A Method to Design SEC-DED-DAEC Codes With Optimized Decoding

Pedro Reviriego, *Member, IEEE*, Jorge Martínez, Salvatore Pontarelli, and Juan Antonio Maestro, *Member, IEEE*

***Abstract*—Single error correction–double error detection–double adjacent error correction (SEC-DED-DAEC) codes have been proposed to protect SRAM devices from multiple cell upsets (MCUs). The correction of double adjacent errors ensures that the most common types of MCUs are corrected. At the same time, SEC-DED-DAEC codes require the same number of parity check bits as traditional SEC-DED codes. The main overhead associated with SEC-DED-DAEC codes is the increase in decoding complexity that can impact access time and circuit power and area. In this paper, a method to design SEC-DED-DAEC codes with optimized decoding is presented and evaluated. The proposed scheme starts by setting some constraints on the parity check matrix of the codes. Those constraints are then used to simplify the decoding. The proposed scheme has been implemented and evaluated for different word-lengths. The results show that, for data words of 32 bits, the scheme can be implemented with the same number of parity check bits as SEC-DED codes. For 16 and 64 bits words, an additional parity check bit is required, making the scheme less attractive. With the proposed method, the decoders can be optimized for area or speed. Both implementations are evaluated and compared with existing SEC-DED-DAEC decoders. The results show that the proposed decoders reduce significantly the circuit area, power, and delay.**

***Index Terms*—SEC-DAEC codes, Multiple Cell Upsets (MCUs), error correction codes, SRAM memories.**

I. Introduction

EMBEDDED SRAM memories are an important fraction of the circuit area in modern System on Chip (SoCs) and its importance is expected to increase in the future. Soft errors are a reliability issue for SRAM memories and an increasing percentage of radiation induced soft errors in SRAM memories affect several memory cells causing Multiple Cell Upsets (MCUs) [1]. The presence of MCUs means that traditional Single Error Correction Double Error Detection (SEC-DED) codes [2], can no longer provide an effective protection unless interleaving is used. Interleaving places bits that belong to the same logical word in cells that are physically apart thus ensuring that the errors caused by an MCU affect at most one bit per word [3]. This can be done because the errors caused by an MCU affect nearby cells located close to the impact of the radiation particle [4]. However, the use of interleaving can impact the area, power and delay of the memory and may not be practical for small memories and for some types of memories like Content Addressable Memories [5]–[7]. In those cases, an alternative is to use multi-bit error correction codes. For example in [8], the use of Double Error Correction (DEC) Bose-Chaudhuri-Hocquenghem (BCH) codes was studied. However, these codes require a large number of additional parity bits and a significant decoding complexity. To avoid these issues, codes that can correct adjacent errors can be used as those are the ones caused by MCUs in most cases. In the last years, several codes have been proposed to correct double adjacent errors [6], [9], [10]. These codes are known as Single Error Correction Double Error Detection Double Adjacent Error Correction (SEC-DED-DAEC) codes. Protection to cover also MCUs that affect more than two adjacent cells has also been investigated [11], [12].

The use of SEC-DED-DAEC codes does not typically require additional parity check bits and therefore does not increase the memory size. The main issue for these codes is that they increase decoder complexity as more error patterns have to be identified to correct the errors. The complexity increase impacts the decoder area, power and delay thus affecting the performance of the memory. This is especially relevant for embedded memories as there can be hundreds of them in a SoC and they have to operate a clock speed.

A method to reduce the decoding complexity for some SEC and SEC-DED codes has been recently proposed in [13]. The scheme focuses on codes with constant weight on the data columns of the parity check matrix. This enables a simplification of the error location phase of decoding. In this paper, those ideas are extended so that they can be applied to SEC-DED-DAEC codes that meet some requirements. In addition to presenting the scheme, codes for commonly used word lengths are derived using an automated process. The decoders for these codes are then compared with those of traditional SEC-DED-DAEC codes to illustrate the benefits of the proposed scheme in reducing decoding complexity.

