International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 04 | Apr 2020 www.irjet.net p-ISSN: 2395-0072

Design and Implementation of FFT using Compressor with XOR gate Topology

Ch.V.N.S.L. Meghana [1], A. Hemanth Reddy[1], A. Vamsi Krishna[1], A. Akhil[1], U. Uma Maheswara Rao[2]

[1]Student, B.Tech , ECE department, Bapatla Engineering College, Bapatla, India
[2]Asst Professor, ECE department, Bapatla Engineering College, Bapatla, India

---

Abstract -*One of the popular transformation techniques that was developed to transform one domain signal into other domain signal is FFT. Though it has vast applications in Digital Image Processing, Spectral Analysis, Noise Cancellation , Digital Signal Processing, Telecommunication, and in some transforming data applications like (OFDM), etc … it consumes high power and time taken structures. Researchers also pointing to reduce the power consumption and other factors for the efficient usage of FFT. Butterfly structure plays a major role in the computation of FFT containing Multipliers and Adder units. In this paper we design a power efficient dedicated structures aiming for the implementation of low power Fast Fourier Transform (FFT), using adder compressors, with a new XOR gate topology in which it increasing the efficient use of power, LUTs utilization and time complexity(delay). These structures are design with completed simulation and synthesis using Xilinx Vivado2016.4 design suit.*

Key Words*: *Fast Fourier Transform(FFT), new-XOR gate topology, LUT, Adder Compressor[4][7]

1. INTRODUCTION

FFT is a Fast Fourier Transform FFT is a faster version of past Discrete Fourier Transform (DFT). FFT utilizes some clever algorithms to do the same thing as the DFT, because these algorithms involve less computation due to the efficient usage of Butterflies. These butterflies performs the calculation of complex terms which involves the multiplication of input data by its appropriate coefficient terms, it is a speedy and discrete nature of the FFT that allows us to analyze a signal spectrum , in pitch correction software. We give a relatively short survey of the FFT for arbitrary finite abelian groups.

In 1965, the first ever introduced to implement the FFT is Cooley and Tukey as a signal flow graph. Later by eliminating the weaknesses of Cooley-Tukey[13], CORDIC[9] was introduced with implementing Twiddle Factor. Since this CORDIC using high Latency to overcome this Y.H.Hu introduced Adaptive CORDIC(ACor) method[11] and this process of is goes on. These algorithms are best at their publish of time because of their efficiency. Since the world is moving faster and faster we need faster propagation and the less area of implementation on chip are becoming the highlighting points in todays world.

Power efficient implementation of FFT is useful, because the FFT has wide range of applications and less power requirement is necessary. By using less power architectures will be reflected in efficient utilization of Power. And the modern trend is not to delay the work or for the faster transmission of data. Since the power and faster transmission are relative to each other. For the faster transmission we need high power. But by using the new-XOR gate topology architecture we can attain both efficient utilization in power and less delay. In this paper[5] the implementation of FFT uses the Multi Objective Genetic Algorithm(MOGA) in a 16-point Radix-4 single path delay feedback pipelined Fast Fourier Transform. They operates the processor at an acceptable signal to noise ratio (SNR). Also, it reduces the switching activity. Thus the power consumption requirement is also reduced. Multiplier is one of the basic block of FFT. These are used for twiddle factor multiplications while computing FFT using normal multiplier, the computation speed i.e, the time required for performing computations is more. So the need of low power and high speed multiplier is increasing. Hence, in this paper[12] the normal multiplier is replaced with the Vedic multiplier[8]. It is based on vertical and crosswise structure. The results of Vedic multiplier [8]compared to normal multiplier is reduced by 50% in time complexity. FFT implementation are quite common without estimating the LUT block utilization, path delay, power reduction architectures. One such paper[10] propose architecture, which produces high throughput, smaller area and less latency. This is a Multiplier less architecture, which uses shift and add operations to realize the complex multiplication operations, thus, it increases the clock frequency and reduces the latency with a lesser number of gates when compared with the usual architectures.

Our approach has didactic advantages over the usual ones. The FFT converts a signal from the time domain to frequency domain. It has diverge applications digital recording, sampling, Noise Cancelation, etc.. We are using Adder Compressor[1] architecture which is designed by XOR gates has similar operations with respect to the simple adders by using this we are aiming to reduce the LUT, power, delay are essential parameter when compared to other algorithms or architecture which will be discussed in this report further.

