Journal of Electronic Testing (2024) 40:525–537
https://doi.org/10.1007/s10836-024-06135-5

RESEARCH

Interleaved Counter Matrix Code in SRAM Memories for Continuous Adjacent Multiple Bit Upset Correction

Ahilan A[1] · Anusha Gorantla[2] · Gladys Kiruba[3] · Asmaa A. Hamad[4] · Mohamed M. Hassan[4] · Venkatram N.[5] · Sindhu T. V.[6]

Received: 22 January 2024 / Accepted: 20 August 2024 / Published online: 27 September 2024

Abstract
SRAM memory systems are suffering from an increase in data due to the aggressive CMOS integration density. The frequency of Multiple Cell Upsets (MCUs) on SRAM memory is increasing, which is resulting in the increasing use of ECCs. Speed is slowed down by ECCs due to their overhead, both in memory bits and decoding times. In this research, Continuous Adjacent Multiple Bit Upset Correction (CAMBUC)has been proposed to greatly reduce the redundancy and improve correction coverage.In addition, the Interleaved Counter Matrix Code (ICMC) is proposed to simplify the Encoder and decoder circuits which reduces the delay. The proposed method predicts MBU before decoding and maximum parallel 8 error bits are corrected using the proposed method. Using this method, the error correction code's parity check matrix is automatically and effectively constructed, simply stating its error detection and/or correction capabilities. Combinatorial counting operations introduce error detection and error prediction. The proposed ICMC makes use of the decimal and hamming algorithms to achieve the highest level of error detection. In comparison to the conventional decoder, which can rectify double and 8-adjacent errors, the assessment results utilizing the proposed CAMBUC approach demonstrate significant reductions in area, power, and delay. The obtained findings demonstrate that the proposed method has a comparatively long mean time to failure (MTTF) when compared to existing approaches. At the same time, the suggested scheme's delay overhead is, respectively, 20.5%, 14.6%, 11.2%, 8.5%, and 3.5% less than that of the existing Hamming, CMC, ECC for 5-bit adjacent error, eMRSC, and DICE-7-bit ECCmethods.

Keywords Error Correction Codes · SRAM Memory · Multiple bit Upsets · Matrix

1 Introduction

Reliability is an essential requirement in space applications. Memory devices are repositories that can either permanently (like ROM) or temporarily store data (like RAM) [1]. Computers employ semiconductor memories for storage because of their low power consumption, inexpensive cost, andhigh capacity. A single bit of data can be stored in each storage component of the memory. As a result, it is crucial to consider how radiation-sensitive memory is to guarantee the dependability of electronic systems. Most electronic systems

Responsible Editor: N. A. Touba

✉ Sindhu T. V.
sindhutv1983@gmail.com

Ahilan A
listentoahil@gmail.com

[1] Department of Electronics and Communication Engineering, PSN College of Engineering and Technology, Tirunelveli, Tamil Nadu, India

[2] Department of Electronics and Communication Engineering, Raghu Engineering College, Visakhapatnam, Andhra Pradesh, India

[3] Bangalore College of Engineering and Technology, Chandapura, Bangaluru, India

[4] Department of Biology, College of Science, Taif University, P.O. Box 11099, Taif 21944, Saudi Arabia

[5] Department of Electronics and Communication Engineering, Koneru Lakshmaiah Education Foundation, Green Fileds, Vaddeswaram 522302, Andhra Pradesh, India

[6] Departmenmt of Electronics and Communication Engineering, IES College of Engineering, Thrissur, Kerala, India

use static random-access memories (SRAMs) as a reliable component [2].

Due to their rapid random-access speeds and compatibility with the CMOS logic process, static random-access memories (SRAM) have been employed extensively for many years [3]. With CMOS technology, the storage capacity of the memory system rises as the distance between memory cells narrows. Thus, the multiple bits upset (MBU) and single event upset (SEU), two well-known single event effects, are two radiation-induced soft mistakes [4]. Multiple cell upsets (MCU) have also grown to be a significant issue in the memory design process, even if SEUs are a big concern in space and terrestrial applications. The likelihood of having many errors grows as a result of the memory error rate in memories increasing as a result of the ongoing technology shrinkage [5].

Frequently used to ensure that data in circuits and memory is free from soft errors are error correction codes, or ECCs. A decoder finds and fixes any faults while reading from the memory, and an encoder computes those bits when writing to the memory. Any one of the following can affect a codeword bit: two double adjacent bit error patterns, one single bit error pattern, and three distinct triple adjacent bit error patterns. These cases of decoded bit error patterns are gated by the control signals of DAEC and TAEC, as observed in the selection circuit. As a result, memory access time is directly impacted by the encoding and decoding latency [6]. SRAMs have often addressed MCUs using a straightforward single-error-correcting-double-error-detecting (SEC-DED) ECC [7]. The memory cells impacted by the MCU are physically close together and frequently neighboring. This is because mistakes are made along the particle's route. Therefore, even when an SEC-DED code is utilized, MCUs can produce numerous mistakes in a simple sentence, failing. Although more potent ECCs like Bose-Chaudhuri-Hocquenghem (BCH)and multi-layered codes can directly counteract MBUs, they have larger space and latency costs as compared to the more straightforward SEC-DED codes. However, because the encoding and decoding circuitry in these sophisticated codes are more intricate, these codes have higher delay, area, and power overheads [8]. This can be avoided by utilizing interleaving, which physically spaces out the bits of a word such that an MCU can only change one bit per word [9].

MCUs will produce neighboring mistakes in the absence of interleaving. In such a situation, SEC codes can lead to an incorrect correction when two adjacent bits are incorrect, whereas SEC-DED codes can lead to an incorrect repair when three adjacent bits are incorrect [10, 11]. In both situations, this could result in Silent Data Corruption (SDC), in which the system carries on operating despite being ignorant of an error. As a result, the system may behave improperly, and data may become even more damaged. Particularly in applications involving space, these overheads must be kept to a minimum. Cosmic-ray-induced soft faults provide significant hurdles for space application. Failure of the electronic system might result from soft errors that alter the memories' storage state. Error correction codes (ECCs), a generic mitigation method used at the system level to remove the influence of soft errors, are employed to prevent system corruption. MBU emerges as the primary error pattern as feature sizes decrease due to the integration of more memory cells in the energy deposited range of radiation particles.To mitigate the effects of ambient radiation, error correcting codes (ECCs) are commonly used in contemporary nanoscale devices with storage capacity. Because of this, many cutting-edge ECC algorithms work to prevent data corruption when a single logical memory word experiences varying quantities of mistakes, or bitflips. Multiple bit upsets (MBUs) or equivalent can still occur when two distinct events hit neighboring cells in the same word, even when bit interleaving prevents a single particle (such as a proton or neutron) from flipping several cells in the same word.

In this research, Interleaved Counter Matrix Code (ICMC) has been proposed for Continuous Adjacent Multiple Bit Upset Correction(CAMBUC)which greatly reduces the redundancy and improves correction coverage. In addition, the proposed method predicts MBU before decoding, and maximum parallel 8 error bits are corrected using the proposed method.Using this method, the error correction code's parity check matrix is automatically and effectively constructed, simply stating its error detection and/or correction capabilities. Combinatorial counting operations introduce error detection and error prediction. The proposed ICMC makes use of the decimal and hamming algorithms to achieve the highest level of error detection. The outcomes demonstrate that the suggested codes greatly minimize encoder and decoder latency.One of the primary contributions of the proposed method is;

- The proposed method predicts the errorbefore detecting and correcting the errors in the 32-bit code.
- The proposed strategy has low redundancy, and ICMC codes provide quick encoding and decoding operations with reasonable power and area overhead.
- The proposed method significantly performs parallel 8-bit error correction that makes the proposed method achievemaximum correction coverage.
- Moreover, the Interleaved Counter Matrix Code (ICMC) is proposed to make the Encoder and decoder circuits simpler which reduces the delay.