The rest of the paper is organized as follows: Section II presents an overview of existing SEC-DED and SEC-DED-DAEC codes focusing on their decoding. Then in Section III the proposed SEC-DED-DAEC codes and the optimized decoding implementations are presented. In Section IV, the decoders are implemented and compared to existing SEC-DED-DAEC codes in terms of decoding complexity. Finally the conclusions are summarized in Section V.

II. Traditional SEC-DED-DAEC Codes

Most error correction codes used to protect memories are systematic linear block codes [14]. A systematic linear block

Manuscript received March 17, 2014; accepted June 19, 2014. Date of publication July 11, 2014; date of current version September 2, 2014. This work was supported by the Spanish Ministry of Science and Innovation under Grant AYA2009-13300-C03-01.

P. Reviriego, J. Martínez, and J. A. Maestro are with Universidad Antonio de Nebrija, 28040, Madrid, Spain (e-mail: previrie@nebrija.es; jmartine@nebrija.es; jmaestro@nebrija.es).

S. Pontarelli is with the National Inter-University Consortium for Telecommunications (CNIT), 43124 Parma, Italy (e-mail: pontarelli@ing.uniroma2.it).

Digital Object Identifier 10.1109/TDMR.2014.2332364

1530-4388 © 2014 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

code takes a word of k bits and produces a word of n bits by adding n-k parity check bits. The encoding operation can be expressed as a matrix multiplication by a generating matrix G. Similarly, the decoding starts with the syndrome computation that is obtained by multiplying the stored word by the parity check matrix H. In the absence of errors, the syndrome is an all zero vector. When a single error occurs, the syndrome is equal to the column of the H matrix that corresponds to the erroneous bits. Therefore, when all columns in H are different single errors can be located and corrected. When a double error occurs, the syndrome is equal to the modulo two addition of the columns of the affected bits. To achieve double error detection, a common technique is to use only odd-weight columns in the parity check matrix [2]. When that is done a double error is detected when the syndrome has an even weight. Odd weight syndromes are assumed to correspond to single errors. To implement Double Adjacent Error Correction (DAEC), the sum of any two adjacent columns has to be different from the sum of any other two adjacent columns and any single column [6].

The generating and parity check matrixes of SEC-DED-DAEC codes can be obtained using different algorithms that search for matrixes that meet the required conditions. Those are:

1) All columns in H are different and non-zero.
2) All columns in H have an odd-weight with data columns having a weight larger than one.
3) The sums of two adjacent columns in H are all different and also different from all columns and non zero.

Additional conditions may be added to minimize the number of ones in the matrixes in order to reduce encoding and decoding complexity or to reduce the probability of miscorrection when a triple error occurs [15].

The decoding of both SEC-DED and SEC-DED-DAEC codes is performed in three steps: syndrome computation, error location and error correction [16]. The syndrome computation is basically to re-compute the parity check equations. The error location phase is for single errors to compare the syndrome with each of the columns in the H matrix. For double adjacent errors, the comparison is done with the sum of the adjacent columns. Finally error correction can be done with a XOR gate.

To illustrate the decoder, let us consider a SEC-DED-DEAC code with $k = 16$ and $n = 22$ presented in [6] that has the following parity check matrix:

$$\begin{matrix} 1100110001110100100000 \\ 0001010011011001010000 \\ 1010100111100011001000 \\ 1001001110010110000100 \\ 0110101100001101000010 \\ 0111011000101010000001. \end{matrix} \quad (1)$$

This matrix meets all the requirements and the code is therefore a SEC-DED-DAEC code. The structure of the decoder is shown in Fig. 1. The first phase of decoding is a set of XOR gates that computes the syndrome bits $(\mathrm{s_i})$. Then a set of AND gates

