IPHV8I2043X
International Journal Of Advanced Research and Innovation -Vol.8, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

DESIGN OF EFFICIENT CODING SCHEMES FOR FAULT-TOLERANT PARALLEL FILTERS

[#1]DASARI KIRAN KUMAR, M.Tech Student,
[#2]M.SIVANARAYANA, Associate Professor,
DEPT. OF ECE,
DRK INSTITUTE OF SCIENCE AND TECHNOLOGY, BOWRAMPET (V), TS, INDIA.

ABSTRACT: As the complexity of communications and signal processing systems increases, so does the number of blocks or elements that they have. In many cases, some of those elements operate in parallel, performing the same processing on different signals. A typical example of those elements is digital filters. The increase in complexity also poses reliability challenges and creates the need for fault-tolerant implementations. A scheme based on error correction coding has been recently proposed to protect parallel filters. In that scheme, each filter is treated as a bit, and redundant filters that act as parity check bits are introduced to detect and correct errors. In this brief, the idea of applying coding techniques to protect parallel filters is addressed in a more general way. In particular, it is shown that the fact that filter inputs and outputs are not bits but numbers enables a more efficient protection. This reduces the protection overhead and makes the number of redundant filters independent of the number of parallel filters. The proposed scheme is first described and then illustrated with two case studies. Finally, both the effectiveness in protecting against errors and the cost are evaluated for a field-programmable gate array implementation.

Key Terms— Error correction codes (ECCs), fast Fourier transforms (FFTs), soft errors.

I.INTRODUCTION

The complexity of communications and signal processing circuits increases every year. This is made possible by the CMOS technology scaling that enables the integration of more and more transistors on a single device. This increased complexity makes the circuits more vulnerable to errors. At the same time, the scaling means that transistors operate with lower voltages and are more susceptible to errors caused by noise and manufacturing variations [1]. The importance of radiation-induced soft errors also increases as technology scales [2]. Soft errors can change the logical value of a circuit node creating a temporary error that can affect the system operation. To ensure that soft errors do not affect the operation of a given circuit, a wide variety of techniques can be used [3]. These include the use of special manufacturing processes for the integrated circuits like, for example, the silicon on insulator. Another option is to design basic circuit blocks or complete design libraries to minimize the probability of soft errors. Finally, it is also possible to add redundancy at the system level to detect and correct errors. One classical example is the use of triple modular redundancy (TMR) that triples a block and votes among the three outputs to detect and correct errors. The main issue with those soft errors mitigation techniques is that they require a large overhead in terms of circuit implementation. For example, for TMR, the overhead is >200%. This is because the unprotected module is replicated three times (which requires a 200% overhead versus the unprotected module), and additionally, voters are needed to correct the errors making the overhead >200%. This overhead is excessive for many applications. Another approach is to try to use the algorithmic properties of the circuit to detect/correct errors. This is commonly referred to as algorithm-based fault tolerance (ABFT) [4]. This strategy can reduce the overhead required to protect a circuit. Signal processing and communications circuits are well suited for ABFT as they have regular structures and many algorithmic properties [4]. Over the years, many ABFT techniques have been proposed to protect the basic blocks that are commonly used in those circuits. Several works have considered the protection of digital filters [5], [6]. For example, the use of replication using reduced precision copies of the filter has been proposed as an alternative to TMR but with a lower cost [7]. The knowledge of the distribution of the filter output has also been recently exploited to detect and correct errors with lower overheads [8]. The protection of fast Fourier transforms (FFTs) has also been widely studied [9], [10]. As signal-processing circuits become more complex, it is common to find several filters or FFTs operating in parallel. This occurs for example in filter banks [11] or in multiple-input multiple-output (MIMO) communication systems [12]. In particular, MIMO orthogonal frequency division modulation (MIMO-OFDM) systems use parallel

