EAI Endorsed Transactions
on Scalable Information Systems

Research Article EAI.EU

Fault Detection and Analysis in SRAM through Self-Refreshing Operation

P. Nirmalkumar [1], K. Mythily[2] , Deepa Jose[3,*], B. Arun Kumar [4]

[1,2]Department of Electronics and CommunicationEngineering, CEG Campus – Anna University Chennai, India
[3]Department of ECE, KCG College of Technology, Karapakkam, Chennai-600097, India
[4]School of Computer Science and Engineering, SCOPE, VIT-AP University, Amaravati, Andhra Pradesh, India

Abstract

Numerous soft faults in SRAM memory emerge as technological innovations scales down, resulting in single and several cell upsets. The increased use of transistors in space applications has rendered semiconductor devices more vulnerable to soft errors caused by harm from radiation. A single event upset (SEU) is occurring whenever a soft error produces a tiny bit flipped in a storage device. Because SEU faults affect system performance, they must be addressed as soon as possible. Error-correcting codes, like the method known as (7,4) hamming codes, were devised and their decoding and encoding procedures were verified. It also used to detect single errors that can be fixed. It also helps detect double errors, SECDED however repair of double errors is tough. The decoding and encoding techniques of these approaches were investigated, and all computational findings had been verified and executed in a Xilinx NEXYS 4 DDR FPGA board.

Keywords: SRAM memory, Single Event Upset, soft error, (7,4) Hamming code, SECDED

Received on 10 July 2023, accepted on 02 October 2023, published on 16 October 2023

doi: 10.4108/eetsis.4025

1. Introduction

The level of sophistication of electronic devices and systems has grown as a result of advancements in IC fabrication technologies. Transistor scaling allows for the execution of complex functions without suffering a substantialarea cost. Greater use of electricity is a result of a rise in IC concentration. The operational voltages have been reducedfurther in order to meet the minimal power allowance. Due toray attacks, the durability of the IC is impaired in use in space due to both the decrease in dimension and voltage scalability [1-2]. The voltage changes and potential for mistakes that result from ray hits on semiconductor substances are caused by charged particles.

Soft errors occur when alpha particle neutrons emerge from the chip's packing material. Transients occurring at circuit nodes damage the data contained in memories. When a single data bit is corrupted, this is known as being a single event upset; when many cells are affected, this is referred to as several cell upsets. Because they merely taint the data and do not harm the entire device, they are referred to as soft error. Furthermore, as bit cell density grows, several nearby cells become corrupted, resulting in many multiple bit and multiple cell upsets.

Total Ionised Dosage (TID), singular events latch-up (SEL) & single events upset (SEU) are all examples of radioactivity-related impacts that have been noted. While SEU is the instantaneous result of an individual radiated charge particle, SEL and TID have long-lasting consequences. Higher-energy ions caught in gates of transistors result in a long-lasting radioactive effect known as TID. This alters the transistors' cutoff voltage, leading to functional faults and increased current loss. The gate becomes thicker as the TID rises. When the oxide around the gate depth is beneath 12 nm, or IC manufacturing technique smaller than 0.35m, TID is seen to be significantly reduced.

As a consequence, for current strong-submicron IC

[*] Corresponding author. Email: deepajose11@gmail.com

fabrication processes, the TID toughness is very high. On theopposite side, in the deeper submicron era, SEL and SEU have grown more difficult for ICs. This is caused by the ongoing lowering of IC design distortion margins brought about by smaller feature shapes and sizes, less supply volts, and greater operating frequency. SEL produces a lot of energy and current, which eventually destroys the chip. Bit flip in memory or pulsating/glitches in internal regulate/data signals of ICs are common effects of SEU. According to reports, pulses or glitch lengths fall between a couple of hundred picoseconds to a several nanoseconds. According to NASA, SEU error rates for commercial parts for low-earth orbit (a distance of 500 kilometres) are expected to be 105 errors/bit-day throughout the north and south hemispheres [3-7].

The consequences of soft-error are frequently taken into account to ensure adequate operation in aerospace applications. There are several options for reducing these consequences. For instance, to increase the likelihood of success when soft errors are made during an operation, the process is repeated numerous times. However, the likelihoodof getting accurate results is slim if the errors involve controlling signals in the hardware [8]. A hardware resilience solution like Triple Module Redundancy (TMR) can increaseradiation resistance at the design level.

However, due to the redundancy, this solution requires arise in both area and power. Furthermore, errors in cells containing making choices elements or at a clock root are challenging to identify and exact. Special techniques, such as semiconductors-on-insulator, may improve the SEU challenges at the level of the device but are unable to eradicate them and are extremely costly to produce. To reduce the data disruption triggered by SEU, the storage RAM bit-cell is changed at the level of the circuit to contain extra devices [9-12].

The SRAM cell is the primary part of the SRAM clusters. Each cell holds a single element of data. As long asan SRAM cell receives power, it does not require periodic revival. It provides constant perusal and composition of tasks to be performed in it. The traditional 6T SRAM cell consists of two intersect-coupled inverters connected to matching piece lines via accessing transistor. These entry transistors construct the information to be stored, and the input to be read is completed by connecting the opposite piece wires to the sensing speakers.

The Static Noise Margin measures the stability of SRAM cell design. The SNM statistics may be derived for three distinct SRAM jobs, notably the READ, WRITE, and HOLD activities. The SNM representation is created by assuming the VTC curve of the cell's two inverter devices, which results in a double-lobed bend referred to as the butterfly shape bend. The largest possible square produced from a curve provides sustainability information. Because ofits exceptionally low space use, the traditional 6T SRAM configuration is commonly used. Regardless, it has extremely poor read and write stability and thus appears for the framework of an effective SRAM, which is cell.