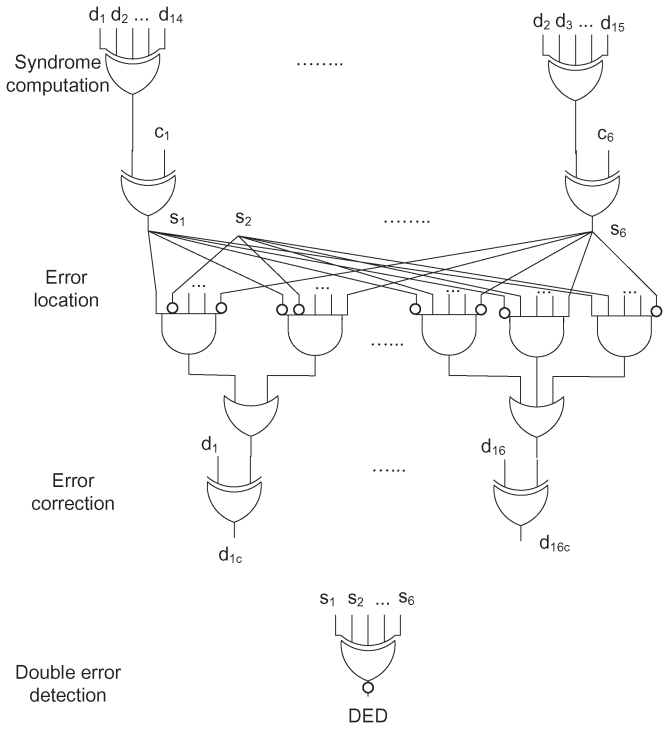

Fig. 1. Structure of the decoder for the SEC-DED-DAEC code.

is used to locate errors by comparing the syndrome with the columns in H and finally correction can be implemented with an XOR gate. Since for this code $\mathrm{n} - \mathrm{k} = 6$ the AND gates that locate error patterns will have six inputs of which only a few are shown in the Figure. For each bit there will be one single error pattern and one or two double adjacent patterns. In the Figure, the first data bit has only one adjacent pattern (columns 1 and 2) while the bit 16 has two adjacent patterns (columns 15 and 16 and columns 16 and 17). Those patterns are combined with an OR gate whose output is the input to the correction gate. Finally, the logic to implement double error detection is also shown in the figure.

The decoding of SEC-DED codes is similar but since there are less error patterns to correct, the error location logic is simpler.

III. Proposed SEC-DAEC Codes

As mentioned before, traditional SEC-DED codes have odd-weight in the data columns of their parity check (H) matrixes [2]. Therefore single errors produce a syndrome with an odd number of ones and double errors a syndrome with an even number of ones. This feature is used in SEC-DED codes to implement the DED feature. In the following, it will be used to optimize the implementation of DAEC.

Recently, a method to optimize the decoding of SEC and SEC-DED codes with constant weight has been presented [13]. This scheme is based on the observation that when all the data columns in the H matrix have the same weight, errors can be located by checking only that the ones in the syndrome match those in the data column. The modification reduces significantly the cost of locating errors and therefore lowers the decoding complexity.

A similar decoding optimization can be applied to a SEC-DAEC code when the code is designed to meet the following conditions (in addition to those required for a code to be SEC-DED-DAEC):

1) All the data columns of the parity check (H) matrix have a constant odd weight w_o larger than one.
2) The sum (modulo two) of any two adjacent columns involving a data column has a constant even weight w_e larger than w_o.

In that case decoding can be done as follows:

1) Compute the syndrome.
2) If the syndrome has an odd number of errors, correct the data column that matches the ones in the syndrome (if any).
3) Correct the adjacent columns whose sum (modulo two) matches the ones in the syndrome (if any).

To illustrate the scheme, the code with the H matrix given by (2) as follows will be used:

$$\begin{aligned}
&010000000000000001000000\\
&100101001010000000100000\\
&001010100001101000010000\\
&101100010101011000001000\\
&000010111011010100000100\\
&111001101100110100000010\\
&010111010110101100000001. \qquad (2)
\end{aligned}$$

This code meets the conditions required to implement the optimized decoding. It can be observed that the weight of all the data columns (w_o) is three and the sum of any two adjacent data columns (w_e) is four. In addition it must be noted that the code has seven parity check bits, one more than the matrix in (1).

