International Journal of LNCT, Vol 2(7)
ISSN (Online): 2456-9895

A review on efficient parallel filter design of coding scheme with lower delay and area profile

Kavita Goswami[1] and A.K.Shrivastava[2]
M.Tech. Scholar, LNCT Bhopal[1]
Assistant Professor, LNCT Bhopal[2]

Abstract
The increasing demand of low complexity and error tolerant design in signal processing systems is a reliability issue at ground level. Complex circuit is consistently affected by soft errors in modern electronic circuits. Different algorithms have been used in earlier techniques for achieving fault tolerant coverage design. In real time application systems, numbers of blocks operating in parallel are frequently used. As complexity is continuously increasing , design and reliability engineers will demand to address several key areas to enable advanced SoC (System-on-chip) products in commercial sector. Many techniques has been identified for error free circuit in which fault tolerant parallel filter using Error correction codes is latest.

Keywords
Parallel filter, Fault tolerant, Soft error, Error correction codes.

1.Introduction
Traditionally, the problem of computational fault-tolerance has been solved through modular redundancy. In this technique, several identical copies of the system operate in parallel using the same data, and their outputs are compared with voter circuitry. If no errors have occurred, all outputs will agree exactly. Otherwise, if an error has occurred, the faulty module can be easily identified and the correct output determined. Modular redundancy is a general technique and can be applied to any computational task. Unfortunately, it does not take advantage of the structure of a problem and requires a large amount of hardware overhead relative to the protection afforded. A more efficient method of protecting computation is to use an arithmetic code and tailor the redundancy to the specific operation being performed[1-6].

Arithmetic codes are essentially error-correcting codes whose error detecting and correcting properties are preserved during computation. Arithmetic codes offer performance and redundancy advantages similar to existing error-correcting codes used to protect communication channels. Unfortunately, arithmetic codes exist for only a limited number of operations.

This thesis addresses the general problem of designing an arithmetic code to protect a given computation. A practical arithmetic code must satisfy three requirements:
1. It must contain useful redundancy.
2. It must be easily encoded and decoded.
3.Its inherent redundancy must be preserved throughout computation.

The first two requirements are shared by error-correcting codes for protecting data transmission, while the third requirement is unique to arithmetic codes. This host of requirements makes designing practical arithmetic codes an extremely difficult problem. We solve this problem through two key insights. First, we note that many important arithmetic operations can be modeled using group theory. This provides a mathematically rigorous and well-defined foundation for our analysis. Second, we begin our study by focusing on the third requirement of an arithmetic code: that its structure be preserved throughout computation[7-11].

Error-detecting and error-correcting codes provide fault tolerance while using information redundancy, i.e. the data includes additional information (check bits) that can verify the correctness of the data before it is used (error- detection), or even correct erroneous data bits (error-correction). Different error-detecting and error-correcting codes have been proposed including parity codes, cyclic codes, arithmetic codes etc[12-16].

2.System model
The system model is presented in Figure 1. The architecture shown in Figure 2 consists of two processing nodes (processors), a shared memory and a Compare & Control Unit (CCU) connected through a shared bus. In such architecture RRC is performed as follows. Each job is duplicated and concurrently executed on both processing nodes. At a given time (a checkpoint request), the execution of the job is interrupted and a checkpoint is taken at each node. The checkpoint includes sufficient information such that the job can be resumed from that particular point. We consider a checkpoint to be represented as the

state (status) of a processing node. Once the states of both nodes are obtained, each processing node sends its state to the CCU. The CCU compares the states from both processing nodes. If the states match, i.e. no errors are detected; the CCU stores one of the states in memory and signals to the processing nodes to continue with the execution of the job. If the states do not match, i.e. an error is detected; the CCU loads the most recently saved state from memory and sends it to both processing nodes forcing them to roll-back the execution of the job.