2. IMPLEMENTATION OF FFT

Functionally, the FFT decomposes the set of data to be transformed into a series of smaller data sets to be transformed. Then it decomposes those smaller sets into even smaller sets[2]. At each stage of processing, the results of the previous stage are combined in a special way. Finally, it calculates the DFT of each small data set. DFT can be computed using the below given formula:

$$X[k] = \sum_{n=0}^{N-1} x(n)(W_N^{nk}) \;\; k = 0,1,2 \ldots N-1$$

……(Eq 2.1)

$$W_N^{nk} = e^{-\frac{2\pi jnk}{N}} \;\; k = 0,1,2,\ldots\ldots\text{(Eq 2.2)}$$

A significant disadvantage of this DFT calculation is the computational complexity. For a group of length N, it has a multifaceted nature given as: $O(N^2)$ henceforth it's anything but a productive strategy and here the FFT comes into the image. FFT altogether diminishes the quantity of calculations required for a grouping of length N from $O(N^2)$ to O(N logN) [2] where log is the base-2 logarithm. FFT works by decaying a N point time space signal into N time area flags each made out of a solitary point. Thus the FFT is Computed by using the Eq 2.1 and prominent Twiddle Factors are given by Eq 2.2. The subsequent advance is to ascertain the N recurrence spectra relating to these N time area signals. Ultimately, the N spectra are incorporated into a solitary recurrence range. So we split the N-point information succession into two N/2 point information groupings as mention in below eq 2.3 and eq 2.4. F1(n) and F2(n), comparing to the even-numbered and odd-numbered tests of x(n) respectively.

$F_1(n)=x(2n)$ …….(Eq 2.3)

F_1 is the even data se

$F_2(n) =x(2n+1)$ …….(Eq 2.4)

F_2 is the odd data set

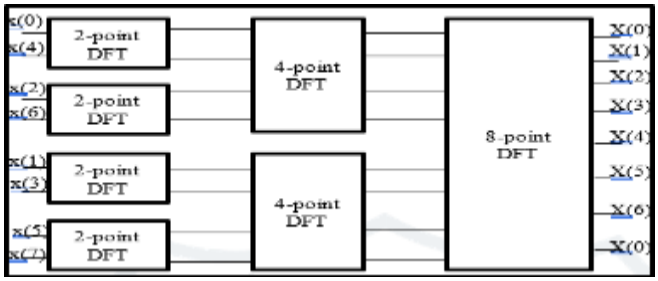

Fig :2.1 8-point Radix-2 FFT using Butterfly structure

The fig.2.1 FFT, we have to break an informational collection data into smaller informational collections and afterward process the DFT of every little set. Fig.1 portrays the execution of 8-point DFT. We see that calculation is acted in three phases, beginning with the calculation of four 2-point DFTs, at that point two 4-point DFTs, lastly one 8-point DFT.

3. ADDER COMPRESSOR

Adder Compressors[1][4][7] are broadly utilized in quick and low force multiplier designs, and they are fundamentally created by Exclusive-or (XOR) and Multiplexer (MUX) gates. So, Compressor is also termed as new-XOR gate topology. A compressor may be a device which is usually utilized in multipliers to scale back the operands while adding terms of partial products. A typical M-N compressor takes M equally weighted input bits and produces N-bit binary number. The most widely recognized structures introduced in writing are 3:2, 4:2, 5:2, and 7:2 compressors [7]. In any case, for the butterfly structures involved right now, the two first referenced compressor could be utilized. These rudiments structures can be utilized as building square to create higher order Compressor.

One of the main speed enhancement techniques utilized in modern digital circuits is that the ability to feature numbers with minimal carry propagation. The basic idea is that three numbers are often reduced to 2, during a 4:2 compressor, by doing the addition while keeping the carries and the sum separate[6].This means that all of the columns can be added in parallel without relying on the result of the previous column, creating a two output "adder" with a time delay that is independent of the size of its inputs. The sum and carry can be recombined in a normal addition to form the correct result. This process may seem more complicated and pointless, but the power of this technique is that any amount, number of additions can be added together in this manner. It is only the ultimate recombination of the ultimate carry and sum that needs a carry propagating addition.

