Extending 4-Bit Burst Error-Correction Codes with Convolutional Interleaving

Anna Maria P.I.[1] Sindhu T.V.[2]
[1]Student [2]Assistant Professor
[1,2]Department of Electronics and Communication Engineering
[1,2]IES College of Engineering, Thrissur, India

Abstract— The reliability of electronic systems lies in the fact that it can store and retrieve accurate data. Since the data storing elements, memories perform an essential role in space application. The space radiation cause error in the data saved in the memory. As technology advances and cell interval distance decrease, the memory cell area is getting reduced, and more cells get affected by this space radiation. Existing error-correcting codes can use for correcting up to 4-bit burst errors. I propose a technique that extends quadruple adjacent error correction codes with convolutional interleaving to improve reliability. The interleaving and deinterleaving procedure of the proposed method is illustrated with an example. The encoders, decoders, interleaver and deinterleaver implemented using VHDL in Xilinx ISE Design Suit 14.7. The simulation result shows that a burst error that affects more than four adjacent bits can be corrected using the proposed method.

Keywords: Convolutional Interleaving, Quadruple Adjacent Error Correction Codes, Deinterleaving, Interleaver

I. INTRODUCTION

Error correction codes (ECCs) with sterling correction ability are commonly used to protect memory from multiple bit upsets (MBUs). System on chip (SOC) and Application specific integrated circuits (ASIC) extensively use memories. In these applications, a broad section of the circuit area is allocated for memories. Due to this, memories tolerate more space radiation than other components. Hence, the sensitivity of memories to radiation has become a matter of concern to guarantee electronic systems' consistency. Soft errors induced due to radiation in the form of the single event upset (SEU) and multiple bit upset (MBU) are two eminent single event effects in current static random access memories (SRAMs). As semiconductor technology progresses from the submicrometer technology to the ultra-deep submicrometer (UDSM) technology, the radius affected by a particle includes more cells since the size of memory cells is smaller.

When high-energy radiation like protons from the sun or cosmic rays from beyond our galaxy hits the primary memory cell, it leaves a destructive trail of charged particles along the transport track. These generated electron-hole pairs give rise to soft errors by altering the bits stored in the memory cell leading to data falsification and system crash. For large feature sized transistors, one single memory cell may affect because of a radiation event. That is, only the SEU happens. Here, the use of single error-correction (SEC)-double error-detection (DED) codes is sufficient to save the memory from radiation consequences.

As the feature size comes to the DSM range, the critical charge becomes reducing. As shown in Fig. 1.2, the memory cell area scales down for each succeeding technology node, causing many memory cells disturbed by a particle hit. For the CMOS bulk technology, with cell interval distance decreasing, the electron-hole pairs generated in the substrate can diffuse to neighboring cells and produce MBUs. This compares with the FDSOI technology, which segregates transistors and restricts the multicollection mechanism. Hence, the multicollection mechanism is more eminent for bulk technology, and the MBU probability is higher. To acquire protection over MBUs, ECCs that correct adjacent bit errors or multiple bit errors are utilized. However, multiple bit error-correction codes (ECCs) can correct various bit not only in the adjacent bits but also errors in any error models, the complexity of the decoding process, and the limitation of the code block size reduce their usage. Meanwhile, from the generation principle of MBUs, depending on the initial angle of incidence and scattering angle of the secondary particles, the type of MBUs vary. From this, the dominant error patterns among the multiple bit errors are adjacent bit errors. Therefore, to harden memory against MBUs, adjacent bits correction ECCs becomes widely used at the design stage.

The capability of adjacent bits correction codes mainly concentrates on the double adjacent error correction (DAEC), triple adjacent error correction (TAEC), 3-bit burst error correction (BEC), and very near on Quadruple adjacent error correction (QAEC)[1]. A replacement to codes which corrects adjacent errors is to use an SEC or SEC-DED code merge with memory cells interleaving. Interleaving confirms that cells contained in the same logical word are assigned physically aside. This indicates that a burst error that affects multiple adjacent cells, spread to many words, each having one or two-bit error that can be corrected by an SEC or DEC codes, respectively. Therefore, it is preferable to use SEC or DEC plus interleaving, or a code that can correct adjacent errors will be design-dependent. But if the burst errors affect more adjacent bits, ECCs alone will not be efficient because the generation of such kind of ECC is sophisticated.