The 8T is the one SRAM cell that was used in this work. The read bit line of the 8T SRAM column is detached from all internal cellular nodes, which distinguishes it from precedingcells. As a consequence, there is no cell disruption throughoutthe reading process, showing improved radiation resistance. Because the proposed self-refresh method requires common reading operations for error detection and repair, an 8T SRAM, also known that has no read disruption is preferred. The decoupling SRAM unit allows for more versatility when performing a real self-refresh function, which includes correction of errors and identification. We use the ECC with Single Correction of Errors and Double Detection of Errors (SECDED) to manage SEUs in the SRAM to create area- efficient radio tolerance. If the decoder output changes by onebit, it is readily fixed. If the decoder output changes by two bits, it cannot be fixed; it can only be identified. The primary objective for this research is to identify and correct these types of faults so that transmission of data and receipt regulareven when there is noise interference through the channel.

2. Materials and Methods

2.1 8T SRAM CELL

The SRAM of 8T on the other hand, offers two decoupling channels for perusing and composing. This demonstrates excellent peruse and creates strength, making ita better choice for preparing the SRAM display. The 8T SRAM cell is made up of a pair of bit lines (BL and BLB) connected by twin NMOS access transistors, and a hubwherein the bit is stored is connected to the entrance of an additional transistor whose power supply is connected to ground. This transistor's channel is linked to the source of another transistor, and a line that provides control for read action is transmitted to the transistor's entrance called to be the Read Word Line (RWL). The Read Bit Line, or RBL, provides the groove yield and is precharged prior to being reviewed. When bit 1 is constructed using BL, device N5 turns on, once RWL is provided, device N6 switches on, draining the charge stored in RWL and producing a corresponding yield.

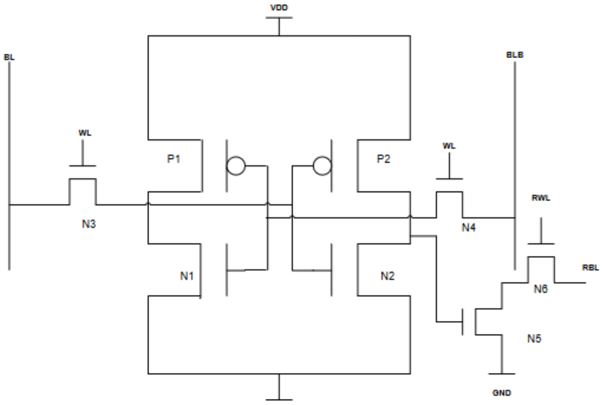


Figure 1. Conventional 8T SRAM

2.1 ERROR CORRECTION TECHNIQUES

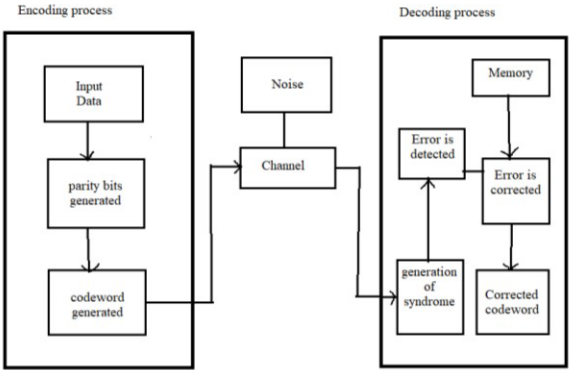


Figure 2. Block diagram for error detection and error correction

Fig. 2 shows a parity check bit producer is a hybrid device on the transmission side that receives the original information as input and creates a bit of parity as return based on the original information. The device is used to send the original information with bits of parity. Those parity bits constitute both odd & even. The corresponding even parity amount of one is into evens, whereas an odd parity amount of ones is uneven.

This LCPC code is implemented to create the code term. This has minimal complexity as well as a low memory need. In an instance of a busy channel, syndromes analysis is an extremely efficient translation method. It additionally comes with the shortest distance decoding method and the smallest look up table.

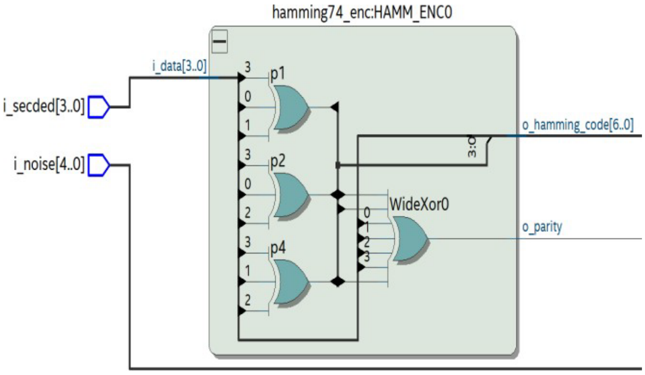


Figure 3. Hamming code Encoder Diagram

Linear correction codes are widely utilised in telecommunication and storage applications. Hamming codeencoder diagram is shown in Fig 3. These are used to identify and correct one-bit errors with a least amount of redundancies. The hamming code (7,4) is shown here. Where 7 indicates the entire code word, 4 represents the data bits, and 7-4=3 the bits for parity must be introduced to the info bits. The disadvantage of this method is the fact that it is limited to use for one bit error identification and rectification. There are three bits of parity P1, P2, P4 and four bits for data D1, D2, D3, D4 in this Hamming Code Encoding method, then the entire encrypted coded word contains of seven bits are P1, P2, D1, P3, D2, D3, D4. The values that follow are the parity expressions:

P1=D3^D5^D7

P2=D3^D6^D7

P3=D5^D6^D7

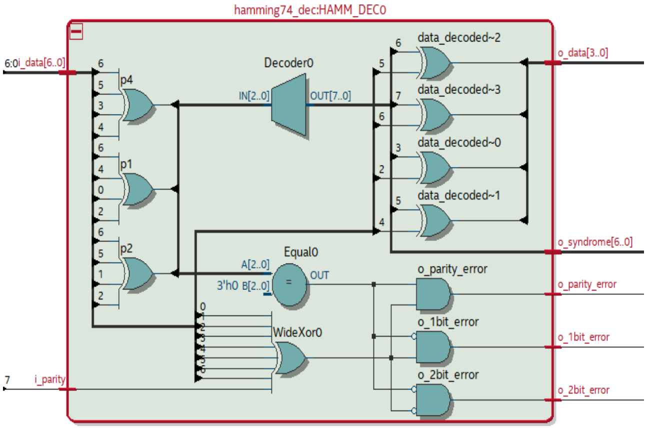


Figure 4. Hamming code Decoder Diagram

Fig. 4 shows there are three check bits that are utilised C 1, C 2, and C3, which is as well as four bits for data plus threeparity bits, as in the encoder phase. It contains a 3-to-8-bit decoder that accepts three verification bits as input and provides eight outputs, one of which is an error-free signal, and the others undergo a XOR operation with each of their data and bits of parity. The formulae for bits to check are as follows.

C1=D3^D5^D7^P1

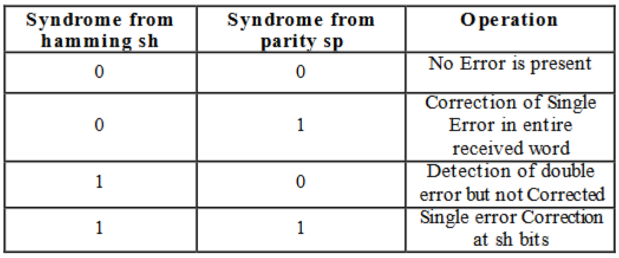
C2=D3^D6^D7^P2

C3=D5^D6^D7^P3

Hamming codes are expanded to provide for single correction of errors and double detection of errors. As a result, these are referred to as SEC- DED ("Single Error Correction Double Error Detection") or expanded hamming code. They needed an additional bit for double detection of errors. It is computed as an even parity for the entire encrypted output. The (7,4) Hamming code is turned into the(8,4) SEC-DED code by adding an additional bit, say d7. d7 is an additional bit and may be represented in the following manner:

$$d7 = d0 + d1 + d2 + d3 + d4 + d5 + d6$$

All bits are XOR together. In this case, the method of decoding is employed to check two requirements: the level of parity of the transmitted word and the syndrome calculation. The operation table for Single Correction of Error and Double Detection of Errors is represented in Fig. 5.



Syndrome from hamming sh	Syndrome from parity sp	Operation
0	0	No Error is present
0	1	Correction of Single Error in entire received word
1	0	Detection of double error but not Corrected
1	1	Single error Correction at sh bits

Figure 5. SECDED operation table

2.2 Read operation for 8T SRAM

The process of reading action is started by connecting the read end of the bit line onto VDD, as is done in the traditional one. The action of reading a word line (RWL) turns on to the access mosfet M5. When the value that is recorded at Q is '0,' mosfet M6 will be ON, then RBL will be linked with the ground immediately through the discharging of M5&M6 transistor. This means that the information stored within SRAM at Q is a value of zero. The8T SRAM which is used for read operation is shown in Fig.6.

If the quantity recorded at Q is '1', the M6 mosfet will be turned off, so there will be no power pathway for RBL, and the data in RBL will be VDD, indicating that the valuesrecorded at Q was '1'.

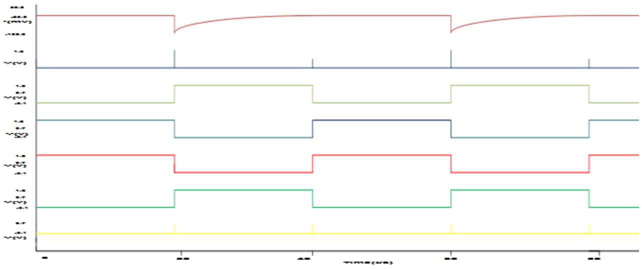


Figure 5. 8T SRAM operation

2.3 Write operation for 8T SRAM

The data writing function of the 8T SRAM cell in Fig. 7is identical to that of the ordinary 6T SRAM. The write function in 8T SRAM is demonstrated and detailed below. The bit line must supply no voltage and VDD through the bit line (BLbar) in order to write '0'. And the write word lineis triggered, causing each of the transistors M3 as well as M4 to turn on. As a result, the quantity in a bit line is saved at Q. As a result, '0' is saved at Q.

Similarly, the letter '1' is most likely conveyed in asimilar way. The bit line must have a value of VDD, and the voltage on the bit line bar must have a value of zero volts. Because WWL has been set up for write operations, the values specified in bit lines are stored at the specific nodes, so at Q, the resultant value has logical '1' and at Qbar, the value has logical '0'. When contrasted to the basic SRAM functioning, there is no difference in the operation of writing.