3.1 4:2 COMPRESSOR

The simplest and therefore the most generally used compressor is that the 3-2 compressor which is additionally referred to as a full adder. It has Three inputs to be summed up and provides two outputs. Similarly, a 4-2 compressor also can be built from two Cascaded 3-2 compressor circuits. The conventional implementation of a 4-2 compressor consists of two serially connected full adders, as shown in fig.3.1 Different structures of 4-2 compressors are reported in literature and these are governing by the essential equation as Follows:

The improved structure of a 4-2 compressor circuit, using multiplexers (MUX) and XOR gates, is introduced in Fig.1. This structure presents a decreased basic way, where the maximum delay is given by three XOR gates[1].The last entirety aftereffect of the 4-2 compressor

is given by combination of Sum, Cout and Carry terms, and the equations Eq 4.1, Eq 4.2 and Eq 4.3 respectively[3].

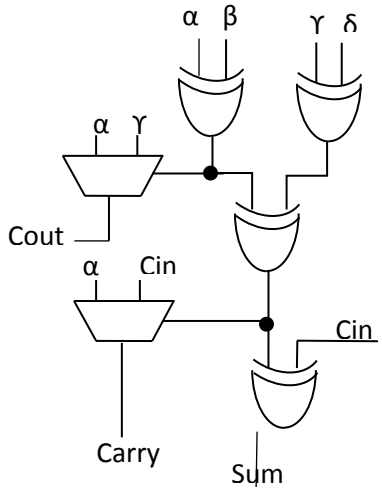

Fig 3.1 Block Diagram Of 4-2 compressor

The expressions of Sum, Carry and Cout are given as follows[3]

Sum = α ⊕ β ⊕ϒ ⊕ δ ⊕ Cin ……(Eq 4.1)

Cout = (α ⊕ β)*ϒ + (α ⊕ β)'* α …….(Eq 4.2)

Carry = (α ⊕ β ⊕ϒ ⊕ δ)*Cin+

((α ⊕ β ⊕ϒ ⊕ δ)')* δ δ …….(Eq 4.3)

One of the main speed enhancement techniques utilized in modern digital circuits is that the ability to feature numbers with minimal carry propagation. The basic idea is that three numbers can be reduced to 2, in a 3:2 compressor, by doing the addition while keeping the carries and therefore the sum separate[6]. This means that all of the columns can be added in parallel without relying on the result of the previous column, creating a two output "adder" with a time delay that is independent of the size of its inputs[6]. The sum and carry can be recombined in a normal addition to form the correct result. This process may seem more complicated and pointless, but the power of this technique is that any amount, number of can be added together in this manner. It's only the ultimate recombination of the ultimate carry and sum that needs a carry propagating addition. 3:2 compressor is also known as full adder. It adds three one bit binary numbers, a sum and a carry[1]. The full adder is typically a component during a cascade of adders. The carry input for the complete adder circuit is from the carry output from the cascade circuit. Carry output from full adder is fed to a different full adder.

The characteristics of the 4:2 compressors are[6]:

1. The outputs represent the sum of the five inputs, so it is indirectly a 5 bit adder.
2. Both carriers are having equal weighting time.
3. To avoid carry propagation, the value of Cout is purely depends on the four major inputs but not on carry input from previous stage.
4. The Cout signal forms the input to the Cin of a 4:2 of subsequent column.

The basic execution of a 4-2 Compressor is accomplished by using two full-adders (FA) To include parallel numbers cells.4:2 Compressor is made out of two sequentially associated full adders. With insignificant convey proliferation we use Compressor adder rather than other adder. Compressor is an advanced present day circuit which is utilized for rapid speed with least number of gates requiring structure method. This compressor turns into the fundamental apparatus for quick duplication including procedure quick processor and lesser region.The typical 4:2 compressor are capable of adding 4bits along with one carry propagated from previous stage, in turn producing a 3bit output. So the Compressor has 4 input bits α, β, ϒ, δ and 2 outputs Sum and Carry along with a Carry-in (Cin) and a Carry-out (Cout).