As the technology enters to the UDSM, the memory cell area again dropping and memories having atomic-size transistors develops. More memory cells in the word-line direction can be accommodated by ions with the order of magnitude in micrometer, as shown in Fig. 1.2 then the three bits previously considered [2]. This indicates that the SEC-DAEC-TAEC codes may not be practical to ensure memory consistency, demand codes with more advanced correction ability. Therefore, extending the correction ability of QAEC codes would be of interest, especially if it can be done without adding extra parity bits. A combination of convolutional interleaving with ECCs produces great achievement in error correction.

This project presents an improvement of quadruple adjacent error correction (QAEC) codes with conventional

interleaving. The QAEC code design technique for low redundancy was considered from two angles: error space satisfiability and unique syndrome satisfiability. QAEC codes with for 16, 32, and 64 data bits are discussed in Chapter 5. From the angle of the design of the integrated circuits, two criteria had been used to optimize the decoder complexity and decoder delay in the QAEC codes in ECCs level: a) reducing the overall quantity of ones in the parity check matrix and b) reducing the quantity of ones in the weightiest row of the parity check matrix. Moreover, the QAEC codes were found on the conventional recursive backtracing algorithm. The algorithm of QAEC codes is a function of weight restriction and recording the past procedure.

The algorithm used in QAEC remarkably improves the performance of previous 3-bit BEC codes. Combining convolutional interleaving to QAEC provides better performance. The encoders, decoders, interleaver, and deinterleaver are implemented in VHDL.

II. Literature Survey

Robert Baumann [3] examines the influence of radiation effects on advanced electronic systems with an emphasis on observing the newest scaling tendencies for memory and sequential logic, and the forthcoming risks for refining product consistency. As scaling from one-megabit to one-gigabit technology, the DRAM single bit SER has been reduced by about four to five times per generation. But DRAM SER is expected to increase with increasing frequency. Due to reductions in operating voltage and node capacitance, the SRAM single bit SER increased with each succeeding generation.

E. Ibe et al. [4] predict the drifts in terrestrial neutron-induced soft-errors in SRAMs down to 22 nm process by using the Monte-Carlo simulator CORIMS, which is validated to have less than 20% variations from experimental data. The simulation results indicate Soft-error rates per device in SRAMs will increase x6-7 from 130 nm to 22 nm process. As SRAM is scaled down to a lesser dimension, SEU is dominated more extensively by low energy neutrons (< 10 MeV). This may cause severe effects in the reliability design of logic devices. From [3] and [4], the protection of memory against errors- the area of interest of proposed work- must be increased since the radiations extensively affect the memories.

Hsiao [5] demonstrated a new way of constructing a class of SEC-DED codes that uses the same number of check bits as the Hamming SEC-DED code but is superior in cost, performance, and reliability. The necessity of having a minimum number of 1's in the rows of the parity-check matrix permits the fast generation of check bits and syndrome bits. Because of the minimum number of 1's, there is a saving in code implementation. This feature permits minimizing the hardware. Finally, it was shown that this class of codes has better error detecting capability for triple and quadruple errors than do the conventional modified Hamming codes.

A. Dutta et al. [6] derived a method to correct the error through heuristic search, which can identify and correct the most likely double bit errors in memory. The codes in [7] can correct all single error, all double bit error, and all adjacent double-bit error. The decoding process has five steps; most of them are adopted for QAEC codes. The drawback is the possibility of miscorrection for a small subset of multiple errors.

A. Neale et al. [7] present a proposed class of ECCs to explore the check-bit design space between the conventional SEC-DED and BCH DEC codes. The degree of adjacent error detection depends on the number of check-bits used to implement the code. The proposed code's particular case offers both triple adjacent error detection and double adjacent error correction for the same number of check-bits as the conventional SEC-DED code. This coding scheme offers an orthogonal approach to enhance protection and an attractive solution when the practicality of bit interleaving is either limiting or not feasible.

L. J. Saiz-Adalid et al. [8], new SEC-DAEC-TAEC and 3-bit burst error correction codes have been proposed to protect SRAM memories from MCUs. This codes cover the most common memory word lengths. The codes have been synthesized and compared with existing SEC-DAEC codes, and the results show that the new codes provide significant reductions in the number of ones, parity check matrix, and a maximum number of ones in its rows.

