Saiz-Adalid, L.; Reviriego, P.; Gil, P.; Pontarelli, S.; Maestro, JA. (2015). MCU Tolerance in SRAMs through Low Redundancy Triple Adjacent Error Correction. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 23(10):2332-2336. doi:10.1109/TVLSI.2014.2357476.

MCU Tolerance in SRAMs through Low Redundancy Triple Adjacent Error Correction

Luis-J. Saiz-Adalid, Pedro Reviriego, Pedro Gil, Salvatore Pontarelli and Juan-A. Maestro

***Abstract*—Static Random Access Memories (SRAMs) are key in electronic systems. They are used not only as standalone devices but also embedded in Application Specific Integrated Circuits. One key challenge for memories is their susceptibility to radiation induced soft errors that change the value of memory cells. Error Correction Codes (ECCs) are commonly used to ensure correct data despite soft errors effects in semiconductor memories. Single Error Correction-Double Error Detection (SEC-DED) codes have been traditionally the preferred choice for data protection in SRAMs. During the last decade, the percentage of errors that affect more than one memory cell has increased substantially, mainly due to Multiple Cell Upsets (MCUs) caused by radiation. The bits affected by these errors are physically close. To mitigate their effects, ECCs that correct single errors and double adjacent errors have been proposed. These codes, known as SEC-DAEC, require the same number of parity bits as traditional SEC-DED codes and a moderate increase in the decoder complexity. However, MCUs are not limited to double adjacent errors, because they affect more bits as technology scales. In this brief, new codes that can correct triple adjacent errors and 3-bit burst errors are presented. They have been implemented using a 45nm library and compared to previous proposals, showing that our codes have better error protection with a moderate overhead and low redundancy.**

***Index Terms*— Error correction codes, memory, SEC-DED, SEC-DAEC, SEC-DAEC-TAEC, burst error correction codes.**

I. Introduction

STATIC Random Access Memories (SRAMs) are an important component in most electronic systems. They are also commonly embedded into Application Specific Integrated Circuits (ASICs) and account for a significant portion of the circuit area. The importance of embedded memories is also expected to increase in the near future [1]. This widespread use makes the reliability of SRAM memories critical to ensure reliable electronic systems. One of the challenges for memories is radiation induced soft errors [2]. Soft errors change the value stored in one or more memory cells and can lead to data corruption and system failure. In older technology nodes, most soft errors affected only one memory cell. However, as technology scaled, Multiple Cell Upsets (MCUs) have become more common and are a significant percentage of the errors in current technology nodes [3].

Error Correction Codes (ECCs) are widely used to protect memories from soft errors [4]. Single Error Correction-Double Error Detection (SEC-DED) codes can be used when only single bit errors need to be corrected [5]. This was the case for old technologies but also for advanced technology nodes when sufficient interleaving distance is used. Interleaving ensures that an MCU affects cells that belong to different words by placing the memory cells of the same word physically apart [6]. Its effectiveness is based on the observation that the cells affected by a soft error event are physically close and located in the area affected by the impact of the radiation particle [7]. The use of interleaving has implications for the memory design and can impact area and delay [8]. The routing is more complex, and power consumption is increased. Interleaving cannot always be used in small memories nor can be practically applied to content addressable memories. It also impacts the aspect ratio, which can be an issue for embedded memories [9],[10].

When interleaving is not a good option, ECCs that correct multiple nearby bit errors can be used. The simplest codes (SEC-DAEC) correct double adjacent errors in addition to single bit errors. They can be implemented with the same number of parity check bits as SEC-DED codes and with a moderate increase in the decoder complexity [8]. Several SEC-DAEC codes have been proposed recently to protect memories [8],[9],[11],[12]. One issue with most SEC-DAEC codes is that double non-adjacent errors can be mistaken for a correctable error causing an erroneous correction. Therefore, they cannot guarantee the detection of double random errors and are not DED codes as SEC-DED codes. However, the codes can be extended to ensure that this does not occur in general or for error patterns that affect nearby bits but are not double adjacent. This has been explored in [9],[12],[13] and in some cases requires additional parity check bits.