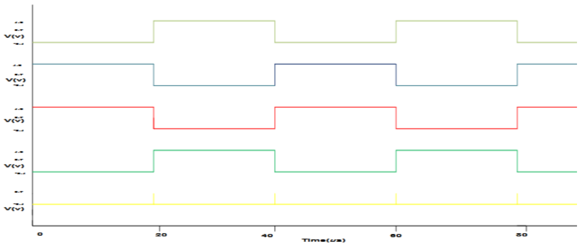


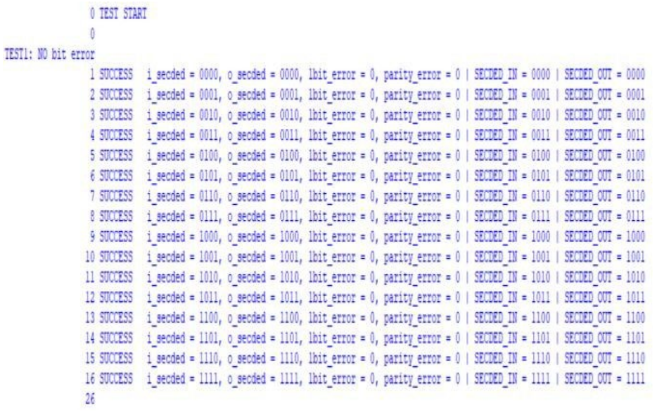
Figure 7. 8T SRAM Write operation

2.4 Transcript Output

The transcript output for no bit error, one biterror, & double bit error are discussed here.

2.4.1 Transcript Output for no bit error

The generated combinations of 16 possible input data is loaded in the vivado tool. Here no noise is added. So the input is the same as the output. The output results are shown below Fig. 8.

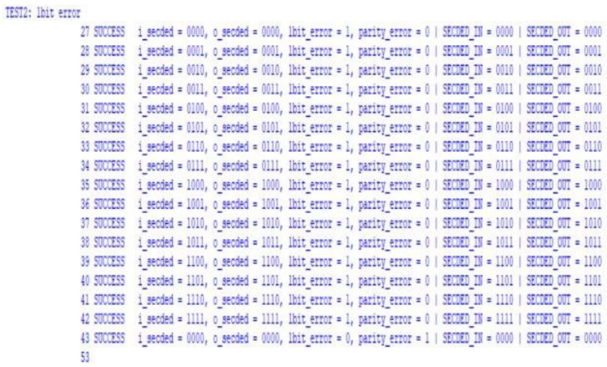


```
          0 TEST START
          0
TEST1: NO bit error
          1 SUCCESS   i_secded = 0000, o_secded = 0000, 1bit_error = 0, parity_error = 0 | SECDED_IN = 0000 | SECDED_OUT = 0000
          2 SUCCESS   i_secded = 0001, o_secded = 0001, 1bit_error = 0, parity_error = 0 | SECDED_IN = 0001 | SECDED_OUT = 0001
          3 SUCCESS   i_secded = 0010, o_secded = 0010, 1bit_error = 0, parity_error = 0 | SECDED_IN = 0010 | SECDED_OUT = 0010
          4 SUCCESS   i_secded = 0011, o_secded = 0011, 1bit_error = 0, parity_error = 0 | SECDED_IN = 0011 | SECDED_OUT = 0011
          5 SUCCESS   i_secded = 0100, o_secded = 0100, 1bit_error = 0, parity_error = 0 | SECDED_IN = 0100 | SECDED_OUT = 0100
          6 SUCCESS   i_secded = 0101, o_secded = 0101, 1bit_error = 0, parity_error = 0 | SECDED_IN = 0101 | SECDED_OUT = 0101
          7 SUCCESS   i_secded = 0110, o_secded = 0110, 1bit_error = 0, parity_error = 0 | SECDED_IN = 0110 | SECDED_OUT = 0110
          8 SUCCESS   i_secded = 0111, o_secded = 0111, 1bit_error = 0, parity_error = 0 | SECDED_IN = 0111 | SECDED_OUT = 0111
          9 SUCCESS   i_secded = 1000, o_secded = 1000, 1bit_error = 0, parity_error = 0 | SECDED_IN = 1000 | SECDED_OUT = 1000
         10 SUCCESS   i_secded = 1001, o_secded = 1001, 1bit_error = 0, parity_error = 0 | SECDED_IN = 1001 | SECDED_OUT = 1001
         11 SUCCESS   i_secded = 1010, o_secded = 1010, 1bit_error = 0, parity_error = 0 | SECDED_IN = 1010 | SECDED_OUT = 1010
         12 SUCCESS   i_secded = 1011, o_secded = 1011, 1bit_error = 0, parity_error = 0 | SECDED_IN = 1011 | SECDED_OUT = 1011
         13 SUCCESS   i_secded = 1100, o_secded = 1100, 1bit_error = 0, parity_error = 0 | SECDED_IN = 1100 | SECDED_OUT = 1100
         14 SUCCESS   i_secded = 1101, o_secded = 1101, 1bit_error = 0, parity_error = 0 | SECDED_IN = 1101 | SECDED_OUT = 1101
         15 SUCCESS   i_secded = 1110, o_secded = 1110, 1bit_error = 0, parity_error = 0 | SECDED_IN = 1110 | SECDED_OUT = 1110
         16 SUCCESS   i_secded = 1111, o_secded = 1111, 1bit_error = 0, parity_error = 0 | SECDED_IN = 1111 | SECDED_OUT = 1111
         26
```

Figure 8. Transcript Output for no bit error

2.4.2 Transcript Output for one bit error

The generated combinations of 16 posssible input data is loaded in the vivado tool. Here noise is added. The output data has one location is modified. The output results are shown in below Fig. 9.

