Microelectronics Reliability 139 (2022) 114826

Contents lists available at ScienceDirect

Microelectronics Reliability

journal homepage: www.elsevier.com/locate/microrel

Review paper

A survey on two-dimensional Error Correction Codes applied to fault-tolerant systems

David Freitas [a,*], César Marcon [b], Jarbas Silveira [a], Lirida Naviner [c], João Mota [a]

[a] *Federal Institute of Ceará, Fortaleza, CE, Brazil*
[b] *Pontifical Catholic University of Rio Grande do Sul, Porto Alegre, RS, Brazil*
[c] *LTCI, Télécom Paris, Institut Polytechnique de Paris, 91128 Palaiseau, France*

ARTICLE INFO

Keywords:
Error Correction Code (ECC)
2D-ECC
Reliability
Fault tolerance

ABSTRACT

The number of memory faults operating in radiation environments increases with the electronic device miniaturization. One-dimensional (1D) Error Correction Codes (ECCs) are not efficient in mitigating these problems requiring two-dimensional (2D)-ECCs for providing superior error correction capacity with proportionally less energy and area consumption. The significant increase in publications in this area demands a study to guide and subsidize research decisions, mainly to determine a standardization method for comparing and evaluating ECCs. We propose a Systematic Literature Review (SLR) to investigate the most important features of 2D-ECCs used for mitigating faults in memories. This SLR revealed the most used ECCs, data size and redundancy overhead, encoder and decoder implementation technology, fault injection methods, and evaluation metrics. Besides, we extracted some ECC trends, such as reusing the encoder inside the decoder and targeting the three-dimensional (3D)-ECC to increase the error correction efficacy. The experimental results describe important research decisions of great value for this scientific community.

1. Introduction

Requirements such as reducing energy consumption and increasing the processing capacity of specific applications, such as those installed in space systems, have driven the integration of many devices with reliable operation in high radiation environments. Due to the limited space operation resources, it is essential to design these systems efficiently in area and energy consumptions. Additionally, recent Integrated Circuit (IC) manufacturing technologies increase the susceptibility to radiation-induced errors, often imposing the manufacture of these systems with older and/or robust process technology [1,2]. This radiation susceptibility is a crucial factor to be considered since space missions place a high demand on memory, needing to operate without errors that can cause a system malfunction [3].

Operation at low voltage levels is of great importance for space applications, where energy resources are limited. Current low-cost satellites have an even smaller energy budget, as the total weight of the satellite is often reduced due to the restriction on using heavy batteries and power supplies. The most efficient way to achieve low voltage is to operate all the chip components in a region close to the threshold, reducing static and dynamic energy consumption significantly. However, ICs designed to operate at low voltage levels are much more susceptible to electromagnetic radiation effects than those operating at conventional voltage levels. Radiation-induced errors have been studied since the 1960s, as shown in the studies [4–6]. Alpha particles and neutrons collisions affect memory reliability; when they reach the silicon substrate, the voltage level of the memory cell changes; as these particles create charge carriers, increasing the number of errors [7,8].

This Systematic Literature Review (SLR) focuses on transient errors, also known as soft errors, caused by voltage fluctuations or electromagnetic radiation effects that flips the content of the memory cells; conversely, permanent errors, also known as hard errors, are caused by the effects during the manufacturing process or large amounts of radiation that physically damage the memory cells [9–12].

A soft error is a subset of a Single Event Effect (SEE) [13] classified into (i) Single Bit Upset (SBU) - when occurs a single event bitflip in a single cell [14,15]; (ii) Multiple Cell Upset (MCU) – when a single event changes two or more memory cells [16–20]; (iii) Multiple Bit Upset (MBU) - when a single event flips the content of two or more cells in the same word [23]; (iv) Single Event Transient (SET) - when a single event

[*] Corresponding author.
E-mail addresses: davidciarlinifreitas@gmail.com (D. Freitas), cesar.marcon@pucrs.br (C. Marcon), lirida.naviner@telecom-paris.fr (L. Naviner), mota@gtel.ufc.br (J. Mota).

https://doi.org/10.1016/j.microrel.2022.114826
Received 22 March 2022; Received in revised form 12 September 2022; Accepted 8 October 2022
Available online 19 October 2022

causes a voltage failure in a circuit [24–27]; (v) Single Event Functional Interrupt (SEFI) - when a single event causes functionality loss due to disturbance of registers, clocks, reset, etc. [28–30]; and (vi) Single Event Latch-up (SEL) - when a single event causes a non-normal high current and requires a power reset [31–34]. Single Event Upset (SEU) is also used as a synonym for SBU or MBU [13].

Error Correction Code (ECC) is the technique most used to mitigate memory failures; it was initially proposed by Hamming [35] to avoid errors on a relay-based machine at Bell Labs. The ECC basis adds check bits to a set of data bits to perform a codeword for detecting and correcting errors using Boolean logic relations, usually XOR operations. This technique requires an encoder to calculate the check bits and associate them to the data bits producing a codeword; a decoder analyzes the codeword, detecting and correcting possible errors [36–38].

Device miniaturization increasingly leads to MCU [39,40] that many times one-dimensional (1D)-ECCs are inefficacious to correct [41]. Thus, n-dimensional codes have been used, mainly two-dimensional (2D)-ECCs like product codes, because they provide higher detection and correction power with proportionally less energy and area consumption [41].

This work carries out an SLR to consolidate and synthesize the evidence found on 2D-ECCs for mitigating memory faults. Section 2 presents all the SLR planning and conduction phases to capture the most relevant works in the 2D-ECC area, considering the last seven years. Section 3 describes 2D-ECC fundamentals and the advantages of two-dimensional encoding over one-dimensional encoding. SLR enabled us to analyze and classify the selected works according to the organization of bits, redundancy type and size, number of data bits, type of target application, and analysis methods in Sections 4 to 5. Therefore, enabling us to investigate the 2D-ECC trends presented and discussed in Section 8. Finally, Section 9 concludes this work by highlighting some gaps we captured in this research topic.

To the best of our knowledge, no similar work has been published. It is intended to present the methods and ways of illustrating results so that comparisons among ECCs are made fairly. We also conduct an in-depth, state-of-the-art analysis to identify the main concepts and issues addressed.

2. Systematic Literature Review (SLR)

An SLR is a study that standardizes the entire review process, excluding bias and exposing reliable conclusions [42]. This systematic review is based on [42–46], whose methodology provides consistent means of answering research questions objectively and impartially.

The recent growth in the number of papers in the field of ECCs indicates the need to synthesize evidence found in an in-depth analysis of the state of the art. Identifying the main concepts and issues addressed allows for consolidating and standardizing the proposed methods and making fair comparisons among the ECC proposals.

There is no single method for implementing an effective SLR. The research shows that an SLR usually covers planning, conducting, and summarizing phases subdivided into other sub-phases. The planning encompasses the research objective and SLR employed protocol. The conducting identifies and selects studies based on planning and performs data extraction and synthesis. Finally, the summary phase reports and evaluates the synthesized data. This section presents the five subphases comprising the planning and conducting of this SLR.

2.1. Research objective

This SLR investigates 2D-ECCs developed between 2015 and 2022. This section displays the main aspects of the investigated works, such as the ECC organization, target application, and error analysis methods. These aspects are linked to the research questions discussed next.

2.2. Research question (RQ)

This SLR development contemplates the primary research objective with the following research questions (RQs).

RQ1. What are the most used 2D-ECCs?

Besides identifying important ECCs, this RQ seeks to explain the reason for using 2D-ECCs and allows classifying them and their targeting applications.

RQ2. How have the ECC techniques been verified/tested?

The answer to this question shows how the techniques have been validated, for example, concerning memory size (this is directly linked to the redundancy bits) and whether there is any additional technique beyond the ECC.

RQ3. What works are employed to compare to other ECCs?

This RQ allows researchers to define methods used to compare ECCs and choose the best one for each given scenario.

2.3. Selection criteria

Table 1 describes the selection criteria used in the search process to define the main requirements of the selected papers, avoiding the need for complete work analysis. Note that the inclusion criterion is the topic to meet the research objective - 2D-ECC papers to mitigate memory errors. The exclusion criteria specify papers published before 2015, not available in English, duplicated, and do not meet the inclusion criterion.

2.4. Search process

Fig. 1 shows the search process that we established based on the Snowballing Forward (SF) technique [47,48], followed by an automatic search in five bibliographic databases.

2.4.1. Snowballing Forward (SF)

SF selects new papers based on the ones that cite the paper being examined [47–50]. One possibility for an initial set is to identify seminal papers with various citations in a systematic review. We found three papers from the same group that proposed the Matrix code; together, these three papers have 109 citations, respectively, 22, 24, and 63 [51–53]; therefore, set C_1 encompasses the 109 works that cite Matrix. The search for these papers was carried out on Google Scholar, using key terms and the number of citations.

We examined the title, summary, year, and parts of the text of all papers of set C_1, applying inclusion and exclusion criteria to build set C_2. The next step checks all papers cited in C_2, followed by the same type of analysis performed in C_1 for creating set C_3, and so on. Since the SF process did not return new papers, we completed the search process in set C_5, containing 24 papers related to the research objective.

2.4.2. Automatic search

This step starts by creating the string used for automatic search in the

Table 1
Selection criteria.

Inclusion criterium	
IC-1	Paper presenting 2D-ECC to mitigate memory errors

Exclusion criterium	
EC-1	Published before 2015
EC-2	Not available in English
EC-3	Duplicated work
EC-4	Paper does not meet the IC-1 criterium

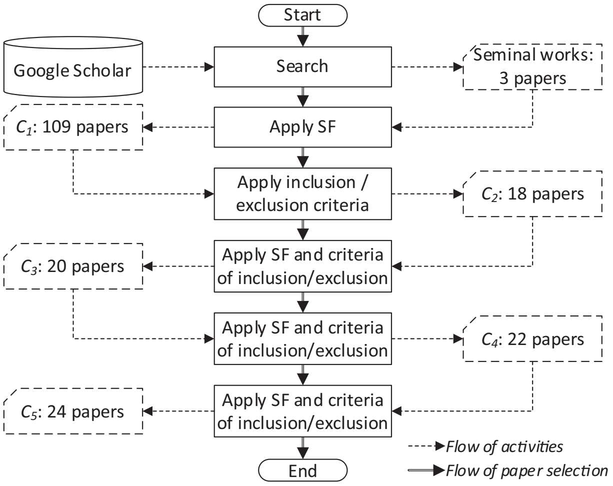


Fig. 1. Papers selection process using the SF technique along with inclusion and exclusion criteria; these steps selected 24 papers.

Institute of Electrical and Electronics Engineers (IEEE) Xplore Digital Library, Elsevier Scopus, Association for Computing Machinery (ACM) Digital Library, American Scientific Publishers (ASP), and Web of Science databases encompassing reputed journals and conferences in science and technology. We carried out this research between October 2020 and March 2022, restricting to the exclusion criteria of Table 1.

The search string created based on the keywords most used in the 24 papers of C_5 is: *("error correction" OR "error detection" OR "ECC" OR "EDAC") AND ("MCU" OR "MBU" OR "upset" OR "multiple-bit upset" OR "multiple cell upset ") AND ("matrix" OR "product code") AND ("memory" OR "space" OR "critical")*. It is important to highlight that only "EDAC" was included in the most used keywords in C_5, as the authors were aware of some papers that used this acronym to identify Error Detection and Correction (EDAC).

Fig. 2 shows the automatic search on the databases, which resulted in the set C_6 containing 125 papers. We created the new set C_7 with eleven additional papers, deleting the repeated papers and applying the exclusion criteria. Finally, sets C_5 and C_7 were joined to form the final set C_8 comprising 35 papers. Besides, we performed a manual search to avoid losing any paper in the databases; this manual search did not include any additional work. The SLR representativeness is worth noting, as the 35 selected papers cover several ECCs, data and redundancy sizes, and error injection and evaluation methods.

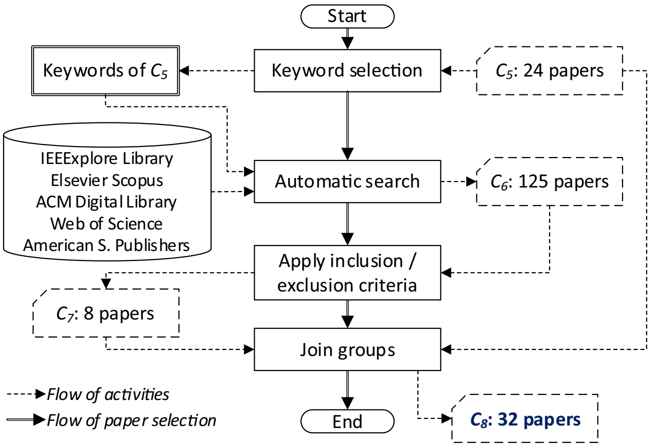


Fig. 2. Papers selection process using the automatic search in science and technology databases.

2.5. Data extraction

After the selection process, we applied a data extraction method to each work in C_8, aiming to answer the research questions employed to guide the proposed organization. Table 2 displays that the data extraction method includes: Metadata, ECC structure and organization, target application and manufacture technology, methods and metrics employed in ECC evaluation, and future works in ECC.

Table 3 presents a summary of the data collected during the evaluation of the 35 primary studies; this summary allows a comparative analysis of the following elements: (i) Work – identification of the work and authors; (ii) Year – allowing to identify the number of works and ECC tendencies along the years; (iii) Classification – containing the terminology adopted to classify the 2D-ECC types; (iv) Target application – aiming to correlate the proposed ECC type with a given target application; (v) Data size and redundancy overhead – the data and redundancy sizes of the codeword enable us to verify ECC tendencies and define ECC overhead metrics; (vi) Fault injection – aiming to understand how the works validate or evaluate their ECC proposals; (vii) CMOS Technology – enabling to compare the manufacture technologies employed on the encoder/decoder synthesis. It is important to emphasize that each 2D-ECC potentially has a specific organization and number of check bits, implying different latency, power, and area consumption of the encoder and decoder. In addition, error correction and detection rates vary depending on the number of errors and the error pattern (e.g., exhaustive, adjacent, and burst). These aspects prevent defining the most suitable 2D-ECC for any scenario. To provide insights for a potential designer, some works (see Section 7.4) employ ad-hoc comparison metrics to correlate physical and logical aspects of an ECC. However, these metrics do not follow a pattern, leaving both the metric and the selection process biased. Consequently, we prefer only presenting the potentialities of each 2D-ECC, leaving the manuscript reader to choose the 2D-ECC based on the target architecture requirements.

