IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 25 April 2024, accepted 10 May 2024, date of publication 17 May 2024, date of current version 24 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3402532

APPLIED RESEARCH

A Hybrid Technique Based on ECC and Hardened Cells for Tolerating Random Multiple-Bit Upsets in SRAM Arrays

DANIEL GIL-TOMÁS, LUIS J. SAIZ-ADALID, JOAQUÍN GRACIA-MORÁN, J. CARLOS BARAZA-CALVO, AND PEDRO J. GIL-VICENTE, (Member, IEEE)

Instituto ITACA, Universitat Politècnica de València, 46022 Valencia, Spain

Corresponding author: Joaquín Gracia-Morán (jgracia@itaca.upv.es)

This work was supported by the Ministerio de Ciencia e Innovación/Agencia Estatal de Investigación by project MCIN/AEI/10.13039/501100011033 under Grant PID2020-120271RB-I00.

ABSTRACT MBU is an increasing challenge in SRAM memory, due to the chip's large area of SRAM, and supply power scaling applied to reduce static consumption. Powerful ECCs can cope with random MBUs, but at the expense of complex encoding/decoding circuits, and high memory redundancy. Alternatively, radiation-hardened cell is an alternative technique that can mask single or even double node upsets in the same cell, but at the cost of increasing the overhead of the memory array. The idea of this work is to combine both techniques to take advantage of their respective strengths. To reduce redundancy and encoder/decoder overheads, SEC Hamming ECC has been chosen. About hardened cells, well-known and robust DICE cells, able to tolerate one node upset, have been used. To assess the proposed technique, we have measured the correction capability after a fault injection campaign, as well as the overhead (redundancy, area, power, and delay) of memory and encoding/decoding circuits. Results show high MBU correction coverages with an affordable overhead. For instance, for very harmful 8-bit random MBUs injected in the same memory word, more than 80% of the cases are corrected. Area overhead values of our proposal, measured with respect to double and triple error correction codes, are less than x1.45. To achieve the same correction coverage only with ECCs, redundancy, and overhead would be much higher.

INDEX TERMS Error correcting codes, random multiple-bit upsets, radiation-hardened cells, soft errors, static RAM.

I. INTRODUCTION

In current integrated circuit technologies, reliability has a growing importance due to progressive scaling, new materials and devices, and more demanding mission profiles [1]. SEUs (Single-Event Upsets) are a big reliability challenge in safety-critical fields and harsh environments, such as aerospace, autonomous vehicles, nuclear plants, and even DNN (Deep Neural Network) accelerators [2], [3], [4]. SEU is a change in the state of a storage element inside a device or system, provoked by high-energy cosmic particles, alpha particles from packaging materials, or electromagnetic interference. It may affect to a single bit or to multiple bits, manifesting as SBUs (Single-Bit Upsets) or MBUs (Multiple-Bit Upsets) [5], [6], [7], [8]. It has been observed that MBUs are increasing their impact due to the raised integration density of chips. These errors affect mainly to SRAM (Static RAM), as it is the most vulnerable component of the chip due to its large area and the trend to scale the supply voltage to diminish leakage power [9].

Usually, it is considered that MBUs affect adjacent bits. To tolerate them, techniques based on the combination of interleaving and ECCs (Error Correction Codes) (mainly Single-Error Correcting (SEC)-Hamming) are applied [10], [11]. However, the reduction of the technology size has given rise to new patterns in soft errors that affect the rows and columns of the array. Thus, the affected cells are not always adjacent. Also, different cluster shapes have

The associate editor coordinating the review of this manuscript and approving it for publication was Luca Cassano.

© 2024 The Authors. This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

been observed [12], [13], [14]. Therefore, it is expected that random MBUs will have an increasing incidence in memory arrays.

Powerful ECCs can cope with random MBUs, but at the expense of complex encoder/decoder circuits and high memory redundancy. In addition, complex ECCs may require extra cycles to decode data, which can result in performance penalty [9], [16].

On the other hand, hardened cells are another technique used to tolerate SEUs. They allow masking SNUs (Single-Node Upsets) or even DNUs (Double-Node Upsets) in the same cell, at the expense of approximately doubling the number of transistors [17], [18]. The advantage is that no encoding/decoding circuits are needed. Main drawbacks are the cell overhead and the limited ability to tolerate MNUs (Multi-Node Upsets) in the same cell, although some designs have been proposed to mitigate these problems [19], [20]. Internal MNUs that are not tolerated manifest as SBUs or MBUs in the array of bits.

The idea of this work is to propose a hybrid approach for tolerating random MBUs, combining ECCs with hardened cells, in order to take advantage of their respective strengths. As far as the authors know, this hybrid technique has not been considered before. Some other hybrid techniques has been found in the literature, although they present a high overhead and difficulties in tolerating some cases of MBUs. For instance, [15] combines parity per byte and duplication.

In this way, the objective of this paper is to achieve the following improvements with respect to a pure ECC approach:

i) Higher error correction coverage against random MBUs.
ii) Higher speed in the access to memory.
iii) Lower encoding/decoding overhead, maintaining redundancy at an acceptable level.

To reduce redundancy and encoder/decoder overheads, a simple ECC such as an SEC Hamming code [21] has been chosen. For the hardened cell, the DICE (Dual Interlocked Storage Cell) cell [17] has been used. It is one of the best-known SEU hardened cell. It is robust and it shows an acceptable leakage consumption [9]. As pointed above, there exist other cells capable of tolerating two-node upsets, but they entail an additional increase in the number of transistors and memory overhead [16].

To evaluate the performance of our proposal, we have carried out a comparative analysis with some typical ECCs that are used in SRAM to tolerate ramdom MBUs. We have chosen several DEC (Double Error Correction) codes:matrix-based CLC (Column Line Code), OLS (Orthogonal Latin Square code), BCH (Bose–Chaudhuri–Hocquenghem code) [22], [23], [24], [25], and a TEC (Triple Error Correction) code [26].

To assess the proposed technique, we have measured the correction coverage by using fault injection, as well as the overheads of memory and encoding/decoding circuits.

The paper is organized as follows. Section II describes the main scheme of our proposal and its components. We also make a theoretical comparison with some pure ECC approaches. Section III includes the experimental assessment of the proposed technique. We have measured the correction capability after a fault injection campaign, as well as the overhead (redundancy, area, power, and delay) of memory and encoding/decoding circuits. In Section IV, several future improvements to the proposed design have been suggested, to further increase the tolerance against MBUs, keeping the overhead within affordable limits. Finally, Section V presents some conclusions.

II. HYBRID TECHNIQUE PROPOSED

A. OVERALL DESCRIPTION

The proposed technique combines SEC Hamming ECC with DICE hardened cells. Fig. 1 shows this hybrid proposal. A memory word consists of k data bits and c code redundant bits. Data bits are stored in DICE hardened cells, and code bits are in standard 6T (i.e., composed of 6 transistors) SRAM cells. As explained, code bits are generated according to the SEC Hamming ECC.

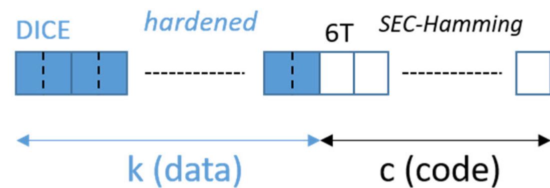

FIGURE 1. Hybrid proposal. Structure of a memory word.

SEC Hamming is a simple, low-redundant ECC, able to correct 1-bit error in the codeword. DICE is a well-known hardened cell, capable of tolerating a single node upset in the cell.

The idea is to strengthen the data bits with the hardened cells and, in case of error, correct it with SEC-Hamming.

Next, we summarize the main characteristics of SEC Hamming and DICE hardened cell, as well as the correction capability of the proposed technique.

1) SELECTED ECC: SEC HAMMING

Single Error Correction (SEC) Hamming codes [21] can correct an erroneous bit with simple and fast encoding and decoding operations, and the lowest redundancy. A linear block code, like Hamming, can be described by its parity check matrix **H** [27]. It is an $(n-k) \times n$ binary matrix, where n represents the number of bits of the codeword, and k is the number of data bits. The matrix for a Hamming code must accomplish two requirements:

- All columns must be nonzero.
- All columns must be different.

To correct any single-bit error in the codeword, the number of redundant bits must satisfy the condition: $2^c \geqslant k+c+1$, being $c = n-k$ the number of redundant bits. When $2^c = k + c + 1$, the code is known as perfect. Therefore, perfect Hamming codes only exist for codeword lengths that are a power of 2 minus one: (7, 4), (15, 11), (31, 26), (63, 57), etc., where the pairs represent (n, k).

Nevertheless, Hamming codes can be easily built for any word length by shortening a longer code.

Shortening is a technique that allows creating new codes from existing ones, reducing data bits but maintaining the same number of parity bits. In this way, the new code maintains the same error coverage properties. In practice, shortening is performed by eliminating columns from the original parity check matrix.

In this work, we have applied our scheme to 32-bit data words (i.e. $k = 32$). The perfect Hamming code with $k \geqslant 32$ is the (63, 57) code, so $c = 6$. Shortening this code, the (38, 32) Hamming code can be obtained. The columns to eliminate can be selected under different criteria. Frequently, the eliminated columns are those that represent the highest values or those with higher Hamming weight (thus allowing simpler operations). In our proposal, we have eliminated the columns with Hamming weight equal to 2 (i.e., with two 1s). The reason will be explained later, as it is a key point of our proposal.