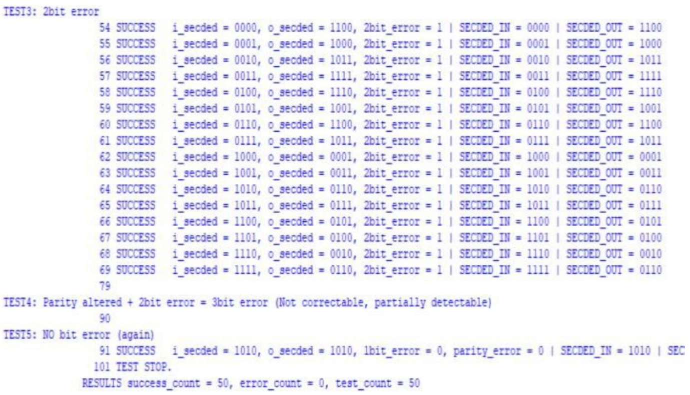


```
TEST2: 1bit error
          27 SUCCESS   i_secded = 0000, o_secded = 0000, 1bit_error = 1, parity_error = 0 | SECDED_IN = 0000 | SECDED_OUT = 0000
          28 SUCCESS   i_secded = 0001, o_secded = 0001, 1bit_error = 1, parity_error = 0 | SECDED_IN = 0001 | SECDED_OUT = 0001
          29 SUCCESS   i_secded = 0010, o_secded = 0010, 1bit_error = 1, parity_error = 0 | SECDED_IN = 0010 | SECDED_OUT = 0010
          30 SUCCESS   i_secded = 0011, o_secded = 0011, 1bit_error = 1, parity_error = 0 | SECDED_IN = 0011 | SECDED_OUT = 0011
          31 SUCCESS   i_secded = 0100, o_secded = 0100, 1bit_error = 1, parity_error = 0 | SECDED_IN = 0100 | SECDED_OUT = 0100
          32 SUCCESS   i_secded = 0101, o_secded = 0101, 1bit_error = 1, parity_error = 0 | SECDED_IN = 0101 | SECDED_OUT = 0101
          33 SUCCESS   i_secded = 0110, o_secded = 0110, 1bit_error = 1, parity_error = 0 | SECDED_IN = 0110 | SECDED_OUT = 0110
          34 SUCCESS   i_secded = 0111, o_secded = 0111, 1bit_error = 1, parity_error = 0 | SECDED_IN = 0111 | SECDED_OUT = 0111
          35 SUCCESS   i_secded = 1000, o_secded = 1000, 1bit_error = 1, parity_error = 0 | SECDED_IN = 1000 | SECDED_OUT = 1000
          36 SUCCESS   i_secded = 1001, o_secded = 1001, 1bit_error = 1, parity_error = 0 | SECDED_IN = 1001 | SECDED_OUT = 1001
          37 SUCCESS   i_secded = 1010, o_secded = 1010, 1bit_error = 1, parity_error = 0 | SECDED_IN = 1010 | SECDED_OUT = 1010
          38 SUCCESS   i_secded = 1011, o_secded = 1011, 1bit_error = 1, parity_error = 0 | SECDED_IN = 1011 | SECDED_OUT = 1011
          39 SUCCESS   i_secded = 1100, o_secded = 1100, 1bit_error = 1, parity_error = 0 | SECDED_IN = 1100 | SECDED_OUT = 1100
          40 SUCCESS   i_secded = 1101, o_secded = 1101, 1bit_error = 1, parity_error = 0 | SECDED_IN = 1101 | SECDED_OUT = 1101
          41 SUCCESS   i_secded = 1110, o_secded = 1110, 1bit_error = 1, parity_error = 0 | SECDED_IN = 1110 | SECDED_OUT = 1110
          42 SUCCESS   i_secded = 1111, o_secded = 1111, 1bit_error = 1, parity_error = 0 | SECDED_IN = 1111 | SECDED_OUT = 1111
          43 SUCCESS   i_secded = 0000, o_secded = 0000, 1bit_error = 0, parity_error = 1 | SECDED_IN = 0000 | SECDED_OUT = 0000
          53
```

