ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization *Volume 8, Special Issue 1, March 2019*

7th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '19)

19th-20th March 2019

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

Design and Implementation of CORDIC-based FFT Algorithm in FPGA System

Naganaik Mudhavathu, Dr.P.Karpagavalli

Research Scholar, Department of Electronics and Communication Engineering, Sri Satya Sai University of Technology & Medical Sciences, Bhopal, Sehore (M.P.), India

Associate Professor, Department of Electronics and Communication Engineering, Sri Satya Sai University of Technology & Medical Sciences, Bhopal, Sehore (M.P.), India

ABSTRACT: This paper presents a designing scheme of high-speed real-time serial pipelined Fast Fourier Transform (FFT) processor on FPGA which is based on Coordinate Rotation Digital Computer (CORDIC) algorithm. The CORDIC algorithm will reduce the hardware complexity compared to the direct implementation of the butterflies using complex multipliers. This paper presents a pipelined, reduced memory and low power CORDIC-based architecture for fast Fourier transform implementation. The proposed algorithm utilizes a new addressing scheme and the associated angle generator logic in order to remove any ROM usage for storing twiddle factors. CORDIC is implemented by a simple hardware through repeated shift-add operations Low power is achieved by the using the Coordinate Rotation Digital Computer algorithm in the place of conventional multiplication and furthermore, dynamic power consumption is reduced with no delay penalties

KEYWORDS: FFT, FPGA, CORDIC, ROM, Twiddle Factors.

I. INTRODUCTION

Fast Fourier transform (FFT) is used for reducing the complexity of computations in Discrete Fourier Transform (DFT). Fast Fourier transform (FFT) is among the most widely used operations in digital signal processing. Often, a high performance FFT processor determines most of the design metrics in many applications such as image processing, sonar, general filtering, spread-spectrum communications, convolution, etc. Many of them require a good precision and real-time response. Software Defined Radio (SDR) and Synthetic Aperture Radar (SAR) are among the fastest growing application areas ; in particular, Orthogonal Frequency-Division Multiplexing (OFDM) is currently a focus of research and development . Efficient hardware realization of FFT with small area, low-power dissipation and real time computation is a significant challenge in portable devices.
A FFT processor consists of control logic (address generator for data and twiddle factor accesses), butterfly calculation units and a memory bank. For FFT processors, butterfly operation is the most computationally demanding stage. Generally, an FFT processor utilizes only one butterfly unit to perform all calculations iteratively, and the “in-place” memory access strategy is required for the least amount of memory. With “in-place” strategy, the outputs of a butterfly operation are stored back to the same memory location of the inputs, saving the memory usage by one half. However, correct memory addressing scheme is required to avoid the data conflict. Here, we implements an efficient addressing scheme to realize the Serial-in and Serial-out and “in-place” memory accessing; which produces an output at every clock cycle .
Traditionally, a butterfly unit consists of complex adders and multipliers. The butterfly operation can be realized by Coordinate Rotation Digital Computer (CORDIC) without using any dedicated multiplier hardware. CORDIC algorithm is suitable for the butterfly operations in FFT since it requires only add and shift operations, making it very hardware efficient. In the conventional FFT processor, a large ROM space is needed to store all the twiddle factors. To reduce the chip area, a twiddle factor generator has been proposed. The twiddle factor angles are stored in a ROM for the butterfly operation in a CORDIC-based FFT processor. Additionally, the CORDIC-based butterfly can be twice faster than traditional multiplier-based butterflies in VLSI implementations. In this study, we propose a new angle

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 8, Special Issue 1, March 2019

7th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '19)

19th-20th March 2019

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

generation method and angles are generated on real time basis using proposed address decoding scheme. Here, there is no need of storing the twiddle factor angle values. By using the control signal which is based on the current FFT stage and the location of the butterfly in a stage, different twiddle factor angles can be generated in the simple and regular manner. With this approach, full memory requirements of an FFT processor can be reduced by more than 23%. Fundamentals of CORDIC algorithm and the design of CORDIC-based FFT processor are described in below CORDIC based FFT algorithm in FPGA system are presented.

In this, we propose a CORDIC based FFT algorithm which eliminates the need for storing the twiddle factor angles. The algorithm generates the twiddle factor angles successively by an accumulator. With this approach, full memory requirements of an FFT processor can be reduced by more than 20%. Memory reduction improves with the increased radix size. Since the critical path is not modified with the CORDIC angle calculation, system throughput does not change.

