Advanced Technology using Vedic Multiplier for Efficient Area in Parallel FFTs

[1]KALLURI MANIDEEPTHI, [2]Mr. K. NIRANJAN REDDY, [3]Mrs. N. NEELIMA
[1]PG Scholar, Dept. of ECE (VLSI), CMR Institute of Technology, Kandlakoya (v), Hyderabad, TS, India.
[2]Associate Professor, Dept. of ECE, CMR Institute of Technology, Kandlakoya (v), Hyderabad, TS, India.
[3]Associate Professor, Dept. of ECE, CMR Institute of Technology, Kandlakoya (v), Hyderabad, TS, India.

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. Fast Fourier transforms (FFTs) plays a key role in many communication and signal processing systems. 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. The proposed work exploits a technique to implement fault tolerance parallel FFT with reduced low complexity of circuit area and power. To reduce the area we used Vedic Multiplier in place of Booth's Multiplier.
Low power and area efficient adder and multiplier have always been a fundamental requirement of high performance processors and systems. To design a Fast Fourier Transform (FFT) its speed greatly depends on the multiplier and adder. Carry select adder is one of the fast adders used in many processors to increase their speed with reduced size and low power, reduced complexity. Carry Select Adder (CSLA) that uses multiple pairs of Ripple Carry Adder (RCA) uses moderate delay, larger area and high power. Vedic multiplier is an ancient form of multiplication which performs the multiplication operation faster. It uses 16-sutras. Here we used Urdhva Tiryakbhyam Sutra, to reduce the number of steps in the multiplication method. So the time, area and delay are reduced. Vedic multiplier and carry select adder can be used to design a Fast Fourier Transform (FFT) which produces an output at a very faster time and the delay, area can be reduced.

Keywords: FFTs, Vedic Multiplier, Urdhva Tiryakbhyam Sutra, Xilinx

1. INTRODUCTION

The CMOS technology scaling has made today's designs more susceptible to radiation induced soft errors. Soft errors can alter the logical output of a circuit node creating an error that affects the system functionality. The problem becomes more complexity of the soft error rate exponentially increases with that of circuits scaling. Single Event Upsets (SEU) also affects the reliability of the circuit due to variation in set-up and hold time. Various methods have been adopted earlier to mitigate soft errors. Specifically designing libraries used for complex circuit and modified manufacturing process such as the silicon on insulator design also used for minimizing the error probability. Adding redundancy to keep the design free from temporary errors is also adopted in existing designs. Five modular redundancy mitigation techniques recover the Error Module used to overcome the unintended behaviour of the system by adding redundancy. It adds two identical designs and a voter along with the original design to produce correct results. It increased the area overhead which not suitable for complex designs. Some techniques have been introduced to eliminate this problem by making changes in the algorithm (ABFT). Based on algorithms, the use of the Parseval theorem or sum of squares check is one the frequently used method. The SOS check states that the SOSs of the inputs to the FFT are equal to the SOSs of the outputs of the FFT. This correlation can be used to detect errors while using multiple FFTs.
Triple Modular Redundancy (TMR) and Hamming Codes have been used to protect different circuits against Single Event Upsets (SEUs). In this paper, the use of a Novel Hamming approach on FIR Filters is studied and implemented in order to provide low complexity, reduce delay and area efficient protection techniques for higher bits data. A novel Hamming code is proposed in this paper, to increase the efficiency of higher data bits. In this paper, they have proposed technique used to demonstrate, how the lot of overhead due to interspersing the redundancy bits, their subsequent

removal, pad to pad delay in the decoder and consumption of total area of FIR filter for higher bits are reduced. These are based on the novel hamming code implementation in the FIR filter instead of conventional hamming code used to protect FIR filter. In this scheme hamming code used for transmission of 7-bit data item.

Detecting and correcting errors such as critical reliability are difficult in signal processing which increases the use of fault tolerant implementation. In modern signal processing circuits, it is common to find several filters operating in parallel. Proposed is an area efficient technique to detect and correct single errors occurring in pairs of parallel filters that have either the same input data or the same impulse response. There are number of techniques can be used to protect a circuit from errors. Those ranges from modifications in the manufacturing process of circuits to reduce the number of errors do not affect system functionality. The general technique of adding redundancy known as Triple Modular Redundancy, but it requires area and power three times the unprotected filter. Sometime they may not be acceptable in some application in terms of low power and design requirements. So the idea is generalized to show that parallel filters can be protected using error correction codes (ECCs) in which each filter is the equivalent of a bit in a traditional ECC. This new scheme allows more efficient protection when the number of parallel filters is large.

In Fig.1 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

$$X1c=X-X2-X3-X4. \text{ --------} \rightarrow 1$$

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).