iFFTs/FFTs for modulation/demodulation [13]. MIMO-OFDM is implemented on long-term evolution mobile systems [14] and also on WiMax [15]. The presence of parallel filters or FFTs creates an opportunity to implement ABFT techniques for the entire group of parallel modules instead of for each one independently. This has been studied for digital filters initially in [16] where two filters were considered. More recently, a general scheme based on the use of error correction codes (ECCs) has been proposed [17]. In this technique, the idea is that each filter can be the equivalent of a bit in an ECC and parity check bits can be computed using addition. This technique can be used for operations, in which the output of the sum of several inputs is the sum of the individual outputs. This is true for any linear operation as, for example, the discrete Fourier transform (DFT). In this brief, the protection of parallel FFTs is studied. In particular, it is assumed that there can only be a single error on the system at any given point in time. This is a common assumption when considering the protection against radiation-induced soft errors [3]. There are three main contributions in this brief. 1) The evaluation of the ECC technique [17] for the protection of parallel FFTs showing its effectiveness in terms of overhead and protection effectiveness. 2) The proposal of a new technique based on the use of Parseval or sum of squares (SOSs) checks [4] combined with a parity FFT. 3) The proposal of a new technique on which the ECC is used on the SOS checks instead of on the FFTs.

II. PARALLEL FILTERS

The impulse response h[n] completely defines a discrete time filter that performs the following operation on the incoming signal x[n]:

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

The impulse response can be infinite or be nonzero for a finite number of samples. In the first case, the filter is an infinite impulse-response (IIR) filter, and in the second, the filter is a finite impulse-response (FIR) filter. In both cases, the filtering operation is linear such that

$$y_1[n] + y_2[n] = \sum_{l=0}^{\infty} (x_1[n-l] + x_2[n-l]) \cdot h[l]. \quad (2)$$

This property can be exploited in the case of parallel filters that operate on different incoming signals, as shown on Fig. 1. In this case, four filters with the same response process the incoming signals x1[n], x2[n], x3[n], and x4[n] to produce four outputs y1[n], y2[n], y3[n], and y4[n]. To detect and correct errors, each filter can be viewed as a bit in an ECC, and redundant filters can be added to form parity check bits [13]. This is also illustrated in Fig. 1, where three redundant filters are used to form the parity check bits of a classical single error correction Hamming code [14]. Those correspond to the outputs z1[n], z2[n], and z3[n]. Errors can be detected by checking if

$$\begin{aligned} z_1[n] &= y_1[n] + y_2[n] + y_3[n] \\ z_2[n] &= y_1[n] + y_2[n] + y_4[n] \\ z_3[n] &= y_1[n] + y_3[n] + y_4[n]. \end{aligned} \quad (3)$$

When some of those checks fail, an error is detected. The error can be corrected based on which specific checks failed. For example, an error on filter y1 will cause errors on the checks of z1, z2, and z3. Similarly, errors on the other filters will cause errors on a different group of zi. Therefore, as with the traditional ECCs, the error can be located. To correct the error, the failing output is reconstructed from the correct outputs. For example, when an error on y1 is detected, it can be corrected by making

$$y_{c1}[n] = z_1[n] - y_2[n] - y_3[n]. \quad (4)$$

This ECC-based scheme reduces the protection overhead compared with the use of TMR. Table I summarizes the number of redundant filters needed for different parallel filter configurations. It can be observed that the number grows with the logarithm in base two on the number of filters. Therefore, the cost is much smaller than TMR, in which the number of filters is tripled. The cost reductions were confirmed by some case study implementations in .

In this ECC-based scheme, the coding of the redundant filters is based on simple additions that replace the XOR binary operations in traditional ECCs. However, since both the inputs and outputs of the filters are sequences of numbers, a more general coding can be used. This type of coding has been explored for linear time-invariant systems (see, for example [15] and [16]) but not for parallel filters. In those works, the processing of the linear system is modified to incorporate error detection and correction mechanisms. This is different from the approach proposed in this brief, where inputs are encoded but the processing of the filters is not modified. In the following, the use of a coding scheme for parallel filters in which the redundant filters are constructed as linear combinations of the original filters with arbitrary coefficients is explored.