S. Shamshiri.et al. [9] suggest error-locality-aware codes for SRAM memories to correct single-bit or multi-bit upsets and even physical defects missed by manufacturing testing or age-induced defects. They gave a general description of the coding problem by merging the idea of random error correction with burst error correction. This work proposed a code for 16-bit memories that corrects four local or two global errors. Syndrome analysis is used to demonstrate the chance of designing codes to correct any arbitrary number of local and global errors for memories of other sizes.

The literature [5]-[9] examines the previous ECCs and the growth of ECCs from single bit correction to triple adjacent error correction.

Srijidtra M et al. [10] implement the convolutional interleavers into MPEG-4 image transmission systems for the Rician fading channels. The complication of the convolutional interleaver is considerably lesser than the random interleaver. With and without a convolutional interleaving process, the performance of the image transmission system was detected. Significant enhancement (about 2.5 dB) is visually noticeable in some image areas, especially on the right eye, with convolutional interleaving. The simulation results show some improvement on the PSNR performance of about 1.4-5 dB for the slow fading channels.

This literature shows the improvements in the results of using the convolutional interleaving in different applications. Centered on these studies, the convolution interleaving is selected to improve the burst error capability of memories.

III. Existing QAEC Codes

A. Binary Block Linear Codes

Codes for SEC-DAEC-DED, SEC-DAEC-TAEC, 3-bit BEC, and very nearly QAEC have been proposed in earlier

works. Mainly most of these codes are binary linear block codes. Based on some laws for linear block codes construction, the design of these codes was developed. The adopted QAEC codes are also binary linear block codes and follow the same construction rules. The number of data bits, k, redundancy bits, (n − k), and the block size of the encoded-word, n are the parameters commonly used to explain binary codes. An (n, k) code is described by its generator matrix G or parity check matrix H in

$$G = [R_{k\times(n-k)} \cdot I_{k\times k}],\ H = [R^T \cdot I_{(n-k)}] \quad (1)$$

where Ik×k is the identity matrix, R is the matrix with size k × (n − k), and RT is the transpose of R. The generator matrix G is used to encode the data bits in the encoding process through the operation in (2).

$$y = x \cdot G \quad (2)$$

where x(x0, x1, . . . , xk−1) are the data bits to be encoded, and y(y0, y1, . . . , yn−1) is the codeword. With the help of parity check matrix H in the decoding process, the received codeword is obtained through the operation in (3).

$$S = w \cdot H^T \quad (3)$$

where w (w_0, w_1, . . . , w_{n-1}) is the received codeword, S(s_0, s_1, . . . , s_{n-k-1}) is the syndrome, and a remarkable element in error correction. The errors introduced into the received code can be described using (4)

$$w = y + f\ (f_0, f_1, \ldots, f_{n-1}) \quad (4)$$

where f (f0, f1, . . . , fn−1) is the error vector pointing that an error happens in the ith bit when fi = 1. When multiple bit errors appear in the received codeword w with the error vector f = (f0, f1, . . . , fn−1), the syndrome of the code considering the error vector in decoding process can be calculated by the method in (5)

$$S = f \cdot H^T \quad (5)$$

This equation formulates the relationship between the syndrome and the corresponding error pattern. Given the detailed form of the parity matrix H, when one error appears in the m^{th} bit, the related syndrome is equal to the m^{th} column vector. When errors occur in the m^{th} bit and the n^{th} bit, the related syndrome is equal to the XOR result of the m^{th} column vector and the n^{th} column vector. Hence, one error can be corrected or detected if it satisfies the following rules.

1) Correctable Restriction: The particular syndrome vector is unique in the set of syndromes.
2) Detectable Restriction: The particular syndrome vector is non-zero.

B. Code Design Technique

This section explains the code design technique for QAEC codes. The process used depends on syndrome decoding. The scanning of the specifications regarding parity check bits and the articulation of the issue is similar to some previous studies like [6] and [8]. The QAEC codes construction can be divided into error space satisfiability problems and unique syndrome satisfiability.