2) SELECTED HARDENED CELL: DICE (DUAL INTERLOCKED STORAGE CELL)

Fig. 2 shows the schematic of DICE cell, built with transistors. DICE is a worthwhile candidate for rad-hard operation in modern technologies with scaled supply voltages [9]. A summary of the characteristics of the cell are [17]:

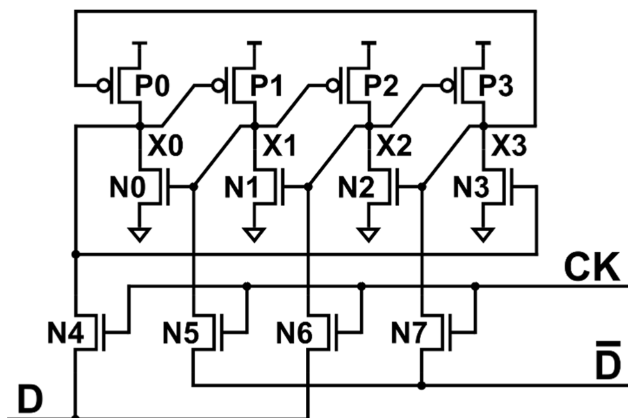

FIGURE 2. DICE hardened cell [17].

- It is suitable for static RAM cells using submicron CMOS Technology.
- It is a 12T cell (i.e. it is built with 12 transistors).
- It relies on the principle of dual node feedback control. The four nodes X0-X3 store the data as two pairs of complementary values (X0-X2 and X1-X3). Transistors form two opposite feedback loops, a clockwise PMOS loop, P0-P3, and an anti-clockwise NMOS loop, N3-N0. In this way, it is capable to tolerate one soft error at any sensitive node.
- The area is approximately twice that of cell 6T.
- If two sensitive nodes that store the same logic state (X0-X2 or X1-X3) are flipped simultaneously, the cell is upset. Nevertheless, the probability of this event occurrence can be reduced if the critical nodes are spaced on the cell's layout.

3) CORRECTION CAPABILITY AND REDUNDANCY

SEUs are soft errors, that is, no hardware damage happens, and it is correctable by changing the state of the storage element back to its original value. Single-Bit Upset means a single storage location upset from a single particle strike. Multiple-Bit Upset means multiple upsets in a logical word from a single particle strike [5]. Taking this into account, we can analyze the correction capability of our proposal from Fig. 1. SNUs in data cells are masked by the DICE cell intrinsic redundancy, while an SNU in a code cell becomes a single error (SBU) that can be corrected by the Hamming code. Therefore, our proposal tolerates all SBUs.

Regarding MBUs, 2-node upsets may affect in different ways:

- Both upsets affect the same data cell: two nodes are flipped, and the cell becomes erroneous (not really in all cases, only if the two nodes store the same logic state); as all other cells are correct, it manifests as a single error that can be corrected by the Hamming code. Notice that hardened memory cells can be viewed as composed of 2 single cells, due to their redundant structure. From this point of view, a 2-node upset can be considered equivalent to a 2-bit upset in the same hardened cell.
- Both upsets affect different data cells: as each upset affects only one of the nodes of the DICE cell, both cells mask the effect of the fault, and no error is activated.
- One upset affects a data cell and the other one affects a code cell: the first one is masked by the DICE cell, so only the error in the code cell is activated; it is a single error that can be corrected by the Hamming code.
- Both upsets affect code bits: in this case, a double error is manifested. It is known that Hamming codes cannot correct double errors. Anyway, data bits are all correct. If the decoding process preserves the integrity of these bits, the result will be correct. To do so, syndromes for double errors in code bits must not match with syndromes for single errors in data bits. This can be achieved by avoiding columns with Hamming weight two in the parity check matrix. Using this criterion when shortening the perfect code, a double error in code bits does not affect data bits, leading to the right decoding. Thus, we have applied it in our proposal.

Therefore, our proposal also tolerates all 2-node upsets. This model allows cell (electronic) and array (logical) domains to be processed uniformly. In addition, this model will facilitate the process of fault injection, as will be seen in Section III (Experimental Assessment).

In the same way, all 3-node upsets that affect only data cells are corrected (in the worst case, two upsets in the same cell activate a single error, that can be corrected by the Hamming code). In our model of two single cells for each hardened cell, this can be considered equivalent to a 3-bit upset.

Obviously, 3-bit upsets that occur in code cells cannot be tolerated, because code cells don't have redundancy.

MBUs are generally tolerated if each upset affects a different data cell. Only when MBU upsets two or more data cells (or three or more code cells), SEC-Hamming leads to a wrong decoding.

In summary, the proposed technique is capable of tolerating:

- All single errors.
- All double errors.
- All triple errors in data cells.
- All MBUs ($\geqslant 4$) in independent data cells.

Thus, the proposed technique predicts a good capability to cope with MBUs, mainly in data cells.

Notice that MBUs ($\geqslant 3$) in code cells or upsets affecting data and code cells can provoke miss-corrections of data bits. This can be solved by using more powerful ECC codes, or hardened cells in the c bits, but at the expense of higher redundancy and memory overhead. In this work, we have studied the version with less overhead, using the SEC Hamming code with 6T cells for the c redundant bits. In Section IV (Potential Improvements), we analyze some proposals to strengthen the code bits.

About the redundancy of our proposal:

$$\text{Redundancy} = \frac{\textit{Redundant bits}}{N^{\circ}\ \textit{data bits}} \approx \frac{k+c}{k} \tag{1}$$

We have assumed that the area of the DICE cell is approximately twice the area of the 6T cell, considering that the number of transistors is double (12T). This implies that data cells present a 100% of redundancy, that is, we can assume that for k data bits, we use another k redundant bits.

From (1), the redundancy of our proposal is higher than 100%. For $k \gg c$, redundancy approaches the 100%. In this proposal, the objective has been to reduce c as much as possible, using a very simple ECC.

B. COMPARISON WITH PURE ECCs

Next, we will compare our proposal with some typical ECCs usually used in SRAMs to tolerate multiple errors. We have selected several DEC (Double Error Correction) codes, such as matrix-based CLC [22], OLS (high speed) [23], BCH (low redundancy) [24], and LRRO (Low Redundancy and Reduced Overhead) [25], an improvement of BCH; and a TEC (Triple Error Correction) code [26], named LR TEC. Both LRRO and LR TEC have been designed by our research group.

Table 1 shows the redundancy and the correction capabilities of these ECCs for 32-bit codewords. SEC Hamming has also been included.

Replacing in eq. (1) $k = 32$ and $c = 6$, the redundancy of our proposal is:

$$\textit{Redundancy} \approx \frac{32+6}{32} = 1.1875 = 118.75\%$$

Although the redundancy of our proposal is higher in all cases, the overhead due to the encoding/decoding circuits is reduced by using a simple code like SEC Hamming.

TABLE 1. Redundancy and correction capability.

Code	Redundancy (%)	Correction capability
SEC Hamming (38,32)	18.75	SEC
CLC (65,32)	103	DEC
OLS (55,32)	72	DEC
BCH (44,32)	37.5	DEC
LRRO (44,32)	37.5	DEC
LR TEC (52,32)	62.5	TEC
Our proposal	118.75	TEC in data cells, n-EC in independent data cells, n≥4

Regarding the correction capability, it is observed that our proposal can improve the results for large MBUs. This would be suitable for harsh environments or in safety-critical applications.

Next, we present some estimates that we have calculated about area, power, and delay overheads when comparing our proposal (hybrid design) with the designs based only on ECCs.

1) AREA OVERHEAD ESTIMATION RELATIVE TO ECCs

We define area overhead as:

$$OV_A = \frac{A'_{mem} + A'_{ECC}}{A_{mem} + A_{ECC}} \tag{2}$$

where A_{mem} is the memory area; A_{ECC} is the encoder/decoder area; A'_{mem} is the memory area of our proposal; and A'_{ECC} is the encoder/decoder area of our proposal.

We assume that $A'_{ECC} < A_{ECC}$, because the idea is to use a simple ECC together with hardened cells. The different memory areas can be calculated as:

$$A_{mem} = N \times (k+c) \tag{3}$$

$$A'_{mem} \approx N \times (2k + c') \tag{4}$$

where N is the number of words in memory, k is the number of data bits, c is the number of redundant bits, and c' is the number of redundant bits of our proposal. $c' < c$ by deliberately using a very simple ECC. We consider the DICE cell to be about twice the size of 6T cell.

In the typical case that memory size contributes much more significantly than encoder/decoder to the total area, $A_{mem} \gg A_{ECC}$ and $A'_{mem} \gg A'_{ECC}$. Then,

$$OV_A \approx \frac{A'_{mem}}{A_{mem}} = \frac{2k + c'}{k+c} = \frac{2 + (r'-1)}{1+r} \tag{5}$$

where r is the redundancy of the ECC, defined by Eq. 6, and r' is the redundancy of our proposal, calculated by Eq. 7.

$$r = \frac{c}{k} \tag{6}$$

$$r' = \frac{k+c'}{k} = 1 + \frac{c'}{k} \tag{7}$$

Table 2 shows the values of OV_A by applying Eq. (5) to each ECC, using the values of r and r' of Table 1. In the worst case, comparing with the simplest ECC (SEC Hamming),

TABLE 2. Memory area overhead relative to ECCs.

Code	OV_A
SEC Hamming (38,32)	1.84
CLC (65,32)	1.08
OLS (55,32)	1.27
BCH (44,32)	1.59
LRRO (44,32)	1.59
LR TEC (52,32)	1.35

the memory area overhead is 1.84. If we compare with DEC and TEC codes, in all cases the overhead is below 1.6, and even in some cases (CLC code), our proposal is similar.

