IPHV8I2040X
International Journal Of Advanced Research and Innovation -Vol.8, Issue .II
*ISSN Online: 2319 – 9253*
*Print: 2319 – 9245*

# ERROR DETECTION AND CORRECTION OF PARALLEL FFTS USING PARSEVALS CHECK

**[#1]KUCHIPUDI NARASIMHA RAO, M.Tech student,**
**[#2]MANDAVA VINUSHA, Assistant Professor,**
**Dept of ECE,**
**DRK INSTITUTE OF SCIENCE AND TECHNOLOGY, BOWRAMPET (V), TS, INDIA.**

**ABSTRACT:** Soft errors are the most common problems faced now days. They could be corrected using software functions .Communications and signal processing systems are no exceptions to this trend. For some applications, an interesting option is to use algorithmic-based fault tolerance (ABFT) techniques that try to exploit the algorithmic properties to detect and correct errors. Signal processing and communication applications are well suited for ABFT. One example is fast Fourier transforms (FFTs) that are a key building block in many systems. Several protection schemes have been proposed to detect and correct errors in FFTs. Among those, probably the use of the Parseval or sum of squares checks is the most widely known. In modern communication systems, it is increasingly common to find several blocks operating in parallel. So we implement fault tolerant parallel FFts using error correction codes (redundant bits) and sum of squares (parsevals check).

*Keywords: ABFT, parseval check ,sum of squares.*

## 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 [10] or in multiple-input multiple-output (MIMO) communication systems . In particular, MIMO orthogonal frequency division modulation (MIMO-OFDM) systems use parallel iFFTs/FFTs for modulation/demodulation. MIMO-OFDM is implemented on long-term evolution mobile systems [8] and also on WiMax [9]. 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

IPHV8I2040X
International Journal Of Advanced Research and Innovation -Vol.8, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

independently. This has been studied for digital filters initially in where two filters were considered. More recently, a general scheme based on the use of error correction codes (ECCs) has been proposed . 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 transforms (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 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

The two proposed techniques provide new alternatives to protect parallel FFTs that can be more efficient than protecting each of the FFTs independently. The proposed schemes have been evaluated using FPGA implementations to assess the protection overhead. The results show that by combining the use of ECCs and Parseval checks, the protection overhead can be reduced compared with the use of only ECCs as proposed. Fault injection experiments have also been conducted to verify the ability of the implementations to detect and correct errors

## II. EXSISITNG METHOD

### 2.1 Fast Fourier Transform

Fast Fourier Transform (FFT) algorithm converts a signal from time domain into a sequence in the frequency domain [8]. Fast Fourier transforms are widely used for many applications which include engineering, science, and mathematics. It computes transformations through DFT matrix. The FFT operation starts with decomposing N-point time domain signal and calculating N frequency spectra and finally forming a single spectrum.

### 2.2 The Discrete Fourier Transform (DFT)

Discrete Fourier Transform (DFT) is an important unit in many communication applications like OFDM, etc. DFT is also measured as one of the tools to act upon frequency analysis of discrete time signals. The Discrete Fourier Transform is a continuous Fourier transform for the use of discrete functions. Given a real sequence as the input, the DFT outputs them as a sequence of complex numbers. The mathematical representation of the transform is If an N – point DFT is implemented directly, the necessity of arithmetic units is of the order of O(N2) that is N2 multiplications and N (N-1) additions. Thus FFT is used for designing the DFT. Depending on inputs being real or complex, the design of adders and multipliers are formed.

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-\frac{j2\pi kn}{N}}, n = 0,1, \ldots\ldots N-1 \qquad (1)$$

The reduction of computational complexity algorithm for DFT is made possible by using a divide and conquers approach. This approach decomposes a larger DFT into smaller one forming a collective FFT algorithm. Let us consider N-point DFT. It is one of the well-organized ways to implement Discrete Fourier Transform (DFT) due to its compact use of arithmetic blocks. The FFT and inverse FFT of an N point signals are given below.