The following is a list of paper highlights: The second section explains a summary of the relevant work. The proposed set of codes is presented in the third section along with examples. The various outcomes of the ECC evaluation are discussed in the fourth section. The fifth section brings the paper to a conclusion.

2 Literature Survey

In recent days, several error correction codes have been designed by researchers, mainly for reducing the redundancy of the code. This limitation prompted researchers to develop a set of solutions to improve the performance of ECCs. Some of those methods are illustrated briefly in this section.

In 2019, Saiz-Adalid, L.J., et al. [12] proposed a series of error control codes called Ultrafast codes for extremely fast-decoding and encoding operations. These codes are particularly well suited to situations where the encoder and decoder circuits must operate with extremely low latency. The error coverage has been verified using several implemented and simulated instances of ultrafast codes.

In 2021, Prasad, S.V.S., et al. [13] enhanced three-bit burst error-correction coding by including quadruple adjacent error correction (QAEC). Since QAEC's H matrices are used to display the acquired three-bit error correction, no additional parity check bits are needed. The findings indicate that the rules require an appropriate amount of time and space to complete the required extension of corrective capability. The 45-nm library was used to develop encoders and decoders.

In 2016, Appathurai, A. and Deepa, P. [14] introduced counter matrix code (CMC), which makes use of combinational one'sparity and counter generator with minimal redundant memory overhead. The specific amount of upsets is forecasted by CMC based on the error predictor, even before the actual mistake detection and correction procedure starts. With better corrective coverage, the suggested technique has a lower MTTR than I3D, DMC, and MC by 0.3, 0.2, and 1.8 milliseconds, respectively.

In 2015, Reviriego, P., et al. [15] introduced a (24,12) extended Golay code's single and double-adjacent error-correcting parallel decoder. To implement it effectively, the decoder makes use of the code's characteristics. The results demonstrate that notable savings in space, latency, and electrical power consumption can be accomplished in comparison to the traditional SEC-DAEC decoder.

Li, J., et al. [16] introduced a 3-bit quadruple adjacent error correction burst error correction algorithm in 2017. (QAEC). For 16 data bits, the proposed method has found every possible solution, while for 32 and 64 data bits, an optimization algorithm has been used to find all possible solutions. The 4-bit burst error patterns are not fixed by this method, only the quadruple adjacent error.

In 2018, Gracia-Moran, J., et al. [17] developed a set of ECCs that, while preserving or even improving memory error coverage, greatly reduce the redundancy introduced. There is also a reduction in the overhead for latency, space, and power. These new codes were created utilizing the adaptable unequal error control (FUEC). These codes are now a viable solution for crucial embedded system applications. Because of their poor redundancy, these codes perform noticeably worse than 4-bit burst errors.

In 2013, Guo, J., et al. [18] proposed per-word DMC to guarantee the accuracy of memory. The introduced code used a decimal approach to detect flaws, allowing for a greater number of problems to be found and fixed. The suggested DMC's only shortcoming is that more redundant bits are needed to achieve improved memory dependability.

In 2014, Saiz-Adalid, L.J., et al. [19] proposed 3-bit burst error correcting codes and SEC-DAEC-TAEC to safeguard SRAM memory toward MCUs. This paper presents novel algorithms for resolving 3-bit burst and triple adjacent faults. In contrast to the SEC-DAEC codes already in use, the synthesis findings demonstrate that the recommended codes can be applied successfully and only add a minor amount of complexity.

In 2021, Bhargavi, C., et al. [20] suggested codes for memories. The power, area, and latency of each of the three distinct H-Matrices utilized for the decoding and encoding operations are computed. The suggestion to modify the H-Matrix-based codes to facilitate fast decoding may be relevant. It is possible to further enhance mistake correction by lowering the parity check bit count.

Problem Statement:

Many approaches that have been proposed recently for improving the design of ECC for memory applications could be used to enhance burst error correction codes [21, 22]. Triple adjacent error correction (TAEC), and double adjacent error correction (DAEC) are the main areas of focus for adjacent bits correcting technology. These error-correcting codes feature more redundant bits, but because of their poor redundancy and delay, their performance is significantly reduced.

For instance, hamming codes [23] have the lowest redundancy for correcting single-bit errors. Hamming codes and parity checks in a matrix are used in matrix coding to allow for the correction of two erroneous bits. Column-line-code (CLC), a more contemporary technique, corrects up to two consecutive erroneous bits using extended Hamming codes and parity bits. The parity-check matrix for the Hamming (7, 4) and syndrome decoding methods is shown in Tables 1 and 2.

It is possible to extend Hamming codes to fix single faults and find double random errors. Additionally, there are the SEC-DED extended Hamming codes (13, 8), (22, 16), (39, 32), and (72, 64). Redundancy declines with increasing data word lengths, just like in the SEC codes. In this study, Continuous Adjacent Multiple Bit Upset Correction (CAMBU), which considerably lowers redundancy, has been developed. It uses Interleaved Counter Matrix

Table 1 Hamming (7, 4) code based Encoding formula

D_0	D_1	D_2 P_0	D_3	D_4 P_1	D_5 @@	D_6 @@	Encoding formulas
0	1	0	0	1	1	1	$D_0 = P_1 \oplus P_2 \oplus P_3$
1	0	0	1	1	1	0	$D_1 = P_1 \oplus P_2$
0	0	1	0	0	1	1	$D_3 = P_0 \oplus P_2 \oplus P_3$

Table 2 Hamming (7, 4) code for syndrome bits

U_0	U_1	U_2 @@	U_3	U_4 @@	U_5 @@	U_6 P_3	Syndrome bits
0	1	0	0	1	1	1	$S_0 = U_1 \oplus U_4 \oplus P_2 \oplus P_3$
1	0	0	1	1	1	0	$S_1 = P_1 \oplus P_2$
0	0	1	0	0	1	1	$S_2 = P_0 \oplus P_2 \oplus P_3$

Code (ICMC). Additionally, the suggested technique corrects up to 8 erroneous bits in parallel and anticipates MBU prior to decoding. The "Proposed ICMC Methodology" section details the proposed approach.

3 Proposed ICMC Methodology

In this research, Continuous Adjacent Multiple Bit Upset Correction (CAMBUC) has been proposed to reduce the redundancy. In the proposed CAMBUC, a group of 32-bit data are taken as input to the encoder and decoder, for achieving lower redundant bits. To achieve the highest error detection capabilities, the proposed method makes use of a new Interleaved counter matrix code (ICMC) decimal algorithm. The proposed ICMC makes use of the decimal and hamming algorithms to achieve the highest level of error detection. In addition, the proposed method predicts MBU before decoding, and maximum parallel 8 error bits are corrected using the proposed method. Using this method, the error correction code's parity check matrix is automatically and effectively constructed, simply stating its error detection and/or correction capabilities. In error detection and error prediction, combinatorial counting procedures are introduced. To complete a particular task, the suggested technique makes use of an error locator, error corrector, and syndrome calculator. It can be observed that obtaining the syndrome bits requires recalculating the redundant bits from the received data bits D and comparing the results to the initial redundant bit set. The error finder then uses H and S to determine and localize the exact bits that are exhibiting problems. The error bits' values can finally be reversed in the error corrector to rectify these problems. Figure 1 shows the proposed method's overall flow.