In the general case, when A_{ECC} is not negligible with respect to A_{mem}, then an even better result can be expected because $A'_{ECC} < A_{ECC}$:

$$OV_A = \frac{A'_{mem} + A'_{ECC}}{A_{mem} + A_{ECC}} \approx \frac{\frac{A'_{mem}}{A_{mem}}}{1 + \frac{A_{ECC}}{A_{mem}}} < \frac{A'_{mem}}{A_{mem}} \quad (8)$$

Hardened cells can be applied selectively in certain critical areas (register bank, L1-cache, ...). The smaller the memory size, the better overhead is obtained, as the "hardened" part contributes less to the overhead.

2) POWER OVERHEAD ESTIMATION RELATIVE TO ECCs

Considering that power consumption is proportional to the number of transistors, a similar trend to the area overhead is expected. That is, a value lower than 2x. In Section III (Experimental Assessment), power consumption (static + dynamic) of memory cell and encoder/decoder circuits has been measured. The hardened cell shows a slightly higher value of 2x than the non-hardened 6T cell. In the ECC decoder, our proposal shows important reductions in the power overhead.

3) DELAY OVERHEAD ESTIMATION RELATIVE TO ECCs

We must consider the following times:

- Read (memory cell) + ECC decoder.
- Write (memory cell) + ECC encoder.

We define the delay overhead as:

$$OV_D = \frac{D'_{mem} + D'_{ECC}}{D_{mem} + D_{ECC}} \quad (9)$$

where D_{mem} is the memory delay (read or write); D_{ECC} is the ECC delay (encoder or decoder); D'_{mem} is the memory delay of our proposal; and D'_{ECC} is the ECC delay of our proposal.

In some cases, ECC can require extra cycles to verify the data. If we assume that D_{mem}, $D'_{mem} \ll D_{ECC}$, then

$$OV_D \approx \frac{D'_{ECC}}{D_{ECC}} < 1 \quad (10)$$

Our proposal would improve speed by using a very simple ECC. The delays measured in Section III corroborate this prediction.

In the general case, considering that the delay of the hardened cell is not negligible, the prediction is slightly worse:

$$OV_D = \frac{D'_{mem} + D'_{ECC}}{D_{ECC}} = \frac{D'_{mem}}{D_{ECC}} + \frac{D'_{ECC}}{D_{ECC}} \quad (11)$$

III. EXPERIMENTAL ASSESSMENT

To assess the proposed technique, we have calculated the following parameters:

a) Error correction coverage.
b) Memory cell overhead.
c) Encoder/decoder overhead.

Overhead includes three aspects: delay, area, and power.

A. FAULT INJECTION

Fault injection is a well-known experimental technique used to assess the tolerance of a system against the occurrence of faults [28]. About the type of faults to inject, the more representative fault models must be selected. As said before, SEUs are the most common in SRAM cells, particularly SBUs (Single-Bit Upsets), and increasingly frequent MBUs (Multiple-Bit Upsets) [3], [5]. MBUs can be adjacent or random, the latter with a growing impact. Clusters of affected cells with different shapes, not always adjacent, have been observed [12], [13].

To study the error correction coverage of our proposal, we have used a simulator that we have developed to inject different types of errors [29]. Fig. 3 shows the basic scheme. Comparing the input and output words, the simulator can check if the error injected leads to a right or a wrong decoding. Repeating the process for all errors of a given size, it is possible to count the number of corrected errors with respect to the total number of injected errors. That is, it is possible to calculate the error correction coverage.

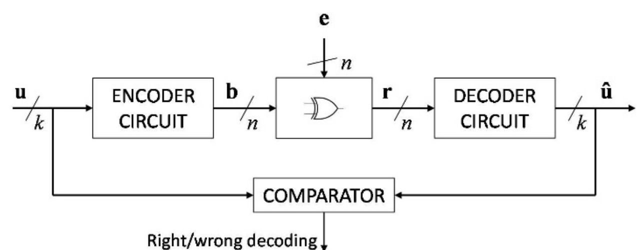

FIGURE 3. Block diagram of the fault injection simulator [29].

In this work, we have injected single errors (size = 1), as well as multiple random errors with sizes varying from 2 to 8. This represents SBUs and MBUs with different sizes. We consider that 8 random errors in the same memory word is an enough high value, taking into account the values observed in some radiation test experiments on real SRAM arrays [12], [13], [30].

We must remark that we have not injected errors according to their probability of occurrence, as our goal is to measure the error correction coverage of each system. Specifically, we have injected each type of fault randomly in all bits of the

TABLE 3. MBU error correction coverage.

MBU size	Proposal	Hamming	BCH DEC	LRRO DEC	OLS DEC	CLC	LR TEC
1	100%	100%	100%	100%	100%	100%	100%
2	100%	0.3%	100%	100%	100%	100%	100%
3	99.6%	0.1%	3.1%	2.4%	44.4%	91.7%	100%
4	98.5%	0.0%	0.2%	0.2%	12.9%	68.3%	13.8%
5	96,4%	0.0%	0.0%	0.0%	3.0%	37.7%	1.4%
6	92.9%	0.0%	0.0%	0.0%	0.5%	15.4%	0.2%
7	88.0%	0.0%	0.0%	0.0%	0.1%	5.7%	0.0%
8	81.8%	0.0%	0.0%	0.0%	0.0%	1.9%	0.0%

codeword to verify the error correction capability. Obviously, the probability of occurrence decreases as the size of the MBU increases, but the objective is to test the proposed technique against MBUs of different sizes, even the most harmful ones.

All blocks of the fault injection tool have been developed in C, using the bitwise logic operators for an accurate simulation of the hardware behavior. Encoder and decoder circuits can be easily obtained from the parity-check matrix that defines the ECC. These circuits are implemented in C as encoding and decoding functions. Changing the simulator for a different ECC is as simple as adjusting the word lengths and replacing the encoding and decoding functions for the new ECC.

The error correction coverage percentage has been calculated as:

$$C_{correc} = \frac{Errors_Corrected}{Errors_Injected} \times 100 \quad (12)$$

where *Errors_Corrected* is the number of errors corrected by the ECC, and *Errors_Injected* is the total amount of errors injected for a given error size.

To simplify the fault injection, hardened memory cells have been viewed as composed of 2 single cells. As stated above, the circuitry required for a hardened cell has two pairs of complementary nodes, so the intrinsic redundancy is two. Single (i.e., size = 1) errors are tolerated, but size = 2 errors can change the value stored in a hardened cell. The codeword length for our proposals has been considered accordingly. For example, Hamming (38, 32) code has a 38-bit codeword and can be stored in 38 standard memory cells, that can be affected by 38 different single-bit upsets. But, if data bits are stored in 32 hardened cells and parity bits are stored in normal 6T cells (our proposal), 70 different single-bit upsets (that is, $32 \times 2 + 6$) may affect the storage of a codeword. This is a model to ease fault injection at logical level. The injected SBUs are internal SNUs.

B. MBU CORRECTION COVERAGE

Fig. 4 and Table 3 show the results of the fault injection campaign. Error correction coverage has been calculated according to Eq. (12).

As commented before, injected errors can be single (SBU) or multiple (MBU). The 6T cell cannot tolerate any SBU. The DICE cell can tolerate SBUs. In case of two (or more) faults in the same DICE cell, the cell is upset.

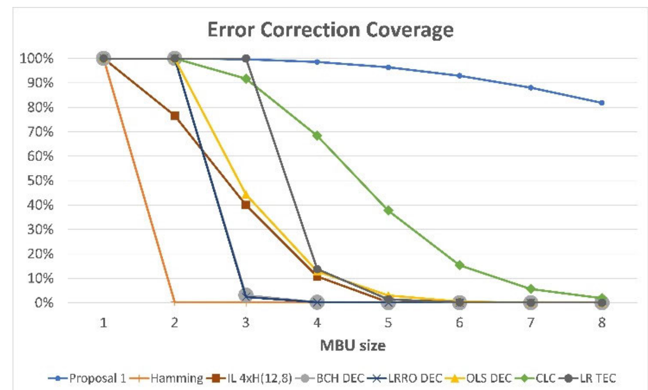

FIGURE 4. MBU correction coverage.

It is observed that SEC, DEC, and TEC ECCs drastically degrade their performance after 1, 2, and 3 errors, respectively, while our proposal significantly improves the results for large MBUs. Coverages over 90% are observed for 6-bit MBUs, and over 80% for 8-bit MBUs.

To achieve these results with pure ECCs, redundancy should be very high, and consequently, the related overhead of the memory and the ECC circuits will also raise a lot. For instance, the number of redundant bits for $k = 32$ in BCH codes (which achieve the minimum redundancy) required to tolerate MBUs of size 6, 7, and 8, are 42, 49, and 56 bits [27], respectively. This results in redundancy values of 131.25%, 153.125%, and 175%, well above 118.75% of our proposal (see Eq. (1)).

In addition, the circuits (encoder/decoder) of the ECC are much more complex than ours, which are based on the SEC Hamming code.

Note that another common hybrid technique has also been included in the figure: interleaving combined with SEC-Hamming. IL 4xH(12,8) means that the codeword is divided into 4 groups of 8 interleaved bits, so that each group is protected by a different SEC-Hamming (12,8). In this way, the interleaving distance is 4. This technique shows poor results compared to our proposal. The reason is that the interleaving technique is focused on tolerating adjacent MBUs, and its effectiveness is quickly degraded against random MBUs.

These results confirm the usefulness of our proposal for critical systems, or for those systems working in

harsh environments, where reliability is paramount, and random MBUs are expected.

C. HARDENED CELL DESIGN

Here, we intend to measure the overhead of the hardened cell with respect to standard 6T SRAM cell. We have used Microwind layout editor [31] to make a full-custom design of the cells.