3. 2D-ECC foundations

2D-ECC is a way of organizing and correlating data and check bits in two dimensions, typically employing row and column or diagonal bit arrangements. Elias [88] was a pioneer in the 2D-ECC area, exploring the iteration of simple ECCs to transmit messages in the noisy channels of the old telegraph systems. The same author proposed to organize each group of successive symbols in a rectangular matrix, checking each row

Table 2
Data extraction.

Extracted data	Description
1 - Metadata	
1 - Metadata	Title, authors, publication year and citations
2 – ECC structure and organization	
2.1 Data size	Identifies the data size and code organization
2.2 Redundancy	Evaluates redundancy overhead
2.3 ECC type	Allows classifying the 2D-ECC used in the work
3 – Target application and technology	
3.1 Target application	Identifies the features of the target application
3.2 Target technology	Identifies technology used for synthesis
4 – ECC evaluation method	
4.1 Fault injection	Examines the applied fault injection methods
4.2 Error coverage	Analysis of detected and corrected errors
4.3 Evaluation metric	Evaluates performance metrics
5 – ECC Trends	
5.1 Trends	Examines ECC tendencies and future works

Table 3
Summary of data collected from the 32 primary studies (by year and alphabetically ordered).

Work	Year	Classification	Target application	Data size (bits)	Data rate (%)	Redundancy overhead (%)	Fault injection	Manufacturing technology
Ahilan et al. [57]	2015	Extended Product Code	Generic	32	50.0	100.0	Random	180 nm
Anitha et al. [58]	2015	Extended Product Code	Space	32	47.1	112.3	–	–
Erozan et al. [41]	2015	Product Code	Generic	32	40.5	146.9	Adjacent	–
Liu et al. [59]	2015	Product Code	Space	16	50.0	100.0	–	90 nm
Rahman et al. [60]	2015	Mixed Code	Generic	64	70.3	42.2	Exhaustive	–
Castro et al. [61]	2016	Product Code	Space	16	40.0	150.0	Adjacent	45 nm
Mandal et al. [62]	2016	Mixed Code	Generic	49	65.3	53.1	Random	–
Manoj et al. [63]	2016	Extended Product Code	Critical	32 64	47.1 47.1	112.5 112.5	–	180 nm
Sundary et al. [64]	2016	Extended Product Code	Generic	20	43.5	130.0	–	–
Yedere et al. [65]	2016	Extended Product Code	Generic	32	48.5	106.3	–	90 nm
Afrin et al. [54]	2017	Straightforward 2D-ECC	Generic	32	50.0	100.0	Exhaustive	–
Kamatchi et al. [66]	2017	Extended Product Code	Space	32	53.3	87.5	–	–
Liu et al. [67]	2017	Extended Product Code	Space	32	42.1	137.5	Adjacent	65 nm
Raha et al. [68]	2017	Mixed Code	Generic	32	53.3	87.5	–	–
Silva et al. [69]	2017	Mixed Code	Critical	16	50.0	100.0	Adjacent	65 nm
Tambatkar et al. [70]	2017	Mixed Code	Space	32	47.8	112.5	–	45 nm
Athira et al. [71]	2018	Product Code	Generic	32	53.3	87.5	Random	90 nm
Goerl et al. [72]	2018	Mixed Code	Critical	32	44.4	125.0	Random	–
Li et al. [56]	2018	Straightforward 2D-ECC	Generic	16, 16 32, 32 64, 64	57.1, 40.0 69.6, 53.3 80.0, 69.6	75.0, 150.0 43.8, 75.0 25.0, 43.8	–	65 nm
Gracia-Moran et al. [55]	2018	Straightforward 2D-ECC	Generic	32 32	80.0 66.7	25.0 50.0	Adjacent	45 nm
Silva et al. [73]	2018	Product Code	Critical	16 16 16	41.0 40.0 29.6	143.8 150.0 237.5	Adjacent	65 nm
Magalhães et al. [74]	2019	Mixed Code	Critical	16	40.0	150.0	Adjacent	65 nm
Priya et al. [75]	2019	Extended Product Code	Generic	32	50.0	100.0	–	–
Zhang et al. [76]	2019	Extended Product Code	Space	32	50.0	100.0	Adjacent	180 nm
Freitas et al. [77]	2020	Product Code	Space	16	25.0	300.0	Exhaustive	65 nm
Kumar et al. [78]	2020	Mixed Code	Generic	32	69.6	43.8	–	–
Neelima et al. [79]	2020	Mixed Code	Generic	64 64	67.4 62.1	48.4 60.9	–	28 nm
Rohde et al. [80]	2020	Mixed Code	Space	64	44.4	125.0	–	–
Sai et al. [81]	2020	Mixed Code	Generic	32	57.1	75.0	–	45 nm
Silva et al. [82]	2020	Product Code	Critical	32	49.2	103.1	Adjacent	65 nm
Silva et al. [83]	2020	Mixed Code	Critical	32 32	57.1 50.0	75.0 100.0	Adjacent	65 nm
Freitas et al. [84]	2021	Product Code	Space	16	33.3	200.0	Adjacent	65 nm
Sen et al. [85]	2021	Extended Product Code	Critical	32	50.0	100.0	–	–
Freitas et al. [86]	2022	Product Code	Critical	16	33.3	200.0	Exhaustive	65 nm
Freitas et al. [87]	2022	Product Code	Space	16	33.3	200.0	Exhaustive	65 nm

"–" Means that the work does not contain the information about the subject.

with a C1 code and each column with a C2 code, forming a 2D-ECC [88]. Over time, multidimensional codes have evolved into different structural and coding formats, requiring adequate analysis and classification for their understanding and usage.

Let d be the minimum Hamming distance, representing the minimum number of bits that differs between two codewords, then (1) and (2) calculate the maximum number of errors that a code can correct EC or detect ED based on d [89,90].

$$EC = \lfloor \frac{d-1}{2} \rfloor \tag{1}$$

$$ED = d - 1 \tag{2}$$

These equations are exclusive, i.e., EC or ED, but not EC and ED simultaneously. The simultaneity relationship among EC, ED, and d implies using (3) instead of (2). In other words, if an application targets correcting and detecting errors simultaneously, ED is reduced.

$$ED = d - EC - 1 \tag{3}$$

A 2D-ECC organization enables checking a word of data by crossing ECCs, offering a natural bit-interleaving provided by the 2D structure. A 2D-ECC employs more optimized encoding to achieve the same correction rate as a 1D-ECC, consuming less area and energy.

For example, Fig. 3(a) displays a 1D-ECC with 4 data bits, requiring 3 bits for a Hamming encoding. These extra bits provide a Hamming distance of 3, which by (1) and (2) allows for correcting 1 error or detecting 2 errors. Fig. 3(b) shows a 4 × 4 data matrix that employs this encoding in a 1D model, requiring a 4 × 3-bit check matrix; since all words are independent, d is maintained. Note that maintaining the

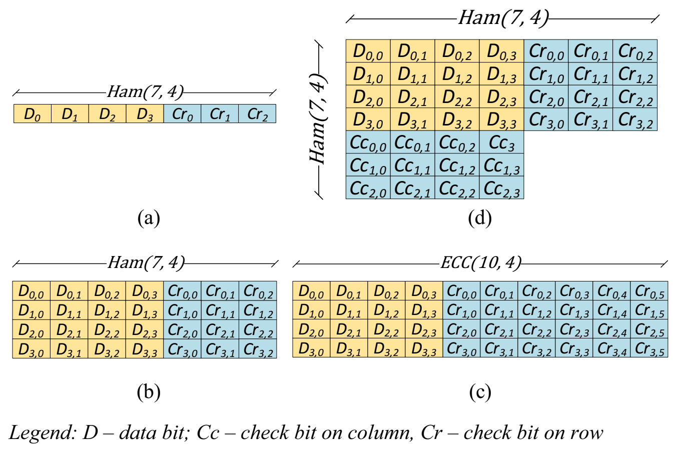

Ham(7, 4)
D_0 D_1 D_2 D_3 Cr_0 Cr_1 Cr_2
(a)
Ham(7, 4)
Ham(7, 4)
$D_{0,0}$ $D_{0,1}$ $D_{0,2}$ $D_{0,3}$ $Cr_{0,0}$ $Cr_{0,1}$ $Cr_{0,2}$
$D_{1,0}$ $D_{1,1}$ $D_{1,2}$ $D_{1,3}$ $Cr_{1,0}$ $Cr_{1,1}$ $Cr_{1,2}$
$D_{2,0}$ $D_{2,1}$ $D_{2,2}$ $D_{2,3}$ $Cr_{2,0}$ $Cr_{2,1}$ $Cr_{2,2}$
$D_{3,0}$ $D_{3,1}$ $D_{3,2}$ $D_{3,3}$ $Cr_{3,0}$ $Cr_{3,1}$ $Cr_{3,2}$
$Cc_{0,0}$ $Cc_{0,1}$ $Cc_{0,2}$ Cc_3
$Cc_{1,0}$ $Cc_{1,1}$ $Cc_{1,2}$ $Cc_{1,3}$
$Cc_{2,0}$ $Cc_{2,1}$ $Cc_{2,2}$ $Cc_{2,3}$
(d)
Ham(7, 4)
$D_{0,0}$ $D_{0,1}$ $D_{0,2}$ $D_{0,3}$ $Cr_{0,0}$ $Cr_{0,1}$ $Cr_{0,2}$
$D_{1,0}$ $D_{1,1}$ $D_{1,2}$ $D_{1,3}$ $Cr_{1,0}$ $Cr_{1,1}$ $Cr_{1,2}$
$D_{2,0}$ $D_{2,1}$ $D_{2,2}$ $D_{2,3}$ $Cr_{2,0}$ $Cr_{2,1}$ $Cr_{2,2}$
$D_{3,0}$ $D_{3,1}$ $D_{3,2}$ $D_{3,3}$ $Cr_{3,0}$ $Cr_{3,1}$ $Cr_{3,2}$
(b)
ECC(10, 4)
$D_{0,0}$ $D_{0,1}$ $D_{0,2}$ $D_{0,3}$ $Cr_{0,0}$ $Cr_{0,1}$ $Cr_{0,2}$ $Cr_{0,3}$ $Cr_{0,4}$ $Cr_{0,5}$
$D_{1,0}$ $D_{1,1}$ $D_{1,2}$ $D_{1,3}$ $Cr_{1,0}$ $Cr_{1,1}$ $Cr_{1,2}$ $Cr_{1,3}$ $Cr_{1,4}$ $Cr_{1,5}$
$D_{2,0}$ $D_{2,1}$ $D_{2,2}$ $D_{2,3}$ $Cr_{2,0}$ $Cr_{2,1}$ $Cr_{2,2}$ $Cr_{2,3}$ $Cr_{2,4}$ $Cr_{2,5}$
$D_{3,0}$ $D_{3,1}$ $D_{3,2}$ $D_{3,3}$ $Cr_{3,0}$ $Cr_{3,1}$ $Cr_{3,2}$ $Cr_{3,3}$ $Cr_{3,4}$ $Cr_{3,5}$
(c)
Legend: D – data bit; Cc – check bit on column, Cr – check bit on row

Fig. 3. (a) 1D-ECC with 4-bit data and 3 check bits; (b) 4 independent 1D-ECC of (a) in a matrix format; (c) 4 × 7 matrix including four 1D-ECC with 4-bit data and 6 check bits; (d) a 4 × 4 data matrix encoded by a 2D-ECC.
Legend: D – data bit; Cc – check bit on column, Cr – check bit on row.

Hamming distance *d* means guaranteeing the same *EC* and *ED* power for any error pattern. Therefore, the ECC of Fig. 3(b) ensures only one error correction for the entire matrix. However, as there are more codewords, the probability of spreading errors into the matrix increases, allowing to correct four errors if each one is in a single codeword (row).

Fig. 3(d) illustrates that a simplified 2D-ECC can protect this data matrix by inserting a second 3 × 4 matrix of check bits, enabling the correlation of each data bit with the corresponding ECCs of the rows and columns. In this 2D-ECC format, a one-bit variation of the data matrix affects both the row and column codes, increasing the Hamming distance to 6 and, consequently, the correction and detection power. However, the complexity of both ECCs, applied to the row and column, are equal and equivalent to the 1D complexity.

A fair comparison of correction and detection power requires the same number of check bits as the 2D-ECC version; thus, every 4 data bits are followed by 6 check bits, as depicted in the 1D-ECC of Fig. 3(c). This format leads to a Hamming distance of 6 and potentially the same *ED* and *EC* efficacy (requires a special ECC built to protect 4 data bits using 6 check bits). However, the new ECCs are 10 bits long; thus, the checking sequence is longer, requiring more complex encoding and decoding circuits, consuming more area and energy.

Additionally, the miniaturization of electronic components increases the number of errors due to SEUs, preventing using 1D-ECCs in many critical applications. For example, Rao et al. [94] present error patterns with seven bitflips arranged in a two-dimensional way; Radaelli et al. [109] analyze error patterns with more than eight bitflips through radiation tests with different energy levels. In several cases, these error patterns are not corrected by 1D-ECCs, requiring the more significant detection and correction capability of 2D-ECCs [98].

4. 2D-ECC classification

A 2D-ECC is characterized by having data and/or redundancy bits in two dimensions, normally named row and column. This definition allows including any 1D-ECC physically organized in rows and columns in the 2D-ECC class. We defined as Straightforward 2D-ECC (S2E) the codes organized in this 2D physical structure, but that remain to correct errors with 1D-algorithms. Fig. 4 exemplifies an S2E containing 16 data bits organized in a matrix format; each row is *encoded independently* with Ham(7,4), a short representation of the Hamming code implemented with three redundancy bits to protect four data bits [35]. Examples of S2Es are found in the works [54–56].