II. FAST FOURIER TRANSFORM (FFT) ALGORITHMS

FFTs of length $N=2^M$ equal to a power of two are, by far, the most commonly used. These algorithms are very efficient, relatively simple, and a single program can compute power-of-two FFTs of different lengths. As with most FFT algorithms, they gain their efficiency by computing **all** DFT points simultaneously through extensive reuse of intermediate computations; they are thus efficient when many DFT frequency samples are needed. The simplest power-of-two FFTs are the decimation-in-time radix-2 FFT and the decimation-in-frequency radix-2 FFT; they reduce the length-$N=2^M$ DFT to a series of length-2 DFT computations with twiddle-factor complex multiplications between them. The radix-4 FFT algorithm similarly reduces a length-$N=4^M$ DFT to a series of length-4 DFT computations with twiddle-factor multiplies in between. Radix-4 FFTs require only 75% as many complex multiplications as the radix-2 algorithms, although the number of complex additions remains the same. Radix-8 and higher-radix FFT algorithms can be derived using multi-dimensional index maps to reduce the computational complexity a bit more. However, the split-radix algorithm and its recent extensions combine the best elements of the radix-2 and radix-4 algorithms to obtain lower complexity than either or than any higher radix, requiring only two-thirds as many complex multiplies as the radix-2 algorithms. All of these algorithms obtain huge savings over direct computation of the DFT, reducing the complexity from $O(N2)$ to $O(N\log N)$.

The radix-2 decimation-in-time and decimation-in-frequency fast Fourier transforms (FFTs) are the simplest FFT algorithms. Like all FFTs, they gain their speed by reusing the results of smaller, intermediate computations to compute multiple DFT frequency outputs.

A fast Fourier transform, or FFT, is not a new transform, but is a computationally efficient algorithm for the computing the DFT. The length-N

DFT, defined as

$$X(K) = \sum_{n=0}^{N-1} x(n) e^{-\left(\frac{j2\pi kn}{N}\right)}$$

$$X(K) = \sum_{n=0}^{N-1} x(n) W^{kn}$$

$$W_N = e^{-j\left(\frac{2\pi}{N}\right)}$$

When N is a power of two, the FFT based on the Cooley –Tukey algorithm is most commonly used in order to compute the DFT efficiently. The Cooley – Tukey algorithm reduces the number of operations (N^2) for the DFT to ($N \log_2 N$) for the FFT. In accordance with this the FFT calculated in a series of $n= \log_p N$ stages where p is the base of the radix. The other popular algorithm is the radix-4 FFT, which is even more efficient than the radix-2 FFT.

radix-2 decimation-in-time FFT

The radix-2 decimation-in-time algorithm rearranges the discrete Fourier transform (DFT) equation into two parts: a sum over the even-numbered discrete-time indices $n=[0,2,4,\ldots,N-2]$ and a sum over the odd-numbered indices $n=[1,3,5,\ldots,N-1]$ as in Equation:

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization *Volume 8, Special Issue 1, March 2019*

7th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '19)

19th-20th March 2019

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

$$X(K) = \sum_{n=0}^{N-1} x(n)W^{kn}$$

$$X(K) = \sum_{n=0}^{N/2-1} x(2n)W^{2kn} + W^{k} \sum_{n=0}^{N/2-1} x(2n+1)W^{2kn}$$

we can write

$$W_N^2 = \left(e^{-j2\pi/N}\right)^2 = e^{-\frac{j2\pi}{N}/2} = W_{\frac{N}{2}}$$