Measurements from 65nm and 45nm SRAMs are reported in [10]. As can be observed in Fig. 1, although most MCUs have a length of 2, there is a non-negligible percentage of faults affecting 3 adjacent bits, and it is expected to grow as technology scales. Similar results are presented in [14]. This means that SEC-DAEC codes may no longer be effective to correct soft errors. The next step is to design codes that can correct multiple adjacent errors [15],[16],[17].

Manuscript received 31-March-2014. This work was partially supported by the Universitat Politècnica de València under the research project DesTT (SP20120806), and by the Spanish Ministry of Science and Education under projects AYA2009-13300-C03 and ARENES (TIN2012-38308-C02-01). This brief is part of a collaboration in the framework of COST ICT Action 1103 "Manufacturable and Dependable Multicore Architectures at Nanoscale".

Luis-J. Saiz-Adalid and Pedro Gil are with Instituto ITACA, Universitat Politècnica de València, Camino de Vera, s/n, 46022, Valencia, Spain (phone: +34-963877000; fax: +34-963877579; email: {ljsaiz, pgil}@itaca.upv.es).

P. Reviriego and J.A. Maestro are with Universidad Antonio de Nebrija, C/ Pirineos, 55 E-28040 Madrid, Spain (phone: +34-914521100; fax: +34-914521110; email: {previrie, jmaestro}@nebrija.es).

Salvatore Pontarelli is with Consorzio Nazionale Interuniversitario per le Telecomunicazioni, CNIT, Via del Politecnico 1 - 00133 Rome, Italy (phone +39-0672597369; email: pontarelli@ing.uniroma2.it).

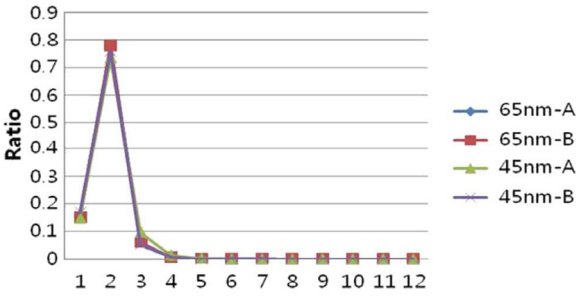

Fig. 1. Ratio of length for MCU events [10].

As stated above, double and triple adjacent errors are the most common patterns for multiple errors. However, 3-bit burst errors are also possible. A burst error is a group of adjacent bits where, at least, the first and the last bits are in error. So, double almost adjacent errors (2 bits in error separated by a right bit) must be studied. For a better understanding, Fig. 2 shows how a high energy particle strike of the same size (the balls) may provoke single or multiple cell upsets depending on the size of the memory cells (the prisms). It is noticeable that, although the fault affects several bits (3 in the case presented in Fig. 2 (b)), not all of them become in error [16]. Consider that a high energy particle forces a high logic level in the affected memory cells, for example. The stored bits only become in error if the value previously stored was a low logic level. In other words, an MCU affecting 3 bits could manifest as a triple adjacent error, but also as a double almost adjacent error, a double adjacent error, a single error or even it could not cause error.

(a) Single Cell Upset (b) Multiple Cell Upset

Fig. 2. Single and Multiple Cell Upsets in memory elements.

Thus, SEC-DAEC-TAEC codes and 3-bit burst ECCs would be useful. These codes have been explored in [17] for data words of 16 bits. In that case, an additional parity check bit was required compared to a SEC-DED code. The decoding complexity is also larger than that for SEC-DED or SEC-DAEC codes as more syndromes have to be checked for error detection. More powerful codes are presented in [16], combining double error and burst error correction for data words of up to 24 bits, but at the cost of a high redundancy.

In this brief, the design of SEC-DAEC-TAEC and 3-bit burst ECCs for different data word lengths is considered. In particular, codes for 16, 32 and 64 data bits are presented. Two optimization criteria have been used to select the codes: minimizing the total number of ones in the parity check matrix or the maximum number of ones in its rows. The first criterion is commonly used to minimize the decoder complexity while the second is used to optimize its speed [8]. The parity check matrices for all the proposed codes are provided. These results can be useful to designers that need to deal with multiple bit errors in SRAM memories.

In addition, the encoders and decoders for these codes have been modelled in HDL, synthesized for a 45nm library [18], and compared to other existing proposals. The results show that our codes offer better coverage, with the same redundancy (for SEC-DAEC-TAEC codes) or only one bit more (for 3-bit burst error correction) than SEC-DED or SEC-DAEC codes. Furthermore, the results show that the proposed codes can be efficiently implemented and require only a moderate area and latency overhead versus existing SEC-DAEC codes.

