ARCHITECTURAL IMPLEMENTATION OF CORDIC UNIT AND ITS APPLICATIONS

N PRASADD

2013

Department of Electronics and Communication Engineering

NATIONAL INSTITUTE OF TECHNOLOGY ROURKELA

Architectural Implementation of CORDIC Unit and Its Applications

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology

In

VLSI DESIGN AND EMBEDDED SYSTEM

By

N Prasad

Department of Electronics and Communication Engineering

National Institute Of Technology

Rourkela

2011 – 2013

Architectural Implementation of CORDIC Unit and Its Applications

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Technology
In
VLSI DESIGN AND EMBEDDED SYSTEM

By

N Prasad

211EC2099

Under the Guidance of
Prof. Ayas Kanta Swain

Department of Electronics and Communication Engineering

National Institute Of Technology

Rourkela

2011 – 2013

To my family, friends and faculty members

DEPARTMENT OF ELECTRONICS AND
COMMUNICATION ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY, ROURKELA
ODISHA, INDIA-769008

CERTIFICATE

This is to certify that the Thesis Report entitled "**Architectural Implementation of CORDIC Unit and Its Applications**", submitted by **Mr. N Prasad** bearing roll no. **211EC2099** in partial fulfilment of the requirements for the award of **Master of Technology in Electronics and Communication Engineering** with specialization in "**VLSI Design and Embedded Systems**" during session 2011 - 2013 at National Institute of Technology, Rourkela is an authentic work carried out by him under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other university/institute for the award of any Degree or Diploma.

Place: Rourkela
Date: June 03, 2013.

Prof. Ayas Kanta Swain
Dept. of E.C.E
National Institute of Technology
Rourkela – 769008

Acknowledgement

I would like to express my gratitude to my thesis guide **Prof. Ayas Kanta Swain** for his guidance, advice and support throughout my thesis work. I am especially indebted to him for teaching me both research and writing skills, which have been proven beneficial for my current research and future career. Without his endless efforts, knowledge, patience, and answers to my numerous questions, this research would have never been possible. The experimental methods and results presented in this thesis have been influenced by him in one way or the other. It has been a great honour and pleasure for me to do research under supervision of Prof. Ayas Kanta Swain. Working with him has been a great experience. I would like to thank him for being my advisor here at National Institute of Technology, Rourkela.

Next, I want to express my respects to **Prof. K. K. Mahapatra, Prof. S. Meher, Prof. D.P. Acharya, Prof. S. K. Patra, Prof. N. Islam, Prof. Pramod K. Tiwari, Prof. Samit Ari, Prof. Poonam Singh** for teaching me and also helping me how to learn. They have been great sources of inspiration to me and I thank them from the bottom of my heart.

I would like to thank to all my faculty members and staff of the Department of Electronics and Communication Engineering, N.I.T. Rourkela, for their generous help for the completion of this thesis.

I would like to thank all my friends and especially my classmates for thoughtful and mind stimulating discussions we had, which prompted to think beyond the obvious. I've enjoyed their companionship so much during my stay at NIT, Rourkela.

I am especially indebted to my parents for their love, sacrifice, and support. My parents are my first teachers, after I came to this world and I have set of great examples for me about how to live, study and work. I am grateful to them for guiding my steps on the path of achievements since my infanthood.

N Prasadd

Abstract

The ubiquity of digital signal processing (DSP) has made increasing demand to develop area efficient and accurate architectures in carrying out many nonlinear arithmetic operations. One such architecture is CORDIC unit which has many applications in the field of DSP including implementing transforms based on Fourier basis. This report presents architecture of CORDIC, embedded with a scaling unit that has only minimal number of adders and shifters. It can be implemented in rotation mode as well as vectoring mode. The purpose of the design is to get a scaling free CORDIC unit preserving the design of original algorithm. The proposed design has a considerable reduction in hardware when compared with other scaling free architectures. The analysis of error for different word lengths and different input ranges for fixed word length gives a better choice to choose the parameters. The error in rotation mode for 16 bit data path, obtained for ordinate equivalent input is 0.073% and for abscissa equivalent input is 0.067%. We also report architecture of a Discrete Fourier Transform (DFT) core that is implemented using low latency CORDIC. A scaling unit has been included to get scaled outputs. The reported DFT core architecture, which is for 8-point real sequence, has 22 adders in total, in addition to 2 CORDIC units. Direct Digital Synthesizers (DDSs) or Numerically Controlled Oscillators (NCOs) are nowadays prominently used in the applications of RF signal processing, satellite communications, etc. This report also brings out the FPGA implementation of one such DDS which has quadrature outputs. The proposed DDS design, which is based on pipelined CORDIC, has considerable improvement in terms of spurious free dynamic range (SFDR) compared to other existing designs at reduced hardware. The maximum sampling frequency of the proposed DDS design is 107.216 MHz. The SFDR of proposed DDS is -96.31 dBc. This report also proposes multiplier-less architecture for the implementation of radix-2^2 folded pipelined complex FFT core based on CORDIC technique. The number of points considered in the work is sixteen and the folding is done by a factor of four.

CONTENTS

LIST OF FIGURES

Figure No. Page No.

LIST OF TABLES

ACRONYMS

VLSI	Very Large Scale Integration
EDA	Electronic Design Automation
DSP	Digital Signal Processing
CORDIC	Coordinate Rotation Digital Computer
QRD	QR Decomposition
SVD	Singular Value Decomposition
FPGA	Field Programmable Gate Array
DDS	Direct Digital Synthesizer
ASIC	Application Specific Integrated Circuit
DFT	Discrete Fourier Transform
FFT	Fast Fourier Transform
SFDR	Spurious Free Dynamic Range
NCO	Numerically Controlled Oscillators
DDFS	Direct Digital Frequency Synthesis
DAC	Digital to Analog Converter
LUT	Look Up Table
ROM	Read Only Memory
PAC	Phase to Amplitude Converter
SNR	Signal to Noise Ratio
MSR	Mixed Scale Rotation
rms	Root Mean Square

SDP	Slice Delay Product
HDL	Hardware Description Language
VHDL	Very high speed integrated circuit Hardware Description Language
MATLAB	MATrix LABoratory
ISE	Integrated Services Environment
RF	Radio Frequency
FDA	Fully Dedicated Architecture
XST	Xilinx Synthesis Technology
DC	Design Compiler
OFDM	Orthogonal Frequency Division Multiplexing
DRC	Design Rule Check
ISim	ISE Simulator

Chapter 1

Overview

1.1 Introduction

The advances in the very large scale integration (VLSI) technology and the advent of electronic design automation (EDA) tools have been directing the current research in the areas of digital signal processing (DSP), communications, etc. in terms of the design of high speed VLSI architectures for real-time algorithms and systems which have applications in the above mentioned areas. The development rate of VLSI technology was predicted by Gordon Moore and since 1965 newer technologies have been developed by the industry fitting his predicted curve, which was introduced as the so called Moore's law. These advances have provided momentum to the designers for transforming algorithm into architecture. Many DSP algorithms use elementary functions like logarithmic, trigonometric, exponential, division and multiplication. Two of the ways of implementing these functions are by using table lookup method and through polynomial expansions. The above mentioned methods require large number of multiplications/divisions and additions/subtractions.

COordinate Rotation DIgital Computer (CORDIC), a special purpose computer to compute many non-linear and transcendental functions, was proposed by Volder in 1959 [1]. The functions that can be computed using a CORDIC computer include trigonometric, logarithmic, exponential, hyperbolic, multiplication, division, square root, etc. [2]. Though it initially served the purpose of navigation systems, it later became a popular tool to implement several digital systems especially in the areas of digital signal processing, communications, computer graphics, etc. [3]. The simplicity of CORDIC is that it can compute any of the above mentioned functions using shifts and additions which are of the form $x \pm 2^{-i} * y$. The operating mode and the coordinate system chosen are two key factors to compute the desired functions in the CORDIC. Many signal processing and communication systems operate CORDIC in circular coordinate system and in either of rotation or vectoring modes.

1.2 History

CORDIC was first introduced by Jack E Volder for his navigation applications in 1959. The then introduced CORDIC was applicable for evaluating the trigonometric identities that involved in plane coordinate rotation and transformation between polar and rectangular coordinates [1].

Later, in 1971, J. S. Walther unified the CORDIC that was earlier introduced by Volder for evaluating multiple elementary functions based on circular, linear, and hyperbolic coordinate systems depending on which function is to be computed [2].

Over the last 54 years, the architecture of CORDIC has seen multiple changes and multiple variants of the initially proposed algorithm have emerged [3] – [4]. Architectures such as low-latency pipelined CORDIC, mixed-scaling-rotation (MSR) CORDIC, scaling-free CORDIC have seen much usage in modern digital systems [5] – [11].

1.3 CORDIC Applications

Applications of CORDIC can mainly be seen in the following areas:

1. Matrix decomposition
2. Signal and image processing
3. Communications
4. Computer graphics
5. Robotics

In matrix decomposition, CORDIC is used to compute QR decomposition (QRD), singular value decomposition (SVD), and eigenvalue estimation [3]. In signal and image processing, CORDIC is used in fixed/adaptive filtering, computing discrete transforms of fourier basis, image enhancement operations, etc. [3]. In communications, CORDIC is mostly used in direct digital synthesis, digital and analog modulation, envelope detection, etc. [3]. In computer graphics and robotics, CORDIC is used in solving direct and inverse kinematics for robot manipulators, 3D vector rotation in computer graphics, etc. [3].

1.4 Motivation and Problem Statement

Thus, we can realize that CORDIC is one of the most determining arithmetic techniques that has found far-reaching applications in digital systems. Area efficient and power efficient systems are always the preferred choice of any designer. Thus, as CORDIC based systems are both area-efficient and power-efficient, one can design architectures for emerging digital systems based on CORDIC.

The problems that have been addressed here are the analysis of CORDIC unit for small data-width (up to 16 bits and sometimes 20 bits) and to develop a scale free CORDIC unit with the mapping mechanism that maps the angle to entire 360^{o}, efficient FPGA design of an 8-point DFT core, FPGA design of a direct digital synthesizer (DDS), and FPGA design of a folded pipelined radix-2^2 complex FFT core. The problem also addresses the application specific integrated circuit (ASIC) design of DDS and FFT core.

1.5 Alignment of the Thesis

Chapter 2 gives the overview of CORDIC and its variants. This includes the different modes of operation, coordinate systems applicable, etc. It also gives an overview of DDS, DFT, Radix-2^2 FFT.

Chapter 3 discusses the architectural implementation of the scale free CORDIC unit with corrected scale factor and proposed scaling architectures.

Chapter 4 discusses the architectural implementation of the proposed low latency scaled CORDIC based DFT core.

Chapter 5 discusses the architectural implementation of pipelined CORDIC based quadrature direct digital synthesizer (Q DDS) with improved SFDR.

Chapter 6 discusses the architectural implementation of proposed CORDIC based radix-2^2 folded complex FFT core.

Chapter 7 concludes the thesis and defines the future work related to the thesis.

Chapter 2

Background

2.1 Introduction

Coordinate Rotation Digital Computer (CORDIC) unit has become an essential and inevitable hardware block in modern engineering and scientific applications. It serves many applications such as solving trigonometric and transcendental equations, in digital signal processing (DSP) for Fourier basis based orthogonal transforms, in computer technology for 3D graphics, in digital communication systems for modulation and demodulation of the signals, etc. [1]–[4]. The conventional CORDIC was first implemented by Volder, in 1959 [1]. CORDIC works by rotating the coordinate system through constant prefixed angles until the angle is reduced to zero. Figure 2.1 shows the coordinate rotation of a vector in 2D circular coordinate system.

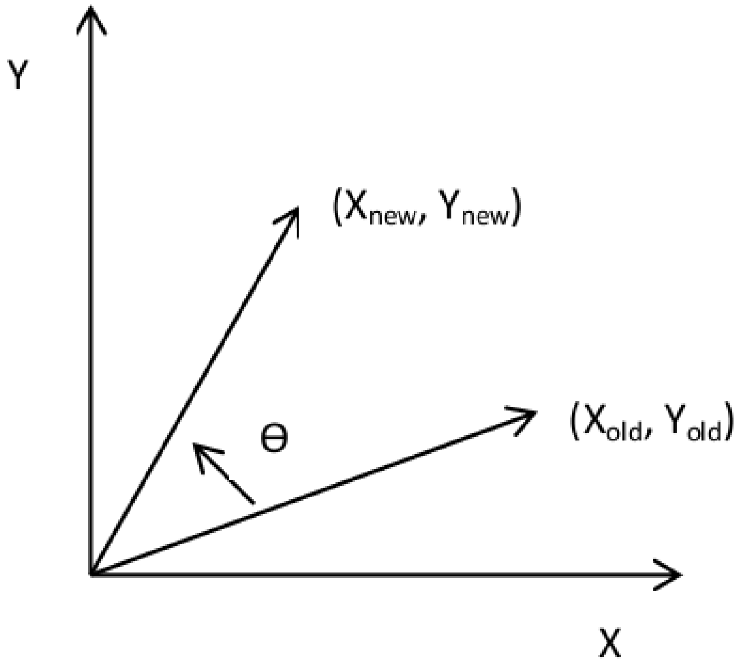


Figure 2.1. Rotation of a vector by an angle in 2D circular coordinate system

Frequency synthesizers, sometimes also called oscillators, are an essential unit of many communication systems. With the maturity of digital systems, communication systems are employing digital sub systems in their units, thus making the usage of digital systems more ubiquitous. Direct digital synthesizers (DDS) are a class of frequency synthesizers in digital domain, which generate waveforms of desired frequencies [14]. Sometimes also called numerically controlled oscillators (NCO), these generate waveforms like sine, cosine, triangular, square or rectangular, saw tooth, etc. As mentioned earlier, these have wide applications in satellite communication systems, RF signal processing, etc. Many communication systems require quadrature inputs, for example both sine and cosine, for their systems thus bringing in the need of design of DDS which can generate quadrature outputs.

This chapter gives an overview of the theory of CORDIC, DDS, DFT and Radix-2^2 FFT.

2.2 Overview of CORDIC

CORDIC, acronym of COordinate Rotation DIgital Computer, which is also known as Volder's algorithm and digit-by-digit method, is a simple and efficient method to calculate many functions including trigonometric, hyperbolic, logarithmic, etc. Its main advantage lies in its less hardware overhead and reduced latency. It performs complex multiplications using simple shifts and additions. Jack E. Volder described the modern CORDIC algorithm in 1959 for his navigation applications, which consisted of vector rotations [1]. It can be used in building low complexity finite state CPUs. Further in 1971, J. S. Walther generalized the algorithm by allowing it to calculate functions such as hyperbolic, exponential, logarithms, multiplications, divisions and square-roots [2]. The key usage of CORDIC lies in selecting it's one of the operating modes, rotation or vectoring, and one of the coordinate systems, which being either circular, or linear or hyperbolic. Usage of CORDIC in circular coordinate system results in direct outputs like and through which we can calculate other trigonometric identities. Implementing CORDIC in hyperbolic coordinate system results in functions such as and from which we can get other hyperbolic, logarithmic functions. CORDIC implementation in linear coordinate system results in calculating functions like multiplication, division. Several variants of CORDIC, both scaled and un-scaled, have been designed and among them, scale-free CORDIC architectures have gained popularity in terms of scale-factor compensation [5] – [11].

