International Journal of Engineering Research in Electronic and Communication Engineering (IJERECE) Vol 3, Issue 4, April 2016

Error Correction in Parallel Filters Using Error Correction Codes

[1]Saju Raju, [2]Sreejesh Kumar R
[1] PG Student [VLSI and ES] [2] Assistant professor,
Department of ECE, TKM Institute of Technology, Kollam
[1]sajuraju1991@gmail.com, [2] sjkumar.pillai@gmail.com

***Abstract*— Digital filters are widely used in communication and signal processing systems. In modern signal processing circuits, it is common that several filters are operating in parallel. In some cases, the reliability of those systems are very critical and thus a fault tolerant filter implementation is necessary. Triple Modular Redundancy (TMR) protect the filters from errors by adding redundancy at the logic or system level. But it requires larger area and power for protecting the digital filters from errors. This drawback leads to an alternate solution, which is the error correction codes (ECC) used to protect parallel filters from errors. In this work, hamming code is used as an Error correction code (ECC). It mainly focus on detection and correction of multiple bit errors occurring in parallel filters that have either the same impulse response or the same input data. This technique provides more efficient implementations when the number of parallel filters are large. This scheme focus on more powerful protection using advanced error correction codes (ECC) that can correct failures in multiples modules.**

***Index Terms*—Error Correction Codes(ECC),Filters, Soft errors .**

I. INTRODUCTION

Finite Impulse Response (FIR) filter is digital filters that has finite impulse response and are extensively used in signal processing and communication system applications like noise reduction, echo cancellation, image enhancement, speech and waveform synthesis etc. In signal processing, a digital filter is a device or process that removes some unwanted component from a signal. Digital filters are mainly used for two general purposes, they are separation of signals which is combined and restoration of signals that have been distorted .It means by removing some frequencies it will suppress interfering signals and reduce background noise. Parallel filters are commonly used in modern signal processing and communication systems. It perform the same processing on different incoming signals and that are mainly used in multiple input multiple output systems. In this systems, soft errors causes reliability problems which is the major challenge for electronic systems.

A number of techniques can be used to protect the filters from errors. Those range from modifications in the manufacturing process of the circuits which reduce the number of errors in the systems. Then redundancy can be added at the system level or logic level to ensure that system functionality not affected by the errors. One classical example is the use of Triple Modular Redundancy (TMR) , the design is tripled and a majority vote of the outputs is used to correct errors. But it requires larger area and power for protecting the digital filters from errors. Other techniques used to protect the filters from errors are structural dual modular redundancy and Residue Number Systems. Both these techniques are focused on protection of a single filter. Parallel filters can be protected by treating each bit as a filter and additional filters are added to act as parity check bits using Error Correction Code (ECC). This means that for single bit error correction, the number of redundant filters needed is the same as the number of bits needed in a traditional Hamming code. This same technique can be extended for multiple bit error correction.

II. LITERATURE REVIEW

There are a number of techniques by which a circuit can be protected from errors. These techniques ranges from modifications of circuits during manufacturing to reduce the errors to adding redundancy at system level or logic level to make sure that the errors do not affect the functionality of system[2]. A technique known as triple modular redundancy (TMR) is used generally to add redundancy. The triple modular redundancy not only triplicates the design but also adds voting logic to correct errors. But the major drawback of the technique is increase in area and power of the circuit which is not acceptable in most of the applications[3].

One of the most widely used signal processing circuits is digital filters. Several techniques are used to protect them from errors. The most commonly used one is

finite-impulse response (FIR) filters[4]. Reduced precision replicas was introduced to reduce the cost of modular redundancy implementation in FIR filters. To detect errors relationship between memory elements of an FIR filter and the input sequence is considered[5]. The FIR properties at word level to also in other schemes to achieve fault tolerance[6]. Residue number systems[7] and arithmetic codes[8] were also widely used to protect filters. Finally, different implementation structures of the FIR filters were used to correct errors having only one redundant module[9]. In all the above techniques, the protection single filters only.

However, in case of multiple filters that operate in parallel, protection can be addressed by considering the whole set of parallel filters to be protected as a block. In this case, parallel filters having same response but different input signal. The single error correction can be implemented with just one redundant copy. Thus, by comparing with TMR significant cost will reduced. This technique enables efficient implementation when there are a large number of parallel filters.

III. PROPOSED SYSTEM

The basic block diagram for the Error correction in parallel FIR Filters using Error Correction Codes is shown in the Figure 1. The input signals are given to the parallel filters and these signals are coded in coding block and given to the redundant filters. Then the outputs of the both the parallel filters and redundant filters are compared. If both the outputs are same then there is no error correction is needed. If any mismatch occurs, then Error correction block correct that error and the result is obtained at the output stage.