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn} \qquad (2)$$

$$x(n) = \frac{1}{N} \sum_{n=0}^{N-1} X(k) W_N^{-kn} \qquad (3)$$

Where

$$W_N^{kn} = e^{-j\frac{2\pi}{N}kn}$$

Basic concept of 4-points DIF FFT circuit which applies Radix-2 architecture is in Figure 1 shows calculation signal flow graph about discrete Fourier coefficient of N=4. Here W40, W41 are the twiddle factors of the four point Fast Fourier Transform. Note, under the arrow of is subtraction; twiddle factor at the top of line is multiplication. In butterfly processing element that is shown red line make be corresponding to next block in the slide diagram. Another butterfly processing element colors are same. This is the 4 points FFT circuit for parallel input. Generally, FFT analyzes an input signal sequence by using decimation-in-frequency (DIF) or decimation-intime (DIT) decomposition to design an efficient signalflow graph (SFG). Here, the paper work focuses DIF decomposition because it matches with various pipelined designs. x(0), x(1), x(2) and x(3) are the input time domain signals with 1, -1 and –j as the

IPHV8I2040X
International Journal Of Advanced Research and Innovation -Vol.8, Issue .II
ISSN Online: 2319 – 9253
Print: 2319 – 9245

twiddle factors producing X(0), X(1), X(2) and X(3) as the frequency domain outputs.

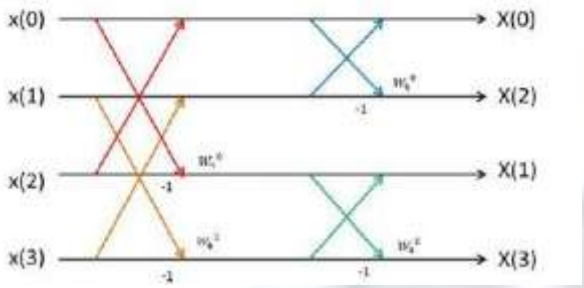


FIG1:SIGNAL FLOW GRAPH

## III. FFTS USING ERROR CORRECTION

The starting point for our work is the protection scheme based on the use of ECCs that was presented in [9] for digital filters. This scheme is shown in Fig. 1. In this example, a simple single error correction Hamming code [18] is used. The original system consists of four FFT modules and three redundant modules is added to detect and correct errors. The inputs to the three redundant modules are linear combinations of the inputs and they are used to check linear combinations of the outputs. For example, the input to the first redundant module is

$x5 = x1 + x2 + x3$ …….(1)

and since the DFT is a linear operation, its output z5 can be used to check that

$z5 = z1 + z2 + z3$…….. (2)

This will be denoted as c1 check. The same reasoning applies to the other two redundant modules that will provide checks c2 and c3. Based on the differences observed on each of the checks, the module on which the error has occurred can be determined. The different patterns and the corresponding errors are summarized in Table I. Once the module in error is known, the error can be corrected by reconstructing its output using the remaining modules. For example, for an error affecting z1, this can be done as follows:

$z1c[n] = z5[n] - z2[n] - z3[n]$…… (3)

| $c_1 c_2 c_3$ | Error Bit Position |
|---|---|
| 0 0 0 | No error |
| 1 1 1 | $z_1$ |
| 1 1 0 | $z_2$ |
| 1 0 1 | $z_3$ |
| 0 1 1 | $z_4$ |
| 1 0 0 | $z_5$ |
| 0 1 0 | $z_6$ |
| 0 0 1 | $z_7$ |

**ERROR LOCATION IN THE HAMMING CODE**

of the number of original FFTs. For example, to protect four FFTs, three redudant FFTs are needed, but to protect eleven, the number of redundant FFTs in only four. This shows how the overhead decreases with the number of FFTs.

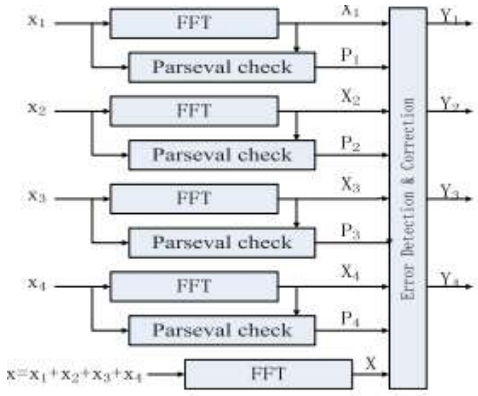


**Fig. 2. Parity-SOS (first technique) fault-tolerant parallel FFTs.**

In Section I, it has been mentioned that over the years, many techniques have been proposed to protect the FFT. One of them is the Sum of Squares (SOSs) check [4] that can be used to detect errors. The SOS check is based on the Parseval theorem that states that the SOSs of the inputs to the FFT are equal to the SOSs of the outputs of the FFT except for a scaling factor. This relationship can be used to detect errors with low overhead as one multiplication is needed for each input or output sample (two multiplications and adders for SOS per sample).