3.1 Interleaved Counter Matric code (ICMC) Encoder

The quantity of redundant bits needed directly relates to the cost of the ECC approach. In the proposed ICMC, 32-bit data is taken as input to the encoder and ICMC itself is part of the decoderfor achieving lower redundant bits. The symbols are inserted into a k1 × k2 matrices (k = k1 × k2), where the values of k1 and k2 indicate the two rows after the D-bit data and P parity bits are separated into k symbols of m bits (D = k × m). D0 through D31 are the cells that contain bits of information. Two 16-bit symbols make up this 32-bit word. Symbol 1 included the data bit $D_0 - D_{15}$ and parity bits of $P_0 - P_8$ moreover, Symbol 1 included the data bit $X_{16} - X_{31}$ and parity bits of $P_9 - P_{17}$. Here, 32-bit data words with two symbols were employed in the proposed ICMC approach. The maximum correction capability (i.e., the largest MCU size that can be corrected) and the number of redundant bits, however, change according to the values used for the data and symbols. Consequently, it is important to carefully modify m and k to maximize correction capability and reduce redundant bit count.If a group contains more words, fewer redundant bits are needed; this is demonstrated by the required amount of parity bits for the group length. In H (50, 32), for instance, the redundant bit computation yields 50, while the 32-bit computation yields 48, a reasonably low redundancy when compared to current methods. To enhance error identification and correction, the combinational one's counter operation of the proposed ICMC is applied to data bits to forecast and minimize the number of redundant bits.

To generate the codes, several hamming coding rules and decimal integers were used. As an example of the proposed approach, 32-bit data is employed; in Fig. 2, P stands for parity bits and D for data bits. This matrix code behaves in the following fundamental ways. The main data input (D_i) is split up into multiple-bit groupings. Because it can detect and repair all continuous 8-bit burst mistakes as well as all

single errors, the Matrix code used in this work outperforms previous methods in correction and detection.

A 32-bit data word requires eighteen code bits in the proposed CAMBUC. For this code, "P" stands for a parity bit and "D" for data bits in the parity-check matrix H. After H has been determined, designing the encoder/decoder circuitry is rather simple. When related data bits that are ones in the first row of the H-Matrix are XORed, the first redundant bit for the first 32 bits of data, from D0 to D31, can be found. To complete the remaining six superfluous bits, repeat this method. Applying the encode and decode procedures on a 32-bit data word with the CAMBUC (46, 32) code introduces an 8 adjacent-bit error concurrently with the data.

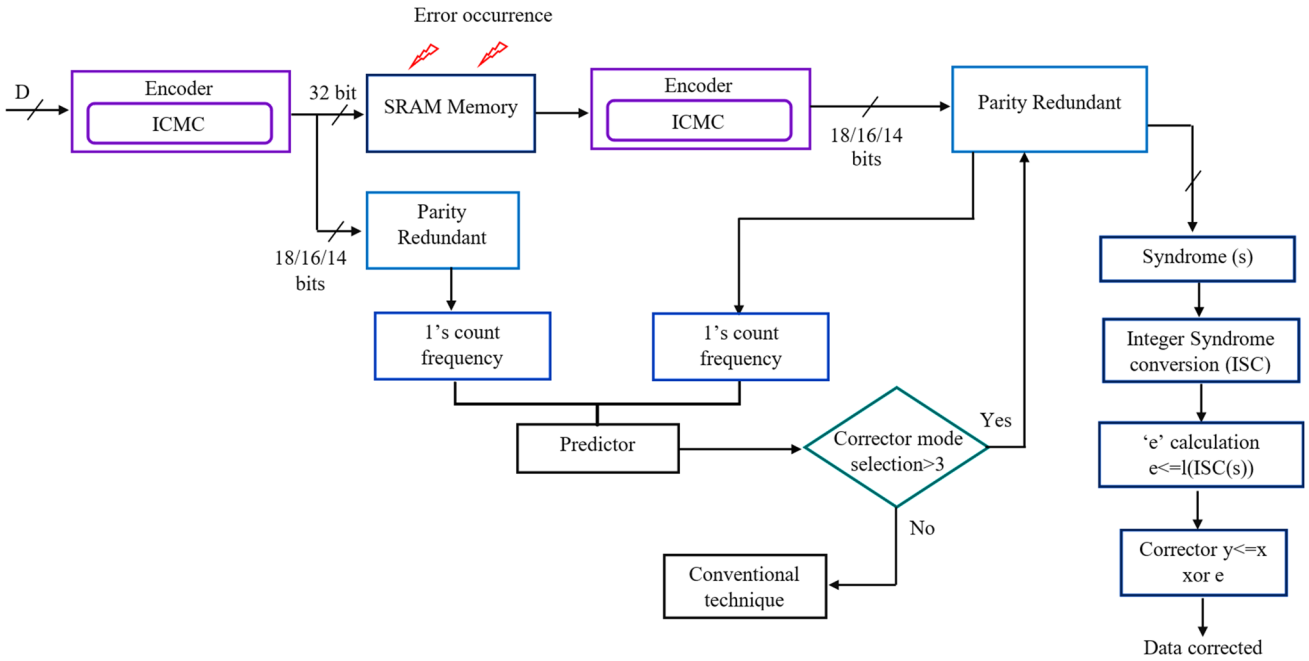


Fig. 1 Proposed CAMBUC methodology

P_0	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}	D_{11}	D_{12}	D_{13}	D_{14}	D_{15}	$Symbol_1$
P_9	P_{10}	P_{11}	P_{12}	P_{13}	P_{14}	P_{15}	P_{16}	P_{17}	X_{16}	X_{17}	X_{18}	X_{19}	X_{20}	X_{21}	X_{22}	X_{23}	X_{24}	X_{25}	X_{26}	X_{27}	X_{28}	X_{29}	X_{30}	X_{31}	$Symbol_2$

Fig. 2 Layout of a 32-bit data word for (32, 18)

$$H(50,32) =$$

$$\begin{array}{l}
\quad \overbrace{(P_{17},P_{16},\ldots\ldots\ldots P_9}^{Parity_2}\overbrace{(P_8,P_7,\ldots\ldots\ldots P_0}^{Parity_1}\overbrace{(D_{31},D_{30},\ldots\ldots\ldots\ldots\ldots\ldots D_{16}}^{Symbol_2}\overbrace{(D_{15},D_{14},\ldots\ldots\ldots\ldots\ldots\ldots D_0}^{Symbol_1}) \\
\left[\begin{array}{cccccccccccccccccccccccccccccccccccc}
1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0&1&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0 \\
0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&1&0&0&0&1&0&0&1&0&0&0&1&0&0&0&1&1&0&0&0&1&0&0&1&0&1&0 \\
0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&0&1&0&0&1&0&0&0&1&0&0&1&0&0&0&1&0&1&0&0&1&0&0&0&1 \\
0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&1&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&1&0&0&0 \\
0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&1&0&1&1&0&0&0&0&1&0&0&0&0&1&0&0&1&0&0&1&0&1 \\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&1&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&1&0&0&1&0 \\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&1&1&0&0&0&1&0&0&1&0&0&0&0&1&0&0&1&1&0&0 \\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&0&1&0&0&0&1&0&0&0&0&1&0&0&0&1&0&1&0&0&0&1 \\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&1&0&0&0&1&0&0&1&1&0&0&0&1&0&0&1&0&0&0&0&1&0&0&1&1&0&0
\end{array}\right]
\end{array}$$

Fig. 3 Parity-check matrix H for the (50, 32) ICMC code

The parity-check matrix H utilizing ICMC for the (50, 32) is displayed in Fig. 3. A 32-bit data word's error detection and correction are handled by this matrix. The matrix employs 18 parity bits (50 − 32) to guarantee the integrity of the data because it comprises 50 rows and 32 columns. The matrix H is designed to rectify single-bit, two-bit adjacent, and three-bit burst faults, as well as to detect and correct up to eight-bit burst errors. The parity-check matrix H for the (48, 32) ICMC technique is displayed in Fig. 4. With 48 rows and 32 columns, it makes use of 16 parity bits (48 − 32). The parity-check matrix H for the (46, 32) ICMC technique is displayed in Fig. 5. This matrix has 46 rows and 32 columns and uses 14 parity bits (46 − 32). Figures 3, 4 and 5 display the various configurations of the parity-check matrix for the ICMC code. While considering redundancy and performance, each is adjusted for a different level of error detection and repair.