6T cell layout has been extracted from the editor's library, with a feature size of 250 nm. It is the minimum available size in Microwind 3.5 Lite, the free version used. Although large MBUs are not very critical for such "old" technology, the main objective of this section is to compare the overhead of 6T and DICE cells, using the same technology. Results can be generalized to more recent technologies, where an increasing impact of MBUs is expected.

Layout of 6T cell is shown in Fig. 5, which corresponds to the transistor's schematic shown in Fig. 6. Data and /Data signals of Fig. 5 correspond to Q and /Q of Fig. 6. WL is the Word Line signal, that selects the cells of a word during read or write operations. The measured area of the 6T cell is about 25 μm^2.

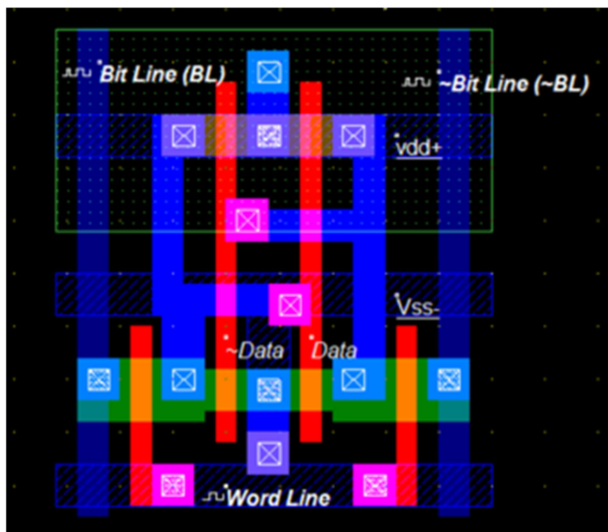

FIGURE 5. Layout of 6T SRAM cell.

As the library of the layout editor does not contain the DICE cell, and we have not found libraries with free DICE cells, we have designed the layout ourselves. Starting from the diagram with transistors of Fig. 2, we have generated the Logic Graph of Fig. 7, intending to obtain a layout with a "line of diffusion" structure (an unbroken row of transistors in which abutting source/drain connections are made). This is commonly used for standard cells in automated layout systems [32]. It is observed that the graph does not admit Euler paths (paths through all the nodes of the graph, such that each arc is visited only once) for the Pull-Up Network (PUN) and the Pull-Down Network (PDN), so the layout presents diffusion breaks, as it can be seen in the derived stick-diagram

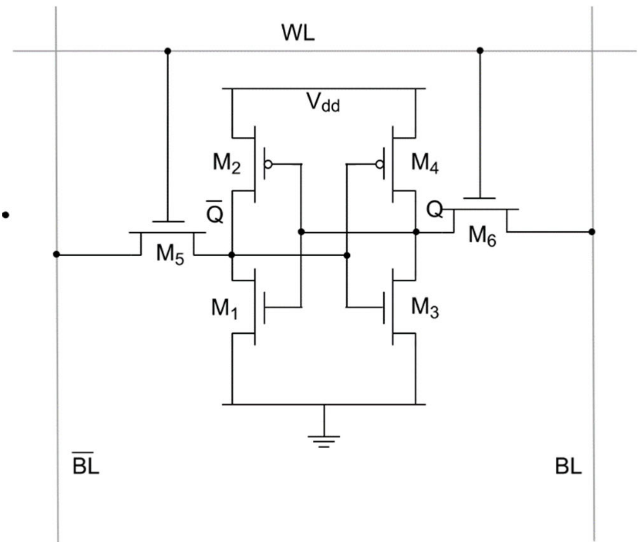

FIGURE 6. Transistors scheme of 6T SRAM cell.

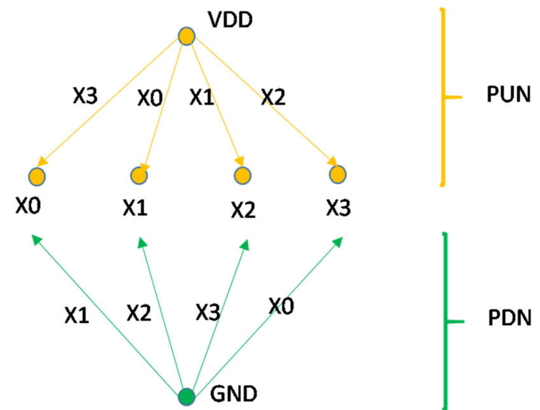

FIGURE 7. Logic graph of DICE hardened cell.

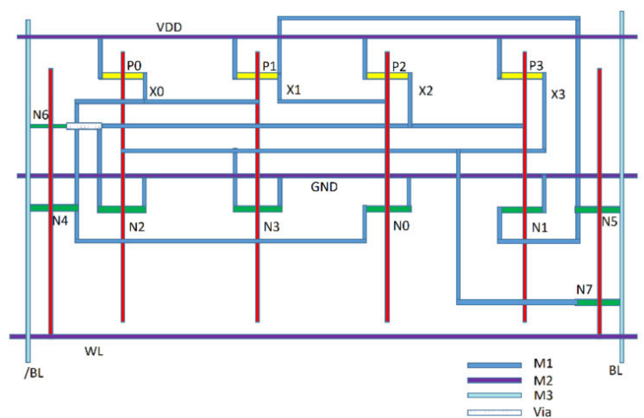

FIGURE 8. Stick diagram of DICE hardened cell.

of Fig. 8. M1, M2, and M3 refer to metal layers. N is the N-diffusion and P is the P-diffusion. BL and /BL are the bit line and its complementary, and WL is the word line that selects the cell to read and write.

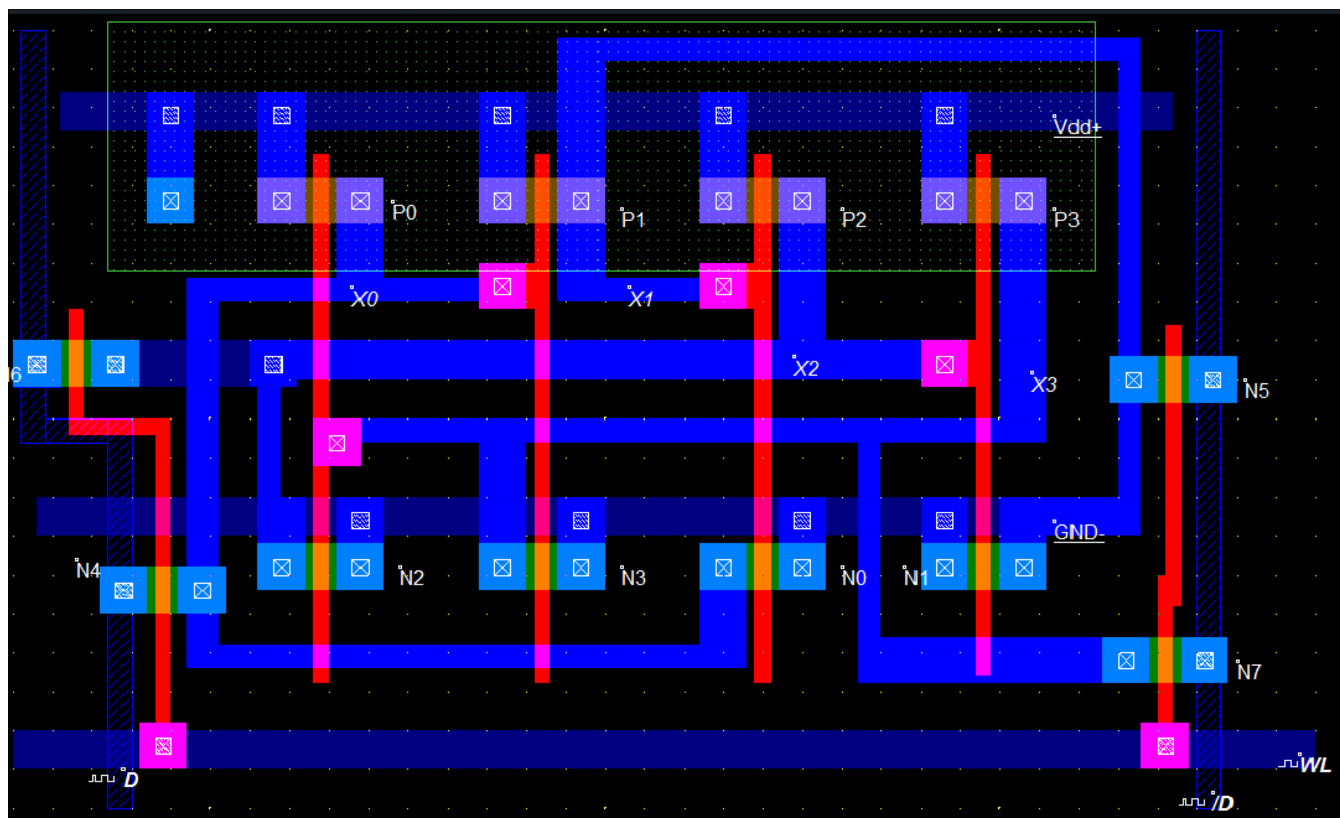

FIGURE 9. Layout of DICE hardened cell.

The final layout is shown in Fig. 9. It uses λ-based scalable CMOS design rules, with $\lambda = 0.125\ \mu$m, half of the feature size (250 nm) and $V_{DD} = 2.5$ V. The measured area of the DICE cell is about 50 μm^2, twice the area of the 6T cell. This is an expected result, as the number of transistors is twice.

D. MEMORY CELL OVERHEAD

The area of DICE cells is twice the area of the 6T cells, as we have verified in the layout editor.