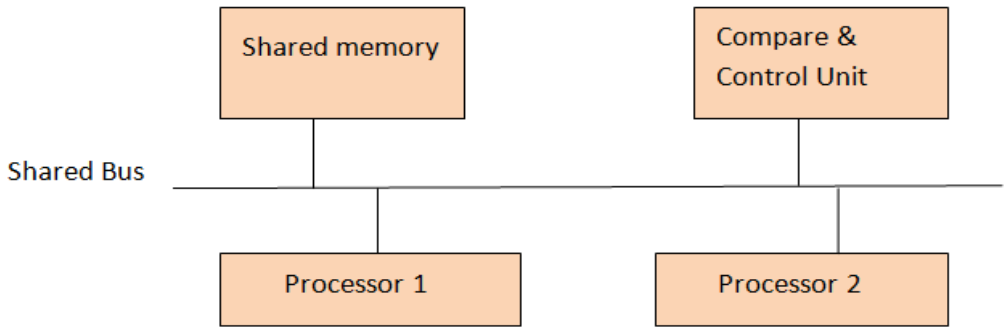

Figure1 System model

3.Significant issues dealt in previous work

Certain hardware components are critical for a given application, whereas others are not. Also, some portions of the application code are executed repeatedly at regular intervals, whereas others are not. These two aspects together determine how robust the application is to soft errors, and contributes the derating built into the system which is driven by this application. This also allows the application to intrinsically mask or recover from transient faults occurring in certain hardware components. Let us consider a few scenarios in which the application execution is not affected by soft errors.

I.Present day integrated circuits are built with lot of programmability and with a large application base in mind. Particular portions of the logic may never be required for a particular application.
II. A register or memory bit content may get corrupted due to a soft error, but not be accessed thereafter at all.
III. A register or memory bit content may get corrupted due to a soft error, but it is updated before the next access to it is made.

The knowledge about the safety criticality of the end applications running on a system incorporating the circuit modules helps to identify the scenarios listed above and suitably choose and incorporate soft error mitigation hardware in these circuit modules. In this chapter, we propose a hardware-software co-design methodology to identify such critical hardware components in different safety critical applications. Soft error mitigation techniques can be applied to only this subset of critical hardware components, thereby reducing the implementation overhead. We illustrate this trade-off through an application scenario in an embedded control system executing a safety critical control loop.

4.Problem identification

In the previous work the efficient coding schemes for fault - tolerant parallel filters are the filters that has been used in the filter bank for the communication channel in the protection status. The data's communicated through this channel length have been arrived from the associated architecture In many cases, the filters perform the same processing on different incoming signals as there is a tendency to use multiple-inputmultiple-output systems. This parallel operation can be exploited for fault tolerance. In fact, reliability is a major challenge for electronic systems. In particular, soft errors are an important issue, and many techniques have been proposed over the years to mitigate them. Some of these techniques modify the low-level design and implementation of the integrated circuits to prevent soft errors from occurring. Other techniques work at a higher abstraction level by adding redundancy that can detect and correct errors. The proposed method some error correcting code can be designed and can be expended with new low bit or higher bit architecture and concluded with parameters like area, delay and power can be determined.

5.Proposed design

An efficient coding scheme for eleven parallel filter has been proposed in this work illustrated in figure 5.1 General Form of Proposed Coding Scheme. There are 4X6 coding matrix having input vector X_1, X_2, X_3, X_4, and output vectors y_1, y_2, y_3, y_4, Two modules are there original modules and redundant modules are designed to work parallel. A general form of coding scheme of proposed work has given in figure 5.2. A redundant module generate a redundant signal Z_5 and Z_6 for input signal Z_1, Z_2,Z_3,Z_4.