A. FIR Filter Realization

The signed binary sequence x(n) is given as an input to the filter. Input x(n) is delayed N times, thus get input as x(n),x(n-1),x(n-2).....x(n-N). These inputs are multiplied by the coefficients $h_0,h_1,h_2.....h_N$ respectively. The coefficients are generated from the Matlab FDA Tool. These multiplied inputs $h_0x(n),h_1x(n-1),.....h_Nx(n-N)$ are summed to get the output y(n):

$$y[n] = \sum_{l=0}^{N-1}(x[n-l].h[l]) \quad (1)$$

B. Single Bit Error Correction

In this system each filter is considered as a bit in the hamming code. For constructing hamming code two parameters are needed, they are block length (n) and number of message bits (k). So hamming code can be denoted as (n,k). The values of 'n' and 'k'can be calculated by following equation:

$$\text{Block length}(n)=2^m-1 \quad (2)$$
$$\text{Number of message bits}(k)=2^m-m-1 \quad (3)$$
$$\text{Number of parity check bits}(m)=n-k \quad (4)$$
$$\text{Miminium distance } (d_{min})=3 \quad (5)$$

The number of parity check bits m can be selected as 4. By substituting value of m in following equation, then we get n=15 and k=11.From this, n represents the total number of filters used in the system, number of original filters is represented as k and m is used to denote the number of redundant filters. The input of the four parallel filters are denoted as x1,x2,x3,x4,x5,x6,x7,x8,x9,x10,x11 which are single bit and the input of the redundant filter are denoted as x12,x13,x14,x15 which are obtained by the xor combination of the inputs

$$x12=x1\oplus x2\oplus x4\oplus x5\oplus x7\oplus x9\oplus x11 \quad (6)$$
$$x13=x1\oplus x3\oplus x4\oplus x6\oplus x7\oplus x10\oplus x11 \quad (7)$$
$$x14=x2\oplus x3\oplus x4\oplus x8\oplus x9\oplus x10\oplus x11 \quad (8)$$
$$x15=x5\oplus x6\oplus x7\oplus x8\oplus x9\oplus x10\oplus x11 \quad (9)$$

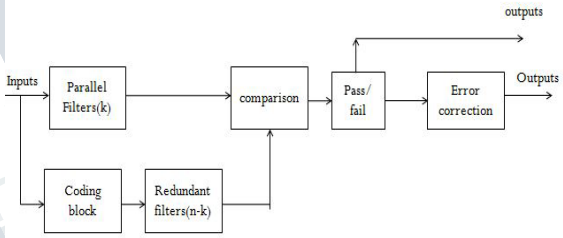

Fig 1:Block Diagram Of Error Correction System

Even if an error occur in one of the data bit or check bit they can be recovered and stored later . This can be performed by recomputing the parity check bits and results are compared with the values stored. In the example considered, an error on x1 will cause errors on x12 and x13; an error on x2 will cause errors on x5 and x6; an error on x3 in x5 and x7; and finally an error on x4 in x6 and x7. Therefore, the data bit in error can be located and the error can be corrected.

Hamming codes can be constructed using Matrix method. Two matrices are of great importance in the construction of Hamming Codes and identification of errors respectively. They are generator matrix and parity check matrix. The generator matrix is used to generate the code words and the parity check matrix is used to detect errors in the transmitted code words.

The G matrix comprises of n rows and K columns. The data bits are K bit length and simply multiply with the G

matrix. This will produce a codeword having a length of n-bit then divide the message into blocks of k bits length and do the multiplication with the G matrix. The multiplication will repeated till the input bits are exhausted. To check for errors, multiply the codewords with H matrix. The H matrix comprises of n rows and n-K columns. This multiplication gives the position of the bit that is corrupted. the result will be '0', If there is no error. This error position is termed as Syndrome and the technique is called Syndrome Decoding.
The simplest way of code construction is to first construct the parity check matrix (H) and then derive the Generator matrix(G) from it. H matrix is constructed with n-K columns and n rows. So in this case, the H matrix is a 15×4 matrix. For this simply write the binary representation of numbers from 1 to 15 column wise (just omit the number'0'). So the parity check matrix in non systematic form can be represented in columns as 0001, 0010 ,0011, 0100, 0101, 0110, 0111, 1000, 1001,1010,1011,1100,1101,1110,1111. It can be written in any order. Finally, H matrix of dimension n × (n-k) can be obtained by transposing the resulting matrix . Accordingly the G Matrix will also change and get an entirely different set of codewords.

Encoding is done by computing y=x·G and error detection is done by computing $s = y \cdot H^T$, where the operator (·) represents module two addition and multiplication. Correction is done using the syndrome vector s in order to identify the bit in error. The error position corresponding to the values of s is shown in Table 1. Once the erroneous bit is identified, it is corrected by simply inverting the bit.