Regarding delay and power, Fig. 10 shows the chronogram of a 1→0 write operation in the 6T cell. The delay between the WL signal activation and the change in the internal Data signal is 24 ps. The measured power consumption is 20.706 μW.

Fig. 11 shows the same operation, but in the DICE cell. The delay between the activation of the WL signal and the change in the internal X0 signal is 48 ps. The measured power consumption is 46.84 μW. The delay between WL and X1 (the complementary value of the stored data) is a bit higher (64.8 ps, see Fig. 12), due to the positive feedback process of the memory cell.

If we compare the DICE and the 6T cells, DICE cell presents approximately double values for area, delay and consumption. This is consistent with the fact that the number of transistors is double.

E. ECC ENCODER/DECODER OVERHEAD

We have synthesized encoder and decoder circuits for all ECCs. To do this, we have implemented them in VHDL, and using CADENCE software [33], we have carried out a logic synthesis for 250 nm technology (the same feature size as memory cells) by using the Oklahoma State University System on Chip (SoC) Design Flows [34]. Standard cells are based on SCMOS (MOSIS scalable) design rules. Power voltage and temperature conditions are 2.5 V and 25° C, respectively. Although 250 nm is not a very modern technology, the main goal of this section is to compare the overhead of encoders and decoders using the same technology.

Tables 4 and 5 show the encoder and decoder overheads (area, delay, and power) of the different ECCs implemented. Remember that our proposal uses SEC Hamming. From the tables, Fig. 13 to 15 have been generated. These graphs compare the area, power, and delay of the different ECCs with respect to the SEC Hamming. They represent the quotient between the value of each ECC with respect to the value of the SEC Hamming (our proposal). Accordingly, our proposal is normalized to 1. Each graph shows two values, related to the encoder and the decoder.

As LR TEC presents a much higher area and power overhead than the rest of the ECCs (due to its bigger correction coverage), Fig. 13 and 14 have been represented with a vertical logarithmic scale. In this way, the values of DEC ECCs are

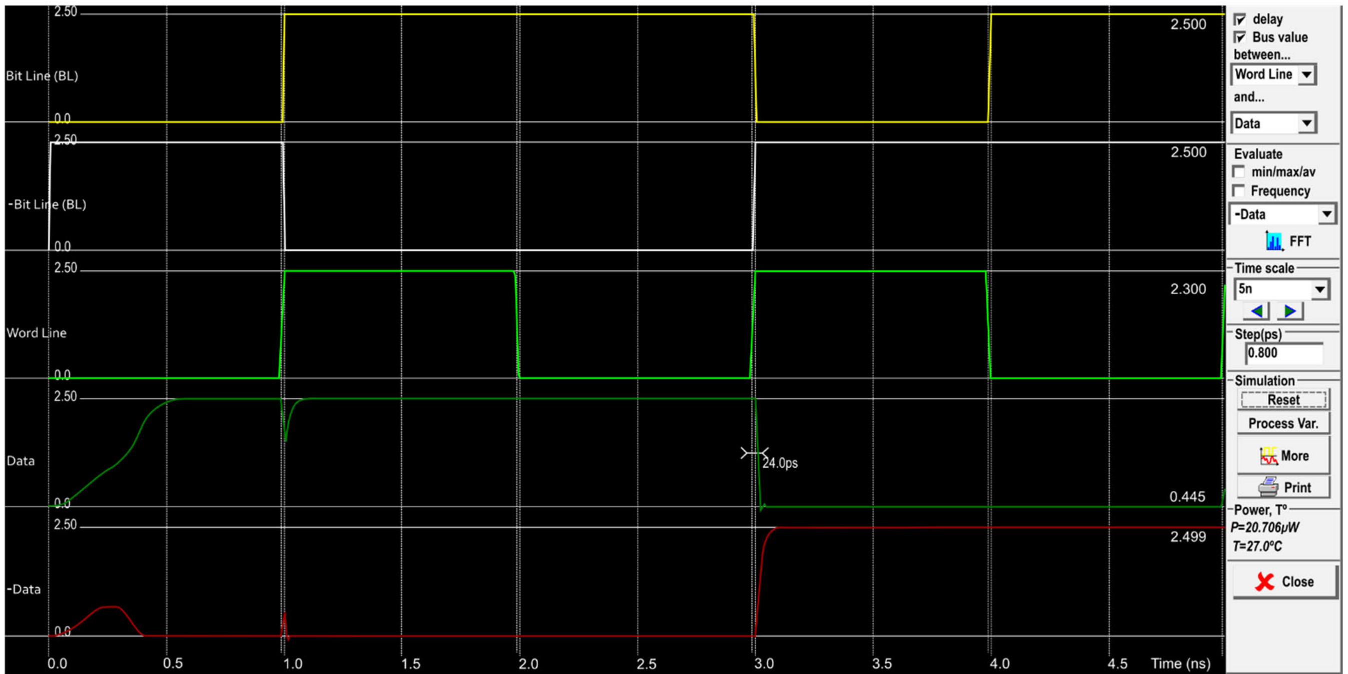

FIGURE 10. 6T SRAM cell. Power consumption and time delay (WL→Data) of a write operation.

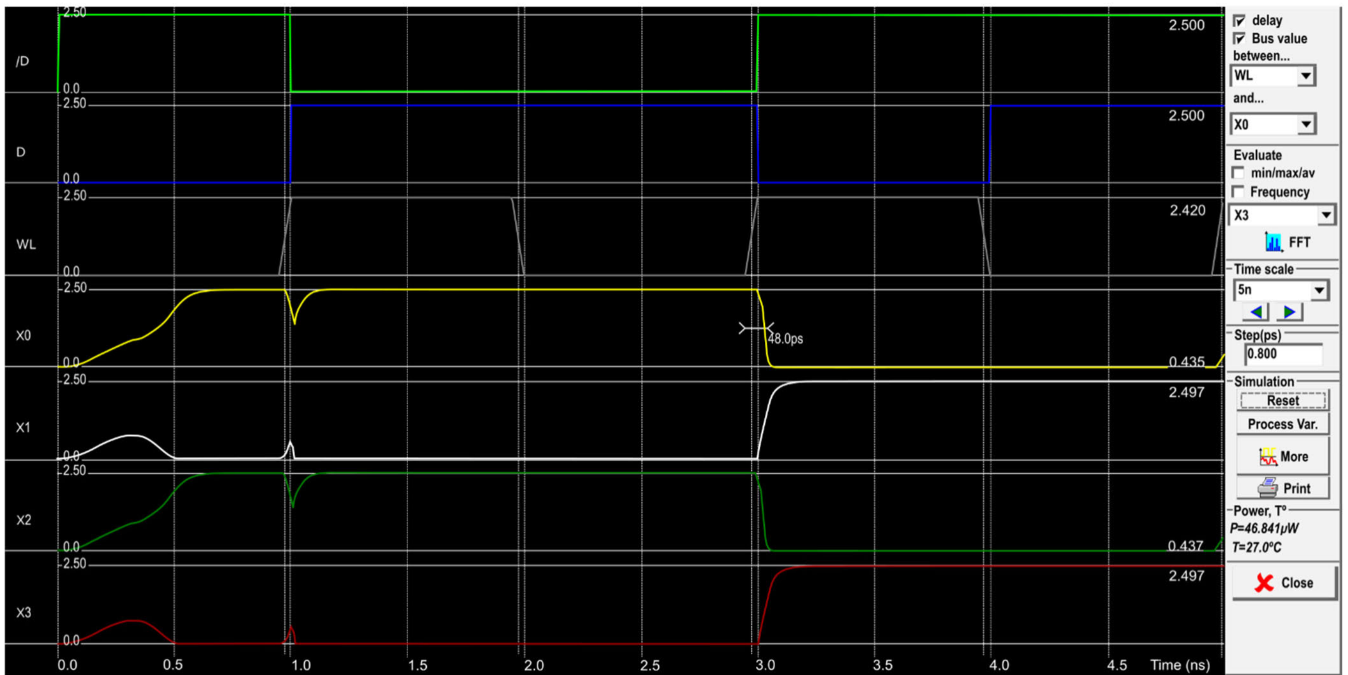

FIGURE 11. DICE hardened cell. Power consumption and time delay (WL→X0) of a write operation.

better appreciated. Fig. 15 has a linear scale, as the difference between TEC and the other codes isn't so big.

As expected, decoders' overhead values are bigger than encoders ones, because decoder circuits are more complex.