i.e

$$W_N^2 = W_{\frac{N}{2}}$$

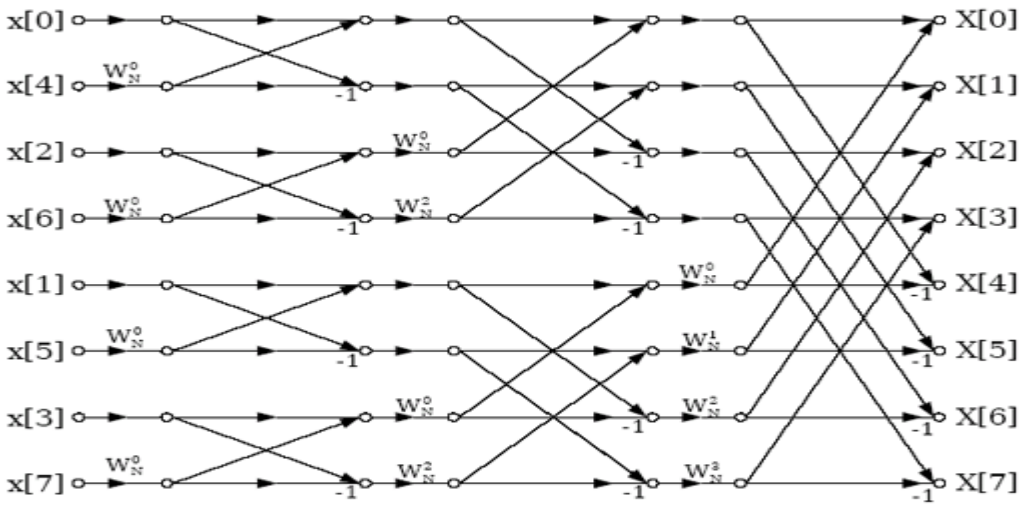

Figure 1: Radix-2 Decimation-in-Time FFT algorithm for a length-8 signal

In a decimation-in-time radix-2 FFT as shown in Figure, the input is in bit-reversed order (hence "decimation-in-time"). That is, if the time-sample index *n* is written as a binary number, the order is that binary number reversed. The bit-reversal process is illustrated for a length-*N*=8

For N=8

In-order index	In-order index in binary	Bit-reversed binary	Bit-reversed index
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

Table-1:- bit-reversal process for a length-*N*=8

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization *Volume 8, Special Issue 1, March 2019*

7th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '19)

19th-20th March 2019

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

Radix-2 decimation-in-frequency algorithm:-

The same radix-2 decimation in frequency can be applied recursively to the two length-*N*/2 DFTs to save additional computation. When successively applied until the shorter and shorter DFTs reach length-2, the result is the radix-2 decimation-in-frequency FFT algorithm.

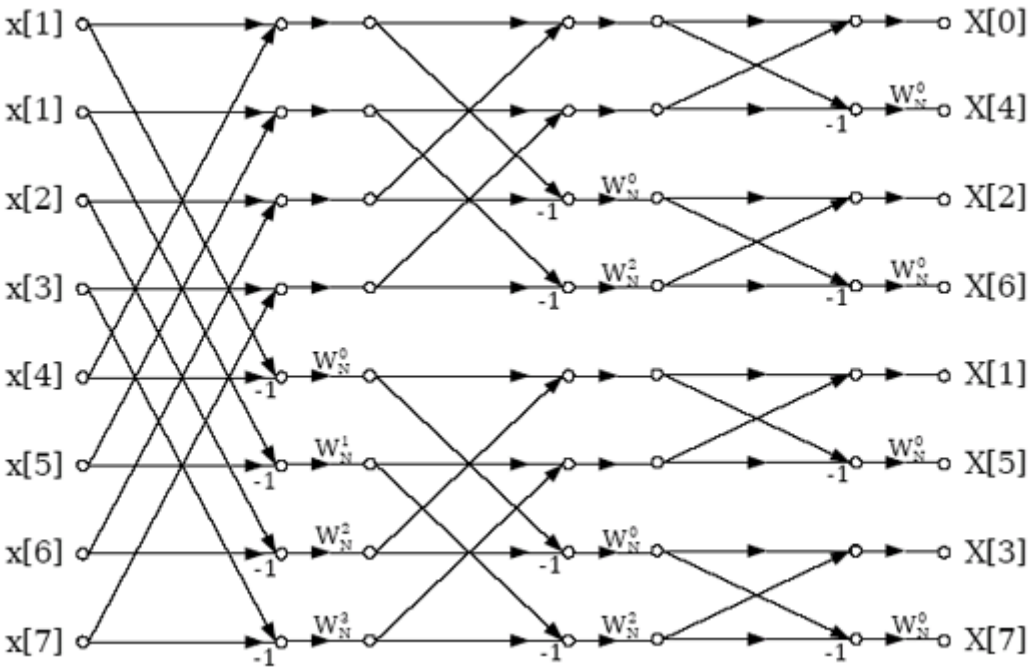

Figure 2:Radix-2 decimation-in-frequency FFT for a length-8 signal

The radix-4 FFT equation is listed below:

$$X(K) = \sum_{n=0}^{\frac{N}{4}-1} (x(n) + (-j)^k x\left(n + \frac{N}{4}\right) + (-1)^k x\left(n + \frac{N}{2}\right) + (j)^k x\left(n + \frac{3N}{4}\right)) W_N^{nk}$$

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization *Volume 8, Special Issue 1, March 2019*

7th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '19)

