IMPLEMENTATION OF CORDIC II ALGORITHM IN FASTFOURIER TRANSFORM

N. Subhathra Devi
PG student, Department of Electronics and Communication Engineering,
Pondicherry Engineering College,
Puducherry, India.
subhathra3094@gmail.com

S. Tamilselvan
Associate Professor, Department of Electronics and Communication Engineering,
Pondicherry Engineering College,
Puducherry, India.
tamilselvan@pec.edu

Abstract- This paper presents the CORDIC II based Fast Fourier transform. CORDIC II algorithm which differs from conventional CORDIC in the used set of micro rotation, where CORDIC II uses minimum number of adders which leads to a reduced latency. This algorithm is implemented in Fast Fourier transform computation, the use of CORDIC II in FFT results in the elimination of multipliers, saves area, delay and power.

Keywords: *CORDIC II, Fast Fourier Transform (FFT), Discrete Fourier Transform (DFT), Friend angles, Uniformly Scaled Redundant (USR) CORDIC, Nano Rotation.*

I. Introduction

The most widely used operations in digital signal processing is Fast Fourier Transform. For many applications such as Orthogonal Frequency-Division Multiplexing (OFDM), Synthetic Aperture Radar (SAR) and software defined radio, FFT processor is the key component and it determines most of the design metrics. Efficient hardware realization of FFT with small area, low-power dissipation and real time computation is a significant challenge in portable devices of embedded systems. FFT processor consists of memory banks, butterfly calculation units, and control logic (data and twiddle factor accesses require for address generator). Mostly, FFT processor require only one butterfly unit to realize all calculations iteratively. To save the memory usage by one half, the outputs of a butterfly operation are stored back to the same memory location of the inputs, to avoid the data conflict,correct memory addressing scheme is required.

Micro rotation is the benefit of CORDIC (**CO**ordinate **R**otation **DI**gital **C**omputer) algorithm which is calculated by simple shift-and-add operations [1]. This simple and efficient algorithm is used to calculate hyperbolic and trigonometric functions, typically converging with one digit or bit per iteration. The main principle of this algorithm is simple: It breaks down the rotation angle into sum of angles and carries out the rotation by a series of micro rotation. Among general rotators, numerous class of CORDIC algorithm are discussed in the literature. Some works combine with diverse micro rotation stages into a single stage [2], [3] in order to reduce the number of iterations. Micro rotation which is based on repeating and skipping are discussed in [4]. Some approaches in micro rotation are based on Taylor series approximation [5]-[7]. Reduction in instruction set is designed in CORDIC algorithm with minimum number of adders, they call it as CORDIC II algorithm which differs from the approaches used in the previous set of micro rotations, called angle set [8]. This new set provides a fast convergence of the rotation angle which leads to reduce latency and a smaller number of adders. The Fast Fourier Transform (FFT) is an efficient algorithm which is used to calculate the Discrete Fourier Transform (DFT) and it is a core component of OFDM transmitter and receiver. In communication systems, large-point of Fast Fourier Transform are required due to demand in higher data rates such as 1024/2048 etc [9]. In previous work they implemented CORDIC based fast Fourier transform [10], their design is to replace the sine and cosine twiddle factors in standard FFT architecture by repeating CORDIC rotations which allows in reduction of read-only memory (ROM).

In this work, the old paradigm was overcomed by implementing CORDIC II algorithm in FFT computation to realize the operation with reduced memory requirements. Where CORDIC II uses

minimum number of adders which leads to reduce latency. While implementing this advantage in FFT computation this lead to reduce the area, latency and power. Instead of storing actual twiddle factors in a ROM, dedicated memory bank are needed to store actual twiddle factor angles for butterfly operation.

II. Fast Fourier Transform

The Fast Fourier Transform (FFT) is an efficient algorithm which is used to calculate the Discrete Fourier Transform (DFT). The slow and complex computation of DFT is given by the equation as,

$$X(k) = \sum_{n=0}^{N-1} x(n)\, W_n^{kn} \quad where,\ k = 0,1,2,3,\ldots\ldots N\text{-}1 \qquad (1)$$

where x(n) is the input sequence of real and complex numbers, $W_n$is the complex number of twiddle factor and X(k) is the computation which involves the multiplications and summations of complex numbers.Cooley und Tukey [11] proposed FFT algorithm in 1965, in which they used repetition in the DFT calculation and it also leads to minimize multiplications and additions. Where time domain is converted into frequency domain and vice versa, primarily in signal processing, such conversion is obtained with the help of Fourier Transform (FT) and discrete fourier transform. Where input signals are sampled into time domain, so discrete fourier transform takes place in the world of digital. In discrete input signal, DFT is applied as an input and, frequency characteristics are obtained as an output.