If a code has k data bits and b check bits, it can be represented in 2^k different ways. If one error appears, there are (k +b) bit locations for a single error with output area of (k + b) · 2^k values. If vicinal errors occur, it has (k + b − 1) bit positions for double adjacent errors with output area of (k+b−1) · 2^k values, (k +b−2) bit locations for triple adjacent errors with output area of (k +b −2) · 2^k values, . . . , (k +b −(N − 1)) bit locations for N adjacent errors with output area of (k + b−(N −1)) · 2^k values. If nearly adjacent errors occur, it has (k+b−2) bit locations for errors in 3-bit case with output area of (k + b − 2) · 2^k values, (k + b − 3) bit locations for each type of errors in 4-bit case with output area of (k + b − 3) · 2^k values. To attain the codes that can correct the errors with particular fault models, the sum of the output area value of error patterns should be less than or equal to the whole output area value 2^{k+b}.

A set of error models can be rectified if the corresponding error patterns' syndrome is unique as per the rules of binary block linear codes. For the QAEC code, the error patterns are (. . . , 1111, . . .) for QAEC, (. . . , 111, . . .) for TAEC, (. . . , 11, . . .) for DAEC, (. . . , 1, . . .) for SEC, and (. . . , 101, . . .) for 3-bit nearly adjacent errors correction.The code design is a type of Boolean satisfiability problem. Usually, the solution to this problem is based on the recursive backtracing algorithm

C. Searching Algorithms and Tool Development

In the construction of the QAEC codes, the first step is to ensure the check bits' number. The number of the check bits is fixed, considering the low redundancy. Codeword for data with 16 bits has seven check bits, data with 32 bit has eight, and data with 64 bit has nine.

The recursive backtracing algorithm is the main feature of the algorithm of QAEC codes. The initial matrix is an identity matrix In-k, and the corresponding syndromes of the error patterns are kept in the syndrome set. Then, a column vector added to the right side of the initial matrix, took from the $2^{n-k} - 1$ column options. This process is called as column-added action. Now, for the newly added column, corresponding syndromes are intended. The new syndromes will be inserted into the syndrome set if they are different from the current syndrome set; the column-added action is fruitful. The initial matrix with an added column then updates as the base matrix. Or else, the column-added action collapse and another new column from the options are chosen. If all the options are checked, and still the column-added action not providing the solution, one column from the right side of the base matrix is removed. The corresponding syndromes are also excluded. Then, the algorithm continues the column added action until the matrix columns attain the required value.

For the recursive backtracing algorithm, a high quantity of computing means and computing time is required. The matrix operation was replaced with decimal operation by conversing the column vectors into decimal numbers to decelerate the computational delay of the algorithm action. For QAEC codes, the reasonable computation time was set to one week to consistent with [7].

The column used to create the matrix should have as small weight as possible, for both optimization criteria mentioned previously. To reduce the weight of the matrix, the weight of the added columns is limited. This accelerates the steps to find a better result.To reduce the computation time of the searching process for QAEC codes, the algorithm is made as a function of past procedure recording and based on the column weight restriction technique.

IV. INTERLEAVING

Established many random error correction codes to ensure data fidelity. However, ECC is not useful in fighting bursts of errors, i.e., a cluster of consecutive or linked erroneous code symbols due to the bursty nature of errors. Interleaving is a method of rearranging code symbols to spread bursts of errors over multiple codewords that ECC can correct. Error-correcting codes are mostly aimed to correct random errors. Interleaving converting bursts errors into random errors becomes an effective means to fight error bursts.

The data bits are settled over several code blocks during interleaving before storing to memory by the interleaver. This helps to spread lengthy burst error sequences between many blocks. These errors look like independent random errors or burst errors with small lengths when the decoder rearranges the blocks. By using the error-correcting algorithm, the decoder detects and corrects the errors.

A. Types of Interleaving

The interleaving method broadly classified into periodic interleaving and pseudo-random interleaving. The message is arranged in a repeating series of bytes in periodic interleaving. Before the transmission, the interleaver receives data bits as blocks and implements equal permutations on the blocks. For illustration, the sequential blocks of code may be written to a matrix in a column-wise style and then read out from the matrix in a raw-wise manner. Block interleaving is an example of periodic interleaving. Pseudo-random interleaving comprises reordering the message blocks in a pseudo-random sequence created by specific algorithms.

B. Block interleaver

Block Interleaver is a common interleaver hired in digital data transmission. Compared with other types of interleavers that have more complications in realizing, the block interleaver is comfortable and straightforward. A block interleaver is similar to a row-column matrix of dimension A×B, where A is the total number of rows, and B is the total number of columns. The data is written to the row-column matrix row-wise to execute interleaving and read back in column-wise. Inter-row or inter-column permutations are not used. Thus the interleaved sequence holds some periodic arrangement. The block deinterleaver implements the reverse operation of the interleaver in which the information is written to the AxB row-column matrix column-wise and read back in row-wise.

