FAULT TOLERANT UNMATCHED PARALLEL FILTERS BASED ON EFFICIENT CODING SCHEMES

Yerraguntla Rajeswari[1], U.V. Ratna Kumari [2]
[1]PG Student, Dept. of ECE, University College of Engineering, Kakinada, JNTUK,
[2]Professor, Dept. of ECE, University College of Engineering, Kakinada, JNTUK,

Abstract- Better communication for the implementation of a total number of blocks or modules is optimized for communication and digital processors. Due to the increasing number of blocks in the data communication system, increased complexity and reduced data reliability. In many cases, these modules operate the same way and perform the same signal processing on the data, and they are usually digital filters. To deal with the issue of reliability, implementation needs to be fault tolerant. Therefore, the newly proposed schemes based on error detection and error correction for the protection of parallel filters are consistent with reliability issues. In which every filter is treated as a bit and redundant filters are treated as parity check bits. And the outputs of filters are numbers and not bits, which decreases the protection overhead and makes redundant filter count autonomous of the sum of parallel filters. This method should be applied in Verilog / VHDL with appropriate filters. As an extension of the similar technique will be executed in the Verilog / VHDL in unmatched parallel filters, that is, filters consistent with altered impulse responses and corrected filter results from ECS circuit are stored in fault tolerant memory, where faults are protected by the novel Decimal Matrix Coding structure from multiple cells upsets in memory.

Keyword: ECS scheme, Error detection and Error correction, Verilog/VHDL, DMC scheme.

I. INTRODUCTION

Filters are widely used in digital signal processing and communication systems. In some cases, the reliability of these systems is crucial, and demand for sustainable implementation of the filter. Over the years, there are several methods that have been proposed use the structure and properties of the filter to achieve error tolerance. With scalable technology, it allows for more complex systems, including multiple filters. In these complex systems, it is usually found that some of the filters run in parallel, e.g., by applying different inputs to the same filter or same input to different filters.

Currently, a simplified method is proposed using a consistent filter presence for error determination. The general idea is to show that a consistent filter can be protected by the use of a debug (ECC) code [1], in which each filter is an equivalent bit in the traditional ECC. In this case, an equivalent number of checks are calculated and stored in memory so that errors can be identified and corrected when reading the data. Later, the proposed effective coding scheme (ECS) [2] allows for more effective protection when the number of parallel filters is large. In particular, it was shown that the resources and outputs of the filter are non-binary and this allows for more effective protection. This reduces the invoice amount of protection and redundant filters become independent of the number of concurrent filters. Finally, for filters with regular features and properties, this scheme can be used for detecting and debugging at a lower cost than TMR.

Due to the reduced CMOS to nano scale, and memory combined with increased flow of electronic systems, soft error rates in memory cells are growing rapidly, especially if available memory is in space operation; due to atmospheric ionization effects, alpha particles, and cosmic rays [3]. Although a single disorder is a major concern of memory reliability, some comorbidities (MCU) have become a serious problem of reliability in memory programs [4]. The remainder of this short, first, existing memory protection is outlined on the basis of the DMC[5]. The proposed coding scheme is then presented, illustrating the actual research.

II. LITERATURE SURVEY

Improved decoding algorithm for high reliable reed muller coding

C. Argyrides& D.K. Pradhan Sep 2007

As the CMOS technology scaling to nm, low cost, high density, high speed integrated circuits with low supply voltage has increased the probability of fault occurrence in the memories. This leads to significant reliability issues, increasing failure rates, especially SRAM. Some methods to reduce debugs are triple modular redundancy and error correction codes (ECC). Soft errors are an important issue in the reliability of memory. Soft bugs won't damage the device; they just corrupt the data that are processed. On detection of soft errors, they can be corrected by writing corrected data in place of erroneous data. Highly reliable systems use a method of error correction, but in many systems it is difficult to correct data or even impossible to detect errors. To prevent bug fixes in the stored data, using such error code-based debugging, such as block coding, the processes of encoding and decoding are important for impacting time needed and complexity involved in memory access. Multi cellular soft errors have become a matter of reliability concerns in some cases, than a single cell error. The BCH code, Reed Solomon code are used for MCU management, but the power, area and delay cost of these codes are very high through high-complexity architectural encoding and decoding. Decimal codes use the technique of encoder reusing thus reducing cost and complexity. DMC improves memory reliability, which improves error-correction capabilities.

Parallel double error correcting code design to mitigate multi-bit upsets in srams

R.Naseer and J. Draper, Sep 2008

During transmission through the communication network, information can be corrupted by physical/logical faults which will cause the entire system to fail. Therefore, the communication system must also contain device to test and correct errors to provide a secure and reasonable communication. So far, a number of error detection and error correction codes have been developed for different purposes. A few of them are Burger codes and checksums, Cyclic

Redundancy codes, Hamming codes, Residue codes, Nordstrom-Robinson codes and Turbo codes for error correction, and BCH codes and modified Residue codes for multi error correction. These codes may work well in some cases, but not in all conditions and environments. However, due to the constant increase in size, speed and complexity of the data transmission, the overall efficiency is reduced. Therefore, it is important to feel the need for new and modified methods.

Usually, some of the faults are associated with magnetic fields, electrical and weather effects, such as thunderstorms, sunlight, etc. They can impact internal communications (e.g., communication between internal nodes in computer) and external communications (e.g., wireless ,digital and satellite communications). Conventional memories employ Single-Error Correcting and Double-Error Detecting (SEC-DED) methods. But in telecommunications with large data packets in the system, there is a need for a methodical approach to correct multiple errors along with multiple errors detection. Therefore, the task of each receiving system would be checking for errors, and then try to correct the issue, otherwise request retransmission or use other means. In this regard, many designers have developed methods that are only good in certain circumstances and environments.

III. SCHEMATIC OF DMC FAULT-TOLERANT MEMORY

The schematic of fault-tolerant memory is showed in Fig.3.1 First, throughout the encoding process, information bits D are fed to the DMC encoder, and then the horizontal redundant bits H then vertical redundant bits V are found from the DMC encoder. After encoding process is finished; the generated DMC codeword is kept in the memory. If MCUs happen in the memory; these faults can be rectified in the decoding (read) method. Due to the benefit of decimal process, the projected DMC has complex fault-tolerant capability with lower performance expenses. In the fault-tolerant memory, the ERT method is proposed to decrease the area overhead of further circuits and will be presented in the subsequent sections.

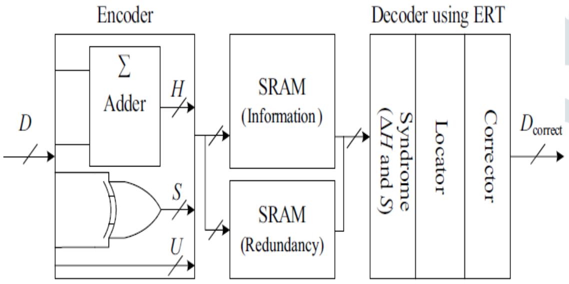


Fig: 3.1 schematic of 32 bit with DMC.

32- Bit DMC Encoder:

In the DMC system, first, the divide-symbol then arrange-matrix concepts are done, i.e., the N-bit word is separated into k symbols of m bits (N = k × m), and these symbols are agreed in a $k_1\times k_2$ 2-D matrix ($k = k_1\times k_2$, wherever the values of k_1 and k_2 signify the numbers of rows and columns in the logical matrix correspondingly). Another, the horizontal redundant bits H are formed by executing decimal integer adding of particular symbols per row. Here, every symbol is observed as a decimal integer. Thirdly, the vertical V bits are widely accepted among the binary operation bits per column. It should be noted that the distribution of symbols and the arrangement of the matrix are performed in a logical rather than physical way. Therefore, the proposed DMC does not require unchanged changes to the physical memory structure.