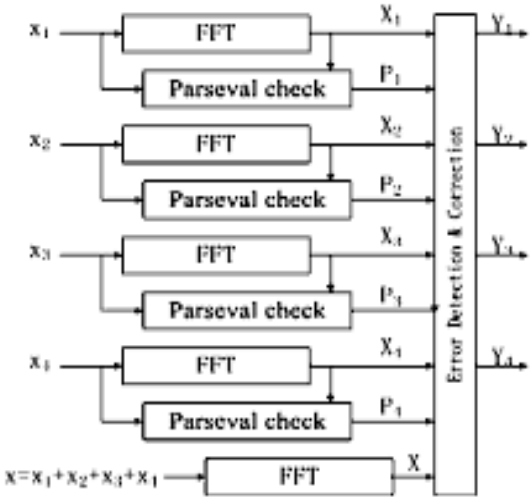
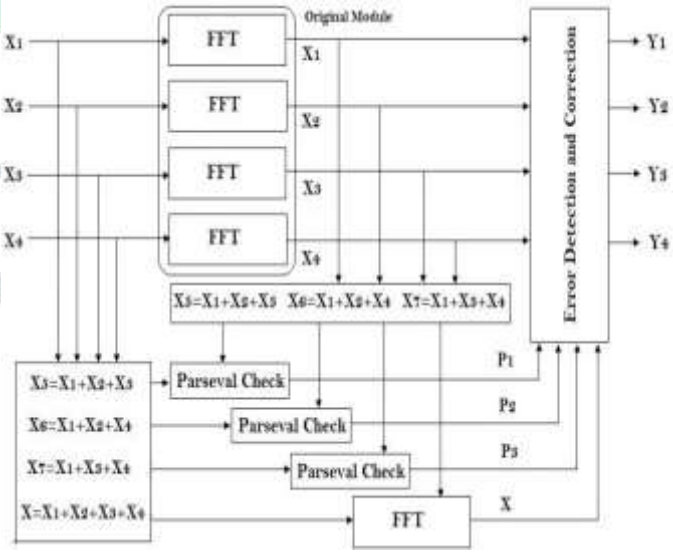


Fig.1. Parity-SOS (first technique) fault-tolerant parallel FFTs.

Another possibility to combine the SOS check and the ECC approach is instead of using an SOS check per FFT, use an 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. 2. The main benefit over the first parity SOS scheme is to reduce the number of SOS checks needed. The error location process is the same as for the ECC scheme 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).

Fig.2. Parity-SOS-ECC (second technique) fault-tolerant parallel FFTs



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.1 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.

Types	FFTs	SOS Checks
PARITY-SOS	1	K
PARITY–SOS-ECC	1	$1+\log_2(k)$

Table.1. Overhead of Different Techniques

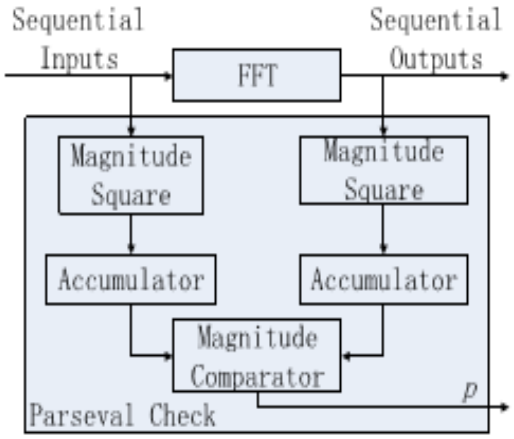


Fig.3. Implementation of SOS Check

Figure 3 shows implementation architecture of SOS Check. The sequential input and output of the 4-point FFT is fed to the magnitude square block. The magnitude comparator compares the input and output for verifying the FFT output. If both are equal then the FFT is soft error free.

A new technique with Vedic multiplier in place of Booth multiplier, Thus it will decrease area overhead and delay due to the FFTs. So that it will improve the performance efficiency and low area overhead. Below section will describe about Vedic multiplier-Urdhva Tiryakbhyam sutra.

2. VEDIC SUTRA- URDHVATIRYAKBHYAM:

For suggested framework we have a tendency on region unit estimation enter snake Unit, at present it will be supplanted Eventually Tom's perusing holy content multiplier element. By completing this we have the capacity will get less energy consumption, exactness and lessened delay. Those sixteen holy content Sutras apply to Furthermore shade almost each limb from claiming math. They apply significantly to propelled issues directing, including an oversized assortment from claiming scientific operations. Around these sutras, Urdhva Tiryakbhyam Sanskrit writing may be that those best for acting duplication. The utilization from claiming this Sanskrit writing will be stretched out on double duplication also. This Sanskrit writing interprets to "Vertical Furthermore crosswise". It uses singularly legitimate what's more operation, 0. 5 adders Furthermore full adders should perform duplication wherever that incomplete stock region unit produced in the recent past real duplication. This protects a considerable amount of time interim. What will be a greater amount it's a tough technique for duplication. Think as of two 8-bit numbers, An (a8-a1) Furthermore b (b8-b1) wherever particular case to eight speaks to odds from the any rate critical touch of the vast majority paramount touch. That ultimacy item may be spoken to Toward p (P16-P1). In fig. 4, the regulated procedure for duplication about two 8-bit numbers utilizing Urdhva Tiryakbhyam sutra may be illustrated. Those odds of the amount Furthermore amount region unit diagrammatic Toward specks Also Additionally the 2 approach would speaks to the legitimate Furthermore operation between the odds that gives the halfway item terms. In the average style for Urdhva Tiryakbhyam sutra based mostaccioli number, exclusively full-adders and half-adder region unit utilized for expansion of the halfway items. But, that inclination from claiming full-adder is confined should expansion of singularly three odds at once.