COordinate Rotation DIgital Computer (CORDIC) is an entire-transfer computer, containing a special serial arithmetic unit consisting of three shift registers, three adder-subtractors, and special interconnections [1]. It is mostly used to solve the trigonometric relationships that are involved in plane coordinate rotation and conversion from rectangular to polar coordinates and vice versa. Using a pre-determined set of conditional additions or subtractions, the CORDIC arithmetic can be controlled for solving either set of the equations given below.

$$X_{new} = K(X_{old} \cos\theta - Y_{old} \sin\theta)$$

$$Y_{new} = K(X_{old} \sin\theta + Y_{old} \cos\theta)$$

Or

$$R = K\sqrt{X_{old}^2 + Y_{old}^2}$$

$$\theta = \tan^{-1}\frac{Y_{old}}{X_{old}} \tag{2.2.1}$$

In above set of equations, K is an invariable constant. The above set of equations can be used in calculating the coordinates of the vector (X_{new}, Y_{new}) from the vector (X_{old}, Y_{old}) which is rotated by an angle of θ in a 2D circular coordinate system. This is shown in figure 2.1. The first set of equations given in equation (2.2.1) can be written in matrix form as shown below:

$$\begin{bmatrix} X_{new} \\ Y_{new} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} X_{old} \\ Y_{old} \end{bmatrix} \tag{2.2.2}$$

Rewriting equation (2.2.2) using notations we get,

$$V_{new} = RV_{old}$$

(2.2.3)

In equation (2.2.3), $V_{new} = \begin{bmatrix} X_{new} \\ Y_{new} \end{bmatrix}$, $R = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$ and $V_{old} = \begin{bmatrix} X_{old} \\ Y_{old} \end{bmatrix}$. Modifying the R matrix in equation (2.2.3) as shown below results in,

$$R = \cos\theta \begin{bmatrix} 1 & -\tan\theta \\ \tan\theta & 1 \end{bmatrix}$$

(2.2.4)

Now rewriting equation (2.2.4) we get

$$R = KR_P$$

(2.2.5)

Where, $K = \cos\theta$ and $R_P = \begin{bmatrix} 1 & -\tan\theta \\ \tan\theta & 1 \end{bmatrix}$. In equation (2.2.5), R_P is called pseudo-rotation matrix. Considering the angle of rotation θ, in the way which benefits in including in the digital systems design, we have the following.

$$\theta = \sum_{i=0}^{N-1} \sigma_i \alpha_i \; and \; \sigma_i = \pm 1$$

(2.2.6)

The rotation through the angle θ can be performed as a sequence of incremental rotations, in this context fixed rotations, in such a way that after N rotations, the sum of the incremental rotation angles nearly equals the required rotation. The residue that remains after N rotations can be ignored as it does not affect the value of the angle that has to be rotated.

The convergence range of θ for conventional CORDIC is $-1.7432 < \theta < 1.7432$. Using the non-restoring decomposition for α_i we have the ith iteration as shown below.

$$R_P(i) = \begin{bmatrix} 1 & -\sigma_i 2^{-i} \\ \sigma_i 2^{-i} & 1 \end{bmatrix}$$

(2.2.7)

The total scale factor, K, can be given as the product of the incremental scale factors obtained at each incremental rotation, as shown below

$$K = \prod_{i=0}^{N-1} K_i \approx 0.60725$$

(2.2.8)

Thus, the pseudo-rotation described in (2.2.7) can be implemented using only adder-subtractors and shifters. A small look-up table (LUT) of the size N comes into usage when we consider bit-serial implementation of the Volder's algorithm. For $N \leq 16$, the value of K deviates from the value mentioned in (2.2.8). This is further explained in [26].

The final set of equations of CORDIC in circular coordinate system is shown below.

$$X_{i+1} = \left(X_i - \sigma_i 2^{-i} Y_i\right) * K_i$$

$$Y_{i+1} = \left(Y_i + \sigma_i 2^{-i} X_i\right) * K_i$$

$$Z_{i+1} = Z_i - \sigma_i \theta_i$$

$$\sigma_i = \begin{cases} sign(Z_i), \ for\ rotation\ mode \\ -sign(Y_i), \ for\ vectoring\ mode \end{cases}$$

$$\theta_i = \arctan(2^{-i})$$

(2.2.9)

A set of generalized equations of CORDIC in 2D coordinate systems is shown below.

$$X_{i+1} = \left(X_i - m\sigma_i 2^{-i} Y_i\right) * K_i$$

$$Y_{i+1} = (Y_i + \sigma_i 2^{-i} X_i) * K_i$$

$$Z_{i+1} = Z_i - \sigma_i \alpha_i$$

$$m = \begin{cases} +1, & circular\ coordinate\ system \\ 0, & linear\ coordinate\ system \\ -1, & hyperbolic\ coordinate\ system \end{cases}$$

$$\alpha_i = \begin{cases} \tan^{-1} 2^{-i}, & circular\ coordinate\ system \\ 2^{-i}, & linear\ coordinate\ system \\ \tanh^{-1} 2^{-i}, & hyperbolic\ coordinate\ system \end{cases}$$

(2.2.10)

The value of K_i in equation (2.2.10) depends on the coordinate system chosen.

2.3 Direct Digital Synthesizer

Direct digital frequency synthesis (DDFS) or DDS uses digital hardware blocks in generating arbitrary waveforms of different frequencies. The essential hardware blocks of a simple DDS are phase accumulator, phase to amplitude converter, digital to analog converter (DAC) and a low pass (reconstruction) filter. Figure 2.2 shows DDS in its simple form.

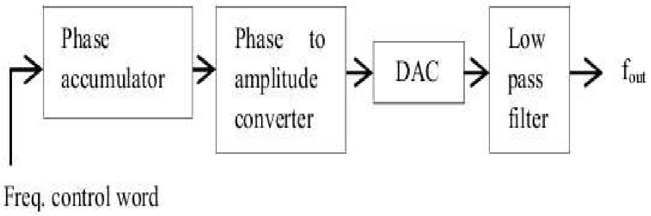


Figure 2.2. A direct digital synthesizer

The synthesizer unit has mainly two inputs, one being frequency control word and the other being system clock, which is used many times as sampling clock. The output frequency of the synthesizer depends upon the value of frequency control word.

The first block, phase accumulator, accumulates the phase at each clock cycle depending on the required output frequency of the frequency synthesizer. The input to the block, being the frequency control word determines the output frequency. If the accumulator is of N bit wide, and the period of the output wave being 2π rad, we can consider that $2^N = 2\pi$ rad. Let the sampling frequency of the system be F_s and the frequency of output is F_{out}. Depending on the required frequency (F_{out}), the step of accumulation is determined. It is determined using the following relation.

$$\Delta ACC = \frac{F_{out}}{F_s} 2^N$$

(2.3.1)

Where, ΔACC is step of accumulation. We can consider that the accumulator is a 2^N modulo adder. The output of accumulator, which is N bit wide, is fed into phase to amplitude converter that converts the input phase equivalent to output amplitude equivalent.

When ΔACC = 1, the phase accumulator accumulates with minimum accumulation step and the minimum frequency thus generated is given by

$$F_{out(min)} = \frac{F_s}{2^N}$$

(2.3.2)

The maximum output frequency of the DDS according to Nyquist sampling theorem is given by

$$F_{out(max)} = \frac{F_s}{2}$$

(2.3.3)

In practice, $F_{out(max)}$ is taken lesser than the mentioned value in equation (2.3.3).

The second block of the DDS is phase to amplitude converter. As the name suggests, this block generates the amplitude of the output waveform in form of bit strings, by considering phase as its input. There exist different methods of mapping from phase to amplitude. A number of mapping methods for generation of sine wave are given in [15]. These involve look up table (LUT) method, interpolation method, CORDIC based, etc. LUT methods require ROMs to store the phase amplitude equivalents of the wave. Many ROM based methods are discussed in [16]. One good method using analog interpolation is described in [17]. Of all methods, CORDIC based methods give better performance in terms of phase quantization noise and amplitude quantization noise.

The output of phase to amplitude converter is given as input to DAC which converts digitally represented waveform to analog one. The precision of DAC has to match with the precision of the digital output of phase to amplitude converter (PAC). Let the depth of phase input of PAC be M bits and depth of amplitude output be P bits. The two performance parameters of the DDS are spurious free dynamic range (SFDR) and signal to noise ratio (SNR).

SFDR is the ratio of strength of the desired fundamental signal to the strongest spurious signal present in the output. When the carrier level is assumed to be at 0 dB, the formula for maximum level of spurs (S_{max}), is given by

$$S_{max} = -6.02M + 3.92\ dB \tag{2.3.4}$$

Where, M is the precision of phase input of phase to amplitude converter. Thus, the formula for SFDR is given by

$$SFDR = 6.02M - 3.92\ dB \tag{2.3.5}$$

Similarly, SNR is given by

$$SNR = -6.02P - 1.76\ dB$$

(2.3.6)

Quantization effects exist in phase information as well as in amplitude information. It is observed that, SFDR is related to phase quantization and SNR is related to amplitude quantization. Observing equations (2.3.4) and (2.3.6) gives that M = P + 1. This means that spurs are caused not because of phase truncation, but because of amplitude quantization.

Several methods of improving SFDR are discussed in [18]. Since sine and cosine are of much interest nowadays, the methods to improve SFDR as well as SNR are being concentrated only on the above waveforms. Methods such as phase-dithering, noise-shaping, odd-number approach, etc. have shown considerable improvement in SFDR of the DDS.

2.4 Discrete Fourier Transform

DFT is one of the tools that perform frequency analysis of discrete time signals. Using DFT, a discrete time sequence can be represented by the samples of its spectrum, in the frequency domain. The transform is mathematically represented as follows:

$$X[k] = \sum_{n=0}^{N-1} x[n]\, e^{-\frac{j2\pi kn}{N}}$$

$$x[n] = \frac{1}{N}\sum_{k=0}^{N-1} X[k]\, e^{\frac{j2\pi kn}{N}}$$

(2.4.1)

Due to its computational complexity for direct implementation, many efficient ways have been put up. One of the most efficient and common ways to implement DFT is through Fast Fourier Transform (FFT). The reason behind implementing using FFT is its reduction in computational complexity and low latency compared to direct implementation. Many FFT architectures have been developed such as radix-4, radix-2, hybrid, split radix. For an N-point DFT, direct implementation requires $O(N^2)$ arithmetic operations, which are $N(N-1)$ additions and N^2 multiplications. Whereas for FFT, the order of arithmetic operations is$O(N\log N)$, that is $N\log N$ additions and $\frac{N}{2}\log N$ multiplications.

2.5 Radix-2^2 Fast Fourier Transform

The radix-2^2 FFT algorithm is well known for its simple structure as that of radix-2 FFT algorithm and computational complexity as that of radix-4 FFT algorithm. The generalized expression for the radix-2^2 FFT algorithm is given below.

$$X(4k_1+2l_1+l_2) = \sum_{n=0}^{\frac{N}{4}-1}\left[\left[x(n)+(-1)^{l_1}x\left(n+\frac{N}{2}\right)\right]+(-1)^{l_2}(-j)^{l_1}[x\left(n+\frac{N}{4}\right) + (-1)^{l_1}x\left(n+\frac{3N}{4}\right)]W_4^{3l_1+6l_2}\right]W_N^{n(l_1+2l_2)}W_{\frac{N}{4}}^{nk_1} \tag{2.5.1}$$

2.6 Conclusions

This chapter presented the overview of CORDIC unit, direct digital synthesizer, discrete fourier transform, and radix-2^2 fast fourier transform.

Chapter 3

Scale Free CORDIC Unit with Corrected Scale Factor

3.1 Introduction

The conventional CORDIC was first implemented by Volder, in 1959 [1]. The basic equations of the algorithm for circular coordinate system are shown below.

$$X_{i+1} = \left(X_i - 2^{-i}Y_i\right) * K_i$$

$$Y_{i+1} = \left(Y_i + 2^{-i}X_i\right) * K_i$$

$$Z_{i+1} = Z_i - \tan^{-1} 2^{-i}$$

(3.1.1)

The above set of equations is considered for positive angle of rotation. If the angle is negative, the arithmetic signs get reversed. The index i represents the number of iteration of the unit since the number of iterations depends on the precision we require. Figure 3.1 shows the CORDIC stage for ith iteration. The scaling constant for ith iteration, $K_{i,}$ is formulated as below.

$$K_i = \cos\left(\tan^{-1} 2^{-i}\right)$$

(3.1.2)

The congregate constant, obtained after all iterations is shown as

$$K = \prod_i K_i$$

(3.1.3)

Where i = 0 to N-1, N is the number of bits in the XY data path or the precision of the inputs. Thus, mathematical value of K approximates to be $K = 0.60725$.

To eliminate the hardware required to compute the above constant after performing the algorithm, many have proposed alternate algorithms [5] – [8]. The work proposed in [5], with parallel compensation of scale factor, has shown two methods namely double rotation method and bit analysis method, compensates the scaling factor in parallel while carrying out the algorithm. Though the scale factor is compensated in parallel, additional hardware such as multiplexers and adders gets added in each stage of iteration making the architecture a bulky one. The one presented in [6], called MSR_CORDIC algorithm carries out computations fastly compared to conventional CORDIC but it also has a drawback of additional shifters (2i+1 shifters) and adders which increase hardware.

The design proposed in [7], uses additional shifters and adders compared to conventional architecture. This design even depends on a parameter called basic shift, which limits the angle of rotation and more care has to be taken while mapping the angle to entire coordinate space. The above design is called modified virtually scaling free CORDIC. Another architecture using generalized micro-rotation selection is proposed in [8]. In this, they have approximated the angles of sin and cos using Taylor series expansion, as shown below.

$$\sin\alpha = \alpha - (3!)^{-1}\alpha^3 + (5!)^{-1}\alpha^5 \dots$$
$$\cos\alpha = 1 - (2!)^{-1}\alpha^2 + (4!)^{-1}\alpha^4 \dots \tag{3.1.4}$$

This recursive architecture though has better performance compared to that of others in same family has hardware overhead compared to conventional CORDIC. The advantage of this architecture is that it has lesser slice delay product compared to that of other scaling free architectures.

In enhanced scaling free CORDIC proposed in [9], they have used radix 4 booth encoding to perform the algorithm. The disadvantage of this is that it performs rotation only in one direction. From this architecture, it is evident that even this has higher hardware compared to conventional CORDIC but the advantage lies in faster computation of the vector rotation.

Thus, the architectures mentioned above for obtaining scale free CORDIC mostly have much hardware overhead compared to conventional CORDIC which makes the designers to concentrate on designing scale free architectures which have lesser or comparable hardware overhead to that of conventional CORDIC. Though latency is another issue in these designs, pipelined designs always have better latency compared to fully dedicated architectures. The proposed design has been implemented using the stages mentioned in figure 3.1.

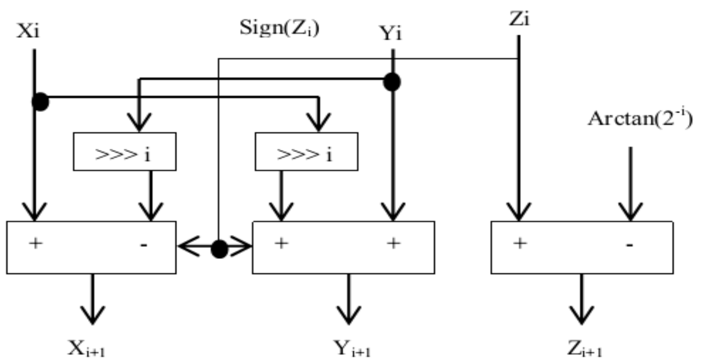


Figure 3.1. CORDIC stage for ith iteration

Figure 3.2 shows the CORDIC stage used in [7] for first half iterations.

