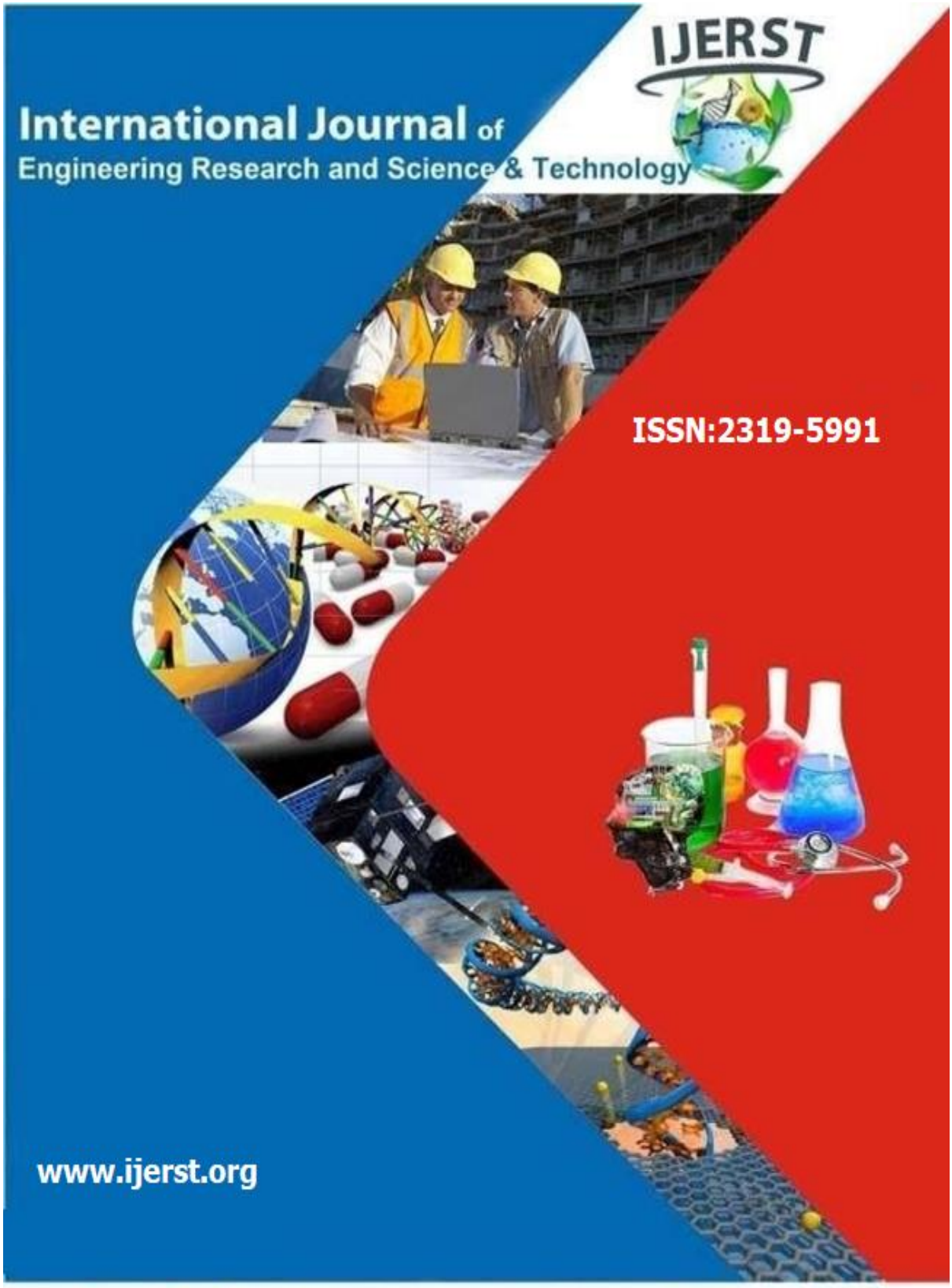
ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025



E-mail: editor@ijerst.org or ijerst.editor@gmail.com

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

DESIGN AND IMPLEMENTATION OF ERROR DETECTION AND CORRECTION SYSTEM FOR SEMICONDUCTOR MEMORY APPLICATIONS

[1] N.Sri Hari,[2] S.Madhav Rao, [3] Machavaram Pallavi
[1]Associate Professor,[2]Assistant Professor,[3]Student
Department of Electronics and Communication Engineering
Prakasam Engineering College

ABSTRACT
On behalf of technology scaling, on-chip memories in a die undergoes bit errors because of single events or multiple cell upsets by the ecological factors such as cosmic radiation, alpha, neutron particles or due to maximum temperature in space, leads to data corruption. Error detection and correction techniques (ECC) recognize and rectify the corrupted data over communication channel. In this paper, an advanced error correction 2-dimensional code based on divide-symbol is proposed to weaken radiation-induced MCUs in memory for space applications. For encoding data bits, diagonal bits, parity bits and check bits were analyzed by XOR operation. To recover the data, again XOR operation was performed between the encoded bits and the recalculated encoded bits. After analyzing, verification, selection and correction process takes place.