The expression for redundant signal can be expressed as

$$X5=a51X1+a52X2+a53X3+a54X4 \quad \text{.....(1)}$$

$$X6=a61X1+a62X2+a63X3+a64X4 \quad \text{.....(2)}$$

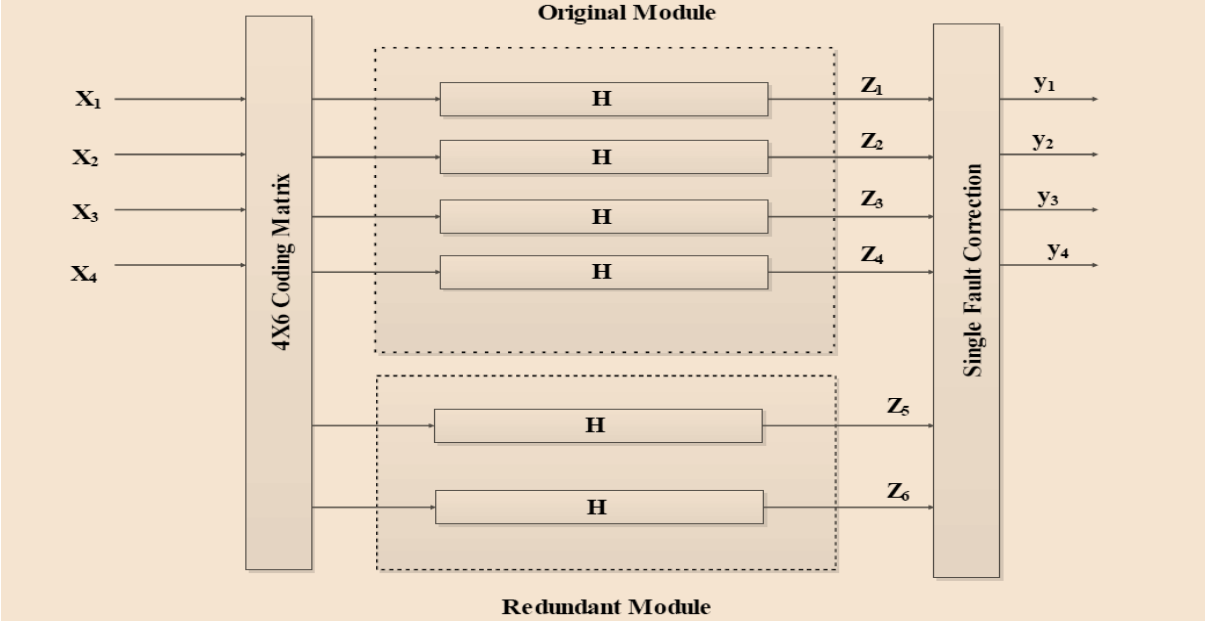

Figure 2 General Form of Proposed Coding Scheme

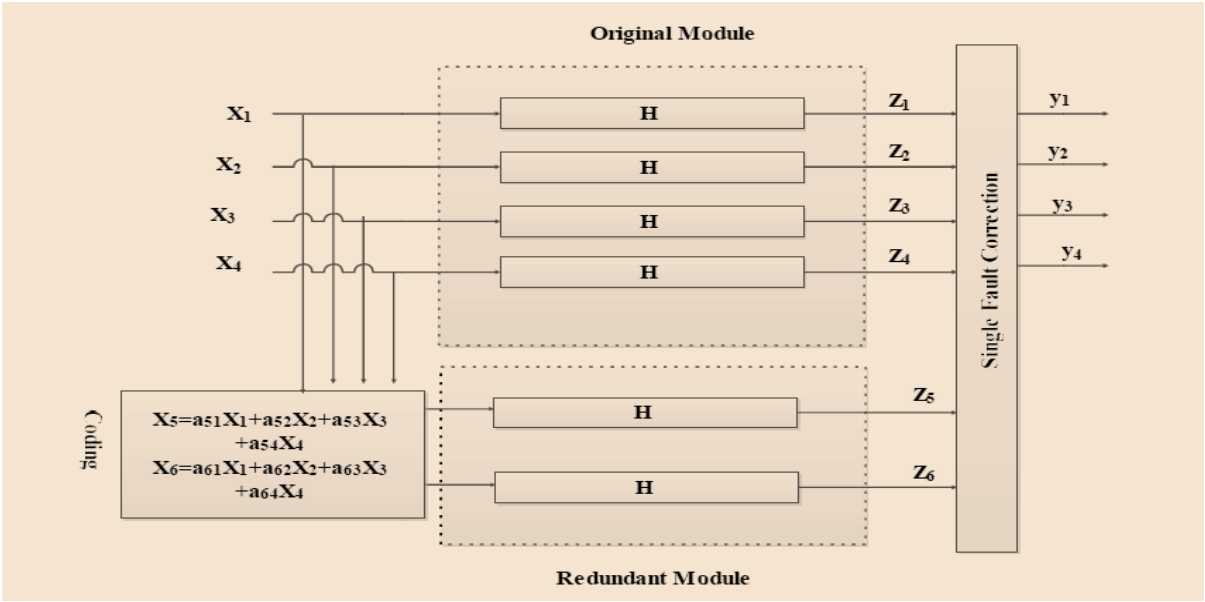

Figure 3 Practical coding scheme to protect four parallel filters

It can be examined that from Figure 2 that the number increases with the logarithm in construct two in light of the quantity of filters. Along these lines, the cost is considerably smaller.