Afrin and Sadi [54] propose a 4 × 16 matrix ECC with 32 data bits and 32 redundancy bits. Each 8 data bits of each row is independently encoded. The first redundancy bit stores the first data bit inverted. The other seven redundancy bits are XOR operations between a pair of bits - 1st and 2nd, 2nd and 3rd, 3rd and 4th, and so on. Gracia-Moran et al. [55] introduce the Flexible Unequal Error Control (FUEC) methodology, developed to satisfy a certain number of syndromes to correct adjacent errors. The authors present two 2D-ECCs with 8 and 16 bits of redundancy but with the same error coverage, designed to correct 1-bit errors and 2 and 3 adjacent bits in the same row or column. Li et al. [56] propose two 2D-ECCs with 32 data bits for correcting up to 3 burst bitflips. The ECCs add 24 and 14 redundancy bits for 4 × 8 and 2 × 16 data formats, respectively. Additionally, codes are organized using interleaving. Although this work focuses on 2D-ECCs whose coding is two-dimensional, we included this ECC class for completeness and understanding.

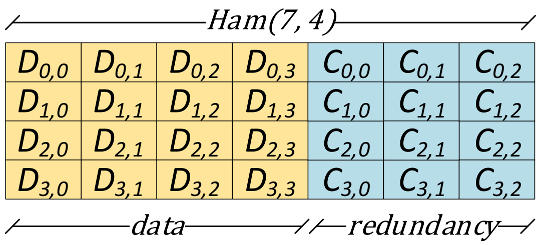

Ham(7, 4)
$D_{0,0}$ $D_{0,1}$ $D_{0,2}$ $D_{0,3}$ $C_{0,0}$ $C_{0,1}$ $C_{0,2}$
$D_{1,0}$ $D_{1,1}$ $D_{1,2}$ $D_{1,3}$ $C_{1,0}$ $C_{1,1}$ $C_{1,2}$
$D_{2,0}$ $D_{2,1}$ $D_{2,2}$ $D_{2,3}$ $C_{2,0}$ $C_{2,1}$ $C_{2,2}$
$D_{3,0}$ $D_{3,1}$ $D_{3,2}$ $D_{3,3}$ $C_{3,0}$ $C_{3,1}$ $C_{3,2}$
data
redundancy

Fig. 4. An S2E example containing 16-bit data (matrix D); each row of the matrix is encoded independently with Ham(7,4).

The 2D-ECC group that we are interested in is complementary to S2E; they present some level of encoding intersection between the dimensions as a common characteristic, i.e., at least one bit of data or redundancy change implies encoding both dimensions. We organized this complementary group into three classes: Product Code (PC), Extended Product Code (EPC), and Mixed Code (MC). Fig. 5 shows the number of papers for each ECC class per year, and Table 4 correlates 2D-ECC classes to works, highlighting the encoding methods that the 2D-ECC employs. Probably, the low number of publications in 2021 and 2022 came from the pandemic, as many conferences were canceled, and the average review time of journals drastically increased.

4.1. Product Code (PC)

Product Code (PC) is an ECC treated as a product of two codes enabling to build long codes based on small ones.

Let α and β be the number of columns composing the data and redundancy areas, and let δ and ε be the number of rows composing the data and redundancy areas, respectively, such that $\gamma = \alpha + \beta$ and $\theta = \delta + \varepsilon$. Then, each row of a PC is encoded using the $C_1(\gamma, \alpha, d_1)$ code, and each column is encoded using the $C_2(\theta, \delta, d_2)$ code, forming the $C_1 \times C_2$ code; therefore, each bitflip in the data region affects both the row and column of the corresponding bit. Fig. 6(a) illustrates the basic PC structure. Also, PC adds a region containing check bits of check bits, increasing the minimum Hamming distance and, consequently, the code correction potential [91]. The PC minimum distance d_{PC} is computed by

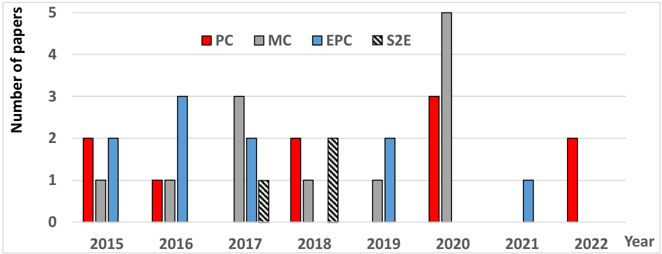


Fig. 5. The graph indicates the number of works by classification (S2E, PC, MC, and EPC) divided by the publication year.

Table 4
Class of 2D-ECC used in each work.

Class	Work	Encoding method
EPC	[57,75]	XOR operations and parity
	[58,63,65,85]	Decimal sum and parity
	[76]	Parity
	[64,66,67]	Hamming and parity
MC	[60,62,69,78,79]	Parity
	[72]	Parity and duplication
	[68,70,74]	Hamming and parity
	[81]	Hamming
	[83]	Logic operations and parity
	[80]	XOR operations, hamming and parity
PC	[61,73,82]	Extended hamming and parity
	[41]	LDPC and parity
	[59]	MED and parity
	[71]	Hamming and parity
	[77,84,86,87]	Extended hamming in rows and columns
S2E	[54]	XOR and NOT operations
	[55]	XOR operations and FUEC
	[56]	3-Burst error ECC in rows

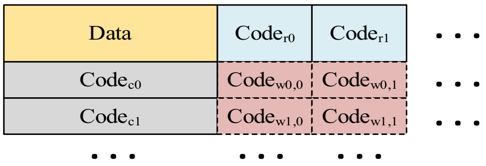


Fig. 6. Basic structures of (a) PC and (b) modified PC (based on [88]).

multiplying the distances of each 1D-ECC that make up the product code, i.e., $d_{PC} = d_1 \times d_2$.

PC increases the theoretical correction and detection capacity but also increases the redundancy costs, implying more area and energy consumption. Some authors proposed the *modified PC* to reduce the associated redundancy costs, which do not have the check bits of the check bits [89,90]. Fig. 6(b) illustrates the structure of a modified PC, and (4) displays its minimum distance calculation d_{mPC}.

$$d_{mPC} = d_1 + d_2 - 1 \tag{4}$$

Based on this explanation, we can cite the works of [41,59,61,71,73,77,82,84,86,87] as ECCs in PC format. In [61,73,82], the authors use extended Hamming in rows and parity in columns to encode each word. These works present iterative decoding using the row and column check bits to correct data, and the check bits of check bits to correct both the row and column check bits. Each correction can enable a new error correction iteratively, increasing the error correction capacity of the code. Freitas et al. [77] implemented the Product Code for Space Applications (PCoSA), a PC applying extended Hamming to rows, columns, and check bits of check bits areas. In 2022, Freitas et al. [87] designed an optimized version of PCoSA with 16 bits less redundancy called the Optimized Product Code for Space Applications (OPCoSA). The same authors [84] propose Line Product Code (LPC), a modified PC that reaches near error correction rates of PCoSA by improving the decoding algorithm. In 2022, Freitas et al. [86] designed a new version of LPC with specific single error correction algorithms and an innovative double error correction technique. The works [41,59,71] propose modified PCs that employ a single row of parity bits to encode columns and a more complex code to encode rows. Erozan and Çavus [41] propose to codify each row employing the Euclidian Geometry Low-Density Parity Check (EG-LDPC). Athira and Yamuna [71] employ five Hamming bits to encode each one of the four rows of 8 data bits. Finally, Liu et al. [59] propose using Multi-bit Error Detection (MED), a code capable of detecting multiple errors by applying 4 redundancy bits to each row of 8 data bits.

4.2. Extended Product Code (EPC)

EPC is a special case of PC that uses more than one code per row and/or column; therefore, C_1, C_2, or both are heterogeneous codes. Besides, this class can also have check bits of check bits, regardless of whether they are homogeneous or heterogeneous codes, as exemplified in Fig. 7.

The works [57,58,63–67,75,76,85] are classified as EPCs. Kamatchi et al. [66] propose the modified Decimal Matrix Code (MDMC) for encoding 32 data bits in a 2 × 16 matrix, with each row divided into four 4-bit regions. Each row is encoded by adding the two odd regions with the two even regions, totaling 10 bits per row since each sum requires 5 bits. Besides, each column in the data region is encoded with a parity bit. The authors in [57,75] divide the 32 bits of data into two rows of 16 bits. Each row encodes two sets of four bits executing a bitwise XOR operation; thus, eight redundancy bits are added per row. Columns are encoded using parity. Manoj and Babu [63] divided the 64 data bits into two rows of 32 bits each. A sum is applied to each set of two four-bit words for each row, resulting in five bits. This same structure is done four times per row, adding 20 redundancy bits; also, columns are encoded with parity bits. Anitha and Jeevidha [58] describe a similar technique presented in [63], but for a 32-bit codeword implemented in two rows of 16 bits, totaling ten redundancy bits per row. Yedere and Pamula [65] perform another similar organization; the 32 data bits are divided into two rows. The first eight bits are added to the last eight bits for each row, resulting in nine redundancy bits per row. Sen et al. [85] propose a DMC-based error correction technique, decreasing redundancy and improving the error correction rate. Liu et al. [67] split 32 data bits into eight rows. Each row encodes the four bits with parity and Hamming. However, there is no ECC to encode the four columns of the data region. The column coding is done only in the Hamming check bit region, and parity is used for each pair of bits. Zhang et al. [76] split 32-bit data into four rows. Their proposal uses parity for every two bits in the row (adds four per row) and column (adds 2 per column), totaling 32 redundancy bits. Finally, Sunary and Logisvary [64] organize a 20-bit codeword in a 4 × 5 format; the ECC encodes each of the four rows using Hamming and each of the five columns using parity every two bits.

4.3. Mixed Code (MC)

MC is a 2D-ECC class containing at least one bit of data or redundancy whose change implies encoding both dimensions but cannot be classified as PC or EPC. The works of [60,62,68–70,72,74,78–81,83] are examples of MC; Fig. 8 shows the 2D-ECCs proposed on works [69,79].

Mandal et al. [62] propose the Modified Matrix Code (MMC) for FPGA-based systems. MMC corrects multiple errors in a 7 × 7 data matrix, employing 13 parity bits encoded diagonally and vertically. Additionally, MMC has two redundancy bits associated with each column; one redundancy stores the XOR of the even bits, and the other one stores the XOR of the odd bits.

Goerl et al. [72] present the Parity per Byte and Duplication (PBD) technique that uses parity for each byte and duplicates the content. For example, a 32-bit word (4 bytes) requires four parity bits, and the resulting 36 bits are duplicated, totaling 32 data bits and 40 redundancy bits.

Fig. 8(a) illustrates the Matrix Region Section Code (MRSC) developed by Silva et al. [69], a structure of 16 data bits and 16 redundancy

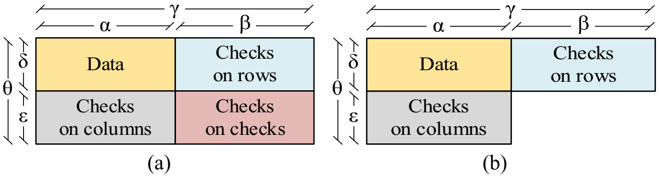
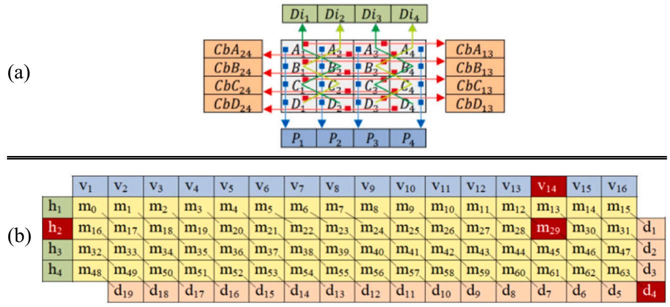


Fig. 7. Structure of an EPC.



	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}	v_{16}	
h_1	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7	m_8	m_9	m_{10}	m_{11}	m_{12}	m_{13}	m_{14}	m_{15}	
h_2	m_{16}	m_{17}	m_{18}	m_{19}	m_{20}	m_{21}	m_{22}	m_{23}	m_{24}	m_{25}	m_{26}	m_{27}	m_{28}	m_{29}	m_{30}	m_{31}	d_1
h_3	m_{32}	m_{33}	m_{34}	m_{35}	m_{36}	m_{37}	m_{38}	m_{39}	m_{40}	m_{41}	m_{42}	m_{43}	m_{44}	m_{45}	m_{46}	m_{47}	d_2
h_4	m_{48}	m_{49}	m_{50}	m_{51}	m_{52}	m_{53}	m_{54}	m_{55}	m_{56}	m_{57}	m_{58}	m_{59}	m_{60}	m_{61}	m_{62}	m_{63}	d_3
		d_{19}	d_{18}	d_{17}	d_{16}	d_{15}	d_{14}	d_{13}	d_{12}	d_{11}	d_{10}	d_9	d_8	d_7	d_6	d_5	d_4

Fig. 8. Examples of Mixed Codes: (a) MRSC [69], (b) HVD [79].

bits, totaling a 4 × 8 matrix. MRSC implements the redundancy in (i) a parity bit for each one of the four diagonals in the data area; (ii) a parity bit for each one of the four rows of the data area; and (iii) Two check bits for each row resulting from the XOR operations between bits 1 and 3 and between bits 2 and 4.

Sai et al. [81] propose a 2D-ECC for detecting and correcting multiple errors for a 4 × 8 data matrix. The ECC applies Ham(7,4) to each of the eight 4-bit diagonals of the data region, totaling an increase of 24 redundancy bits.

Neelima and Subhas [79] propose an ECC based on Horizontal-Vertical-Diagonal (HVD) in 4 × 16 and 2 × 32 data formats, including 39 and 67 bits of redundancy, respectively. The authors call this 3D encoding technique, shown in Fig. 8(b), because it encodes bits horizontally, vertically, and diagonally. Each of the rows, columns, and diagonals has a parity bit for both formats. Rahman et al. [60] propose the Horizontal-Vertical-Double-Bit-Diagonal (HVDD) technique that can correct up to three errors, and it is similar to the technique proposed in [79]; however, in HVDD, the diagonals have two parity bits. Tambatkar et al. [70] use HVD with Hamming applied to the redundancy bits to increase the error correction rate. The codeword has 4 × 8 data with 4-row parity bits, 8-column parity bits, and 11-diagonal parity bits, totaling 23 bits organized in two 8-bit words and one 7-bit word. These three words are encoded with three Hamming, each with 4 check bits, totaling a further 12 redundancy bits; the final codeword has 35 redundancy bits.

Silva et al. [83] developed the extended MRSC (eMRSC), a version that extends the 16 data bits of MRSC [69] to 32 bits. The authors propose a new region scheme to reduce redundancy bits while maintaining a high error correction rate; they show the experimental results with two codeword structures with 24 and 32 redundancy bits.