So, an extensive number from claiming phases are obliged on get that last item. Higher request compressors talked about to next area could a chance to be utilized should include more than 3 odds at once (upto 7 bits) what's more Consequently cam wood diminish the intermediate phases.

The multiplier may be dependent upon an algorithm Urdhva Tiryakbhyam (Vertical & Crosswise) about ancient administration Indian Vedic math. Urdhva Tiryakbhyam sutra will be a general duplication equation pertinent wills the greater part cases about duplication. It truly intends "Vertically what's more crosswise". It is dependent upon a novel idea through which the era about at halfway items might be done with that simultaneous expansion for these incomplete results.

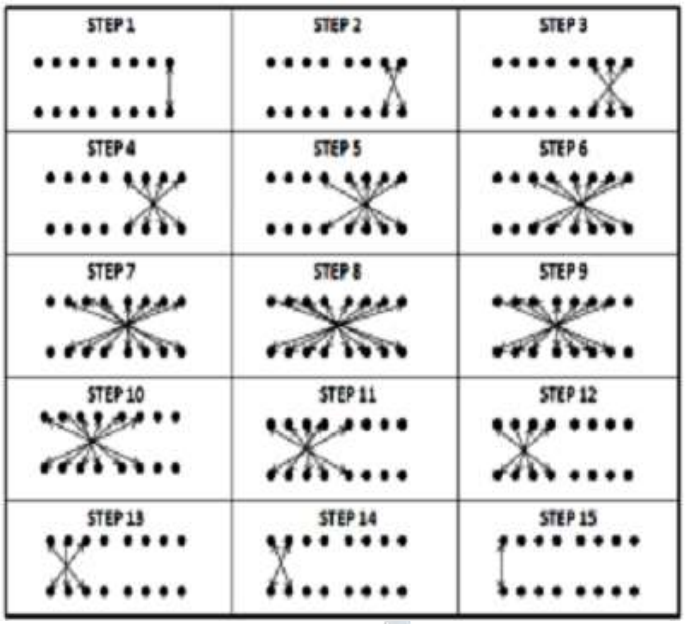


Fig.4. 8-bit binary multiplication using Urdhva Tiryakbhyam Sutra.

The parallelism previously, era of fractional items What's more their summation will be gotten utilizing Urdhawa Triyakbhyam. That net focal point will be that it lessens the need for microprocessors will work during progressively helter skelter clock frequencies. Same time a higher clock recurrence by and large brings about expanded preparing power; its disservice may be that it also builds energy dispersal which brings about higher gadget working temperatures. Toward adopting that Vedic multiplier, microprocessors designers could effectively circumvallated these issues should stay away from calamitous gadget disappointments. Those preparing force of multiplier cam wood undoubtedly a chance to be expanded toward expanding the enter Furthermore yield information transport widths since it need a truly An standard structure. Because of its general structure, it might make effectively design clinched alongside a silicon chip. That multiplier need those focal point that concerning illustration those amount for odds increases, entryway delay What's more zone builds precise gradually as contrasted with different multipliers. Consequently it will be time, space Also force proficient. It will be showed that this structural engineering may be very productive As far as silicon area/speed.

I. Algorithm for 8 X 8 Bit Multiplication Using Urdhva Triyakbhyam (Vertically and crosswise) for two Binary numbers –

$$CP = X0 * Y0 = C$$

$$CP = X1 * Y0 + X0 * Y1 = D$$

$$CP = X1 * Y1 = E$$

```
A =      A7A6A5A4      A3A2A1A0
           X1              X0
B =      B7B6B5B4      B3B2B1B0
         Y1              Y0
                X1  X0
              * Y1  Y0
-------------------------------------------------------
                F E D C
```