To explain the DMC project, we have assumed a 32-bit word, as shown in fig.3.2.The cells since D_0 to D_{31} are information bits. This 32-bit word has been separated into eight symbols of 4-bit. k_1= 2 and k_2= 4 have been selected concurrently. H_0–H_{19} are horizontal check bits; V_0 over V_{15} are vertical check bits. Although it should be declared that the maximum capacity, correction (i.e., maximum size of MCUs can be corrected) and the number of redundant bits are different when different values for K and m are selected. Therefore, K and m should be adjusted metrics to increase the ability to carefully adjust and reduce the number of redundant bits.

For example, in this case when k = 2 × 2 and m = 8, only one bit error can be slightly corrected and this number of redundant bits is when k = 80 × 4 and 4 m = 2, 3. Slightly the error can be corrected and the number of redundant bits is reduced to 32, but when k = 2 × 4 and m = 4, the maximum correction capacity is up to 5 bits and this number of redundant bits is 72 In this paper, a In order to improve the reliability of memory, the error correction capability was first considered, so k = 2 × 4 and 4 m = to construct the DMC.

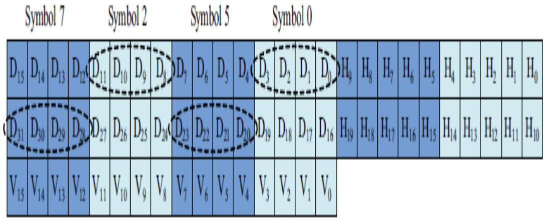

Symbol 7
Symbol 2
Symbol 5
Symbol 0
D_{15} D_{14} D_{13} D_{12} D_{11} D_{10} D_9 D_8 D_7 D_6 D_5 D_4 D_3 D_2 D_1 D_0 H_9 H_8 H_7 H_6 H_5 H_4 H_3 H_2 H_1 H_0
D_{31} D_{30} D_{29} D_{28} D_{27} D_{26} D_{25} D_{24} D_{23} D_{22} D_{21} D_{20} D_{19} D_{18} D_{17} D_{16} H_{19} H_{18} H_{17} H_{16} H_{15} H_{14} H_{13} H_{12} H_{11} H_{10}
V_{15} V_{14} V_{13} V_{12} V_{11} V_{10} V_9 V_8 V_7 V_6 V_5 V_4 V_3 V_2 V_1 V_0

Fig: 3.2 32 bit word can be divided as 8 symbols with k=2x4 and m=4.

Horizontal redundant bits are considered as follows.

$H_9\ H_8\ H_7H_6\ H_5 = D_7\ D_6\ D_5\ D_4 + D_{15}\ D_{14}\ D_{13}\ D_{12}$ (1)

$H_4H_3H_2\ H_1\ H_0 = D_3\ D_2\ D_1\ D_0 + D_{11}\ D_{10}\ D_9D_8$ (2)

Likewise for the horizontal redundant bits $H_{14}H_{13}H_{12}\ H_{11}\ H_{10}$ and $H_{19}\ H_{18}H_{17}H_{16}\ H_{15}$, where "+" signifies decimal integer addition.

Vertical redundant bits are considered as follows

$V_0 = D_0 \oplus D_{16}$ (3)

$V_1 = D_1 \oplus D_{17}$ (4)

Likewise for the rest vertical redundant bits. The encoding can be achieved by decimal and binary addition processes from (1) to (4). The encoder that calculates the redundant bits by multi bit adders then XOR gates is exposed in Fig.3.3. In this figure, $H_{19}-\ H_0$ are horizontal redundant bits, $V_{15}-\ V_0$ are vertical redundant bits, and the outstanding bits $U_{31}-\ U_0$ are the evidence bits which are straight derivative from D_{31} to D_0. The empower signal En will be clarified in the following section.

The next fig.3.3 shows the 32 Bit DMC Encoder which involves of adder circuits and EX-OR gates to compute horizontal and vertical redundant bits.

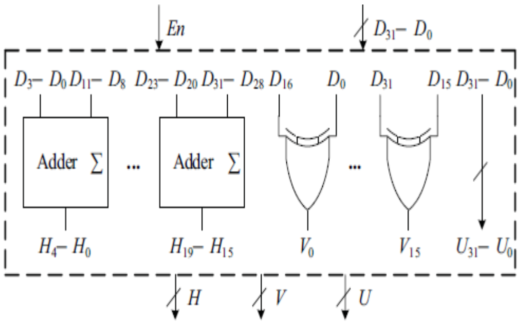


Fig: 3.3 32 -bit DMC encoder structure using multi bit adders and XOR gates

32 Bit DMC Decoder:

To find a word being modified, the decoding method is required. For example, first, the new redundant bits $H_4H_3H_2H_1H_0$' and V_0'-V_3' are produced by the conventional retrieved information bits D'. Later, the horizontal syndrome bits $\Delta H_4H_3H_2H_1H_0$ and the vertical syndrome bits $S_3 - S_0$ can be designed as follows.

$\Delta H_4H_3H_2H_1H_0 = H_4H_3H_2H_1H_0' - H_4H_3H_2H_1H_0$ (5)

$S_0 = V_0' \wedge V_0$ (6)

Also for the rest vertical syndrome bits, where "–" signifies decimal integer subtraction. While $\Delta H_4H_3H_2H_1H_0$ and $S_3 - S_0$ are equal to zero, the stored codeword has unique sign bits in symbol 0 where no faults occur. When $\Delta H_4H_3H_2H_1H_0$ and $S_3 - S_0$ are nonzero, the convinced errors (the number of errors is 4 in this case) are perceived and located in symbol 0, and then these faults can be modified by

$D_{0correct} = D_0 \wedge S_0$ (7)

The subsequent figure 3.4 expressions the 32-bit DMC decoder which is made up of the next sub modules, and each performs an exact task in the decoding procedure: syndrome calculator, error locator, and error corrector. It can be perceived from this fig. that the redundant bits necessarily be recomputed from the obtained information bits D' and compared to stored redundant bits in order to get the syndrome bits ΔH and S. Then error locator usages ΔH and S to perceive and locate which bits certain faults occur in. Lastly, in the fault corrector, these errors can be modified by reversing the values of error bits.

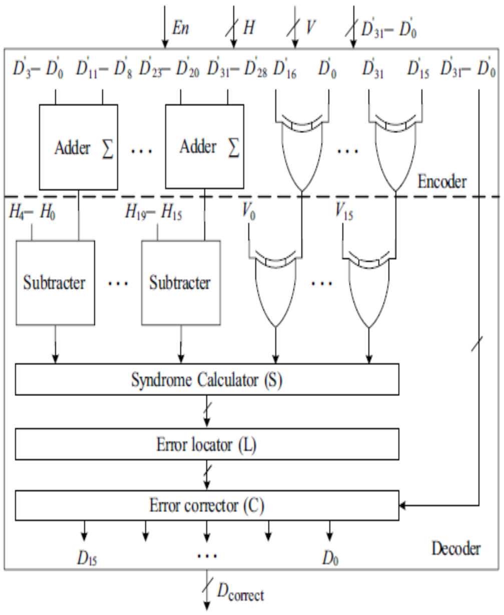
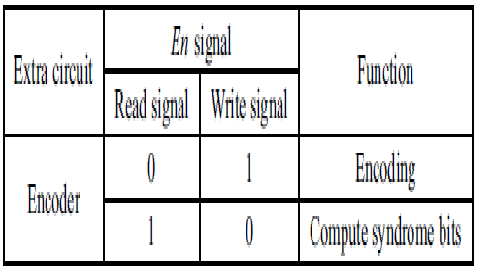


Fig: 3.4 32 Bit DMC decoder



Extra circuit	En signal		Function
	Read signal	Write signal	
Encoder	0	1	Encoding
	1	0	Compute syndrome bits

Fig: 3.5 Encoder Reuse Technique

Encoder Reuse Technique:

In this system, the circuit area of DMC is reduced by reprocessing its encoder. This is named the ERT. The ERT can decrease the area of DMC without troubling the entire encoding and decoding methods.