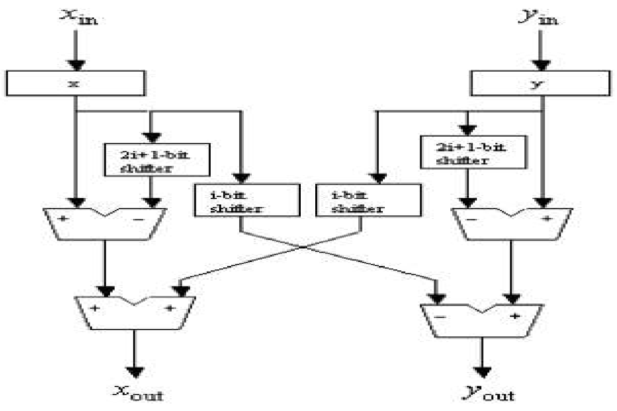


Figure 3.2. CORDIC stage for i<b/2 [7]

3.2 Proposed Architecture with Corrected Scale Factor

From (3.1.3), the value of the congregate constant is 0.60725. This is theoretical value and practical values deviate from the mentioned value and the deviation is different for different ranges of inputs. Table 3.1 shows the values of congregate scale factor for different ranges of inputs. For analysis, we considered the data path width of 16 bits.

TABLE 3.1. VARIATION OF FOR DIFFERENT RANGES OF INPUTS FOR 16 BIT DATA PATH (TAKEN USING SIMULATION)

Xin = Yin	Xout	Yout	Scale factor x	Scale factor y
15	24	26	0.6250	0.5769
63	103	104	0.6117	0.6058
511	837	846	0.6105	0.6040
4095	6718	6768	0.6096	0.6051
16383	26882	27075	0.6094	0.6051

Thus, from table 3.1, the scaling factor approaches to the mathematical value at higher width of data path, which is more complex in terms of hardware. This is also shown in figure 3.3. Thus, we approximated the scale factor for the data path with width as 16 bits and developed our CORDIC unit with the architecture for corrected scaling factor. In table 3.1, scale factor for x input and for y input is different. Thus our scaling architecture has concentrated to build separate scaling blocks for x data path as well as for y data path. The practical values considered in designing our scaling units are given below.

$$K_x = 0.6128190983$$

$$K_y = 0.6010718683$$

(3.2.1)

The values given in the equation (3.2.1) are rms values of possible scale factor values, thus making it more robust for a particular data path. Since we are following the original algorithm in developing our scale free CORDIC unit, the basic structure of CORDIC unit does not change. After the last stage of iteration, we have introduced the scaling stages, which take one extra clock cycle to scale the coefficients and to give the outputs. Since our design is implemented using pipelining, latency issues are taken care of. As seen in equation (3.2.1), there is much difference between theoretical and practical congregate constant.

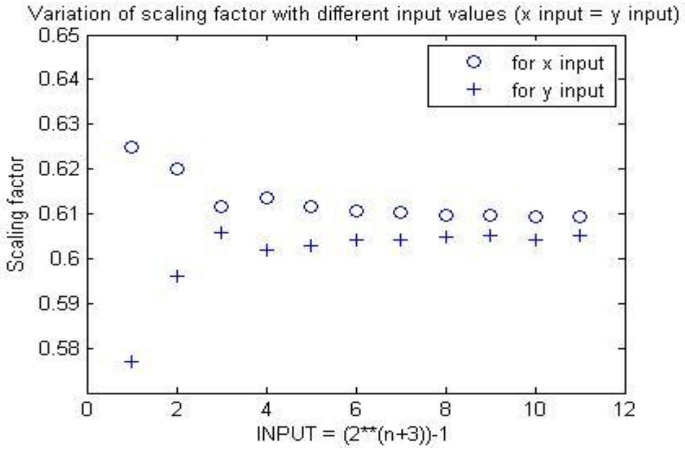


Figure 3.3. Variation of scale factor for different ranges of inputs for 16-bit data path

The scaling units are designed using hardwired shifters and adders thus minimising latency issues those arise in adding the hardware to the existing design. The scaling units are well approximated to get more accurate outputs. Figure 3.4 shows the scaling unit for X data path and figure 3.5 shows the scaling unit for Y data path.

In figures 3.4 and 3.5, '>>> i' indicates right shift by i bits. The adder compressor array can be designed using carry save architecture or ripple carry architecture, depending on speed and area requirements.

The schematic of the top module of the proposed design is shown in figure 3.6, that shows where the scalers of figures 3.4 and 3.5 fit in the design. The sub module, CORDIC_OC_PEPELINE, in the figure 6 is a cascade of 16 stages of CORDIC units, each looks like the one mentioned in figure 3.1. The architecture of sub module SCALER_X_10102012 is based on the one in figure 3.4 and that of SCALER_Y_10102012 is based on the one in figure 3.5.

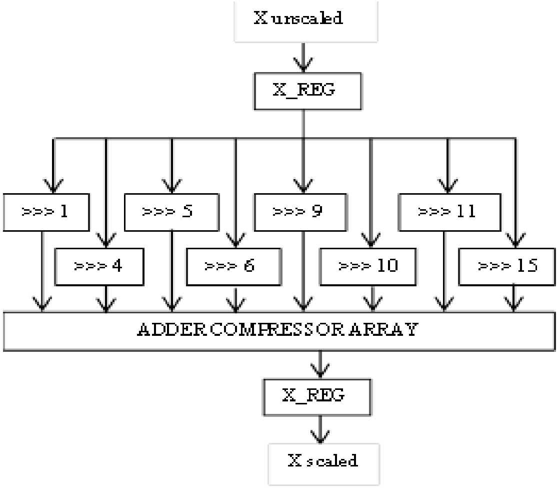


Figure 3.4. Scaling unit for X data path (16-bits)

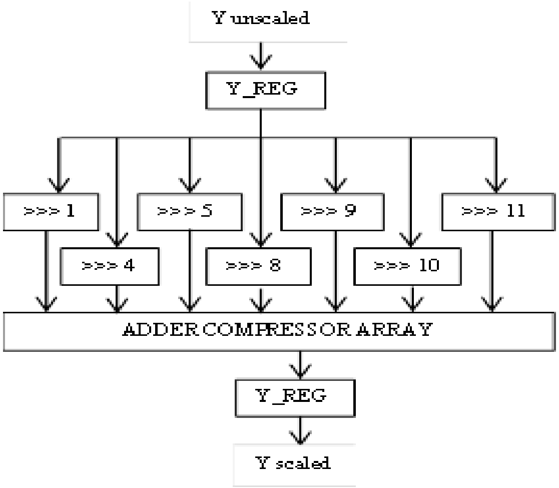


Figure 3.5. Scaling unit for Y data path (16-bits)

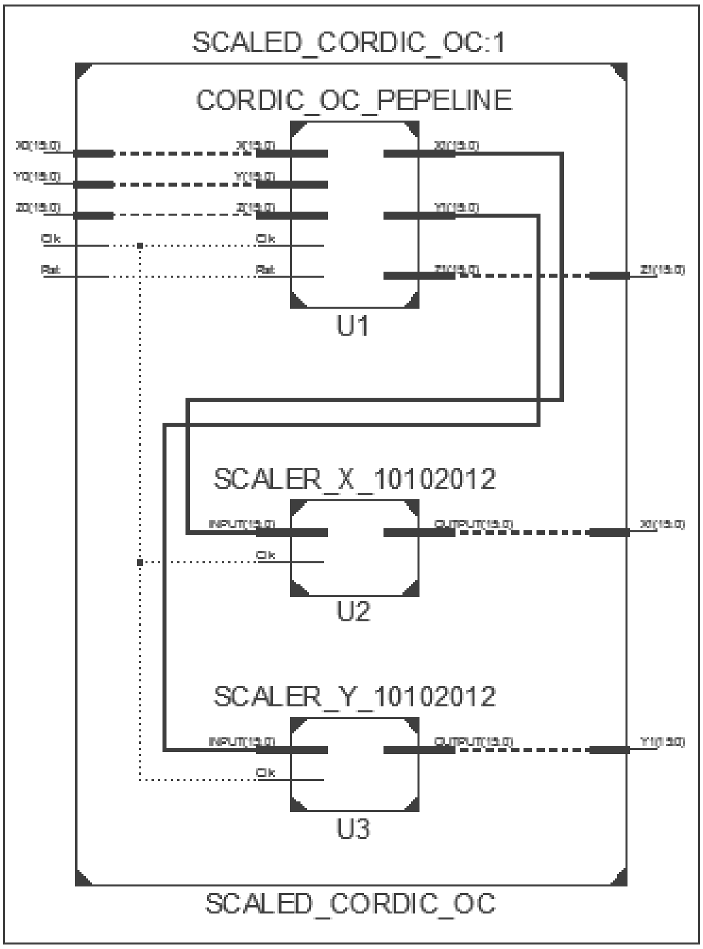


Figure 3.6. Schematic of top module of proposed design (taken using Xilinx ISE)

3.3 Results of the Proposed Design

The error analysis is done for the above scaling units after embedding them to the un-scaled CORDIC unit. Table 3.2 compares output and input for different range of inputs, showing the accuracy of scaled CORDIC unit. The same is shown in figure 3.7.

TABLE 3.2. COMPARISON OF X AND Y OUTPUTS WITH INPUTS FOR 16 BIT DATA PATH (TAKEN USING SIMULATION)

Xin = Yin	Xout	Yout	% error in x	% error in y
15	14	13	6.67	13.33
63	61	61	3.17	3.17
255	252	252	1.18	1.18
1023	1019	1018	0.39	0.49
4095	4090	4089	0.12	0.15
16383	16372	16371	0.067	0.073

Thus from table 3.2, it is evident that, increase in range of inputs increases the accuracy in scaled outputs. For the proposed design, the error between inputs and scaled outputs, for maximum range of inputs is 0.067% for abscissa equivalent input (x data path) and 0.073% for ordinate equivalent input (y data path). This can be further reduced in increasing the width of the data path.

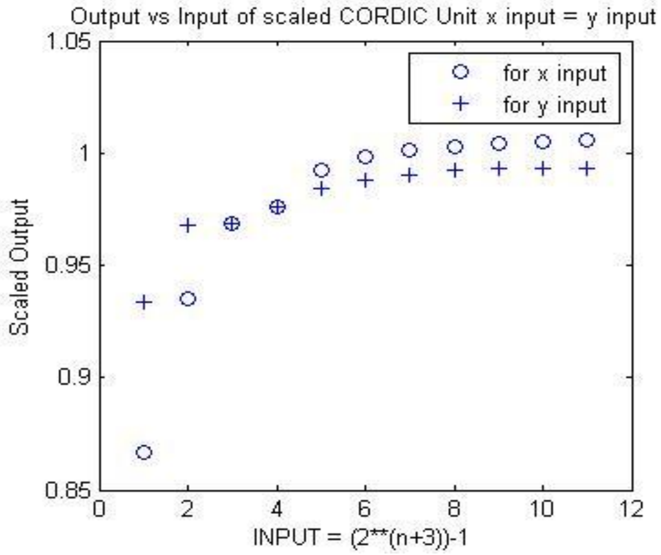


Figure 3.7. Plot of input/output vs. input for 16-bit data path

The proposed method is most suitable if the angle of rotation is fixed, since the scaling factor can be well approximated to get accurate outputs. Figure 3.8 shows the scaled output of CORDIC unit when angle of rotation is 45^0.

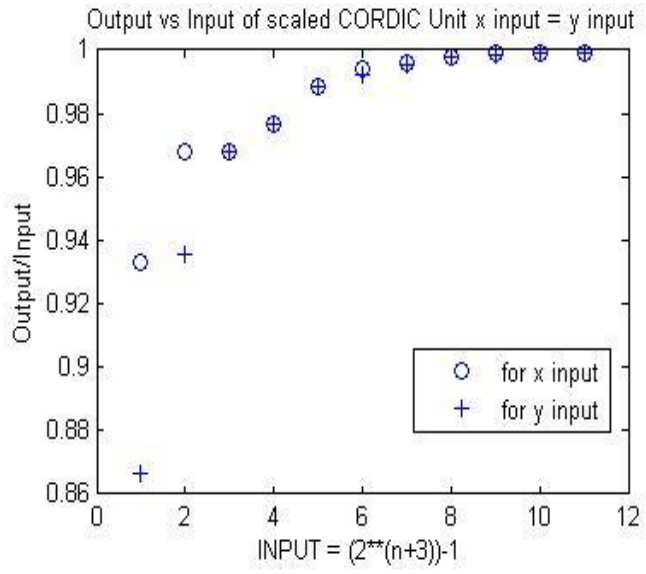


Figure 3.8. Plot of output/input vs. input for angle of 45° for 16-bit data path

Thus from figure 3.8, accuracy of more than 99.99% can be obtained. The figure 8 is plotted using the output data of Xilinx ISim for input range from 15 to 16383. The range is mentioned as X label in figure 3.8. Table 3.3 shows the device utilization summary when the design is implemented in Xilinx XC3S500E-4FG320 FPGA.

TABLE 3.3. DEVICE UTILIZATION SUMMARY OF PROPOSED DESIGN FOR 16 BIT DATAPATH

Logic utilization	**Used**	**Available**	**Utilization**
Number of Slices	503	4656	10%
Number of Slice Flip Flops	798	9312	8%
Number of 4 input LUTs	984	9312	10%

The maximum frequency of operation is 75.593 MHz. The total number of adders required by the proposed design is 62. Thus the slice delay product (SDP) can be given by

$$SDP = \frac{Number\ of\ slices * Worst\ case\ iterations}{Freq.in\ MHz}$$

(3.3.1)

As the number of worst case iterations in the proposed design is 16, the slice delay product comes to 106.465. Figure 3.9 shows the Xilinx simulation result of scaled outputs, when scaled through mathematical congregate constant. For the set of figures 3.9 and 3.10, values in first two rows represent X input and Y input respectively. Zero in third row represents the angle of rotation is 0, which is obvious. The corresponding outputs are represented in rows 6 and 7, for X and Y, respectively. Thus from figure 3.9 the error in the outputs even for maximum range of inputs is more than 3%. It shows that theoretical congregate constant cannot be considered in scaling, when the designs are done based on conventional CORDIC algorithm. Figure 3.10 shows the Xilinx simulation output of the CORDIC unit with the proposed scaling units.

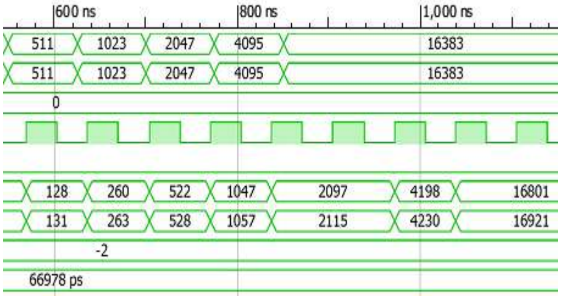


Figure 3.9. Xilinx ISE simulation result for scale factor K = 0.60725 (16-bit data path)

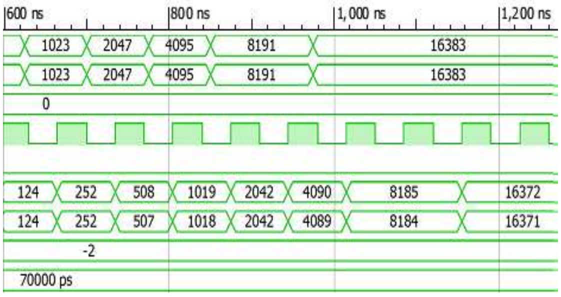


Figure 3.10. Xilinx ISE simulation result with proposed scale constants (16-bit data path)

Thus from figure 3.10, it is evident that, the proposed design of scaling units promises more accurate outputs when better range of inputs are considered. As mentioned earlier, the proposed scaling units can also be implemented while doing vectoring mode operations in circular coordinate system.