IPHV8I2043X
International Journal Of Advanced Research and Innovation -Vol.8, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

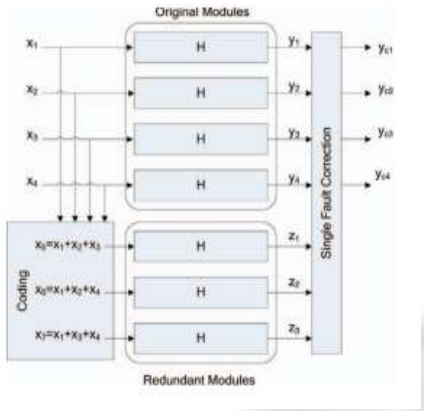

Fig. 1. ECC-based scheme for four filters and a Hamming code (see [13]).

III. ERROR CORRECTION CODES WITH PARALLEL FILTERS

The proposed scheme is illustrated in Fig. 2 for the case of four parallel filters. The input signals are encoded using a matrix with arbitrary coefficients to produce the signals that enter the four original and two redundant filters. In its more general form, this coding matrix A can be formulated as

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \\ a_{51} & a_{52} & a_{53} & a_{54} \\ a_{61} & a_{62} & a_{63} & a_{64} \end{pmatrix}. \quad (5)$$

Therefore, the input signal to the ith filter is of the form (i = 1, 2,... , or 6), i.e.,

$$v_i[n] = a_{i1} \cdot x_1[n] + a_{i2} \cdot x_2[n] + a_{i3} \cdot x_3[n] + a_{i4} \cdot x_4[n]. \quad (6)$$

In a practical implementation, the first four rows of the matrix would be an identity matrix so that the inputs to the original filters are the incoming signals. With this coding scheme, the outputs of the filters, i.e., y1[n], y2[n], y3[n], and y4[n], can be obtained as follows:

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}_{1235} = (A_{1235})^{-1} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \\ z_5 \end{pmatrix} \quad (7)$$

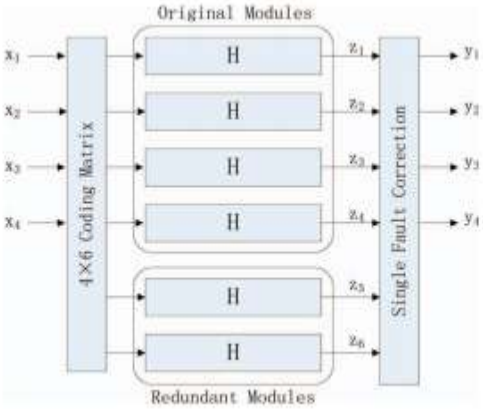

Fig. 2. Proposed coding scheme in general form

where A1235 is a sub matrix of A, including the first, second, third, and fifth rows. This process can be repeated with different sub matrixes of A, for example, with A1236, A2345, and A2346. In the error-free case, all the recovered versions of y1[n], y2[n], y3[n], and y4[n] will be the same. When there are differences, an error is detected. For example, suppose that

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}_{1235} \neq \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}_{1236} \quad \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}_{2345} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{pmatrix}_{2346} \quad (8)$$

which means that there is an error among filters {1 2 3 5 6} and that filters {2 3 4 5 6} are correct. Therefore, the faulty filter is filter 1. Then, the error can be corrected by taking the final outputs from a set that does not include filter 1. The error correction and detection logic can be simplified assuming that there is only a single error. In that case, checking only that, for each recovered set, the sums of the values y1[n] + y2[n] + y3[n] + y4[n] are equal is enough. In more detail, four checks are needed, each involving five filters and excluding one. For example, if branch 1 is excluded, the error checking would be

$$\begin{cases} s_1^1 = \bar{w}_{2345}(z_2 z_3 z_4 z_5)^T \\ s_1^2 = \bar{w}_{2346}(z_2 z_3 z_4 z_6)^T \\ e_1 = s_1^1 - s_1^2 \end{cases} \quad (9)$$

in which w¯2345 = [1111](A2345)−1, and w¯2346 = [1111] (A2346)−1. Similarly, if filters 2, 3, and 4 are excluded, the e2, e3, and e4 error signals can be generated as follows:

IPHV8I2043X
International Journal Of Advanced Research and Innovation -Vol.8, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

$$\begin{cases} s_2^1 = \bar{w}_{1345}(z_1 z_3 z_4 z_5)^T \\ s_2^2 = \bar{w}_{1346}(z_1 z_3 z_4 z_6)^T \\ e_2 = s_2^1 - s_2^2 \end{cases},$$

$$\begin{cases} s_3^1 = \bar{w}_{1245}(z_1 z_2 z_4 z_5)^T \\ s_3^2 = \bar{w}_{1246}(z_1 z_2 z_4 z_6)^T \\ e_3 = s_3^1 - s_3^2 \end{cases} \quad \begin{cases} s_4^1 = \bar{w}_{1235}(z_1 z_2 z_3 z_5)^T \\ s_4^2 = \bar{w}_{1236}(z_1 z_2 z_3 z_6)^T \\ e_4 = s_4^1 - s_4^2. \end{cases} \tag{10}$$

Then, with the error vector e = [e1, e2, e3, e4], the faulty filter can be located. The process is listed in Table II, in which "#" stands for a nonzero value. Once the faulty filter is identified, the error can be corrected by reconstructing the outputs using the remaining filters. For the proposed coding scheme to work, the encoding matrix has to fulfill some conditions. Let us consider the calculation of e1 and define (A2345)−1 and (A2346)−1 as

$e_1 e_2 e_3 e_4$	Filter in error
0 0 0 0	No error
0 # # #	z_1
# 0 # #	z_2
# # 0 #	z_3
# # # 0	z_4
# # # #	z_5
# # # #	z_6

LOCATION OF THE FAULTY FILTER

$$(A_{2345})^{-1} = \begin{pmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \\ b_{31} & b_{32} & b_{33} & b_{34} \\ b_{41} & b_{42} & b_{43} & b_{44} \end{pmatrix} \quad (A_{2346})^{-1} = \begin{pmatrix} c_{11} & c_{12} & c_{13} & c_{14} \\ c_{21} & c_{22} & c_{23} & c_{24} \\ c_{31} & c_{32} & c_{33} & c_{34} \\ c_{41} & c_{42} & c_{43} & c_{44} \end{pmatrix}. \tag{11}$$

Then

$$\begin{cases} \bar{w}_{2345} = [1111](A_{2345})^{-1} = \left[\sum_{i=1}^{4} b_{i1}, \sum_{i=1}^{4} b_{i2}, \sum_{i=1}^{4} b_{i3}, \sum_{i=1}^{4} b_{i4}\right] \\ \bar{w}_{2346} = [1111](A_{2346})^{-1} = \left[\sum_{i=1}^{4} c_{i1}, \sum_{i=1}^{4} c_{i2}, \sum_{i=1}^{4} c_{i3}, \sum_{i=1}^{4} c_{i4}\right] \end{cases} \tag{12}$$

and finally

$$\begin{cases} s_1^1 = \bar{w}_{2345}(z_2 z_3 z_4 z_5)^T = z_2 \sum_{i=1}^{4} b_{i1} + z_3 \sum_{i=1}^{4} b_{i2} \\ \qquad + z_4 \sum_{i=1}^{4} b_{i3} + z_5 \sum_{i=1}^{4} b_{i4} \\ s_1^2 = \bar{w}_{2346}(z_2 z_3 z_4 z_6)^T = z_2 \sum_{i=1}^{4} c_{i1} + z_3 \sum_{i=1}^{4} c_{i2} \\ \qquad + z_4 \sum_{i=1}^{4} c_{i3} + z_6 \sum_{i=1}^{4} c_{i4} \\ e_1 = z_2 \left(\sum_{i=1}^{4} b_{i1} - \sum_{i=1}^{4} c_{i1}\right) + z_3 \left(\sum_{i=1}^{4} b_{i2} - \sum_{i=1}^{4} c_{i2}\right) \\ \quad + z_4 \left(\sum_{i=1}^{4} b_{i3} - \sum_{i=1}^{4} c_{i3}\right) + z_5 \sum_{i=1}^{4} b_{i4} - z_6 \sum_{i=1}^{4} c_{i4}. \end{cases} \tag{13}$$