From Fig, 3.5 it can be perceived that the DMC encoder is reused for finding the syndrome bits in DMC decoder. So, the entire circuit area of DMC can be decreased as an outcome of using the existing circuits of encoder. Also, figure 3.5 shows the projected decoder with an enable signal En to decide whether the encoder desires to be a portion of the decoder. In other words, the En signal is used for dividing the encoder from the decoder, and it is under the control of the write and read signals in memory. So, in the encoding (write) method, the DMC encoder acts only as an encoder to perform the encoding actions. But, in the decoding (read) method, this encoder works for calculating the syndrome bits in the decoder. These clearly display how the area above of extra circuits can be significantly reduced.

Area, delay, power analysis of DMC:

The succeeding table 3.1 represents area, delay, power analysis of DMC associated to further codes.

ECC CODES	AREA		POWER		DELAY	
	µm²	%	mw	%	ns	%
DMC	41572.6	100	10.8	100	4.9	100
PDS	486778.1	1170.9	2211.1	2047.2	18.7	381.6
MC	77933.7	187.5	24.7	228.7	7.1	144.9
Hamming	58409.4	140.5	20.5	189.8	6.7	136.7

Table 3.1: Area, delay and power analysis of DMC

For 32-bit DMC 16 errors can be modified as exposed in below tabular form associated to residual codes.

ECC CODES	The Number of Errors in a word															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DMC	100	100	100	100	10	92	84	76	66	60		47	40	31	22	11
PDS	100	100	100	08	0	0	0	0	0	0	0	0	0	0	0	0
MC	100	100	76	54	35	14.2	6.7	0.6	0	0	0	0	0	0	0	0
Hamming	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3.2: In 32-bit DMC 16 errors can be corrected

V. PROPOSED SCHEME

In this section, proposed scheme is presented through a case study in Fig, 3.5. Proposed system reflects storing of corrected filter outputs from ECS scheme in signal processing applications in a fault tolerant memory protected with DMC. For a digital filter if N no.of non-zero outputs are present at its output, each of size m-bit ; then to store n x m –bits we consider DMC protected fault tolerant memory of any available size and later multiple such memories are employed to store all n x m –bits.

As explained in section III , error correction capability and number of redundant bits produced vary in DMC protected memory scheme based on no. of symbols and no. of bits per symbols considered.

To exemplify the use of the future scheme, one case study that considers storing of modified filter outputs from a set of 4 parallel filters which are protected by ECS scheme is accessible here.

Here, a set of 4 parallel unmatched filters i.e. digital filters with different impulse responses, which are given identical input x are protected by ECS scheme. In ECS system, for single error correction no.of redundant filters essential is always 2 independent of no. of novel modules. Here, for assume each filter has 8 non zero outputs each of 8-bits is considered in below case study. y1, y2, y3, y4 are corrected filter outputs from ECS single fault correction module. So y1, y2, y3, y4 each provides 64-bit data. Here 2 32-bit DMC protected memories are used to store this 64 bit data. 1st 32 bits are sent to one memory and next 32 bits are stored in next memory which are protected using above described DMC scheme.

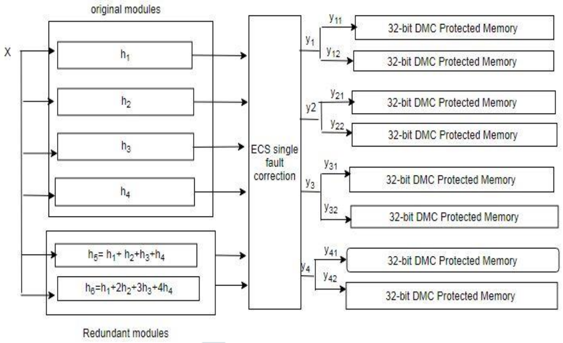
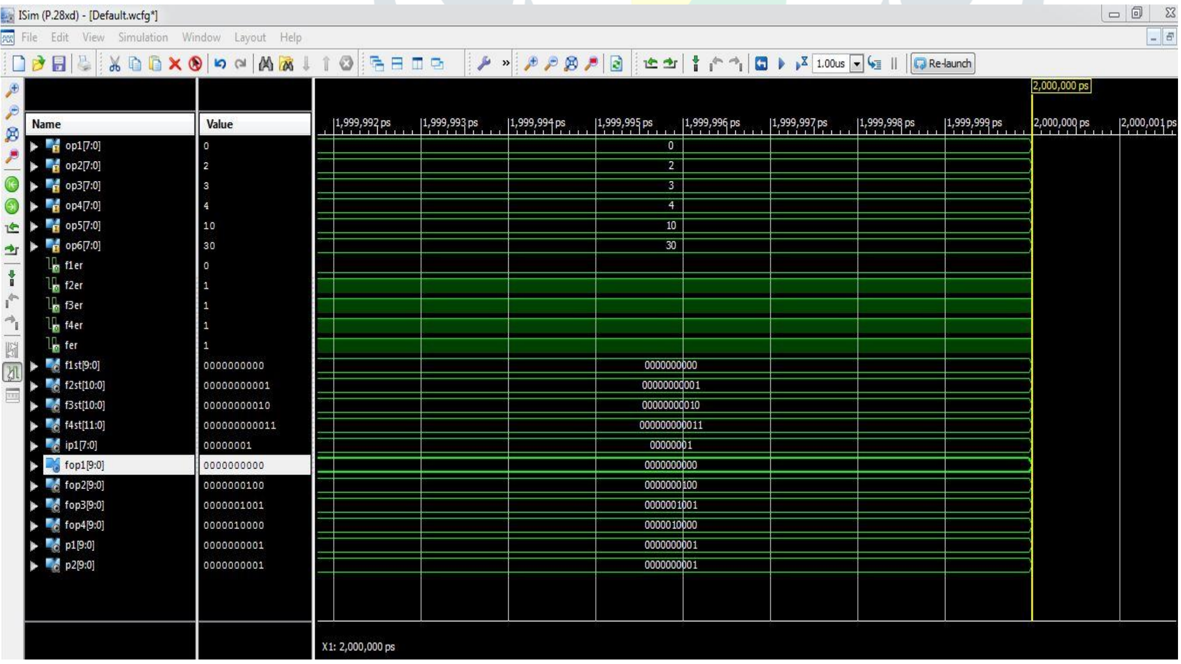


Fig. 3.5 A case study for Proposed scheme

Likewise, this proposed scheme also helps in storing any number of filter outputs in any fault tolerant memory of available size.