Table 3.4 shows the comparison of CORDIC units with proposed method and the one in [10], in terms of hardware, for 20 bits. The table clearly shows that CORDIC unit based on proposed method has lesser hardware as well as higher frequency. Since the design is implemented using pipeline, the count of flip flops has increased.

TABLE 3.4. COMPARISON WITH DESIGN IN [10] FOR 20 BIT DATA PATH

	[10]	**Proposed method**
4 input LUTs utilized	1907	1588
Slices utilized	984	812
Max. frequency (MHz)	56.351	61.331

3.4 Conclusions

In this chapter, we presented new scaling units for X and Y data paths separately considering the practical congregate scaling constants. The proposed approximation is word length dependent and based on requirement of accuracy, the word length of the data paths can be varied. CORDIC unit with the proposed scaling units is implemented for different ranges of inputs and error analysis is done, considering the word length of data path as 16 bits. The error of the CORDIC unit with proposed scaling unit is 0.067% for X data path and 0.073% for Y data path, thus making the design more accurate. The hardware requirement of CORDIC unit with proposed scaling units is less or comparable to that of other scale free CORDIC architectures. The maximum frequency, at which the design can be implemented, in Xilinx XC3S500E-4FG320 FPGA, fabricated in 90 nm process technology, is 75.593 MHz and its slice delay product is 106.465.

Chapter 4

Low Latency Scaled CORDIC Based DFT Core

4.1 Introduction

The usage of CORDIC is appreciated in the DFT and FFT designs since it compensates for the structures of multipliers. Much research has been done on this. The problem in them is for every butterfly unit, a CORDIC unit is required for computing the twiddle factor. The proposed design has seen very less number of CORDIC units used, as compared to the designs in [12] and [13].

Many architectural designs have been developed for CORDIC [3] - [4]. One among them, which has low latency compared to the one in [1] is explained in [11]. According to this, for rotation mode, after the CORDIC iteration j = n/2, where n is the total number of bits in the resultant vector, the x and y coordinates ($X_{n/2+1}$ and $Y_{n/2+1}$) are rotated by the remaining angle, $w_{n/2+1,}$ as follows:

$$X_f = \left(X_{\frac{N}{2}+1} + Y_{\frac{N}{2}+1} W_{\frac{N}{2}+1}\right) K_{\frac{N}{2}+1}$$

$$Y_f = (Y_{\frac{N}{2}+1} - X_{\frac{N}{2}+1} W_{\frac{N}{2}+1}) K_{\frac{N}{2}+1}$$

(4.1.1)

With logarithmic delay, using a tree of counters, the multiplication by w can be performed. The constant multiplication is done only after performing the linear approximation to rotation.

Figure 4.1 shows the conventional CORDIC unit with scaling architectures and figure 4.2 shows the low-latency scaled CORDIC unit. The difference mentioned above is clearly evident from the two figures in terms of linear approximation of rotation and the logarithmic delay of the multiplier. Section 4.2 introduces the proposed DFT core, section 4.3 shows the simulation results and discussions of the proposed design, and section 4.4 concludes the chapter.

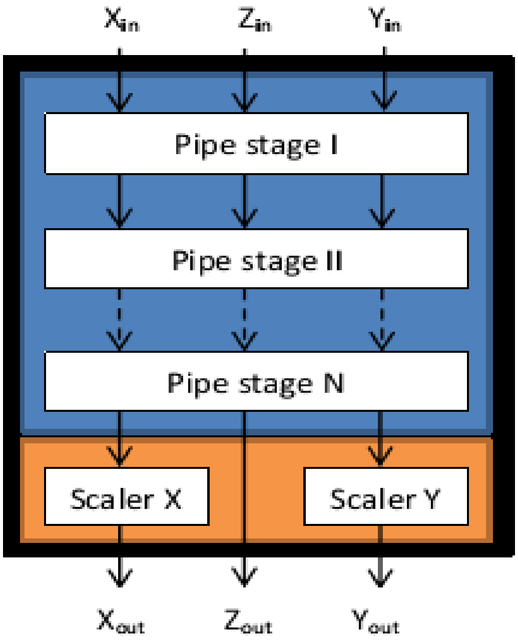


Figure 4.1. Conventional scaled CORDIC unit

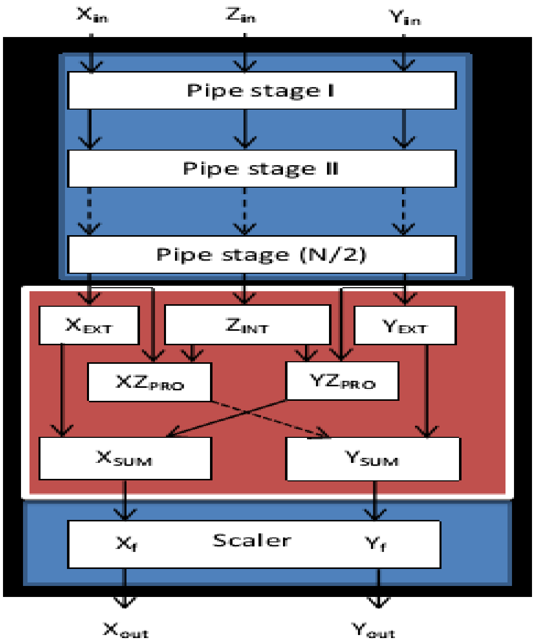


Figure 4.2. Low-latency scaled CORDIC unit

4.2 Proposed DFT Core

The proposed DFT core uses CORDIC unit in its rotation mode since the expression required using CORDIC is of the form, $x\cos\theta$ or $x\sin\theta$. The design is split into real and imaginary parts, as shown below:

$$X[k] = \sum_{n=0}^{N-1} x[n]\cos\frac{2\pi kn}{N} - j\sum_{n=0}^{N-1} x[n]\sin\frac{2\pi kn}{N}$$

(4.2.1)

By exploring the symmetry in additions and using DFT property of symmetry, two computation units are designed, each for real term and complex term.

Figure 4.3 shows the proposed architecture that computes imaginary coefficients, the ones which contain sine terms and similarly figure 4.4 shows the architecture that computes real coefficients, the ones which contain cosine terms. In both cases, one scaled CORDIC unit is used.

From the architecture, it is clear that, for computing sine terms, one CORDIC unit and 9 adder/subtractors are required, in addition to three inverters (INV), which negate the numbers based on 2's complement logic. The CORDIC unit is applied here in rotation mode.

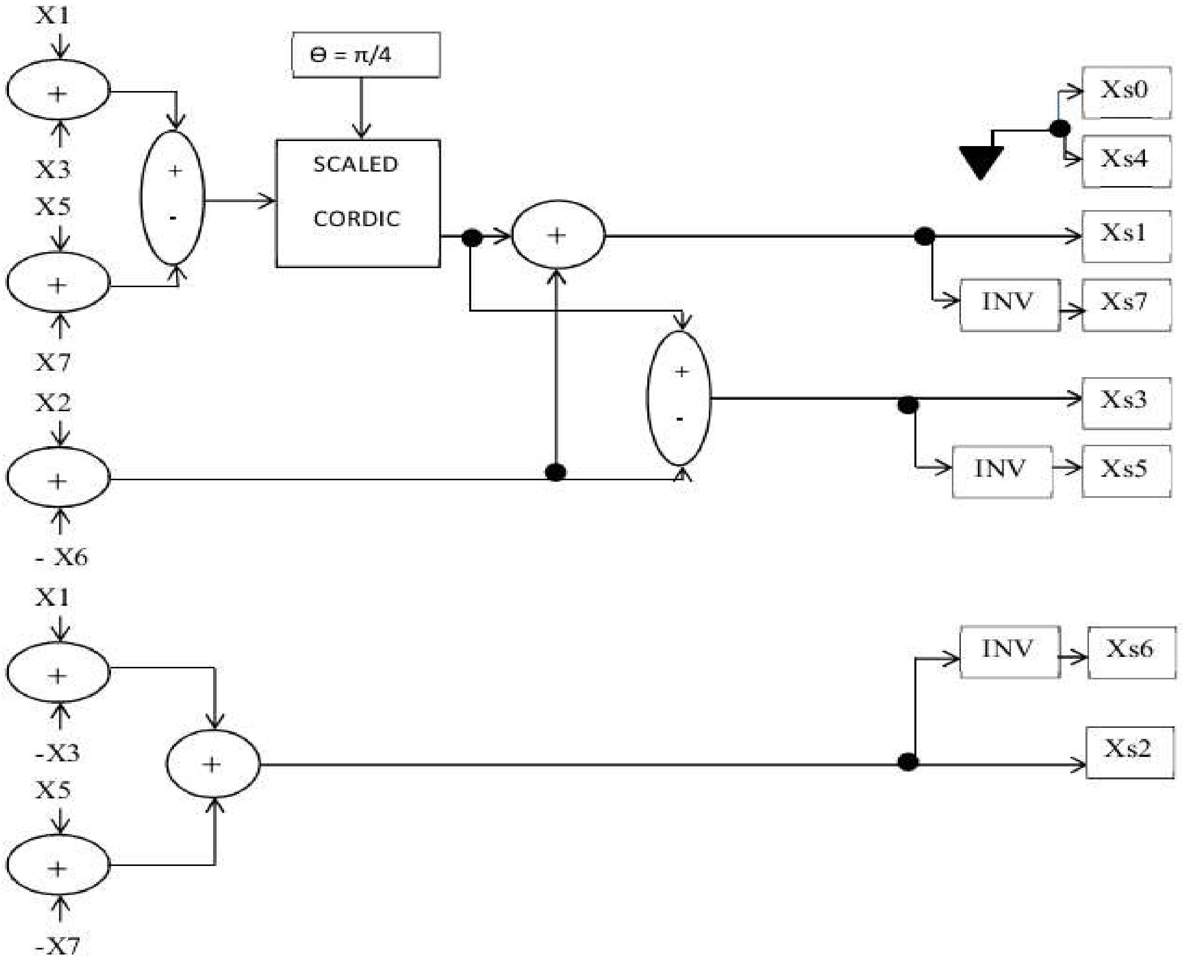


Figure 4.3. Design using CORDIC to compute imaginary coefficients

Similarly, to compute cosine terms, 13 adder/subtractors, and one CORDIC unit, are required. Thus, the total number of adder/subtractors required for 8-point DFT is 22, which is less than the number of adder/subtractors required to implement the same using FFT algorithm.

To maintain the width of the data path fixed, the scaling constant is appropriately considered [26].

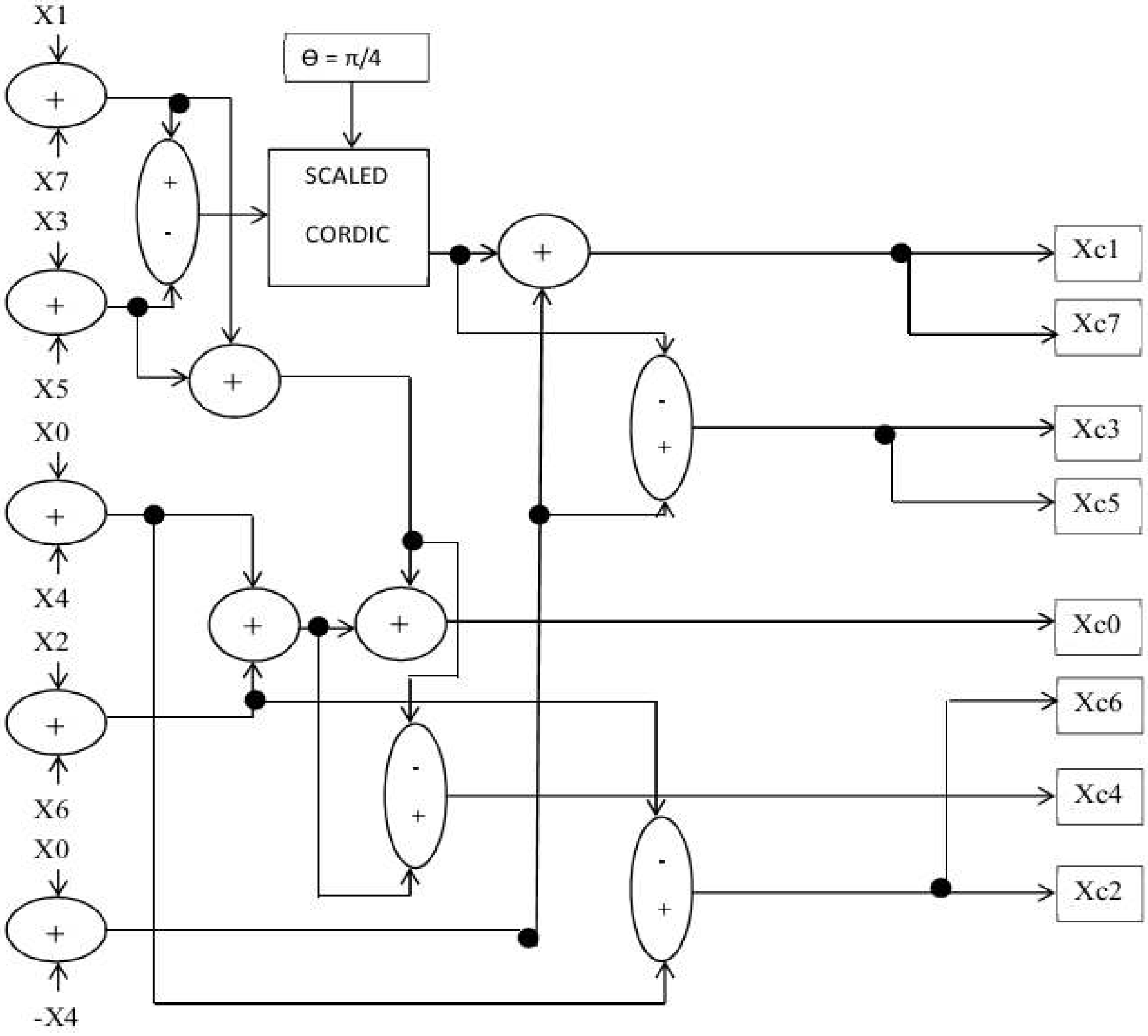


Figure 4.4. Design using CORDIC to compute real coefficients

4.3 Simulation Results and Discussion

The simulation of the proposed design is done using Xilinx ISE 10.1. The FPGA device used to map the design is XC2VP30. The hardware description language (HDL) used to write the behavioural description of the design is VHDL. All the inputs and outputs are represented using signed 2's complement number system. The width of the data path considered is 16 bits. The representation of the input vectors is considered as both integer representation and fixed-point representation. The selection of inputs is to be chosen carefully as the width of the data path is fixed. This represents that no overflow should occur while carrying out the arithmetic operations.