The rest of the brief is organized as follows: in section II the procedure used to design the codes is briefly described. The SEC-DAEC-TAEC and 3-bit burst ECCs are presented in section III. Section IV performs a comparison with some of the previous SEC-DAEC codes. Finally, the conclusions are summarized in section V.

II. Code Design Technique

The process used to design the codes is based on formulating the problem as a Boolean Satisfiability problem. An algorithm developed by the authors is used to solve it and to obtain a parity check matrix which defines the code to be designed. Although a detailed explanation of the methodology (which can be found in [19]) is out of the scope of this brief, it is briefly summarized in the following. A syndrome based decoding [20] is assumed for the codes as has been done for the existing SEC-DAEC codes.

After determining the values of n and k for the code to be designed (where k is the length of the original data word and n is the length of the final encoded word), the first step is the selection of error patterns to be corrected. In this case, the SEC feature is represented with error vectors (…1…), error vectors for DAEC shows the pattern (…11…), for DAAEC (…101…), and for TAEC (…111…), where the dots represents zero or more 0's (correct bits) and the 1's are the bits in error.

The next step is to find the parity check matrix $\mathbf{H}$ that satisfies the conditions (1) and (2), where $\boldsymbol{E}_+$ represents the set of error vectors to be corrected, and $\boldsymbol{E}_\Delta$ is the set of error vectors to be detected.

$$\mathbf{H}\cdot\mathbf{e}_i^T \neq \mathbf{H}\cdot\mathbf{e}_j^T; \forall \mathbf{e}_i, \mathbf{e}_j \in \boldsymbol{E}_+ \mid \mathbf{e}_i \neq \mathbf{e}_j \quad (1)$$

$$\mathbf{H}\cdot\mathbf{e}_i^T \neq \mathbf{H}\cdot\mathbf{e}_j^T; \forall \mathbf{e}_i \in \boldsymbol{E}_\Delta, \mathbf{e}_j \in \boldsymbol{E}_+ \quad (2)$$

To find the matrix, a recursive backtracking algorithm is used. It checks partial matrices and adds a new column only if the previous matrix satisfies the requirements. In this way, the algorithm starts with a **partial_H** matrix, with $n–k$ rows and only one column. New columns are added and the new partial matrices are checked recursively. Both the initial and the added columns must be non-zero, so there are $2^{n-k}-1$ combinations for each column. A detailed explanation of this algorithm can be found in [19].

The complete execution of the algorithm is commonly unfeasible. Nevertheless, the first solutions are usually found quickly, if the code exists. The algorithm may choose better solutions according to different optimization criteria, but finding the best solution requires finishing the process. In this brief, two criteria have been used:

- Smallest Hamming weight of **H**: this solution commonly offers circuits with the lowest number of logic gates in a hardware implementation of the encoder and syndrome computation.
- Smallest Hamming weight of the heaviest row of **H**: the logic depth of each parity or syndrome bit generator depends on the Hamming weight of the associated row. The heaviest row is a factor for the maximum speed of the encoder and the decoder circuits.

Once selected the **H** matrix, it is easy to determine the logic equations to calculate each parity and syndrome bit, and the syndrome lookup table. They are required for the encoder and decoder implementation. An example is also included in [19].

III. Proposed Codes

As discussed in the introduction, the objective is to design SEC-DAEC-TAEC and 3-bit burst ECCs for data word sizes more commonly used in memories. As in [8], words of 16, 32 and 64 data bits are considered. So, these will be the values of k for each code. Since for each word size there are many possible codes and parity check matrices, the algorithm tries to optimize the code in two ways, as mentioned before. One of the optimizations tries to reduce the total number of ones in the parity check matrix as this reduces the decoder complexity [8]. The other optimization criterion tries to reduce the maximum number of ones in the rows of the parity check matrix. This in turn reduces the decoder delay [8].