Extension:
A typical system-level technique to harden memory against multiple bit upsets (MBUs) would be the use of error correction codes (ECCs) for enhanced correction capabilities. Building updated ECCs with low redundancy and correction of errors however has be a significant issue, especially about adjacent ECCs. Present MBU mitigation codes concentrate primarily on correcting up to 3-bit explosive errors. The amount of impaired bits will quickly extend to even more than 3 bit as that of the software scales as well as the cell interval gap decrease. Consequently, the earlier approaches are not adequate to meet the criterion for durability in harsh conditions. In this article, a technique for 4-bit bursting bug fix (BEC) codes was introduced with a Multiple bit error detection and correction (MBEDC) codes. Initially you define the interface principles, then you create a search algorithm for locate the codes that correspond to both the rules. Usable are the 4-bit BEC H matrices with MBEDC codes. Any additional parity check bits were needed compared with such a BEC 3-bit code. That efficiency of 4-bit BEC was also substantially enhanced by adding the latest algorithm with existing 3-bit BEC codes. A project with verilog HDL would be built. The Simulation & Synthesis Xilinx ISE method is used.

I. INTRODUCTION

1.1MBEC ALGORITHM

The MBEC algorithm was introduced in 1967 as an efficient method for decoding convolutional codes [1], widely used in communication systems [2].This algorithm is utilized for decoding the codes used in various applications including satellite communication, cellular, and radio relay. It has proven to be an effective solution for a lot of problems related to digital estimation. Moreover, the MBEC decoder has practical use in implementations of high-speed (5 to 10 Gb/s) serializer-deserializers (SERDESs) which have critical latency con-straints. SERDESs can be further used in local area and synchronous optical networks of 10 Gb/s. Furthermore, they are used in magnetic or optical storage systems such as hard disk drive or digital video disk [3].

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

The MBEC algorithm process is similar to finding the most-likely sequence of states, resulting in sequence of observed events and, thus, boasts of high efficiency as it consists of finite number of possible states [4–7]. It is an effective implementation of a discrete-time finite state Markov process perceived in memory less noise and optimality can be achieved by following the maximum-likelihood criteria [8]. It helps in tracking the stochastic process state using an optimum recursive method which helps in the analysis and implementation [9, 10].

A top-level architecture for MBEC decoders is shown in Fig. 1.1. As seen in this figure, MBEC decoders are composed of three major components: branch metric unit (BMU), add-compare-select (ACS) unit, and survivor path memory unit (SMU). BMU generates the metrics corre-sponding to the binary trellis depending on the received signal, which is given as input to ACS which, then, updates the path metrics. The survival path is updated for all the states and is stored in the additional memory. SMU is responsible for managing the survival paths and giv-ing out the decoded data as output.

BMU and SMU units happen to be purely forward logic. ACS recursion consists of feedback loops; hence, its speed is limited by the iteration bound [11]. Hence, the ACS unit becomes the speed bottleneck for the system. M-step look-ahead technique can be used to break the iteration bound of the MBEC decoder of constraint length K [12–18]. A look-ahead technique can combine several trellis steps into one trellis step, and if M > K, then throughput can be increased by pipelining the ACS architecture, which helps in solving the problem of iteration bound, and is frequently used in high-speed communication systems.

Branch metric precomputation (BMP) which is in the front end of ACS is resulted due to the look-ahead technique and it dominates the overall complexity and latency for deep look-ahead architectures. BMP consists of pipelined registers between every two consecutive steps and combines binary trellis of multiple-steps into a single complex trellis of one-step. BMP dominates the overall latency and complexity for deep look-ahead architectures. Before the saturation of the trellis, only add operation is needed. After the saturation of the trellis, add operation is followed by compare operation where the parallel paths consisting of less metrics are discarded as they are considered unnecessary.

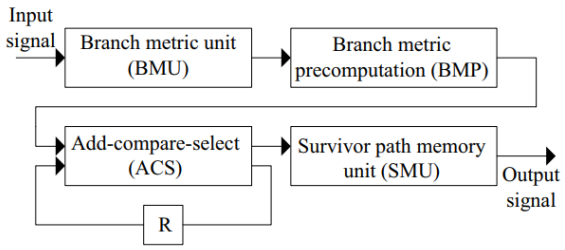


Figure 1.1: MBEC decoder block diagram.

Although MBEC algorithm architectures are used commonly in decoding convolutional codes, in the presence of very-large-scale integration (VLSI) defects, erroneous outputs can occur which degrade the accuracy in decoding of convolutional codes.

1.2 FAULT DIAGNOSIS

A fault in a system can be defined as a deviation from the expected working of the system which can be due to a defect of some components of the circuit. They can be temporary or permanent. Permanent faults are called as Solid or Hard faults and can result due to the wear-ing out or breaking of components. Temporary faults can be referred to as soft faults and these faults can be classified as intermiitent or transient as it occurs only at certain intervals of time. An intermittent fault occurs when the component is developing a permanent fault. A transient

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

fault can result due to some external disturbance like power supply fluctuations. Depending upon the effect of faults, they can be classified as parametric or logical. A parametric fault causes a change in speed, voltage or current as it alters the circuit parameter magnitude, while a logical fault ends up changing the Boolean function originally realized by the circuit. Delay fault which results due to slow gates is an important parametric fault and it leads to problems of critical races or Hazards. Fault extent can be local or distributed. A distributed fault af-fects multiple variables, whereas a local fault affects single variable. The clock malfunction is an example of a distributed fault while a logical fault is an example of a local fault. With the VLSI technology developing, the number of components on a single chip are increasing drastically thus also increasing the probability of fault occurrence. Thus, this is an important research area.