19th-20th March 2019

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

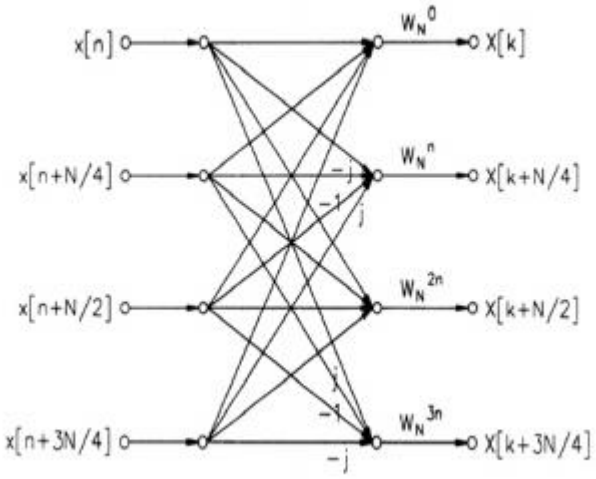

Figure 3: Radix-4 FFT Algorithm

III. CORDIC Algorithm

An Introduction to the CORDIC Algorithm

CORDIC (coordinate rotation digital computer) is a hardware-efficient iterative method which uses rotations to calculate a wide range of elementary functions.

This article reviews the basics of this algorithm and later demonstrates how we can use CORDIC to calculate the sine and cosine of a given angle.

Rotate to Perform a Wide Range of Operations

For the time being, let's forget about electronics and go back to high-school mathematics to see which operations can be achieved by simply rotating a vector.

Suppose that we have an efficient system that receives a vector and rotates it by an arbitrary angle θ. Choosing the origin as the centre of rotation, we will get to the point (x_1,y_1) by rotating the point (x_0,y_0) by θ

$$x_R=x_{in}\cos(\theta)-y_{in}\sin(\theta)$$
$$y_R=x_{in}\sin(\theta)+y_{in}\cos(\theta) \quad \longrightarrow (1)$$

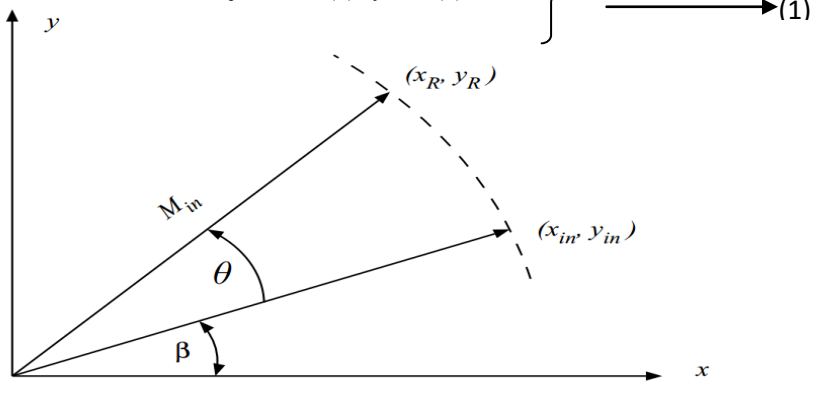

Figure 4:- Rotating the input vector by θ

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 8, Special Issue 1, March 2019

7th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '19)

19th-20th March 2019

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

If we choose $y_{in}=0$ and $x_{in}=1$, after rotation we will have

$$\left.\begin{aligned} x_R&=\cos(\theta)\\ y_R&=\sin(\theta)\end{aligned}\right\} \quad \text{----------(2)}$$

Therefore, we can simply calculate sine and cosine of an arbitrary angle through rotation.

For another example of the functions that can be calculated from rotation, consider the vector magnitude, $\sqrt{{x_{in}}^2+{y_{in}}^2}$. To achieve this, we only need to rotate the input vector so that it is aligned with the x-axis. In this way, $y_R=0$ and the x component will give the vector magnitude.

Interestingly, the list of the functions that can be calculated from rotation is relatively long. Inverse trigonometric functions such as arctan, arcsin, arccos, hyperbolic and logarithmic functions, polar to rectangular transform, Cartesian to polar transform, multiplication, and division are some of the most important operations that can be obtained from variants of rotation.

The CORDIC algorithm attempts to provide a hardware-efficient method for calculating these functions. "Hardware-efficient" means that the algorithm avoids the use of multipliers and relies on only shifts and additions/subtractions. Note that other methods of implementing these functions, such as utilizing power series, usually need dedicated multipliers.

The Basics of CORDIC

Equation 1 can be simplified to:

$$\begin{bmatrix}x_R\\ y_R\end{bmatrix}=cos\theta\begin{bmatrix}1 & -tan\theta\\ tan\theta & 1\end{bmatrix}\begin{bmatrix}x_{in}\\ y_{in}\end{bmatrix} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

The above equation shows that for one rotation, we need to perform 4 multiplications (plus some additions/subtractions).