The proposed decoder is shown in Fig. 2. The structure is the same as for the traditional SEC-DED-DAEC code in Fig. 1. For the single error patterns, the "not DED" signal is used to ensure that the number of ones in the syndrome is odd. The main difference is that the error location logic is now implemented with four input AND gates instead of six input gates. The inverters at some inputs of those gates are also no longer needed. This results in a simplification of the decoding logic as will be seen in the evaluation presented in the next section. In a general case, the complexity required to locate the errors for a block with k data bits and n-k check bits is: n-k-1 XOR (2 input) gates, k AND ($w_o + 1$ input) gates and k AND (w_e input) gates. This compares with a traditional SEC-DED-DAEC code that requires 2k AND (n-k input) gates and k inverters.

The decoder in Fig. 2 reduces the circuit area compared to a traditional SEC-DED-DAEC decoder but is slower. This can be explained as the "not DED" signal introduces a long path in the error identification and correction logic.

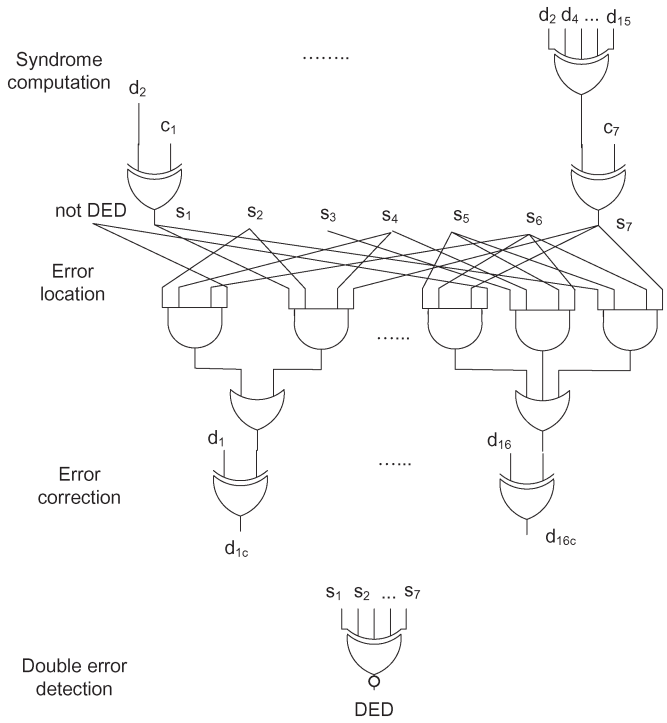

Fig. 2. Structure of the first proposed (area optimized) decoder for the SEC-DED-DAEC code.

An alternative implementation of the decoder optimized for speed can be done as follows:

1) Compute the syndrome.
2) Correct the data column that matches the ones and zeroes in the syndrome (if any).
3) Correct the adjacent columns whose sum (modulo two) matches the ones in the syndrome (if any).

In this second implementation, the single error patterns are identified with both zeroes and ones. This requires k AND (n-k input) gates. However, the double adjacent errors are identified with k AND (w_e input) gates and therefore the decoding is still simplified. At the same time the DED signal is not used avoiding the impact on delay. In the rest of the paper, the first implementation will be referred to as proposed scheme one and the second one as proposed scheme two.

In both cases, codes that meet the conditions required to apply the proposed decoding methods are needed. To that end, an automated procedure has been used to generate the codes. A program has been implemented in JAVA to construct the matrixes adding columns one at a time and checking the conditions at each step.