For word length of n, a SEC-DAEC-TAEC code has to identify and correct $n+n-1+n-2 = 3n-3$ error patterns and a 3-bit burst ECC $n+n-1+n-2+n-2 = 4n-5$ error patterns. This means that the number of different syndromes has to be larger than $3n-3$ in the first case and larger than $4n-5$ in the second, which in turn means that the number of parity check bits $n-k$ should be such that $2^{n-k}>3n-3$ and $2^{n-k}>4n-5$, respectively. It is important to note that these conditions are necessary but not sufficient: the lowest value of n meeting the above conditions minimizes the code redundancy, but it does not guarantee the existence of a code with the required features. If it does not exist, the alternatives are to increase the number of parity bits (i.e. increasing n, as k is commonly a value fixed by design) or to reduce the number of error patterns to be corrected.

The described optimization algorithm has been used to design the codes by setting n to the lowest value that met the previous inequalities. As stated above, if no solution is found, n can be increased by one until valid codes are obtained. The optimization algorithm has been stopped after a practical execution time (about one week). Therefore, the codes presented may not be the optimum ones but any difference in terms of number of ones should be small. The results for 3-bit burst ECCs required one more parity check bit than a SEC-DED code in all cases. This is in line with previous results for words of 16 bits in [17]. However, in most cases, SEC-DAEC-TAEC codes can be implemented with the same number of parity bits needed for a SEC-DED code.

The parity check matrices **H** of the 3-bit burst ECCs obtained are given in Figs. 3 to 8. **H** matrices of the SEC-DAEC-TAEC codes obtained are given in Figs. 9 to 14.

A particular case is the (22, 16) SEC-DAEC-TAEC code. Using our methodology, no code with these features has been found. Although the number of syndromes available ($2^{22-16}-1 = 63$) is just enough to fit the errors to be corrected ($3 \cdot 22-3 = 63$), no solution was found after long executions of the algorithm using different configurations. This can be due to the fact that no spare syndromes are available and therefore the error patterns to be corrected have to match perfectly to the syndromes. As stated above, although the values of n and k meet the inequality $2^{n-k}>3n-3$, this is not a sufficient condition, at least from our search results. If an increase in the redundancy (using higher values of n) is not desired, the solution to find a code is to reduce the number of error patterns to be corrected. In this case, we decided to find a SEC-DAEC code with a percentage of correction of triple adjacent errors as high as possible, while the rest are detected. This code offers a compromise solution, and shows the flexibility of the methodology used to design the codes. In this case, the code found and presented here corrects 18 out of 20 triple adjacent errors, i.e. 90%. The other 2 errors are detected but not corrected.

The Hamming distance is 3 for all the codes presented. Since all the codes are SEC, the minimum distance is at least 3. Also, as all codes have columns with weight 2, an error on one of those columns and on the two parity bits will produce a valid codeword. Therefore, there are words at distance 3. An interesting treatment for Hamming distance when burst errors are considered can be found in [16].

$$\mathbf{H} = \begin{bmatrix} 10000001001101001000011 \\ 01000000100110111001000 \\ 00100000001001000011001 0 \\ 00010001000100010010100 \\ 00001000100010010001001 \\ 00000100010001001011010 \\ 00000010001000100100100 \end{bmatrix}$$

Fig. 3. Parity check matrix for the (23,16) 3-bit burst ECC optimized to reduce the total number of ones.

$$\mathbf{H} = \begin{bmatrix} 1000000010\ 0101100100\ 0110001110\ 0110000110 \\ 0100000001\ 0010010010\ 0010010011\ 0001110100 \\ 0010000000\ 1001000101\ 0000100100\ 1100011000 \\ 0001000010\ 0001001000\ 0001011001\ 0011001100 \\ 0000100001\ 0000000100\ 1100010010\ 1010010001 \\ 0000010000\ 1000100010\ 0010001001\ 0100100011 \\ 0000001000\ 0100010001\ 0001000100\ 1000100010 \\ 0000000100\ 0010001000\ 1000100000\ 0001001001 \end{bmatrix}$$

Fig. 4. Parity check matrix for the (40,32) 3-bit burst ECC optimized to reduce the total number of ones.

$$\mathbf{H} = \begin{bmatrix} 1000000001001001001011001110010000100011110000110100100100100000011000010 \\ 0100000000100100100100100100000001100110011100001000111001100010000100100 \\ 0010000000010010000010000011001001001100100110010011000010000111001001000 \\ 0001000000100000010010000100110010000000100100100100101000011001100000011 \\ 0000100000100001000000010010010001011010001010100100001001001000100011001 \\ 0000010000010000100001000000000100101001000101000110010101000100010010100 \\ 00000010000001000010000100000001000000100100001100001001010110100100101010 \\ 0000000100000010000100001000100001000000000100100110100000010010010010001 \\ 0000000010000001000010000100010010001000000000000000100100001001001100100 \end{bmatrix}$$