Raha, Vinodhini, and Murty [68] present the Horizontal-Vertical Parity and Diagonal Hamming (HVPDH) method to protect a 4 × 8 data matrix. HVPDH adds 28 redundancy bits organized in 4-row and 8-column parity bits and 4 Hamming check bits for each of the 4 diagonal 8-bit words.

Rohde and Martins [80] propose a 2D-ECC with 64 bits of data organized with interleaving in five 11-bit words and one 9-bit word. The code has 4 check bits for each of the 6 rows and a further 56 parity bits for rows, columns, data, and check of the calculated check bits, totaling 80 check bits.

Magalhães, Alcântara and Silveira [74] propose the Parity Hamming Interleaved Correction Code (PHICC), designed to correct multiple errors in a 4 × 4 data matrix. PHICC employs extended Hamming (3 check bits and a parity bit) for each row, and a parity bit for each column of the codeword, treating data and check bits in an interleaved way.

Kumar et al. [78] proposed a technique that uses only 14 parity bits to correct adjacent errors in a 32-bit data matrix, resulting in an efficacy equal to the Matrix code, reducing redundancy bits, consumed area, dissipated power, and delay.

4.4. Final remark - encoding method

Six of the ten PC works apply parity as the C_2 method to code columns. Also, seven use extended Hamming as C_1, C_2, or both encoding methods. An important fact is that 100 % of the PC works utilize extended Hamming or parity. 100 % of the eight EPC papers apply parity, and three also apply Decimal Sum, a coding technique that uses the binary sum of n-bit words. Finally, 13 of the 14 MC works implement parity, five employ Hamming, and 100 % apply one of these two techniques. Thus, the encoding methods most used in 2D-ECC implementations are Hamming and parity; their implementation simplicity, which produces low area consumption, power dissipation, and latency, is the main reason for their usage.

5. Data size and redundancy metrics

This section compares the data size of the 2D-ECC words together with metrics for redundancy overhead assessment.

5.1. Data size

The SLR analysis revealed that most 2D-ECCs are designed to operate with standard memory-processor buses encompassing 16, 32, or 64 bits. Together with the regular matrix format of 2D-ECCs, these data sizes force the implementation of 1D-ECCs with 4 or 8 bits in rows and columns, producing 4 × 4, 4 × 8, 8 × 4, or 8 × 8 data matrices. Some 2D-ECCs implement 32 and 64 bits using 16 or 32 bits in rows, making 2 × 16, 4 × 16, and 2 × 32 data matrices. Exceptions are found in two special data organizations [62,64] containing 20 and 49 data bits. Sunary and Logisvary [64] employed 20-bit data since their work targets a 20-bit multiplier; the authors also explored their proposed correction model for cases in the range of 10 to 128 bits. Mandal et al. [62] use a 7 × 7 data matrix, resulting in 49 bits, but they explain that the same ECC organization can be applied to other matrix sizes.

Table 5 shows that more than half of the works (i.e., [41,54–58,63,65–68,70–72,75,76,78,81–83,85]) assess 2D-ECCs targeting 32-bit data memories. Ten authors carry out experiments with 16-bit memories [56,59,61,69,73,74,77,84,86,87] and five authors work with 64-bit memories [56,60,63,79,80].

Note that Table 5 has >35 ECCs, as some papers discuss more than one ECC size. Li et al. [56] propose a 32-bit ECC in a 4 × 8 data region and evaluate the proposed scheme for 16 and 64 bits, showing the number of redundancy bits added, the ability to correct errors, and the encoder and decoder latencies. Manoj and Babu [63] propose a 64-bit ECC, but also presented the implementation with 32 bits, showing the number of redundancy bits and error correction ability.

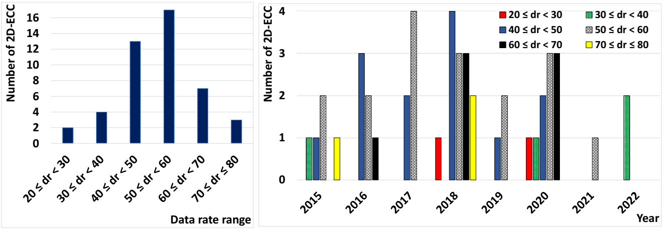
Fig. 9 shows the number of 2D-ECC per year regarding the memory size. Most studies are for proposals of 32-bit data, which had the most significant number of publications in 2017, 2018, and 2020. Besides, in all years, the number of 2D-ECCs with 32-bit data is always equal to or greater than the number of 2D-ECCs of other data sizes, except for 2022. However, this work only analyzed works until March 2022. Finally, until 2020 ECCs have shown a growth in 64-bit proposals; however, the number of works and sampling time is not representative.

5.2. Redundancy metrics

The number of redundancy bits is one of the determining factors in detecting and correcting errors. Since 2D-ECCs are normally used to

Table 5
Relationship between number of 2D-ECCs and data size.

	16 bits	20 bits	32 bits	49 bits	64 bits
Number of papers	13	1	24	1	7
Percentage	28.3 %	2.2 %	52.2 %	2.2 %	15.2 %



(a) (b)

Fig. 9. Number of papers published per year, from 2015 to 2020, in relation to the data size.

mitigate critical system failures, these codes typically have numerous redundancy bits. Additionally, the proportion of the redundancy bits in relation to the data or codeword bits directly influences the memory storage area and the implementation costs of the encoding and decoding circuits.

Let k, r, and n be the numbers of data, redundancy, and codeword bits, respectively, such that (5) correlates these values; then, (6), (7), and (8) show how to calculate *data rate* (*dr*), *redundancy rate* (*rr*) and *redundancy overhead* (*ro*), respectively.

$$n = k + r \tag{5}$$

$$dr = k/n \tag{6}$$

$$rr = r/n \tag{7}$$

$$ro = r/k \tag{8}$$

dr, referenced in some works as *code rate*, is the percentage of *k* data bits in the *n* codeword bits. *rr* and *ro* are metrics that regard the redundancy impact on the ECC. *rr* and *ro* are the percentages of redundancy bits *r* concerning the *n* and *k*, respectively. Usually, low-cost ECCs have high values of *dr* and low values of *rr* and *ro*. We explore this section using only *dr* and *ro* since *rr* is complementary to *dr*. For example, Ham(7,4), a short representation of the Hamming code with $k = 4$ and $r = 3$, has $dr = \frac{4}{7} = 57.1\%$, $rr = \frac{3}{7} = 42.9\%$, and $ro = \frac{3}{4} = 75\%$. Figs. 10 and 11 show the number of 2D-ECC works according to *dr* and *ro*, and the publication year. These figures present 46 proposals for 2D-ECC; this number is higher than the 35 selected works because some works have more than one proposal. This is the case of [73], which proposes two 16-bit codes, including 23 and 38 redundancy bits. The same is true for other works such as [55,56,63,79,83], which change the number of redundancy or data bits.

Fig. 10(a) shows the total number of 2D-ECCs according to the *dr* range. The graphic was divided into six ranges; the leftmost range represents codes with a lot of redundancy in relation to the size of the final word, and the rightmost range represents codes with a low redundancy rate concerning the total encoded data. On the one hand, 30 ECCs (65 %) have *dr* between 40 % and 60 %, meaning that most current 2D-ECCs use an average of 50 % of their area for redundancy; i.e., they present ECC with a sound tradeoff between error correction efficacy and implementation and usage costs. Besides, seven [57,61,63,67,68,70,73] of the nine papers with more than three citations are in this *dr* range. On the other hand, only four proposals are within the limits of low and high *dr* ranges. Three works [55,56,60] describe very low-cost codes ($70 \le dr \le 80$), justifying that many applications require low-energy decoder implementations and low-memory usage with acceptable error correction capacity. Furthermore, only two other works [73,77] present very high-cost codes ($20 \le dr < 30$), justifying that critical or space applications require ECCs with more correction capacity and thus higher redundancy rates, as the process technology decreases, rising the MBU rates.

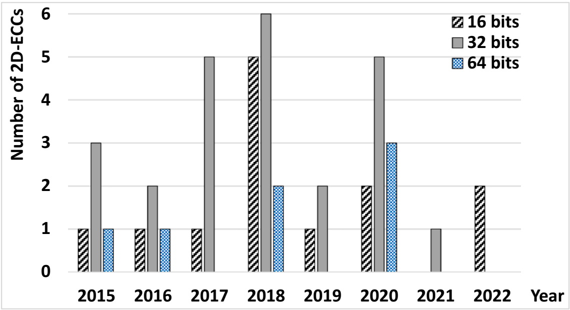
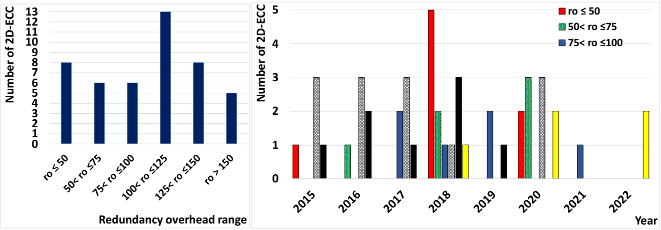


Fig. 10. Number of 2D-ECC proposals (a) per *dr* range and (b) per year.



(a) (b)

Fig. 11. Number of 2D-ECC proposals (a) per *ro* range and (b) per year.

Fig. 10(b) aims to correlate the data illustrated in Fig. 10(a) with the years in which the works took place. Fig. 10(b) shows the number of proposals has increased in recent years, signaling a trend in the 2D-ECC research.

Fig. 11(a) shows the number of 2D-ECC according to six *ro* ranges. The leftmost and rightmost ranges represent codes that added little and a lot of data redundancy, respectively. Almost 50 % of the 2D-ECCs have a *ro* between 75 % and 125 %, and among the nine proposals with more than three citations, four are in this range [61,66,68,70].

Finally, Fig. 11(b) correlates the data shown in Fig. 11(a) with the years in which the works were published. It is worth mentioning that works with very high *ro* rates are showing recent growth, as can be seen in 2018 [73], 2020 [77], 2021 [84], and 2022 [86,87]. Besides, 2D-ECCs with *ro* in the range (100, 150] have had publications in all evaluated years, representing practically 50 % of all ECCs.

6. Target application

The analysis of the selected works revealed that many authors define the characteristics of ECCs based on the target application, which is very sensitive to the environment in which it is inserted. Most of these works describe general techniques and algorithms to mitigate memory failures. Still, there is an intense concentration of works that use 2D-ECCs in critical applications, especially those subject to high collision rates of alpha particles and neutrons, such as in a space environment. Therefore, we classified the 2D-ECC works according to the target application in: (i) *Space* – works focusing only on space applications; (ii) *Critical* – works targeting any critical application without focusing on the space ones; and (iii) *Generic* – works that target any application that requires mitigating memory faults without a specific target. Table 6 illustrates this arrangement.

The target application criterion allows a certain degree of subjectivity since many studies are not specific about this criterion. Aiming to

Table 6
2D-ECC applications.

Work	App
[58,59,61,66,67,70,76,77,80,84,87]	Space
[63,69,72–74,82,83,85,86]	Critical
[41,54–57,60,62,64,65,68,71,75,78,79,81]	Generic

minimize this subjectivity, papers without explicit mention of the target application were classified according to the works used to compare with the proposed ECC. Besides, all works that were not possible to specify the target application were classified as *Generic*.

The three classes defined here have works published almost every year of the period selected in this SLR, showing continuous research regardless of the target application.

Works defined as *Space* tend to be implemented with 2D-ECCs classified as PC or EPC. The works classified as *Space* are characterized by discussing and working with multiple errors and by the high redundancy rates of their ECCs, presenting a $dr = 43.3\%$ and a $ro = 142.5\%$, on average; i.e., codes have on average 36 % bits of redundancy than data bits.

Most works defined as *Critical* have 2D-ECCs classified as PC and MC, although some have ECCs classified as EPC. The code redundancy rates are comparatively lower than those obtained with works classified as *Space*, showing $dr = 45.0\%$ and $ro = 128.1\%$, on average.

Most of the 2D-ECCs (51.2 %) are not focused on specific applications, being defined as *Generic*. These 2D-ECCs have a low redundancy rate compared to *Space* and *Critical* works. Essentially all *Generic* works have more data bits in their ECCs than redundancy bits, reaching $dr = 56.0\%$ and a $ro = 84.1\%$, on average. Additionally, all 2D-ECCs classified as S2E, the largest number of MCs and PCs or EPCs with low redundancy, make up the *Generic* group.

7. Analysis methods

This section presents methods and metrics used in the studies to assess the proposed 2D-ECCs, more specifically, (i) methods of fault injection to explore the capabilities of correction and detection error according to error patterns; (ii) meantime to failure used to estimate the ECC reliability over its use; (iii) evaluation costs for implementing and operating the decoders and encoders in given manufacturing technology; and (iv) metrics for reaching a multiobjective function.

7.1. Fault injection method

Error Detection and Correction are the primary objectives of an ECC; the accurate evaluation of these objectives is critical in the ECC design. It is essential to define error patterns to carry out fair comparisons among ECCs. This SLR found five types of error injection patterns: Adjacent, Exhaustive, Random, Burst, and 36 predefined error patterns. Table 3 presents the papers and the error injection method used.

Adjacent errors are used in the works [41,55,61,67,69,73,74,76,82–84] for imitating the structure of MCUs that occur around a certain neighborhood, as indicated by the works of [92–97]. In [61,73,83], the authors verified one million words generated pseudo-randomly for each scenario; they placed patterns ranging from one to eight errors in adjacent cells. The authors in [69] performed a million pseudo-random positions for each scenario, varying from one to seven errors; the authors of [41] performed a similar analysis, extending to 12-error patterns.

The authors in [82] evaluate the ECC efficacy by employing a set of 10,000 patterns for each scenario ranging from one to eight errors, considering adjacent errors with a maximum distance of one bit. The authors in [55] evaluated the ECC proposal with experiments that considered injection of errors of the type (i) simple, (ii) horizontal adjacent from two to eight bits, (iii) vertical adjacent from two to five bits, and (iv) squares in 2 × 2, 3 × 3, and 4 × 4 formats. In [67], all adjacent patterns from one to four errors were added in the codeword. The authors of [84] describe that the incidence of radiation or heating errors occurs within a certain neighborhood. They proposed a failure model that considers a central and 24 adjacent cells, forming a 5 × 5 area, as a possible error region. Finally, in [76], the authors describe experimental results showing that a radiation event generates a maximum of up to four errors.