The coding of the redundant filters proposed in this work are basically depends simple additions operation that replace the XOR binary operations in traditional ECCs. However, since both the data sources and yields of the filters are arrangements of numbers, a more general coding can be used. This type of coding has been explored for linear time-invariant systems refer to Eq. no (1) and Eq. no(2)

Practical coding plan to ensure four parallel filters. The input signals are encoded utilizing a matrix with arbitrary coefficients to generate the signals that enter the four original and two redundant filters the redundant signal is X_5 and X_6.

The error correction and detection logic can be simplified assuming that there is only a single error.

Figure5.3 has given the RTL schematic of proposed Efficient Parallel Filter Design Coding Scheme with Lower Delay and Area Profiles. The RTL schematic of proposed work has been illustrated n figure5.3. Obtained from the synthesis of proposed design on Xilinx 13.1 ISE design suite in device family Vertex 4 FPGA.

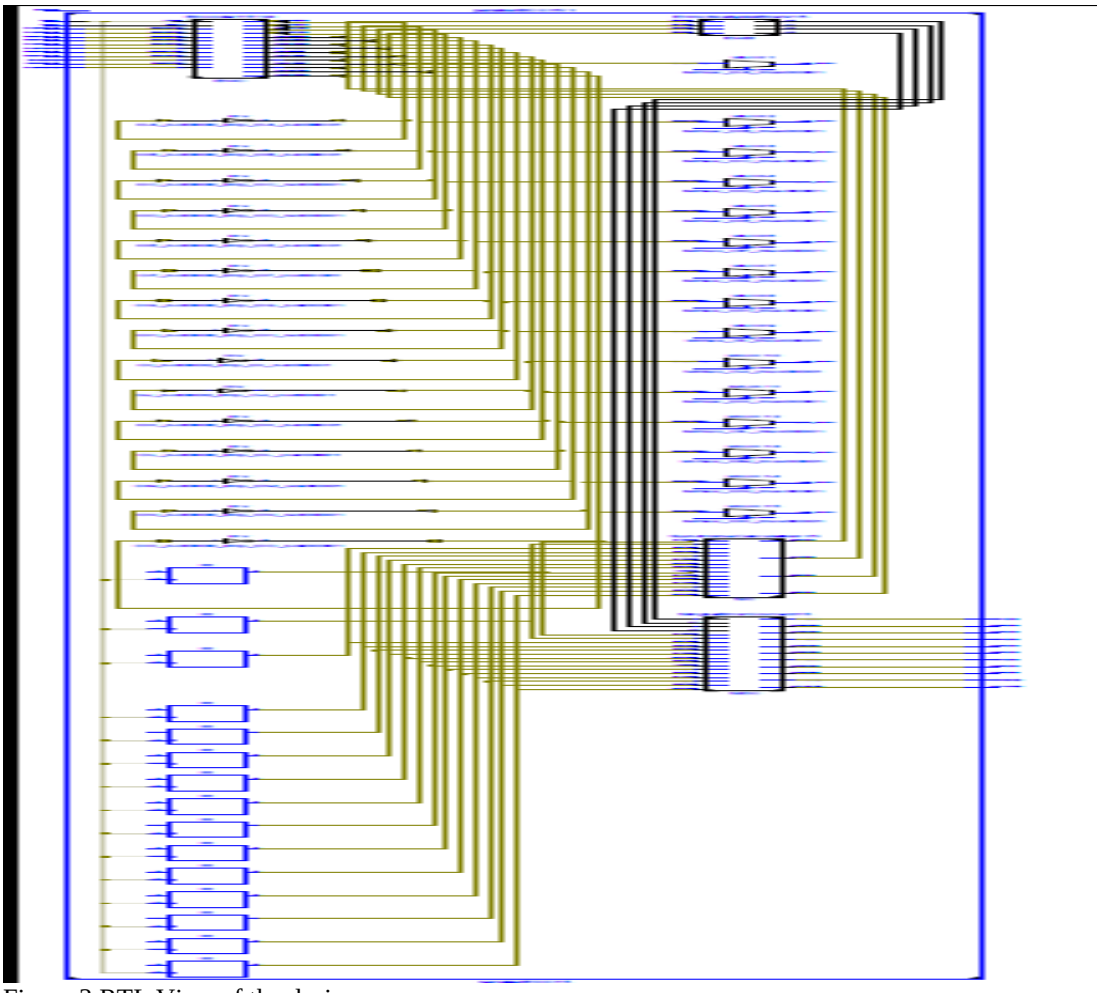

Figure 3 RTL View of the design

6.Synthesis report

Proposed Design of an efficient coding scheme of parallel filter has been implemented and synthesised in Xilinx ISE design suite 1.3.1 using FPGA vetex 4 XC4VLX80 device family. And Simulated using Isin Simulator. The Synthesis screen of proposed design has been given in figure 6.1. It is clear from device utilization summary under synthesis report that the proposed design has batter performance as compared to existing work with respect to area calculated in

number of slices flip-flops and IO buffs. The comparison between proposed work and existing work has given in Figure 4.