Figure 9. Transcript Output for one bit error

2.4.3 Transcript Output for double bit error

The generated combinations of 16 posssible inputdata are loaded in the vivado tool. Here noise is added. The output data has double bit is changed. The output results are shown below Fig. 10.



```
TEST3: 2bit error
             54 SUCCESS   i_secded = 0000, o_secded = 1100, 2bit_error = 1 | SECDED_IN = 0000 | SECDED_OUT = 1100
             55 SUCCESS   i_secded = 0001, o_secded = 1000, 2bit_error = 1 | SECDED_IN = 0001 | SECDED_OUT = 1000
             56 SUCCESS   i_secded = 0010, o_secded = 1011, 2bit_error = 1 | SECDED_IN = 0010 | SECDED_OUT = 1011
             57 SUCCESS   i_secded = 0011, o_secded = 1111, 2bit_error = 1 | SECDED_IN = 0011 | SECDED_OUT = 1111
             58 SUCCESS   i_secded = 0100, o_secded = 1110, 2bit_error = 1 | SECDED_IN = 0100 | SECDED_OUT = 1110
             59 SUCCESS   i_secded = 0101, o_secded = 1001, 2bit_error = 1 | SECDED_IN = 0101 | SECDED_OUT = 1001
             60 SUCCESS   i_secded = 0110, o_secded = 1100, 2bit_error = 1 | SECDED_IN = 0110 | SECDED_OUT = 1100
             61 SUCCESS   i_secded = 0111, o_secded = 1011, 2bit_error = 1 | SECDED_IN = 0111 | SECDED_OUT = 1011
             62 SUCCESS   i_secded = 1000, o_secded = 0001, 2bit_error = 1 | SECDED_IN = 1000 | SECDED_OUT = 0001
             63 SUCCESS   i_secded = 1001, o_secded = 0011, 2bit_error = 1 | SECDED_IN = 1001 | SECDED_OUT = 0011
             64 SUCCESS   i_secded = 1010, o_secded = 0110, 2bit_error = 1 | SECDED_IN = 1010 | SECDED_OUT = 0110
             65 SUCCESS   i_secded = 1011, o_secded = 0111, 2bit_error = 1 | SECDED_IN = 1011 | SECDED_OUT = 0111
             66 SUCCESS   i_secded = 1100, o_secded = 0101, 2bit_error = 1 | SECDED_IN = 1100 | SECDED_OUT = 0101
             67 SUCCESS   i_secded = 1101, o_secded = 0100, 2bit_error = 1 | SECDED_IN = 1101 | SECDED_OUT = 0100
             68 SUCCESS   i_secded = 1110, o_secded = 0010, 2bit_error = 1 | SECDED_IN = 1110 | SECDED_OUT = 0010
             69 SUCCESS   i_secded = 1111, o_secded = 0110, 2bit_error = 1 | SECDED_IN = 1111 | SECDED_OUT = 0110
             79
TEST4: Parity altered + 2bit error = 3bit error (Not correctable, partially detectable)
             90
TEST5: NO bit error (again)
             91 SUCCESS   i_secded = 1010, o_secded = 1010, 1bit_error = 0, parity_error = 0 | SECDED_IN = 1010 | SEC
            101 TEST STOP.
          RESULTS success_count = 50, error_count = 0, test_count = 50
```

Figure 10. Transcript Output for double bit error

2.4.4 Simulation output of SECDED

All the 16 possible combinations of a 4-bit input data results are shown in Fig. 11. With noise and without noise are added here**.**

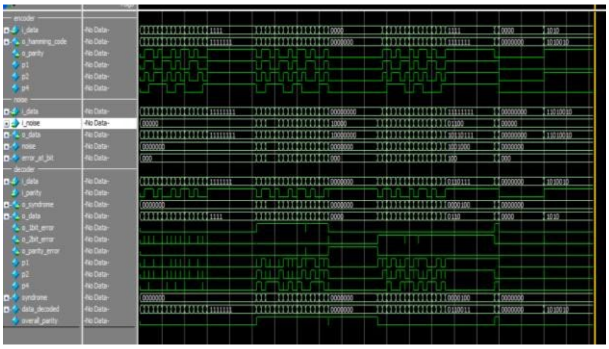
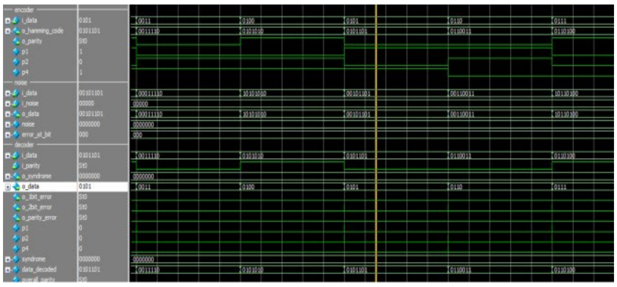


Figure 11. Simulation waveform for SECDED

2.4.5 Output for No Bit error

In Fig. 12 the Encoder [3:0] = 0101 is introduced and no noise is added in the bit position so the given input of the codeword is same as the decoder output. Here no bits are changed because there is no noise included hence it has no bit error.

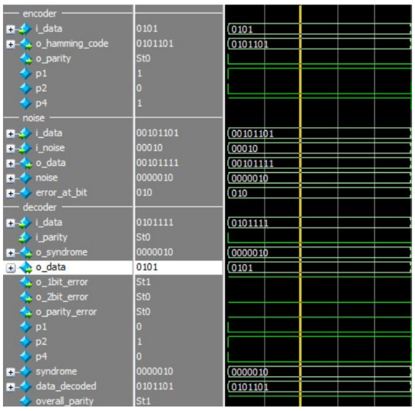


```
encoder
i_data            0101
o_hamming_code    0101101
o_parity          St0
p1                1
p2                0
p4                1
noise
i_data            00101101
i_noise           00000
o_data            00101101
noise             0000000
error_at_bit      000
decoder
i_data            0101101
i_parity          St0
o_syndrome        0000000
o_data            0101
o_1bit_error      St0
o_2bit_error      St0
o_parity_error    St0
p1                0
p2                0
p4                0
syndrome          0000000
data_decoded      0101101
overall_parity    St0
```

Figure 12. Output for No bit error

2.4.6 Output for one Bit error

In Fig. 13 the encoder input [3:0] = 0101 is introduced and noise is added with the hamming code encoder. The output of decoder has 1 bit error detection and correctiontook place as a result.