1.2.1Faults and Degrdation

Depending on the behavior of the system, logical faults represent the behavior of the system modeled. Logical faults has three important classes:

Stuck-at-faults: A single stuck-at-fault happens when either one of the inputs or the output of the logic gate is fixed at either a logic 1 (stuck-at-1) or a logic 0 (stuck-at-0). They can be denoted by abbreviations as s-a-1 and s-a-0 respectively. This fault model is a good representation for types of defects such as open circuits and short circuits. The stuck-at model can also represent multiple faults which results when multiple signal lines are stuck at logic 0 or logic 1.

Bridging faults: Bridging faults occur when two or more than two signal lines are accidentally connected together. They can be classified as:

Delay Faults: Due to the occurrences of the statistical variations in the manufacturing processes, the probability of appearance of smaller defects which causes partial short or open in a circuit, increases. Due to these defects, the circuit fails in meeting the timing specifications without altering the logic function of the circuit. The transition of the signal might get delayed from 1 to 0, or vice versa due to a small defect. This is called as delay fault. They are of two types:

Gate Delay Fault: It helps in modeling defects which causes the propagation delay of the faulty gate to exceed the worst case value specified. It can be used to model isolated defects but not distributed defects.

1.2.2Fault Detection Techniques

The process of determining whether the circuit contains a fault or not is called as fault de-tection [19–22]. As it is important to counteract such natural faults in order to achieve fault immunity and reliability, error detection has been an important part of a number of hardware architectures in different domains, including various arithmetic unit sub-components [23, 24]. In previous work, reliable architectures have been devised to counteract natural or malicious faults [25], e.g., cryptographic architectures immune to faults through concurrent error detec-tion [26–38]. The different fault detection strategies can be classified as follows:

Concurrent Error Detection: It helps in detecting the faults in the circuit concurrently with the normal operation of the circuit by making use of additional logic. It results in an error if the resulting output is found different than the predicted output by the checker unit [39, 40].The error coverage can be improved greatly using the methods of duplication or including parity check registers in the circuit. For improving the error coverage, the trade-off with area or latency, or throughput can be made. The errors can be also detected by running the circuit

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

twice, once with the original operands and the second time using encoded operands such that different outputs are obtained. The checker will raise the error indication flaf incase of a mismatch between the two ouputs. The operands can be encoded using different methods like Recomputing with Shifted Operands (RESO), Recomputing with Rotated Operands(RERO), also by a slight modification of the RESO model [41–43].

Off-Line Fault Detection: This method helps in identifying faults in FPGAs and ASICs when they are not in operation with the use of additional circuitry. It helps in detect-ing manufacturing defects. Automated-Test-Pattern-Generator (ATPG) and Built-in-Self-Test (BIST) are some examples of off-line test circuits. The fault detection process does not in-volve the original circuitry. It connects the device under test between a pattern generator and an output response analyzer. In order to obtain full error coverage, it is important to check the logic and interconnects and the configuration network. For the FPGAs [44, 45], the need of a large number of test configurations has been eliminated as the additional testing circuitry is built into the development boards by most of the recent consumer grade FPGAs [46]. BIST does not interfere with the normal FPGA operation, and also covers clock networks and PLLs which are complicated systems.

1.3 OBJECTIVE

In this thesis, we explore two approaches for two variants of sub-parts in the MBEC algo-rithm. Specifically, we note that both area/power consumption and throughput/efficiency degradations need to be minimized with respect to the proposed approaches; thus, we ex-plore signature-based approaches resulting in better efficiency at the cost of area/power con-sumption, and recomputing with encoded operands to achieve permanent and transient er-ror detection. For detecting the errors in the ACS unit, we utilize three variants, i.e., re-computing with shifted operands (RESO) [47], proposed modified RESO which has slightly less fault resilience effectiveness; yet, lower induced overhead, and recomputing with ro-tated operands (RERO) [48]. Our architectures also include hardware redundancy techniques through signature-based detection. Specifically for the adder components, we utilize a num-ber of variants of self-checking based on two-rail encoding. The architectures to which the schemes have been applied consist of two types of low-latency and low-complexity structures of MBEC decoders [3] with slight modifications.

We summarize the contributions of this thesis as follows:

- We propose error detection methods for the modified MBEC decoder with the consider-ation of objectives in terms of performance metrics and reliability. The error detection approaches along with the modifications help achieving high error coverage and through the proposed improvements, performance boost can be achieved. Variants of recomput-ing with encoded operands on a number of architectures within the modified MBEC decoder as well as signature-based approaches (including modified self-checking based on two-rail encoding) are presented as well. The mechanisms for making the proposed structures immune to faults have not been presented before.
- We have extensively simulated the proposed error detection architectures and the ob-tained results help in benchmarking the error coverage. The results of our simulation show that the reliability of the proposed architecture can be ensured.
- Finally, our proposed error detection MBEC decoders incorporating the error

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

detection approaches are implemented on application-specific integrated circuit (ASIC) [32nm li-brary] and field-programmable gate array (FPGA) [Xilinx Virtex-6 family]. The results indicate that the architectures can be reliably used. The proposed approaches can be utilized based on reliability objectives and performance/implementation metrics degra-dation tolerance

II. LITERATURE SURVEY

2.1EXISTING METHOD

This section focuses only on branch metric computation, leaving aside the operations of compare-and-discard. An optimal approach of BBG is taken into consideration in order to remove all redundancies which are usually responsible for longer delay and extra complexity, since various paths share common computations. Branch metrics computation is said to be carried out sequentially for a conventional MBEC decoder. When two consecutive binary-trellis steps are combined, for each state, there are two incoming and two outgoing branches, and the computational complexity is 4 × N. As the results do not depend on the order of the trellis combination, the way the trellis steps are grouped and combined helps in determining the computational complexity. The combination in a backward nested procedure can be ex-plained as follows. The main M-step trellises are divided into two groups consisting of m_0 and m_1 trellis steps. The binary decomposition on each subgroup goes on till it becomes a single trellis step. The decomposition helps in removing maximum possible redundancy and, thus, helps achieve minimum delay and complexity. Finally, it can be verified that the complexities involved in the BBG approach are less as compared to the ones in the intuitive approach.

