \times t$)
        S$\leftarrow$ $symbol$, t$\rightarrow$ $bits$
        S$\leftarrow$ $S_1 \times S_2$
        $S_1 \leftarrow rows$, $S_2 \leftarrow symbol$
Step 3: Evaluate Horizontal Redundant Bits of Original & Erroneous Data
        $r_{symbol_i} \leftarrow d_{symbol_i} \oplus d_{symbol_{i+1}}$
        $r'_{symbol_i} \leftarrow d'_{symbol_i} \oplus d'_{symbol_{i+1}}$
Step 4: Evaluate Vertical Redundant Bits of Original & Erroneous Data
        $a_i \leftarrow d_i \oplus d_{i+1},\ a'_i \leftarrow d'_i \oplus d'_{i+1}$
        $p_{symbol_i} \leftarrow a_{symbol_i} \oplus a_{symbol_{i+1}}$
        $p'_{symbol_i} \leftarrow a'_{symbol_i} \oplus a'_{symbol_{i+1}}$
Step 5: If $p_{symbol_i} \oplus p'_{symbol_i} = 0$ and $r_{symbol_i} \oplus r'_{symbol_i} = 0$
        then Go to Step 6
        else Go to Step 7
Step 6: No Error
        Go to Step 8
Step 7: Error Correction
        $s_{symbol_i} \leftarrow r_{symbol_i} \oplus r'_{symbol_i}$
        $d_{symbol_{i_{correct}}} \leftarrow d'_{symbol_i} \oplus s_{symbol_i}$
Step 8: END
```

Figure 6: Error detection and correction algorithm

In the following example symbol 0 and symbol, 3 has an error in the position of d_{14} and d_{30}. To detect and correct the symbol at first the horizontal redundant bit is calculated and then the vertical redundant bits are calculated. From the dissimilarity of vertical and horizontal redundant bits the particular row as well as column is determined where the error occurs. Last the horizontal redundant bits are used to correct the symbol by the following system.

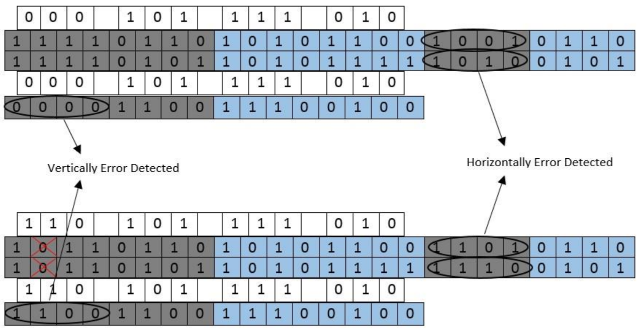


Figure 7: Error detection and correction using proposed methodology

$$r_7r_6r_5r_4 = d_{15}d_{14}d_{13}d_{12} \oplus d_{11}d_{10}d_9d_8$$

$$= 1111 \oplus 0110$$
$$= 1001.$$

$$r_7r'_6r_5r_4 = d_{15}d'_{14}d_{13}d_{12} \oplus d_{11}d_{10}d_9d_8$$

$$= 1011 \oplus 0110$$
$$= 1101.$$

Now d14 bit reverted to zero and it can detect error and it is in the first row of matrix code in symbol 2 and 3.

Here the correction sequence is below.

$$s_3s_2s_1s_0 = r_7r_6r_5r_4 \oplus r_7r'_6r_5r_4$$
$$= 1001 \oplus 1101$$
$$= 0100$$
$$d_{15}d_{14}d_{13}d_{12)\text{ correct}} = d_{15}d'_{14}d_{13}d_{12} \oplus s_3s_2s_1s_0$$
$$= 1011 \oplus 0100$$
$$= 1111$$

Now it is corrected and symbol 0 is fixed. In the similar way we can fix the symbol 3. .

V. Experimental Analysis

This section includes a description of experimental tools and discussion about the obtained results.

A. *Experimental Setup*

The subsequent components are used for the evaluation process of the proposed method.

- Processor: Intel(R) Core (TM) i3 CPU M370
- Installed Memory (RAM): 4 GB
- Hard Drive: 500 Gigabyte
- Operating System: Windows x64
- System Board: Intel Discrete HM57

B. *Experiment Results*

The goal of this segment is the proposed methodology for erroneous data correction as well as detection is compared with existing method such as Hamming, PDS, MC and DMC method with respect to correction coverage, redundant bits and the number of error detected bits.