Our designed structure can be represented is as shown below

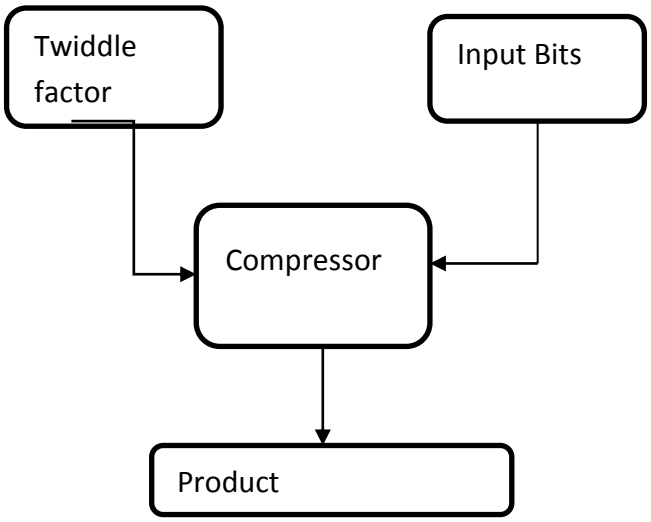

Fig 3.2 Working of Compressor

The input Cin is the output from the corresponding previous compressor The Cout is the output to the compressor is propagated to the next significant stage. The critical path towards the output is smaller in comparison with an equivalent circuits to add 5 bits using full adders. However, in the case of 3-2 Compressor, Also the MUX block at the Sum output gets the selected bit before the inputs arrive and this minimizes the delay to a considerable extent.

From the flowchart of the Compressor Fig 3.2 it is clear that when the Input and Twiddle factors are Multiplied and partial products are generated are added by using the Compressor, So the one Compressor can participate in both Multiplication and Addition, Since the Compressor is having the MUX and XOR gates it is easy to implement.

4. RESULTS

The simulation results are done using Xilinx Vivado 2016.4 design suit for a 7 series family of Artix with device configure as xc7a100t (active) and corresponding package is CSG324 with speed grade of -1. Our design is successfully simulated as per the fig 4.2 as Schematic diagram and Fig 4.1 as output Waveform. This design utilizes 194 LUTs, 0.459W power and 17.608ns as delay and our design is 50% power efficient compared with other FFT using multiplier as Booth multiplier as per the table given Table 4.1.

	Booth Multiplier	Compressor Multiplier
LUTs	206	194
IOBs	192	192
Delay(ns)	19.090	17.608
Power(W)	0.978	0.459

Table 4.1 Comparison between Booth Multiplier and Compressor multiplier in FFT implementation.