In block interleaving, the deinteleaved outputs are read consecutively. The receiver can read a single row only once it receives the whole message and not earlier than that. The receiver also needs a large area for memory to keep the received codewords and has to store the entire data. Thus, these reasons bring two drawbacks, the latency and storage (a large amount of memory). These drawbacks can be removed by using the convolutional interleaver described below.

C. Convolutional Interleaver

Convolutional interleaver is a kind of multiplexer-demultiplexer system. Delay lines are used to gradually increase length in this system. A convolutional interleaver consists of several shift registers. The shift registers have a fixed delay each, which are positive integer multiples of a fixed integer. Every new data to the input of the interleaver is fed to the next shift register, and the previous data in that register becomes part of the interleaver output. A convolutional interleaver has memory; that is, its operation depends not only on current symbols but also on previous symbols. In a typical convolutional interleaver, the delays are non-negative multiples of a fixed integer. (Although a general multiplexed interleaver allows unrestricted delay values).

The output of convolutional interleaver is the diagonal symbols created at the end of every delay line. We can read the first row at the receiver when the input multiplexer switch finishes about half switching. Thus, we need to store a maximum of about half a message at the receiver to read the first row, which considerably diminishes the storage area demand by half. The latency is also reduced by half as only half the message needs to read the first row, which is remarkable progress over the block interleaver. Thus, the entire interleaver memory is divided for transmitter and receiver.

V. PROCEDURES OF PROPOSED METHOD

This section elaborates on the encoding, interleaving, deinterleaving, and decoding procedure of the proposed method. The fundamental theory of encoding and decoding was discussed in Section III and interleaving in Chapter 6.

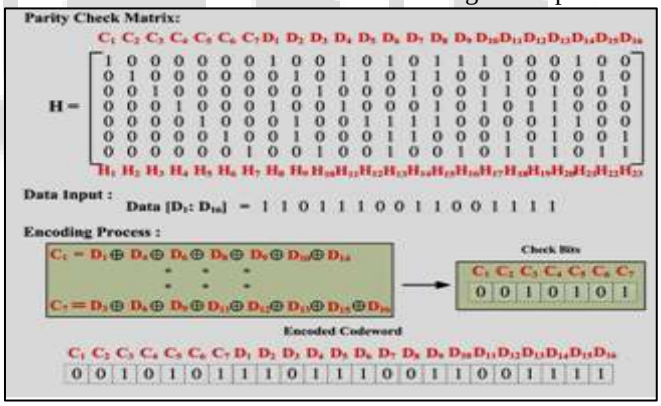

Fig. 1: Procedure of encoding

Illustrated example of encoding with 16-bit data as in Fig.1 with the H matrix. Based on the parity check matrix structure, the check bits are calculated by the corresponding data bits. The encoded codeword is the combination of check bits and data bits, 23 bit long.

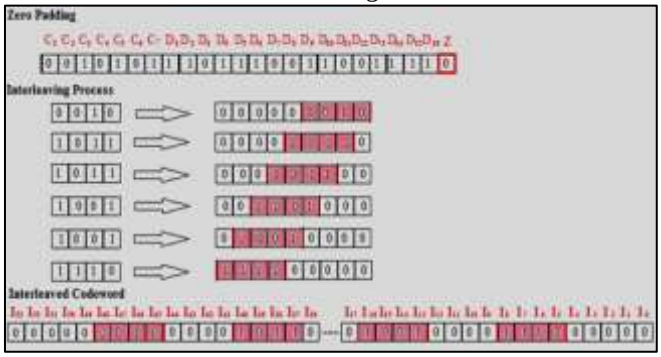

Fig. 2: Procedure of convolutional interleaving

After encoding, interleaving has to perform. In interleaving, the bits in the codeword should divide equally. So the number of bits in the codeword after encoding must be even. Padded a zero to make the number of bits in the codeword even, to the codeword's LSB. Now the codeword is 24 bit long, and it is divided into six blocks, each containing 4 data bits and interleaved the data bits with insignificant bits since convolutional interleaving is using. In each block of the codeword, the irrelevant bits are added differently according to the delay. Each block has 9 bits, and the interleaved codeword of 54 bits is stored in the memory, as shown in Fig.2.