Therefore, the error check will ensure error detection when the sums of the columns in each of the matrixes A−1 2345 and A−1 2346 are different and nonzero. Furthermore, we can define the error detection matrix as C = [¯c1, c¯2, c¯3, c¯4] T , in which

$$\begin{cases} \bar{c}_1 = \left[0, \left(\sum_{i=1}^{4} b_{i1} - \sum_{i=1}^{4} c_{i1}\right), \left(\sum_{i=1}^{4} b_{i2} - \sum_{i=1}^{4} c_{i2}\right), \right. \\ \qquad \left. \left(\sum_{i=1}^{4} b_{i3} - \sum_{i=1}^{4} c_{i3}\right), \sum_{i=1}^{4} b_{i4}, -\sum_{i=1}^{4} c_{i4}\right] \\ \bar{c}_2 = \left[\left(\sum_{i=1}^{4} b_{i1} - \sum_{i=1}^{4} c_{i1}\right), 0, \left(\sum_{i=1}^{4} b_{i2} - \sum_{i=1}^{4} c_{i2}\right), \right. \\ \qquad \left. \left(\sum_{i=1}^{4} b_{i3} - \sum_{i=1}^{4} c_{i3}\right), \sum_{i=1}^{4} b_{i4}, -\sum_{i=1}^{4} c_{i4}\right] \\ \bar{c}_3 = \left[\left(\sum_{i=1}^{4} b_{i1} - \sum_{i=1}^{4} c_{i1}\right), \left(\sum_{i=1}^{4} b_{i2} - \sum_{i=1}^{4} c_{i2}\right), 0, \right. \\ \qquad \left. \left(\sum_{i=1}^{4} b_{i3} - \sum_{i=1}^{4} c_{i3}\right), \sum_{i=1}^{4} b_{i4}, -\sum_{i=1}^{4} c_{i4}\right] \\ \bar{c}_4 = \left[\left(\sum_{i=1}^{4} b_{i1} - \sum_{i=1}^{4} c_{i1}\right), \left(\sum_{i=1}^{4} b_{i2} - \sum_{i=1}^{4} c_{i2}\right), \right. \\ \qquad \left. \left(\sum_{i=1}^{4} b_{i3} - \sum_{i=1}^{4} c_{i3}\right), 0, \sum_{i=1}^{4} b_{i4}, -\sum_{i=1}^{4} c_{i4}\right]. \end{cases}$$

Then, the error detection vector can be expressed as e¯ = Cz¯T , in which e¯ = [e1, e2, e3, e4] T , and z¯ = [z1, z2, z3, z4, z5, z6]. Since the coding matrix A is known, matrix C can be determined in advance and stored for error detection. The complexity of the error detection and correction relies on the design of A. For a practical implementation, the first four filter outputs (z1[n], z2[n], z3[n], z4[n]) should be equal to the final filters outputs (y1[n], y2[n], y3[n], y4[n]) in order to avoid additional computations to reconstruct those outputs. This greatly

IPHV8I2043X
International Journal Of Advanced Research and Innovation -Vol.8, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

simplifies the implementation. In order to illustrate the use of the proposed coding scheme in a practical application, the next section presents two case studies that are then evaluated both in terms of implementation cost and protection effectiveness.

3.1 practical implementation

To illustrate the use of the proposed scheme, two case studies that consider the protection of four and eight parallel filters are presented here. In both cases, a coding matrix that preserves the inputs to the original filters is used. This is illustrated in Fig. 3 for the case of four parallel filters. The corresponding A matrix is the identity matrix on the first four rows, and only the last two rows have generic coefficients. The matrix is

$$A = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ a_{51} & a_{52} & a_{53} & a_{54} \\ a_{61} & a_{62} & a_{63} & a_{64} \end{pmatrix}. \quad (14)$$