3. CP X2 X1 X0 = X2 * Y0 + X0 * Y2 + X1 * Y1 = C
   Y2 Y1 Y0

4. CP X3 X2 X1 X0 = X3 * Y0 + X0 * Y3 + X2 * Y1 + X1 * Y2 = D
   Y3 Y2 Y1 Y0

5. CP X3 X2 X1 = X3 * Y1 + X1 * Y3 + X2 * Y2 = E
   Y3 Y2 Y1

6. CP X3 X2 = X3 * Y2 + X2 * Y3 = F
   Y3 Y2

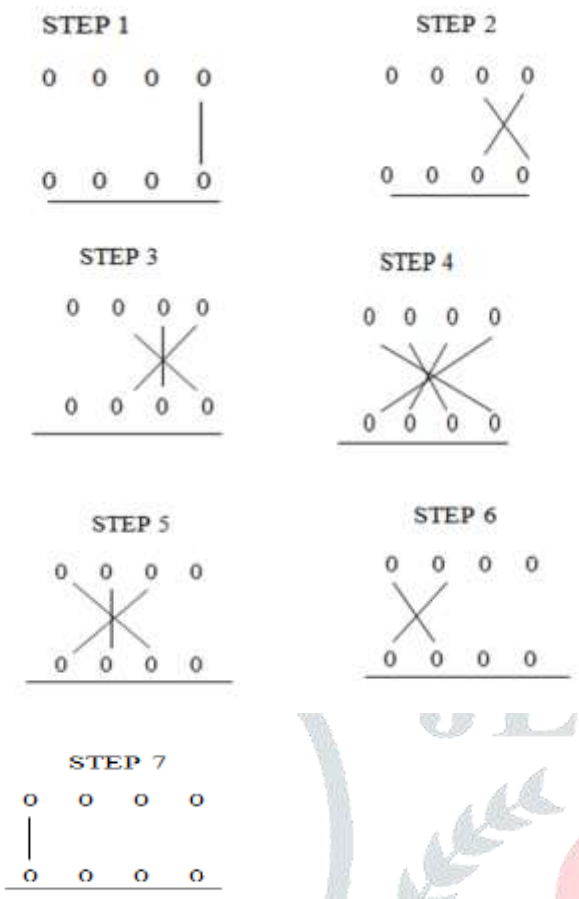
7 CP X3 = X3 * Y3 = G
   Y3

Where CP = Cross Product.

Note: Each Multiplication operation is an embedded parallel 4x4 Multiply module.

To illustrate the multiplication algorithm, let us consider the multiplication of two binary numbers a3a2a1a0 and b3b2b1b0. As the result of this multiplication would be more than 4 bits, we express it as... r3r2r1r0. Line diagram for multiplication of two 4-bit numbers is shown in which is nothing but the mapping of the binary system. For the simplicity, each bit is represented by a circle. Least significant bit r0 is obtained by multiplying the least significant bits of the multiplicand and the multiplier. The process is followed according to the steps shown in Fig4.

II. Line Diagram for Multiplication of Two 4 - Bit Numbers



Firstly, least significant bits are multiplied which gives the least significant bit of the product (vertical). Then, the LSB of the multiplicand is multiplied with the next higher bit of the multiplier and added with the product of LSB of multiplier and next higher bit of the multiplicand (crosswise). The sum gives second bit of the product and the carry is added in the output of next stage sum obtained by the crosswise and vertical multiplication and addition of three bits of the two numbers from least significant position. Next, all the four bits are processed with crosswise multiplication and addition to give the sum and carry. The sum is the corresponding bit of the product and the carry is again added to the next stage multiplication and addition of three bits except the LSB. The same operation continues until the multiplication of the two MSBs to give the MSB of the product. For example, if in some intermediate step, we get 110, then 0 will act as result bit (referred as rn) and 11 as the carry (referred as cn). It should be clearly noted that cn may be a multi-bit number.

Thus we get the following expressions:

r0=a0b0 ----------→2
c1r1=a1b0+a0b1----------→3
c2r2=c1+a2b0+a1b1+a0b2---------- -→4
c3r3=c2+a3b0+a2b1+a1b2+a0b3 --------→5
c4r4=c3+a3b1+a2b2+a1b3 ----------→6
c5r5=c4+a3b2+a2b3 ----------→7
c6r6=c5+a3b3 ----------→8

With c6r6r5r4r3r2r1r0 being the final product. Hence this is the general mathematical formula applicable to all cases of multiplication.

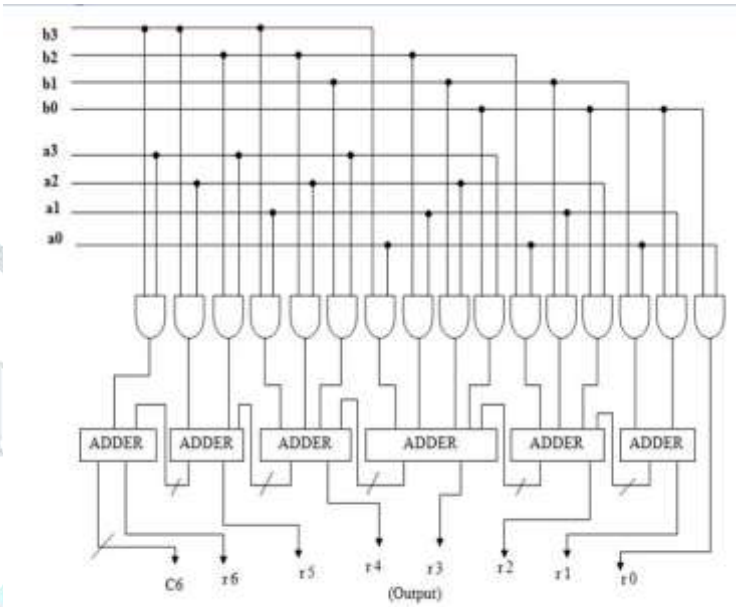


Fig.5. Hardware architecture of the Urdhva Tiryakbhyam multiplier

This hardware design is very similar to that of the famous array multiplier where an array of adders is required to arrive at the final product. All the partial products are calculated in parallel and the delay associated is mainly the time taken by the carry to propagate through the adders which form the multiplication array. Clearly, this is not an efficient algorithm for the multiplication of large numbers as a lot of propagation delay is involved in such cases. To deal with this problem, we now discuss Nikhilam Sutra which presents an efficient method of multiplying two large numbers.