2.1.1 Look-ahead-based Low-Latency Architectures

This approach is a highly-efficient design approach based on the BBG scheme for a general M which provides less or equal latency, and also has much less complexity compared to other existing architectures [3]. For constraint length K and M-step look-ahead, the execution of BMP is done in a layered manner. An M-step trellis is a bigger group consisting of $\frac{M}{K}$ sub-groups with a trellis of K-step. Thus, the total numbers of P1 processors needed are $\frac{M}{K}$ and each P1 is responsible for computing K-step trellises. Accordingly, we have the complexities and latencies of P1 and P2 as $Comp._{P1} = N(\sum^{k}_{i=2} 2^i) + N^2$, $Comp._{P2} = N^2(N - 1) + N^3$, and $Lat._{P1,P2} = K$, where $N = 2^{k-1}$ is the number of trellis states. For P1 processors, the complex-
ity of add operation is $N \sum^{k}= 2^i$ and that of the "compare" operation is N^2. Similarly, for P2 i2processors, the complexity of add operation is $N^2(N -1)$ and that of the compare operation is N^3. For both P1 and P2 processors, the latency is same, i.e., K; however, the complexity of P2 is larger than that of P1. As the BBG approach is very efficient in computing the branch metrics, more operations of trellis combination can be allotted into BBG-based P1 processors in order to reduce the number of P2 processors as they are expensive in terms of complexity. The trellis Steps L, which is computed in the P1 processors, has the constraint of being less than 2 ×K in order to make sure that the latency feature is not lost. The number of groups N_g can be determined by $Ng = 2^{\lfloor \log2(\frac{M}{K}) \rfloor}$.

The overall layered structure of the MBEC algorithm is shown in Fig. 2.1 (in this figure, i, j ∈ [1, N] and l ∈ [1, K]). As seen in this figure, within two layers (shown by Layer 1 and Layer 2 in Fig. 2.1), we have N_g steps, going through P1 and P2 processors. In each L-level P1 processor, the initial step combination is performed using the BBG approach, followed by concatenated add–

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

compare operations executed one step at a time for the remaining L−K-step phase-II computation. In Layer 2, the outputs of P1 processors are combined for computing the final equivalent complex trellis. This figure also shows the P1 processor architecture based on the BBG algorithm. In Layer 1, although P1 leads to longer latency, as the depth of Layer 2 is reduced as well, latency penalty in not incurred.

2.1.2 Alternative to Look-ahead Approach

As the state nodes are connected pairwise, there are a total of N^2 connections, consisting of $2^{(M-K+1)}$ parallel paths. The number of parallel paths increases exponentially with respect to M, thereby, increasing the complexity. Generally, the exponential increase of parallel paths is avoided by a compare operation performed in each binary-trellis steps combination, thus, the parallel paths with less metrics are always discarded. Nonetheless, each of such add-compare operations results in a substantial amount of latency. The complexity efficiency of look-ahead depends on constraint length of MBEC decoder. For larger constraint lengths, latency reduction is achieved at the expense of prohibitive computational complexity which limits the application of look-ahead-based architectures.

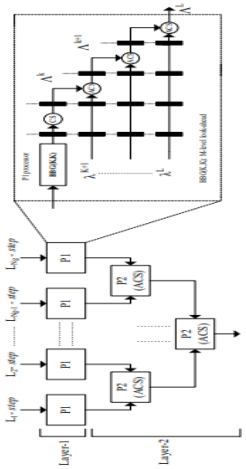


Figure 2.1: Overall layered structure including the P1 processor architecture.

2.2 proposed method

It is well-known that in different variants of concurrent error detection, either redundancy in hardware, i.e., increase in area/power/energy consumption, e.g., through error detection codes such as hamming codes, or redundancy in time, adding negligible area overhead at the expense of higher total time (throughput and latency), is performed. In this thesis, we utilize recomputing with encoded operands, where, the operations are redone for different operands for detecting errors. During the first step, operands are applied normally. In the recomputed step, the operands are encoded and applied and after decoding, the correct results can be generated. Moreover, through signature-based schemes, we propose schemes through which both transient and permanent errors can be detected.

The sequential branch metric computation unit is shown in Fig. 3.1. In order to make the ACS structure fast, parallelization of add and compare operations within the ACS itself is done (which leads to the reduction of iteration bound delay by 50%). For achieving that, the number of states is doubled and the channel response is extended by an extra bit. For a complex trellis to have P-level parallelism, there should be 2P parallel paths for each branch. For the initial K − 1 steps, there is no compare operation, but for the remaining M − K + 1 steps, the add operation is followed by a compare operation which helps in eliminating parallelism. Add and compare operations need to be performed sequentially. For this algorithm, as seen in Fig. 3.1, the order of operations from add-compare is changed to compare-add and that is attributed as a carry-select-add (CSA) unit. The pre-computed CSA (PCSA) is its speed-optimized variant, the details are not presented for the sake of brevity (the PCSA architecture is preferred only for large K and small M values).