Let's examine two sets of data $D(D_0, D_1 \ldots D_{15})$ for symbol 1 and $(D_{16}, D_2 \ldots D_{31})$ for symbol 2—to better understand how encoding and decoding operate. After H has been established, designing the encoder/decoder circuitry is rather simple. The $\oplus$ symbol stands for the XOR operation, which combines the data bits to form each parity bit. These parity bits are then added to the original data bits to generate the encoded message. The following are typical instances of how the check-bits are initially determined throughout the symbol 1 and symbol 2 encoding process:

Symbol 1:

$$
\begin{aligned}
P_0 &= D_8 \oplus D_9 \oplus D_{10} \oplus D_{11} \oplus D_{15}\\
P_1 &= D_1 \oplus D_5 \oplus D_6 \oplus D_{10} \oplus D_{14}\\
P_2 &= D_0 \oplus D_4 \oplus D_7 \oplus D_9 \oplus D_{13}\\
P_3 &= D_3 \oplus D_4 \oplus D_8 \oplus D_{12}\\
P_4 &= D_0 \oplus D_2 \oplus D_5 \oplus D_{10} \oplus D_{15}\\
P_5 &= D_2 \oplus D_5 \oplus D_9 \oplus D_{14}\\
P_6 &= D_0 \oplus D_4 \oplus D_5 \oplus D_8 \oplus D_{13}\\
P_7 &= D_1 \oplus D_3 \oplus D_7 \oplus D_{12}\\
P_8 &= D_2 \oplus D_3 \oplus D_6 \oplus D_{11}
\end{aligned}
$$

Symbol 2:

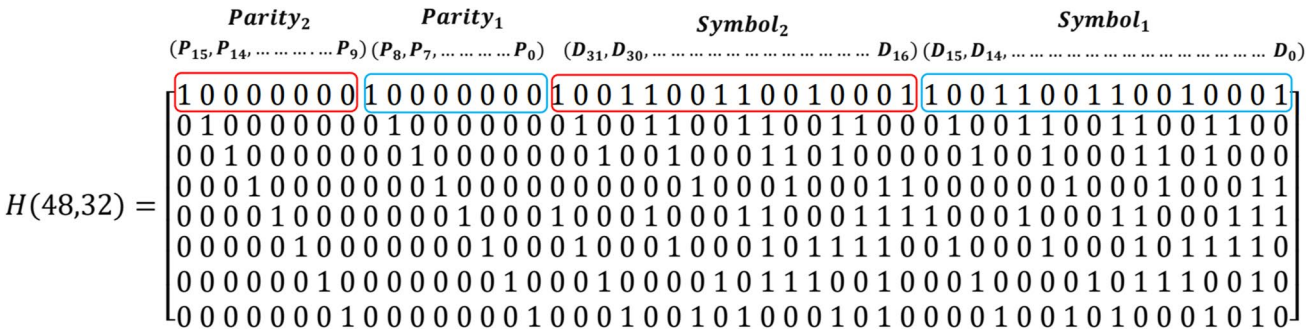


Fig. 4 Parity-check matrix H for the (48, 32) ICMC code

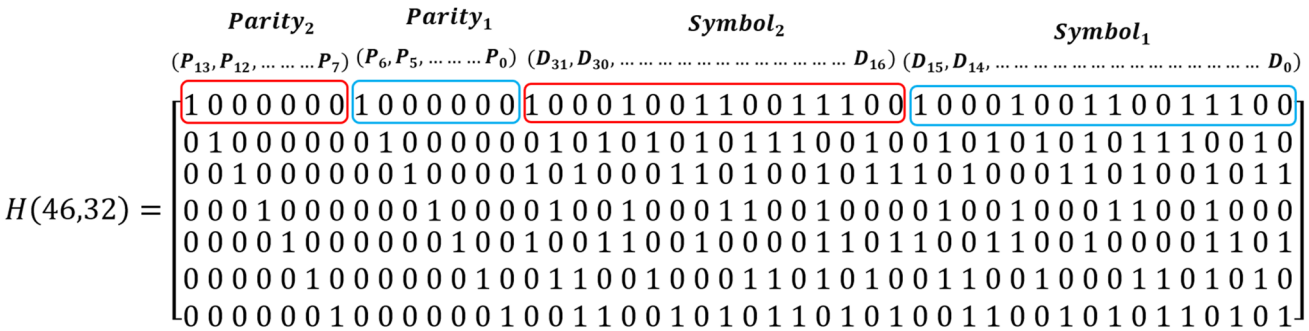


Fig. 5 Parity-check matrix H for the (46, 32) ICMC code

$$P_9 = D_{24} \oplus D_{25} \oplus D_{26} \oplus D_{27} \oplus D_{31}$$
$$P_{10} = D_{17} \oplus D_{21} \oplus D_{22} \oplus D_{27} \oplus D_{30}$$
$$P_{11} = D_{16} \oplus D_{20} \oplus D_{23} \oplus D_{25} \oplus D_{29}$$
$$P_{12} = D_{19} \oplus D_{20} \oplus D_{24} \oplus D_{28}$$
$$P_{13} = D_{16} \oplus D_{18} \oplus D_{21} \oplus D_{26} \oplus D_{31}$$
$$P_{14} = D_{18} \oplus D_{21} \oplus D_{25} \oplus D_{30}$$
$$P_{15} = D_{16} \oplus D_{20} \oplus D_{21} \oplus D_{24} \oplus D_{29}$$
$$P_{16} = D_{17} \oplus D_{19} \oplus D_{23} \oplus D_{28}$$
$$P_{17} = D_{18} \oplus D_{19} \oplus D_{22} \oplus D_{27}$$

The parity bits are located using the aforementioned equations, and they are then added to the front portion of the data to create the code word $P(P_0, P_1 \ldots P_{17})$, which is then stored in memory. Furthermore, the syndrome formulas given in H can be used to ascertain whether an error happened. To look for mistakes in the data that was received, utilize the syndrome formulae. The syndrome values (S) will be 0 in the absence of an error. The syndrome values will be non-zero if there is an error, which will help locate the error and indicate its presence. Parity check bits for the received data must be recalculated and compared to the received parity check bits as part of the procedure. An error can be found and fixed by utilizing the non-zero syndrome that is produced when there is a difference between the parity check bits that are received and those that are recalculated.

Symbol 1:

$$S_0 = P_0 \oplus D_8 \oplus D_9 \oplus D_{10} \oplus D_{11} \oplus D_{15}$$
$$S_1 = P_1 \oplus D_1 \oplus D_5 \oplus D_6 \oplus D_{10} \oplus D_{14}$$
$$S_2 = P_2 \oplus D_0 \oplus D_4 \oplus D_7 \oplus D_9 \oplus D_{13}$$
$$S_3 = P_3 \oplus D_3 \oplus D_4 \oplus D_8 \oplus D_{12}$$
$$S_4 = P_4 \oplus D_0 \oplus D_2 \oplus D_5 \oplus D_{10} \oplus D_{15}$$
$$S_5 = P_5 \oplus D_2 \oplus D_5 \oplus D_9 \oplus D_{14}$$
$$S_6 = P_6 \oplus D_0 \oplus D_4 \oplus D_5 \oplus D_8 \oplus D_{13}$$
$$S_7 = P_7 \oplus D_1 \oplus D_3 \oplus D_7 \oplus D_{12}$$
$$S_8 = P_8 \oplus D_2 \oplus D_3 \oplus D_6 \oplus D_{11}$$