Fig. 5. Parity check matrix for the (73,64) 3-bit burst ECC optimized to reduce the total number of ones.

$$\mathbf{H} = \begin{bmatrix} 10000001001110010000001 \\ 01000000100100001110010 \\ 00100000001001110001010 0 \\ 00010001000100101001100 \\ 00001000100010010011100 \\ 00000100010000101100101 \\ 00000010001001001001011 \end{bmatrix}$$

Fig. 6. Parity check matrix for the (23,16) 3-bit burst ECC optimized to reduce the maximum number of ones in a row.

$$
\mathbf{H}=\begin{bmatrix}
1000000010\;0101100100\;0110001110\;0100000000\\
0100000001\;0010010010\;0010010011\;0001010010\\
0010000000\;1001000101\;0000100100\;1100110100\\
0001000010\;0001001000\;0001011001\;0011001001\\
0000100001\;0000000100\;1100010010\;1000011100\\
0000010000\;1000100010\;0010001001\;0110000101\\
0000001000\;0100010001\;0001000100\;1010100110\\
0000000100\;0010001000\;1000100000\;0001101011
\end{bmatrix}
$$

Fig. 7. Parity check matrix for the (40,32) 3-bit burst ECC optimized to reduce the maximum number of ones in a row.

$$
\mathbf{H}=\begin{bmatrix}
1000000001001001001011001110010000100011110000110100100100100000010000000\\
0100000000100100100100100100000001100110011100001000111001100010001000100\\
0010000000010010000010000011001001001100100110010011000010000111000100000\\
00010000010000000100100001001100100000001001001001001010000011001100101110\\
00001000001000010000000010010010001011010001010100100001001001000100110110\\
0000010000010000100001000000000010010100100010100011001010100010001001100 1\\
0000001000001000010000100000000100000010010000110000100101011010010110010 1\\
0000000100000100001000010001000010000000010010011010000010010010010010010\\
0000000010000010000100001000100100010000000000000000000100100001001001001001
\end{bmatrix}
$$

Fig. 8. Parity check matrix for the (73,64) 3-bit burst ECC optimized to reduce the maximum number of ones in a row.

$$
\mathbf{H}=\begin{bmatrix}
1000001010010011100010\\
0100000101001110101000\\
0010001000110111110110\\
0001000100010001000101\\
0000100101000100111010\\
0000010010111010101101
\end{bmatrix}
$$

Fig. 9. Parity check matrix for the (22,16) SEC-DAEC-quasiTAEC (90% correction-10% detection) code optimized to reduce the total number of ones.

$$
\mathbf{H}=\begin{bmatrix}
100000010101010000101100110110010010010\\
010000001010001100100010100100001100101\\
001000010000100010000101010100100001101\\
000100001001000000111011001110100101001 0\\
000010000100100100000001010111010010100 1\\
000001000010010010010000010011111010100\\
000000100000000100100100100100101010101 0
\end{bmatrix}
$$

Fig. 10. Parity check matrix for the (39,32) SEC-DAEC-TAEC code optimized to reduce the total number of ones.

$$
\mathbf{H}=\begin{bmatrix}
1000000010\;1010100001\;0100010101\;0010001100\;0001001000\;1101010010\;1000001001\;10\\
0100000001\;0100011000\;0010010000\;0111011011\;1110010101\;0000010001\;0010100101\;00\\
0010000010\;0001000100\;0001000111\;0100110010\;0001010010\;0100110110\;1101011001\;01\\
0001000001\;0010000010\;1000110100\;0001000101\;0100101001\;0010001110\;0010001000\;01\\
0000100000\;1001001000\;0000001001\;0101100000\;1010100010\;0010010100\;1010010111\;00\\
0000010000\;0100100100\;1000000000\;1000101010\;1001001101\;1001001001\;0101011010\;00\\
0000001000\;0000010010\;0101000000\;0010010100\;0100100100\;0101100100\;0100100100\;10\\
0000000100\;0000000001\;0010101010\;1000000001\;0010010010\;1010101011\;1011110011\;11
\end{bmatrix}
$$