IV. EXPERIMENTAL RESULT

:-

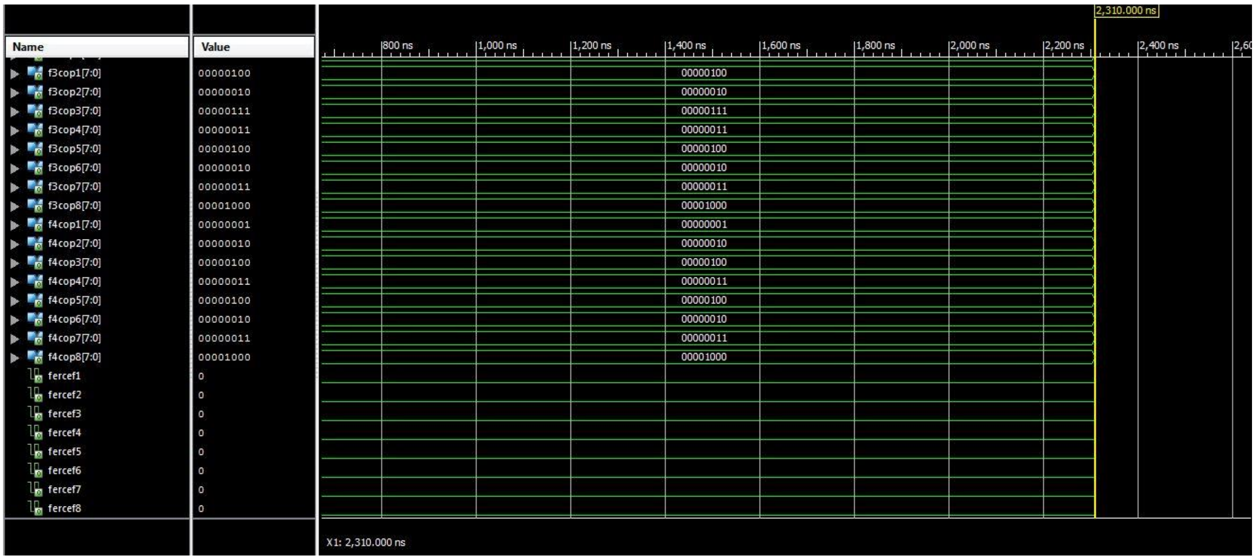


Name	Value
op1[7:0]	0
op2[7:0]	2
op3[7:0]	3
op4[7:0]	4
op5[7:0]	10
op6[7:0]	30
f1er	0
f2er	1
f3er	1
f4er	1
fer	1
f1st[9:0]	0000000000
f2st[10:0]	00000000001
f3st[10:0]	00000000010
f4st[11:0]	000000000011
ip1[7:0]	00000001
fop1[9:0]	0000000000
fop2[9:0]	0000000100
fop3[9:0]	0000001001
fop4[9:0]	0000010000
p1[9:0]	0000000001
p2[9:0]	0000000001

X1: 2,000,000 ps

Fig 4.1 Simulation results for error detection module in ECS scheme

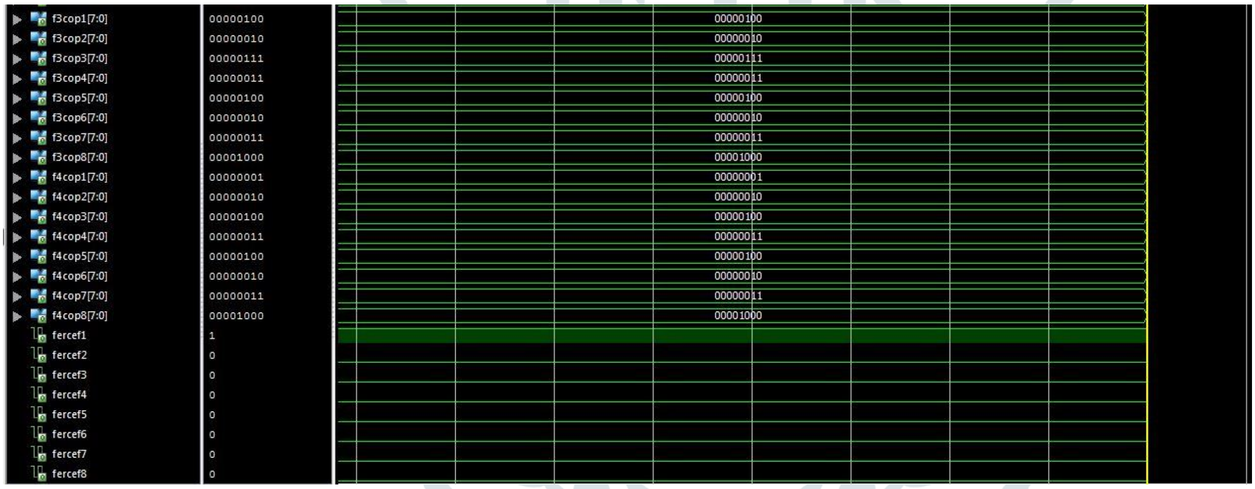
Observation: When filter 1 output is made intentionally to zero i.e. introducing error then corresponding error detection variable fop1 nothing but parity1 become all zeroes so indicating error is in filter1



Name	Value
f3cop1[7:0]	00000100
f3cop2[7:0]	00000010
f3cop3[7:0]	00000111
f3cop4[7:0]	00000011
f3cop5[7:0]	00000100
f3cop6[7:0]	00000010
f3cop7[7:0]	00000011
f3cop8[7:0]	00001000
f4cop1[7:0]	00000001
f4cop2[7:0]	00000010
f4cop3[7:0]	00000100
f4cop4[7:0]	00000011
f4cop5[7:0]	00000100
f4cop6[7:0]	00000010
f4cop7[7:0]	00000011
f4cop8[7:0]	00001000
fercef1	0
fercef2	0
fercef3	0
fercef4	0
fercef5	0
fercef6	0
fercef7	0
fercef8	0

Fig 4.2 Simulation results for overall ECS scheme for 4 parallel filters without introducing error

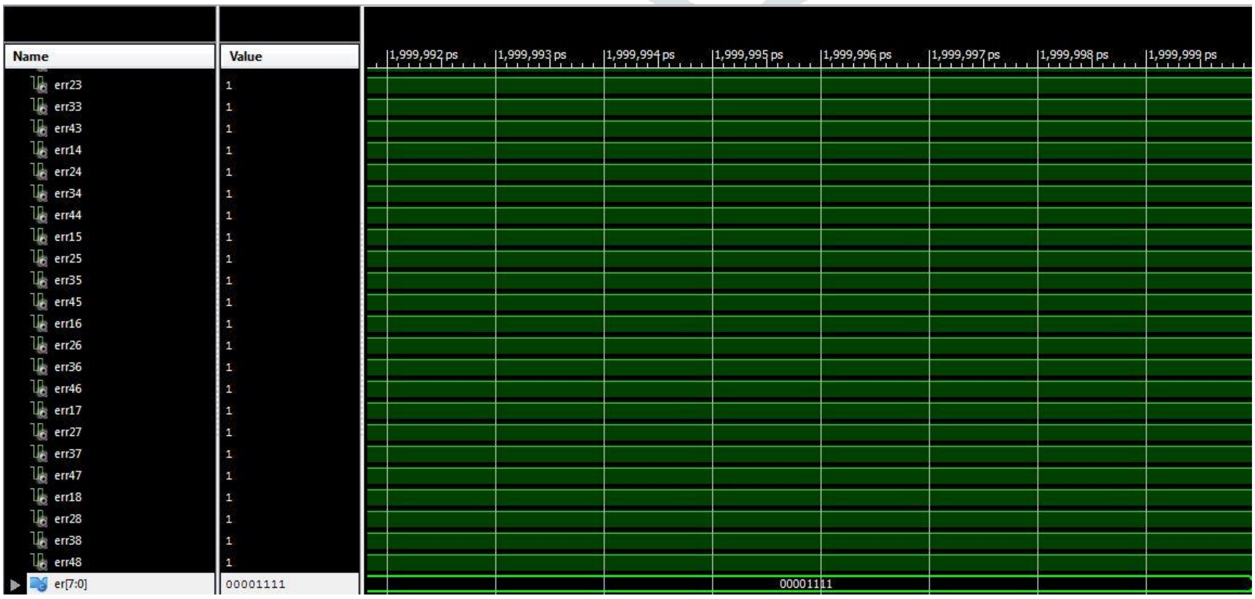
Observation: When none of the filter outputs are made intentionally to some other value i.e. introducing no error then corresponding error detection variable fercef1 to fercef8 nothing but parity1 become all zeroes so indicating no error is present in none of the filters' samples from 1 to 8.