The works [54,60,77,86,87] employ exhaustive analysis of errors in the experimental results but limit the maximum number of errors to avoid a lot of computational time. Afrin and Sadi [54] injected all error possibilities a 32-bit data, but only in the data region, i.e., the redundancy region is not evaluated. The authors in [77,86,87] simulated all combinations from one to seven, eleven, and six bitflips, respectively. Finally, Rahman et al. [60] do not detail the injection method, but they explain their solution detects all error combinations and corrects up to three errors.

The fast evaluation of non-polarized scenarios with different criticality levels leads to works like [57,62,71,72], adopting randomness as a way of injecting errors in their 2D-ECC experiments. Mandal et al. [62] compared the ECC efficacy in a 32 × 32 memory randomly injecting patterns with up to 40 errors. Goerl et al. [72] randomly select from 1 to 10 bitflips per round error scenario to be verified in a 32-bit register. The experiments proposed by Athira and Yamuna [71] injected one million test vectors with one to eight errors distributed in random positions. Finally, Ahilan and Deepa [57] mention that their experiments were verified by employing hundreds of random errors.

Let l be the number of contiguous bits in a word (i.e., the burst length); then, a burst error is a binary pattern of l bits, where at least the first and last bits are with error. Note that adjacent errors are a specific type of burst error in which all the error bits are contiguous [81,98,99]. The works [55,56,68,70,75,79,81] exemplify burst errors. The work of Gracia-Moran et al. [55] focuses on designing a 2D-ECC to mitigate adjacent errors, but they evaluate the ECC efficacy for tolerating burst errors. Sai et al. [81] explored a 2D-ECC capable of correcting until 8 errors in burst format. Neelima and Subhas [79] explore two works that use the burst error method to analyze the ECC efficacy in 32-bit memory; one can correct up to four errors, and the other can correct up to 11 errors, both in burst format. Li et al. [56] propose an ECC capable of correcting up to three burst errors in each data matrix row. Priya and Vijay [75] evaluate an ECC to mitigate burst errors in a group of information bits affected by radiation. Tambatkar et al. [70] describe the existence of methods to detect and correct single, multiple, or burst errors, implying various redundancy overloads and energy consumption. Finally, Raha, Vinodhini, and Murty [68] describe an ECC based on Multidirectional Parity Code for mitigating errors in the data region and Hamming for increasing the ability to correct burst errors in a noisy environment.

Rao et al. [94] simulated neutron incidence using a commercial assessment tool. The tool uses a nuclear database, input details about the radiation environment, and the target memory layout to calculate the distribution of the current pulses generated for each memory cell. These pulses of current are then injected into a SPICE netlist to extract the SEU and MBU rates. The evaluation was performed using a 45 nm CMOS technology for an SRAM memory. Fig. 12 shows the 36 adjacent error patterns with the respective probability of occurrence. Rohde and Martins [80], Freitas et al. [77], and Ogden and Mascagni [92] are works that used this standard for 2D-ECC efficacy evaluation.

The works [56,58,59,63–66,68,70,75,79–81,85] do not show error

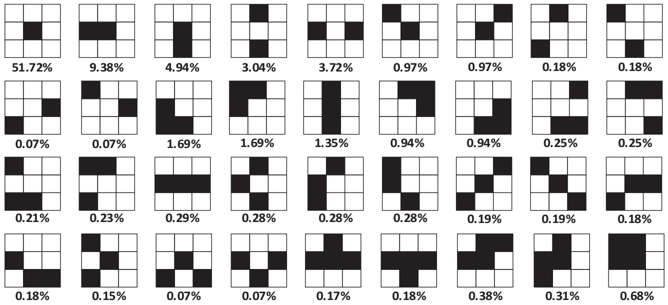


Fig. 12. Single and multiple error patterns in a 45 nm SRAM with the respective probability of occurring each error pattern (adapted from [94]).

injection method for ECC valuation. Instead, some of these works only report the maximum number of errors that the code can correct, and others present a mathematical formulation to prove the ECC efficacy.

7.2. Reliability and MTTF

The works [58,74,76,77,83,84,87] present the ECC efficacy degradation across time, employing the Mean Time To Failure (MTTF) metric, which is build based on Eqs. (9) to (13) proposed in [51]. The objective of these works is to compare ECCs according to the failure probability in a given time interval, assuming that errors can be cumulative over time and the number of errors is proportional to the codeword size. Thus, although the increase in redundancy bits tends to raise the error correction rate, this redundancy increase also raises the probability of errors occurring over time.

Scrubbing is a memory error cleaning technique consisting of reading each memory address, correcting the error bits based on ECC, and writing the corrected data at the same address [100–103]. Since the incidence of memory errors can occur with both spatial and temporal distances, the MTTF metric and scrubbing technique can be explored, for example, in two situations of systems containing memory protected by ECC: (i) considering that the system cannot perform scrubbing in memory, as it operates in a critical situation, such as a space mission that implies reduced battery consumption, or (ii) considering that the system can perform scrubbing in memory in any time interval. In the first case, MTTF is a limit for the ECC efficacy degradation criterion since errors remain cumulatively in memory and no error recovery can be made throughout the system operation. In the second case, MTTF can define a tradeoff between the desired ECC efficacy and the scrubbing period. Thus, the choice of ECCs with different efficacy and costs of implementation and operation can be compensated by a shorter scrubbing period.

Let i be the number of errors, λ the error rate of a single bit per day (typical λ value used in experiments is 10^{-5} upsets/bit/day [51]), and assuming that (i) transients errors occur with a Poisson distribution, and (ii) bitflip occurrences are statistically independent, then (9) estimates $P_{in}(t)$ as the probability of i error occurrence in a given memory word with n bits at time t. Eq. (10) computes $P_n(t)$ - the probability of having errors in memory due to the rate λ over time.

$$P_{in}(t) = \binom{n}{i}\left(1 - e^{-\lambda t}\right)^{i} e^{-\lambda t(n-i)} \tag{9}$$

$$P_n(t) = 1 - e^{-n\lambda t} \tag{10}$$

Let σ be the maximum number of errors for which the ECC was evaluated, and $\varepsilon(i)$ is the error coverage rate for each of the i errors, (11) estimates the memory reliability in time t considering ε of a given ECC.

$$r(t) = 1 - P_n(t) + \sum_{i=1}^{\sigma} P_{in}(t) \times \varepsilon(i) \tag{11}$$

Since M is the number of addresses in memory and each memory address consists of a single codeword, then (12) calculates $R(t)$, which is the reliability at time t of all memory. Note that $M = 1$ means the evaluation of a memory encompassing a single codeword.

$$R(t) = r(t)^{M} \tag{12}$$

Fig. 13, extracted from [84], illustrates the type of result that (12) can provide for comparing ECCs, as well as the aggressiveness effect of λ for reducing reliability.

Finally, (13) shows the MTTF of a memory protected by an ECC is estimated by the reliability integral in time interval t.

$$MTTF = \int_0^t R(t)dt \tag{13}$$

Silva et al. [83] computed reliability and MTTF for M= 1, 8, and 16, and $\lambda = 10^{-5}$ for 8000 operating days. Freitas et al. [84] treat memory reliability for 15,000 days using $M = 1$ and three values of λ (10^{-4}, 10^{-5}, and 10^{-6}); the same authors explore in [77] and in a thousand address memory ($M = 1000$). Zhang et al. [76] also calculate the reliability of a thousand address memory using $\lambda = 10^{-5}$. Magalhães, Alcântara and Silveira [74] evaluate the MTTF of 2D-ECC using $M = 16$ and $\lambda = 10^{-4}$, 10^{-5} and 10^{-6}. Additionally, Anitha and Jeevidha [58] do not discuss the reliability formulation but use it to explore the ECC efficacy as a function of time. Finally, Argyrides et al. [53] expanded their work presented in [51] for evaluating memory sizes ranging from 1 Mb to 128 Mb with $\lambda = 10^{-5}$; they provide a new formulation to reach MTTF from the Mean Error to Fail (METF) metric, but we did not found mentioning of this last approach on the SLR works.

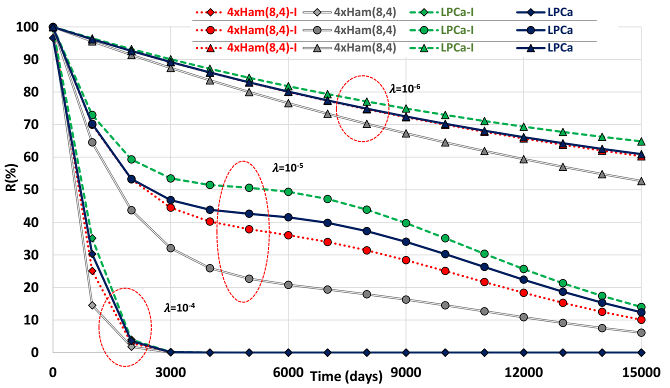


Fig. 13. Reliability of four 2D-ECCs regarding time and three values of λ (extracted from [84]).

7.3. Process technology

The logical or physical implementations are other features usually described in ECC works; Fig. 14 shows the 2D-ECC manufacturing technologies most evaluated.

Twenty-two of all the evaluated papers (62 %) describe the manufacturing technology used in the encoder and decoder 2D-ECC synthesis. Of this total, [57,63] implement two technologies, resulting in the 24 works of Fig. 14. The SLR analysis shows a tendency to explore technologies with 65 nm or less and the disappearance of older technology evaluations. Most works, 75 %, are implemented at 45 or 65 nm, having a peak on 65 nm technology in 2020. The last exploration of 90 nm CMOS technology occurred in 2018, and the last work evaluating 180 nm technology was published in 2016. Still, in 2019 [76], the authors performed some experiments referring to previous work employing 180 nm. Finally, only in 2020, one work explored a technology below 45 nm.

The SLR analysis showed that all syntheses have CMOS as the base technology, with 28 nm as the technological limit, far from the recent

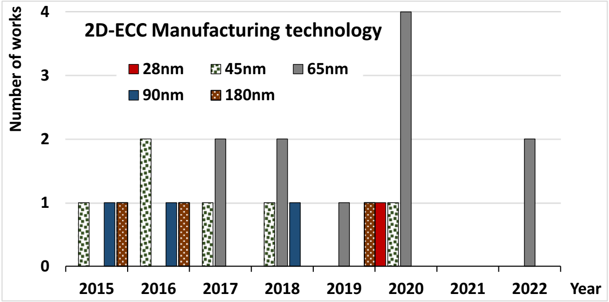


Fig. 14. Number of 2D-ECC works organized by the manufacturing technology and year.

sub-5 nm CMOS technologies. Although most of these studies report that the latest technologies are more susceptible to errors, none present experiments on the efficacy of ECCs for correcting or detecting errors in the face of technology variation. Besides, we did not find spatiotemporal error patterns concerning technology variation or investigations about error patterns in different memory operation environments. Filling these gaps allows defining error injection patterns to explore and validate ECC proposals.

7.4. Multiobjective metrics for ECC assessment

The target application requirements, including correcting and detecting error effectiveness, implementation and operation efficiency in the memories, and encoding and decoding circuits, determine the ECC choice. Thus, studies as [55,61,69,73,82,83], which were based on the seminal papers [51,53], propose multiobjective metrics to assess correction capabilities and ECC error detection associated with the encoder and decoder costs for implementing, such as occupied area and operating, such as consumed energy.

Argyrides, Zarandi, and Pradhan [51] propose Correction Coverage per Cost (*CCC*) and Detection Coverage per Cost (*DCC*) metrics, defined in (14) and (15), respectively, to evaluate the ECC efficacies in detecting and correcting errors regarding power, delay and area costs.

$$CCC_i = \frac{CR_i}{Power \times Delay \times Area} \tag{14}$$

$$DCC_i = \frac{DR_i}{Power \times Delay \times Area} \tag{15}$$

The authors normalized *Area*, *Power*, and *Delay* according to a physical implementation without ECC protection, and *CR* (correction rate) and *DR* (Detection rate) are presented in percentage values. Note that the correction and detection rates depend on the number of errors; thus, we chose to express CR_i, DR_i, CCC_i, and DCC_i according to the number of errors *i*, generating discrete graphics, as exemplified in Fig. 15. In [53], Argyrides, Pradhan, and Kocak employed only the CCC_i metric proposed in [51]. The works [51,53] do not describe if the implementation and operation costs are extracted from the syntheses of decoder, encoder, or both.

Castro et al. [61] improve CCC_i, and DCC_i metrics proposing the Total Coverage per Cost (TCC_i), as shown in (16). TCC_i covers detection and correction performances regarding the number of errors *i* and employing only the decoder implementation and operation costs in the denominator. The authors use costs associated only with the decoder.

$$TCC_i = \frac{CR_i \times DR_i}{Area \times Power \times Delay} \tag{16}$$

Silva et al. [69] use the TCC_i metric of [61] regarding both encoder and decoder implementation and operation costs. *Area* and *Power* were achieved by adding the individual values of the encoder and decoder, but *Delay* is considered the highest value between the encoder and decoder since the authors explain this value defines a frequency rate, which is not cumulative. All values were normalized by dividing by the smallest value. Additionally, the same authors of [69] use the TCC_i metric in [73] but consider only the decoder to compose the synthesis costs.

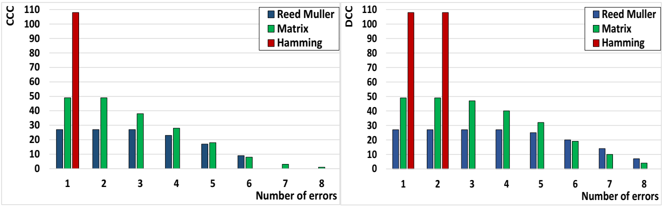


Fig. 15. Correction Coverage per Cost (*CCC*) and Detection Coverage per Cost (*DCC*) according to the number of errors (based on [51]).