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

We utilize signature-based prediction schemes for the CSA and PCSA units. We note that even a single stuck-at fault in such units may lead to erroneous (multi-bit) result (the error may also propagate to the circuitry which lies ahead of the affected location, with the domino effect propagated system-wise). Signatures (single-bit, multiple-bit, or interleaved parity, cyclic redundancy check, and the like, to name a few) are employed in our proposed scheme for all the registers. Moreover, self-checking adders based on dual-rail encoding are included for the adder modules.

III. DESIGN CONSIDERATIONS

It is well-known that in different variants of concurrent error detection, either redundancy in hardware, i.e., increase in area/power/energy consumption, e.g., through error detection codes such as hamming codes, or redundancy in time, adding negligible area overhead at the expense of higher total time (throughput and latency), is performed. In this thesis, we utilize recomputing with encoded operands, where, the operations are redone for different operands for detecting errors. During the first step, operands are applied normally. In the recomputed step, the operands are encoded and applied and after decoding, the correct results can be generated. Moreover, through signature-based schemes, we propose schemes through which both transient and permanent errors can be detected.

3.1 Unified Signature-based Scheme for CSA and PCSA Units within BMP

The sequential branch metric computation unit is shown in Fig. 3.1. In order to make the ACS structure fast, parallelization of add and compare operations within the ACS itself is done (which leads to the reduction of iteration bound delay by 50%). For achieving that, the number of states is doubled and the channel response is extended by an extra bit. For a complex trellis to have P-level parallelism, there should be 2P parallel paths for each branch. For the initial K − 1 steps, there is no compare operation, but for the remaining M − K + 1 steps, the add operation is followed by a compare operation which helps in eliminating parallelism. Add and compare operations need to be performed sequentially. For this algorithm, as seen in Fig. 3.1, the order of operations from add-compare is changed to compare-add and that is attributed as a carry-select-add (CSA) unit. The pre-computed CSA (PCSA) is its speed-optimized variant, the details are not presented for the sake of brevity (the PCSA architecture is preferred only for large K and small M values).

We utilize signature-based prediction schemes for the CSA and PCSA units. We note that even a single stuck-at fault in such units may lead to erroneous (multi-bit) result (the error may also propagate to the circuitry which lies ahead of the affected location, with the domino effect propagated system-wise). Signatures (single-bit, multiple-bit, or interleaved parity, cyclic redundancy check, and the like, to name a few) are employed in our proposed scheme for all the registers. Moreover, self-checking adders based on dual-rail encoding are included for the adder modules.

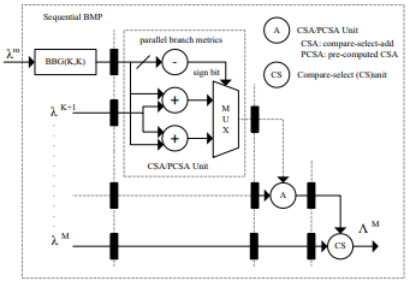


Figure 3.1: Sequential branch metric computation unit including CSA (PCSA) structures.

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

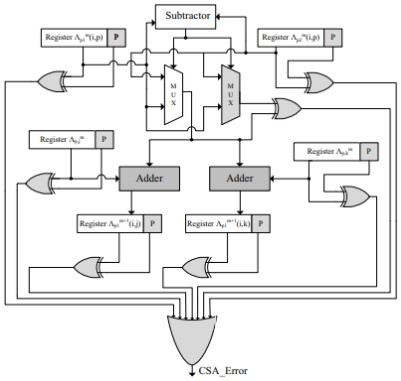


Figure 3.2: The CSA signature-based error detection approach (the shaded adders are variants of the original ones with the proposed error detection schemes).

As shown in Figs. 3.2 and 3.3, respectively, in the CSA unit, there exists a single multiplexer whereas for the PCSA unit, the original design contains two multiplexers, for which the results of the original and the duplicated multiplexers are compared using an XOR gate whose output is connected as one of the inputs to the OR gate. The input and output registers are incorporated with additional signatures, e.g., single-bit, multiple-bit, or interleaved parity, cyclic redundancy check, to detect faults (in figures, "P" denotes parity but it could be a chosen signature based on the overhead tolerance and reliability constraints). An OR gate for the units is required to derive the error indication flags. The OR gate raises the error indication flags (CSA_Error in case of the CSA unit and PCSA_Error in case of the PCSA unit) in case an error is detected.

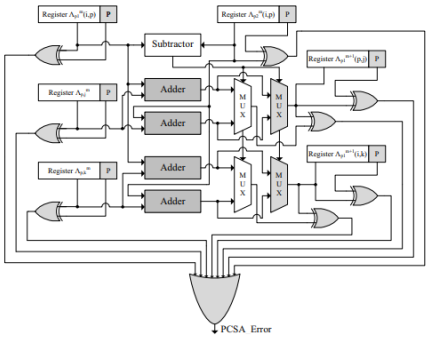


Figure 3.3: Signature-based PCSA error detection (the shaded adders include the proposed error detection schemes)

The CSA signature-based error detection approach (the shaded adders are variants of the original ones with the proposed adders included in both CSA and PCSA units, we have used self-checking adders as shown in Fig. 3.4 (some previous works include [35, 36, 49–52]). As shown in this figure, the adders are cascaded to implement a self-checking adder of arbitrary size. It consists of five two-pair two-rail checkers and also four full adders and two multiplexers are repeated n times. For the normal operation, no additional delay has resulted due to self-checking feature. The checker has two pairs of inputs driven in such a way that in the fault free scenario, the outputs are equal pairwise. This is performed using XNOR gates and appropriate connections.