For a data word length of 32 bits, the derived codes have the same number of parity check bits as SEC-DED codes while for 16 and 64 bits one additional bit is needed. In a general case, the number of parity check bits has to be such that there are at least k combinations of the bits when taken in groups of w_o and w_e. For example, for $n - k = 6$ there are only 15 combinations of weight four and therefore it is not possible to find a code for $k = 16$. This is a necessary condition for the code to exist but it does not guarantee that a code can be built. The automated procedure can be then used to find a code and when it fails to

TABLE I
PARAMETERS OF THE PROPOSED SEC-DED-DAEC CODES

k	*n-k*
16	7
32	7
64	9

do so, the number of parity check bits can be increased and the process can be repeated again.

The parity check matrixes for the derived codes are shown in (2) for $\mathrm{k} = 16$, in the following for $\mathrm{k} = 32$:

$$\begin{array}{l} 00001100000001011101010101010100000001 \\ 10110001011100110000010001010110100000 \\ 01101010110111010011100100000100100000 \\ 11000101001010000100111110100010000010 \\ 01010111100100001001001010001100001000 1 \\ 10011010010010100110001000111000000 1001 \\ 00100000101001101010100011101000000100 \end{array} \quad (3)$$

and in (4), shown at the bottom of the page, for $\mathrm{k} = 64$. For 16 and 64 bits one extra parity bit is needed and matrixes are found in systematic form. For $\mathrm{k} = 32$ and $\mathrm{n} - \mathrm{k} = 7$ there are 35 combinations of weight three and four. Since only 32 of each are needed, a code can be found. However, the automated procedure was only capable of constructing the first 31 data columns of the matrix. To complete the matrix, the remaining data bit was manually placed after the check bits to meet the conditions. The parity check bits had also to be rearranged to that end. It is important to note that the code is still systematic and that (3) only shows how the bits would be placed in the memory. When reading or writing the words, the data can be rearranged so that encoding and decoding are the same as for the other codes.

The parameters of the codes are summarized in Table I. As mentioned before, the number of parity check bits is the same as for SEC-DED codes for $\mathrm{k} = 32$ while for $\mathrm{k} = 16$ and 64 an additional bit is needed.

TABLE II
AREA ESTIMATES (IN $\mu\mathrm{m}^2$) FOR SYNTHESIS WITH MAXIMUM EFFORT IN AREA

k	SEC-DED-DAEC in [6]	Proposed scheme one	Proposed scheme two
16	721	529	584
32	1,512	1,222	1,443
64	2,423	2,498	3,045

TABLE III
DELAY ESTIMATES (IN NANOSECONDS) FOR SYNTHESIS WITH MAXIMUM EFFORT IN AREA

k	SEC-DED-DAEC in [6]	Proposed scheme one	Proposed scheme two
16	0.89	1.12	0.88
32	1.30	1.57	1.12
64	1.48	1.83	1.43

TABLE IV
POWER ESTIMATES (IN mW) FOR SYNTHESIS WITH MAXIMUM EFFORT IN AREA

k	SEC-DED-DAEC in [6]	Proposed scheme one	Proposed scheme two
16	0.65	0.56	0.57
32	1.63	1.61	1.76
64	2.34	3.78	4.07

IV. EVALUATION

The proposed decoders have been implemented in HDL and synthesized for a 45 nm library [17]. The SEC-DED-DAEC codes presented in [6] have also been implemented to assess the benefits of the proposed decoding implementations. The decoders have been synthesized using Synopsis Design Compiler. Two settings were used for synthesis, maximum effort to optimize area and maximum effort to optimize delay. The first configuration is useful when circuit area is the priority while the second is relevant for high speed memories in which delay is critical. The area, delay and power estimates are summarized in Tables II–IV for area optimized synthesis and in Tables V–VII for delay optimized synthesis.