III. Implementation of 32x32 Bits Vedic Multiplier

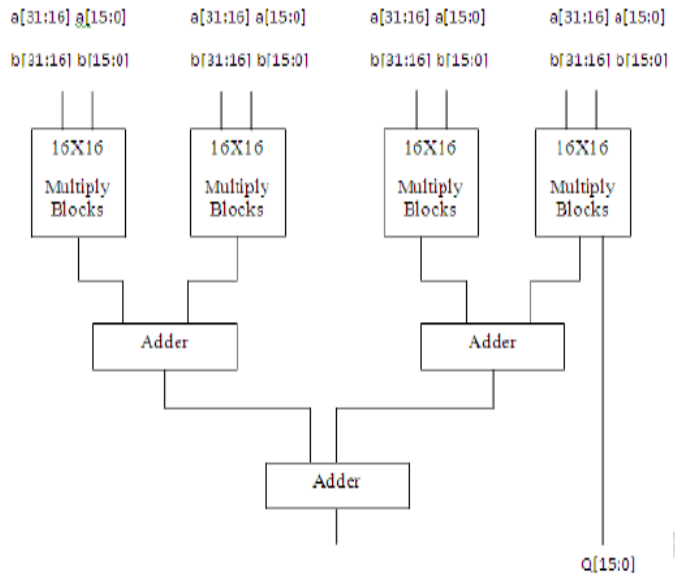


Fig.6. 32x32 Bits Proposed Vedic Multiplier

For Multiplier, first the basic blocks, that are the 2x2 bit multipliers have been made and then, using these blocks, 4x4 block has been made by adding the partial products using carry save adders and then using this 4x4 block, 8x8 bit block, 16x16 bit block and then finally 32 x 32 bit Multiplier as shown in figure 6.

IV. SIMULATION RESULT:

A. First Technique:

The written Verilog HDL Modules have successfully simulated and verified using Modelsim III 6.4b and synthesized using Xilinx ISE 13.2.

Simulation:

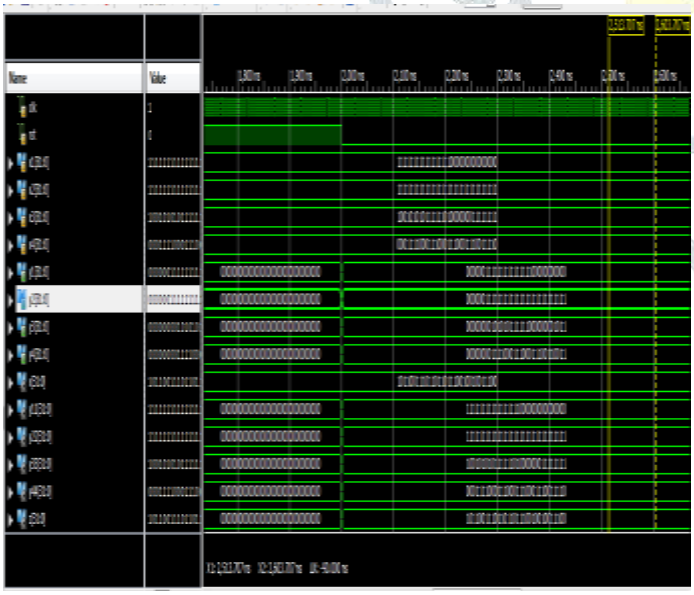


Fig.7. Simulated output wave form (First Technique)

In Fig.7, the inputs are given to the parallel FFTs, and then the outputs generated from those are fed to the parity SOS to detect and correct the errors. 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. Another possibility to combine the SOS check and the ECC approach is instead of using an SOS check per FFT, use an ECC for the SOS checks. Then as in the parity-SOS scheme, an additional parity FFT is used to correct the errors. Here, in place of Booth's multiplier we used Vedic multiplier. In proposed system we tend to area unit measurement Input Adder Unit, currently it is replaced by sacred text multiplier factor. By doing this we are able to get less power consumption, high accuracy and reduced delay.