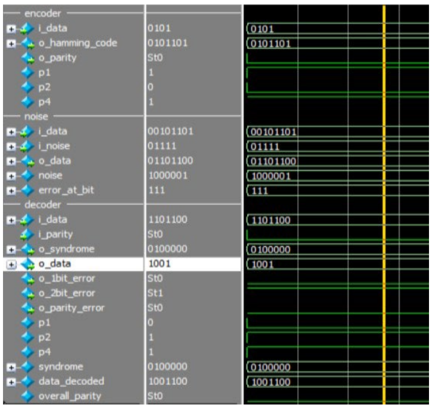


```
encoder
i_data            0101
o_hamming_code    0101101
o_parity          St0
p1                1
p2                0
p4                1
noise
i_data            00101101
i_noise           00010
o_data            00101111
noise             0000010
error_at_bit      010
decoder
i_data            0101111
i_parity          St0
o_syndrome        0000010
o_data            0101
o_1bit_error      St1
o_2bit_error      St0
o_parity_error    St0
p1                0
p2                1
p4                0
syndrome          0000010
data_decoded      0101101
overall_parity    St1
```

Figure 13. Output for one bit error

2.4.7 Output for Double Bit error

In Fig. 14 the encoder input [3:0] = 0101 is introduced and noise is added with the hamming code encoder. The outputof decoder has 2 bit error identification take place as a result.



```
encoder
i_data            0101
o_hamming_code    0101101
o_parity          St0
p1                1
p2                0
p4                1
noise
i_data            00101101
i_noise           01111
o_data            01101100
noise             1000001
error_at_bit      111
decoder
i_data            1101100
i_parity          St0
o_syndrome        0100000
o_data            1001
o_1bit_error      St0
o_2bit_error      St1
o_parity_error    St0
p1                0
p2                1
p4                1
syndrome          0100000
data_decoded      1001100
overall_parity    St0
```

Figure 14. Output for Double bit error

Fig. 15 shows the RTL design of the circuit. Here the output of the encoder is added with noise and given as the decoder input. This decoder shows the types of the error according to the input data.

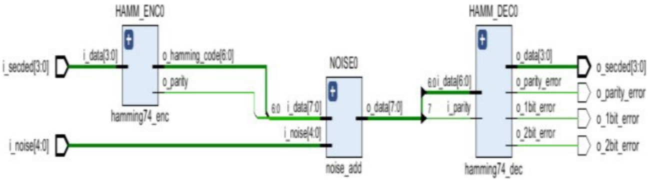


Figure 15. RTL Design

2.4.8 Hardware Implementation

2.4.8.1 Output for single bit error

The encoder input data [3:0]=0101 is modified into seven bits as 0101101 by using the hamming code. Thenadding noise randomly, the LSB of the second bit position is changed and given as input of the decoder then the data isdecoded and get the o_data as 0101.

2.5.8.2 Output for double bit error

The encoder input data [3:0]=0101 is modified into seven bits as 0101101 by using the hamming code. Then adding noise randomly, the LSB of the initial bit locationand the MSB of the initial bit location is changed and given as input of the decoder then the data is decoded and get the o_data as 1001. Here the data is only detected, it is not corrected.

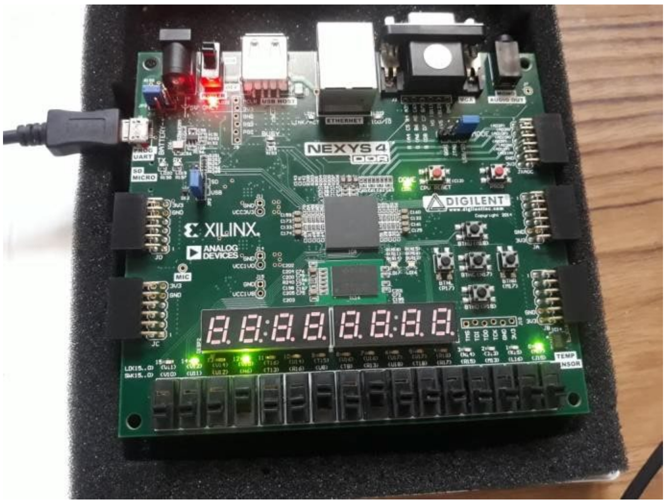


Figure 16. FPGA results for Single bit error

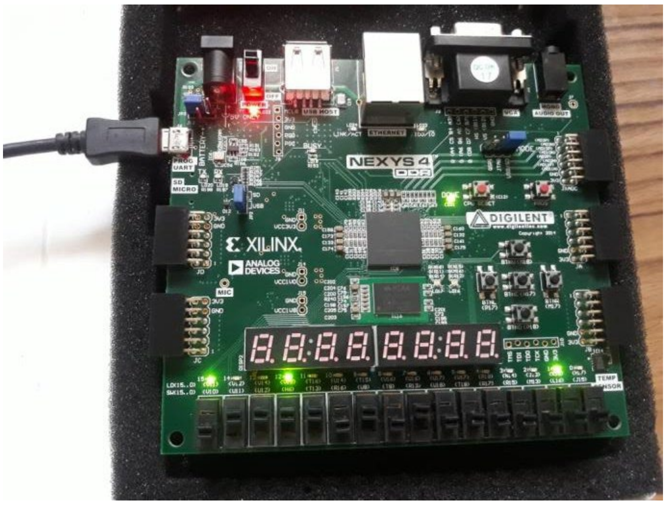


Figure 17. FPGA results for Double bit error

3. Conclusion

As a result of different soft error in SRAM memory, single cell as well as multi-cell upset were produced. The reads data that is stored on a periodic basis and evaluates for correctable and identifiable errors caused by SEUs. The self-refresh function is carried out by providing input as any possible combination of the SRAM output. Error Correction Codes technique (7,4) hamming code encoder is used here. Encoding and Decoding processes with noise and without noise were summarized in Fig. 18. The simulation results implemented by using Verilog Coding in vivado 17.4 tool.