Name	Value
f3cop1[7:0]	00000100
f3cop2[7:0]	00000010
f3cop3[7:0]	00000111
f3cop4[7:0]	00000011
f3cop5[7:0]	00000100
f3cop6[7:0]	00000010
f3cop7[7:0]	00000011
f3cop8[7:0]	00001000
f4cop1[7:0]	00000001
f4cop2[7:0]	00000010
f4cop3[7:0]	00000100
f4cop4[7:0]	00000011
f4cop5[7:0]	00000100
f4cop6[7:0]	00000010
f4cop7[7:0]	00000011
f4cop8[7:0]	00001000
fercef1	1
fercef2	0
fercef3	0
fercef4	0
fercef5	0
fercef6	0
fercef7	0
fercef8	0

Fig 4.3 (a)

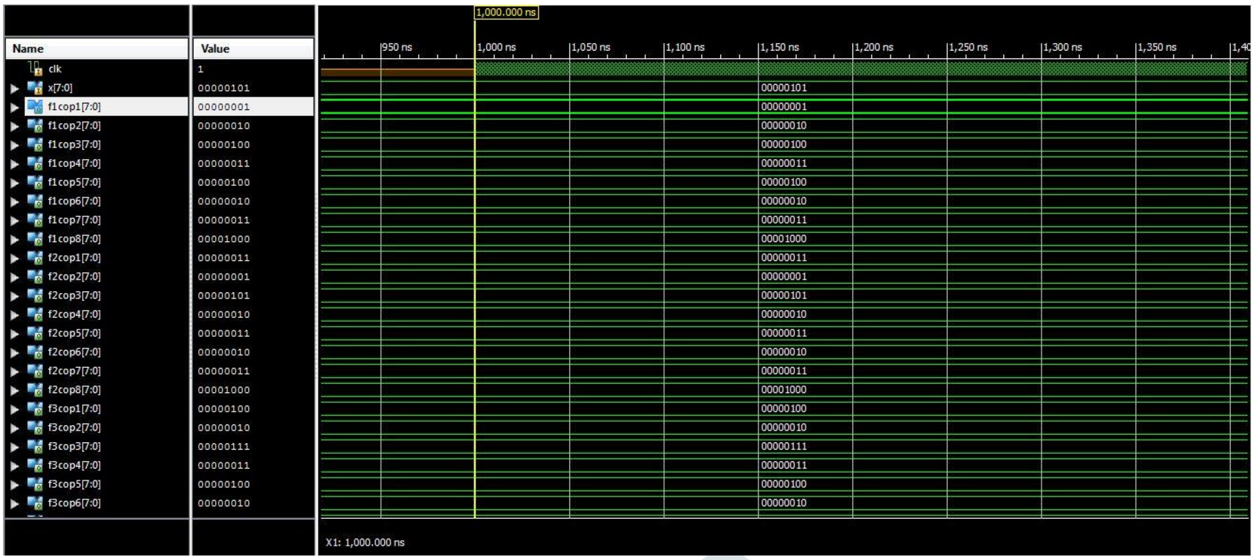
Observation: When 1st output sample of filter 1 is made error i.e. introducing error then corresponding error detection variable fercef1 become high and remaining are all zeroes so indicating error is in 1st sample of one of the filters.



Name	Value
err23	1
err33	1
err43	1
err14	1
err24	1
err34	1
err44	1
err15	1
err25	1
err35	1
err45	1
err16	1
err26	1
err36	1
err46	1
err17	1
err27	1
err37	1
err47	1
err18	1
err28	1
err38	1
err48	1
er[7:0]	00001111

Fig 4.3 (b)

Observation: er[7:0] is the error value introduced in the 1st output sample of filter 1.

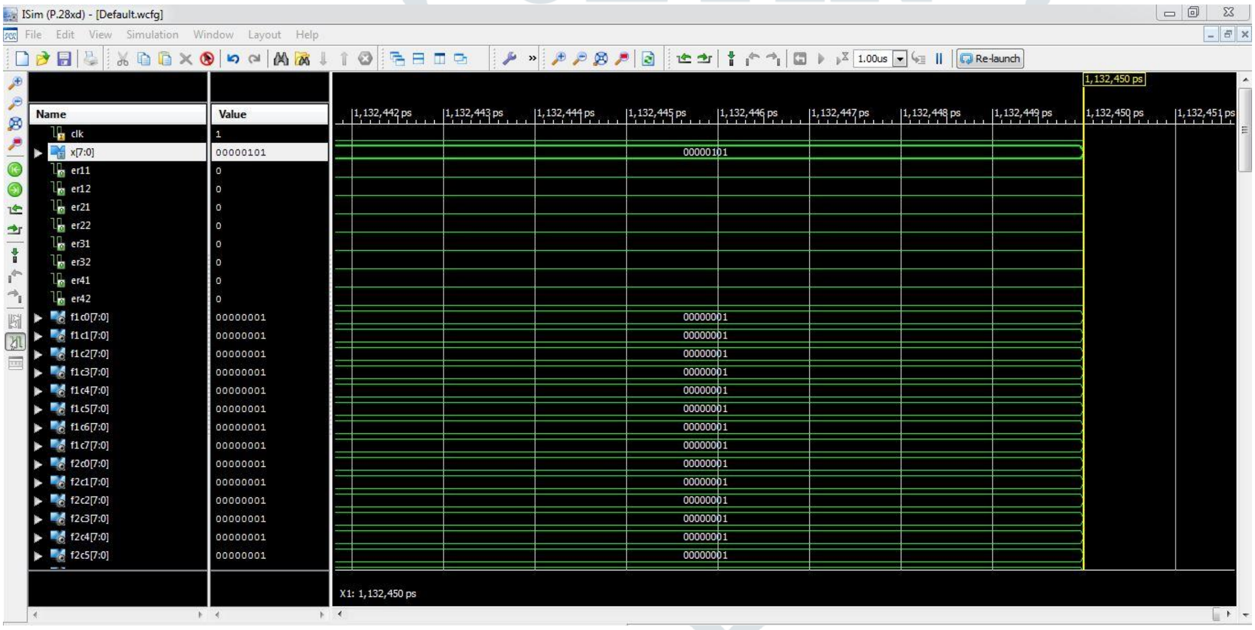


Name	Value
clk	1
x[7:0]	00000101
f1cop1[7:0]	00000001
f1cop2[7:0]	00000010
f1cop3[7:0]	00000100
f1cop4[7:0]	00000011
f1cop5[7:0]	00000100
f1cop6[7:0]	00000010
f1cop7[7:0]	00000011
f1cop8[7:0]	00001000
f2cop1[7:0]	00000011
f2cop2[7:0]	00000001
f2cop3[7:0]	00000101
f2cop4[7:0]	00000010
f2cop5[7:0]	00000011
f2cop6[7:0]	00000010
f2cop7[7:0]	00000011
f2cop8[7:0]	00001000
f3cop1[7:0]	00000100
f3cop2[7:0]	00000010
f3cop3[7:0]	00000111
f3cop4[7:0]	00000011
f3cop5[7:0]	00000100
f3cop6[7:0]	00000010