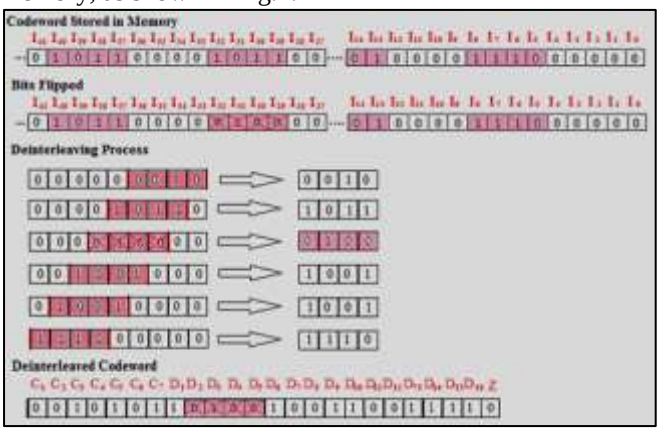

Fig. 3: Procedure of conventional deinterleaving

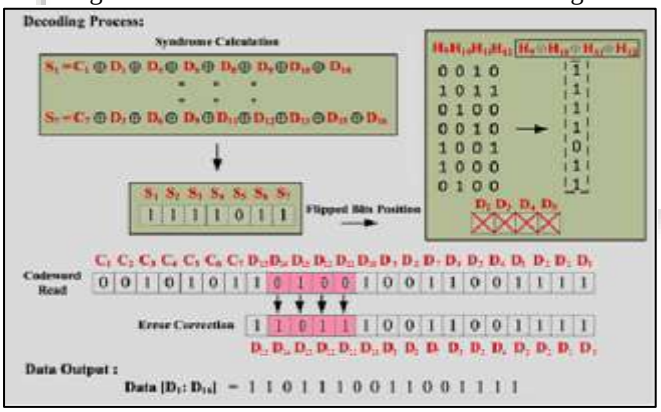

Fig. 4: Procedure of decoding

Due to the particle hit in memory, the contents stored to corresponding memories are flipped, causing MBUs. To explain deinterleaving, considered quadruple adjacent bits are flipped on I29, I30, I31, and I32 of the interleaved codeword. In the deinterleaving process- the opposite of interleaving- the interleaved codeword is equally divided into six blocks, each of 9bits. Removed the insignificant bits, the codeword of 24 bit is obtained. Illustrated the deinterleaving process in Fig.3.

Eliminated the zero-padded to the LSB before decoding. Next, calculating the syndrome using the stored check bits and data bits and the parity check matrix in the decoding process. The flipped bits can be located through the corresponding relationship between the syndrome and the XOR result of the columns mentioned in Section III, corrected the errors from the storage stage in the memory effectively with the flipped bits inverted. The above descriptions detail the whole procedure of the proposed method.

VI. Simulation and Result

Xilinx ISE Design Suite 14.7 is used as coding software and done the simulations using the ISim simulator. Used VHDL as the programming language. It is done the simulations in two parts. The first part contains the encoding and interleaving procedures that are performed before the data stored in the memory. The second part illustrates the errors in the codeword stored in memory, deinterleaving, and decoding.

Fig.5: Simulation result of encoding and interleaving

The Fig.5. shows the simulation result of encoding and interleaving. The 16-bit data, 7-bit check bits, 23-bit codeword, 24-bit zero-padded codeword, and interleaved codeword can be seen in the simulation result.

Three cases of errors are considered to elaborate on the correction ability of the proposed method. In the first case, let the radiation cause error only in data bits. Adjacent 4 bits of data are flipped, and it can be simply corrected using the QAEC codes. The simulation result in Fig.6 shows this case.

Fig. 6: Simulation result of deinterleaving and decoding – error in 4 data bits

The second type of burst error causes error in 6 bits in the interleaved codeword, marked with a round in the simulation result shown in Fig.7. Out of these six bits, two bits are insignificant bits added during the convolutional interleaving process. Therefore these two bits will be removed at the time of deinterleaving. The burst error will be reduced to 4 adjacent bits, showing that a burst error that affects more than four adjacent bits can be corrected using the QAEC codes with conventional interleaving. And the number of bits that are flipped due to radiation can be extended to 8 adjacent bits which can be corrected using the proposed method. This feature is the attractive part of this project.