Figure 4.5 shows the simulation result for integer representation. Table 4.1 compares the results of outputs for the above sequence obtained in simulation, with that of in MATLAB. Table 4.2 shows the comparison between outputs obtained in MATLAB and in Xilinx ISE for fixed point representation.

TABLE 4.1. COMPARISON RESULTS OF OUTPUTS OBTAINED FOR INTEGER REPRESENTATION IN MATLAB AND XILINX ISE

MATLAB simulation results (decimal)	**Xilinx ISE simulation results (decimal)**
167	167
50.53 + 16.27 i	49 + 16 i
-13 – 18 i	-13 – 18 i
-14.53 – 41.73 i	-13 – 42 i
67	67
-14.53 + 41.73 i	-13 + 42 i
-13 + 18 i	-13 + 18 i
50.53 - 16.27 i	49 - 16 i

The number of arithmetic operations required for the proposed design is less, as mentioned earlier. Table 4.3 gives out the comparison of number of arithmetic units required to perform 8-point DFT. For higher point DFT also, the proposed method occupies less area with less hardware.

TABLE 4.2. COMPARISON RESULTS OF OUTPUTS OBTAINED FOR FIXED POINT REPRESENTATION IN MATLAB AND XILINX ISE

MATLAB simulation results (decimal)	**Xilinx ISE simulation results (decimal)**
21.2969	21.296875
-2.6587 – 8.1744 i	-3.2421875 – 9.8515625 i
-4.9570 + 6.6055 i	-5.04296875 + 6.60546875 i
8.9478 – 0.0181 i	9.046875 + 0.0078125 i
1.7266	1.726525
8.9478 + 0.0181 i	9.046875 – 0.0078125 i
-4.9570 – 6.6055 i	-5.04296875 – 6.60546875 i
-2.6587 + 8.1744 i	-3.2421875 + 9.8515625 i

TABLE 4.3. COMPARISON OF NUMBER OF ARITHMETIC UNITS REQUIRED TO PERFORM 8 POINT DFT

	Direct implementation	**FFT implementation**	**Proposed design implementation**
Adder/Subtractors	56	24	22
Multipliers	64	12	0
CORDIC rotations	0	0	2

Thus, the proposed design has less hardware requirements than the other two implementations. The design can be extended to higher point till the minimum angle of computation does not fall below the precision of the CORDIC unit. The design can be similarly extended to higher dimensions as much reduction in the hardware can be observed when there is an increase in dimension.

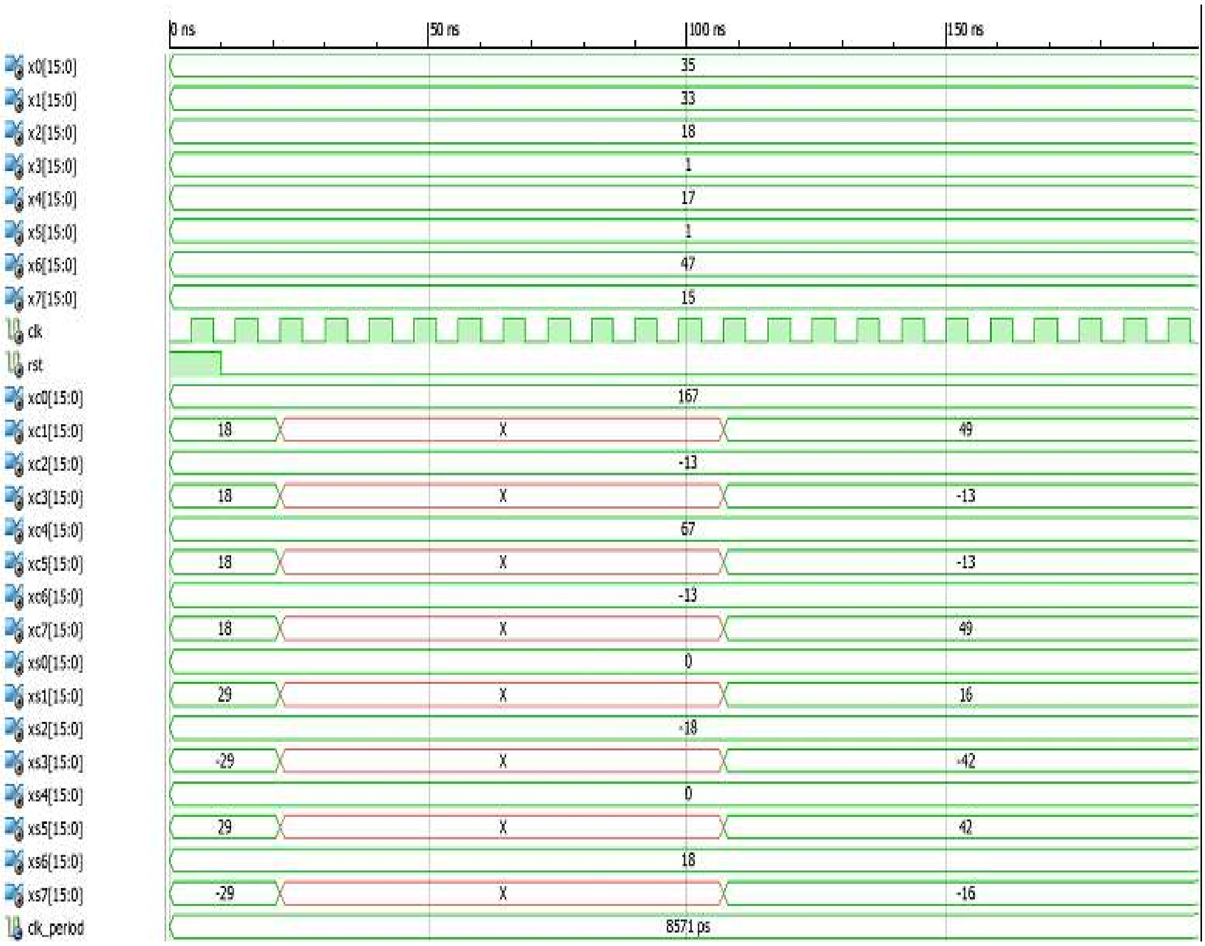


Figure 4.5 Simulation results of proposed design in Xilinx ISE for integer representation

Table 4.4 shows the device utilization summary of the proposed design using Xilinx XC2VP30 FPGA, with which the maximum frequency of operation is 157.832 MHz.

TABLE 4.4. DEVICE UTILIZATION SUMMARY OF THE PROPOSED DESIGN USING XILINX XC2VP30 FPGA

Logic utilization	**Used**	**Utilization**
Slices	614	4%
Slice Flip Flops	640	2%
4 input LUTs	1010	3%

4.4 Conclusions

We have thus proposed a low hardware complex design to implement DFT using low latency scaled CORDIC. By exploring the symmetry properties of the transform, number of additions/subtractions has been minimised so as to achieve minimal power dissipation, minimal area and improved latency. The design that is proposed has been implemented in Xilinx FPGA, XC2VP30, which is fabricated using 0.13 µm technology. The proposed design has seen very few exclusive multipliers (0 for 8-point DFT) and less number of adders compared to other traditional methods, [12] and [13].

Chapter 5

Pipelined CORDIC Based Quadrature DDS with Improved SFDR

5.1 Introduction

Frequency synthesizers, sometimes also called oscillators, are an essential unit of many communication systems. With the maturity of digital systems, communication systems are employing digital sub systems in their units, thus making the usage of digital systems more ubiquitous. Direct digital synthesizers (DDS) are a class of frequency synthesizers in digital domain, which generate waveforms of desired frequencies [14]. Sometimes also called numerically controlled oscillators (NCO), these generate waveforms like sine, cosine, triangular, square or rectangular, saw tooth, etc. As mentioned earlier, these have wide applications in satellite communication systems, RF signal processing, etc. Many communication systems require quadrature inputs, for example both sine and cosine, for their systems thus bringing in the need of design of DDS which can generate quadrature outputs.

DDS offers many advantages over analog oscillators such as extremely precise tuning resolution of the output frequency, fast hopping of phase which reduces phase related errors, remotely controllable, better match of quadrature outputs when required, etc.

The phase to amplitude block of DDS decides the nature of output of the synthesizer. The construction of such block in the proposed design is done based on pipelined CORDIC, thus generating quadrature outputs, as it has ability to generate both sine and cosine waves at a time. Other methods such as look up table method also exist in constructing the phase to amplitude conversion blocks in designing DDS.

The mode of CORDIC implemented in design of DDS' phase to amplitude block is rotation mode and the coordinate system used is circular coordinate system. The pipeline design of CORDIC facilitates the reduction in latency and also improves the speed of the design.

Pipelined CORDIC [2], has advantage over other fully dedicated architectural CORDIC (FDA) since it improves the speed of operation and also gives out varying outputs at every clock cycle, for varying inputs. The ith iteration stage of pipelined CORDIC is similar in structure to that of one in FDA except that registers are used to store the input and output values of ith iteration.

Figure 5.1 shows a stage of pipelined CORDIC. In designing dedicated pipelined architectures, one of either IN_REGS or OUT_REGS is eliminated to improve latency of the design.

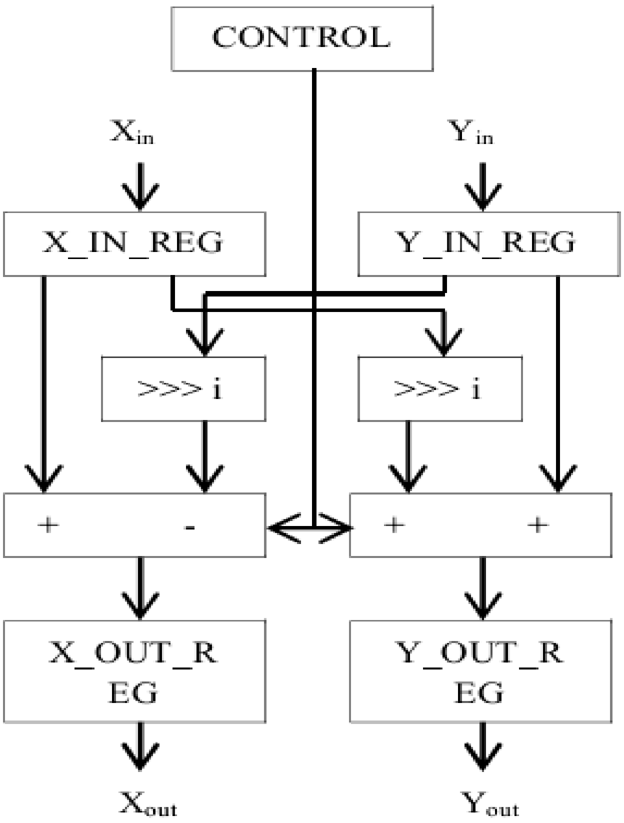


Figure 5.1. Pipelined CORDIC stage for i^{th} iteration

The rest of the chapter is organised as follows. Section 5.2 explains the proposed design methodology, section 5.3 shows the results of the proposed design and section 5.4 concludes the paper.

5.2 Proposed Design of DDS

As mentioned earlier, the proposed design of DDS is based on pipelined CORDIC, which acts as phase to amplitude converter. The phase accumulator has a precision of 18 bits, which corresponds to that of the input, which is frequency control word. The output of phase accumulator is given as input to mapped CORDIC block, which has a mapping mechanism that maps the pipelined CORDIC over the entire 2π range. The mapping mechanism works by considering the three most significant bits of the inputs. That is, depending on the values of the three MSBs, the angles are mentioned in one of the eight octants. Figure 5.2 shows the phase accumulator schematic of proposed design.

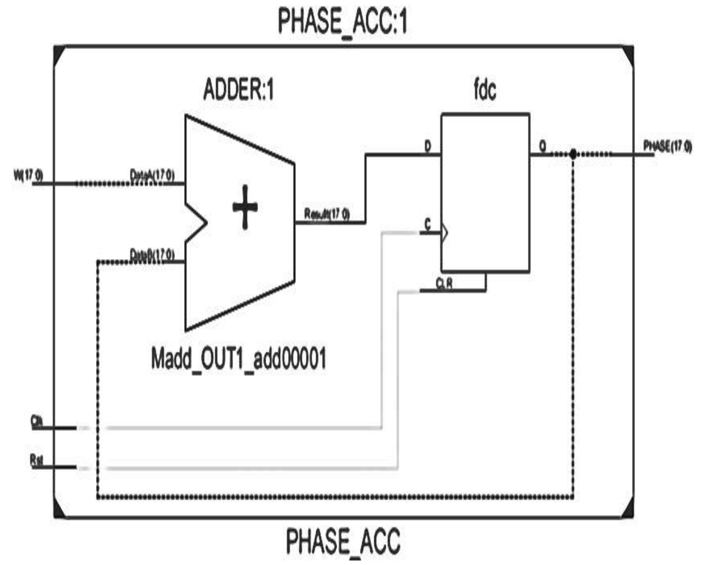


Figure 5.2. Phase accumulator unit of proposed design

The output of the phase accumulator, which is of 18 bit wide, is given as phase input to the mapped CORDIC unit. The mapper mechanism maps the CORDIC unit over entire 2π range by utilizing first three most significant bits of the phase input. The CORDIC unit utilizes only 15 bits to compute the amplitude of the quadrature wave outputs. The method used to store arctangent values inside the CORDIC unit makes the amplitude of the outputs accurate, till 4 places after the decimal.

Figure 5.3 shows one stage of pipelined CORDIC of proposed design.

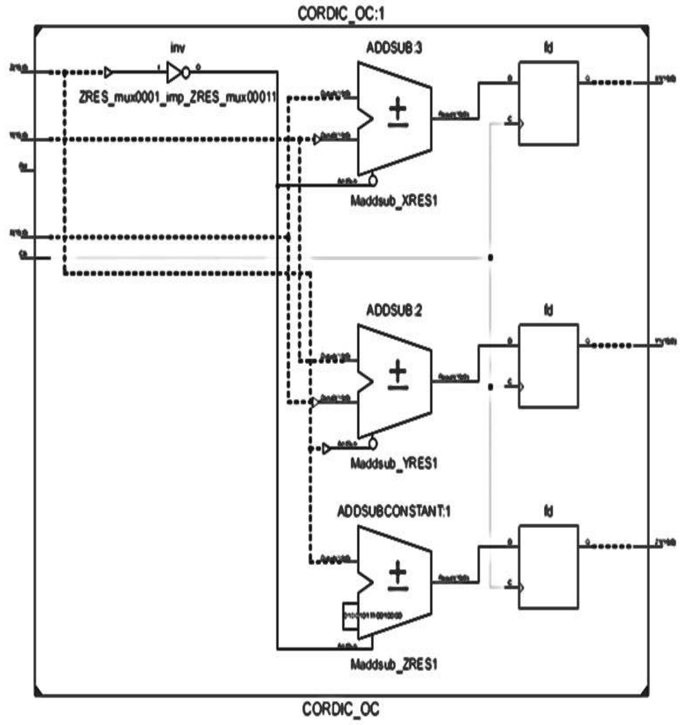


Figure 5.3. A stage of pipelined CORDIC in proposed design

The shifting in intermediate stages in the proposed design is made hardwired, thus improving latency and reducing the delay of the design. The schematic of top module is shown in figure 5.4.

The check output of the top module ensures the correct increment of the accumulation step. In the proposed design, the amplitude is represented using 16 bits. Thus, the total number of amplitude quantization levels is 2^{15} since the most significant bit represents the sign of the amplitude.