The CORDIC algorithm resorts to two fundamental ideas to achieve rotation without multiplication. The first fundamental idea is that rotating the input vector by an arbitrary angle θ_d is equal to rotating the vector by several smaller angles, θ_i, i=0,1,…,n, provided $\theta_d=\sum_{i=0}^{n}\theta_i$

The second fundamental idea is that we can choose the small elementary angles in a way that $\tan(\theta_i)=2^{-i}$ for i=0,1,…,n. In this way, multiplication by $\tan(\theta_i)$ can be achieved by a bitwise shift which is a much faster operation compared to multiplication. You may wonder if, for a given θd, we can satisfy the conditions of these two fundamental ideas simultaneously, i.e. $\theta_d=\sum_{i=0}^{n}\theta_i$ while $\tan(\theta_i)=2^{-i}$

Therefore, omitting the scaling factor term from Equation 3, we obtain the following equations for the CORDIC algorithm:

$$\left.\begin{aligned} x[i+1]&=x[i]-\sigma i 2^{-i}\, y[i]\\ y[i+1]&=y[i]+\sigma i 2^{-i}\, x[i]\end{aligned}\right\} \quad \text{----------(4)}$$

where $\sigma i\in\{+1,-1\}$, determines the sign of the elementary angle and will be discussed in the next section of the article. The above equation shows that the algorithm will always perform a certain number of rotations with the predefined angles (for example, 12 rotations), and the only thing that the algorithm determines is whether each rotation will go clockwise or counter-clockwise during each iteration (choose σi equal to +1 or -1).

The described procedure is actually a negative feedback mechanism which computes the overall rotation and compares that to a reference value and chooses the new rotations in a way that minimizes the error signal ($\theta_{error}=\theta_d-\sum_{i=0}^{n}\theta_i$)

The above mechanism is taken into account by adding another equation to the set of expressions in Equation 4 as follows:

$$\left.\begin{aligned} x[i+1]&=x[i]-\sigma i 2-iy[i]\\ y[i+1]&=y[i]+\sigma i 2-ix[i]\\ z[i+1]&=z[i]-\sigma_i\tan-1(2^{-i})\end{aligned}\right\} \quad \text{----------(5)}$$

This equation accumulates the angle of all the previous rotations and compares that with an initial value z[0]

The following figure shows the implementation of a single iteration in the CORDIC algorithm:

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization *Volume 8, Special Issue 1, March 2019*

7th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '19)

19th-20th March 2019

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

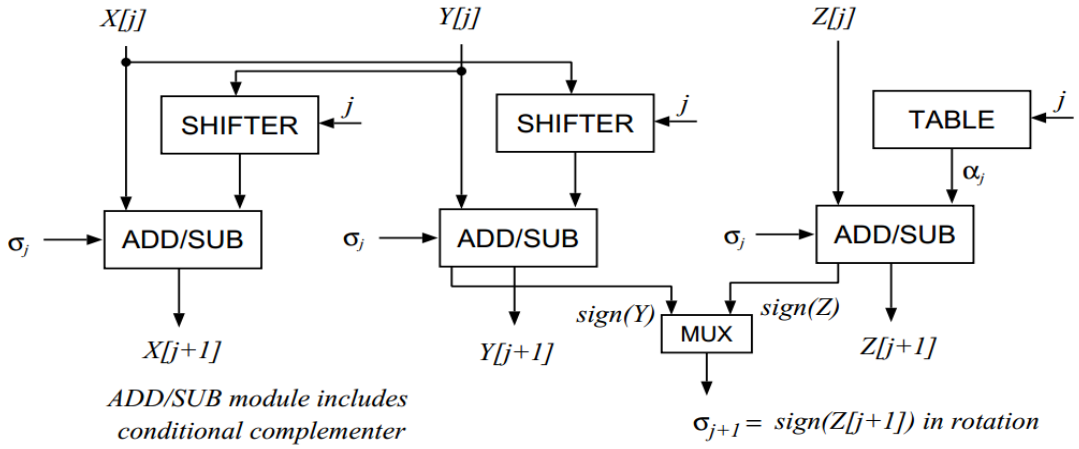

Figure 5. Implementation of Equation 4 for one iteration

IV. CORDIC-BASED FFT ALGORITHM IN FPGA SYSTEM

There are a number of ways to implement a CORDIC processor. The ideal architecture depends on the speed versus area tradeoffs in the intended application. First we will examine an iterative architecture that is a direct translation from the CORDIC equations. From there, we will look at a minimum hardware solution and a maximum performance solution.