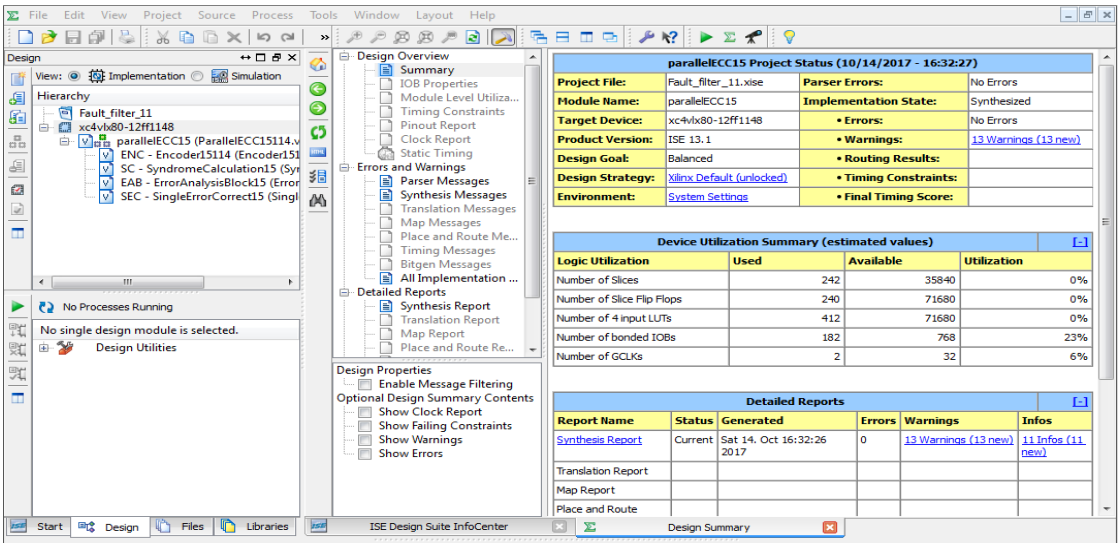

Figure 6 Synthesis screen of proposed work

The comparison between the proposed work and exsisting work has been given in Table 1.

Table 1 Resource comparison for eight parallel filters

	ECC Based	Proposed	Saving
Slices	11589	9872	14.8%
Flip-Flops	5286	3994	24.4%
LUTs	21872	19136	12.5%

Table 2 Resource comparison for eleven parallel filters

	ECC Based	Proposed	Saving
Slices	35840	305	99%
Flip-Flops	71680	240	99%
LUTs	71680	412	99%

