

Multiplierless Tunable Architecture for Signal Processing Transforms

A Thesis Submitted to
Nirma University
In Partial Fulfilment of the Requirements for
The Degree of
Doctor of Philosophy
in
Technology & Engineering

By
Pratik Pravinkumar Trivedi
(13EXTPHDE95)



Electronics and Communication Engineering Department

Institute of Technology, Nirma University

Ahmedabad, Gujarat, India

February 2020

Nirma University
Institute of Technology

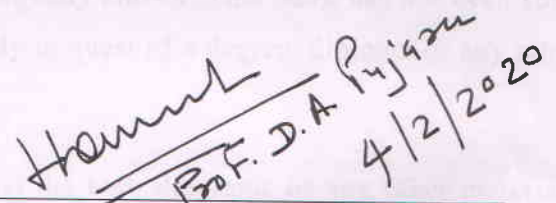
CERTIFICATE

This is to certify that the thesis entitled **Multiplierless Tunable Architecture for Signal Processing Transforms** has been prepared by **Mr. Pratik P Trivedi (13EXTPHDE95)** under my supervision and guidance. The thesis is his own original work completed after careful research and investigation. The work of the thesis is of the standard expected of a candidate for Ph.D. Programme in **Electronics and Communication Engineering** and I recommend that it be sent for evaluation.

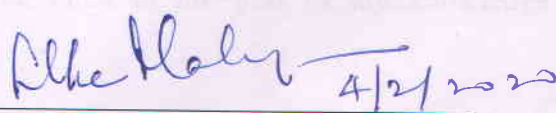
Date:


Dr. Tanish H. Zaveri,
Guide

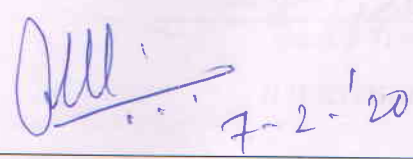
Forwarded through:


Prof. D.A. Pujari
4/2/2020

(i) Head, Electronics and Communication Engineering Department


Alka Kalyan
4/2/2020

(ii) Dean, Faculty of Technology and Engineering


Ill
7-2-'20

(iii) Dean, Faculty of Doctoral Studies and Research

To:

Executive Registrar
Nirma University

ABSTRACT

Signal processing algorithms like Discrete Fourier Transform, Discrete Cosine Transform, and Fast Fourier Transform find various applications in the field of Image processing, Wireless communication, Robotics, and many others. It covers basically three operations viz. Multiply, Shift and Accumulate. Hence if the input data goes on rising as in cases where high resolution is required the amount of multiply operations also rises significantly. For example, the number of complex multiplication operations in case of Discrete Fourier Transform is N^2 , where N is the number of points. Latency becomes an important issue which needs to be addressed in today's era as we, humans, thrive for the fastest systems with maximum resolution. Multiplierless techniques for this purpose has been always a research area as it helps in reduction of the later part. Multipliers bound to increase the latency especially in the algorithms which use complex multiplications, for instance to evaluate a single complex multiplications minimum four real multiplications are required. Hence, in techniques where number of such complex multiplications need to be evaluated, latency increases to an exponential amount as in case of Discrete Fourier transform. To reduce latency we need to either emphasize on reduction in amount of data to be processed or change the processing structure which can affect the overall time to output. There are three broad techniques found in the literature for addressing this issue. Complex Multiplication techniques itself requires four real multiplication and two adders and hence it becomes practically infeasible for the case where large amount of data needs to be transformed. Coordinate rotation of digital computer (CORDIC) (Volder) based techniques are well known for the Multiplierless implementation of the sinusoids. However it carries certain drawbacks viz. large number of iterations and accuracy. This thesis addresses the issues of Multiplierless implementation of the rotation for two different cases viz. CORDIC based techniques and Coefficient combined selection and Shift and Add implementation (CCSSI) (Garrido, Qureshi, and Gustafsson). It proposes improvement to the existing CORDIC based approach as well as CCSSI. Platform used for the implementation of the proposed approach is MATLAB. At the end the work presents a tunable multiplier less architecture for implementation of sinusoidal as well as non-sinusoidal transforms.