By reviewing the Table-1 the observation result should be like that with proposed methodology correction for coverage is 100% up to five bits. Though correction coverage decreases gradually for greater than five bits' error combination and for 16 bits errors it is 11.8% where in DMC method the correction coverage differs from bit to bit like for two bits error it gives 95.16% correction coverage but from 3 bits error combination correction coverage decreases gradually for 16 bits' error DMC method gives 9.8%. Proposed Method (Table-1) shows high error correction performance compare to DMC, PDS, MC, and Hamming.

Table 1
Correction for Coverage

ECC	Data Word Error Percentage															
Systems	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Proposed Method	100	96.37	100	99.62	100	92.4	85.8	75.7	67.8	60.4	55.8	46.3	41.2	30.5	24.5	11.8
DMC (%)	100	95.16	99.87	99.59	99.97	91.6	84.7	74.3	66.7	58.4	54.5	45.7	40.0	29.6	22.3	9.8
PDS [9] (%)	100	100	100	0.8	0	0	0	0	0	0	0	0	0	0	0	0
MC [15] (%)	100	100	76.4	54.3	35.1	14.2	6.7	0.6	0	0	0	0	0	0	0	0
Hamming (%)	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

The following Figure 8 shows that proposed method is much better than MC method, but it is not that much better than DMC code. This proposed method is slightly better than DMC method. The curve of MC is sharp falling from bit 6 which turned to zero in bit 8 and remains zero for the rest of the bits. DMC method is much better than MC which is falling from bit 6 but creates a ramp along the path. Proposed method is better than these two methods which returns better result than both MC and DMC method

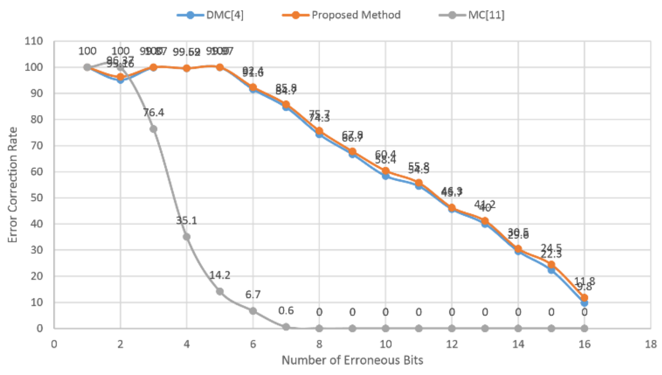


Figure 8: Error correction coverage of different methods

Following Figure 9 shows the compression of the overhead of different error correction code. Overhead of

DMC method is 113% where the proposed method overhead is much better than the DMC method it is 100% it means it requires 32 redundant bits for 32 bits erroneous data. The overhead of another method like as PDS, MC, hamming is better than the proposed method it is 60% for PDS method, 88% for MC method and only 22% for Hamming code. Though their overhead is better than the proposed method the detection and correction rate is not better than the proposed method.

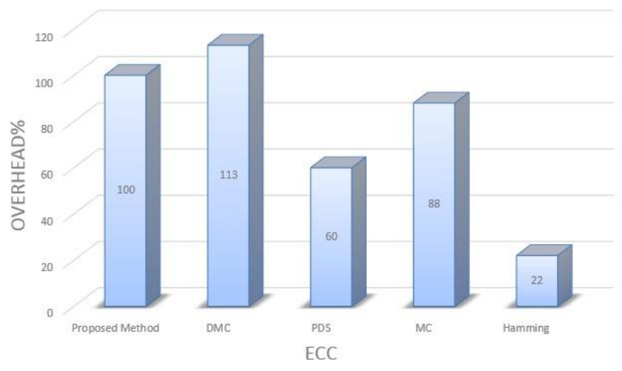


Figure 9: Error correction coverage of different methods

From the situation we can say that the proposed technique needs less redundant bits then DMC method. But it required more redundant data bits then the other proposed methods like MC, Hamming and PDS but there is no need to say that these methods are better than proposed one. Because the analysis of area overhead of memory cell. It is connected with β which represents coding efficiency. Table 2 represents the redundant bits of all these codes which will clarify the consequences.

Table 2
Redundant Bits

ECC	Information Bits	Redundant Bits	β
Proposed Method	32	32	50%
DMC	32	36	52.9%
PDS	32	19	37.3%
MC	32	28	46.7%
Hamming	32	7	17.9%

$$\boldsymbol{\beta} = \frac{Redundant\ bits}{Information\ bits + Redundant\ bits}$$

As the equation tells the value of the β can be obtained by calculating two components redundant bits and information bits. The value of β is very important because it reviles a equally important bit of information called memory cell overhead. The value of β is directly proportional to the value of memory cell overhead. That means if the value of β is lower the memory cell overhead will also be lower. Now, we can discuss various correction technique and the limitation compared to the proposed scheme. First of all, the table shows that Hamming code and MC [12] has the ability to retain a persistent correction capability. Though the value of β is lower in Hamming code but it serves noting. For Matrix Code the reason for constant correction capability is Hamming code. Hamming code has an issue of limited error correction so, which is why the MC gives us the poor result. PDS has a less value of β than the proposed scheme which is a good thing. But the access time of memory is severely affected due to higher delay overhead. The development of CMOS technology helped us to observe and jot down the result. Because the number of MCUs augmented intensely. The rate of observation is more than three errors and that occurred in 90-nm technology in radiation test [11], [12]. The β value is higher in proposed scheme but this is the best choice for protecting memory because the capability of correction is maximum than other existing technique like Hamming, MC, and PDS because they are not accurate.