Symbol 2:

$$S_9 = P_9 \oplus D_{24} \oplus D_{25} \oplus D_{26} \oplus D_{27} \oplus D_{31}$$
$$S_{10} = P_{10} \oplus D_{17} \oplus D_{21} \oplus D_{22} D_{27} \oplus D_{30}$$
$$S_{11} = P_{11} \oplus D_{16} \oplus D_{20} \oplus D_{23} \oplus D_{25} \oplus D_{29}$$
$$S_{12} = P_{12} \oplus D_{19} \oplus D_{20} \oplus D_{24} \oplus D_{28}$$
$$S_{13} = P_{13} \oplus D_{16} \oplus D_{18} \oplus D_{21} \oplus D_{26} \oplus D_{31}$$
$$S_{14} = P_{15} \oplus D_{18} \oplus D_{21} \oplus D_{25} \oplus D_{30}$$
$$S_{15} = P_{16} \oplus D_{16} \oplus D_{20} \oplus D_{21} \oplus D_{24} \oplus D_{29}$$
$$S_{16} = P_{17} \oplus D_{17} \oplus D_{19} \oplus D_{23} \oplus D_{28}$$
$$S_{17} = P_{18} \oplus D_{18} \oplus D_{19} \oplus D_{22} \oplus D_{27}$$

The syndrome will now be compared to various combinations of columns in the H matrix using the XOR operation. If the results match, the method will take the position of the columns that produced an XOR operation equal to the syndrome as the error position in the given codeword, flip the bits at that position, and extract the final 16 bits of the codeword as data. $(P_0, P_1 \ldots P_{17}, D_{31}, D_{30} \ldots D_0)$ is the codeword that was obtained after decoding. One-bit errors, two-bit burst errors, three-bit burst errors, and eight-bit burst mistakes can all be corrected by the suggested Interleaved Counter Matrix Code (ICMC). One additional bit of code will allow this to happen. In this instance, the proposed code requires 18 parity bits for a 32-bit data word. P_i and $D_i$are the parity and data bits, respectively, just like in the case of the proposed method. In the same way, designing the encoder/decoder circuitry is quite simple starting from H. For 32-bit data, the proposed ICMC approach only requires 16 parity bits. No code with these features has been discovered using the methodology.

The proposed method has produced a parity-check matrix that has been tuned to provide the shortest delay for the proposed codes, i.e., with fewer 1s in the rows with the most 1s. The proposed codes have relatively little

Table 3 Number of code bits for proposed with existing methods

Code	Number of code bits	Redundancy (%)	Burst error detection	Burst error correction
Matrix [21]	28	87.5	100% of 2-bit error	100% of single bit error
DMC [18]	36	112	100% of 2-bit error	100% of single error, 2-bit error
CMC [14]	24	81.35	100% of 6-bit error	100% of single error, 2, 3-, 4-, 5- & 6-bit error
ECC for5-bit adjacent error [24]	9	31.5	100% of 5-bit burst error	100% of single error, 2-bit and 3,4, 5-bit error
eMRSC [25]	56	75	100% of 4-bit burst error	100% of single error, 2-bit and 3, 4-bit error
DICE hardened cells with 7 bit burst error correction [26]	6	118.5	100% of 7-bit burst error	100% of single error, 5-bit and 6, 7-bit error
Proposed CAMBUC	14	43.75	100% of parallel 8-bit burst errors	100% of single bit error, 2, 3,4,8-bit parallel burst errors

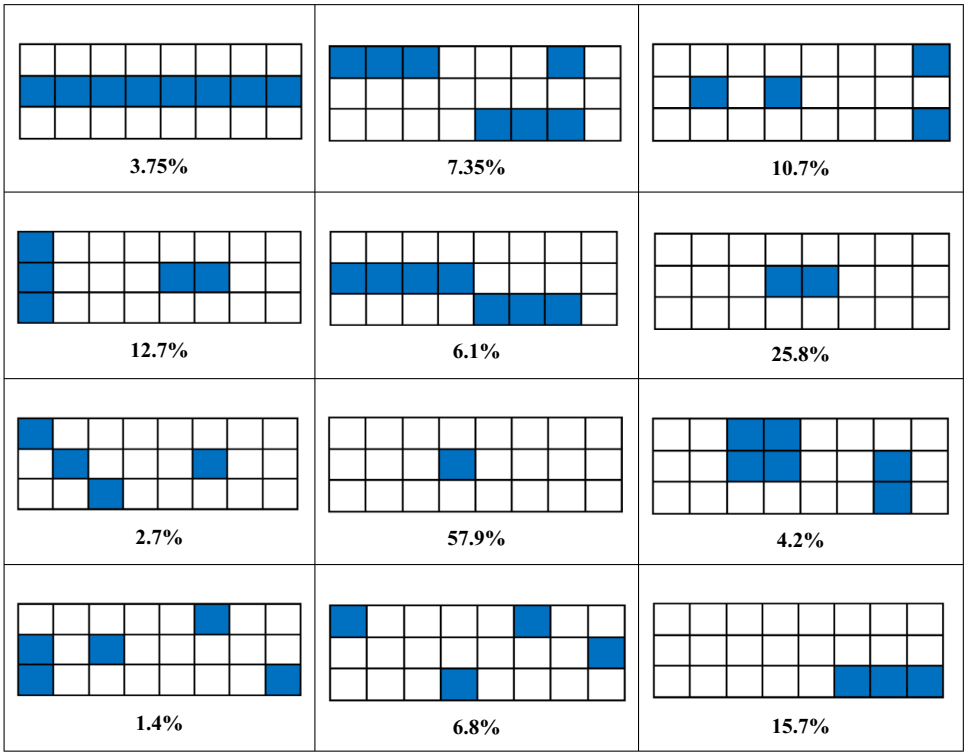


Fig. 6 MBU patterns of high occurrence probabilities error correction [27]

redundancy in terms of the required code bits. The following equation has been used to calculate the redundancy.

$$Redundency = \frac{Number\ of\ parity\ bits}{Number\ of\ data\ bits} \times 100 \quad (1)$$

Since it takes these extra bits to determine whether an error has occurred, low redundancy is essential. This method decreases the amount of storage space that can hold data bits by increasing redundancy. Approximately 695 MB of primary data can be stored by using the recommended code.