The thesis provides two different contributions in the field.

- 1) It proposes an efficient approach for the implementation of the Mixed Scaling and Rotation CORDIC (Lin and Wu) algorithm and also improves its SQNR by weighted amplifying factors.

- 2) The second contribution provides Coefficient combined & shift and add implementation (CCSSI) (Garrido, Qureshi, and Gustafsson) based approach to design Multiplierless rotators for various sinusoidal as well as non-sinusoidal transforms adding case of multiple constant rotators also. A novel tunable Combined co-efficient scaling and shift and add approach is proposed which takes into the following parameters.
 - Number of bits,
 - Number of adders,
 - Maximum allowable error
 - Number of points.

The approach improves the range of coefficients with respect to number of adders (the range taken is from 2 to 10 adders), and number of bits (the range taken is from 1 to 64 bits), compared to the existing approaches and is shown in the results in Table 4.16. It also presents the Multiplierless architecture for the tunable parameter shown above.

**Nirma University
Institute of Technology**

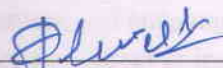
DECLARATION

I, **Pratik P Trivedi**, registered as Research Scholar, bearing Registration No. **13EXTPHDE95** for Doctoral Programme under the **Faculty of Technology and Engineering** of Nirma University do hereby declare that I have completed the course work, pre-synopsis seminar and my research work as prescribed under R. Ph.D. 3.5.

I do hereby declare that the thesis submitted is original and is the outcome of the independent investigations / research carried out by me and contains no plagiarism. The research is leading to the discovery of new facts / techniques / correlation of scientific facts already known. This work has not been submitted to any other University or Body in quest of a degree, diploma or any other kind of academic award.

I do hereby further declare that the text, diagrams or any other material taken from other sources (including but not limited to books, journals and web) have been acknowledged, referred and cited to the best of my knowledge and understanding.

Date: 4/2/20


Pratik P Trivedi

(13EXTPHDE95)

I endorse the above declaration made by the student.

Date: 4/2/20


Dr. Tanish N. Zaveri

Guide

ACKNOWLEDGEMENT

First and foremost, I would like to express my eternal gratitude to God and my parents, whose blessings have made me what I am today.

I am always thankful to my thesis supervisor, Late Dr. Tanish Zaveri , for his valuable suggestions, inspiring guidance and consistent support. I am thankful to Nirma University for providing an opportunity to carry out this research. I am thankful to Dr. P N Tekwani, Director Research and Innovation Cell, Nirma University for his valuable support. I am also thankful to Dr. Rajesh N Patel, Director, Institute of Technology, Nirma University for providing opportunity and support to carry out this research. I wish to express my sincere gratitude to the Nirma University for allowing me to join Ph.D. Programme and providing support during the course of this programme.

I am also very grateful to my research progress committee members, Dr. Nagendra P Gajjar, Professor, Institute of Technology, Nirma University, Ahmedabad and Dr. N.M.Patel, Professor, Birla Vishwakarma Mahavidyalaya (BVM) V.V. Nagar for their insightful suggestions and discussion. Their constructive criticism and comments help me to nurture and strengthen my research work.

I am also grateful to Dr.Sanjay Chaudhary, Dean, School of engineering and Applied Science, Ahmedabad University, Dr. Mehul Raval, Professor, Pandit Deendayal Petroleum University, Gandhinagar who actively encouraged me at each and every stage of my work. I also want to thank all my colleagues at School of engineering and Applied Science, Ahmedabad University for their immense support and guidance.

I express my sincere thanks to Dr. Dhaval Shah, Dr. Akash Mecwan, Dr. Ruchi Gajjar, Dr. Vijay Savani, Dr. Piyush Bhatasana, for their fruitful suggestions and encouragement. I would also like to thank all friends, colleagues and well-wishers for their direct or indirect support in the successful completion of this work.

Special thanks to my friends and my daughter for their motivational support and unconditional love. The time devoted to this thesis is from their accounts.