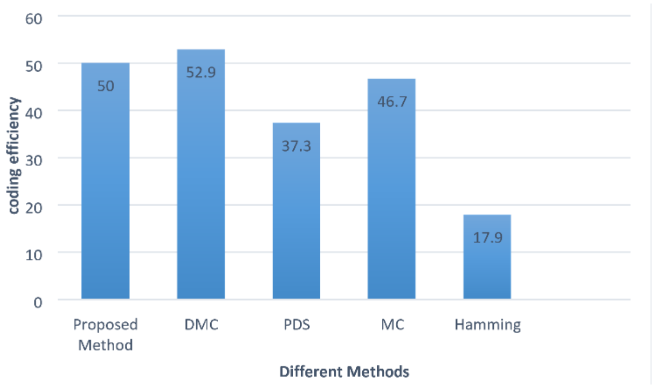


Figure 10: Coding efficiency graph

From Figure 11 it can be clearly perceived that the failure of error detection rate goes down for the even number of erroneous bit combinations of DMC method. But it works more properly and it can detect erroneous bits for the odd number of error bits' combinations. But the proposed method can detect up to 100% of erroneous data for an odd number of error bits' combinations.

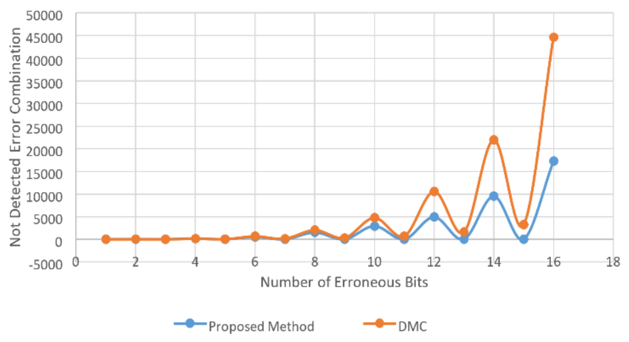


Figure 11: Failure of detection of erroneous bits

To check the number of erroneous bits in a code word, different methods take different times. For the low number of erroneous bits the difference of run time of DMC and the proposed method is negligence but when the bit size increase the run time differs much. For 32 bits error detection and correction, it takes 0.0132 seconds to run the DMC method algorithm and the Proposed method requires 0.01501 seconds to run the algorithm. The following Figure 12 shows the comparison of run time of two methods for different types of error bit size.

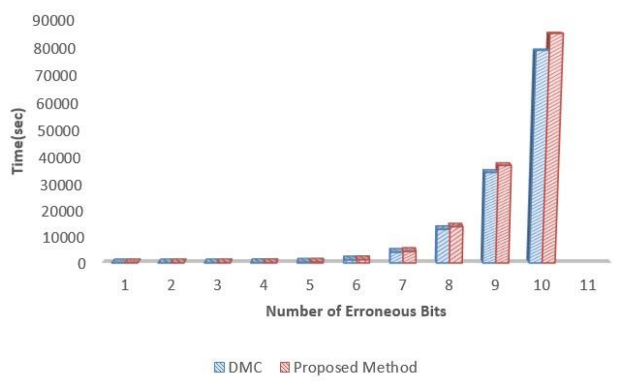


Figure 12: Run time comparison for different bits

The comparison between the correction rate of DMC and that of the proposed method is shown in the following Figure13. The proposed method shows more correction rate than DMC. It gives the correction coverage up to 16 bits of error. For 2 bits and 4 bits' error, the proposed method shows 100% correction coverage. The correction coverage rate decreased with the increase of erroneous bits

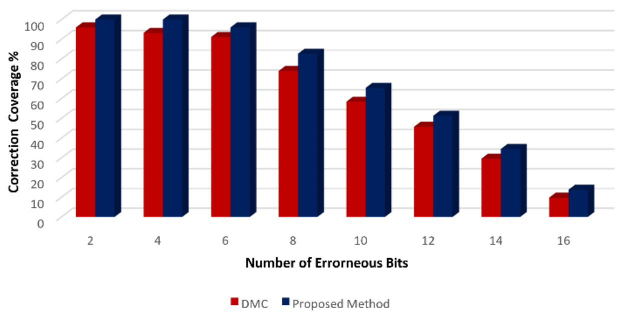


Figure 13: The Comparison between the Correction Rate of DMC and That of this modified Method

VI. CONCLUSION

In this paper, to increase the memory capability and reliability, an enhanced version of DMC is projected. The attained consequences show that the projected method has more tolerance level against huge MCUs in memory. Although the projected system requires more redundant bits than other methods to maintain the higher reliability of memory, the reliability is increased to a great extent. For safety critical systems, or in electronic hostile environments, or the systems where reliability is a matter of great concern, the proposed method shows better effectiveness with respect to reliability. There are scopes to enhance this research in terms of reducing information overhead without sacrificing other nonfunctional properties.