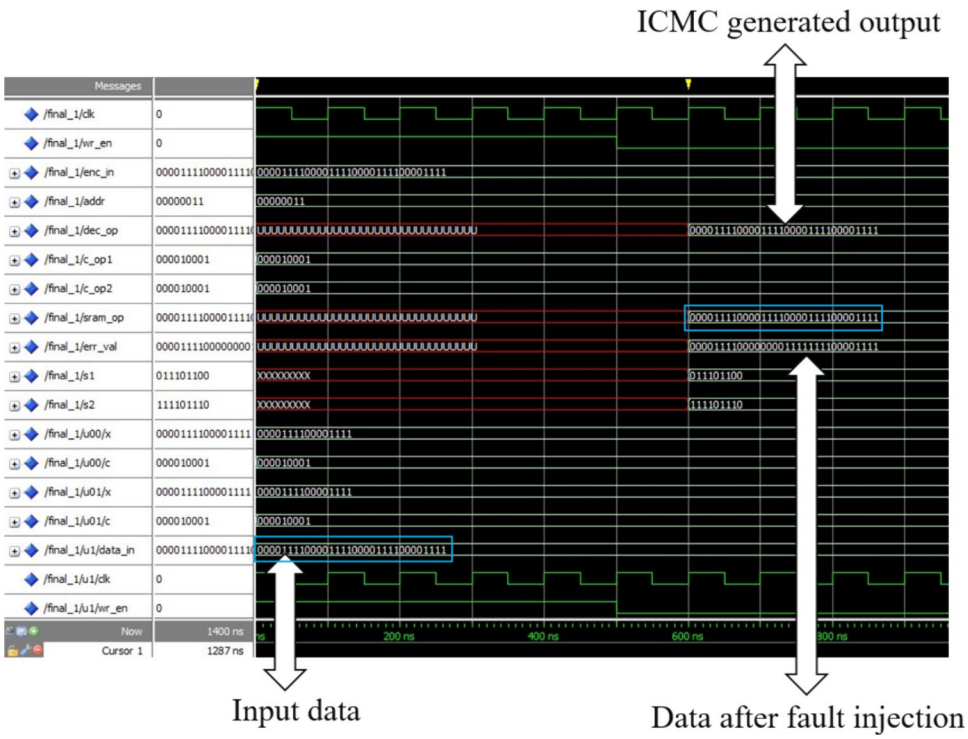


Fig. 7 Simulation output for proposed ICMC

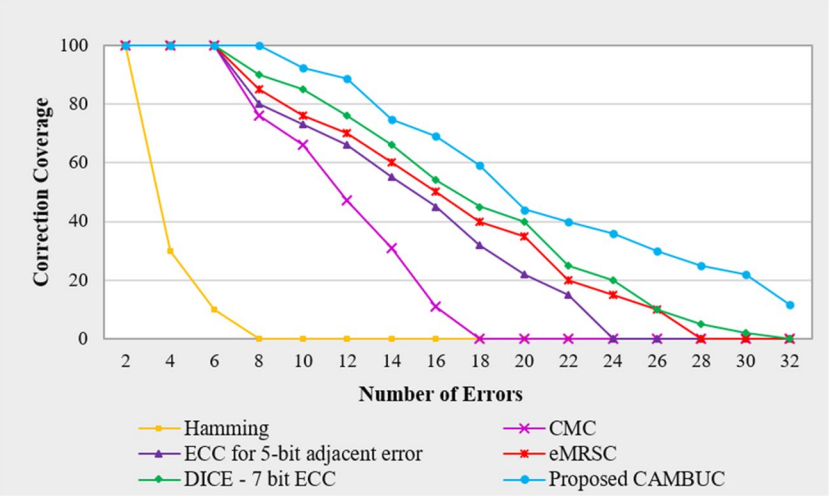


Fig. 8 Correction coverage for various ECCs

Table 4 Performance comparison of MTTF analysis

λ (upsets/bits per day)	Hamming [23]	CMC [14]	ECC for 5-bit adjacent error [24]	eMRSC [25]	DICE-7-bit ECC [26]	Proposed CAM-BUC
10^{-3}	110.4	945.45	993.5	993.7	1077.6	1124.2
10^{-4}	247.7	1148.8	1199.5	1256.4	1295.8	1344.8
10^{-5}	2477.4	10485.6	11975.8	12758.7	13187.7	13547.9

As can be observed, even after accounting for the proposed codes' improved coverage properties, there has been a large increase in the storage that is available (Table 3).

4 Results and Discussion

The complexity and implementation costs of the suggested codes are compared to those of the current approaches in this section. HDL that was functionally tested with numerous inputs after Model Sim simulation. The Synopsys Design Compiler and SMIC 0.18 µm technologies were used to construct the encoder and decoder. Measurements

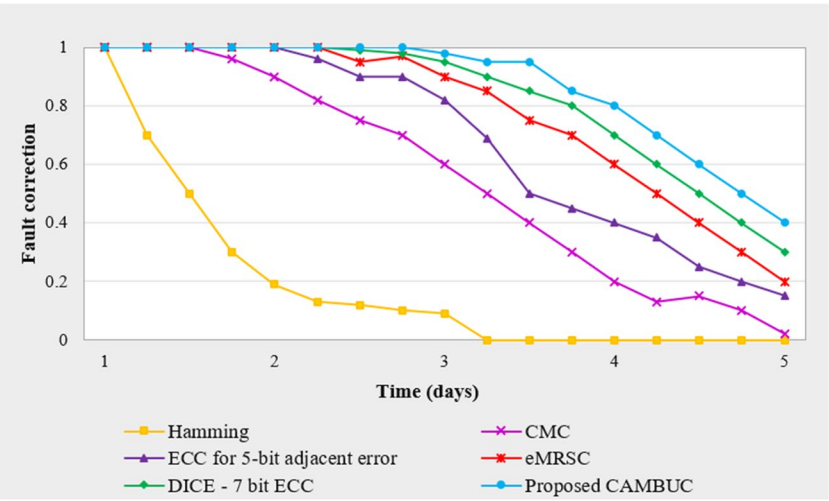


Fig. 9 Correctable probability of different techniques

have been made of the extra circuits' critical path delay, size, and power. Figure 6 shows the Continuous adjacent errors MBU patterns of high occurrence probabilities error correction. This section concludes with an analysis of the results.The FPGA chip can run at a base clock rate of 40 MHz in space applications using the suggested ICMC approach, while it may be able to operate at higher clock rates. It is located on a sbRIO-9636 board manufactured by National Instruments, Inc. in Austin, Texas, USA. The board also has a 400 MHz PowerPC real-time CPU. Onboard ADC and DAC chips make it simpler to input and output analog data into and from the FPGA.

The simulation results for the proposed Interleaved Counter Matric Code (ICMC) are displayed in Fig. 7. 32-bit data is first provided as an input and is kept in the SRAM memory, indicated by the blue box. By explicitly inserting errors into the data, the technique known as "injection" is used to assess the resilience of the error-correcting system in the input data. The proposed ICMC approach fixes the error and restores the data to its initial state.

4.1 Error coverage evaluation

The proposed method allows different sorts of errors to be injected to calculate the error coverage of the various ECCs. The input and output words can be compared by the simulator to establish whether the inserted defect leads to a correct or faulty decoding. Each code's correction coverage has been estimated as

$$Cov_{cor} = \frac{E_c}{E_I} \times 100 \tag{2}$$

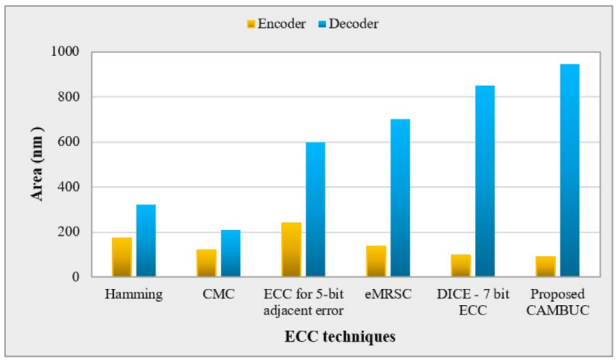
where E_c represents the error correction code was able to correct, and E_I is the total number of mistakes that were injected during the duration of the burst. The findings for each code's correction coverage are shown in Fig. 7. Various intra-word error test cases were used to determine the corrective coverage of different MBU mitigation strategies, including Hamming, CMC, ECC for 5-bit adjacent error, eMRSC, and DICE-7 bit ECC.

Figure 8 depicts the correction coverage of various ECC techniques. The proposed CAMBUC approach accomplishes 55% error correction in 32-bit mistakes while doing 100% error correction for parallel 8-bit faults. For error correction up to 6 bits, CMC performs 100%, while for error correction up to 26 bits, 12.3%. Since they can each correct up to 1-6-bit burst faults, as envisaged, the proposed CAMBUC codes offer superior correction coverage than the existing error correction codes. These findings demonstrate how our proposed method offers 8-bit rectification while also having effective tolerance against big MCUs that outperform other codes.