In mathematical form, while performing inverse DFT, which is very similar to the standard DFT on frequency domain, which give backs the result as time domain.ie., various frequency components of the input signal are given, when the signals are converted into frequency domain, which leads to removal of certain unwanted frequency components. These concepts are used in image or audio compression and filters on communication signals. Based on summation, a finite series of products of input signal values and trigonometric functions, Discrete Fourier Transform (DFT) is a very computationally intensive process for direct computation. These set of algorithms are collectively called as Fast Fourier Transform, which is very efficient in terms of computation. The computational efficiency of FFT algorithm increases, when the value of 'N' increases. In FFT, the N point DFT x(n) is splitted into even numbered and odd numbered samples, the above equation (1) can be written as,

$$X(k) = \sum_{m=0}^{\frac{N}{2}-1} x(2m)\, W_n^{2km} + \sum_{m=0}^{\frac{N}{2}-1} x(2m+1)\, W_n^{k(2m+1)} \qquad (2)$$

Let the N point DFT is splitted into n/2 point data sequences. Let f_1 (m) contains even number of samples and f_2 (m) contains odd number of samples, hence above equation becomes,

$$X(k) = \sum_{m=0}^{\frac{N}{2}-1} f_1(m)\, (W_N^2)^{km} + \sum_{m=0}^{\frac{N}{2}-1} f_2(m)\, (W_n^2)^{km}.W_N^k \qquad (3)$$

$$X(k) = \sum_{m=0}^{\frac{N}{2}-1} f_1(m)\, W_{N/2}^{km} + W_N^k \sum_{m=0}^{\frac{N}{2}-1} f_2(m)\, W_{N/2}^{km} \qquad (4)$$

N/2 point DFT of f_1 represents first summation and N/2 point DFT of f_2 represents second summation. This splitting of sequence is called decimation, hence input sample is done on time domain sequence it is called Decimation in time (DIT).where W_N^k is the twiddle factor,

$W_N^k = e^{-j2\pi k/N} = \cos\frac{2\pi k}{N} - j\sin\frac{2\pi k}{N}$.

$$X(k) = F_1(k) + W_N^k\, F_2(k), \quad k = 0, 1,\ldots,N\text{-}1 \qquad (5)$$

Since $F_1(k)$ and $F_2(k)$ are periodic with period N/2, then $F_1(k + N/2) = F_1(k)$ and $F_2(k + N/2) = F_2(k)$ and the factor becomes, $W_N^{k+N/2} = -W_N^k$. hence above equation can be written as,

$X\left(k+\frac{N}{2}\right) = F_1(k) - W_N^k F_2(k)$ (6)

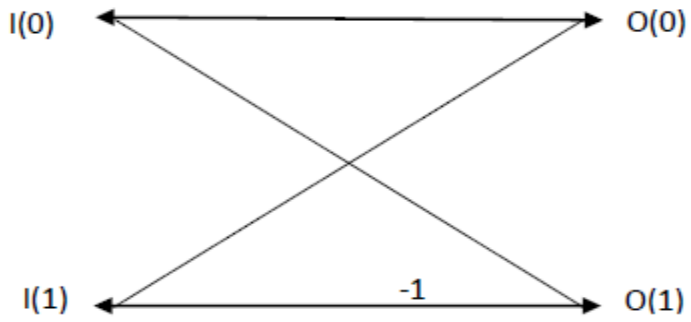

Figure 2.1: Radix-2 Butterfly

The fundamental or basic computation in FFT is called butterfly computation. It produces two complex outputs that become butterfly inputs in the next stage. From one stage to next stage, the number of input and output remains the same (N/2). Radix-2 or Radix-4 butterflies are called as the elementary 2-point and 4-point FFTs, which are implemented by simple addition and subtraction operations. The number of points with power of 4 determines the radix-4 butterfly computation to calculate FFT. The number of points with power of 2 determines the radix-2 butterfly computation to calculate FFT. The radix-2 consists of 1 and -1 which can be easily converted to additions and subtractions. The radix-4 additionally requires complex multiplication with j and –j, the real and imaginary parts of the operands are mixed, still it is possible to calculate the Radix-4 butterfly entirely by additions and subtractions. Radix 8 requires the actual multiplication, which makes them less attractive. If more elementary operations are required for radix-4 butterfly, it provides an overall reduction of operation in radix-4 FFT algorithm, more number of butterfly operations are required for radix-2 algorithm. The required number of points with power of 4, which is the best choice for the butterfly operations.