In the case of maximum effort to optimize area, it can be observed that the proposed scheme one provides significant circuit area savings for $\mathrm{k} = 16$ (26.6%) and 32 (19.2%) while for

$$\begin{array}{l} 0111010101010101000000000100000000000000000000000000000000000000100000000 \\ 1100101010101010111011010011010101000100000000000000000000000000010000000 \\ 0010000000100000101110101000101010101010110101010100000000000000001000000 \\ 1000111000001000000100001010100000110001101010101001010010100000000100000 \\ 0001001110000010010100000000011010010011011010000010101000011010000010000 \\ 1000010011110000000001110000001100011000100001101011000101010110000001000 \\ 0001100001000111100000100001000111100000001100110000101110110101000000100 \\ 0010000000011100001001000101110000001101010000011110011011001101000000010 \\ 0100000100000001010010011110000001000110000111000101110101101011000000001 \end{array} \quad (4)$$

TABLE V
AREA ESTIMATES (IN μm^2) FOR SYNTHESIS WITH MAXIMUM EFFORT IN DELAY

k	SEC-DED-DAEC in [6]	Proposed scheme one	Proposed scheme two
16	1,023	822	862
32	2,220	1,774	2.112
64	3,595	3,254	4,129

TABLE VI
DELAY ESTIMATES (IN NANOSECONDS) FOR SYNTHESIS WITH MAXIMUM EFFORT IN DELAY

k	SEC-DED-DAEC in [6]	Proposed scheme one	Proposed scheme two
16	0.44	0.56	0.43
32	0.53	0.67	0.51
64	0.70	0.73	0.59

TABLE VII
POWER ESTIMATES (IN MW) FOR SYNTHESIS WITH MAXIMUM EFFORT IN DELAY

k	SEC-DED-DAEC in [6]	Proposed scheme one	Proposed scheme two
16	3.41	2.95	2.86
32	8.36	8.07	9.05
64	12.88	16.80	18.64

$k = 64$, the area is roughly the same. These benefits in circuit area come at the expense of a significant impact on delay as can be seen in Table III. The proposed scheme two has worse area but lower delay as expected. In particular, it also reduces the area compared to the traditional decoders for $k = 16$ (19.0%) and 32 (4.6%) while for $k = 64$, the area is significantly worse. The results for power consumption show similar values for all implementations for $k = 16$ and 32 while the traditional SEC-DED-DAEC decoder outperforms the proposed schemes for $k = 64$. In summary, when the priority is to optimize circuit area, the first proposed technique is useful for $k = 16$ and $k = 32$ and the second proposed technique can also be used to reduce both area and speed.

In the case of maximum effort to optimize speed, the second proposed technique provides significant benefits for $k = 64$ (15.7%) while the gains are smaller for $k = 16$ and $k = 32$. This comes at the cost of a significant impact in both circuit area and power for $k = 64$. An interesting case is $k = 16$ in which the second proposed implementation outperforms the traditional decoder in area, delay and power. For the first proposed technique, delay is worse than traditional decoders for all word-lengths considered. In summary, the second proposed scheme can be used when the priority is to optimize circuit delay and in some cases it will also reduce area and power.

Finally, an important parameter of SEC-DED-DAEC codes is the probability of miscorrection when a double non adjacent error occurs. This has been evaluated by generating all possible non-adjacent double error patterns and checking the percentage of them that produce a syndrome value that is the same as that of a double adjacent error. The results are reported in Table VIII that also includes the percentages for the SEC-DED-DAEC codes presented in [6]. It can be observed that the probabilities are lower for 16 and 64 bits data words but slightly higher for 32 bits data words. This can be explained as in the first case (16 and 64) the proposed codes require one more extra bit than the codes in [6]. This means that there are more syndrome values that are not used for adjacent error correction and therefore the probability is lower.