As seen from figure 5.4, there are two outputs of the proposed design corresponding to the quadrature outputs of sine and cosine. Thus, the proposed design generates quadrature outputs using single architecture. As a whole, the inputs and outputs of the proposed design are frequency control word for setting the output frequency, clock for setting sampling frequency, quadrature waves and check signal for checking accumulation index.

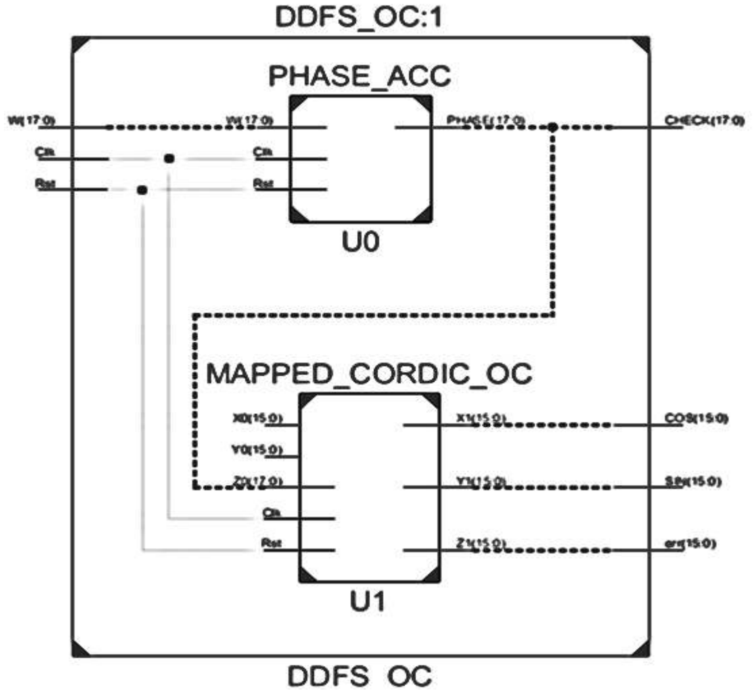


Figure 5.4. Schematic of top module of proposed design

5.3. Results of Proposed Design

The proposed design is implemented using Xilinx XC2VP30-7FF896 FPGA. The software tools used are Xilinx XST for synthesis of the design and Modelsim for simulation verification. Table 5.1 shows the device utilization summary of the proposed design when implemented on Xilinx FPGA. Since the proposed design has been designed using pipelined CORDIC, the number of registers, in turn the number of flip flops required has increased.

Figure 5.5 shows the Modelsim simulation that shows quadrature outputs of the proposed design. From the equation (2.3.5), the SFDR of the proposed design has to be 94.2 dBc. But, the calculated value came out to be 96.31 dBc. Power analysis of the proposed design is done using Xilinx Xpower Analyzer tool. Table 5.2 shows the power distribution of the proposed design for different clock frequencies.

TABLE 5.1. DEVICE UTILIZATION SUMMARY OF PROPOSED DESIGN ON XC2VP30-7FF896 FPGA

Device Utilization Summary			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	486	13696	3%
Number of Slice Flip Flops	788	27392	2%
Number of 4 input LUTs	968	27392	3%
Number of bonded IOBs	86	556	15%
Number of GCLKs	1	16	6%

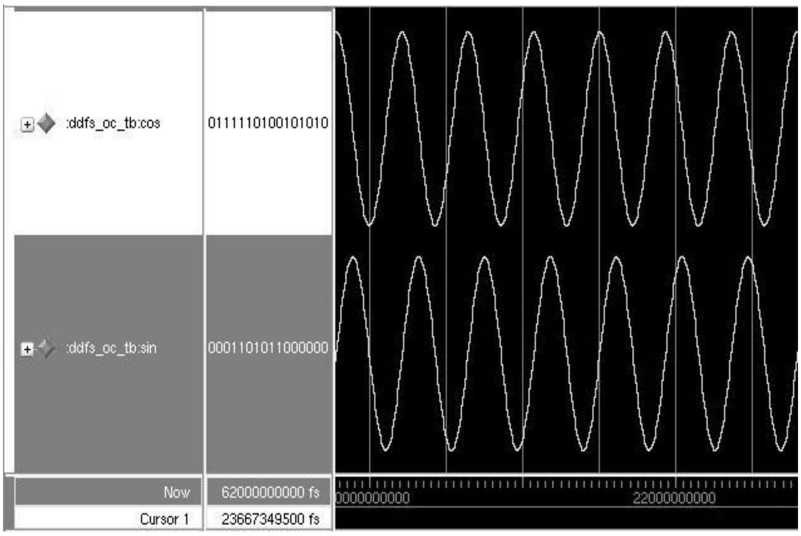


Figure 5.5. Quadrature outputs of proposed design in ModelSim

TABLE 5.2. POWER DISTRIBUTION OF PROPOSED DESIGN ON XILINX XC2VP30-7FF896 FPGA

Frequency (MHz)	Clocks (mW)	Logic (mW)	Signals (mW)	IOs (mW)
10	2.12	1.28	2.39	2.56
20	4.24	8.01	10.13	8.97
30	6.36	16.43	20.23	17.66
40	8.48	26.51	33.19	29.22
50	10.60	43.11	58.38	57.42

Figure 5.6 shows the graph of the power distribution of proposed design.

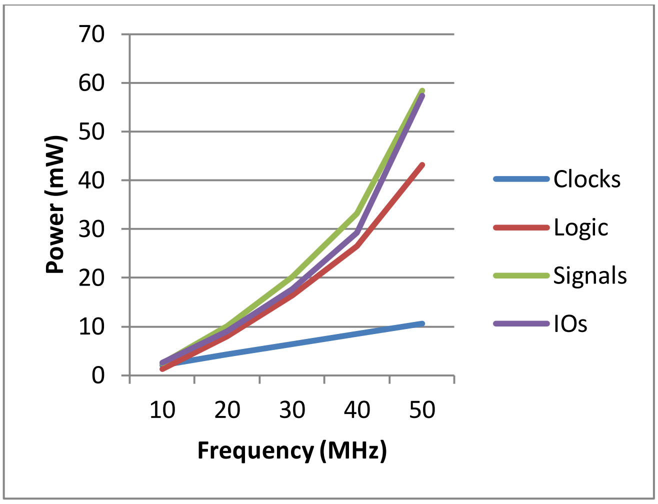


Figure 5.6. Power distribution of proposed design on SC2VP30-7FF896 FPGA

Table 5.3 shows the total power consumed by the design proposed for different clock frequencies, when mapped onto the FPGA. Figure 5.7 shows the graph of total power consumption of the proposed design.

TABLE 5.3. TOTAL POWER CONSUMPTION OF PROPOSED DESIGN ON XILINX XC2VP30-7FF896 FPGA

Frequency (MHz)	**Total Quiescent Power (W)**	**Total Dynamic Power (W)**	**Total Power (W)**
10	0.10312	0.02785	0.13097
20	0.10312	0.05642	0.15954
30	0.10312	0.09379	0.19691
40	0.10312	0.13742	0.24054
50	0.10312	0.16954	0.27266
165.9	0.10312	0.55502	0.65815

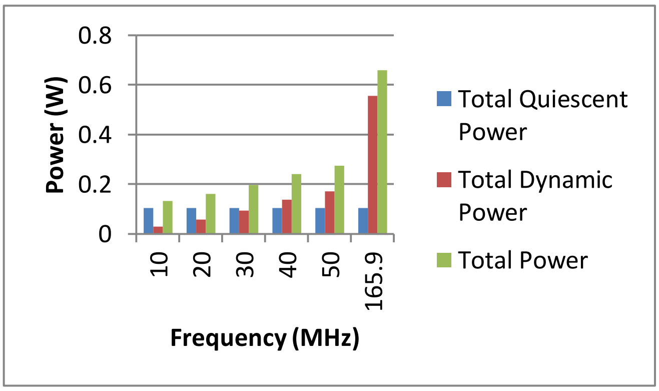


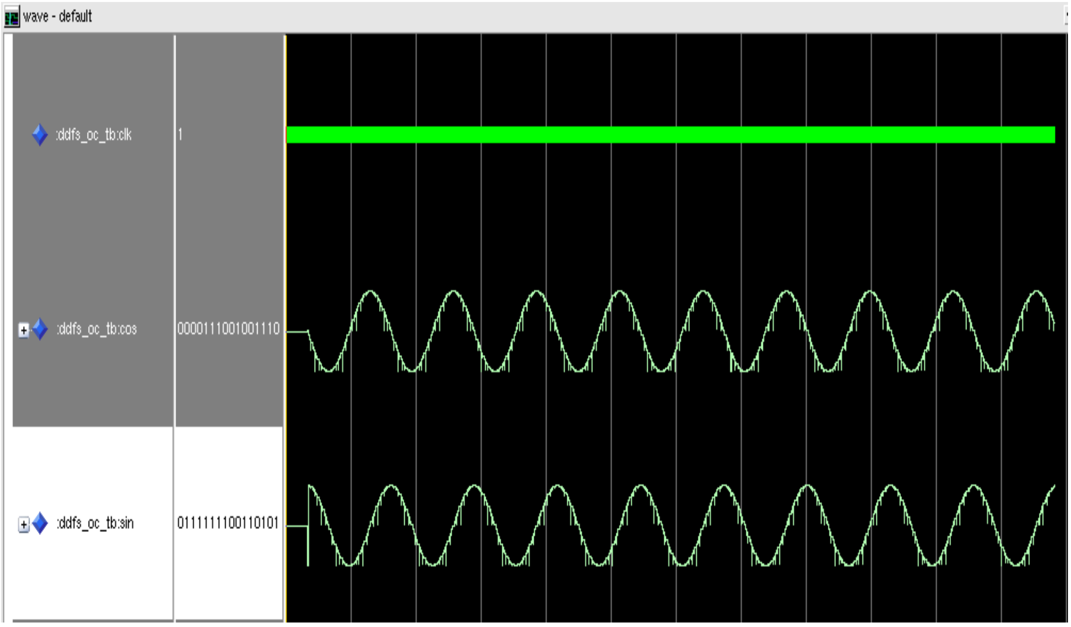
Figure 5.7. Total power consumption of proposed design when mapped on Xilinx XC2VP30-7FF896 FPGA

Table 5.4 shows the comparison of proposed design with few existing designs. From the comparison in table, the proposed design is better in all aspects. However, the performance of the proposed design can be still improved if we carefully design the phase accumulator and CORDIC's angle data path.

TABLE 5.4. COMPARISON TABLE OF PROPOSED DESIGN WITH EXISTING DESIGNS

CORDIC based DDFS	**Madisetti [19]**	**Swartzlander [20]**	**Sung [21]**	**Proposed Design**
Maximum sampling rate (MHz)	80.4	1018	100	165.939
SFDR (dBc)	81	90	84.4	96.31
Output Resolution (bits)	16	16	16	16

Figure 5.8 shows the post synthesis gate level simulation of proposed design in ASIC flow using Synopsys DC. Figures 5.9 and 5.10 show the outputs of the proposed design when implemented physically on FPGA. The outputs are seen using ChipScope Pro.

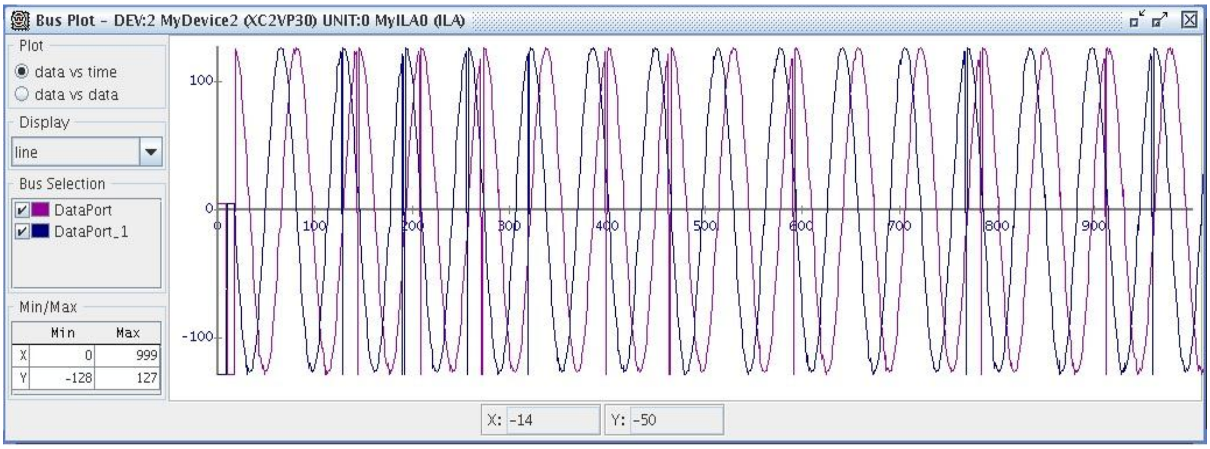


:ddfs_oc_tb:clk 1

:ddfs_oc_tb:cos 0000111001001110

:ddfs_oc_tb:sin 0111111100110101

Figure 5.8. Post synthesis mapped outputs of the proposed design (quadrature outputs)



Bus Plot - DEV:2 MyDevice2 (XC2VP30) UNIT:0 MyILA0 (ILA)

Plot

● data vs time

○ data vs data

Display: line

Bus Selection

☑ 🟪 DataPort

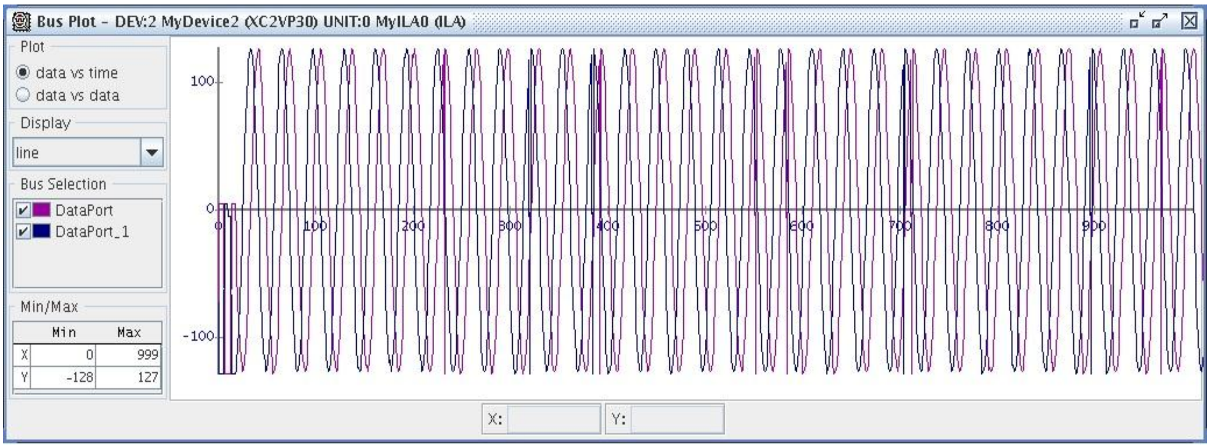
☑ 🟦 DataPort_1

Min/Max

	Min	Max
X	0	999
Y	-128	127

X: -14 Y: -50

Figure 5.9. DDS output in FPGA when EN = 01



Bus Plot - DEV:2 MyDevice2 (XC2VP30) UNIT:0 MyILA0 (ILA)

Plot

● data vs time

○ data vs data

Display: line

Bus Selection

☑ 🟪 DataPort

☑ 🟦 DataPort_1

Min/Max

	Min	Max
X	0	999
Y	-128	127

X: Y:

Figure 5.10. DDS output in FPGA when EN = 11

Figure 5.11 shows the routed layout of the proposed design using Cadence SoC Encounter. The technology used is 180 nm. Table 5.5 shows the summary of ASIC implementation of proposed design.