Contents

| | |
|--|----|
| Chapter 1 Introduction..... | 12 |
| 1.1 Motivation..... | 13 |
| 1.2 Objectives..... | 14 |
| 1.3 Contribution of the thesis..... | 15 |
| 1.4 Organization of the thesis..... | 16 |
| Chapter 2 Literature Review and Background Theory..... | 18 |
| 2.1 Rotators used in hardware..... | 18 |
| 2.2 Standard CORDIC..... | 19 |
| 2.3 Angle Recoding CORDIC..... | 21 |
| 2.4 Unified CORDIC..... | 21 |
| 2.5 Reconfigurable CORDIC..... | 23 |
| 2.6 Enhanced Scaling Free CORDIC..... | 23 |
| 2.7 EEAS CORDIC..... | 25 |
| 2.8 Mixed Scaling and Rotation CORDIC (MSR CORDIC)..... | 26 |
| 2.9 CORDIC II: A new Improved Algorithm..... | 27 |
| 2.10 Multiplication by two integers using minimum number of adders..... | 28 |
| 2.10.1 Multiplierless Constant Multiplication Algorithms..... | 29 |
| 2.11 Coefficient Combined Selections and Shift and Add Implementation..... | 30 |
| Chapter 3 Proposed Approach based on Mixed Scaling Rotation CORDIC..... | 35 |
| 3.1 MSR Algorithm..... | 36 |
| 3.2 The Proposed Enhanced MSR Scheme..... | 38 |
| 3.3 The Proposed Algorithm..... | 41 |
| 3.4 Results and Discussions..... | 43 |

| | |
|--|----|
| Chapter 4 Coefficient Combined Selections and Shift and Add Implementation based Approach..... | 48 |
| 4.1 Proposed Approach based on CCSSI | 48 |
| 4.2 Coefficients Selection for Non-Sinusoidal Transform..... | 58 |
| 4.3 Coefficient Selection based on the number of adders..... | 60 |
| 4.4 Realization of the proposed Architecture..... | 68 |
| 4.5 Results and Discussions | 70 |
| Chapter 5 Conclusions and Future Scope..... | 77 |
| 5.1 Conclusions..... | 77 |
| 5.2 FutureScope..... | 78 |
| Publication | 79 |
| References..... | 80 |

List of Tables

| | |
|--|----|
| Table 4.1 MCR Remaining Kernel for $b = 5$ and $\text{max.error} = 0.05$ for two angles..... | 51 |
| Table 4.2 MCR Remaining Kernel for $b = 6$ and $\text{max.error} = 0.05$ for two angles..... | 52 |
| Table 4.3 MCR Remaining Kernel for $b = 7$ and $\text{max.error} = 0.05$ for two angles..... | 52 |
| Table 4.4 MCR Remaining Kernel for $b = 8$ and $\text{max.error} = 0.05$ for two angles..... | 53 |
| Table 4.5 MCR Remaining Kernel for $b = 5$ and $\text{max.error} = 0.05$ for five angles..... | 54 |
| Table 4.6 MCR Remaining Kernel for $b = 6$ and $\text{max.error} = 0.05$ for five angles..... | 55 |
| Table 4.7 MCR Remaining Kernel for $b = 7$ and $\text{max.error} = 0.05$ for five different angles. | 56 |
| Table 4.8 MCR Remaining Kernel for $b = 8$ and $\text{max.error} = 0.05$ for five angles..... | 57 |
| Table 4.9 MCR Remaining Kernel for $N = 8$ | 59 |
| Table 4.10 MCR Remaining Kernel for $N = 16$ | 60 |
| Table 4.11 Remaining Kernel for 4 Adders..... | 62 |
| Table 4.12 Remaining Kernel for 6 Adders..... | 63 |
| Table 4.13 Remaining Kernel for 8 Adders..... | 64 |
| Table 4.14 Remaining Kernel for 10 Adders..... | 65 |
| Table 4.15 MCR Remaining Kernel based on the proposed approach..... | 66 |
| Table 4.16 Comparison of Multiplierless Rotator of various approaches with the Proposed approach..... | 67 |
| Table 4.17 Number of unique coefficient obtained..... | 75 |