Comparison plot of proposed work v/s previous work

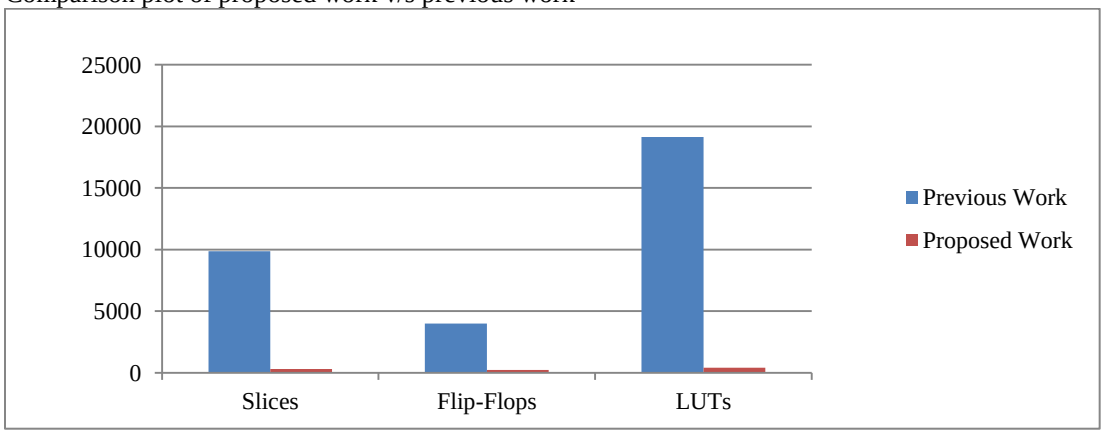

Figure 7 Plot 6.1: Proposed work v/s previous work

7.Conclusion and future scope

Proposed filter design presented in this work have been implemented in HDL and mapped standard verification environment of the FPGA in Virtex 4 XC4VLX80 device family. The design of a efficient coding scheme for eleven parallel filter architecture which is fault tolerant is suitable for applications that require low power consumption and a occupied small silicon area. The main advantage of the design is that they can be implemented in any FPGA; meaning that they are not dependent on the platform. The comparison of result evaluated that the proposed design have better device utilization as compared to existing design. The extended design is having better area and speed profiles than previous less number of parallel filters. There's a lot work to be done in the field of design parallel filter for fault tolerant computation with power management and delay management as well as area optimization and filtering level enhancement.

References

[1] Gao Z, Reviriego P, Xu Z, Su X, Wang J, Maestro JA. Efficient coding schemes for fault-tolerant parallel filters. IEEE Transactions on Circuits and Systems II: Express Briefs. 2015; 62(7):666-70.

[2] Gao Z, Reviriego P, Pan W, Xu Z, Zhao M, Wang J, Maestro JA. Fault tolerant parallel filters based on error correction codes. IEEE Transactions on very large scale integration (VLSI) systems. 2015; 23(2):384-7.

[3] Muralidharan KB, Kumar GS, Bhasi M. Fault tolerant state management for high-volume low-latency data stream workloads. In Data Science & Engineering (ICDSE), International Conference on 2014 (pp. 24-7). IEEE.

[4] Madhuri N, Doradla SR. Fault tolerant control of Hybrid power filter. In Advanced Communication Control and Computing Technologies (ICACCCT), 2014 International Conference on 2014 May 8 (pp. 235-9). IEEE.

[5] Park J, Park J, Bhunia S. VL-ECC: Variable data-length error correction code for embedded memory in DSP applications. IEEE Transactions on Circuits and Systems II: Express Briefs. 2014; 61(2):120-4.

[6] Asuvaran A, Senthilkumar S. Low delay error correction codes to correct stuck-at defects and soft errors. In advances in engineering and technology (ICAET), International Conference on 2014 (pp. 1-7). IEEE.

[7] Reviriego P, Pontarelli S, Bleakley CJ, Maestro JA. Area efficient concurrent error detection and correction for parallel filters. Electronics letters. 2012; 48(20):1258-60.

[8] Vaidyanathan PP. Multirate Systems and Filter Banks. Englewood Cliffs, NJ: PTR Prentice-Hall.

[9] Sibille A, Oestges C, Zanella A. MIMO: from theory to implementation. Academic Press; 2010.

[10] Kanekawa N, Ibe EH, Suga T, Uematsu Y. Dependability in electronic systems: mitigation of hardware failures, soft errors, and electro-magnetic disturbances. Springer Science & Business Media; 2010.

[11] Nicolaidis M. Design for soft error mitigation. IEEE Transactions on Device and Materials Reliability. 2005; 5(3):405-18.

[12] Chen CL, Hsiao MY. Error-correcting codes for semiconductor memory applications: A state-of-the-art review. IBM Journal of Research and Development. 1984; 28(2):124-34.

[13] Reddy AL, Banerjee P. Algorithm-based fault detection for signal processing applications. IEEE Transactions on Computers. 1990; 39(10):1304-8.

[14] Pontarelli S, Cardarilli GC, Re M, Salsano A. Totally fault tolerant RNS based FIR filters. In On-Line Testing Symposium, 2008. IOLTS'08. IEEE International 2008 (pp. 192-4). IEEE

[15] Gao Z, Yang W, Chen X, Zhao M, Wang J. Fault missing rate analysis of the arithmetic residue codes based fault-tolerant FIR filter design. In On-Line Testing Symposium (IOLTS), 2012 IEEE 18th International 2012 (pp. 130-3). IEEE.

[16] Shim B, Shanbhag NR. Energy-efficient soft error-tolerant digital signal processing. IEEE Transactions on Very Large Scale Integration (VLSI) Systems. 2006; 14(4):336-48.