Synthesis Results:

The created venture is mimicked and checked their usefulness. Once the useful confirmation is done, the RTL display is taken to the union procedure utilizing the Xilinx ISE instrument. In union process, the RTL model will be changed over to the door level netlist mapped to a particular innovation library. Here in this Spartan 3E family, a wide range of gadgets were accessible in the Xilinx ISE apparatus. Keeping in mind the end goal to combination this outline the gadget named as "XC3S500E" has been picked and the bundle as "FG320" with the gadget speed, for example, "- 4".

This design is synthesized and its results were analyzed as follows,

RTL Schematic:

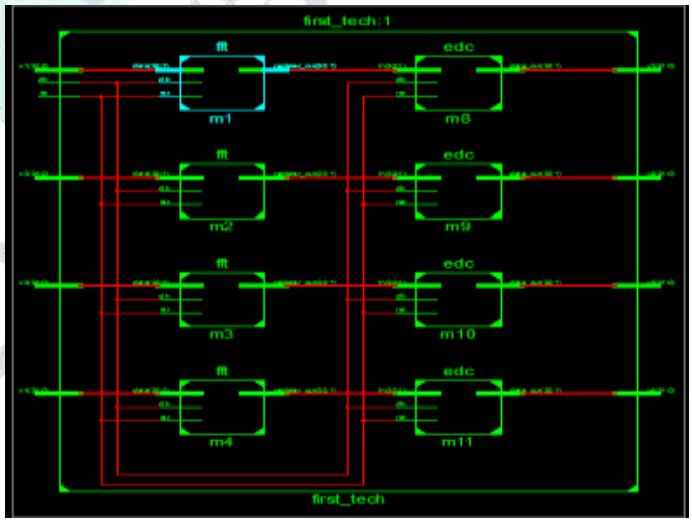


Fig.8. RTL Schematic for Parity-SOS (first technique) Fault-tolerant parallel FFTs by using Vedic multiplier.

The above Fig.8 represents the block diagram for Parity SOS technique. It shows the number of blocks used for the formation of this technique.

Technology Schematic:

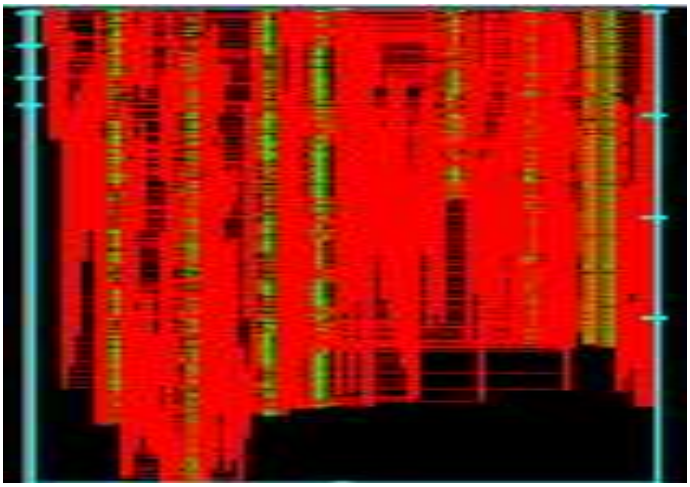


Fig.9. Technology Schematic for Parity-SOS (first technique) Fault-tolerant parallel FFTs by using Vedic Multiplier.

Technology Schematic for Parity SOS technique is the practical implementation of RTL modules. RTL are decomposed into lookup tables in FPGA. Fig.9 shows the technology schematic.

Design Summary:

Table II

Design Summary for the output of First Technique

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	254	4656	5%
Number of Slice Flip Flops	228	9312	2%
Number of 4 input LUTs	409	9312	4%
Number of bonded IOBs	254	232	109%
Number of GCLKs	1	24	4%

The above table II represents the design summary for parity SOS technique gives the area consumed in FPGA device for implementing the technique. Compare to existing technique, in proposed technique area has decreased. It has represented by slices and flip-flops used in the system. In the existed system the slices required 266 but in proposed system only 254 slices are used. So the number slices reduced. i.e area has reduced.

Timing Report:

The below screenshot represents the timing report for Parity SOS technique shows the delay used for the technique**.**

```
Offset:             4.040ns (Levels of Logic = 1)
  Source:           m8/data_26 (FF)
  Destination:      y1<25> (PAD)
  Source Clock:     clk rising

  Data Path: m8/data_26 to y1<25>
                                Gate     Net
    Cell:in->out      fanout   Delay    Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDR:C->Q             1    0.514    0.357  m8/data_26 (m8/data_26)
     OBUF:I->O                 3.169           y1_25_OBUF (y1<25>)
    ----------------------------------------
    Total                      4.040ns (3.683ns logic, 0.357ns route)
                                       (91.2% logic, 8.8% route)
```

B. Second Technique:

Simulation:

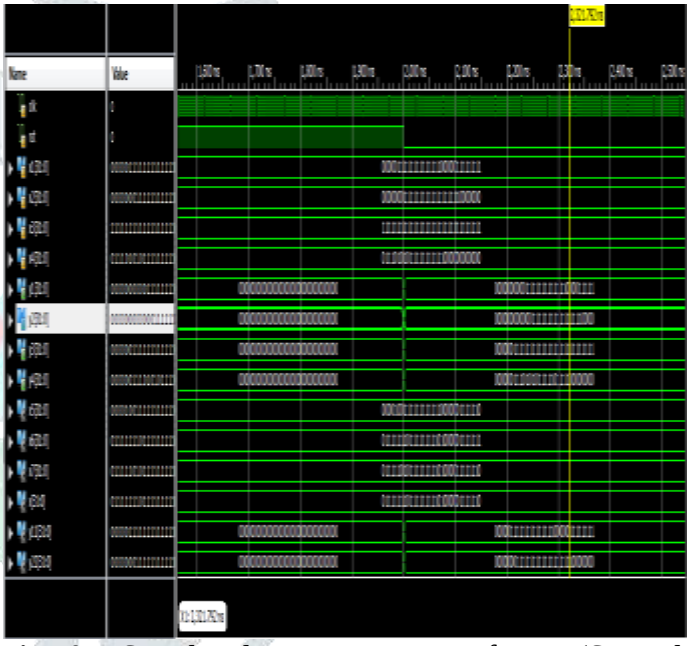


Fig.10. Simulated output wave form (Second Technique)

In Fig.10, The inputs are given to the parallel FFTs, outputs generated from these are fed to the parity SOS-ECC to detect and correct the errors. The main benefit over the first parity SOS scheme is to reduce the number of SOS checks needed. The error location process is the same as for the ECC scheme 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 results show that the second technique, which uses parity FFT and a set of SOS checks that form an ECC, provides the best results in terms of implementation complexity. In terms of error protection, fault injection experiments show that the ECC scheme can recover all the errors that are out of the tolerance range. This project is extended with FFT using Vedic multiplier.

RTL Schematic:

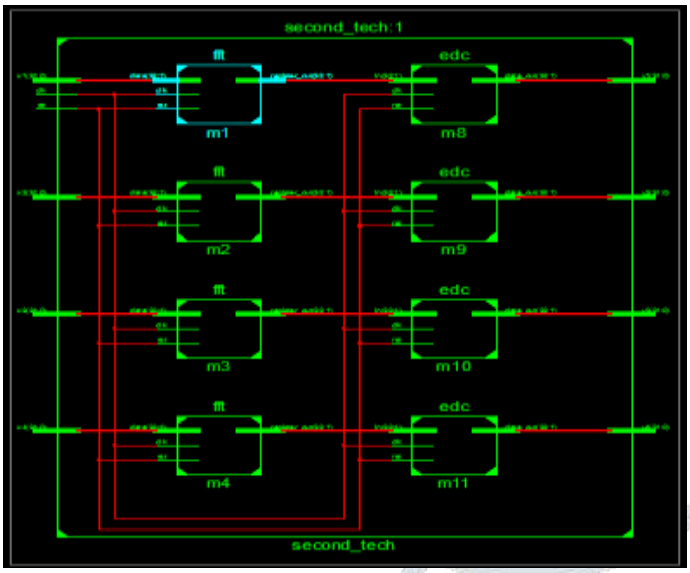


Fig.11. RTL Schematic for Parity-SOS-ECC (second technique) Fault-tolerant parallel FFTs

The above Fig.11 represents the block diagram for Parity SOS-ECC technique. It shows the number of blocks used in formation of this technique.

Technology Schematic:

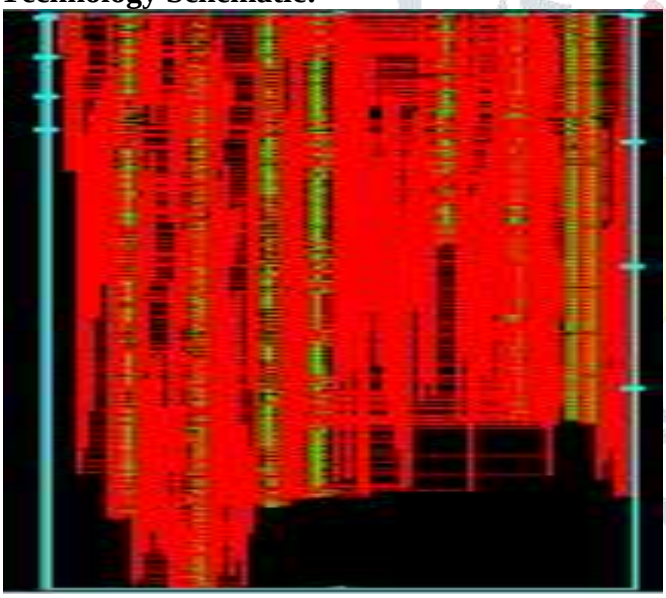


Fig.12. Technology Schematic for Parity-SOS-ECC (second technique) Fault-tolerant parallel FFTs by using Vedic multiplier.

Technology Schematic for Parity SOS-ECC technique is the practical implementation of RTL modules. RTL are decomposed into lookup tables in FPGA. Fig.12 shows the technology schematic.