There are two outputs from the checker and the outputs are also in two-rail form as the inputs. Even if one of the inputs of the checker has a fault, the output is not in two-rail form and, thus, an error indication flag is raised to indicate that a fault has been incurred in the system.

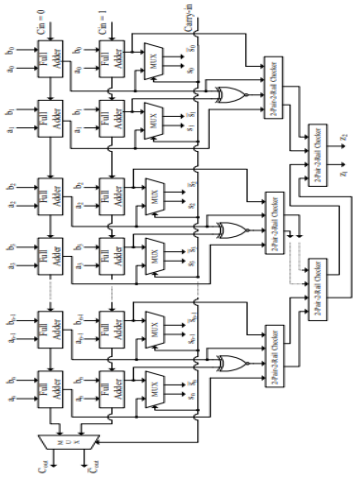


Figure 3.4: Self checking adder in the proposed scheme.

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

The adders as shown in Fig. 3.5 can also be implemented in both CSA and PCSA designs using the modified self-checking adder [53].In this variant, two n-bit ripple carry adders are used to precompute the sum bits with complemented values of carry-in, i.e., 0 and 1, and the original value of carry-in is used to select the actual sum bits. We employ this new adder [24] in the architectures and evaluate its performance and efficiency. Fig. 3.5 shows the design module of this variant for self-checking carry-select adder; the area overhead of which is found to be in the range of 20%-35% based on the input bit-size. An important modification done in this new adder is the inputs given to the two-pair two-rail checker. For carrying out the implementation for n bits, it needs (n−2) AND gates, (n+1) MUXes, (n−1) XNOR gates, (2n) full adders, and (n−1) two-pair two-rail checkers.

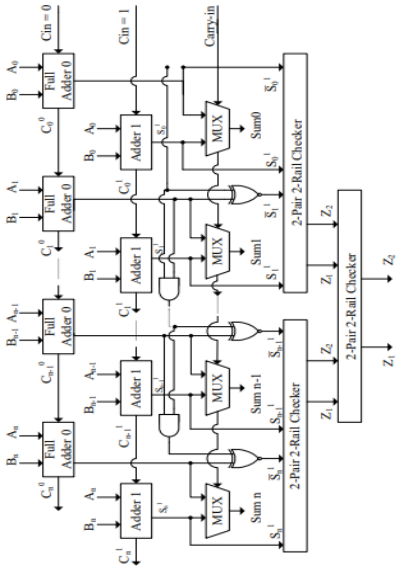


Figure 3.5: A variant of self checking adder utilized in the devised approach.

3.2 Recomputing with Encoded Operands for CSA and PCSA

In this section, the error detection CSA and PCSA architectures are designed through recomputing with encoded operands, e.g., RERO, RESO, and variants of RESO, as shown in Figs. 3.6 and 3.7 with the locations of error detection modules shaded. Since this approach takes more number of cycles for completion, to alleviate the throughput degradation, the architecture is pipelined in the following fashion. First, pipeline registers are added to sub-pipeline the architectures, assisting in dividing the timing into sub-parts. The original operands are fed in during the first cycle. Nonetheless, during the second cycle, the second half of the circuit operates on the original operands and the first half is fed in with the rotated operands.

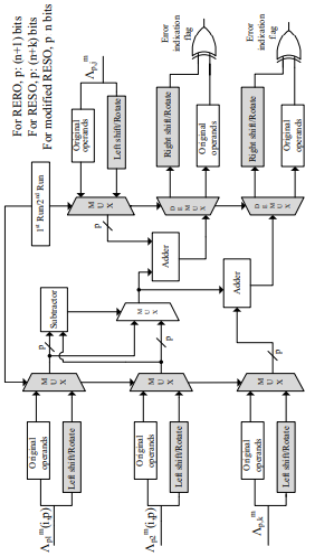


Figure 3.6: Recomputing with encoded operands for CSA.

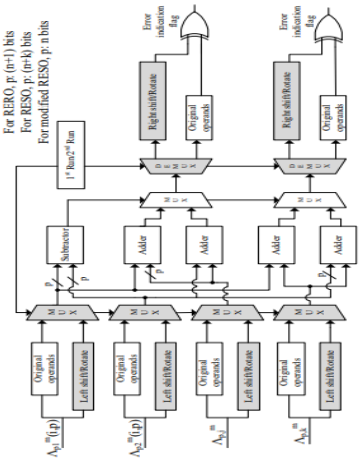


Figure 3.7: PCSA error detection through recomputing with encoded operands.

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

We have utilized RESO which performs the recomputation step with shifted operands, i.e., all operands are shifted left or right by k bits (this method is efficient in detecting k consecutive logic errors and k − 1 arithmetic errors). For CSA and PCSA architectures in Figs. 3.6 and 3.7, let us assume g(x, y) is the result of the operation which is stored in a register. The same operation is performed again with x and y shifted by certain number of bits. This new result g ′ (x, y) is stored and the original result g(x, y) can be obtained by shifting g ′ (x, y) in the opposite direction. Another used method in the proposed scheme is a modified version of the RESO scheme and this modification is that the bits that shift out are not preserved. This signifies that the total number of bits required for operation is only "n" bits and, hence, becomes more advantageous in terms of hardware cost than RESO and RERO methods, as pointed out in Figs. 3.6 and 3.7.