REFERENCES

[1] S. Liu, J. Li, P. Reviriego, M. Ottavi and L. Xiao, "A Double Error Correction Code for 32-Bit Data Words With Efficent Decoding," in IEEE Transactions on Device and Materials Reliability, vol. 18, no. 1, pp. 125-127, March 2018.

[2] Chunhua Qi, Liyi Xiao,MingxueHuo,Tianqi Wang,Rongsheng Zhang,Xuebing Cao, "A 13T radiation-hardened memory cell for low-voltage operation and ultra-low power space applications," 18th International Symposium on Quality Electronic Design (ISQED), pp. 161-165, 2017.

[3] Jinghui Han, Yao Zhou, Hao Ni, Xiao Zheng, Yi Zhao, "A 55nm logic process compatible p-flash memory array fully demonstrated with high reliability,"IEEE 12th International Conference on ASIC (ASICON), pp.419-432, 2017.

[4] Md. Shamimur Rahman, Muhammad Sheikh Sadi, SakibAhammed, "Soft Error Tolerance using Horizontal-VerticalDouble-Bit Diagonal Parity Method," 2nd Int'l Conf. on Electrical Engineering and Information & Communication Technology (ICEEICT), pp.1-6, May 2015.

[5] Jing Guo,Liyi Xiao,Zhigang Mao,Qiang Zhao, "Enhanced Memory Reliability Against Multiple Cell Upsets Using Decimal Matrix Code," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, pp.127-135, 2014.

[6] Valentin Gherman, Samuel Evain, Nathaniel Seymour, Yannick Bonhomme, "Generalized Parity-Check Matrices for SEC-DED Codes with Fixed Parity," On-Line Testing Symposium (IOLTS), 2011 IEEE 17th International,July 2011.

[7] Alexander Stempkovskiy, Dmitry Telpukhov, Sergei Gurov, Tatiana Zhukova, Alena Demeneva, "Rcode for concurrent error detection and correction in the logic circuits,"IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), pp.1430-1433, 2018.

[8] N Lohith Kumar, J. Jenitta, "Amended fast fourier transform algorithm implementation for error correction codes in OFDM,"International Conference on Intelligent Computing and Control (I2C2), pp. 1–5, 2017.

[9] Arul K. Subbiah, Tokunbo Ogunfunmi, "Area-effcient re-encoding scheme for NAND Flash Memory with multimode BCH Error correction," IEEE International Symposium on Circuits and Systems (ISCAS), pp. 1–5, 2018.

[10] Yuval Genga,OlutayoOyerinde, Jaco Versfeld, "Improving the decoding performance of the PTA algorithm for RS codes through the extension of the parity check matrix,"IEEE AFRICON, pp. 171– 174, 2017.

[11] Nithyendra Chandan,Narottam Das,FarzanehFadakarMasouleh, "Analysis of plasmonics-based nanostructured MSM-PDs for enhanced light absorption,"Eleventh International Conference on Sensing Technology (ICST), pp. 1-5,2017.

[12] Bilal Akin, Manish Bhardwaj, Shamim Choudhury, "An Integrated Implementation of Two-Phase Interleaved PFC and Dual Motor Drive Using Single MCU With CLA,"IEEE Transactions on Industrial Informatics, pp.2082-2091,2013.

[13] Sid Zarabi, Egon Fernandes, Isabel Rua,ArmaghanSalehian, Helene Debéda; David Nairn, Lan Wei, "Design and development of a self-contained and non-invasive integrated system for electricity monitoring applications,"IEEE 12th International Conference on ASIC (ASICON), pp. 1125-1128, 2017.

[14] PuppalaRajendhar, P. Pavan Kumar, R. Venkatesh, "Zigbee based wireless system for remote supervision and control of a substation,"International Conference on Innovative Research InElectrical Sciences (IICIRES), pp.1-4, 2017.