In general, we can observe that our proposal improves the overhead values, as the proportion $\frac{Overhead\ ECC}{Overhead\ Our\ Proposal}$ is in most cases bigger than 1:

- Fig. 13 shows significant improvements in the area overhead, especially on decoder circuits. Our proposal improves TEC code by more than 140 times (TEC overhead is excessive). Enhancement over DEC codes varies (approximately) from x1.5 to x11. DEC codes with the highest overheads are BCH and LRRO, which are the ones with the lowest redundancy in memory. In other

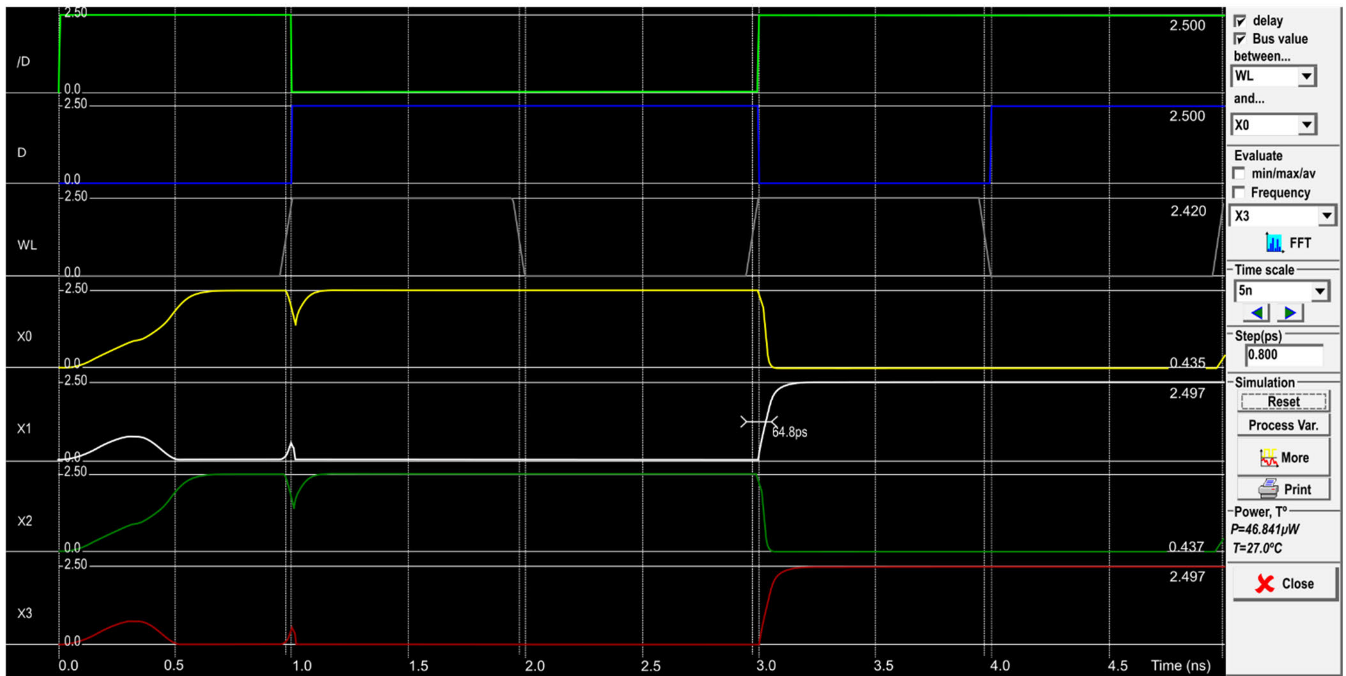

FIGURE 12. DICE hardened cell. Power consumption and time delay (WL→X1) of a write operation.

TABLE 4. Encoder overhead (Area, Delay, Power).

ECC	Area (µm²)	Delay (ps)	Power (nW)		
			Static	Dynamic	Total
SEC Hamming (38, 32)	11585	1789	11,324	2234333,161	2234344,485
BCH DEC (44, 32)	19504	1817	19,947	4067448,846	4067468,793
OLS DEC (55, 32)	19783	604	21,611	2288762,575	2288784,186
LR TEC (52, 32)	23313	1542	25,016	4032574,619	4032599,635
CLC (65, 32)	17672	1663	17,575	3215988,189	3216005,764
LRRO DEC (44, 32)	17003	1719	17,197	3067532,816	3067550,013

TABLE 5. Decoder overhead (Area, Delay, Power).

ECC	Area (µm²)	Delay (ps)	Power (nW)		
			Static	Dynamic	Total
SEC Hamming (38, 32)	25229	2542	24,156	4712099,873	4712124,029
BCH DEC (44, 32)	299495	8142	171,949	71190359,277	71190531,226
OLS DEC (55, 32)	49515	2004	44,157	11517667,538	11517711,695
LR TEC (52, 32)	3655962	10859	1951,689	810223800,505	810225752,194
CLC (65, 32)	42394	3202	41,564	7149043,066	7149084,630
LRRO DEC (44, 32)	284580	6960	164,135	60254644,926	60254809,061

words, a lower redundancy implies a greater complexity of the ECC circuits.

- Power overhead (Fig. 14) follows a similar trend with respect to the area, which is quite expected, because the power depends largely on the number of transistors.
- Improvements in the delay are also observed in Fig. 15, especially in the decoders, the most complex circuits. Values up to x4 have been obtained. The exception is OLS code, which presents the lowest delay.

F. SUMMARY OF THE ASSESSMENT RESULTS

In summary, the results of our proposal show:

- High MBU correction coverage (over 80% for 8-bit MBUs), with an affordable overhead.
- DICE overhead is about 2x with respect to 6T cell.
- Significant improvements have been made with respect to the overhead of ECC encoder/decoder circuitry, especially in decoder circuits, which are the most complex.
- Regarding the delay overhead, although the delay of the hardened cell is approximately double of 6T cell,

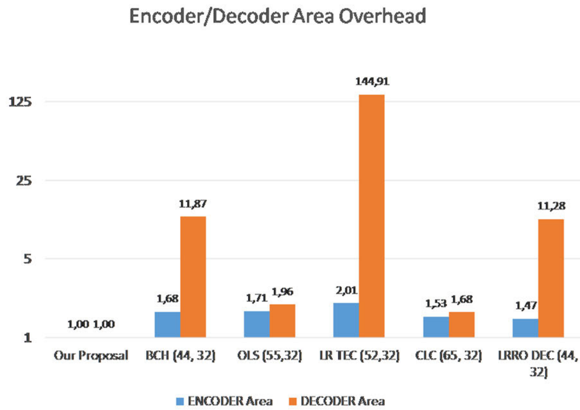

FIGURE 13. Area overhead relative to our proposal.

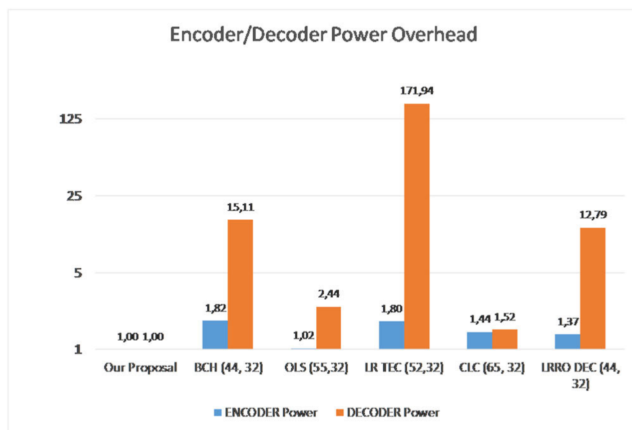

FIGURE 14. Power overhead relative to our proposal.

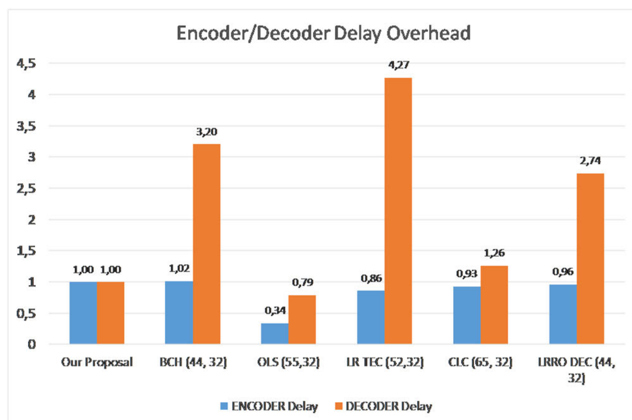

FIGURE 15. Delay overhead relative to our proposal.

the decoder of the SEC used in our hybrid scheme improves delay by more than x3. This would be potentially more evident for complex ECC with software decoding.

IV. POTENTIAL IMPROVEMENTS

In this work, we have combined DICE hardened cells with SEC Hamming code. The use of such a simple code has allowed the reduction of redundancy and overhead while maintaining a high error coverage. However, redundant bits are still the most sensitive to MBUs. As said in Section II, MBUs in code cells can cause mis-corrections of data bits. So, in this section we propose some extensions of our first proposal to strengthen the code bits: i) using hardened cells in the code bits too; ii) using more powerful ECC codes. In any case, this implies more memory redundancy and ECC overhead. Let's see in more detail some of these alternative proposals in the following points. These ideas can be synthesized in future works.

A. PROPOSAL 2: DICE+SEC USING HARDENED CELLS IN ALL BITS

This proposal consists of combining DICE and SEC Hamming, in the same way as in proposal 1 (the original approach), but using hardened cells in both data and parity bits. Fig. 16 shows the scheme of the memory codeword.

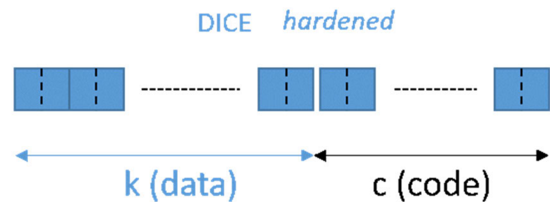

FIGURE 16. Proposal 2: structure of a memory word.

The approximate redundancy of this proposal would be:

$$Redundancy \approx \frac{k + 2c}{k} \tag{13}$$

We have assumed that the size of the hardened cells is twice that of a 6T cell. So, for $k = 32$ and SEC Hamming,

$$Redundancy \approx \frac{32 + 2 \times 6}{32} \times 100 = 137.5\% \tag{14}$$

Compared to the original proposal (Redundancy = 118.75%), it does not seem to represent a large increase, if we notice that the encoder/decoder overhead remains the same, and the correction capability improves notably. In fact, Proposal 2 can correct (the expected enhancements with respect to Proposal 1 have been remarked in bold):

- All single errors.
- All double errors.
- **All triple errors.**
- **All MBUs ($\geqslant$4) in independent cells.**

Fig. 17 shows the results of the fault injection campaign, comparing with the original approach.