List of Figures

| | |
|---|----|
| Figure 1.1 Block diagram of the MSR based system..... | 15 |
| Figure 1.2 Block diagram of the CCSSI based proposed framework..... | 16 |
| Figure 2.1 Block Diagram of the CCSSI based Approach..... | 31 |
| Figure 2.2 (a) Initial design space with the required angles | 33 |
| Figure 2.2 (b) Reduced coefficient based on the delta angle | 33 |
| Figure 2.2 (c) Further reduction of coefficient based on the fixed scaling factor | 33 |
| Figure 3.1 SQNR Comparison between MSR_CORDIC and the proposed scheme..... | 43 |
| Figure 3.2 (a) Comparing the relationship between SQNR performances and scaling factor value of v_{upper} in generalized scheme of MSR-CORDIC and proposed MSR-CORDIC $N_{SPT} = 3$ and $N = 3$ | 45 |
| Figure 3.2 (b) $N_{SPT} = 4$ and $N = 2$ | 46 |
| Figure 3.3 Analysis of MSR-CORDIC and proposed scheme for different combinations of N_{SPT} | 47 |
| Figure 4.1 Block Diagram of the Proposed Approach..... | 49 |
| Figure 4.2 (a) Initial design space with the required angles | 50 |
| Figure 4.2 (b) Reduced coefficient based on the delta angle | 50 |
| Figure 4.2 (c) Further reduction of coefficient based on the fixed scaling factor | 50 |
| Figure 4.3 Realization of rotator for proposed framweork..... | 68 |
| Figure 4.4 Realization of combined rotator design for kernel-1 from Table-4.15. | 69 |
| Figure 4.5 Trade-off between minimum Rotation Error vs. number of bits b..... | 71 |
| Figure 4.6 Trade-off between minimum Rotation Error vs. number of bits b..... | 72 |
| Figure 4.7 Trade-off between minimum Rotation Error vs. number of bits b..... | 73 |
| Figure 4.8 Trade-off between minimum Rotation Error vs. number of bits b..... | 74 |

List of Abbreviations

| | |
|--------|---|
| AR | Angle Recoding |
| CORDIC | Coordinate Rotation Digital Computer |
| CCSSI | Coefficient Combined Selection and Shift and Add Implementation |
| CSD | Canonic Signed Digit |
| DCT | Discrete Cosine Transform |
| DFT | Discrete Fourier Transform |
| DSP | Digital Signal Processing |
| EEAS | Extended Elementary Angle Set |
| FFT | Fast Fourier Transform |
| FIR | Finite Impulse Response |
| FPGA | Field Programmable Gate Array |
| IIR | Infinite Impulse Response |
| MCM | Multiple Constant Multiplication |
| MCR | Multiple Constant Rotator |
| MSR | Mixed Scaling Rotation |
| MVR | Moving Vector Rotation |
| SCM | Single Constant Multiplication |
| SCR | Single Constant Rotator |

Chapter 1

Introduction

Signal processing algorithms like Discrete Fourier Transform, Discrete Cosine Transform, and Fast Fourier Transform find various applications in the field of Image processing, Wireless communication, Robotics, and many others. It covers basically three operations viz. Multiply, Shift and Accumulate (Rabiner and Gold). Hence, if the input data goes on rising as in cases where high resolution is required, the amount of multiply operations also rises significantly. For example, the number of complex multiply operations in case of Discrete Fourier Transform is N^2 , where N is the number of points. Latency becomes an important issue which needs to be addressed in today's era as the target is to design the fastest systems with maximum resolution. To reduce latency, emphasis needs to be given on either reduction in amount of data to be processed or changing the processing structure which can affect the overall time to output. Multiplierless techniques for this purpose has been always a research area as it helps in reduction of the later part. There are three broad techniques found in the literature for addressing this issue. Complex Multiplication techniques itself requires four real multiplication and two adders and hence it becomes practically infeasible for the case where large amount of data needs to be transformed. Coordinate rotation of digital computer (CORDIC) based techniques are well known for the Multiplierless implementation of the sinusoids. However, it carries certain drawbacks viz. large number of iterations and accuracy. This thesis addresses the issues of Multiplierless implementation of the rotation for two different cases viz. CORDIC (Volder; Meher et al.; Aggarwal et al.) based techniques and Coefficient combined selection and Shift and Add implementation (CCSSI) (Garrido, Qureshi, and Gustafsson). It proposes improvement to the

existing CORDIC based approach as well as CCSSI. Platform used for the implementation of the proposed approach is MATLAB. In the end, this thesis work presents a tunable multiplier-less architecture for implementation of sinusoidal as well as non-sinusoidal transforms.