For parallel FFTs, the SOS check can be combined with the ECC approach to reduce the protection overhead. Since the SOS check can only detect errors, the ECC part should be able to implement the correction. This can be done using the equivalent of a simple parity bit for all the FFTs. In addition, the SOS check is used on each FFT to detect errors. When an error is detected, the output of the parity FFT can be used to correct the error. This is better explained with an example. In Fig. 2, the first proposed scheme is illustrated for the case of four parallel FFTs. A redundant (the parity) FFT is added that has the sum of the inputs to the original FFTs as input. An SOS check is also added to each original FFT. In case an error is detected (using P1, P2, P3, P4), the correction can be done by recomputing the FFT in error using the output of the parity FFT (X) and the rest of the FFT outputs. For example, if an error occurs in the first FFT, P1 will be set and the error can be corrected by doing

$X1c = X - X2 - X3 - X4$……
(4)

This combination of a parity FFT and the SOS check reduces the number of additional FFTs to just one and

may, therefore, reduce the protection overhead. In the following, this scheme will be referred to as parity-SOS (or first proposed technique).

Another possibility to combine the SOS check and the ECC approach is instead of using an SOS check per FFT, use an

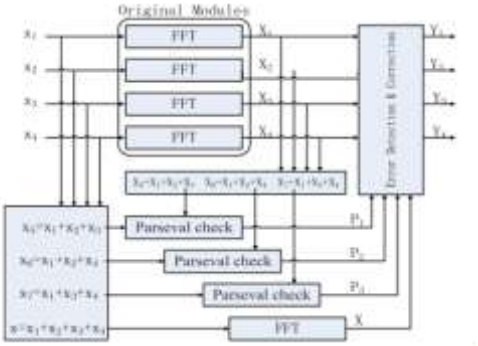


Fig. 3. Parity-SOS-ECC (second technique) fault-tolerant parallel FFTs.

| | FFTs | SOS checks |
|---|---|---|
| ECC | $1+\log_2(k)$ | 0 |
| Parity-SOS | 1 | $k$ |
| Parity-SOS-ECC | 1 | $1+\log_2(k)$ |

**TABLE II**

**OVERHEAD OF THE DIFFERENT SCHEMES TO PROTECT k FFTs**

ECC for the SOS checks. Then as in the parity-SOS scheme, an additional parity FFT is used to correct the errors. This second technique is shown in Fig. 3. The main benefit over the first paritySOS scheme is to reduce the number of SOS checks needed. The error location process is the same as for the ECC scheme in Fig. 1 and correction is as in the parity-SOS scheme. In the following, this scheme will be referred to as parity-SOS-ECC (or second proposed technique).

The overheads of the two proposed schemes can be initially estimated using the number of additional FFTs and SOS check blocks needed. This information is summarized in Table II for a set of k original FFT modules assuming k is a power of two. It can be observed that the two proposed schemes reduce the number of additional FFTs to just one. In addition, the second technique also reduces the number of SOS checks. In Section III, a detailed evaluation for an FPGA implementation is discussed to illustrate the relative overheads of the proposed techniques.

In all the techniques discussed, soft errors can also affect the elements added for protection. For the ECC technique, the protection of these elements was discussed in [17]. In the case of the redundant or parity FFTs, an error will have no effect as it will not propagate to the data outputs and will not trigger a correction. In the case of SOS checks, an error will trigger a correction when actually there is no error on the FFT. This will cause an unnecessary correction but will also produce the correct result. Finally, errors on the detection and correction blocks in Figs. 2 and 3 can propagate errors to the outputs. In our implementations, those blocks are protected with TMR. The same applies for the adders used to compute the inputs to the redundant FFTs in Fig. 1 or to the SOS checks in Fig. 3. The triplication of these blocks has a small impact on circuit complexity as they are much simpler than the FFT computations.