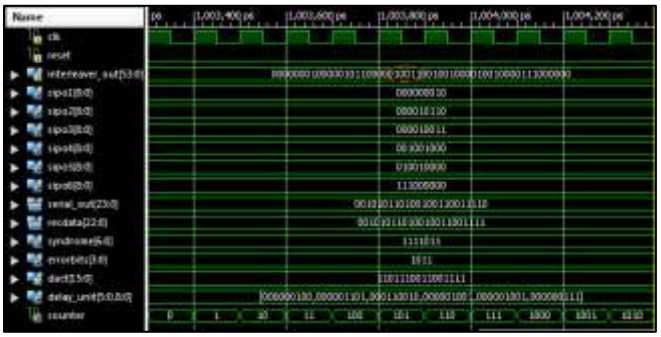

Fig. 7: Simulation result of deinterleaving and decoding – error in 6 bits of the interleaved codeword

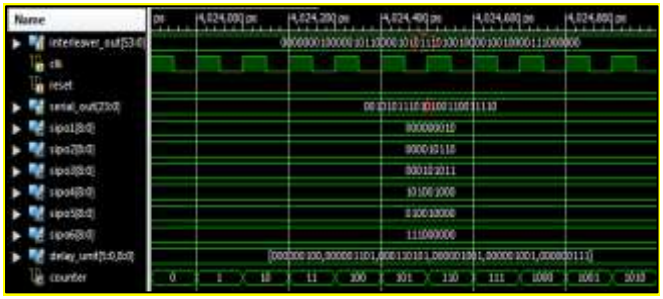

Fig. 8: Simulation result of deinterleaving - error in insignificant bits

The third case considering a burst error of 4 adjacent bits, in which only one bit is data bit. The remaining three are insignificant bits. These will remove during the deinterleaving process as in the second case. So that in the deinterleaved codeword, only one bit has an error, as shown in Fig.8. This error can be corrected using single error correction codes.

VII. Conclusion

Error-correction codes are necessary to ensure the reliability of electronic systems using memories. The existing QAEC can correct errors in four adjacent bits. And interleaving is an excellent method to prevent burst errors. The proposed method combining interleaving with QAEC codes gives better performance, limits the error in data bits to 4 adjacent bits. Even if more than four adjacent bits in the codeword stored in memory get effected, the error-correction can be done using this proposed method-no need of ECC that can correct more adjacent errors, which will be more complicated.

Acknowledgment

I would like to convey my heartfelt gratitude towards my guide Ms. Sindhu T.V for her constant guidance, encouraging help and inspiring words. I am thankful to the department of Electronics and Communication engineering for their support.

References

[1] Jiaqiang Li, Pedro Reviriego, Liyi Xiao, Costas Argyrides, and Jie Li "*Extending 3-bit Burst Error-Correction Codes With Quadruple Adjacent Error Correction*", in IEEE Trans. on Very Large Scale Integration (VLSI) systems, vol. 26, NO. 2, February 2018.

[2] I. Chumakov, A. V. Sogoyan, A. B. Boruzdina, A. A. Smolin, and A. A. Pechenkin, "*Multiple cell upset mechanisms in SRAMs*," in Proc. 15th Eur. Conf. Radiation Effects Compon. Syst., pp. 1–5, Sep. 2015.

[3] Robert Baumann, "*The impact of technology scaling on soft error rate performance and limits to the efficacy of error correction*," in Digest. International Electron Devices Meeting, pp. 329-332, Dec. 2002.

[4] E. Ibe, H. Taniguchi, Y. Yahagi, K.-I. 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.

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

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

[7] 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.

[8] L. J. Saiz-Adalid, P. Reviriego, P. Gil, S. Pontarelli, and J. A. Maestro, "*MCU tolerance in SRAMs through low-redundancy triple adjacent error correction*," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 23, no. 10, pp. 2332–2336, Oct. 2015.

[9] S. Shamshiri and K.-T. Cheng, "*Error-locality-aware linear coding to correct multi-bit upsets in SRAMs*," in Proc. IEEE Int. Test Conf., Nov. 2010, pp. 1–10.

[10] Srijidtra M and Teenarat A, "*Efficient design of convolutional interleavers in mpeg-4 image wireless transmission systems*," in 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, vol. 2, pp. 1-4, May 2009