Table 6 shows the improvement of Proposal 2 with respect to Proposal 1. It is observed that with 8 random errors, the most harmful injected MBU in our experiments, it maintains a correction coverage greater than 96%. This means a very high capability to tolerate large MBUs.

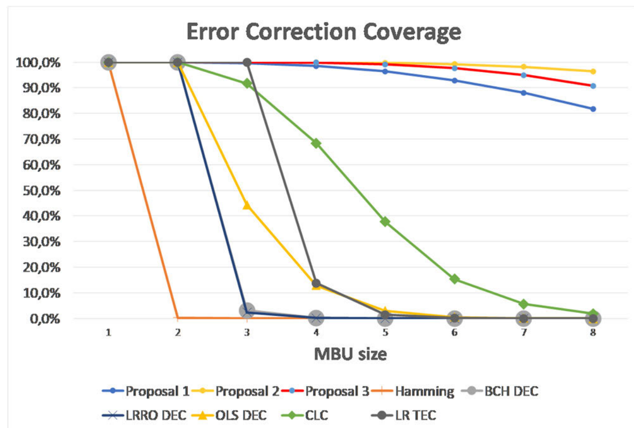

FIGURE 17. MBU correction coverage with some proposals for improvement.

TABLE 6. MBU error correction coverage of proposal 1 and proposal 2.

MBU size	Proposal 1	Proposal 2
1	100%	100%
2	100%	100%
3	99.6%	100%
4	98.5%	99.9%
5	96.4%	99.7%
6	92.9%	99.2%
7	88.0%	98.2%
8	81.8%	96.5%

B. PROPOSAL 3: DICE + DEC

The idea is to combine now DICE cells (only in data bits) with a DEC code, to strengthen the c (code) bits. Fig. 18 shows the scheme of the codeword.

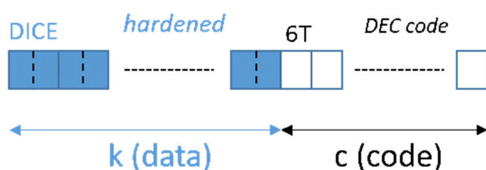

FIGURE 18. Proposal 3: structure of a memory word.

In this case, the approximate redundancy is:

$$Redundancy \approx \frac{k+c}{k} \tag{15}$$

For $k = 32$ and BCH or LRRO (the DEC codes with the lowest redundancy in Table 1),

$$Redundancy \approx \frac{32+12}{32} \times 100 = 137.5\% \tag{16}$$

The same redundancy is observed in Proposal 2. But, in the case of Proposal 3, the overhead of the decoder is much higher (see Fig. 13 to 15), as BCH and LRRO are more complex codes than SEC Hamming.

Regarding the expected correction capability, Proposal 3 can correct (the expected enhancements with respect to Proposal 1 have been remarked in bold):

- All single errors.
- All double errors.
- **All triple errors.**
- **All 4-errors in data cells.**
- **All 5-errors in data cells.**
- **All MBUs (>5) in independent data cells.**

In the initial proposal, Hamming code was selected to avoid double errors in code bits leading to a wrong decoding. Similarly, the DEC code for this proposal has been designed to avoid triple errors in code bits that modify erroneously a data bit. In this way, when triple errors occur in the redundant code bits, the integrity of the data bits is maintained.

Compared with Proposal 2, we can see that the k data bit part is more robust, but the c code bit part is weaker. Overall, Proposal 2 is better than Proposal 3, as shown in Fig. 17 and Table 7.

TABLE 7. MBU error correction coverage of proposal 1, proposal 2 and proposal 3.

MBU size	Proposal 1	Proposal 2	Proposal 3
1	100%	100%	100%
2	100%	100%	100%
3	99.6%	100%	100%
4	98.5%	99.9%	99.8%
5	96.4%	99.7%	99.2%
6	92.9%	99.2%	97.6%
7	88.0%	98.2%	94.9%
8	81.8%	96.5%	90.8%

C. PROPOSAL 4: DICE + TEC

Finally, Fig. 19 shows Proposal 4, which consists of combining hardened cells (in data bits) with a TEC code.

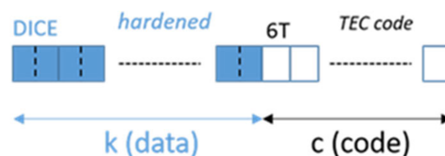

FIGURE 19. Proposal 4: structure of a memory word.

The approximate redundancy is:

$$Redundancy \approx \frac{k+c}{k} \tag{17}$$

For $k = 32$ and LR TEC (see Table 1),

$$Redundancy \approx \frac{32+20}{32} \times 100 = 162.5\% \tag{18}$$

As expected, redundancy is greater, as LR TEC is a more powerful ECC. In addition, the overhead of the decoder is excessive, as can be seen in Fig. 13 to 15.

Regarding the expected error correction capability, Proposal 4 can correct (the expected enhancements with respect to Proposal 1 have been highlighted in bold):

- All single errors.
- All double errors.
- **All triple errors.**
- **All 4-errors.**
- **All 5-errors in data cells.**
- **All 6-errors in data cells.**
- **All 7-errors in data cells.**
- **All MBUs (>7) in independent data cells.**

Again, TEC code has been designed to preserve the integrity of data bits when quadruple errors occur in the code bits, in the same way as SEC (Hamming) and DEC codes employed in proposals 1 and 3, respectively.

As we can see, correction capability is very high, mainly for data cells, but at the cost of an intractable overhead. Fig. 20 compares the four proposals in the range from 90% to 100% of error correction coverage to better see the differences between them. It is observed that Proposal 4 is very similar to Proposal 2, although the overhead is much higher.

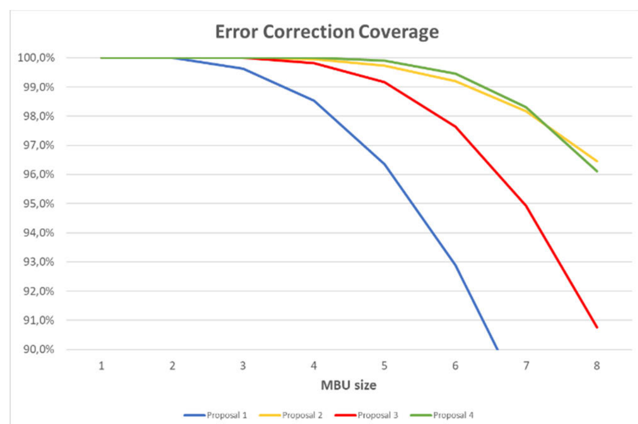

FIGURE 20. Comparison of the error correction coverage of Proposals 1, 2, 3, and 4.

Summarizing, among all the proposed improvements, we think that Proposal 2 achieves the best tradeoff between error correction coverage and overhead.

V. CONCLUSION

In this paper, we have proposed a novel scheme to strengthen SRAM memory against MBUs (Multiple-Bit Upsets). It has been observed that MBUs can be an important issue in integrated circuit technologies that operate in harsh environments or in safety-critical applications. Our proposal is a hybrid design that combines hardened cells with ECCs, two techniques that are usually applied separately.

We have assessed the proposed design, measuring the error correction coverage and the overhead (redundancy, area, power, and delay) of memory and encoder/decoder circuits of the ECC.

Results show high MBU correction coverages with an acceptable overhead. For instance, for 8-bit random upsets injected in the same codeword, more than 80% of the cases are corrected, with area overhead values of our proposal (using an SEC Hamming ECC) lower than x1.45 when compared to DEC and TEC codes. To achieve the same correction coverage values only with ECCs, redundancy and overhead would be much higher.

Finally, we suggest several variations of the proposed design, in order to increase the tolerance against MBUs, keeping the overhead within affordable limits. The study of the improvement suggested concludes that it is better the use of hardened cells to protect the whole codeword using an SEC code, rather than using a stronger generic ECC while maintaining the code bits in non-hardened cells.

REFERENCES

[1] (2022). *International Roadmap for Devices and Systems (IRDS)*. [Online]. Available: https://irds.ieee.org/editions/2022

[2] S. K. Kurinec and K. Iniewski, *Nanoscale Semiconductor Memories: Technology and Application*. Boca Raton, FL, USA: CRC Press, 2014.

[3] J. Barak, M. Murat, and A. Akkerman, "SEU due to electrons in silicon devices with nanometric sensitive volumes and small critical charge," *Nucl. Instrum. Methods Phys. Res. B, Beam Interact. Mater. At.*, vol. 287, pp. 113–119, Sep. 2012.

[4] A. Azizimazreah, Y. Gu, X. Gu, and L. Chen, "Tolerating soft errors in deep learning accelerators with reliable on-chip memory designs," in *Proc. IEEE Int. Conf. Netw., Archit. Storage (NAS)*, Chongping, China, Oct. 2018, pp. 1–10.

[5] *Introduction to Single-Event Upsets*, document WP-01206-1.0, Altera, San Jose, CA, USA, Sep. 2013. [Online]. Available: https://cdrdv2-public.intel.com/650466/wp-01206-introduction-single-event-upsets.pdf

[6] G. Tsiligiannis et al., "Multiple cell upset classification in commercial SRAMs," *IEEE Trans. Nucl. Sci.*, vol. 61, no. 4, pp. 1747–1754, Aug. 2014, doi: 10.1109/TNS.2014.2313742.

[7] G. I. Zebrev, K. S. Zemtsov, R. G. Useinov, M. S. Gorbunov, V. V. Emeliyanov, and A. I. Ozerov, "Multiple cell upset cross-section uncertainty in nanoscale memories: Microdosimetric approach," in *Proc. 15th Eur. Conf. Radiat. Effects Compon. Syst. (RADECS)*, Sep. 2015, pp. 1–5.