TABLE VIII
PERCENTAGE OF MISCORRECTIONS FOR DOUBLE NON-ADJACENT ERRORS

k	SEC-DED-DAEC in [6]	Proposed Codes
16	56.2	45.5
32	53.9	55.2
64	52.9	35.5

V. CONCLUSION

In this paper, a method to optimize the decoding of SEC-DED-DAEC codes has been presented and evaluated. The results show that significant reductions in decoder area and delay can be achieved for data word-lengths commonly used in memories. The proposed scheme places some constraints on the parity check matrix in order to simplify the decoding. In particular, the weight of the columns and of the sum of adjacent data columns is forced to be constant. This can then be used to reduce the error location logic. In particular, two optimized decoder implementations, one for area and one for delay are proposed. The constraints in the parity check matrix can be met without increasing the number of parity check bits for a word-length of 32 bits. For 16 and 64 bits, an additional parity check bit is required. Therefore, the proposed method can be applied directly to 32 bit data memories. For other word lengths, the proposed scheme requires additional memory and is less attractive. The two proposed decoder implementations have also been evaluated and compared to existing SEC-DED-DAEC decoders. The results show that the first can be used to reduce the circuit area and the second the delay.

REFERENCES

[1] E. Ibe, H. Taniguchi, Y. Yahagi, K. Shimbo, and T. Toba, "Impact of scaling on neutron-induced soft error in SRAMs from a 250 nm to a 22 nm design rule," *IEEE Trans. Electron Devices*, vol. 57, no. 7, pp. 1527–1538, Jul. 2010.

[2] M. Y. Hsiao, "A class of optimal minimum odd-weight column SEC-DED codes," *IBM J. Res. Develop.*, vol. 14, no. 4, pp. 395–401, Jul. 1970.

[3] P. Reviriego, J. A. Maestro, S. Baeg, S. Wen, and R. Wong, "Protection of memories suffering MCUs through the selection of the optimal interleaving distance," *IEEE Trans. Nucl. Sci.*, vol. 57, pt. 1, no. 4, pp. 2124–2128, Aug. 2010.

[4] S. Satoh, Y. Tosaka, and S. A. Wender, "Geometric effect of multiple-bit soft errors induced by cosmic ray neutrons on DRAM's," *IEEE Electron Device Lett.*, vol. 21, no. 6, pp. 310–312, Jun. 2000.

[5] A. Neale and M. Sachdev, "A new SEC-DED error correction code subclass for adjacent MBU tolerance in embedded memory," *IEEE Trans. Device Mater. Rel.*, vol. 13, no. 1, pp. 223–230, Mar. 2013.

[6] A. Dutta and N. A. Touba, "Multiple bit upset tolerant memory using a selective cycle avoidance based SEC-DED-DAEC code," in *Proc. 25th IEEE VLSI Test Symp.*, 2007, pp. 349–354.

[7] S. Baeg, S. Wen, and R. Wong, "Minimizing soft errors in TCAM devices: A probabilistic approach to determining scrubbing intervals," *EEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 4, pp. 814, 822, Apr. 2010.

[8] R. Naseer and J. Draper, "Parallel double error correcting code design to mitigate multi-bit upsets in SRAMs," in *Proc 34th Eur. Solid State Circuits Conf.*, Sep. 2008, pp. 22–225.

[9] Z. Ming, X. L. Yi, and L. H. Wei, "New SEC-DED-DAEC codes for multiple bit upsets mitigation in memory," in *Proc. IEEE/IFIP 20th Int. Conf. VLSI Syst. Chip*, 2011, pp. 254–259.
[10] A. Dutta, "Low cost adjacent double error correcting code with complete elimination of miscorrection within a dispersion window for multiple bit upset tolerant memory," in *Proc. IEEE/IFIP 20th Int. Conf. VLSI Syst. Chip*, 2012, pp. 287–290.
[11] X. She, N. Li, and D. W. Jensen, "SEU tolerant memory using error correction code," *IEEE Trans. Nucl. Sci.*, vol. 59, no. 1, pp. 205–210, Feb. 2012.
[12] P. Reviriego, S. Pontarelli, J. A. Maestro, and M. Ottavi, "Low-cost single error correction multiple adjacent error correction codes," *Electron. Lett.*, vol. 48, no. 23, pp. 1470–1472, Nov. 2012.
[13] P. Reviriego, S. Pontarelli, J. A. Maestro, and M. Ottavi, "A method to construct low delay Single Error Correction (SEC) codes for protecting data bits only," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 32, no. 3, pp. 479–483, Mar. 2013.
[14] S. Lin and D. J. Costello, *Error Control Coding,* 2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 2004.
[15] M. Richter, K. Oberlaender, and M. Goessel, "New linear SEC-DED codes with reduced triple bit error miscorrection probability," in *Proc. 14th IEEE IOLTS*, Jul. 2008, pp. 37–42.
[16] C. L. Chen and M. Y. Hsiao, "Error-correcting codes for semiconductor memory applications: A state-of-the-art review," *IBM J. Res. Develop.*, vol. 28, no. 2, pp. 124–134, Mar. 1984.
[17] J. E. Stine *et al.*, "FreePDK: An open-source variation-aware design kit," in *Proc. IEEE Int. Conf. MSE*, Jun. 2007, pp. 173–174.