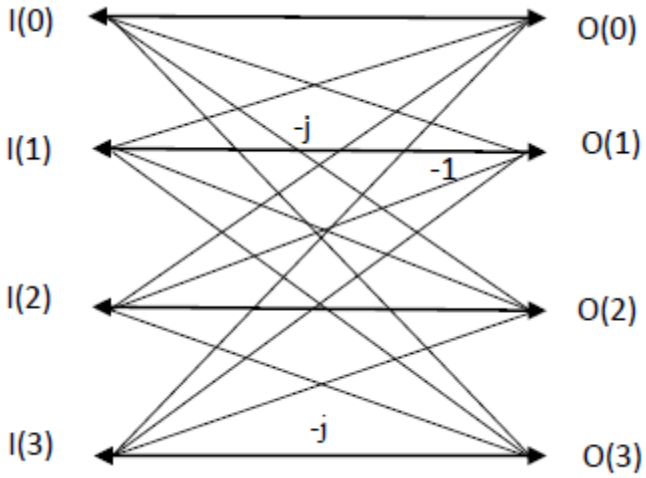

Figure 2.2: Radix-4 butterfly

Butterfly unit requires multipliers, addition and subtraction unit for computation and this acquire large area, cost and power for large point of FFT. To reduce cost, area and power and to increase speed, an unavoidable demand of today is to use CORDIC algorithm in butterfly unit.

III. CORDIC II – OVERVIEW

CORDIC II algorithm which differs from ordinary CORDIC algorithm in the used set of micro rotation, this new group of micro rotation provides fast convergence in the rotation angle. The basic angle sets proposed for CORDIC II algorithm includes: friend angles, uniformly scaled redundant (USR) CORDIC, nano rotation. This algorithm consists of six rotation stages, which is connected in series, with the basic angle sets mentioned above.

For connecting the rotation stages, it consists of several rotation stages which is connected in series, and it is characterized by an input $[-\alpha_{in}, \alpha_{in}]$ and output $[-\alpha_{out}, \alpha_{out}]$ for each rotation stages. Any number of rotation angles include in the rotation stages, where each input is rotated by one of these angles. It can be defined byN-rotator as a rotator with N different angles. If N is even, it includes N/2 coefficients and the values of δi are defined as

$$\delta_1 = \alpha_1$$

$$\delta_i = \frac{(\alpha_i - \alpha_{i-1})}{2}, \text{ i= 2, . . . ,N/2 (7)} \longrightarrow$$

According to this, the angles of input and output are,

$$\alpha_{out} = \max_i(\delta_i) \qquad i = 1, \ldots\ldots, \frac{N}{2} \quad (8) \longrightarrow$$

$$\alpha_{in} = \alpha_{\frac{N}{2}} + \delta_{\frac{N}{2}+1} = \alpha_{\frac{N}{2}} + \alpha_{out} \qquad (9) \longrightarrow$$

The best instance happens when all of the values of δi are equal, i.e., δi= φ, where φ is a constant, which minimizes the remaining angle to the next stage α_{out}. Under these conditions, $\alpha_{out} = \varphi = \alpha_{in}/N$, and the rotation angles are $\alpha i = (2i-1)\alpha_{in}/N$.

If N is odd, it includes (N –1)/2 coefficients with their conjugates α =0◦. The values of δi are defined as

$$\delta_i = \frac{(\alpha_i - \alpha_{i-1})}{2}, \quad \text{i= 2, . . . ,(N-1)/2} \qquad \text{(10)}$$

According to this, the angles of input and output are,

$$\alpha_{out} = \max_i(\delta_i) \qquad i = 1, \ldots\ldots, \frac{(N-1)}{2}$$

$$\delta_{\frac{N+1}{2}} = \alpha_{out}$$

$$\alpha_{in} = \alpha_{\frac{(N-1)}{2}} + \delta_{\frac{(N+1)}{2}} = \alpha_{\frac{(N-1)}{2}} + \alpha_{out} \qquad (11) \longrightarrow$$