To simplify the implementation, those rows should have values that minimize the complexity of multiplications and the increase in the dynamic range in the redundant filters. To that end, the following values can be selected for the last two rows: [a51 a52 a53 a54] = [1 1 1 1] and [a61 a62 a63 a64] = [1 2 3 4]. This produces the following check matrix:

$$C = \begin{bmatrix} \bar{c}_1 \\ \bar{c}_2 \\ \bar{c}_3 \\ \bar{c}_4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 2 & 3 & 1 & -1 \\ -\frac{1}{2} & 0 & \frac{1}{2} & 1 & 1 & -\frac{1}{2} \\ -\frac{1}{3} & -\frac{1}{3} & 0 & \frac{1}{3} & 1 & -\frac{1}{3} \\ -\frac{1}{4} & -\frac{1}{2} & -\frac{1}{4} & 0 & 1 & -\frac{1}{4} \end{bmatrix} \quad (15)$$

and error detection vector

$$\bar{e} = C\bar{z}^T = \begin{bmatrix} z_5 - z_6 + z_2 + 2z_3 + 3z_4 \\ \frac{1}{2}(2z_5 - z_6 - z_1 + z_3 + 2z_4) \\ \frac{1}{3}(3z_5 - z_6 - 2z_1 - z_2 + z_4) \\ \frac{1}{4}(4z_5 - z_6 - 3z_1 - 2z_2 - z_3) \end{bmatrix}. \quad (16)$$

By defining

$$\begin{cases} p_1 = z_5 - (z_1 + z_2 + z_3 + z_4) \\ p_2 = z_6 - (z_1 + 2z_2 + 3z_3 + 4z_4) \end{cases} \quad (17)$$

the check vector can be expressed as

$$\bar{e} = C\bar{z}^T = \begin{bmatrix} (p_1 - p_2) \\ \frac{1}{2}(2p_1 - p_2) \\ \frac{1}{3}(3p_1 - p_2) \\ \frac{1}{4}(4p_1 - p_2) \end{bmatrix}. \quad (18)$$

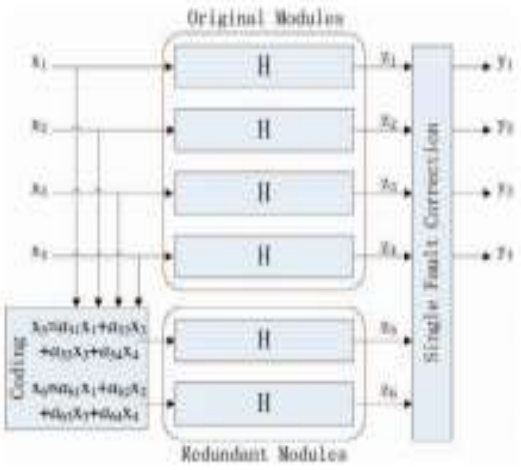

fig. 3. Practical coding scheme to protect four parallel filters

This simplified checking can be also derived by noting that p1 and p2 simply check if the output values of the redundant filters (z5 and z6) match the value reconstructed using the outputs of the original filters (z1, z2, z3, and z4). Therefore, in the absence of errors, both p1 and p2 will be zero. From the coding matrix, for nonzero p1 and p2, it becomes clear that an error on the first filter will make p1 = p2 as both a51 and a61 are one. An error on the second filter will make 2∗p1 = p2 as a52 = 1 and a62 = 2 and so on. Therefore, the vector given by (18) can be used to identify the filter in error using the mapping shown in Table II. For four nonzero values, the faulty filter between the fifth and sixth filters can be identified by checking whether p1 or p2 is nonzero. In fact, to check against Table II, the following simplified vector with lower complexity can be used:

$$\bar{e}_{\text{sim}} = \begin{bmatrix} p_1 - p_2 \\ 2p_1 - p_2 \\ 3p_1 - p_2 \\ 4p_1 - p_2 \end{bmatrix}. \quad (19)$$

This provides a simple implementation as only three multiplications are needed and two of them are by powers of two and only require a shift. Another three multiplications are needed to compute p2. Thus, in total, the scheme requires only six multiplications. This shows that the error location logic can be efficiently implemented.