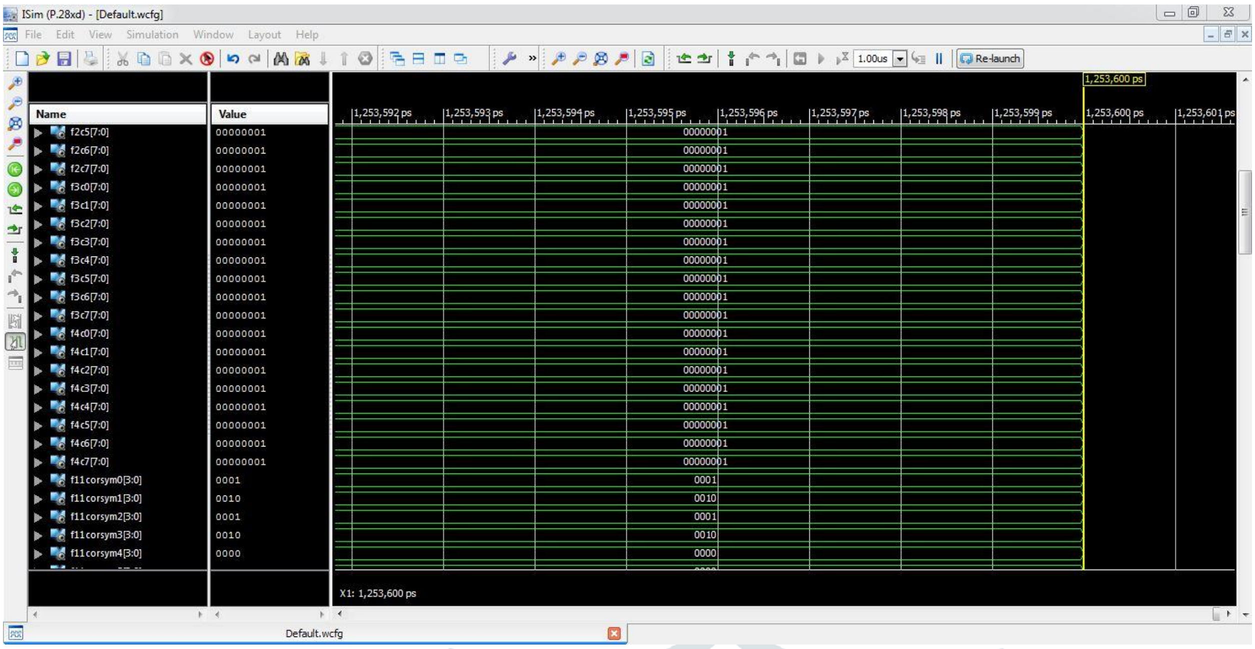
Fig 4.3 (c)

Fig 4. Simulation results for overall ECS scheme with introducing error in f11 i.e. 1st o/p sample of 1st filter

Observation: After error detection and correction,f1cop1 [7:0] is the corrected 1st output sample of filter 1.

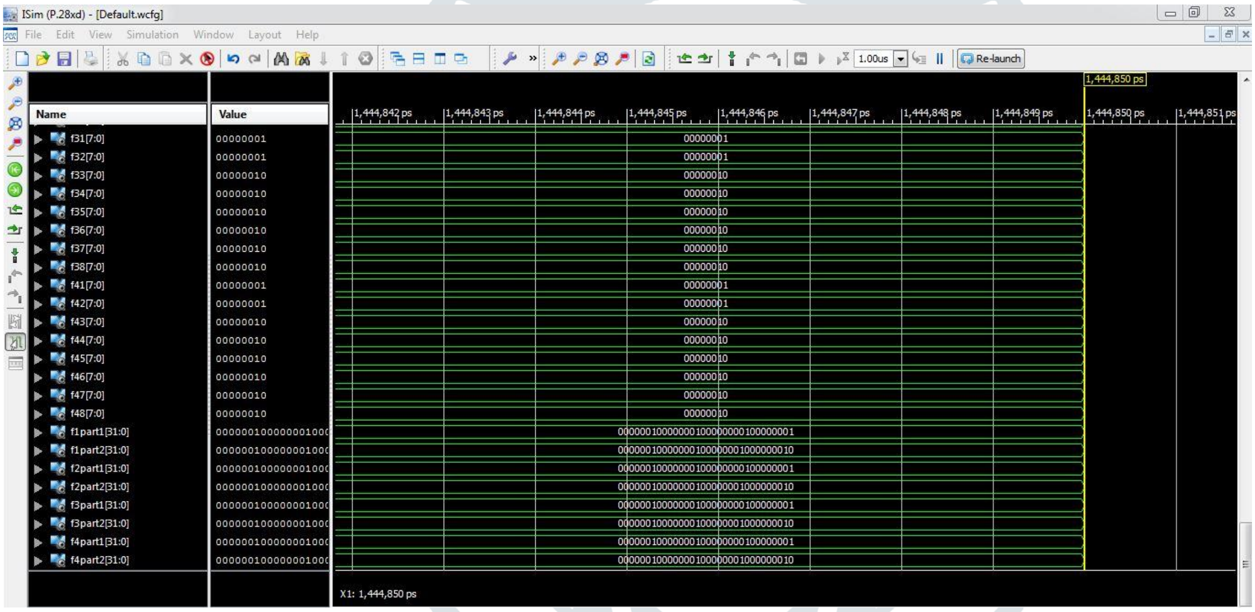


Name	Value
clk	1
x[7:0]	00000101
er11	0
er12	0
er21	0
er22	0
er31	0
er32	0
er41	0
er42	0
f1c0[7:0]	00000001
f1c1[7:0]	00000001
f1c2[7:0]	00000001
f1c3[7:0]	00000001
f1c4[7:0]	00000001
f1c5[7:0]	00000001
f1c6[7:0]	00000001
f1c7[7:0]	00000001
f2c0[7:0]	00000001
f2c1[7:0]	00000001
f2c2[7:0]	00000001
f2c3[7:0]	00000001
f2c4[7:0]	00000001
f2c5[7:0]	00000001



```
Name                Value
f2c5[7:0]           00000001     00000001
f2c6[7:0]           00000001     00000001
f2c7[7:0]           00000001     00000001
f3c0[7:0]           00000001     00000001
f3c1[7:0]           00000001     00000001
f3c2[7:0]           00000001     00000001
f3c3[7:0]           00000001     00000001
f3c4[7:0]           00000001     00000001
f3c5[7:0]           00000001     00000001
f3c6[7:0]           00000001     00000001
f3c7[7:0]           00000001     00000001
f4c0[7:0]           00000001     00000001
f4c1[7:0]           00000001     00000001
f4c2[7:0]           00000001     00000001
f4c3[7:0]           00000001     00000001
f4c4[7:0]           00000001     00000001
f4c5[7:0]           00000001     00000001
f4c6[7:0]           00000001     00000001
f4c7[7:0]           00000001     00000001
f11corsym0[3:0]     0001         0001
f11corsym1[3:0]     0010         0010
f11corsym2[3:0]     0001         0001
f11corsym3[3:0]     0010         0010
f11corsym4[3:0]     0000         0000

X1: 1,253,600 ps
```