Silva et al. [83] proposed CTC_i, which considers both encoder and decoder synthesis costs, as seen in (17) and (18). In [82], the same authors use the CDC_i metric, which is the same CTC_i metric adapted from [83]. It is worth mentioning that CR_i for [82,83] were obtained for adjacent errors, and the results were normalized (divided by the highest value) after calculating CTC_i or CDC_i.

$$CTC_i = \frac{CR_i}{Cost(Encoder) + Cost(Decoder)} \tag{17}$$

$$Cost(Encoder/Decoder) = Area \times Power \times Delay \tag{18}$$

Gracia-Moran et al. [55] highlight the importance of adding redundancy cost for computing a multiobjective metric. Therefore, based on [98], they proposed (19) to calculate the M_i metric, which includes *Redundancy* cost as the *dr* metric described in Section 5, i.e., data rate.

$$M_i = \frac{CR_i \times DR_i}{Area \times Power \times Delay \times Redundancy} \tag{19}$$

We understand that multiobjective metric assists in the ECC selection. Still, the analysis of all the multiobjective metrics used so far to compare 2D-ECCs demonstrates three gaps in this research area: (i) non-inclusion of memory characteristics, (ii) lack of parameterization, and (iii) lack of standardization.

Regarding the target memory characteristics, the analysis was restricted to the relationship between data bits versus redundancy. No analyzed work considered in its multiobjective metrics the physical costs of implementing and operating the memories, such as the energy consumption for writing and reading the codeword; i.e., the implementation and operation analyses were restricted to the encoding and decoding circuits.

The multiobjective metric must have parameterizable objectives to represent target application requirements better. For example, power consumption may be of greater importance than the area occupied for battery-powered applications; thus, attributing weights for each parameter enables us to reach the most promising ECC for a given target application.

Each work chooses a multiobjective metric concerning relevant criteria or objectives for comparing the ECCs. However, there is no standardization of multiobjective metrics that could consider the target application, and the lack of standardization difficult the comparison of ECCs from different works and often leads to biased analysis.

8. Trends on matrix ECCs

The Decimal Matrix Coding with the Encoder Reuse Technique, ECC formats employing three coding axes, and 3D-ECCs deserve special attention as recent matrix code trends.

8.1. Decimal Matrix Coding (DMC) and Encoder Reuse Technique (ERT)

DMC detects errors using decimal integer addition followed by decimal integer subtraction. This decimal algorithm maximizes the error detection efficacy, enhancing memory reliability. DMC divides the matrix row into pairs of size-*n* words and, for each pair, performs a sum adding $n + 1$ redundancy bits producing the codeword. The decoding process employs the same summing circuit of the encoder followed by a decimal integer subtraction. Therefore, there is a tendency to use DMC together with the Encoder Reuse Technique (ERT), which implements the encoder functionally inside the decoder circuit, optimizing the IC area. ERT is founded on the fact that the basic encoding tasks are replicated on the decoder, enabling timing sharing.

Several authors [57,58,63,65,68,71,75] comment on using DMC with ERT or propose ERT architectures. Manoj and Babu [63] and Yedere and Pamula [65] use ERT for implementing DMC encoder/

decoder circuits, concluding the shared circuits have a fundamental role in yield improvement with fewer area costs when compared to the conventional technique. Anitha and Jeevidha [58] proposed a DMC architecture with ERT for reducing area consumption without disturbing the encoding or decoding process. Ahilan and Deepa [57] analyzed the reduction of logic gates in an FPGA employing ERT on a DMC encoder/decoder circuit; their implementation reached a reduction of 67 %; i.e., when using ERT, the system reduced the initial 336 to 112 logic gates. Finally, the authors in [68,75] only comment on the advantages of ERT without implementing a target architecture.

It is worth mentioning that although all works use ERT with DMC, designers can explore this technique in any ECC encoder and decoder architecture. For instance, Athira and Yamuna [71] proposed an ERT architecture that enables the encoder or decoder to employ an enable signal. According to this signal, the encoder or decoder operates while the other remains idle. The authors concluded that using ERT reduces the area and energy consumption of codes like Reed-Muller, Hamming, and DMC significantly concerning a conventional implementation.

8.2. Matrix ECCs with three coding axes

This SLR shows that some works explore three coding axes to increase the error correction and detection efficacy, consequently raising the coding and decoding complexities.

Fig. 16 exemplifies this technique considering a data region with the same number of rows and columns (in green) and redundancy implemented only with parity bits. These parity bits are calculated in vertical (in red), horizontal (in blue), and diagonal (in yellow) directions.

Examples of matrix ECCs with three coding axes are found in works [60,68,70,79,81]. No unique terminology is applied to this technique; the works use the terms 3D Parity Check, Horizontal Vertical Diagonal (HVD), or Multidirectional Parity-Check codes. Some variations exist due to previous work enhancements, resulting in Horizontal Vertical Double Bit Diagonal (HVDD) and Horizontal Vertical Parity and Diagonal Hamming (HVPDH) codes. Although some of these works define the three coding axes as 3D coding, we emphasize that none of these codes deal with the codeword in three dimensions but only in a single plane.

8.3. 3D-ECCs

The first 1D-ECCs were suitable for protecting information transmitted serially and stored in registers. Likewise, 2D-ECCs better mitigate errors in 2D memories. Memories with a greater information volume are organized in a 3D format, often structured on several chips and interconnected with Through Silicon Vias (TSVs). While works such as [84,104–106] use code organizations with one or two dimensions to mitigate faults in 3D memories, the works [107,108] propose 3D-ECCs logically organized in a matrix of three-dimensional memory, being more effective in mitigating 3D memory faults.

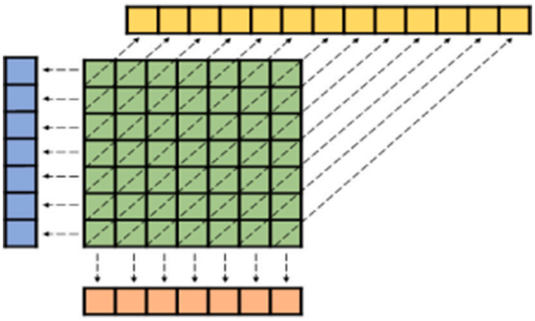


Fig. 16. Encoding a data matrix vertically, horizontally, and diagonally adding parity bits.

Janvars and Farkaš [107] explain the idea of implementing a multidimensional code using parity bits through a minimum distance of 2^d, where d is the code dimension, e.g., for a 3D-ECC, the minimum distance is 8. Let n be the number of bits of each dimension of a cubic codeword; then, the proposed 3D-ECC is an n^3-bit code with $(n-1)^3$ data bits and $3 \times (n-1)^2 + 2 \times (n-1) + n$ redundancy bits. Although the authors detailed a 3D-ECC, they analyzed codes of up to 6D-ECC. Mittelholzer [108], instead of proposing a new 3D-ECC, developed a framework to obtain analytical estimates of the Bit-Error-Rate (BER) performance of 3D product codes under iterative decoding. The author showed that the BER performance has a limiting behavior, and the decoding is successful for sub-limited error probabilities.

Inserting a third coding plane increases the number of redundancy bits in the codeword. To analyze the code scalability when moving to 3D format, we compared the growth of codewords quadratically and cubic structured for 2D and 3D codes, respectively; consequently, both geometric figures grow equally in all dimensions. Additionally, we choose parity bit encoding associated with each region, similar to the 3D-ECC proposed in [107]. Fig. 17(a) shows the explored 2D format containing four regions: (D) square data matrix; (P_R) and (P_C) row and column parity vectors of matrix D, respectively; and (P_{RC}) parity bit of the P_R and P_C intersection. Fig. 17(b) illustrates the evaluated 3D format containing eight regions: (D) cubic data matrix; (P_{XY}), (P_{XZ}), and (P_{YZ}) 2D square parity matrices referring to the XY, XZ, and YZ planes of the D matrix; (P_X), (P_Y) and (P_Z) parity vectors, resulting from the intersection of the XY and XZ, XY and YZ, and XZ and YZ planes, respectively; and (P_{XYZ}) parity bit resulting from the crossing of the P_X, P_Y, and P_Z vectors.

Fig. 18 shows that increasing n (Fig. 17), the number of redundancy bits in 2D- and 3D-ECCs rises linearly and exponentially, respectively. However, Fig. 19 depicts that the exponential growth of redundancy for 3D-ECCs does not significantly affect the data rate (dr - refer to Section 5.2). Additionally, the curve Δ shows that the dr difference between the 2D and 3D codewords has an approximately logarithmic decrease, demonstrating the 3D-ECC scalability.

9. Conclusions

We conducted a Systematic Literature Review (SLR) resulting in a 35 work selection; a thorough analysis of these works allowed us to consolidate five features of 2D-ECC studies used for mitigating memory errors: (i) 2D-ECC classification; (ii) data size and redundancy metrics; (iii) target application; (iv) analysis methods; and (v) trend on matrix ECCs.

We classified 2D-ECCs according to their coding model. Depending on the codeword structure, 2D-ECCs are classified as Product Codes (PC), Extended Product Codes (EPC), or Mixed Codes (MC), if a single bitflip changes two or more encoding regions; otherwise, they are classified as Straightforward 2D-ECC (S2E).

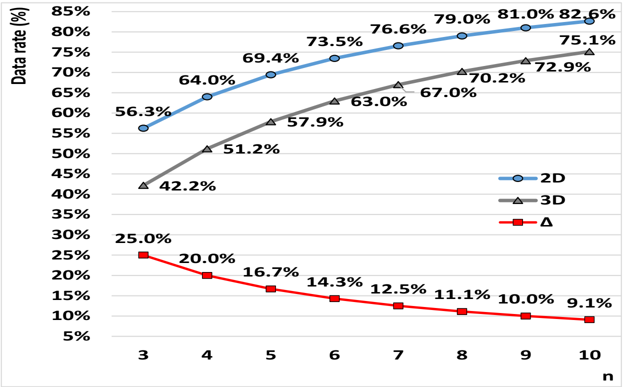


Fig. 17. (a) 2D- and (b) 3D-ECC, considering square and cubic structures with n^2 and n^3 data bits (D), respectively.

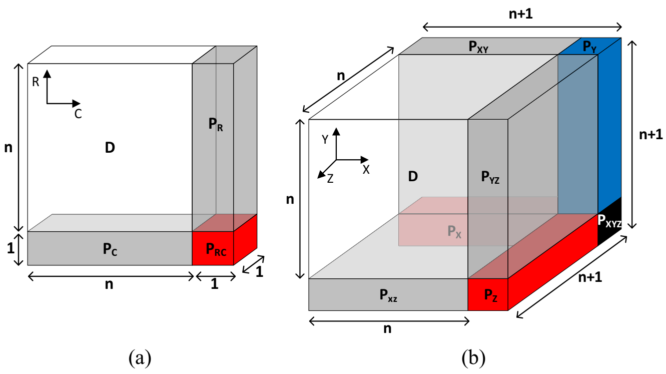


Fig. 18. The increase of redundancy rate considering 2D- and 3D-ECCs with parity check bits; *n* is the number of bits in a single axis of D.

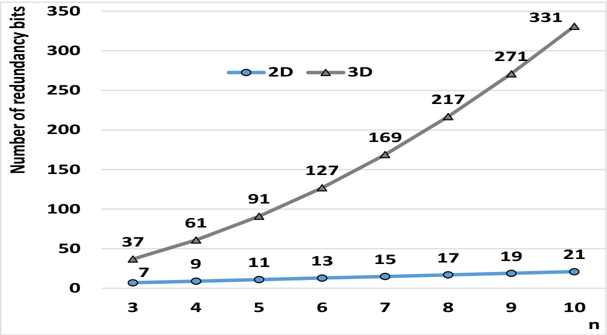


Fig. 19. Data rates of 2D- and 3D-ECCs, and the difference between data rates (Δ) considering only parity check bits; *n* is the number of bits in a single axis of D.

When evaluating the codeword data size, we noticed that more than half of the works being assessed employ 32-bit data ECCs; besides, they use about 50 % of the codeword for redundancy bits, even though there are cases where the redundancy rate is larger, as in critical or space applications. Besides, there is a close relationship between the 2D-ECC classification and the target application; e.g., ECCs classified as PC and EPC are more likely to be used in space applications.

The validation and analysis of the 2D-ECC effectiveness are usually performed by injecting adjacent, exhaustive, or burst errors. 75 % of the works use 45 nm or 65 nm CMOS manufacturing technology to compare encoder and decoder synthesis costs, such as area consumed and power dissipated; additionally, some works propose the use of multi-objective metrics considering error detection and/or correction efficacy together with synthesis costs; however, so far, there is no standardization of these metrics.

The SLR showed some trends in 2D-ECCs, such as using Decimal Matrix Coding (DMC) with the Encoder Reuse (ERT) technique. Also, several authors use three coding axes in the same 2D plane, and some are working on ECCs with three or more dimensions.

The ECC Efficacy and efficiency comparison requires a standardized verification methodology, allowing us to extract advantages/disadvantages and tradeoffs of each proposal regardless of the specific bias defined in each work. However, this SLR showed a diversity of experiments and metrics that hinder fair comparative analysis among ECCs. Therefore, we realize that the results presented here can help us to understand different approaches for finding a standardization of great value for future 2D-ECC proposal analyses.

The systematic review revealed several research gaps in the ECC area; some of these studies are: (i) Analysis of real error patterns occurring in 3D memories since there is a significant variation in the incidence of errors according to memory layer depth; this analysis would enable us to construct a heterogeneous 3D-ECC model to mitigate the error incidence rate in each memory layer; (ii) Exploitation of error patterns and incidence rates based on the manufacturing technology, allowing to understand historical variations of ECC requirements; (iii) Study to determine an ECC evaluation metric that considers all the code characteristics, as well as its application and organization; (iv) The design and implementation of fault-tolerance systems that dynamically change the memory ECCs according to error incidence, allowing us to manage tradeoffs such as reliability and energy consumption; and (v) Employing artificial intelligence techniques to learn and rearrange the ECC bit structure and the error correction algorithm used on top of the ECC structure.

Declaration of competing interest

There is no conflict of interest in this submission.

Data availability

No data was used for the research described in the article.

References

[1] R. Giterman, L. Atias, A. Teman, Area and energy-efficient complementary dual-modular redundancy dynamic memory for space applications, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 25 (2) (2017) 502–509. Feb.

[2] P. Li, W. Dang, T. Qin, Z. Zhang, C. Lv, A competing risk model of reliability analysis for NAND-based SSDs in space application, IEEE Access 7 (2019) 23430–23441.