Fig. 11. Parity check matrix for the (72,64) SEC-DAEC-TAEC code optimized to reduce the total number of ones.

$$
\mathbf{H}=\begin{bmatrix}
1000001010100110011110\\
0100000100010010111010\\
0010001011101011001100\\
0001000100110110010111\\
0000100010101111001010\\
0000010001010011110101
\end{bmatrix}
$$

Fig. 12. Parity check matrix for the (22,16) SEC-DAEC-quasiTAEC (90% correction-10% detection) code optimized to reduce the maximum number of ones in a row.

$$
\mathbf{H}=\begin{bmatrix}
100000010101010000101101000010010010001\\
010000001010001100100010100101001111000\\
001000010000100010000100111110100100110\\
000100001001000000111011000100100101001 0\\
000010000100100100000001001101001010110 1\\
000001000010010010010001010101011001010\\
000000100000000100100100110101010101010 1
\end{bmatrix}
$$

Fig. 13. Parity check matrix for the (39,32) SEC-DAEC-TAEC code optimized to reduce the maximum number of ones in a row.

$$
\mathbf{H}=\begin{bmatrix}
1000000010\;1010100001\;0100010101\;0010001100\;0001001000\;1101010010\;1010010010\;00\\
0100000001\;0100011000\;0010010000\;0111011011\;1110010101\;0000010001\;0001001001\;00\\
0010000010\;0001000100\;0001000111\;0100110010\;0001010010\;0100110111\;0100010100\;10\\
0001000001\;0010000010\;1000110100\;0001000101\;0100101001\;0010001110\;0010011011\;00\\
0000100000\;1001001000\;0000001001\;0101100000\;1010100010\;0010010100\;0111111101\;01\\
0000010000\;0100100100\;1000000000\;1000101010\;1001001101\;1001001001\;0100101010\;11\\
0000001000\;0000010010\;0101000000\;0010010100\;0100100100\;0101100101\;1001111111\;10\\
0000000100\;0000000001\;0010101010\;1000000001\;0010010010\;1010101011\;1110101010\;01
\end{bmatrix}
$$

Fig. 14. Parity check matrix (**H**) for the (72,64) SEC-DAEC-TAEC code optimized to reduce the maximum number of ones in a row.

IV. EVALUATION AND COMPARISON

The proposed codes have been compared to the SEC-DAEC codes presented in [8], and to the 3-bit burst ECC presented in [17]. The encoders and decoders have been modelled in HDL and synthesized for a 45nm library [18]. They have been synthesized twice, optimizing area and delay, respectively.

The results for each configuration are summarized in Table I, where 3 groups of columns are distinguished. In the first part (redundancy and complexity), it can be observed that the new 3-bit burst ECCs require one more parity bit than SEC-DAEC codes, while the proposed SEC-DAEC-TAEC codes can be implemented with the same number of parity bits as SEC-DAEC codes. This is an interesting result as TAEC can be implemented without increasing the memory bitwidth.

In the same part, Hamming weights are displayed, and the optimized parameter for each code is emphasized. In terms of the total number of ones, the new codes provide reductions compared to the SEC-DAEC codes in [8] of up to 22%, showing the optimization ability of our methodology. The reduction in terms of the maximum number of ones in a row is also relevant. Finally, for $k = 16$, our proposals also reduce the number of ones in the matrix compared to the 3-bit burst ECC presented in [17].

As stated above, encoders and decoders have been synthesized twice. In the first run, the tool is configured to optimize area (second part of Table I), and in the second to optimize delay (third part). These settings provide respectively the best area and delay that can be achieved. In both cases area (in μm^2) and delay (in ns) are reported, but the most relevant columns (the optimized parameter for each case) are emphasized in Table I. It can be observed that the area and delay of the encoders for a given block size is similar for most of the codes considered. The same applies for the decoder delay. On the other hand, the area of the decoders is larger for the 3-bit burst codes as they have to identify and correct more patterns. However, in all cases, the proposed codes require less than two times the area of existing SEC-DAEC codes. The results also show that the proposed codes designed to reduce the total number of ones (Figs. 3 to 5 and 9 to 11) require less area for the encoder as expected. Conversely, the codes designed to reduce the maximum number of ones in a row (Figs. 6 to 8 and 12 to 14) have in most cases lower encoder/decoder delay. Therefore, they should be used to optimize area or delay respectively.