TABLE 5.5. ASIC IMPLEMENTATION RESULTS OF THE PROPOSED DESIGN

ASIC implementation results using Synopsys DC	**Process Technology: 0.18µm**
	Proposed DDS
Total cell area	154146.390625
Total dynamic power	13.5618 mW
Add-sub width	16 bits
Slack at 80 MHz	7.68 ns

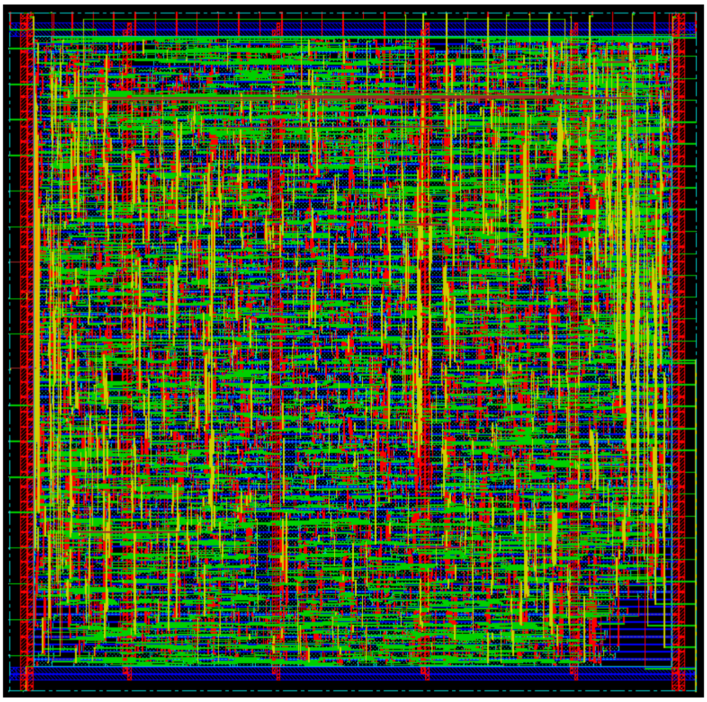


Figure 5.11. Routed layout of proposed DDS in Cadence SoC encounter

5.4 Conclusions

This chapter presents a design of DDS, which is also called a NCO. The proposed design is designed based on pipelined CORDIC unit used as phase to amplitude converter. The proposed design has a SFDR of 96.31 dBc for 16 bit output amplitude resolution. In addition to that, the proposed design produces quadrature outputs which have better phase match thus making the design more useful for most of the communication systems. The simple design of DDS based on CORDIC always is a better choice since implementing this in circuits such as mixers, digital down/up converters is simpler in terms of hardware utilization. The higher the phase accumulator data width, the lesser is the phase quantization error. Hence, amplitude accuracy of the output waveforms is improved. The maximum frequency of operation is 165.939 MHz. The slice delay product of the proposed design is 2.93. The total power consumption of the proposed design at the maximum frequency of operation is 658.15 mW.

Chapter 6

CORDIC Based Radix-2^2 Folded Complex FFT Core

6.1 Introduction

Fast Fourier Transform (FFT) is one of the most widely used algorithms in the field of digital signal processing (DSP) such as to calculate the discrete Fourier transform (DFT) efficiently, filtering, spectral analysis, etc. Many communication systems such as orthogonal frequency division multiplexing (OFDM), digital video broadcasting, etc. employ FFT cores in them [22] – [24].

Techniques in designing DSP systems such as folding, pipelining have always improved performance of the systems in terms of area of hardware, latency, frequency, etc. To determine the control circuits systematically in DSP architectures, the folding transformation is used. In folding, time multiplexing of multiple algorithm operations to a single functional unit is done. Thus, in any DSP architecture, folding provides a means for trading time for area. In general, folding can be used to reduce the number of hardware functional units by a factor of N at the expense of increasing the computation time by a factor of N [25]. To avoid excess amount of registers used in these architectures that occur while folding, there are techniques that compute the minimum number of registers needed to implement a folded DSP architecture. These techniques also help in allocation of data in these registers. Pipelining transformation in DSP architectures reduces critical path, which can be pro-sequenced to either decrease the power consumption or to enhance the sample frequency or clock speed. By introducing the pipelining latches along the data-path, critical path in particular, pipelining technique reduces the effective critical path. This technique has been used in areas such as compiler synthesis and architecture design [25].

The rest of the chapter is organized as follows. Section 6.2 elucidates the proposed design based on CORDIC. Section 6.3 shows the ASIC and FPGA implementation and comparison results of the proposed design with the existing ones. Section 6.4 draws the conclusion of the work mentioned in the chapter.

6.2 Proposed Architecture

The root designs considered for proposed design are ones mentioned in [27] – [28]. The proposed architecture is built replacing complex rotators with CORDIC blocks at required positions, which are capable of mapping the inputs to entire 2π angle. The design is a feedforward architecture which takes 4 inputs for each clock cycle. The number of outputs is also 4 for each clock cycle. The trivial rotators, which rotate by angles that are multiples of π/2, are replaced with simple blocks consisting of inverters that invert inputs according to the given angle. As the proposed architecture is based on the feed-forward architectural design using radix-2^2 FFT, for carrying out 16-point complex FFT, it requires 4 stages. Figure 6.1 shows the architectural design of the proposed architecture.

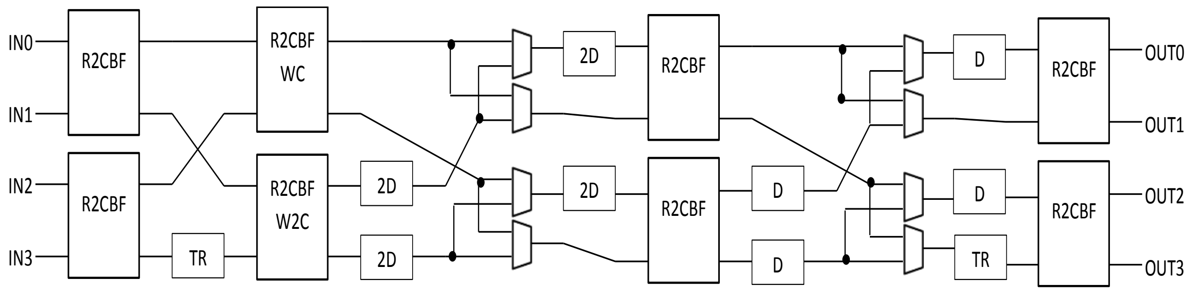


Figure 6.1. Proposed radix-2^2 4-parallel 16-point feedforward complex FFT core using CORDIC

The R2CBF block in the proposed architecture corresponds to the radix-2^2 butterfly with complex input outputs. The trivial rotations are done using the block TR. To carryout 16-point FFT, the trivial angle required is π/2. The blocks R2CBFWC and R2CBFW2C correspond to the radix-2^2 butterflies along with one M CORDIC unit, instead of complex rotator, in first case and two M CORDIC units in second case. The sets of multiplexers along with delay elements are called data shufflers, which are essential units of feedforward FFT architectures as they help in making the data to flow in a proper order. The number of delay elements and the amount of delays in each element in these depend on the stages and also on the number of inputs given to the FFT core.

The internal architecture of R2CBF is a simple radix-2 butterfly’s addition and subtraction. The internal architectures of R2CBFWC and R2CBFW2C are shown in figure 6.2 and figure 6.3 respectively. The internal architecture of delay traverse is same as that of a delay line whose size depends on the amount of delay generated by the CORDIC unit, so as to match the sequence of the data flow.

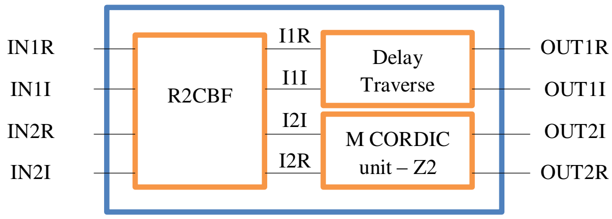


Figure 6.2. Internal architecture of R2CBFWC block

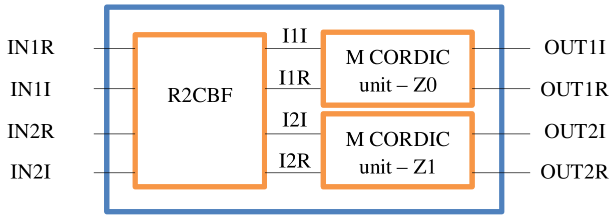


Figure 6.3. Internal architecture of R2CBFW2C block

The only difference between the architectures of R2CBFWC and R2CBFW2C is the number of M CORDIC units present in each of them. The internal architecture of the CORDIC unit is shown in figure 6.4. From figure 6.4, it is clear that the design of CORDIC is fully pipelined, except for the scalers of X and Y.

As mentioned earlier, each pipelined stage of CORDIC unit consists of three adder-subtractors along with two shifters according to the stage of the pipelined unit. The structure of each pipelined stage is given in figure 6.5.

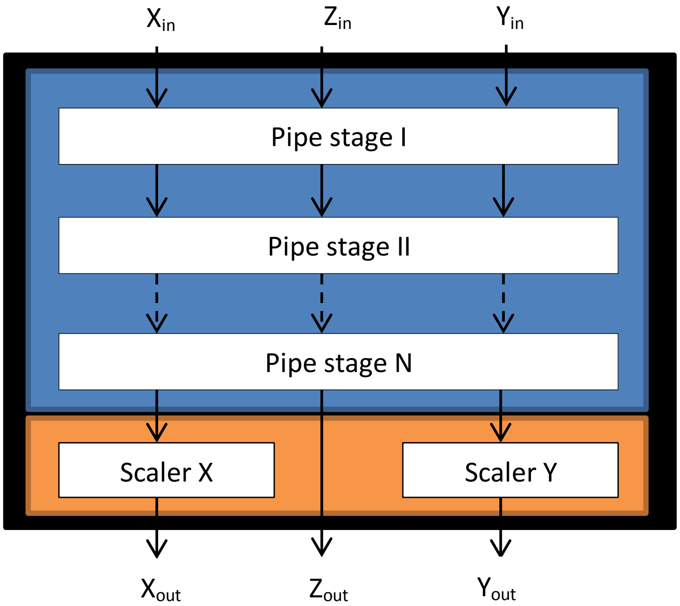


Figure 6.4. Pipelined architecture of the CORDIC unit

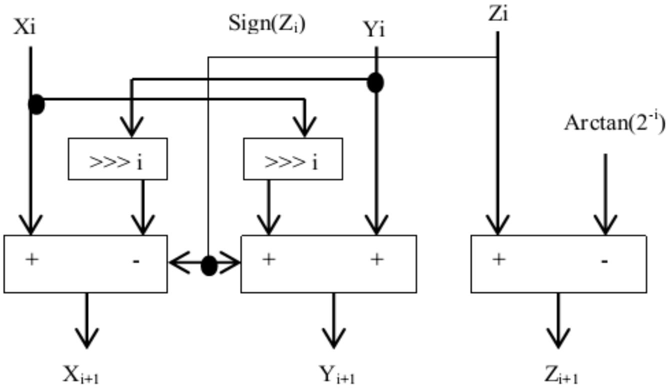


Figure 6.5. Pipe stage of the CORDIC unit

M CORDIC means mapped CORDIC unit that follows the mapping mechanism to map the inputs over entire 2π range according to the angle to be rotated. The internal architecture of the TR block is simple interchange of real and imaginary coefficients with inverting wherever necessary.

The order of inputs taken and outputs produced by the proposed architecture is shown in figure 6.6.

3	2	1	0
11	10	9	8
7	6	5	4
15	14	13	12

(a)

3	1	2	0
11	9	10	8
7	5	6	4
15	13	14	12

(b)

Figure 6.6. Data flow order in the proposed architecture of (a) inputs and (b) outputs

Since the proposed architecture is 4 parallel, the products with the twiddle factors change according to the value of the angle in them. Thus, the angle inputs of the M CORDIC units also change for each clock cycle and repeat for every 4 clock cycles. The set of angle inputs given to the three M CORDIC units is shown in table 6.1.

TABLE 6.1. SET OF ANGLE INPUTS FOR EACH CLOCK CYCLE GIVEN TO THE M CORDIC UNITS

Clock cycle	**Z2**	**Z0**	**Z1**
1	0	0	0
2	π/4	π/8	3π/8
3	π/2	π/4	3π/4
4	3π/4	3π/8	9π/8

6.3 ASIC and FPGA Implementation and Comparison

The proposed architecture is designed for FPGAs using Xilinx ISE. The family of FPGA chosen is Virtex-5 and the device chosen is XC5VSX240T-2FF1738. The proposed architecture is implemented for 16 inputs with a word length of 16 bits. All inputs and outputs are represented by signed 2’s complement number system. Table 6.2 shows the device utilization summary of the proposed architecture on the mapped FPGA.

TABLE 6.2. DEVICE UTILIZATION SUMMARY OF THE PROPOSED DESIGN ON XC5VSX240T-2FF1738

Device Utilization	**Proposed Architecture – I**
Number of occupied slices	1152
Number of slice LUTs	4225
Number of slice registers	2729
Number of bonded IOBs	258

Table 6.3 shows the comparison of proposed architecture in terms of FPGA hardware resources with the existing designs.

TABLE 6.3. COMPARISON OF PROPOSED ARCHITECTURE ON XC5VSX240T-2FF1738 FPGA

Design	**Area**		**Maximum Frequency (MHz)**	**Throughput (MS/s)**	**Slice-delay Product**
	Slices	**DSP48E***			
[27]	386	12	458	1831	-
Proposed Architecture	1152	0	86.79	347	13.273

* 1 DSP48E is equivalent to over 500 slices [http://zone.ni.com/reference/en-XX/help/371599F-01/lvfpga/dsp48e_func/]

The proposed architecture is designed using Synopsys DC and Cadence SoC Encounter for ASIC design flow. The process technology used is 180 nm. The design is checked for design rule check (DRC) as well as for timing. Table 6.4 shows ASIC implementation results of the proposed architectures in 180 nm technology. Figure 6.7 shows the physical layout of the proposed architecture in Cadence SoC Encounter.