4.1Iterative CORDIC Processors:-

An iterative CORDIC architecture can be obtained simply by duplicating each of the three difference equations in hardware as shown in Figure 6. The decision function, d_i, is driven by the sign of the y or z register depending on whether it is operated in rotation or vectoring mode. In operation, the initial values are loaded via multiplexers in to the x, y and z registers. Then on each of the next n clock cycles, the values from the registers are passed through the shifters and adder-subtractors and the results placed back in the registers. The shifters are modified on each iteration to cause the desired shift for the iteration. Likewise, the ROM address is incremented on each iteration so that the appropriate elementary angle value is presented to the z adder-subtractor. On the last iteration, the results are read directly from the adder-subtractors. Obviously, a simple state machine is required keep track of the current iteration, and to select the degree of shift and ROM address for each iteration. The design depicted in Figure 1 uses word-wide data paths(called bit-parallel design). The bit-parallel variable shift shifters do not map well to FPGA architectures because of the high fan-in required. If implemented, those shifters will typically require several layers of logic (i.e., the signal will need to pass through a number of FPGA cells). The result s a slow design that uses a large number of logic cells.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization *Volume 8, Special Issue 1, March 2019*

7th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '19)

19th-20th March 2019

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

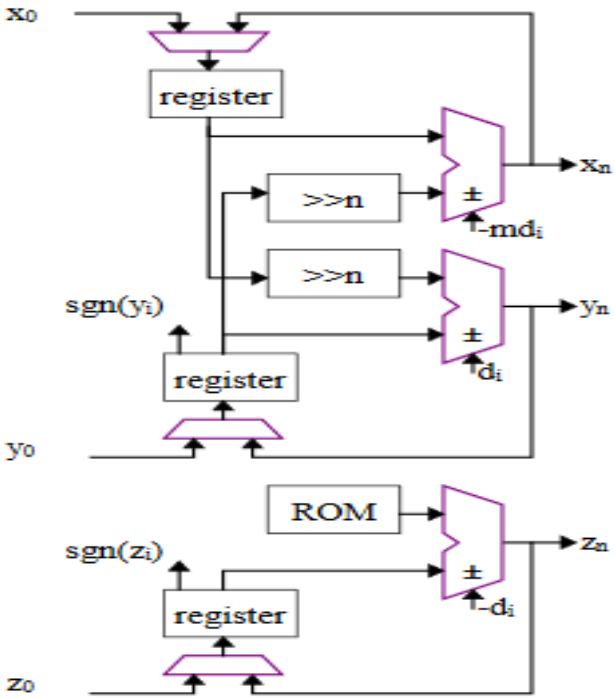

Figure 6:-. Iterative CORDIC structure

A considerably more compact design is possible using bit serial arithmetic. The simplified interconnect and logic in a bit serial design allows it to work at a much higher clock rate than the equivalent bit parallel design. Of course, the design also needs to clocked w times for each iteration (w is the width of the data word). The bit serial design consists of three bit serial adder-subtractors, three shift registers and a serial Read Only Memory (ROM). Each shift register has a length equal to the word width. There is also some gating or multiplexers to select taps off the shift registers for the right shifted cross terms (shifting is accomplished using bit delays in bit serial systems). The bit serial CORDIC architecture is shown in Figure 6. In this design,w clocks are required for each of the n iterations, where w is precision of the adders. In operation, the load multiplexers on the left are opened for w clock periods to initialize the x,y and z registers (these registers could also be parallel loaded to initialize). Once loaded, the data is shifted right through the serial adder-subtractors and returned to the left end of the register. Each iteration requires w clocks to return the result to the register. At the beginning of each iteration, the control state machine reads the sign of the y(or z) register and sets the add/subtract controls accordingly. The appropriate tap off the register for the cross terms is also selected at the beginning of each iteration. During the nth iteration, the results can be read from the outputs of the serial adders while the next initialization data is shifted into the registers.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization *Volume 8, Special Issue 1, March 2019*

7th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '19)