As a summary, the proposed codes improve previous SEC-DAEC designs by implementing either TAEC or 3-bit burst error correction, and also reduce the number of ones in the parity check matrix. The proposed codes can be efficiently implemented, and require only a moderate overhead versus existing SEC-DAEC codes.

TABLE I
COMPARISON OF THE PROPOSED CODES (EMPHASIZED) TO OTHER CODES. AREA EXPRESSED IN μm^2 AND DELAY IN ns.

		REDUNDANCY AND COMPLEXITY			*AREA OPTIMIZATION*				*DELAY OPTIMIZATION*			
					ENCODER		*DECODER*		*ENCODER*		*DECODER*	
k	***Code***	***n-k***	***Total number of ones in* H**	***Max. num. of ones in a row***	***area***	***delay***	***area***	***delay***	***area***	***delay***	***area***	***delay***
16	SEC-DAEC presented in [8]	6	54	10	156	0.47	1006	1.06	229	0.23	1439	0.47
	3-bit burst ECC presented in [17]	7	54	8	131	0.38	1404	1.35	172	0.19	1963	0.47
	3-bit burst ECC proposed in Fig. 3	7	**46**	8	**117**	0.35	**1298**	1.22	167	0.19	1824	0.47
	3-bit burst ECC proposed in Fig. 6	7	49	**7**	124	0.33	1356	1.20	179	**0.19**	1998	**0.46**
	SEC-DAEC-quasiTAEC proposed in Fig. 9	6	**51**	11	**91**	0.44	**1135**	1.23	119	0.21	1528	0.49
	SEC-DAEC-quasiTAEC proposed in Fig. 12	6	56	**10**	135	0.52	1128	1.27	196	**0.23**	1599	**0.49**
32	SEC-DAEC presented in [8]	7	103	15	322	0.53	1902	1.33	387	0.29	2751	0.53
	3-bit burst ECC proposed in Fig. 4	8	**90**	15	**270**	0.43	**2603**	1.35	320	0.26	3658	0.54
	3-bit burst ECC proposed in Fig. 7	8	91	**12**	264	0.45	2628	1.32	376	**0.24**	3599	**0.54**
	SEC-DAEC-TAEC proposed in Fig. 10	7	**92**	15	**279**	0.49	**2202**	1.34	328	0.26	3000	0.54
	SEC-DAEC-TAEC proposed in Fig. 13	7	93	**14**	288	0.55	2336	1.40	374	**0.25**	3190	**0.53**
64	SEC-DAEC presented in [8]	8	232	32	678	0.61	3106	1.75	812	0.33	4227	0.61
	3-bit burst ECC proposed in Fig. 5	9	**180**	25	**566**	0.58	**5279**	1.81	695	0.30	7165	0.62
	3-bit burst ECC proposed in Fig. 8	9	182	**23**	572	0.59	5158	2.01	696	**0.29**	6833	**0.61**
	SEC-DAEC-TAEC proposed in Fig. 11	8	**189**	27	**583**	0.65	**3672**	1.67	703	0.30	4928	0.62
	SEC-DAEC-TAEC proposed in Fig. 14	8	196	**25**	587	0.60	4563	1.87	722	**0.31**	5976	**0.62**

V. CONCLUSIONS

In this brief, new SEC-DAEC-TAEC and 3-bit burst error correction codes have been proposed to protect SRAM memories from Multiple Cell Upsets (MCUs). The codes presented cover the most common memory word lengths. The new codes complement the recently proposed 3-bit burst error correction code for 16-bit words.

The codes have been synthesized and compared to existing SEC-DAEC codes and the results show that the new codes provide significant reductions in the number of ones in the parity check matrix and in the maximum number of ones in its rows. Those reductions should translate into lower complexity for the encoder and syndrome computation. The proposed SEC-DAEC-TAEC codes can be implemented with the same number of parity check bits as SEC-DAEC codes while the 3-bit burst error correction codes require only one additional parity check bit. The synthesis results confirm that the proposed codes can be efficiently implemented, and introduce only a moderate overhead versus existing SEC-DAEC codes.