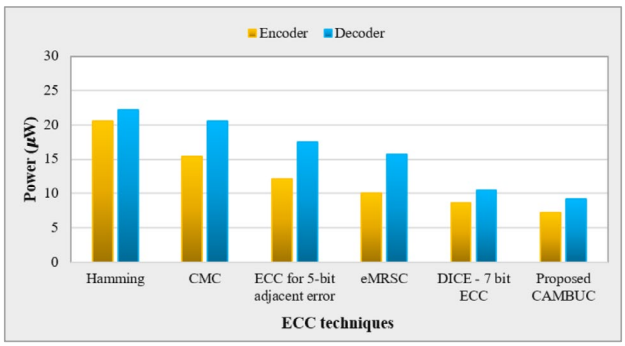
4.2 Reliability estimation

The mean time to failure (MTTF) can be used to examine the proposed ICMC code's dependability. MCUs are thought to access memories with a Poisson distribution. The correctable probability J_s for a memory of M words can be computed by;

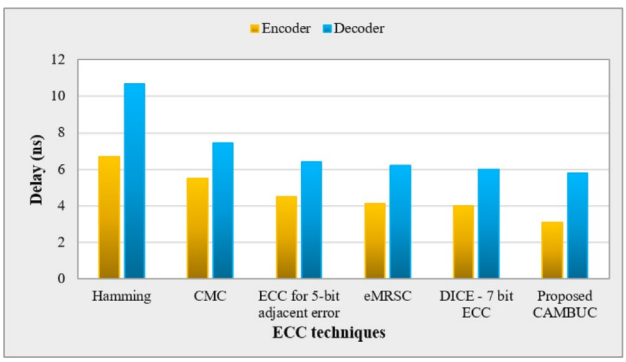
$$J_l = \sum_{p+q+\ldots+t} \frac{D_M^i}{M^i} S_p^1 S_q^2 \ldots S_t^i \tag{4}$$



(a)



(b)



(c)

Fig. 10 Analysis of encoder and decoder (**a**) area (**b**) power (**c**) delay

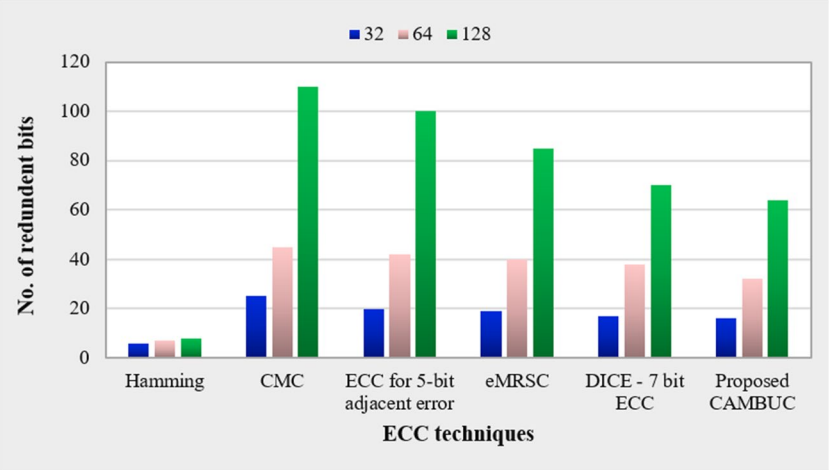


Fig. 11 Redundant bits for various error correction codes

The variables $i\,(i \leq l)$ and S_t^i represent the number of words affected by radiation events, the correctable probability when t radiation events influence i words, and D_M^i is the selection of i from M words (32) in memory. Figure 8 shows the correctable probability $J_s$for several protection code combinations. Evidence is presented that the proposed scheme outperforms the other codes in terms of correctable probability J_s.

The MTTF can then be determined using the preceding equation and Eq. (5), which is the integral of the function.

$$MTTF = \int_0^\infty J_l(t)dt \tag{5}$$

Table 4 shows the MTTFs for various codes for varying event arrival rates. The suggested CAMBUC scheme has a larger MTTF than Existing Hamming, CMC, ECC for 5 bit adjacent error, eMRSC, and DICE-7 bit ECC.by more than 347.9%, 100.4%, 90.6%,80.75%, and 72.5% respectively, according to this table.

4.3 Overhead analysis

The delay, size, and power overheads of the encoder and decoder for each protective code are shown in Fig. 9. In comparison to other codes, the proposed CAMBUC method has a significant reduction. The area in nm^2 ($1\ nm = 10^{-9}$ m) occupied by the various circuits is depicted in Fig. 10a.

As can be seen, the encoders for the proposed method offer a little bit more area than the existing encoders. Their greater redundancy, which causes complicated circuitry in encoders, is what causes these findings. The overhead power consumption (in $\mu\,W, 1\mu\,W = 10^{-6}\,W$)of the several ECCs is shown in Fig. 10b. According to the experimental findings, the proposed CAMBUC codes have the least amount of power consumption (both encoder and decoder). However, the encoding algorithm for the proposed CAMBUC is fairly straightforward, resulting in negligible overheads. Figure 10c depicts the delay caused by the various codes (in $ps, 1ps = 10^{-12}s$). The proposed approach exhibits the lowest delay compared to Hamming, matric code, which exhibits the largest delay.

It is necessary to assess the implementation overheads of these protection codes to assess the performance of error mitigation approaches. In terms of cost and correction coverage per cost, this study analyses overheads (CCC). Cost denotes how many redundant bits are needed to create error correction codes. For 32, 64, and 128 bits, Fig. 11 shows the cost for the proposed and standard coding schemes. As a result, it follows that Hamming code only requires a very small number of duplicated bits, but their ability to rectify errors is only 1. Figure 11 depicts the linear increase of superfluous bits for the longer word lengths of the conventional codes. Compared to all previous codes, the proposed CAMBUC requires less superfluous bits because it can process words in parallel.

5 Conclusion

In this research, Interleaved Counter Matrix Code (ICMC) has been proposed for Continuous Adjacent Multiple Bit Upset Correction (CAMBUC) that greatly reduces the redundancy and improves correction coverage. In addition, the proposed method predicts MBU before decoding and maximum parallel 8 error bits are corrected using the proposed method. Using this method, the error correction code's parity check matrix is automatically and effectively constructed, simply stating its error detection and/or correction capabilities. Combinatorial counting operations introduce error detection and error prediction. The proposed ICMC makes use of the decimal and hamming algorithms to achieve the highest level of error detection. Moreover,

the Interleaved Counter Matrix Code (ICMC) is proposed to simplify the Encoder and decoder circuits simpler which reduces the delay.In comparison to the conventional decoder, which can rectify double and 8-adjacent errors, the assessment results utilizing the proposed CAMBUC approach demonstrate significant reductions in area, power, and delay. The obtained findings demonstrate that the proposed method has a comparatively long mean time to failure (MTTF) when compared to existing approaches. At the same time, the suggested scheme's delay overhead is, respectively, 20.5%, 14.6%, 11.2%, 8.5%, and 3.5% less than that of the existing Hamming, Matrix, DMC, QAUC, and CMC methods.

Acknowledgements The author would like to acknowledge the Deanship of Graduate Studies and Scientific Research, Taif University for funding this work.