Design Summary:

The above table IV represents the design summary for parity SOS-ECC technique gives the area consumed in FPGA device for implementing the technique.

TABLE I11

Design Summary for the output of Second Technique

Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slices	254	4656	5%
Number of Slice Flip Flops	208	9312	2%
Number of 4 input LUTs	409	9312	4%
Number of bonded IOBs	254	232	109%
Number of GCLKs	1	24	4%

The above table III represents the design summary for parity SOS technique gives the area consumed in FPGA device for implementing the technique. Compare to existing technique, in proposed technique area has decreased. It has represented by slices and flip-flops used in the system. In the existed system the slices required 266 but in proposed system only 254 slices are used. So the number slices reduced. i.e area has reduced.

Timing Report:

The below screenshot represents the timing report for Parity SOS-ECC technique shows the delay used for the technique.

```
Offset:              4.040ns (Levels of Logic = 1)
  Source:            m8/data_26 (FF)
  Destination:       y1<25> (PAD)
  Source Clock:      clk rising

  Data Path: m8/data_26 to y1<25>
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FDR:C->Q             1    0.514   0.357  m8/data_26 (m8/data_26)
     OBUF:I->O                 3.169          y1_25_OBUF (y1<25>)
    ----------------------------------------
    Total                      4.040ns (3.683ns logic, 0.357ns route)
                                       (91.2% logic, 8.8% route)
```

TABLE IV

Comparison Table:

Area	Existing		Proposed	
	Parity-SOS Technique	Parity-SOS-ECC Technique	Parity-SOS Technique	Parity-SOS-ECC Technique
Slices	266	266	254	254

LUTs	429	429	409	409
Flip-Flops	228	228	228	228

Table.4. Resources usage for the System.

So that the above table shows the slices, LUTs and flip-flops used in the existing and proposed systems. Comparing both the existing and proposed technologies area and delay has reduced.

V. CONCLUSION

In this brief, 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 parity FFT and a set of SOS checks that form an ECC, provides the best results in terms of implementation complexity. In terms of error protection, fault injection experiments show that the ECC scheme can recover all the errors that are out of the tolerance range. The proposed Vedic multiplier circuit using Urdhva Tiryakbhyam Sutra can be implemented in arithmetic and logical units of a DSP processor replacing the traditional circuits. This project is extended with Vedic multiplier. For the further improvement of the multiplier efficiency, we use Vedic multiplier - Urdhva Tiryakbhyam Sutra. By using this, we can improve the functionality of the magnitude square block in the parseval check. This technique can reduce the area and power for parallel filters with large number of FIR filters.

VI. FUTURE SCOPE:

In Future, use of Vedic multiplier in Digital signal processing will be extended. Since SOS-ECC strategy will takes huge territory, control utilization and number of excess channels increments per each piece will diminish by Vedic Multiplier. Fast and secured error correction will happen.

REFERENCES

[1] Zhen Gao, Pedro Reviriego, Zhan Xu, Xin Su, Ming Zhao, Jing Wang, and Juan Antonio Maestro, "Fault Tolerant Parallel FFTs Using Error Correction Codes and Parseval Checks," IEEE Trans. VLSI, vol. 24, no. 2, February 2016.

[2] 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.

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

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

[5] 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.

[6] 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.

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

[8] 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.

[9] P. P. Vaidyanathanm, Multirate Systems and Filter Banks. Englewood Cliffs, NJ, USA: Prentice-Hall, 1993.

[10]A. Sibille, C. Oestges, and A. Zanella, MIMO: From Theory to Implementation. San Francisco, CA, USA: Academic, 2010.

[11]S. Sesia, I. Toufik, and M. Baker, LTE—The UMTS Long Term Evo-lution: From Theory to Practice, 2nd ed. New York, NY, USA: Wiley, Jul. 2011.

[12]P. Reviriego, S. Pontarelli, C. J. 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.

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

[14]R. W. Hamming, "Error detecting and error correcting codes," Bell Syst. Tech. J., vol. 29, no. 2, pp. 147–160, Apr. 1950.

Authors Profile:

Kalluri Manideepthi received her Bachelors' Degree in 2015 from Electronics and Communication Engineering from Swami Vivekananda Institute of Technology, Hyderabad which is affiliated JNTUH, India. Her areas of interest include VLSI design. She is pursuing M.Tech in VLSI System Design from CMR Institute of Technology.

GUIDE:

K Niranjan Reddy is an Associate Professor in Electronics and Communication Engineering Department, CMR Institute of Technology,

Kandlakoya, Medchal, Telangana. He received his Master of Technology (M. Tech) in VLSI System Design degree in 2005. His research interests are in the domain of Low Power VLSI Design.

CO-GUIDE:

N Neelima is an Associate Professor in Electronics and Communication Engineering Department, CMR Institute of Technology, Kandlakoya, Medchal, Telangana. She received her Master of Technology (M. Tech) degree in VLSI System Design in 2006. Her research interests are image processing and Low power VLSI Design.