The best instance also happens when all of the values of δi are equal, leading to $\alpha_{out} = \varphi = \alpha_{in}/N$. In this case, the rotation angles are $\alpha i = 2i\alpha_{in}/N$. From this analysis, several conclusions can be obtained. First, when the rotation angles are selected carefully, an N-rotator reduces the input by a factor N because $\alpha_{out} = \varphi = \alpha_{in}/N$. This thing proves the efficiency of the CORDIC rotator, where many of the microrotations halve the rotation angle using a 2-rotator. Second, in order to design efficient rotators, with $\alpha_{out} \approx \varphi = \alpha_{in}/N$. α_{in} must be larger than α_{out} for each stages in order to connect in series. This guarantees the convergence of the rotation angle, which leads to a reduced latency and a smaller number of adders.

IV. IMPLEMENTATION

Due to higher data rates in communication systems, large-point FFTs are required such as 1024/2048/4096 etc. In this type of implementation ROM takes most of the chip area, consumes

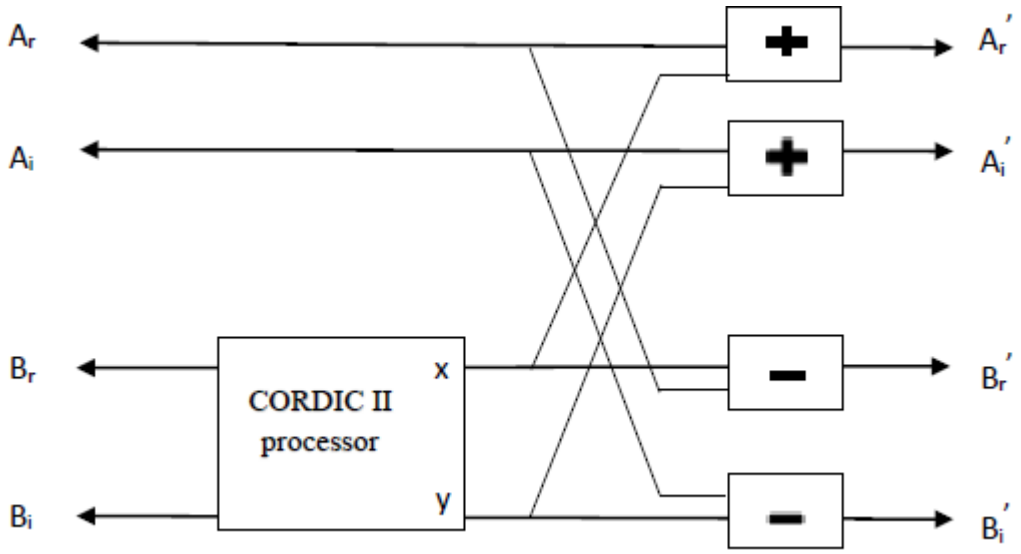

Figure 4.1 : CORDIC II based FFT

more power and limits the speed. Hence poor performance of the FFT is obtainedin terms of power, speed, and area. Where CORDIC II algorithm leads to reduced latency and a smaller number of adders. While implementing this advantage in FFT computation it reduces the area, delay and power. FFT consists of routing unit and processing unit, where Processing unit is for computation of equations and routing unit is for data recirculation and management purpose. CORDIC II based FFT processor needs a memory bank to store twiddle factor angles for butterfly operations, instead of storing actual twiddle factor in ROM. In this work, processing unit is replaced by CORDIC II. The real and imaginary parts of X (k) is taken as the initial values for x and y in CORDIC II, where FFT is designed for 1024 points with radix-4 structure.

V. RESULTS AND DISCUSSIONS

The simulated output for test bench waveform of 1024 point FFT processor using CORDIC II algorithm in shown in figure 4, which is tested for real data inputs. First, data loading process is done, then after computation output data loading is done for both real and imaginary terms.