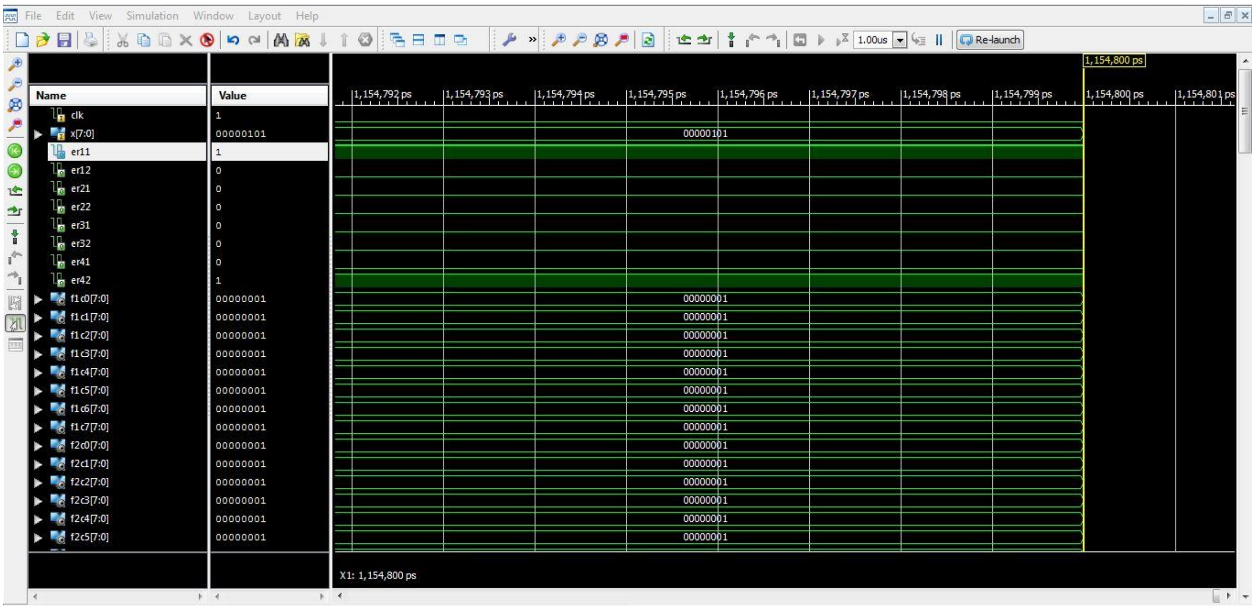


```
Name                Value
f31[7:0]            00000001     00000001
f32[7:0]            00000001     00000001
f33[7:0]            00000010     00000010
f34[7:0]            00000010     00000010
f35[7:0]            00000010     00000010
f36[7:0]            00000010     00000010
f37[7:0]            00000010     00000010
f38[7:0]            00000010     00000010
f41[7:0]            00000001     00000001
f42[7:0]            00000001     00000001
f43[7:0]            00000010     00000010
f44[7:0]            00000010     00000010
f45[7:0]            00000010     00000010
f46[7:0]            00000010     00000010
f47[7:0]            00000010     00000010
f48[7:0]            00000010     00000010
f1part1[31:0]       0000001000000001000    00000010000000100000000100000001
f1part2[31:0]       0000001000000001000    00000010000000100000001000000010
f2part1[31:0]       0000001000000001000    00000010000000100000000100000001
f2part2[31:0]       0000001000000001000    00000010000000100000001000000010
f3part1[31:0]       0000001000000001000    00000010000000100000000100000001
f3part2[31:0]       0000001000000001000    00000010000000100000001000000010
f4part1[31:0]       0000001000000001000    00000010000000100000000100000001
f4part2[31:0]       0000001000000001000    00000010000000100000001000000010

X1: 1,444,850 ps
```

Fig 4.4 Simulation results for overall proposed scheme with no error in any fault tolerant memory

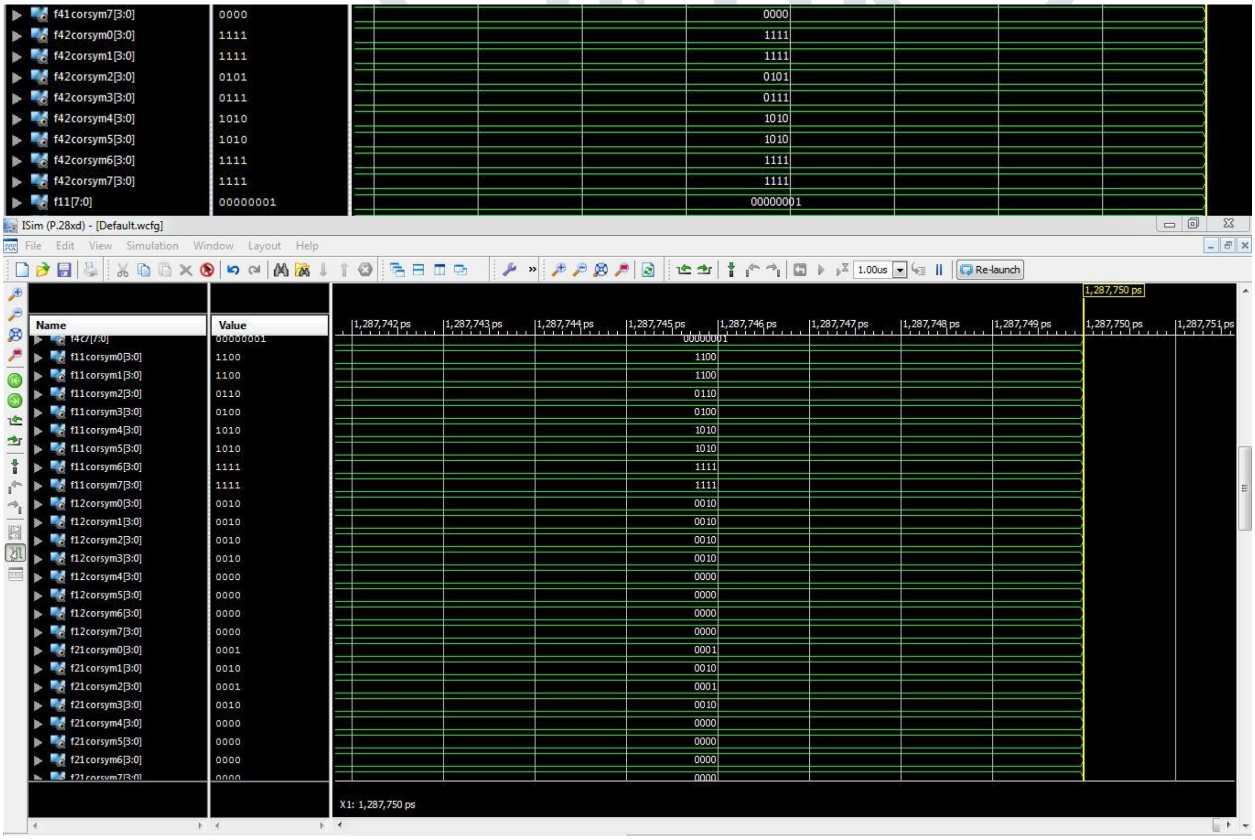
Observation: When none of the filter outputs are made intentionally to some other value i.e. introducing no error in memory then corresponding error detection variableser11 to er42 become all zeroes so indicating no error is present in memory.



Name	Value
clk	1
x[7:0]	00000101
er11	1
er12	0
er21	0
er22	0
er31	0
er32	0
er41	0
er42	1
f1c0[7:0]	00000001
f1c1[7:0]	00000001
f1c2[7:0]	00000001
f1c3[7:0]	00000001
f1c4[7:0]	00000001
f1c5[7:0]	00000001
f1c6[7:0]	00000001
f1c7[7:0]	00000001
f2c0[7:0]	00000001
f2c1[7:0]	00000001
f2c2[7:0]	00000001
f2c3[7:0]	00000001
f2c4[7:0]	00000001
f2c5[7:0]	00000001

Fig 4.5. Simulation results for given case study of overall proposed scheme with error in 1st memory and 8th memory

Observation: When stored data in memory1 and memory 8 made intentionally to some other values i.e. introducing error in memories then corresponding error detection variables er11 and er42 become high so indicating error is present in 1st memory and 8th memory.