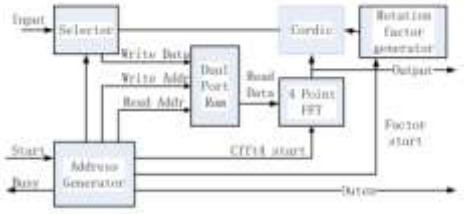


Fig. 4. Architecture of the FFT implementation.

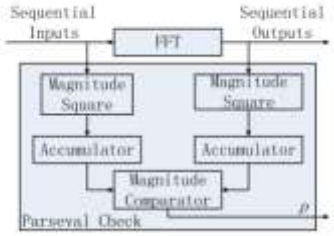


Fig. 5. Implementation of the SOS check.

errors that are actually corrected. This means that an evaluation has to be done both in terms of overhead and error coverage.

## IV. OUTPUT RESULTS

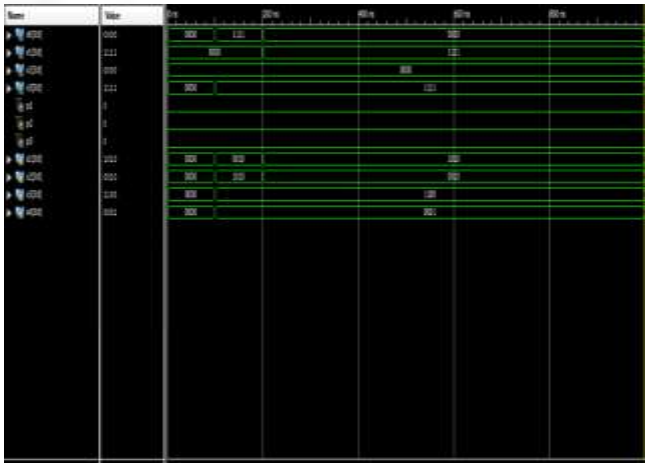

**FIG 6:OUTPUT OF FAULT TOLERANT PARALLEL FFT**

## V.CONCLUSION

The protection of parallel FFTs implementation against soft errors has been studied. Two techniques have been proposed and evaluated. The proposed techniques are based on combining an existing ECC approach with the traditional SOS check. The SOS checks are used to detect and locate the errors and a simple parity FFT is used for correction. The detection and location of the errors can be done using an SOS check per FFT or alternatively using a set of SOS checks that form an ECC.The proposed techniques have been evaluated both in terms of implementation complexity and error detection capabilities. The results show that the second technique, which uses a parity FFT and a set of SOS checks that form an ECC, provides the best results in terms of implementation complexity.

REFERENCES

[1] N. Kanekawa, E. H. Ibe, T. Suga, and Y. Uematsu, Dependability in Electronic Systems: Mitigation of Hardware Failures, Soft Errors, and Electro-Magnetic Disturbances. New York, NY, USA: Springer-Verlag, 2010.

[2] R. Baumann, "Soft errors in advanced computer systems," IEEE Des. Test Comput., vol. 22, no. 3, pp. 258–266, May/Jun. 2005.

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

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

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

[6] S. Pontarelli, G. C. Cardarilli, M. Re, and A. Salsano, "Totally fault tolerant RNS based FIR filters," in Proc. 14th IEEE Int. On-Line Test Symp. (IOLTS), Jul. 2008, pp. 192–194.

[7] B. Shim and N. R. 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.

[8] E. P. Kim and N. R. Shanbhag, "Soft N-modular redundancy," IEEE Trans. Comput., vol. 61, no. 3, pp. 323–336, Mar. 2012.

[9] J. Y. Jou and J. A. Abraham, "Fault-tolerant FFT networks," IEEE Trans. Comput., vol. 37, no. 5, pp. 548–561, May 1988.

[10] S.-J. Wang and N. K. Jha, "Algorithm-based fault tolerance for FFT networks," IEEE Trans. Comput., vol. 43, no. 7, pp. 849–854, Jul. 1994.

## AUTHOR'S PROFILE

**[1].KUCHIPUDI NARSIMHA RAO ,** 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].MANDAVA VINUSHA** Presently Working as Assistant Professor in Department of ECE from DRK Institute of Science & Technology, Bowrampet (V), Hyderabad, Telangana, India.