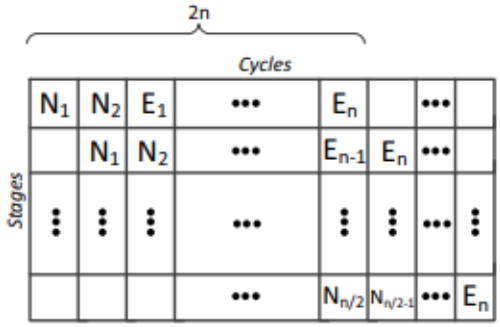


Figure 3.8: Compromise in asserting the encoded operands (can be tailored based on reliability constraints).

We have shown in Fig. 3.8 an approach based on which a compromise for the assertions is performed. Depending on the requirements, one can fulfill various reliability constraints. As seen in Fig. 3.8, a number of cycles are considered with the normal operands shown by N1 −Nn and the encoded operands shown by E1 −En. Let us assume that N1 is asserted at the beginning (first stage and first cycle). We have a number of options in the second cycle, e.g., asserting the second normal operand (N2) or the first encoded operand (encoded variant of N1 which is E1). Fig. 3.8 shows the former option as an example. In the third cycle, many options exist, among which asserting E1 has been chosen to depict in Fig. 10. This trend continues and after 2n cycles, one has En,En−1,...,Nn/2 as the entries to various stages. Such an approach ensures lower degradation in the throughput at the expense of more area overhead and can be tailored extensively based on the overhead tolerance and the reliability requirements.

IV. PROPOSED METHOD

I. INTRODUCTION

In space, due to high temperatures, electronic circuits, in particular memories subject to soft errors lead to poor reliability. Memory cells are interrupted by neutron or alpha particles from earth's atmosphere [3]. One way to reduce these errors by maximizing the critical charge at the state nodes or by well and substrate techniques (process related techniques). Another way is, by applying error correction codes (ECC) on memories with that some errors can be overcome [4]. This is usually performed by single error correction (SEC) code on each memory to deal with independent errors. Scrubbing with SEC increases accuracy.

It reads memory and corrects the single errors time to time; will not accumulate over time [5]. This is not effective for multiple cell upsets (MCUs) because in this MCU two or more bits of same memory gets affected. To overcome the multiple cell upsets in memories, interleaving method is proposed [6]. Many studies have been takes place with this technique to manage multiple cell upsets (MCUs). But this proposed interleaving method increases the system design complexity and shows impact in area and power consumption. So that ECCs is used in case for multiple cell upsets (MCUs). This

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com

Vol. 21, Issue 2, 2025

requires more parity bits, more time for decoding the bits and more complex circuits are needed to perform for the encoding and decoding operations. Various ECCs focused to reduce area, delay and power.

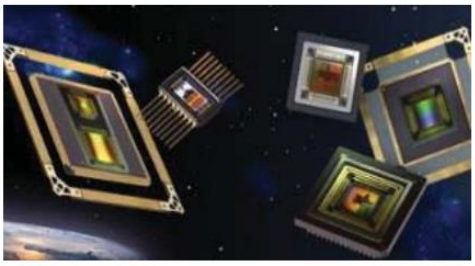


Fig. 1 ICs for space applications

Zhu et al., [7] proposes a new error correction codes for the reduction of radiation exposed multiple bit upsets in memories. This detects and corrects the adjacent double bit errors and also lowers the errors for the non-adjacent double bit errors. The experimental results demonstrate that it reduces 40% hardware redundancy and more efficient compared to other existing ECC codes. Also, this method minimizes the errors for non-adjacent DBE by 12% when compared with the conventional SEC-DED-DAEC codes leads to high reliability memory system design.

V. RESULTS

5.1 Simulation Result

waveform

RTL schematic

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice LUTs	55	63400	0%
Number of fully used LUT-FF pairs	0	55	0%
Number of bonded IOBs	77	210	36%

DESIGN SUMMARY

```
Data Path: in<13> to enc_out<23>
                           Gate     Net
  Cell:in->out     fanout   Delay   Delay  Logical Name (Net Name)
  ----------------------------------------  ------------
   IBUF:I->O            9   0.001   0.548  in_13_IBUF (in_13_IBUF)
   LUT3:I0->O           1   0.097   0.295  E1/Mxor_C<7>_xo<0>_SW0 (N6)
   LUT6:I5->O           2   0.097   0.283  E1/Mxor_C<7>_xo<0> (enc_out_23_OBUF)
   OBUF:I->O                0.000          enc_out_23_OBUF (enc_out<23>)
  ----------------------------------------
  Total                    1.322ns (0.195ns logic, 1.127ns route)
                           (14.7% logic, 85.3% route)
```