The parity check matrices for all the codes are provided in systematic form and can be readily used by designers of systems in which SRAM memories suffer MCUs that affect up to three neighboring bits.

REFERENCES

[1] "The international technology roadmap for semiconductors (ITRS)," [Online] Available: http://www.itrs.net.

[2] R. C. Baumann, "Soft errors in advanced computer systems," *IEEE Des. Test. Comput.,* vol. 22, no. 3, pp. 258–266, May/Jun. 2005.

[3] E. Ibe, H. Taniguchi, Y. Yahagi, K. Shimbo and T. Toba, "Impact of scaling on neutron-induced soft error rate in SRAMs From a 250 nm to a 22 nm Design Rule", *IEEE Trans. on Electron Devices*, vol. 57 , no. 7, pp. 1527-1538, July 2010.

[4] 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.

[5] M.Y. Hsiao "A class of optimal minimum odd-weight column SEC-DED codes", *IBM J. Res. Develop.,* vol. 14, no. 4, pp. 395–301, Jul. 1970.

[6] P. Reviriego, J.A. Maestro, S. Baeg, S. Wen, R. Wong, "Protection of Memories Suffering MCUs through the Selection of the Optimal Interleaving Distance", *IEEE Trans. on Nuclear Science*, vol. 57, no. 4(1) , pp. 2124-2128, Aug. 2010.

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

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

[9] A. Neale and M. Sachdev, "A New SEC-DED Error Correction Code Subclass for Adjacent MBU Tolerance in Embedded Memory", *IEEE Trans on Device and Materials Reliability*,vol. 13, no. 1, pp. 223 - 230, March 2013.

[10] S. Baeg, S. Wen and R. Wong, "SRAM Interleaving Distance Selection with a Soft Error Failure Model," *IEEE Trans. on Nuclear Science*, vol.56, no.4, pp. 2111-2118, Aug. 2009.

[11] Z. Ming, X. L. Yi and L. H. Wei "New SEC-DED-DAEC codes for multiple bit upsets mitigation in memory", in *Proc. of the IEEE/IFIP 20th International Conference on VLSI and System-on-Chip*, 2011, pp. 254 - 259.

[12] 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 *Procs. IEEE/IFIP 20th International Conference on VLSI and System-on-Chip*, 2012, pp. 287 - 290.

[13] P. Reviriego, S. Pontarelli, A. Evans, J.A. Maestro, "A Class of SEC-DED-DAEC codes derived from Orthogonal Latin Square Codes" to appear on IEEE Transaction on VLSI systems.

[14] R.K. Lawrence, and A.T. Kelly, "Single Event Effect Induced Multiple-Cell Upsets in a Commercial 90nm CMOS Digital Technology," *IEEE Trans. on Nuclear Science*, vol.55, no.6, pp. 3367-3374, Dec. 2008.

[15] P. Reviriego, S. Pontarelli, J.A. Maestro and M. Ottavi, "Low Cost Single Error Correction Multiple Adjacent Error Correction Codes", IET Electronics Letters Vol. 48 no. 23, pp. 1470-1472, Nov. 2012.

[16] S. Shamshiri, K.T. Cheng, "Error-Locality-Aware Linear Coding to Correct Multi-bit Upsets in SRAMs," IEEE International Test Conference, pp. 1-10, November 2010.

[17] X. She, N. Li and D.W. Jensen, "SEU Tolerant Memory Using Error Correction Code", *IEEE Trans. on Nuclear Science*, vol.59, no.1, pp. 205-210, Feb. 2012.

[18] J. E. Stine et al., "FreePDK: An open-source variation-aware design kit," *in Proc. IEEE Int. Conf. Microelectronic Systems Education, (MSE'07)*, pp. 173–174, Jun. 2007.

[19] Luis-J. Saiz-Adalid, Pedro-J. Gil-Vicente, Juan-Carlos Ruiz, Daniel Gil-Tomás, J.-Carlos Baraza, Joaquín Gracia-Morán, "Flexible Unequal Error Control Codes with Selectable Error Detection and Correction Levels", in Proc. of Intl. Conf. on Computer Safety, Reliability and Security (Safecomp), 2013, pp. 178-189.

[20] S. Lin and D. J. Costello, "Error control coding" (2nd Ed.). Englewood Cliffs, NJ: Prentice-Hall. 2004.