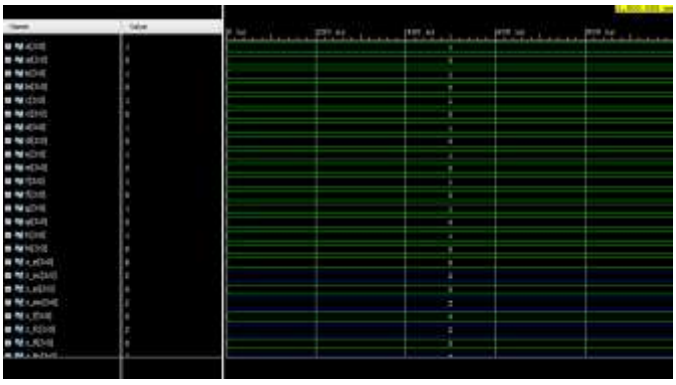

Fig 4.1 Simulation Result of FFT using Compressor

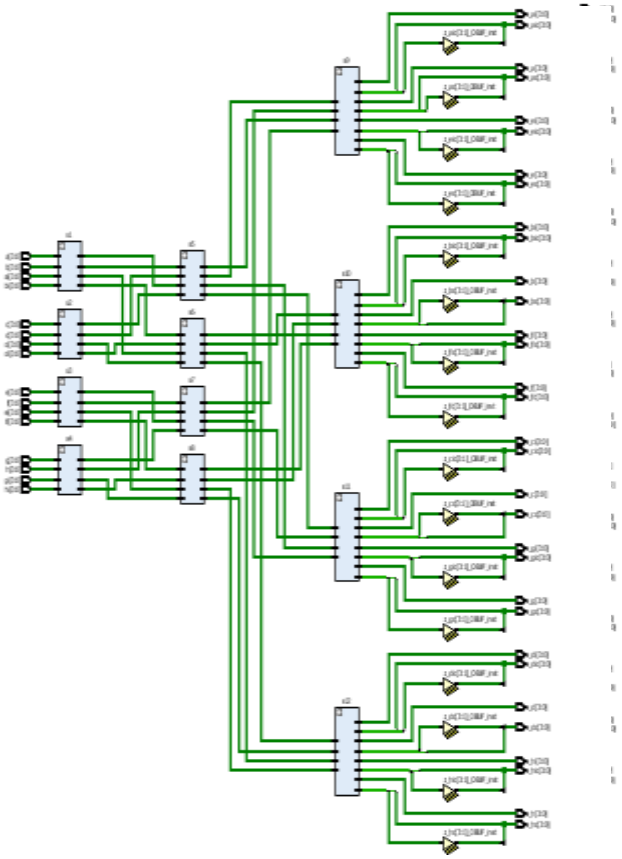

Fig 4.2 Schematic diagram of 8-point FFT using Compressor

5. CONCLUSION

FFT was implemented with the combination of adder and multipliers designed with the help of new XOR gate topology(compressors) consisting of XOR Gate and Multiplexers which consumes less power since it contains less gates and multipliers than any other FFT implementations.

For the implementation of FFT we designed Multipliers with the help of Compressors and Adders, these Compressors are delay efficient structures. The Simulation and Synthesis process are successfully done using Xilinx Vivado 2016.4 design suit.

6. REFERENCES

[1] Mateus Beck Fonseca • Eduardo A. Ce ´sar da Costa • Joa ˜o B. S. Martins,"Design of power efficient butterflies from Radix-2 DIT FFT using adder compressors with a new XOR gate topology". Springer, Analog Integr Circ Sig Process (2012) 73:945–954.

[2] Geetika Pandey, Mandeep Singh Narula , "IMPLEMENTATION OF FFT ALGORITHM", May 2017.

[3] Sowmiya#, K.Stella* and V.M.Senthilkumar, "Design and Analysis of 4-2 Compressor for Arithmetic ApplicationS". Asian Journal of Applied Science and Technology (AJAST)Volume 1, Issue 1, Pages 106-109, February 2017

[4] Anusree T U and Bonifus P L, "A Study on Compressor Adders for Fast Multipliers", vol 5, June 2016

[5] Pang Jia Hong and Nasri Sulaiman, "Design of a Reconfigurable FFT Processor using Multi-objective Genetic Algorithm"

[6] Dalai Gowri Sankar Rao, and K. Jayaram Kumar, "Performance Analysis of 4-2 Compressor and 5-2 Compressor Using Multiplication", Vol.04, Issue.10, October-2016, Pages: 1055-1057.

[7] A.Chandrakala, A. Sreeramulu, L. Srinivas,"Design and Implementation of 4-2 Compressor Design with New Xor-Xnor," ISO 3297:2007 Certified Vol. 3, Issue 7, July 2016

[8] Premananda B S. Samarth S. Pai, Shashank B, Shashank S. Bhat, Design and Implementation of 8-bit Vedic Multiplier Vol-2,Issue 12,Dec 2013.

[9] B. Lakshmi and A. S. Dhar, "Review ArticleCORDIC Architectures: A Survey", Hindawi Publishing 2010.

[10] Mahmud Benhamid, and Masuri Othman, " FPGA Implementation of a Canonical Signed Digit Multiplier-less based FFT Processor for Wireless Communication Applications."

[11] FaveZ Elguibaly, "A-CORDIC: An Adaptive CORDIC algorithm".

[12] Akshata R. Prof. V.P. Gejji, Prof.B.R.Pandurang, Kanchan A. Joshi, "DESIGN OF HIGH SPEED FFT USING VEDIC MATHEMATICS".

[13] J. W. Cooley and J. W. Tukey, "An Algorithm for The Machine Calculation of Complex Fourier Series," Mathematics of Computation, vol. 19, no. 90, pp. 297–297, May1965.