[3] A. Agnesina, J. Yamaguchi, C. Krutzik, J. Carson, J. Yang-Scharlotta, S. Lim, A COTS-based novel 3-D DRAM memory cube architecture for space applications, IEEE Trans. Very Large Scale Integr. VLSI Syst. 28 (9) (Sep. 2020) 2055–2068.

[4] G. Kinoshita, C. Kleiner, E. Johnson, Radiation induced regeneration through the P-N junction isolation in monolithic I/C's, IEEE Trans. Nucl. Sci. 12 (5) (Oct. 1965) 83–90.

[5] C. Kleiner, G. Kinoshita, E. Johnson, Simulation and verification of transient nuclear radiation effects on semiconductor electronics, IEEE Trans. Nucl. Sci. 11 (5) (Nov. 1964) 82–104.

[6] C. Rosenberg, D. Gage, R. Caldwell, G. Hanson, Charge-control equivalent circuit for predicting transient radiation effects in transistors, IEEE Trans. Nucl. Sci. 10 (5) (Nov. 1963) 149–158.

[7] L. Atias, A. Teman, R. Giterman, P. Meinerzhagen, A. Fish, A low-voltage radiation-hardened 13T SRAM bitcell for ultralow power space applications, IEEE Trans. Very Large Scale Integr. VLSI Syst. 24 (8) (Aug. 2016) 2622–2633.

[8] D. Chen, E. Wilcox, R. Ladbury, C. Seidleck, H. Kim, A. Phan, K. LaBel, Heavy ion and proton-induced single event upset characteristics of a 3-D NAND flash memory, IEEE Trans. Nucl. Sci. 65 (1) (Jan. 2018) 19–26.

[9] T. Li, H. Liu, H. Yang, Design and characterization of SEU hardened circuits for SRAM-based FPGA, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 27 (6) (2019) 1276–1283. Jun.

[10] W. Wei, K. Namba, Y. Kim, F. Lombardi, A novel scheme for tolerating single event/multiple bit upsets (SEU/MBU) in non-volatile memories, IEEE Transactions on Computers 65 (3) (2016) 781–790, 1 Mar.

[11] I. Villalta, U. Bidarte, J. Cornejo, J. Lazaro, A. Astarloa, Estimating the SEU failure rate of designs implemented in FPGAs in presence of MCUs, Microelectron. Reliab. 78 (1) (Nov. 2017) 85–92.

[12] A. Neale, M. Jonkman, M. Sachdev, Adjacent-MBU-tolerant SEC-DED-TAEC-yAED codes for embedded SRAMs, IEEE Trans. Circuits Syst. Express Briefs 62 (4) (Apr. 2015) 387–391.

[13] M. Nicolaidis, Soft Errors in Modern Electronic Systems 41, Springer Science, 2001. ISSN 0929-1296.

[14] R. Liu, D. Mahalanabis, H. Barnaby, S. Yu, Investigation of single-bit and multiple-bit upsets in oxide RRAM-based 1T1R and crossbar memory arrays, IEEE Trans. Nucl. Sci. 62 (5) (Oct. 2015) 2294–2301.

[15] Y. Fang, A. Oates, Characterization of single bit and multiple cell soft error events in planar and FinFET SRAMs, IEEE Trans. Device Mater. Reliab. 16 (2) (Jun. 2016) 132–137.

[16] T. Kato, T. Yamazaki, K. Maruyama, T. Soeda, H. Itsuji, D. Kobayashi, K. Hirose, H. Matsuyama, The impact of multiple-cell charge generation on multiple-cell upset in a 20-nm bulk SRAM, IEEE Trans. Nucl. Sci. 65 (8) (Aug. 2018) 1900–1907.

[17] T. Kato, T. Yamazaki, N. Saito, H. Matsuyama, Neutron-induced multiple-cell upsets in 20-nm bulk SRAM: angular sensitivity and impact of multiwell potential perturbation, IEEE Trans. Nucl. Sci. 66 (7) (Jul. 2019) 1381–1389.

[18] Q. Shao, S. Yang, X. Gou, Formal analysis of multiple-cell upset failure based on common cause failure theory, IEEE Trans. Reliab. 70 (4) (Dec. 2021) 1495–1509.

[19] A. Ullah, P. Reviriego, A. Sánchez-Macián, J. Maestro, Multiple cell upset injection in BRAMs for xilinx FPGAs, IEEE Trans. Device Mater. Reliab. 18 (4) (Dec. 2018) 636–638.

[20] A. Pérez-Celis, M. Wirthlin, Statistical method to extract radiation-induced multiple-cell upsets in SRAM-based FPGAs, IEEE Trans. Nucl. Sci. 67 (1) (Jan. 2020) 50–56.

[23] M. Ebrahimi, P. Rao, R. Seyyedi, M. Tahoori, Low-cost multiple bit upset correction in SRAM-based FPGA configuration frames, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 24 (3) (2016) 932–943. Mar.

[24] B. Liu, L. Cai, Monte Carlo reliability model for single-event transient on combinational circuits, IEEE Trans. Nucl. Sci. 64 (12) (Dec. 2017) 2933–2937.

[25] E. Keren, S. Greenberg, N. Yitzhak, D. David, N. Refaeli, A. Haran, Characterization, mitigation of single-event transients in xilinx 45-nm SRAM-based FPGA, IEEE Trans. Nucl. Sci. 66 (6) (Jun. 2019) 946–954.

[26] J. Chen, J. Yu, P. Yu, B. Liang, Y. Chi, Characterization of the effect of pulse quenching on single-event transients in 65-nm twin-well, triple-well CMOS technologies, IEEE Trans. Device Mater. Reliab. 18 (1) (Mar. 2018) 12–17.

[27] Y. Chi, R. Song, S. Shi, B. Liu, L. Cai, C. Hu, G. Guo, Characterization of single-event transient pulse broadening effect in 65 nm bulk inverter chains using heavy ion microbeam, IEEE Trans. Nucl. Sci. 64 (1) (Jan. 2017) 119–124.

[28] A. Sánchez-Macián, P. Reviriego, J. Tabero, A. Regadío, J. Maestro, SEFI protection for nanosat 16-bit Chip onboard computer memories, IEEE Trans. Device Mater. Reliab. 17 (4) (Dec. 2017) 698–707.

[29] A. Sánchez-Macián, P. Reviriego, J. Maestro, Combined SEU, SEFI protection for memories using orthogonal Latin Square codes, IEEE Trans. Circuits Syst. I, Reg. Papers 63 (11) (Nov. 2016) 1933–1943.

[30] L. Artola, S. Ducret, F. Advent, G. Hubert, J. Mekki, SEFI modeling in readout integrated circuit induced by heavy ions at cryogenic temperatures, IEEE Trans. Nucl. Sci. 66 (1) (Jan. 2019) 452–457.

[31] R. Secondo, R. Alía, P. Peronnard, M. Brugger, A. Masi, S. Danzeca, A. Merlenghi, J.-R. Vaillé, L. Dusseau, Analysis of SEL on commercial SRAM memories, mixed-field characterization of a latchup detection circuit for LEO space applications, IEEE Trans. Nucl. Sci. 64 (8) (Aug. 2017) 2107–2114.

[32] A. Youssef, L. Artola, S. Ducret, G. Hubert, Compact modeling of single-event latchup of integrated CMOS circuit, IEEE Trans. Nucl. Sci. 66 (7) (Jul. 2019) 1510–1515.

[33] P. Wang, A. Sternberg, B. Sierawski, E. Zhang, K. Warren, A. Tonigan, R. Brewer, N. Dodds, G. Vizkelethy, S. Jordan, D. Fleetwood, R. Reed, R. Schrimpf, Sensitive-volume model of single-event latchup for a 180-nm SRAM test structure, IEEE Trans. Nucl. Sci. 67 (9) (Sep. 2020) 2015–2020.

[34] M. Mauguet, N. Andrianjohany, D. Lagarde, L. Gouyet, L. Azema, N. Chatry, X. Marie, R. Marec, P. Calvel, D. Standarovski, R. Ecoffet, Single-event latchup in a CMOS-based ASIC using heavy ions, laser pulses, coupled simulation, IEEE Trans. Nucl. Sci. 66 (7) (Jul. 2019) 1516–1522.

[35] R. Hamming, Error detecting, error correcting codes, Bell Syst. Tech. J. 29 (2) (Apr. 1950) 147–160.

[36] S. Liu, J. Li, P. Reviriego, M. Ottavi, L. Xiao, A double error correction code for 32-bit data words with efficient decoding, IEEE Trans. Device Mater. Reliab. 18 (1) (Mar. 2018) 125–127.

[37] S. Liu, P. Reviriego, L. Xiao, Evaluating direct compare for double error-correction codes, IEEE Trans. Device Mater. Reliab. 17 (4) (Dec. 2017) 802–804.

[38] P. Reviriego, S. Liu, L. Xiao, J. Maestro, An efficient single, double-adjacent error correcting parallel decoder for the (24,12) extended golay code, IEEE Trans. Very Large Scale Integr. VLSI Syst. 24 (4) (Apr. 2016) 1603–1606, https://doi.org/10.1109/TVLSI.2015.2465846.

[39] H. Farbeh, F. Mozafari, M. Zabihi, S. Miremadi, RAW-tag: replicating in altered cache ways for correcting multiple-bit errors in tag array, IEEE Trans. Dependable Secure Comput. 16 (4) (2019) 651–664, 1 Jul.-Aug.

[40] A. Olazábal, J. Guerra, Multiple cell upsets inside aircrafts. New fault-tolerant architecture, IEEE Trans. Aerosp. Electron. Syst. 55 (1) (Feb. 2019) 332–342.

[41] A. Erozan, E. Çavus, An EG-LDPC Based 2-dimensional error correcting code for mitigating MBUs of SRAM memories, in: Proceedings of the FPGA World Conference, 2015, pp. 21–26.

[42] G. Alexandre, J. Soares, G. Thé, Systematic review of 3D facial expression recognition methods, Pattern Recogn. 100 (1) (Apr. 2020) 1–16.

[43] A. Bajaj, O. Sangwan, A systematic literature review of test case prioritization using genetic algorithms, IEEE Access 7 (2019) 126355–126375.

[44] M. Al-Sarem, W. Boulila, M. Al-Harby, J. Qadir, A. Alsaeedi, Deep learning-based rumor detection on microblogging platforms: a systematic review, IEEE Access 7 (2019) 152788–152812.

[45] K. Brito, A. de Lima, S. Ferreira, V. Burégio, V. Garcia, S. Meira, Evolution of the web of social machines: a systematic review, research challenges, IEEE Trans. Comput. Soc. Syst. 7 (2) (Apr. 2020) 373–388.

[46] D. van der Linden, I. Hadar, IEEE Transactions on Software Engineering 45 (8) (2019) 736–759, 1 Aug.

[47] E. Mourão, M. Kalinowski, L. Murta, E. Mendes, C. Wohlin, Investigating the use of a hybrid search strategy for systematic reviews, in: Proceedings of the ACM/IEEE International Symposium on Empirical Software Engineering, Measurement (ESEM), 2017, pp. 193–198.

[48] K. Felizardo, E. Mendes, M. Kalinowski, E. Souza, N. Vijaykumar, Using forward snowballing to update systematic reviews in software engineering, in: Proceedings of the International Symposium on Empirical Software Engineering, Measurement (ESEM), 2016, pp. 1–6.

[49] C. Wohlin, Second-generation systematic literature studies using snowballing, in: Proceedings of the International Conference on Evaluation, Assessment in Software Engineering (EASE), 2016, pp. 1–6.

[50] C. Wohlin, Guidelines for snowballing in systematic literature studies, a replication in software engineering, in: Proceedings of the International Conference on Evaluation, Assessment in Software Engineering (EASE), 2014, pp. 1–10.

[51] C. Argyrides, H. Zarandi, D. Pradhan, Matrix codes: multiple bit upsets tolerant method for SRAM memories, in: Proceedings of the IEEE International Symposium on Defect, Fault-Tolerance in VLSI Systems (DFT), 2007, pp. 340–348.

[52] C. Argyrides, P. Reviriego, D. Pradhan, J. Maestro, Matrix-based codes for adjacent error correction, IEEE Trans. Nucl. Sci. 57 (4) (Aug. 2010) 2106–2111.

[53] C. Argyrides, D. Pradhan, T. Kocak, Matrix codes for reliable, cost efficient memory chips, IEEE Transactions on Very Large Scale Integration (VLSI) Systems 19 (3) (2011) 420–428. Mar.

[54] R. Afrin, M. Sadi, An efficient approach to enhance memory reliability, in: Proceedings of the International Conference on Advances in Electrical Engineering (ICAEE), 2017, pp. 170–175, 1 Aug.

[55] J. Gracia-Moran, L. Saiz-Adalid, J. Baraza-Calvo, P. Gil-Vicente, Correction of adjacent errors with low redundant matrix error correction codes, in: Proceedings of the Latin-American Symposium on Dependable Computing (LADC), 2018, pp. 107–114.

[56] J. Li, L. Xiao, J. Guo, X. Cao, Efficient implementations of multiple bit burst error correction for memories, in: Proceedings of the IEEE International Conference on Solid-State, Integrated Circuit Technology (ICSICT), 2018, pp. 1–3.

[57] A. Ahilan, P. Deepa, Modified Decimal Matrix Codes in FPGA configuration memory for multiple bit upsets, in: Proceedings of the International Conference on Computer Communication, Informatics (ICCCI), 2015, pp. 1–5.

[58] B. Anitha, B. Jeevidha, Low overhead decimal matrix code with dynamic network on chip against multiple cell upsets, in: Proceedings of the International Conference on Innovations in Information, Embedded, Communication Systems (ICIIECS), 2015, pp. 1–6.

[59] S. Liu, L. Xiao, J. Guo, Z. Mao, Fault secure encoder, decoder designs for matrix codes, in: Proceedings of the International Conference on Computer-Aided Design, Computer Graphics (CAD/Graphics), 2015, pp. 181–185.

[60] M. Rahman, M. Sadi, S. Ahammed, J. Jurjens, Soft error tolerance using horizontal-vertical-double-bit diagonal parity method, in: Proceedings of the International Conference on Electrical Engineering, Information Communication Technology (ICEEICT), 2015, pp. 1–6.

[61] H. Castro, J. da Silveira, A. Coelho, F. Silva, P. Magalhães, O. de Lima, A correction code for multiple cells upsets in memory devices for space applications, in: Proceedings of the IEEE International New Circuits, Systems Conference (NEWCAS), 2016, pp. 1–4.