19th-20th March 2019

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

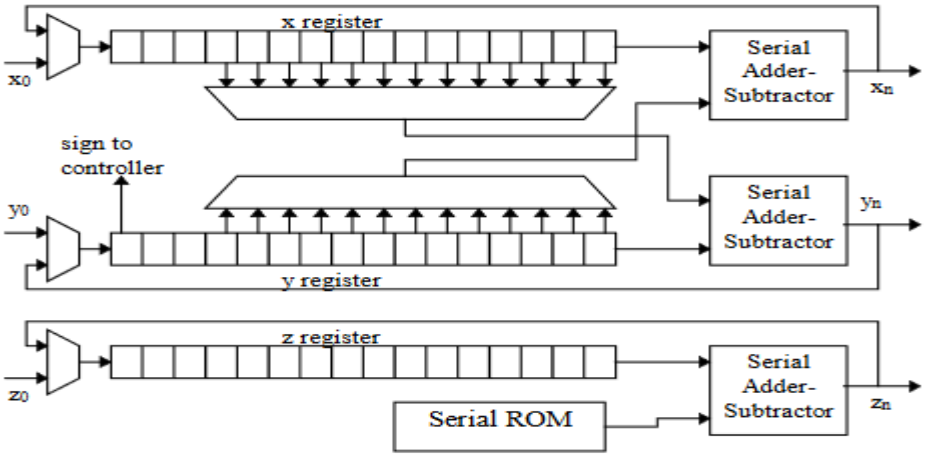

Figure 7:- Bit serial iterative CORDIC

The simplicity of the bit serial design is apparent from figure 7. Even in this case, the wiring of the shift tap multiplexers can present problems in some FPGAs (this is one place where tri-state long lines can come in handy).Even so, the interconnect is minimal and the logic between registers is simple. This combination permits bit clock rates near the maximum toggle frequency of the FPGA. The possibility of using extreme bit clock frequencies makes up for the large number of clock cycles required to complete each rotation

V. RESULTS

In this section, for four families of Xilinx FPGAs we implement the proposed design and discuss the results of the overhead assessment. For the original CORDIC designs the analysis is performed and using Xilinx ISE 14.5 for) the structures of proposed error detection is designed. In seven rotation modules the designs are divided and in each rotation module there has two ordinary or self checking Subtractor modules /Adder modules. Verilog hardware description language has been used for implementation. The design was tested in SPARTAN-3 FPGA (XC3S50). Hardware resources required for the design can be summarized as:
Table 1. FPGA Resource Utilization

Test Condition	Used	Utilization
Number of slice flip-flops	120 of 1492	8%
Number of occupied slices	132 of 782	16%
Number of LUTs	197 of 1612	12%
Number IOBs	32 of 141	22%

VI. CONCLUSION

CORDIC is a powerful algorithm, and a popular algorithm of choice when it comes to various Digital Signal Processing applications. Implementation of a CORDIC-based processor on FPGA gives us a powerful mechanism of implementing complex computations on a platform that provides a lot of resources and flexibility at a relatively lesser cost. Further, since the algorithm is simple and efficient the design and VLSI implementation of a CORDIC based processor is easily achievable.
In this paper a CORDIC module is designed and simulated using Xilinx ISE using VHDL as a synthesis tool. The output of the CORDIC core is analyzed and verified on the test-bench, and compared with the actual values obtained from Matlab. Finally, a joint simulation of MATLAB and MODELSIM is carried out and the results show that the design meets the requirements of real-time FFT processing. The overall design uses pipelined structure, improves the throughput rate of the FFT processor and improves the performance. In addition, its SNR loss is relatively small.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization *Volume 8, Special Issue 1, March 2019*

7th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '19)