TABLE 6.4. ASIC IMPLEMENTATION RESULTS OF THE PROPOSED ARCHITECTURE

ASIC implementation results using Synopsys DC	**Process technology: 0.18µm**
	Proposed Architecture – I
Total cell area	617619.875000
Total dynamic power	36.9739 mW
Add-sub width	16 bits
Slack at 50 MHz	10.38 ns

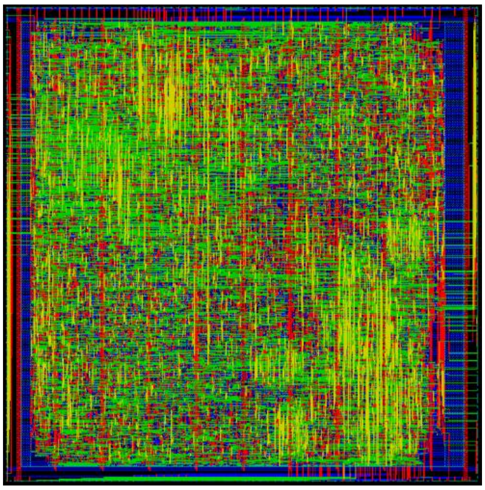


Figure 6.7. Physical layout of proposed architecture

Figure 6.8 shows the Isim simulation result of the proposed design

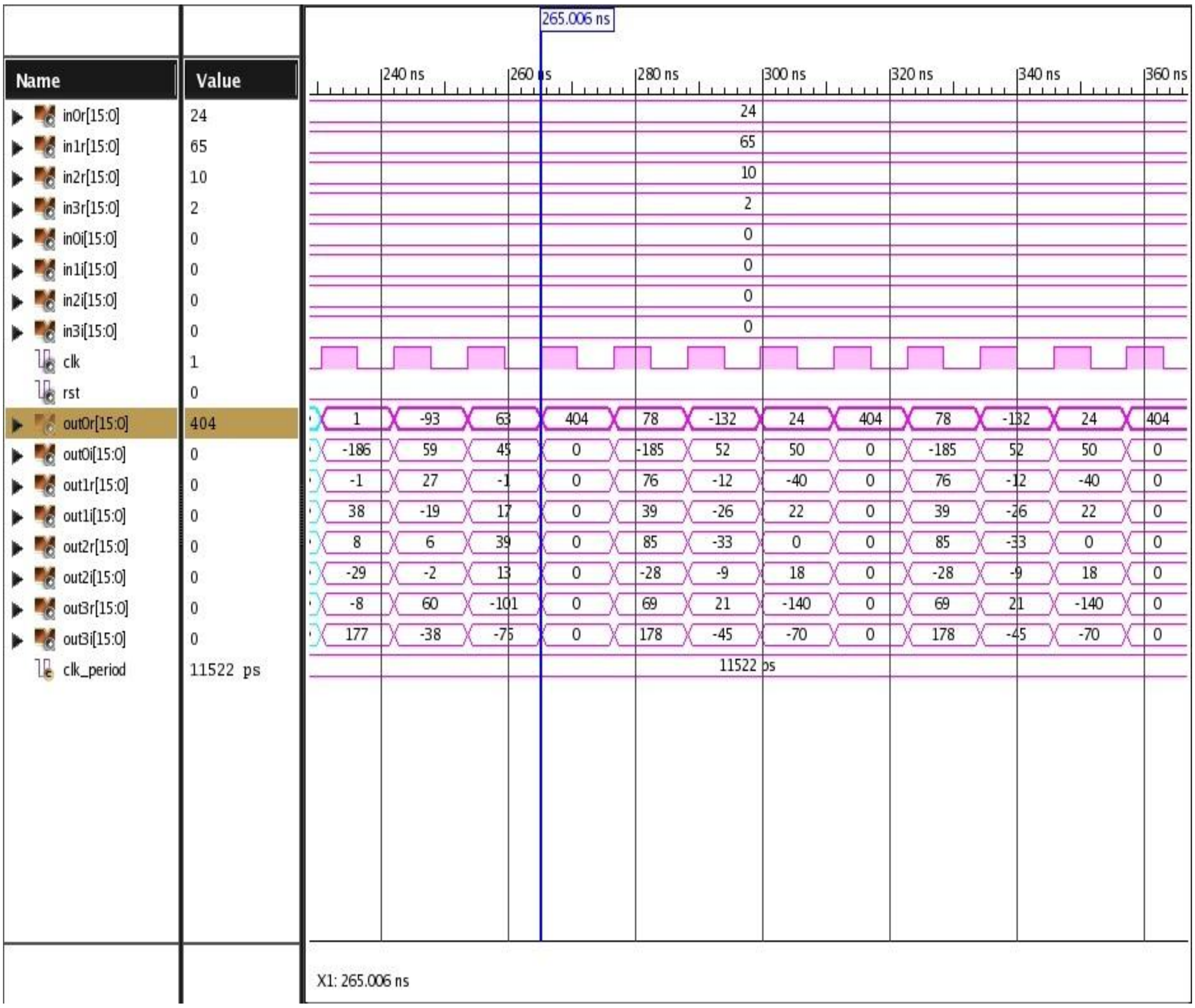


Figure 6.8. Isim simulation of proposed 16-point FFT core

6.4 Conclusions

This paper has reported multiplier-less architecture for radix-2^2 16-point 4-parallel complex FFT core. The algorithm that is considered for designing this is CORDIC. The architecture is designed using pipelining thus increasing the scope of their applicability. The device used for FPGA implementation is XC5VSX240T-2FF1738. The technology used for ASIC is 180 nm. The ASIC design is compliant with its FPGA counterparts thus showing better scope in many applications.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

We presented new scaling units for x and y data paths separately considering the practical congregate scaling constants. The proposed approximation is word length dependent and based on requirement of accuracy, the word length of the data paths can be varied. CORDIC unit with the proposed scaling units is implemented for different ranges of inputs and error analysis is done, considering the word length of data path as 16 bits. The maximum frequency, at which the design can be implemented, in Xilinx XC3S500E-4FG320 FPGA, fabricated in 90 nm process technology, is 75.593 MHz and its slice delay product is 106.465. We also have thus proposed a low hardware complex design to implement DFT using low latency scaled CORDIC. By exploring the symmetry properties of the transform, number of additions/subtractions has been minimised so as to achieve minimal power dissipation, minimal area and improved latency. The design that is proposed has been implemented in Xilinx FPGA, XC2VP30, which is fabricated using 0.13µm technology. The proposed design has seen very few exclusive multipliers (0 for 8-point DFT) and less number of adders compared to other traditional methods. This report also presents a design of DDS, which is also called a NCO. The proposed design is designed based on pipelined CORDIC unit used as phase to amplitude converter. The proposed design has a SFDR of – 96.31 dBc for 16 bit output amplitude resolution. In addition to that, the proposed design produces quadrature outputs which have better phase match thus making the design more useful for most of the communication systems. We also have reported a multiplier-less architecture of radix-2^2 16-point 4-parallel complex FFT core. The algorithm that is considered for designing this is CORDIC. The technology used for ASIC is 180 nm.

7.2 Future Work

Future work includes developing architectures that are related to the CORDIC implementation in linear and hyperbolic coordinate systems.

REFERENCES

[1] J. E. Volder, “The CORDIC trigonometric computing technique,” *IRE Trans. Electron. Comput.*, vol. EC-8, pp. 330–334, Sep. 1959.

[2] J. S. Walther, “A Unified Algorithm for Elementary Functions,” *Proc. Joint Spring Comput. Conf.*, vol. 38, pp. 379–385, Jul. 1971.

[3] P. K. Meher, J. Walls, T.-B. Juang, K. Sridharan, and K. Maharatna, “50 years of CORDIC: Algorithms, architectures and applications,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 9, pp. 1893–1907, Sep. 2009.

[4] B. Lakshmi and A. S. Dhar, “CORDIC Architectures: A Survey,” *VLSI Design*, vol. 2010, pp. 1–19, 2010.

[5] J. Villalba, J. A. Hidalgo, E. L. Zapata, E. Antelo, and J. D. Bruguera, “CORDIC architectures with parallel compensation of scale factor,” *Proc. Application Specific Array Processors Conf.*, pp. 258–269, Jul. 1995.

[6] Zhi-Xiu Lin and An-Yeu Wu, “Mixed-Scaling_Rotation CORDIC (MSR-CORDIC) Algorithm and Architecture for Scaling-Free High-Performance Rotational Operations,” *Proc. Acoustics, Speech, and Signal Processing Conf.*, vol. 2, pp. 653–656, Apr. 2003.

[7] K. Maharatna, S. Banerjee, E. Grass, M. Krstic, and A. Troya, “Modified Virtually Scaling-Free adaptive CORDIC Rotator Algorithm and Architecture,” *IEEE Trans. Circuits Syst. Video Tech.*, vol. 5, no. 11, pp. 1463–1474, Nov. 2005.

[8] S. Aggarwal, P. K. Meher, and K. Khare, “Area-Time Efficient Scaling-Free CORDIC Using Generalized Micro-Rotation Selection,” *IEEE Trans. VLSI Syst.*, vol. 20, no. 8, pp. 1542–1546, Aug. 2012.

[9] F. J. Jaime, M. A. Sanchez, J. Hormigo, J. Villalba, and E. L. Zapata, “Enhanced Scaling-Free CORDIC,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 7, pp. 1654–1662, Jul. 2010.

[10] M. G. Buddika Sumanasena, “A Scale Factor Correction Scheme for the CORDIC Algorithm,” *IEEE Trans. Comput.*, vol. 57, no. 8, pp. 1148–1152, Aug. 2008.

[11] Elisardo Antelo, Julio Villalba and Emilio L. Zapata, “Low-Latency Pipelined 2D and 3D CORDIC Processors,” *IEEE Trans. Computers*, vol. 57, no. 3, pp. 404-417, 2008.

[12] Pooja Choudhary and Dr. Abhijit Karmakar, “CORDIC Based Implementation of Fast Fourier Transform,” *Proc. ICCCT 2011*, pp. 550-555, 2011.

[13] Jayshankar, “Efficient Computation of the DFT of a 2N–Point Real Sequence using FFT with CORDIC based Butterflies,” *Proc. IEEE TENCON 2008*, 2008.

[14] A. L. Bramble, “Direct digital frequency synthesis,” *Proc. 35th Annu. Preq. Contr. Symp*., USERACOM (Ft. Monmouth, NJ), pp. 406–414, May 1981.

[15] J. Vankka, “Methods of Mapping from Phase to Sine Amplitude in Direct Digital Synthesis,” *IEEE Trans. Ultrasonics, Ferroelectrics, and Freq. Control*, vol. 44, no. 2, pp. 526–534, Mar. 1997.

[16] L. Cordesses, “Direct Digital Synthesis: A Tool for Periodic Wave Generation (Part 1),” *IEEE Signal Processing Magazine*, vol. 21, no. 4, pp. 50–54, Jul. 2004.

[17] A. McEwan and S. Collins, “Direct Digital-Frequency Synthesis by Analog Interpolation,” *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 53, no. 11, pp. 1294–1298, Nov. 2006.

[18] L. Cordesses, “Direct Digital Synthesis: A Tool for Periodic Wave Generation (Part 2),” *IEEE Signal Processing Magazine*, vol. 21, no. 5, pp. 110–112, Sept. 2004.

[19] A. Madisetti, A. Y. Kwentus, and A. N. Willson Jr, "A 100-MHz, 16-b, direct digital frequency synthesizer with a 100-dBc spurious-free dynamic range,” *IEEE J. Solid State Circuits*, vol.34, no.8, pp.1034-1043, Aug. 1999.

[20] S. Wang, V. Piuri, and E. E. Swartzlander, “Hybrid CORDIC Algorithms”, *IEEE Transactions on Comput.*, vol.46, no.11, pp.1202–1207, Nov. 1997.

[21] Tze-Yun Sung, Lyu-Ting Ko, and Hsi-Chin Hsin, “Low-Power and High-SFDR Direct Digital Frequency Synthesizer Based on Hybrid CORDIC Algorithm,” *Proc. International Symposium on Circuits and Systems*, pp. 249–252, May 2009.

[22] S.-N. Tang, J.-W. Tsai, and T.-Y. Chang, “A 2.4-GS/s FFT processor for OFDM-based WPAN applications,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 6, pp. 451–455, Jun. 2010.

[23] M. Garrido, K. K. Parhi, and J. Grajal, “A pipelined FFT architecture for real-valued signals,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 56, no. 12, pp. 2634–2643, Dec. 2009.

[24] S. Li, H. Xu, W. Fan, Y. Chen, and X. Zeng, “A 128/256-point pipeline FFT/IFFT processor for MIMO OFDM system IEEE 802.16e,” *Proc. IEEE Int. Symp. Circuits Syst*., 2010, pp. 1488–1491.

[25] Keshab K. Parhi, “*VLSI Digital Signal Processing Systems: Design and Implementation*”, Wiley, 1999.

[26] N Prasad, Ayas Kanta Swain, and K. K. Mahapatra, “Design and error analysis of a scale-free CORDIC unit with corrected scale factor,” *Proc. IEEE PrimeAsia*, pp. 7–12, Dec. 2012.

[27] Mario Garrido, J. Grajal, M. A. Sánchez, and Oscar Gustafsson, “Pipelined Radix-2^k Feedforward FFT Architectures,” *IEEE Trans. VLSI Syst.*, vol. 21, no. 1, pp. 23–32, Jan. 2013.

[28] Manohar Ayinala, Michael Brown, and Keshab K. Parhi, “Pipelined Parallel FFT Architectures via Folding Transformation,” *IEEE Trans. VLSI Syst.*, vol. 20, no. 6, pp. 1068–1080, Jun. 2012.

[29] Koushik Maharanta, A. S. Dhar et al., “CORDIC UNIT,” *United States Patent,* US 7,606,852 B2, Oct. 20, 2009.

[30] David B. Ribner, and Sunder Kidambi, “Direct Digital Synthesizers,” *United States Patent*, US 6,347,325 B1, Feb. 12, 2002.

PUBLICATIONS

RELATED TO THESIS:

[1] **N Prasad**, Ayas Kanta Swain, and K. K. Mahapatra, "Design and Error Analysis of a Scale Free CORDIC Unit with Corrected Scale Factor," *IEEE PrimeAsia Proc.*, pp. 7–12, Dec. 2012.

[2] **N Prasad**, Ayas Kanta Swain, and K. K. Mahapatra, "FPGA Implementation of Low Latency Scaled CORDIC Based Discrete Fourier Transform Core," *ICECIT Proc.*, vol. 2, pp. 67–72, Dec. 2012.

[3] **N Prasad**, Ayas Kanta Swain, and K. K. Mahapatra, "FPGA Implementation of Pipelined CORDIC Based Quadrature Direct Digital Synthesizer with Improved SFDR," *IEEE ICCPCT Proc.*, pp. 756–760, Mar. 2013.

OTHERS:

[4] Abhishek Mankar, **N Prasad,** Ansuman DiptiSankar Das, and Sukadev Meher, "Multiplier-less VLSI Architectures for Radix-2^2 Folded Pipelined Complex FFT Core," *Int. J. Circuit Theory Applicat.*, (COMMUNICATED).

[5] Ansuman DiptiSankar Das, Abhishek Mankar, **N Prasad**, K. K. Mahapatra, and Ayas Kanta Swain, "Efficient VLSI Architectures of Split-Radix FFT using New Distributed Arithmetic," *Int. J. Soft Computing and Eng.*, vol. 3, no. 1, pp. 264–271, Mar. 2013.

[6] Abhishek Mankar, **N Prasad**, and Ansuman DiptiSankar Das, "FPGA Implementation of Retimed Low Power and High Throughput DCT Core using NEDA," *2nd SCES Proc.*, Allahabad, India, Apr. 2013.

[7] Abhishek Mankar, Ansuman DiptiSankar Das, and **N Prasad**, "FPGA Implementation of 16-Point Radix-4 Complex FFT Core using NEDA," *2nd SCES Proc.*, Allahabad, India, Apr. 2013.

[8] Abhishek Mankar, **N Prasad**, and Sukadev Meher, "FPGA Implementation of Discrete Fourier Transform Core using NEDA," *CSNT Proc.*, Gwalior, India, Apr. 2013.