Data Availability Data sharing not applicable to this article as no datasets were generated or analyzed during the current study.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Maity RK, Tripathi S, Samanta J, Bhaumik J (2020) Lower complexity error location detection block of adjacent error correcting decoder for SRAMs. IET Computers Digit Techniques 14(5):210–216
2. Ahilan A, Deepa P (2015) A reconfigurable virtual architecture for memory scrubbers (VAMS) for SRAM based FPGA's. Int J Appl Eng Res 10(10):9643–9648
3. Argyrides CA, Reviriego P, Pradhan DK, Maestro J (2010) A. Matrix-based codes for adjacent error correction. IEEE Trans Nucl Sci 57(4):2106–2111
4. Maity RK, Samanta J, Bhaumik J (2023) Single and double-adjacent error correcting code (SDECC) with lower design overheads and mis-correction rate for SRAMs. Microsystem Technologies, pp 1–12
5. Appathurai A, Deepa P, August (2015) Design for reliablity: A novel counter matrix code for FPGA based quality applications. In 2015 6th Asia Symposium on Quality Electronic Design (ASQED), IEEE, 56–61
6. Senthil Singh C, Sameena Naaz, Saranya G (2024) Iot-Centric Data Protection using deep learning technique for preserving security and privacy in cloud. Int J Data Sci Artif Intell 02(03):81–87
7. Sathiya RR, Rajakumar S, Sathiamoorthy J (2023) Secure blockchain based deep learning approach for data transmission in IOT-enabled healthcare system. Int J Comput Eng Optim 01(01):15–23
8. Maity RK, Samanta J, Bhaumik J (2023) Construction technique and evaluation of high-performance t-bit burst error correcting codes for protecting MCUs. J Circuits Syst Computers 32(09):2350142
9. Ahilan A, Deepa P, January (2015) Modified Decimal Matrix Codes in FPGA configuration memory for multiple bit upsets. In: 2015 International Conference on Computer Communication and Informatics (ICCCI), IEEE, 1–5
10. Das A, Touba NA (2019) A new class of single burst error correcting codes with parallel decoding. IEEE Trans Comput 69(2):253–259
11. Maity RK, Samanta J, Bhaumik J (2022) FPGA-based low delay adjacent triple-bit error correcting codec. In: Internet of Things and its applications: Select Proceedings of ICIA 2020. Springer Nature, Singapore, pp 419–429
12. Saiz-Adalid LJ, Gracia-MoranJ, Gil-Tomas D, Baraza-CalvoJC, Gil-Vicente PJ (2019) Ultrafast codes for multiple adjacent error correction and double error detection. IEEE Access 7:151131–151143
13. Prasad SVS, Natarajan PB, Shankar LB, June (2021) Broadening 3-bit burst error-correction codes with quadruple adjacent error correction. In: 2021 International Conference on Intelligent Technologies (CONIT), IEEE, 1–5
14. AppathuraiAand Deepa P (2016) Radiation induced multiple bit upset prediction and correction in memories using cost efficient CMC. Informacije MIDEM 46(4):257–266
15. ReviriegoP, Liu S, Xiao L, Maestro JA (2015) An efficient single and double-adjacent error correcting parallel decoder for the (24, 12) extended golay code. IEEE Trans Very Large-Scale Integr (VLSI) Syst 24(4):1603–1606
16. Li J, Reviriego P, Xiao L, Argyrides C, Li J (2017) Extending 3-bit burst error-correction codes with quadruple adjacent error correction. IEEE Trans Very Large-Scale Integr (VLSI) Syst 26(2):221–229
17. Gracia-Moran J, Saiz-Adalid LJ, Gil-Tomas D, Gil-Vicente PJ (2018) Improving error correction codes for multiple-cell upsets in space applications. IEEE Trans Very Large-Scale Integr (VLSI) Syst 26(10):2132–2142
18. Guo J, Xiao L, Mao Z, Zhao Q (2013) Enhanced memory reliability against multiple cell upsets using decimal matrix code. IEEE Trans Very Large-Scale Integr (VLSI) Syst 22(1):127–135
19. Saiz-Adalid LJ, Reviriego P, Gil P, Pontarelli S, Maestro JA (2014) MCU tolerance in SRAMs through low-redundancy triple adjacent error correction. IEEE Trans Very Large-Scale Integr (VLSI) Syst 23(10):2332–2336
20. Bhargavi C, Nishanth DV, Nikhita P, Vinodhini M (2014) H-matrix based error correction codes for memory applications. In: 2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), IEEE, 1–5
21. Argyrides C, Pradhan DK, Kocak T (2019) Matrix codes for reliable and cost-efficient memory chips. IEEE Trans Very Large-Scale Integr (VLSI) Syst 19(3):420–428
22. Castro HDS, da Silveira JA, Coelho AA, e Silva FG, Magalhães PDS, de Lima OA, June (2016) A correction code for multiple cells upsets in memory devices for space applications. In: 2016 14th IEEE International New Circuits and Systems Conference (NEWCAS), IEEE, 1–4
23. Hamming RW (1950) Error detecting and error correcting codes. Bell Syst Tech J 29(2):147–160
24. Karan KS, Srikanth N, Agrawal S (2019) A robust code for MBU correction till 5-bit error. In: 2019 International Conference on Communication and Electronics Systems (ICCES). IEEE, pp 1524–1529
25. Silva F, Freitas W, Silveira J, Marcon C, Vargas F (2020) Extended matrix region selection code: an ECC for adjacent multiple cell upset in memory arrays. Microelectron Reliab 106:113582
26. Gil-Tomás D, Saiz-Adalid LJ, Gracia-Morán J, Baraza-Calvo JC, Gil-Vicente PJ (2024) A hybrid technique based on ECC and hardened cells for tolerating Random multiple-bit upsets in SRAM arrays. IEEE Access

27. Ebrahimi M, Rao PMB, Seyyedi R, Tahoori MB (2015) Low-cost multiple bit upset correction in SRAM-based FPGA configuration frames. IEEE Trans Very Large-Scale Integr (VLSI) Syst 24(3):932–943

Ahilan A received Ph.D. from Anna University, India, and working as an Associate Professor in the Department of Electronics and Communication Engineering at PSN College of Engineering and Technology, India. His area of interest includes FPGA prototyping, Computer vision, the Internet of Things, Cloud Computing in Medical, biometrics, and automation applications. Served Guest editor in several journals of Elsevier, Benthom, IGI publishers. Also, have contributed original research articles in IEEE Transactions, SCI, SCIE, and Scopus indexed peer-review journals. He presented various international conference events like ASQED (Malaysia), ESREF (France). He is doing as a reviewer in IEEE Industrial Informatics, IEEE Access, Measurement, Multimedia Tools & Applications, Computer Networks, Medical systems, Computer & Electrical Engineering, neural computing and applications, Cluster Computing, IET Image Processing, and so on. He has IEEE and ISTE membership. He has worked as a Research Consultant at TCS, Bangalore, where he has guided many computer vision projects and Bluetooth Low Energy projects. Hands on programming in MATLAB, Verilog and python at various technical institutions around India.

Anusha Gorantla received a Bachelor's in Electronics and Communication Engineering in 2006 from JNTU Hyderabad, Andhra Pradesh, India. She received an MTech VLSI Design in 2008 and a Ph.D. in Information and Communication Engineering in 2018 from Anna University, Chennai, Tamilnadu, India. Now, she works as an Associate Professor, at the Department of Electronics and Communication Engineering, Raghu Engineering College, Visakhapatnam, Andhra Pradesh, India. Her research includes low-power VLSI Design, approximate computing, and image processing. She has served as a reviewer for various reputed international journals.

Gladys Kiruba received her UG degree in B. E (ECE) in PSN college of Engineering and Technology during the year 2004-2008 and received her master degree MTech (VLSI design) in karunya university during the year 2008-2010. She is currently working as Assistant professor in Bangalore college of Engineering and technology, chandapura, Bangaluru, India.

Asmaa A. Hamad is currently an Assistant Professor at the Department of Botany and Microbiology, Cairo University, Egypt, and she works as an assistant professor at the Biology Department, Faculty of Science, Taif University. She completed his degree in Botany, in 2003 at Cairo University, Egypt. She did his Master's Degree, in 2010 at Cairo University, Egypt. She obtained his Ph.D. at Cairo University, Egypt. His areas of Interest include environmental factors and bioremediation of polluted soil and water. She is now a member of many scientific committees and as reviewer in many scientific journals.

Mohamed M. Hassan is currently a Professor at the Department of Genetics, Menoufiya University of Egypt, and he works as an associate professor at the Biology Department, Faculty of Science, Taif University. He completed his degree in Molecular Biology, in 1999 at Tanta University, Egypt. He did his Master's Degree, in 2004 at the University of Menoufiya, Egypt. He obtained his Ph.D. at the University of Menoufiya, Egypt. His areas of Interest include Antibiotic resistance genes, Genomics, and bioinformatics. He is now a member of many scientific committees and reviewer for many scientific journals.

N. Venkatram is working as Pro-Vice Chancellor and Professor in ECE department in K.L.University since 25 years. He published 100+ research papers in International, National journals and Conferences. He published Scopus and SCI publications of 37 with H-Index of 8. He is reviewer for several international journals including IEEE, Elsevier, Springer, Wiley and Taylor and Francis and served as reviewer, keynote speaker and TPC member for several International conferences. His research interests include deep learning, data science and IoT.

Sindhu T. V. is a Ph D Scholar at Veltech Rangarajan Dr. Sagunthala R & D Institute of Science and Technology, Chennai, India. She is currently working as an Assistant Professor in the department of ECE under IES College of Engineering, Thrissur, Kerala. Her research interested areas are VLSI system design. She has 11 years of Teaching Experience.