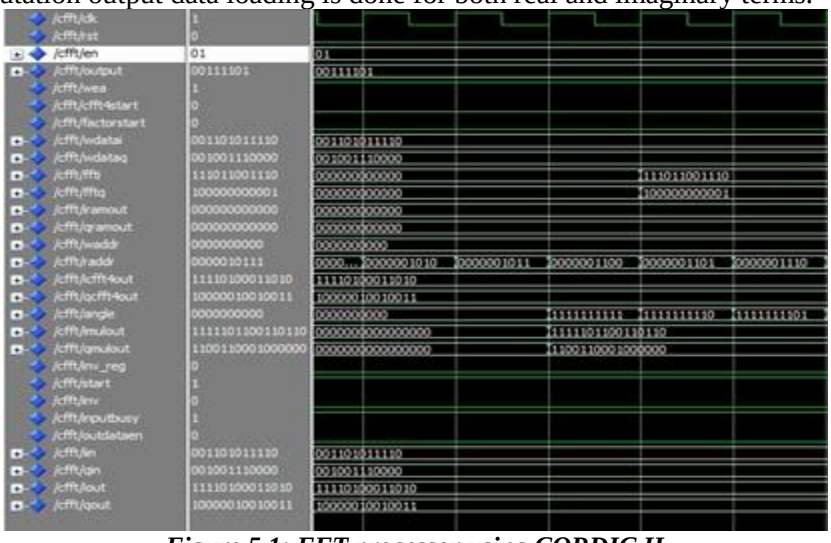

Figure 5.1: FFT processor using CORDIC II

Figure 5 shows the RTL schematic of CORDIC II algorithm based FFT. These RTL schematic are basic logical representation of the circuit in terms of logic primitives which are generated when the design become correct in simulation and synthesis level.

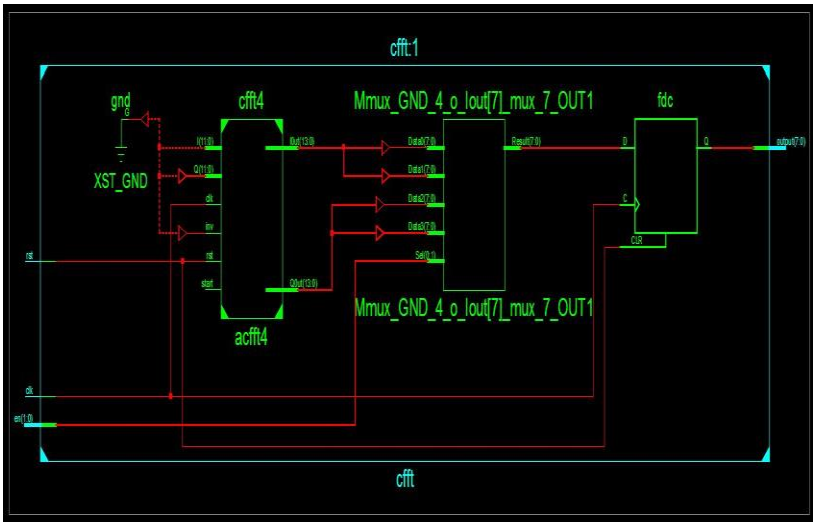

Figure 5.2 :RTL schematic of CORDIC II based FFT

Device	
Family	Artix7
Part	xa7a100t
Package	csg324
Temp Grade	Industrial
Process	Typical
Speed Grade	-2I

On-Chip	Power (W)	Used	Available	Utilization (%)
Clocks	0.000	1	---	---
Logic	0.000	285	63400	0
Signals	0.000	342	---	---
IOs	0.000	76	210	36
Leakage	0.042			
Total	0.042			

Supply Summary	Total	Dynamic	Quiescent
Source	Voltage	Current (A)	Current (A)	Current (A)
Vccint	1.000	0.017	0.000	0.017
Vccaux	1.800	0.013	0.000	0.013
Vcco18	1.800	0.001	0.000	0.001
Vccbram	1.000	0.000	0.000	0.000

	Total	Dynamic	Quiescent
Supply Power (W)	0.042	0.000	0.042

Environment	
Ambient Temp (C)	25.0
Use custom TJA?	No
Custom TJA (C/W)	NA
Airflow (LFM)	250
Heat Sink	Medium Profile
Custom TSA (C/W)	NA
Board Selection	Medium (10"x10")
# of Board Layers	12 to 15
Custom TJB (C/W)	NA
Board Temperature (C)	NA

Thermal Properties	Effective TJA (C/W)	Max Ambient (C)	Junction Temp (C)
	3.3	99.9	25.1

Characterization	
Advance	v0.7,2012-04-23

Figure 5.3: Power Utilization

Table. 1 shows the device utilization summary of CORDIC II algorithm based FFT, which describes the hardware resources on an FPGA are indicated by the number of slices that FPGA has, where slice is just a grouping of LUTs and FFs some connective structure into the repeated unit of FPGA.