Table 1: Error Correction in the Hamming Code

$S_0S_1S_2S_3$	Bit Error Position	Action
0000	No Error	None
1111	X_1	Correct X_1
1110	X_2	Correct X_2
0011	X_3	Correct X_3
1101	X_4	Correct X_4
0101	X_5	Correct X_5
0110	X_6	Correct X_6
0111	X_7	Correct X_7
1100	X_8	Correct X_8
1001	X_9	Correct X_9
1010	X_{10}	Correct X_{10}
1011	X_{11}	Correct X_{11}
1000	X_{12}	Correct X_{12}
0100	X_{13}	Correct X_{13}
0010	X_{14}	Correct X_{14}
0001	X_{15}	Correct X_{15}

C. Multiple bit error correction

The overall system is illustrated in fig 2. It consists of 11 parallel filters and 4 redundant filters. The input of the parallel filters are denoted as x1, x2, x3, x4, x5, x6, x7, x8, x9, x10, x11 which of 8-bitlength and the input to the redundant filters are obtained from the following equation

$$x12=x1\oplus x2\oplus x4\oplus x5\oplus x7\oplus x9\oplus x11 \quad (10)$$
$$x13=x1\oplus x3\oplus x4\oplus x6\oplus x7\oplus x10\oplus x11 \quad (11)$$
$$x14=x2\oplus x3\oplus x4\oplus x8\oplus x9\oplus x10\oplus x11 \quad (12)$$
$$x15=x5\oplus x6\oplus x7\oplus x8\oplus x9\oplus x10\oplus x11 \quad (13)$$

In this system 4-tap FIR filter is used. so four coefficient values are generated from the Matlab FDA tool which are represented as h1,h2,h3,h4 which is of 8 bit length. The four parallel filter output is represented as y1, y2, y3, y4, y5, y6,y7,y8,y9,y10,y11 and the redundant filter output is denoted as z1,z2,z3,z4

Output of parallel filters are:

$$y_1[n] = \sum_{l=0}^{N-1}(x_1[n-l].h[l]) \quad (14)$$
$$y_2[n] = \sum_{l=0}^{N-1}(x_2[n-l].h[l]) \quad (15)$$
$$y_3[n] = \sum_{l=0}^{N-1}(x_3[n-l].h[l]) \quad (16)$$
$$y_4[n] = \sum_{l=0}^{N-1}(x_4[n-l].h[l]) \quad (17)$$
$$y_5[n] = \sum_{l=0}^{N-1}(x_5[n-l].h[l]) \quad (18)$$
$$y_6[n] = \sum_{l=0}^{N-1}(x_6[n-l].h[l]) \quad (19)$$
$$y_7[n] = \sum_{l=0}^{N-1}(x_7[n-l].h[l]) \quad (20)$$
$$y_8[n] = \sum_{l=0}^{N-1}(x_8[n-l].h[l]) \quad (21)$$
$$y_9[n] = \sum_{l=0}^{N-1}(x_9[n-l].h[l]) \quad (22)$$
$$y_{10}[n] = \sum_{l=0}^{N-1}(x_{10}[n-l].h[l]) \quad (23)$$
$$y_{11}[n] = \sum_{l=0}^{N-1}(x_{11}[n-l].h[l]) \quad (24)$$

Checking is done by testing the following equations:

$$z_1[n] = y_1[n] + y_2[n] + y_4[n] + y_5[n] + y_7[n] + y_9[n] + y_{11}[n]$$
$$z_2[n] = y_1[n] + y_3[n] + y_4[n] + y_6[n] + y_7[n] + y_{10}[n] + y_{11}[n]$$
$$z_3[n] = y_2[n] + y_3[n] + y_4[n] + y_8[n] + y_9[n] + y_{10}[n] + y_{11}[n]$$
$$z_4[n] = y_5[n] + y_6[n] + y_7[n] + y_8[n] + y_9[n] + y_{10}[n] + y_{11}[n]$$

Consider an example, an error on filter y1 will cause errors on the checks of z1, z2 .Similarly, errors on the other filters will cause errors on the other check filters. The correction can be done using only four redundant filters. For the parallel filters, correction is achieved by reconstructing the erroneous output using the rest of the data and check outputs. For

example, when an error is detected in filter y1, it can be corrected by using this equation:

$$yc_1 = z_1[n] - y_2[n] - y_4[n] - y_5[n] - y_7[n] - y_9[n] - y_{11}[n]$$

Similarly, other erroneous filter outputs can be calculated with different correction equations used. Error protection is carried out by considering the fact that the redundant module is functionally equivalent to the primary implementation of the original filter. It is focused on protecting single filters against transient errors and relied on recognition of temporal error patterns. It has the advantages of protecting parallel filters against transient and permanent errors, and is valid for use with finite impulse response(FIR) as well as infinite impulse response (IIR) filters.