19th-20th March 2019

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

REFERENCES

[1] H. Huang, L. Xiao, and J. Liu, "CORDICBased Unified Architectures for Computation of DCT/IDCT/DST/IDST", Circuits, Systems, and Signal Processing, Vol.33, No.3, pp.799- 814, 2013.
[2] K. Ray and A. Dhar, "CORDIC-Based VLSI Architectures of Running DFT with Refreshing Mechanism", Journal of Signal Processing Systems, Vol.90, No.2, pp.1-12, 2018.
[3] D. Chen and M. Sima, "Fixed-Point CORDICBased QR Decomposition by Givens Rotations on FPGA", In: Proc. of International Conference on Reconfigurable Computing and FPGAs, pp.327-332, 2011.
[4] P. Meher, J. Valls, Tso-Bing Juang, K. Sridharan, and K. Maharatna, "50 Years of CORDIC: Algorithms, Architectures, and Applications", IEEE Transactions on Circuits and Systems I: Regular Papers, Vol.56, No.9, pp. 1893-1907, 2009.
[5] Y. Liu, L. Fan, and T. Ma, "A Modified CORDIC FPGA Implementation for Wave Generation", Circuits, Systems, and Signal Processing, Vol.33, No.1, pp.321-329, 2013.
[6] R. Shukla and K. Ray, "Low Latency Hybrid CORDIC Algorithm", IEEE Transactions on Computers, Vol.63, No.12, pp.3066-3078, 2014.
[7] S. Aggarwal, P. Meher, and K. Khare, "AreaTime Efficient Scaling-Free CORDIC Using Generalized Micro-Rotation Selection", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol.20, No.8, pp.1542-1546, 2012.
[8] P. Vyas, L. Vachhani, K. Sridharan, and V. Pudi, "CORDIC-Based Azimuth Calculation and Obstacle Tracing via Optimal Sensor Placement on a Mobile Robot", IEEE/ASME Transactions on Mechatronics, Vol.21, No.5, pp.2317-2329, 2016.
[9] H. Nguyen, X. Nguyen, C. Pham, T. Hoang, and D. Le, "A parallel pipeline CORDIC based on adaptive angle selection", In: Proc. of International Conference on Electronics, Information, and Communications, pp.1-4, 2016.
[10] P. Velrajkuma, C. Senthilpar, G. Murthy, and E. Wong, "Low Energy, Improved Speed and High Throughput CORDIC Cell to Improve Performance of Robots' Processor", Asian Journal of Scientific Research, Vol.8, No.3, pp.381-391, 2015.
[11] Chundi Vedasri, SambasivaNayak.R "Design of low power consumption of 8 bit multiplier Using NS Gate" Published in the International Journal of Advanced Scientific Technologies in Engineering and Management Sciences (IJASTEMS-ISSN: 2454- 356X) Volume.3, Issue.4, April.2017 www.ijastems.org Page 6 -8. UGC and ISSN Approved Journal.
[12] T. Hoang, H. Nguyen, X. Nguyen, C. Pham, and D. Le, "High-performance DCT architecture based on angle recoding CORDIC and Scale-Free Factor", In: Proc. of the Sixth International Conference on Communications and Electronics, pp.199-204, 2016.
[13] A. Y. Wu and C. S. Wu, "A unified view for vector rotational CORDIC algorithms and architectures based on angle quantization approach", IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, Vol.49, No.10, pp.1442-1456, 2002.
[14] C. S. Wu, A. Y. Wu, and C. H. Lin, "A highperformance/low-latency vector rotational CORDIC architecture based on extended elementary angle set and trellis-based searching schemes", IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, Vol.50, No.9, pp.589-601, 2003.
[15] J. Mehta and P. Trivedi, "An enhanced mixedscaling-rotation CORDIC algorithm with weighted amplifying factor", In: Proc. of the IEEE International Conference on Digital Signal Processing, pp.527-531, 2016.
[16] R. Sambasiva Nayak, MD.Javeed Ahammed "CMOS/VLSI Circuit for Power Optimization on Portable Devices,"Iinternational Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278 -3075, Volume-8, Issue-12, October-2019, and PP: 4300-4303. DOI: 10.35940/ijitee.L2716.1081219. (Scopus Journals).
[17] M. Madheswaran and T. Menakadevi, "An Improved Direct Digital Synthesizer Using Hybrid Wave Pipelining and CORDIC algorithm for Software Defined Radio", Circuits, Systems, and Signal Processing, Vol.32, No.3, pp.1219-1238, 2012.
[18] H. Huang, W. Wang, B. Wu, and T. Gao, "CORDIC-based fast radix-2 DST algorithms", In: Proc. of the International Conference on Software Intelligence Technologies and Applications & International Conference on Frontiers of Internet of Things, pp.246-249, 2014.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization *Volume 8, Special Issue 1, March 2019*

7th National Conference on Frontiers in Communication and Signal Processing Systems (NCFCSPS '19)

19th-20th March 2019

Organized by

Department of ECE, Adhiyamaan College of Engineering, Hosur, Tamilnadu, India

[19] Sivaprasad Godugunuri, Sambasiva Nayak.R, "Low-Power Differential SRAM design for SOC based on the 25-um Technology" published in the IOP Conference Series: Materials Science and Engineering, IOP Publishing Ltd, UK (Scopus Journals).

[20] A. Banerjee and A. Dhar, "Pipelined VLSI Architecture using CORDIC for Transform Domain Equalizer", Journal of Signal Processing Systems, Vol.70, No.1, pp.39-48, 2012.

[21] Y. H. Hu and H. Chern, "A novel implementation of CORDIC algorithm using backward angle recoding (BAR)", IEEE Transactions on Computers, Vol.45, No.12, pp.1370-1378, 1996.

[22] R. Samba Siva Nayak, Suman.J "FPGA Implementation of Convolutional Interleaver & De-interleaver", Published in the International Journal of Engineering and Social Science (IJESS), ISSN: 2249-9482 Volume-1, Issue-2, January 2012, pp: 96-102.