IPHV8I2043X
International Journal Of Advanced Research and Innovation -Vol.8, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

Finally, when an error is detected, it can be corrected by recomputing the affected filter output using z5 and the remaining original filter outputs. For example, for an error in filter 1, correction is implemented as

$$z_1^{\text{corrected}} = z_5 - (z_2 + z_3 + z_4). \quad (20)$$

The second case study is similar, but eight filters have to be protected. The coding matrix used is shown in (21). The structure is the same as in the first case study and so is the number of redundant filters; as in the proposed scheme, it does not depend on the number of filters. This is a clear advantage over the previous ECC scheme in which the number of redundant filters grows as the number of filters to protect increases. Following the same procedure for the calculation of the detection matrix C, the form of the check vector to detect and locate errors [see (21)] is similar to that of the first case study, but in this case, seven multiplications are needed to compute the check vector and another seven to compute z10. It can be seen that the same matrix structure and correction procedure can be applied to protect any given number of parallel filters. The scheme can detect and correct all errors that affect a single filter. This is the same error correction capability as that of the previous ECC scheme in [13]. Equation (21) is as follows:

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{bmatrix} \quad e_{\text{sim}} = \begin{bmatrix} p_1 - p_2 \\ 2p_1 - p_2 \\ 3p_1 - p_2 \\ 4p_1 - p_2 \\ 5p_1 - p_2 \\ 6p_1 - p_2 \\ 7p_1 - p_2 \\ 8p_1 - p_2 \end{bmatrix} \quad (21)$$

$$\begin{cases} p_1 = z_9 - (z_1 + z_2 + z_3 + z_4 + z_5 + z_6 + z_7 + z_8) \\ p_2 = z_{10} - (z_1 + 2z_2 + 3z_3 + 4z_4 + 5z_5 + 6z_6 + 7z_7 + 8z_8). \end{cases} \quad (22)$$

To summarize, the computational complexity needed to implement error detection and correction in the proposed method is mainly due to the two redundant filter modules. The other operations needed to compute the check vector and correct the errors are simpler, although slightly more complex than those in the ECC-based scheme presented.

IV. OUTPUT WAVEFORMS

Fig 4:output waveform1

Fig 5: output waveform 2

V.CONCLUSION

A new method to implement fault-tolerant parallel filters has been presented in this brief. The proposed scheme exploits the linearity of filters to implement an error correction mechanism. In particular, two redundant filters whose inputs are linear combinations of the original filter inputs are used to detect and locate the errors. The coding of those linear combinations was formulated as a general problem to then show how it can efficiently be implemented. The practical implementation was illustrated with two case studies that were evaluated for an FPGA implementation and compared with a previously proposed technique. That technique relies on the use of ECCs such that each filter is treated as a bit in the ECC. The results show that the proposed scheme outperforms the ECC technique (lower costs achieving similar fault-tolerant capability). Therefore, the proposed technique can be useful to implement faulttolerant parallel filters. Future work will consider applying the scheme to parallel filters that have the same input signal but different impulse responses.

REFERENCES

[1] P. P. Vaidyanathan, Multirate Systems and Filter Banks, Englewood Cliffs, N.J., USA: Prentice Hall, 1993.

[2] A. Sibille, C. Oestges and A. Zanella, MIMO: From Theory to Implementation, New York, NY, USA: Academic, 2010.

[3] N. Kanekawa, E. H. Ibe, T. Suga and Y. Uematsu, Dependability in Electronic Systems: Mitigation of

Hardware Failures, Soft Errors, and ElectroMagnetic Disturbances, New York, NY, USA: Springer Verlag, 2010.

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

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

[6] A. Reddy and P. Banarjee "Algorithm-based fault detection for signal processing applications," IEEE Trans. Comput., vol. 39, no. 10, pp. 1304–1308, Oct. 1990.

[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 subword-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," IEEE Trans. Circuits Syst. II: Exp. Briefs, vol. 58, no. 8, pp. 512–516, Aug. 2011.

Author's profile

[1]. DASRI KIRAN KUMAR, Now Studying M.Tech in VLSI and Embedded Systems stream in Department of Electronics and Communication Engineering in DRK Institute of Science & Technology Bowrampet(V),Telangana, India.

[2]. M.SIVANARAYANA Presently Working as Associate Professor in Department of ECE from DRK Institute of Science & Technology, Bowrampet (V), Hyderabad, Telangana, India.