[8] N. Chechenin and M. Sajid, "Multiple cell upsets rate estimation for 65 nm SRAM bit-cell in space radiation environment," in *Proc. 3rd Intl. Conf. Exhib. Satell. Space Missions*, May 2017, p. 77.

[9] L. Atias, A. Teman, and A. Fish, "Single event upset mitigation in low power SRAM design," in *Proc. IEEE 28th Conv. Electr. Electron. Eng. Israel (IEEEI)*, Dec. 2014, pp. 1–5.

[10] C. W. Slayman, "Cache and memory error detection, correction, and reduction techniques for terrestrial servers and workstations," *IEEE Trans. Device Mater. Rel.*, vol. 5, no. 3, pp. 397–404, Sep. 2005.

[11] S. Baeg, S. Wen, and R. Wong, "SRAM interleaving distance selection with a soft error failure model," *IEEE Trans. Nucl. Sci.*, vol. 56, no. 4, pp. 2111–2118, Aug. 2009.

[12] D. Radaelli, H. Puchner, S. Wong, and S. Daniel, "Investigation of multi-bit upsets in a 150 nm technology SRAM device," *IEEE Trans. Nucl. Sci.*, vol. 52, no. 6, pp. 2433–2437, Dec. 2005.

[13] G. Georgakos, P. Huber, M. Ostermayr, E. Amirante, and F. Ruckerbauer, "Investigation of increased multi-bit failure rate due to neutron induced SEU in advanced embedded SRAMs," in *Proc. IEEE Symp. VLSI Circuits*, Kyoto, Japan, 2007, pp. 80–81, doi: 10.1109/VLSIC.2007.4342774.

[14] M. Maniatakos, M. K. Michael, and Y. Makris, "Vulnerability-based interleaving for multi-bit upset (MBU) protection in modern microprocessors," in *Proc. IEEE Int. Test Conf.*, Nov. 2012, pp. 1–8.

[15] R. C. Goerl, P. R. C. Villa, L. B. Poehls, E. A. Bezerra, and F. L. Vargas, "An efficient EDAC approach for handling multiple bit upsets in memory array," *Microelectron. Rel.*, vols. 88–90, pp. 214–218, Sep. 2018.

[16] R. Rajaei, B. Asgari, M. Tabandeh, and M. Fazeli, "Design of robust SRAM cells against single-event multiple effects for nanometer technologies," *IEEE Trans. Device Mater. Rel.*, vol. 15, no. 3, pp. 429–436, Sep. 2015.

[17] T. Calin, M. Nicolaidis, and R. Velazco, "Upset hardened memory design for submicron CMOS technology," *IEEE Trans. Nucl. Sci.*, vol. 43, no. 6, pp. 2874–2878, Dec. 1996.

[18] J. Guo, L. Zhu, W. Liu, H. Huang, S. Liu, T. Wang, L. Xiao, and Z. Mao, "Novel radiation-hardened-by-design (RHBD) 12T memory cell for aerospace applications in nanoscale CMOS technology," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 25, no. 5, pp. 1593–1600, May 2017.

[19] A. Yan, J. Zhou, Y. Hu, J. Cui, Z. Huang, P. Girard, and X. Wen, "Novel quadruple cross-coupled memory cell designs with protection against single event upsets and double-node upsets," *IEEE Access*, vol. 7, pp. 176188–176196, 2019.

[20] A. Yan, Y. Hu, J. Cui, Z. Chen, Z. Huang, T. Ni, P. Girard, and X. Wen, "Information assurance through redundant design: A novel TNU error-resilient latch for harsh radiation environment," *IEEE Trans. Comput.*, vol. 69, no. 6, pp. 789–799, Jun. 2020.

[21] R. W. Hamming, "Error detecting and error correcting codes," *Bell Syst. Tech. J.*, vol. 29, no. 2, pp. 147–160, Apr. 1950.

[22] F. Silva, A. Muniz, J. Silveira, and C. Marcon, "CLC-A: An adaptive implementation of the column line code (CLC) ECC," in *Proc. 33rd Symp. Integr. Circuits Syst. Design (SBCCI)*, Campinas, Brazil, Aug. 2020, pp. 1–6, doi: 10.1109/SBCCI50935.2020.9189901.

[23] S. Liu, L. Xiao, and Z. Mao, "Extend orthogonal Latin square codes for 32-bit data protection in memory applications," *Microelectron. Rel.*, vol. 63, pp. 278–283, Aug. 2016.

[24] R. Naseer and J. Draper, "DEC ECC design to improve memory reliability in sub-100nm technologies," in *Proc. 15th IEEE Int. Conf. Electron., Circuits Syst.*, Federated optimization in heterogeneous networks, Malta, Aug. 2008, pp. 586–589.

[25] L.-J. Saiz-Adalid, J. Gracia-Morán, D. Gil-Tomás, J.-C. Baraza-Calvo, and P.-J. Gil-Vicente, "Reducing the overhead of BCH codes: New double error correction codes," *Electronics*, vol. 9, no. 11, p. 1897, Nov. 2020.

[26] J. Gracia-Morán, L. J. Saiz-Adalid, J. C. Baraza-Calvo, D. Gil-Tomás, and P. J. Gil-Vicente, "Tolerating double and triple random errors with low redundancy error correction codes," in *Proc. Workshop Innov. Inf. Commun. Technol. (WIICT)*, Valencia, Spain, Jul. 2022, pp. 16–30.

[27] S. Lin and D. J. Costello, *Error Control Coding*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2004.

[28] A. Benso and P. Prinetto, *Fault Injection Techniques and Tools for VLSI Reliability Evaluation*. Boston, MA, USA: Kluwer Academic, 2003.

[29] J. Gracia-Morán, L. J. Saiz-Adalid, D. Gil-Tomás, and P. J. Gil-Vicente, "Improving error correction codes for multiple-cell upsets in space applications," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 10, pp. 2132–2142, Oct. 2018.

[30] B. L. Bhuva, N. Tam, L. W. Massengill, D. Ball, I. Chatterjee, M. McCurdy, and M. L. Alles, "Multi-cell soft errors at advanced technology nodes," *IEEE Trans. Nucl. Sci.*, vol. 62, no. 6, pp. 2585–2591, Dec. 2015.

[31] *Microwind*. Accessed: Apr. 23, 2024. [Online]. Available: https://microwind.net/

[32] N. Weste and D. Harris, *CMOS VLSI Design. A Circuits and System Perspective*. Reading, MA, USA: Addison-Wesley, 2005.

[33] *Cadence: EDA Tools and IP for System Design Enablement*. Accessed: Apr. 23, 2024. [Online]. Available: https://www.cadence.com

[34] *Oklahoma State University System on Chip (SoC) Design Flows*. Accessed: Apr. 23, 2024. [Online]. Available: https://vlsiarch.ecen.okstate.edu/flow/

DANIEL GIL-TOMÁS received the B.Sc. degree in electrical and electronic physics from Universitat de València, Spain, in 1985, and the Ph.D. degree in computer engineering from Universitat Politècnica de València (UPV), Spain, in 1999. He is currently an Associate Professor with the Department of Computer Engineering (DISCA), UPV. He is also a member of fault-tolerant systems (STF) research line with the Institute ITACA, UPV. His research interests include the design and validation of fault-tolerant systems, reliability physics, and reliability of emerging nanotechnologies.

LUIS J. SAIZ-ADALID received the M.Sc. and Ph.D. degrees in computer engineering from Universitat Politècnica de València (UPV), Spain, in 1995 and 2015, respectively.

After 15 years in the industry (IBM, in 1995, and Celestica, in 1998), he is currently an Associate Professor with the Department of Computer Engineering (DISCA), UPV. He is also a member of fault-tolerant systems (STF) research line with the Institute ITACA, UPV. His research interests include the design and implementation of digital systems, the design and validation of fault-tolerant systems, and the design of error control codes.

JOAQUÍN GRACIA-MORÁN received the B.Sc., M.Sc., and Ph.D. degrees in computer engineering from Universitat Politècnica de València (UPV), Spain, in 1995, 1997, and 2004, respectively. He is currently an Associate Professor with the Department of Computer Engineering (DISCA), UPV. He is also a member of fault-tolerant systems (STF) research line with the Institute ITACA, UPV. His research interests include the design and implementation of digital systems, design and validation of fault-tolerant systems, and VHDL-based fault injection.

J. CARLOS BARAZA-CALVO received the B.Sc. and Ph.D. degrees in computer engineering from Universitat Politècnica de València (UPV), Spain, in 1993 and 2003, respectively. He is currently an Associate Professor with the Department of Computer Engineering (DISCA). He is also a member of the fault-tolerant systems (STF) research line with the Institute ITACA, UPV. His research interests include the design and implementation of digital systems, the design and validation of fault-tolerant systems, and VHDL-based fault injection.

PEDRO J. GIL-VICENTE (Member, IEEE) received the B.Sc. degree in electronic engineering from Universitat Politècnica de Catalunya (UPC), Tarragona, Spain, in 1979, and the M.Sc. degree in industrial engineering and the Ph.D. degree in computer engineering from Universitat Politècnica de València (UPV), Spain, in 1985 and 1992, respectively.

He is currently a Professor and the Head of the Department of Computer Engineering (DISCA), UPV. He is also the Head of the fault-tolerant systems (STF) research line with the Institute ITACA, UPV. His research interests include the design and validation of real-time fault-tolerant distributed systems, the dependability validation using fault injection, the design and verification of embedded systems, and the dependability and security benchmarking. He has authored more than 150 research articles on these subjects. He has served as a Program Committee Member for the IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), the European Dependable Computing Conference (EDCC), and the Latin American Symposium on Dependable Computing (LADC).