[62] S. Mandal, R. Paul, S. Sau, A. Chakrabarti, S. Chattopadhyay, A novel method for soft error mitigation in FPGA using modified matrix code, IEEE Embed. Syst. Lett. 8 (4) (Dec. 2016) 65–68.

[63] S. Manoj, C. Babu, Improved error detection, correction for memory reliability against multiple cell upsets using DMC & PMC, in: Proceedings of the IEEE Annual India Conference (INDICON), 2016, pp. 1–6.

[64] M. Sundary, V. Logisvary, Multiple error detection, correction over GF(2m) using novel cross parity code, in: Proceedings of the International Conference on Intelligent Systems, Control (ISCO), 2016, pp. 1–6.

[65] N. Yedere, V. Pamula, Performance analysis of decimal matrix code, modified decimal matrix code, in: Proceedings of the IEEE International Conference on Computational Intelligence, Computing Research (ICCIC), 2016, pp. 1–5, 1 Jul.-Aug.

[66] S. Kamatchi, C. Vivekanandan, B. Thilagavathi, Detection, correction of multiple upsets in memories using modified decimal matrix code, J. Comput. Theor. Nanosci. 14 (3) (Mar. 2017) 1543–1547.

[67] S. Liu, L. Xiao, J. Li, Y. Zhou, Z. Mao, Low redundancy matrix-based codes for adjacent error correction with parity sharing, in: Proceedings of the International Symposium on Quality Electronic Design (ISQED), 2017, pp. 76–80.

[68] P. Raha, M. Vinodhini, N. Murty, Horizontal-vertical parity, diagonal hamming based soft error detection, correction for memories, in: Proceedings of the International Conference on Computer Communication, Informatics (ICCCI), 2017, pp. 1–5.

[69] F. Silva, W. Freitas, J. Silveira, O. Lima, F. Vargas, C. Marcon, An efficient, low-cost ECC approach for critical-application memories, in: Proceedings of the Symposium on Integrated Circuits, Systems Design (SBCCI), 2017, pp. 198–203.

[70] S. Tambatkar, S. Menon, V. Sudarshan, M. Vinodhini, N. Murty, Error detection, correction in semiconductor memories using 3D parity check code with hamming code, in: Proceedings of the International Conference on Communication, Signal Processing (ICCSP), 2017, pp. 0974–0978.

[71] J. Athira, B. Yamuna, FPGA implementation of an area efficient matrix code with encoder reuse method, in: Proceedings of the International Conference on Communication, Signal Processing (ICCSP), 2018, pp. 0254–0257.

[72] R. Goerl, P. Villa, L. Poehls, E. Bezerra, F. Vargas, An efficient EDAC approach for handling multiple bit upsets in memory array, Microelectron. Reliab. 88–90 (1) (Sep. 2018) 214–218.

[73] F. Silva, J. Silveira, J. Silveira, C. Marcon, F. Vargas, O. Lima Jr., An extensible code for correcting multiple cell upset in memory arrays, J. Electron. Test. 34 (4) (Jul. 2018) 417–433.

[74] P. Magalhães, O. Alcântara, J. Silveira, PHICC: an error correction code for memory devices, in: Proceedings of the Symposium on Integrated Circuits, Systems Design (SBCCI), 2019, pp. 1–6.

[75] M. Priya, M. Vijay, Error detection, correction for SRAM systems using improved redundant matrix code, in: Proceedings of the International Conference on Recent Advances in Energy-efficient Computing, Communication (ICRAECC), 2019, pp. 1–8.

[76] F. Zhang, J. Yan, L. Ma, Y. Li, W. Gao, Multi-bit upset mitigation with double matrix codes in memories for space applications, in: Proceedings of the IEEE International Conference on Unmanned Systems, Artificial Intelligence (ICUSAI), 2019, pp. 146–149.
[77] D. Freitas, D. Mota, R. Goerl, C. Marcon, F. Vargas, J. Silveira, J. Mota, PCoSA: a product error correction code for use in memory devices targeting space applications, Integr. VLSI J. 74 (1) (Sep. 2020) 71–80.
[78] K. Kumar, N. Reddy, P. Shanmukh, M. Vinodhini, Matrix based error detection and correction using minimal parity bits for memories, in: Proceedings of the IEEE International Conference on Distributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER), 2020, pp. 100–104.
[79] K. Neelima, C. Subhas, Efficient adjacent 3D parity error detection, correction codes for embedded memories, in: Proceedings of the IEEE International Conference on Electronics, Computing, Communication Technologies (CONECCT), 2020, pp. 1–5.
[80] T. Rohde, J. Martins, Multi-bit-upset memory using new error correction code methodology, in: Proceedings of the IEEE Latin American Symposium on Circuits & Systems (LASCAS), 2020, pp. 1–4.
[81] G. Sai, K. Avinash, L. Naidu, M. Rohith, M. Vinodhini, Diagonal Hamming Based Multi-Bit Error Detection, Correction Technique for Memories, in: Proceedings of the International Conference on Communication, Signal Processing (ICCSP), 2020, pp. 0746–0750.
[82] F. Silva, A. Muniz, J. Silveira, C. Marcon, CLC-A: an adaptive implementation of the column line code (CLC) ECC, in: Proceedings of the Symposium on Integrated Circuits, Systems Design (SBCCI), 2020, pp. 1–6.
[83] F. Silva, W. Freitas, J. Silveira, C. Marcon, F. Vargas, Extended matrix region selection code: an ECC for adjacent multiple cell upset in memory arrays, Microelectron. Reliab. 106 (1) (Mar. 2020) 1–9.
[84] D. Freitas, D. Mota, C. Marcon, J. Silveira, J. Mota, LPC: an error correction code for mitigating faults in 3D memories, IEEE Trans. Comput. 70 (11) (Nov. 2021) 2001–2012.
[85] P. Sen, M. Sadi, N. Ashab, D. Rossi, in: Proceedings of the International Conference on Electronics, Communications and Information Technology (ICECIT), 2021, pp. 1–4.
[86] D. Freitas, C. Marcon, J. Silveira, L. Naviner, J. Mota, New decoding techniques for modified product code used in critical applications, Microelectron. Reliab. 128 (114444) (Jan. 2022) 1–14.
[87] D. Freitas, J. Silveira, C. Marcon, L. Naviner, J. Mota, OPCoSA: an optimized product code for space applications, Integr. VLSI J. 84 (May 2022) 131–141.
[88] P. Elias, Error-free coding, Trans. IRE Prof. Group Inf. Theory 4 (4) (Sep. 1954) 29–37.
[89] F. Macwilliams, N. Sloane, in: The Theory of Error-Correcting Codes, 3rd ed. 16, North-Holland, 1977, pp. 568–570.
[90] R. Zaragoza, in: The Art of Error Correcting Coding, 2nd ed., Wiley, 2006, pp. 170–201.
[91] T. Moon, in: Error Correcting Code – Mathematical Methods, Algorithms, 1st ed. 1, Wiley, 2005, pp. 430–432.
[92] C. Ogden, M. Mascagni, The impact of soft error event topography on the reliability of computer memories, IEEE Trans. Reliab. 66 (4) (Dec. 2017) 966–979.
[93] M. Wirthlin, D. Lee, G. Swift, H. Quinn, A method, case study on identifying physically adjacent multiple-cell upsets using 28-nm, interleaved, SECDED-protected arrays, IEEE Trans. Nucl. Sci. 61 (6) (Dec. 2014) 3080–3087.
[94] P. Rao, M. Ebrahimi, R. Seyyedi, M. Tahoori, Protecting SRAM-based FPGAs against multiple bit upsets using erasure codes, in: Proceedings of the Design Automation Conference (DAC), 2014, pp. 1–6.
[95] H. Quinn, K. Morgan, P. Graham, J. Krone, M. Caffrey, Static proton, heavy ion testing of the Xilinx Virtex-5 device, in: Proceedings of the IEEE Radiation Effects Data Workshop, 2007, pp. 177–184.
[96] J.-L. Leray, J. Baggio, V. Ferlet-Cavrois, O. Flament, Atmospheric neutron effects in advanced microelectronics, standards, applications, in: Proceedings of the International Conference on Integrated Circuit Design, Technology (ICICDT), 2004, pp. 311–321.
[97] S. Satoh, Y. Tosaka, S. Wender, Geometric effect of multiple-bit soft errors induced by cosmic ray neutrons on DRAM's, IEEE Electron Device Lett. 21 (6) (June 2000) 310–312.
[98] J. Gracia-Morán, L. Saiz-Adalid, D. Gil-Tomás, P. Gil-Vicente, Improving error correction codes for multiple-cell upsets in space applications, IEEE Trans. Very Large Scale Integr. VLSI Syst. 26 (10) (Oct. 2018) 2132–2142.
[99] E. Fujiwara, Code Design for Dependable Systems – Theory, Practical Applications 1, Wiley Interscience, 2006. ISBN: 0471756180, 9780471756187.
[100] G. He, S. Zheng, N. Jing, A hierarchical scrubbing technique for SEU mitigation on SRAM-based FPGAs, IEEE Trans. Very Large Scale Integr. VLSI Syst. 28 (10) (Oct. 2020) 2134–2145.
[101] R. Zhang, L. Xiao, J. Li, X. Cao, L. Li, An adjustable, fast error repair scrubbing method based on xilinx essential bits technology for SRAM-based FPGA, IEEE Trans. Reliab. 69 (2) (Jun. 2020) 430–439.
[102] W. Wang, C. Ho, Y. Chang, T. Kuo, P. Lin, Scrubbing-aware secure deletion for 3-D NAND flash, IEEE Trans. Comput. Aided Des. Integr. Circuits Syst. 37 (11) (Nov. 2018) 2790–2801.
[103] A. Stoddard, A. Gruwell, P. Zabriskie, M. Wirthlin, A hybrid approach to FPGA configuration scrubbing, IEEE Trans. Nucl. Sci. 64 (1) (Jan. 2017) 497–503.
[104] A. Macián, F. Herrero, J. Maestro, Reliability of 3D memories using orthogonal Latin square codes, Microelectron. Reliab. 95 (1) (Apr. 2019) 74–80.
[105] H. Chen, C. Wu, T. Mudge, C. Chakranbarti, RATT-ECC: rate adaptative two-tiered error correction codes for reliable 3D die-stacked memory, ACM Trans. Archit. Code Optim. 1 (24) (Sep. 2016) 1–24.
[106] H. Han, J. Chung, J. Yang, READ: reliability enhancement in 3D-memory exploiting asymmetric SER distribution, IEEE Trans. Comput. 67 (8) (Aug. 2018) 1193–1201.
[107] T. Janvars, P. Farkaš, Hard decision decoding of single parity turbo product code with N-level quantization, in: Proceedings of the International Conference on Telecommunications, Signal Processing (TSP), 2015, pp. 1–6.
[108] T. Mittelholzer, Performance analysis of iteratively decoded 3-dimensional product codes, in: Proceedings of the IEEE International Symposium on Information Theory (ISIT), 2018, pp. 1321–1325.
[109] D. Radaelli, H. Puchner, S. Wong, S. Daniel, Investigation of multi-bit upsets in a 150nm technology SRAM device, IEEE Trans. Nucl. Sci. 52 (6) (Dec. 2005) 2433–2437.

David Freitas received a bachelor's degree in mechatronics engineering from the Federal Institute of Ceará (IFCE), Fortaleza, Ceará, Brazil, in 2012. He received a Master's degree in electrical engineering from the Federal University of Paraíba (UFPB), Campina Grande, Paraíba, Brazil, in 2016. He is a Ph.D. student at the Federal University of Ceará (UFC) in the Teleinformatics Engineering Department since 2018. Since 2017, he has been with the Federal Institute of Ceará, where he is currently a Professor. His current research interests are in electronic circuits, Petri Nets, Embedded Systems Hardware, Lead Acid Batteries, Digital Circuit Design (ASIC and FPGA), Fault-Tolerant Systems, and Error Correction Codes.

César Marcon received the Ph.D. degree in computer science from Federal University of Rio Grande do Sul (UFRGS), Brazil, in 2005. He has been a Professor at the School of Computer Science, Pontifical Catholic University of Rio Grande do Sul (PUCRS), Brazil, since 1995. He has >150 papers published in prestigious journals and conference proceedings. Professor Marcon is a Senior Member of the Institute of Electrical and Electronics Engineers (IEEE) and a Brazilian Computer Society member (SBC). He is an Advisor of Ph.D. graduate students at the Graduate Program in Computer Science of PUCRS.

Jarbas Silveira received the Ph.D. degree in teleinformatics engineering from the Federal University of Ceará (UFC) in 2015. He has been an Adjunct Professor with the Teleinformatics Department, UFC, Brazil, since 2009, where he is also with the Engineering Laboratory Computer Systems. His research interests are in the areas of embedded systems on digital circuits, computer architecture, on-chip communication architectures, fault tolerance, and real-time systems.

Lirida Naviner (M'98-SM'06) received the B.Sc. and M.Sc. degrees from the Federal University of Paraiba (UFPB), Brazil, in 1988 and 1990, respectively, and the Ph.D. degree from the Ecole Nationale Supérieure des Télécommunications, France, in 1994. From 1994 to 1997, she was a Faculty Member at UFPB. Since 1998, she has been with the Electronics and Communications Department, Télécom ParisTech, where she is currently a Full Professor. She has led several academic and industrial research projects with national/international partners. She has authored/co-authored over 200 scientific papers and is an inventor of eight patents. She is a member of the startup incubator of ParisTech. She has been honored with the title Chevalier dans L'Ordre des Palmes Academiques by French Government.

João Mota received the B.S. degree in physics from the Federal University of Ceará, Fortaleza, Brazil, in 1978, the M.Sc. degree in electrical engineering from the Catholic University of Rio de Janeiro, Rio de Janeiro, Brazil, in 1984, and the Ph.D. degree in electrical engineering from the State University of Campinas, Campinas, Brazil, in 1992. He is currently a Professor at the Federal University of Ceará and the Assistant Director for Interinstitutional Relationships of its Technology Center, founding member of the Brazilian Society of Telecommunications, member of the Brazilian Society of Health Informatics, adviser to the Student Branch of the Institute of Electrical and Electronics Engineers (IEEE) in FUC, member of the Signal Processing Society and IEEE Communications Society.