Table 5.1: Device Utilization

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	79	126800	0%
Number of Slice LUTs	318	63400	0%
Number of fully used LUT-FF pairs	73	324	22%
Number of bonded IOBs	76	210	36%
Number of BUFG/BUFGCTRLs	1	32	3%

VI. CONCLUSION AND FUTURE SCOPE

A new improved CORDIC II algorithm based FFT computation was designed using VHDL to realize the operation with reduced memory requirements. Due to demand in higher data rates in communication systems, large-point FFTs are required such as 1024/2048/4096 etc. Where CORDIC II based FFT computations effectively achieve iteration process by shift and add operations. The proposed design overcomes with the existing designs in terms of both speed and area. The designed CORDIC II based FFT can meet the speed specifications of most OFDM communication systems, including VDSL, 802.16, DAB and DVB.

REFERENCES

[1] J. E. Volder, "The CORDIC trigonometric computing technique," *IRETransaction Electronic Computer.*, vol. EC-8, no. 3, pp. 330–334, Sep. 1959.
[2] C.-S. Wu and A.-Y. Wu, "Modified vector rotational CORDIC (MVR-CORDIC) algorithm and architecture," *IEEE Trans. Circuits Syst.II*, vol. 48, no. 6, pp. 548–561, Jun. 2001.
[3] R. Shukla and K. Ray, "Low latency hybrid CORDIC algorithm," *IEEE Trans. Computer.*, vol.63, no. 12, pp. 3066–3078, Dec. 2014.
[4] C.-S. Wu and A.-Y. Wu, "Modified vector rotational CORDIC (MVR-CORDIC) algorithm and architecture," *IEEE Trans. Circuits Syst.II*, vol. 48, no. 6, pp. 548–561, Jun. 2001.
[5] S. Aggarwal, P. K. Meher, and K. Khare, "Area-time efficient scaling free CORDIC using generalized micro-rotation selection," *IEEE Transaction Very Large Scale Integr. (VLSI) Syst.*, vol. 20, no. 8, pp. 1542–1546, Aug. 2012.
[6]F. Jaime, M. Sánchez, J. Hormigo, J. Villalba, and E. Zapata, "Enhanced scaling-free CORDIC," *IEEE Trans. Circuits Syst. I*, vol. 57, no. 7,pp. 1654–1662, Jul. 2010.
[7] Y. Liu, L. Fan, and T. Ma, "A modified CORDIC FPGA implementation for wave generation," *Circuits Syst. Signal Process.*, vol. 33, no. 1,pp. 321–329, Jan. 2014.
[8] Mario Garrido, PetterKällström, Martin Kumm, and OscarGustafsson, "CORDIC II: A New Improved CORDIC Algorithm*," IEEE Transactions On Circuits And Systems—II*, vol. 63, no. 2, pp.186-190, Feb. 2016.
[9]Benjamin Heyne, J¨urgen G¨otze, "A Pure CORDIC Based FFT For Reconfigurable Digital

Signal Processing," *IEEE International Conference on signal processing,* pp.1513-1516, April 2015.

[10]Pooja Choudhary, Abhijit Karmakar, "CORDIC Based Implementation of Fast Fourier transform," in *Proc. IEEE International Conference on Computer & Communication Technology(ICCCT),* pp.550-555, Nov.2011.

[11] Michael Dreschmann, Joachim Meyer and Michael Hubner, "Implementation of an ultra-high speed 256-point FFT for Xilinx Virtex-6 devices," *IEEE International conference on industrial informatics,* pp.829-834, Oct.2011

[12]C.-Y. Yu, S.-G. Chen, and J.-C. Chih, "Efficient CORDIC designs for multi-mode OFDM FFT," in *Proc. IEEE International Conference Acoustic Speech Signal Process.*, vol. 3, pp. 1036–1039, May 2006.

[13] M.Garrido and J. Grajal, "Efficient memoryless CORDIC-FFT computation," in *Proc. IEEE International Conference Acoustic Speech Signal Process.*, vol. 2, pp. 113–116, April 2007.

[14] Manikandan.M and Paramasivam.C, " Area and time efficient FFT architecture using hardwired pre-shifted bi-rotation CORDIC design," *International Journal of Innovative Research in Science, Engineering and Technology.*,vol.3, pp.645-651, March 2014.

[15] M. Garrido, O. Gustafsson, and J. Grajal, "Accurate rotations based on coefficient scaling," *IEEE Trans. Circuits Syst. II*, vol. 58, no. 10, pp. 662–666, Oct. 2011.