The results were also dumped in FPGA NEXYS 4 DDR andthe output also verified successfully.

INPUT DATA	NOISE	OUTPUT DATA
0101	0000	0101-same as input
0101	00010	0101-one bit of error resolved
0101	01111	1001-Two bit of error is detected

Figure 18. Summary table

3.1 Future Work

SRAM failures that are induced by an outside occurrence and do not affect the SRAM circuitry. They aretransient in the sense that once an alternate value has been assigned to the affected word, the error is gone. Radioactive particles entering a circuit and dynamic voltage noise during a word read or write are two causes of this type of failure. As semiconductor geometries shrink, soft error become increasingly likely. This is why the bits are closer to one another and the voltage levels are lower, resulting in less robust bit cell storage. Soft errors onlyhave an impact on the system if a system reads the troubledword before it is written. Hence the SEC-DED code has been widened into SECDEDDAEC ("Single Error Correction Double Error Detection Double Adjacent Error Correction"), which may be used for single adjustment of errors, double identification of errors, and adjacent correction of errors.

References

[1] M. S. M. Siddiqui, S. Ruchi, L. Van Le, T. Yoo, I. -J. Chang and T. T. -H. Kim, "SRAM Radiation Hardening Through Self-Refresh Operation and Error Correction," in *IEEE Transactions on Device and Materials Reliability*, vol. 20, no. 2, pp. 468-474, June 2020, doi: 10.1109/TDMR.2020.2994769.

[2] L. A. Aranda, O. Ruano, F. Garcia-Herrero and J. A. Maestro, "Reliability Analysis of ASIC Designs With Xilinx SRAM-Based FPGAs," in *IEEE Access*, vol. 9, pp. 140676-140685, 2021, doi: 10.1109/ACCESS.2021.3119633.

[3] J. Gracia-Morán, L. J. Saiz-Adalid, D. Gil-Tomás and P. J. Gil-Vicente, "Improving Error Correction Codes for Multiple-Cell Upsets in Space Applications," in *IEEE Transactions on Very LargeScale Integration (VLSI) Systems*, vol. 26, no. 10, pp. 2132-2142, Oct. 2018, doi: 10.1109/TVLSI.2018.2837220.

[4] E. Ibe, H. Taniguchi, Y. Yahagi, K. -i. Shimbo and T. Toba, "Impactof Scaling on Neutron-Induced Soft Error in SRAMs From a 250nm to a 22 nm Design Rule," in *IEEE Transactions on Electron Devices*, vol. 57, no. 7, pp. 1527-1538, July 2010,

doi: 10.1109/TED.2010.2047907.

[5] K. Jamal and Dr. P. Srihar et al., “Test Vector Generation using Genetic Algorithm for Fault Tolerant Systems” International Journalof Control Theory and Applications (IJCTA), 9(12), pp. 5591-5598, May 2016.

[6] R. C. Baumann, “Radiation-induced soft errors in advanced semiconductor technologies,” IEEE Trans. Device Mater. Rel., vol. 5, no. 3, pp. 305–316, Sep. 2005, doi: 10.1109/TDMR.2005.853449.

[7] K. Rahul and S. Yachareni, "Low Power and Area efficient ForwardError correction code for SRAM in FPGAs," *2020 8th International Electrical Engineering Congress (iEECON)*, Chiang Mai, Thailand, 2020, pp. 1-4, doi: 10.1109/iEECON48109.2020.229511.

[8] S. Manoj and C. Babu, "Improved error detection and correction for memory reliability against multiple cell upsets using DM C & PM C”, IEEE Annual India Conference (INDICON), Bangalore, pp. 1 - 6, December 2016, doi: 10.1109/INDICON.2016.7839094.

[9] Ming Zhu and Liyi Xiao and Shuhao Li et al., (2010) “Efficient Two-Dimensional Error Codes for Multiple Bit Upsets Mitigation in Memory”. In Defect and Fault Tolerance in VLSI Systems (DFT), IEEE 25th International Symposium on, pages 129– 135.

[10] Y. Kawakami, “Investigation of soft error rate including multi-bit upsets in advanced SRAM using neutron irradiation test and 3D mixed-mode device simulation”, Electron Devices Meeting, 2004. IEDM Technical Digest. IEEE International.

[11] Vinotha, P., Jose, D. (2020). VLSI Implementation of Image Encryption Using DNA Cryptography. In: Balaji, S., Rocha, Á., Chung, YN. (eds) Intelligent Communication Technologies and Virtual Mobile Networks. ICICV 2019. Lecture Notes on Data Engineering and Communications Technologies, vol 33. Springer, Cham. https://doi.org/10.1007/978-3-030-28364-3_17

[12] Dhandapani, N., Mohamed Ashik, M.Z., Bhargav, K.R., Achyuth, N., Jose, D. (2023). VLSI Implementation of BCH Encoder with Triple DES Encryption for Baseband Transceiver. In: Marriwala, N., Tripathi, C., Jain, S., Kumar, D. (eds) Mobile Radio Communications and 5G Networks. Lecture Notes in Networks and Systems, vol 588. Springer, Singapore.