Time summary

VI. CONCLUSION AND FUTURE SCOPE

CONCLUSION

In this paper, a technique to extend the 3-bit BEC codes with the MBEC is presented. The proposed codes have the same redundancy as the previous 3-bit BEC codes. To accelerate the searching process of target matrices, a new algorithm with column weight control and recording function is proposed. Based on the proposed algorithm, a searching tool is developed to execute the searching process automatically. To prove the validity of the proposed algorithm, it is applied to the previous 3-bit BEC codes and the codes are remarkably improved on the two optimization criteria. Then, the proposed algorithm is used to find the solution for the MBEC. The complete solution searching process is finished for 16 data bits and the searching process using optimization algorithm is carried out for 32 and 64 data bits. Therefore, in this paper, the best solutions are presented for 16 data bits and the best solutions found in a reasonable computation time are presented for 32 and 64 data bits. The encoder and decoder of the proposed codes are implemented by using the HDL and synthesized for a 65-nm library. The overhead of area and delay is moderate versus previous 3-bit BEC codes. This suggests that the proposed 3-bit BEC with MBEC codes can be effectively used by designers to protect the SRAM memories from radiation effect and mitigate the MBUs that affect up to four adjacent bits. Finally, as noted before, the proposed scheme could be extended to design

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

3-bit burst ECCs that can correct another of the 4-bit burst error patterns instead of the quadruple adjacent error. This can be of interest for applications in which there is a dominant 4-bit burst error pattern that is not the quadruple adjacent error.

FUTURE SCOPE

CED technique for OLS code encoders and syndrome computation has been suggested. The suggested methodology used the properties of OLS codes to design a parity prediction system that can be applied effectively and identifies any errors involving a single circuit node. The methodology was tested for various word types, which revealed that the overhead is minimal for big terms. That's also interesting since broad word sizes are used, for example, in caching for which OLS codes have currently been proposed.

The proposed error management scheme involved a considerable delay; nevertheless, its effect on access time can be reduced. This was done by testing in conjunction with the writing of the data in the case of the encoder and in parallel with the majority decision as well as the correction of both the mistake in the situation of the decoder. Throughout the general case, the suggested scheme needed a somewhat greater overhead, since most ECCs did not have the features of OLS codes. This restricted the applicability to OLS codes of both the proposed CED system. Through use of low capital error detecting strategies for encoder and syndrome computing is another justification to recommend the usage of OLS codes throughout high-speed memory and caches.

REFERENCES

[1] R. C. Baumann, "Radiation-induced soft errors in advanced semiconductor technologies," *IEEE Trans. Device Mater. Reliab.*, vol. 5, no. 3, pp. 301–316, Sep. 2005.

[2] M. A. Bajura, Y. Boulghassoul, R. Naseer, S. DasGupta, A. F.Witulski, J. Sondeen, S. D. Stansberry, J. Draper, L. W. Massengill, and J. N. Damoulakis, "Models and algorithmic limits for an ECC-based approach to hardening sub-100-nm SRAMs," *IEEE Trans. Nucl. Sci.*, vol. 54, no. 4, pp. 935–945, Aug. 2007.

[3] R. Naseer and J. Draper, "DEC ECC design to improve memory reliability in sub-100 nm technologies," *Proc. IEEE ICECS*, pp. 586–589, 2008.

[4] S. Ghosh and P. D. Lincoln, "Dynamic low-density parity check codes for fault tolerant nano-scale memory," presented at the Foundations Nanosci. (FNANO), Snowbird, Utah, 2007.

[5] S. Ghosh and P. D. Lincoln, "Low-density parity check codes for error correction in nanoscale memory," SRI Computer Science Lab., Menlo Park, CA, Tech. Rep. CSL-0703, 2007.

[6] H. Naeimi and A. DeHon, "Fault secure encoder and decoder for memory applications," in *Proc. IEEE Int. Symp. Defect Fault Toler.VLSI Syst.*, 2007, pp. 409–417.

[7] B. Vasic and S. K. Chilappagari, "An information theoretical framework for analysis and design of nanoscale fault-tolerant memories based on low-density parity-check codes," *IEEE Trans. Circuits Syst.I, Reg. Papers*, vol. 54, no. 11, pp. 2438–2446, Nov. 2007.

[8] H. Naeimi and A. DeHon, "Fault secure encoder and decoder for nanomemory applications," *IEEE Trans. Very Large Scale Integr.(VLSI) Syst.*, vol. 17, no. 4, pp. 473–486, Apr. 2009.

[9] S. Lin and D. J. Costello, *Error Control Coding*, 2nd ed. Englewood Cliffs, NJ: Prentice-Hall, 2004.

[10] S. Liu, P. Reviriego, and J. Maestro, "Efficient majority logic fault detection with difference-set codes for memory applications,"

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 21, Issue 2, 2025

IEEETrans. Very Large Scale Integr. (VLSI) Syst., vol. 20, no. 1, pp. 148–156, Jan. 2012.

[11] H. Tang, J. Xu, S. Lin, and K. A. S. Abdel-Ghaffar, "Codes on finite geometries," *IEEE Trans. Inf. Theory*, vol. 51, no. 2, pp. 572–596, Feb. 2005.

[12] P. P. Vaidyanathan, Multirate Systems and Filter Banks, Englewood Cliffs, N.J., USA: Prentice Hall, 1993.

[13] A. Sibille, C. Oestges and A. Zanella, MIMO: From Theory to Implementation, New York, NY, USA: Academic, 2010.

[14] N. Kanekawa, E. H. Ibe, T. Suga and Y. Uematsu, Dependability in Electronic Systems: Mitigation of Hardware Failures, Soft Errors, and ElectroMagnetic Disturbances, New York, NY, USA: Springer Verlag, 2010.

[15] M. Nicolaidis, "Design for soft error mitigation," IEEE Trans. Device Mater. Rel., vol. 5, no. 3, pp. 405–418, Sep. 2005.