Name	Value
f41corsym7[3:0]	0000
f42corsym0[3:0]	1111
f42corsym1[3:0]	1111
f42corsym2[3:0]	0101
f42corsym3[3:0]	0111
f42corsym4[3:0]	1010
f42corsym5[3:0]	1010
f42corsym6[3:0]	1111
f42corsym7[3:0]	1111
f11[7:0]	00000001

Name	Value
f4c7[7:0]	00000001
f11corsym0[3:0]	1100
f11corsym1[3:0]	1100
f11corsym2[3:0]	0110
f11corsym3[3:0]	0100
f11corsym4[3:0]	1010
f11corsym5[3:0]	1010
f11corsym6[3:0]	1111
f11corsym7[3:0]	1111
f12corsym0[3:0]	0010
f12corsym1[3:0]	0010
f12corsym2[3:0]	0010
f12corsym3[3:0]	0010
f12corsym4[3:0]	0000
f12corsym5[3:0]	0000
f12corsym6[3:0]	0000
f12corsym7[3:0]	0000
f21corsym0[3:0]	0001
f21corsym1[3:0]	0010
f21corsym2[3:0]	0001
f21corsym3[3:0]	0010
f21corsym4[3:0]	0000
f21corsym5[3:0]	0000
f21corsym6[3:0]	0000
f21corsym7[3:0]	0000

Fig 4.6 Simulation result showing corrected outputs of memory8 for case study given of overall proposed scheme with error in memory1 and memory 8

V. CONCLUSION & FUTURE SCOPE

In this work, fault tolerant memory which is protected by Decimal Matrix Coding is used for storing filter outputs coming from parallel fault tolerant filters. We stored filter outputs and as well as horizontal then vertical parity bits designed for filter outputs. When filter outputs are retrieved over horizontal and vertical parity bits are calculated for obtained filter outputs from memory and these are compared with stored horizontal and vertical parity bits.

Thus if any disparity is resulted then it is mere indication of occurrence of at least one error. By error detection and location error is situated and it will be corrected.

However, it should be stated that the maximum correction ability (i.e., the maximum size of MCUs can be corrected) and the number of redundant bits are diverse when the different values for *k* and *m* are selected. Therefore, *k* and *m* should be wisely adjusted to exploit maximum correction capability and minimize the number of redundant bits. For instance, in this case, when $k = 2\times2$ and $m = 8$, only 1-bit error can be corrected and the number of redundant bits is 40. Once $k = 4 \times 4$ and $m =$ 2,3-bit errors can be corrected and the number of redundant bits is condensed to 32. However, when $k = 2 \times 4$ and $m = 4$, the maximum correction capability is up to 5 bits and the number of redundant bits is 36.

Similarly, this proposed system also helps in storing any number of any digital filter outputs in any fault tolerant memory of available size.

The only disadvantage of the DMC scheme measured in this work is that extra redundant bits are essential to maintain advanced reliability of memory, so that a practical combination of *k* and *m* should be chosen to maximize memory reliability and minimize the number of redundant bits based on radiation experiments in actual implementation. Therefore, future work will be conducted for the reduction of the redundant bits and the care of the reliability of the proposed technique.

VI. REFERENCES

[1] Z. Gao *et al.*, "Fault tolerant parallel filters based on error correction codes," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 23, no. 2,pp. 384–387, Feb. 2015.

[2] Zhen Gao, Pedro Reviriego, Zhan Xu, Xin Su, Jing Wang, and Juan Antonio Maestro,"Efficient Coding Schemes for Fault-Tolerant Parallel Filters",
IEEE transactions on circuits and systems - II: express briefs, vol. 62, no. 7, july 2015

[3] D. Radaelli, H. Puchner, S. Wong, and S. Daniel, "Investigation of multi-bit upsets in a 150 nm technology SRAM device," IEEE Trans. Nucl. Sci., vol. 52, no. 6, pp. 2433–2437, Dec. 2005
.

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

[5] Jing Guo, Liyi Xiao, *Member, IEEE*, Zhigang Mao, *Member, IEEE*, and Qiang Zhao, "Enhanced Memory Reliability Against Multiple Cell Upsets Using Decimal Matrix Code " , *IEEE Trans.on very large scale integration(VLSI) systems*

[6] C. L. Chen and M. Y. Hsiao, "Error-correcting codes for semi conductor memory applications: A state-of-the-art review," *IBM J. Res. Develop.*,vol. 28, no. 2, pp. 124–134, Mar. 1984.

[7] S. Pontarelli, G. C. Cardarilli, M. Re, and A. Salsano, "Totally fault tolerant RNS based FIR filters," in *Proc. IEEE IOLTS*, 2008, pp. 192–194.

[8] Z. Gao, W. Yang, X. Chen, M. Zhao and J. Wang, "Fault missing rate analysis of the arithmetic residue codes based fault-tolerant FIR filter design," in *Proc. IEEE IOLTS*, 2012, pp. 130–133.

[9] B. Shim and N. Shanbhag, "Energy-efficient soft error-tolerant digital signal processing," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 14,no. 4, pp. 336–348, Apr. 2006.

[10] Y.-H. Huang, "High-efficiency soft-error-tolerant digital signal processing using fine-grain sub word-detection processing," *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 18, no 2, pp. 291–304, Feb. 2010.

[11] P. Reviriego, C. J. Bleakley, and J. A. Maestro, "Structural DMR: A technique for implementation of soft-error-tolerant FIR filters," *IEEETrans. Circuits Syst. II: Exp. Briefs*, vol. 58, no. 8, pp. 512–516,Aug. 2011.

[12] P. Reviriego, S. Pontarelli, C. Bleakley and J. A. Maestro, "Area efficient concurrent error detection and correction for parallel filters," *IET Electron.Lett.*, vol. 48, no 20, pp. 1258–1260, Sep. 2012.

[14] R. W. Hamming, "Error correcting and error detecting codes," *Bell Sys.*

Tech. J., vol. 29, pp. 147–160, Apr. 1950.

[15] A. Chatterjee, and M. A. d'Abreu, "The design of fault-tolerant linear digital state variable systems: Theory and techniques," *IEEE Trans. Comput.*,vol. 42, no. 7, pp. 794–808, Jul. 1993.

[16] C. N. Hadjicostis, "Coding Approaches to Fault Tolerance in Dynamic Systems," Ph.D. dissertation, MIT Press, Cambridge, MA, USA, 1999

[17] D. Radaelli, H. Puchner, S. Wong, and S. Daniel, "Investigation of multi-bit upsets in a 150 nm technology SRAM device," *IEEE Trans.Nucl. Sci.*, vol. 52, no. 6, pp. 2433–2437, Dec. 2005.

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

[19] C. Argyrides and D. K. Pradhan, "Improved decoding algorithm for high reliable reed muller coding," in *Proc. IEEE Int. Syst. On Chip Conf.*,Sep. 2007, pp. 95–98.

[20] A. Sanchez-Macian, P. Reviriego, and J. A. Maestro, "Hamming SEC-DAED and extended hamming SEC-DED-TAED codes through selective shortening and bit placement," *IEEE Trans. Device Mater. Rel.*,to be published.