Pedro Reviriego (A'03–M'04) received the M.Sc. and Ph.D. (Hons.) degrees in telecommunications engineering from the Technical University of Madrid, Madrid, Spain, in 1994 and 1997, respectively. From 1997 to 2000, he was an R&D Engineer with Teldat, Madrid, working on router implementation. In 2000, he joined Massana to work on the development of Ethernet transceivers. During 2003, he was a Visiting Professor at the Universidad Carlos III de Madrid, Leganés, Spain. From 2004 to 2007, he was a Distinguished Member of the technical staff with LSI Corporation, working on the development of Ethernet transceivers. He is currently with the Universidad Antonio de Nebrija, Madrid. He is the author of numerous papers in international conference proceedings and journals. He has also participated in IEEE 802.3 standardization activities. His research interests include fault-tolerant systems, communication networks, and the design of physical-layer communication devices.

Jorge Martínez received the B.Sc. degree in computer science from Tecnológico de Monterrey, Monterrey, México, and the M.Sc. degree from Universidad Complutense de Madrid, Madrid, Spain, in 2013. He is currently working toward the Ph.D. degree at Universidad Antonio de Nebrija, Madrid. From 1991 to 2000, he was a Software Engineer and a Project Manager in consultancy and engineering companies. Since 2004, he has been a Part-Time Lecturer at Universidad Antonio de Nebrija. His current research interests include software fault tolerance and error corrections codes.

Salvatore Pontarelli received the Master's degree in electronic engineering from the University of Bologna, Bologna, Italy, in 2000 and the Ph.D. degree in microelectronics and telecommunications from the University of Rome "Tor Vergata," Rome, Italy, in 2003. He was with the National Research Council (CNR) and the Italian Space Agency (ASI) and has been a Consultant for various Italian and European companies for projects related to digital design and fault tolerance in digital systems. He is currently a Research Consultant with the National Inter-University Consortium for Telecommunications (CNIT), Parma, Italy. His research interests include the development of highly reliable systems, error detection and correction codes, fault detection and recovery for arithmetic circuits, and hardware for networking applications.

Juan Antonio Maestro (M'07) received the M.Sc. degree in physics and the Ph.D. degree in computer science from the Universidad Complutense de Madrid, Madrid, Spain, in 1994 and 1999, respectively. He has served both as a Lecturer and a Researcher at several universities, such as the Universidad Complutense de Madrid; the Universidad Nacional de Educación a Distancia (Open University), Madrid; Saint Louis University, Madrid; and the Universidad Antonio de Nebrija, Madrid, where he currently manages the Computer Architecture and Technology Group. His current activities are oriented to the space field, with several projects on reliability and radiation protection, as well as collaborations with the European Space Agency. Aside from this, he has worked for several multinational companies, managing projects as a Project Management Professional and organizing support departments. He is the author of numerous technical publications, both in journals and international conferences. His research interests include high-level synthesis and cosynthesis, signal processing, real-time systems, fault tolerance, and reliability.