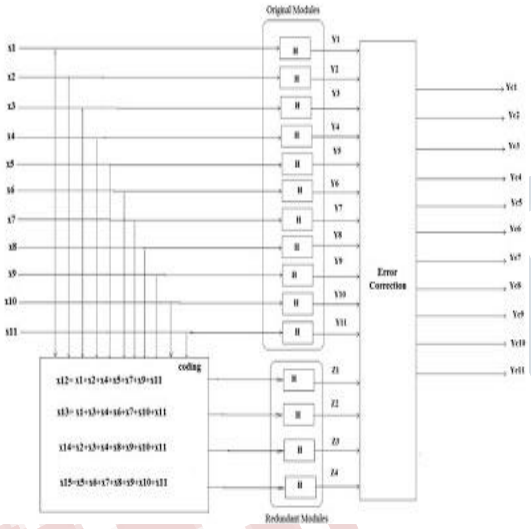

Fig 2: Overall system architecture

IV. SIMULATION RESULTS

The modules are synthesized and simulated using VHDL in Xilinx ISE Design 13.2. Simulation results of error correction in parallel FIR filters are shown in (fig:3) where x1, x2, x3, x4,x5,x6,x7,x8,x9,x10,x11 are the input to parallel filters which are 8-bit length whose outputs are y1,y2,y3,y4,y5,y6,y7, y8, y9, y10, y11 respectively and corrected output of parallel filters are yc1,yc2,yc3,yc4,yc5, yc6, yc7, yc8, yc9, yc10, yc11 respectively. Coefficients of the filters are h1,h2,h3,h4 which are generated with help of matlab fda tool and x12,x13,x14,x15 are the input of redundant filters whose output are z1,z2,z3,z4 respectively. The erroneous data 'v' which is a 8-bit data, which is placed in first filter instead of h1. As a result output of first filter become error and corrected output is obtained in yc1.

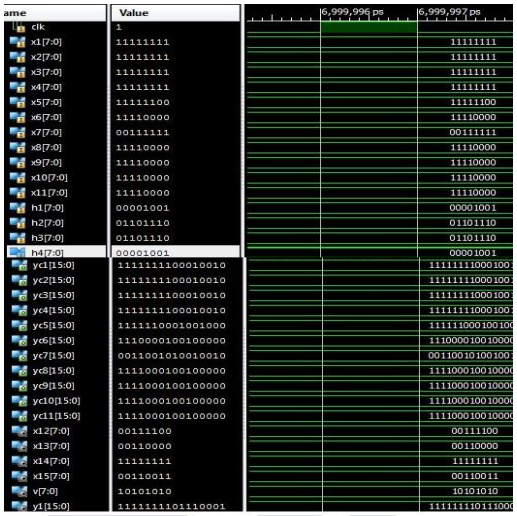

Fig.3: Output of parallel filter

V. CONCLUSION

Fault tolerant parallel filters have been designed which presents a new scheme to protect parallel filters that are commonly found in modern signal processing circuits. This approach is based on applying Error Correction Code (ECC) to the parallel filter's outputs to detect and correct errors. It can be used for parallel filters that have the same response and process different input signals. This technique provides larger benefits when the number of parallel filters is large.

REFERENCES

[1] Zhen Gao, Pedro Reviriego, Wen Pan, Zhan Xu, Ming Zhao, Jing Wang,"Fault Tolerant Parallel Filters Based on Error Correction Codes", *IEEE Transactions On VLSI SYSTEMS*, Vol.23,No.2, February 2015.

[2] M. Nicolaidis, "Design for soft error mitigation," *IEEE Trans. DeviceMater. Rel.*, vol. 5, no. 3, pp. 405–418, Sep. 2005.

[3] R. Hentschke, F. Marques, F. Lima, L. Carro, A. Susin, R. Reis "Analyzing Area and Performance Penalty of Protecting Different Digital Modules with Hamming Code and Triple Modular Redundancy", *Proceedings of the 15 th Symposium on Integrated Circuits and Systems Design 2012*.

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

[5]T. Hitana and A. K. Deb, "Bridging concurrent and non-concurrent error detection in FIR filters," in *Proc. Norchip Conf.*, 2004,pp. 75–78.

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

[7] S. Pontarelli, G. C. Cardarilli, M. Re, and A. Salsano, "Totally fault tolerant RNS based FIR filters," in *Proc. IEEE IOLTS*, Jul. 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*, Jun. 2012, pp. 130–133.

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