1.1. Motivation

Digital signal processing, scientific computing, and other communication applications, signal transforms are a major part of signal analysis. Transform algorithms such as the Discrete Fourier Transform (Meher and Park), Discrete Cosine Transform (Meher and Park) and many more used in many of the digital signal processing applications. These algorithms are designed in a highly structured form and exhibits a large amount of parallelism. Thus, these algorithms are well suited for the hardware implementation as a sequential data-path on a field-programmable gate array (FPGA) (Li et al.; Tang et al.; Möller et al.; Andraka; Andraka) and DSP processors (He and Torkelson, “Design and Implementation of a 1024-Point Pipeline FFT Processor”; Wanhammar). These algorithms require many arithmetic steps to perform such as addition, subtraction and multiplication. There are many processors which handle the complex and huge multiplications which are necessary for the signal transform. Performing multiplication on hardware is much computationally costly as well as it also requires complex hardware which eventually requires a huge space for the development of the hardware.

In today's time with the increase in technological advancement, the requirement for the smaller, portable, cost-effective and efficient performance of any system is necessary. For adaptive signal processing systems, these are the important factors and more importantly the computational cost on the hardware point. Thus such computationally costly and power-consuming operation on the hardware is multiplication. It requires real-time hardware multipliers for adaptive signal processing which consume too much power and require memory which is scare system resources. Portable devices such as mobile phones and other communication devices require such multipliers for the signal analysis at the software level but on contrary the power consumption increases which leads to shorter battery life. Thus, to reduce the power consumption, design of multipliers can be obtained by designing algorithms for signal processing in such a way that requires fewer hardware multipliers.

Signal processing algorithms for the transforms such as Discrete Fourier transform or Fast Fourier transform use complex numbers (Twiddle Factor) (Andersson) which are multiplied with signal and their corresponding frequency analysis is obtained. Here, these complex numbers used in the algorithms are fixed which can be obtained by rotation of a fixed angle on the complex plane. Implementation of getting such a complex number on hardware is done using different Rotators. This rotator takes an angle as an input to it and performs rotation based on the input angle. There are various algorithms such as Coordinate rotation digital computer (CORDIC) (Volder) which uses these rotators. Thus, there arises a need to select a complex number that increases the efficiency of the use of rotators and improves the performance.

The approach used for designing such an architecture is by combining the selection of the coefficient for the transforms and using multiplier-less multiplication algorithms such as Canonical signed digit (CSD) (Voronenko and Püschel).

Single Constant Multiplication (SCM) and Multiple Constant Multiplications (MCM) (Voronenko and Püschel; Möller et al.; Aksoy et al.; Dempster and Macleod, “Constant Integer Multiplication Using Minimum Adders”; Gustafsson). These algorithms use adders and shifter to perform multiplication. These algorithms are limited to constant multiplication but since most of the signal processing algorithms require multiplication with a constant number these multiplierless multiplication algorithms can be implemented on low-cost hardware devices such as field-programmable gate array (FPGA).

1.2 Objective

The objective of this thesis is to design a tunable multiplier-less architecture for determining the coefficients on the basis of tunable parameters like the signal space, the number of adders required, maximum allowable error and angles. It reduces the hardware architecture used in the multiplierless constant rotators for the signal processing transform. It also provides the optimized number of coefficients which will be used by the sinusoidal transforms like Discrete Fourier transform, Discrete Cosine transform, Walsh-Hadamard transform (Ahmed and Rao, “Walsh-Hadamard Transform”) as well as non-sinusoidal transforms like the Binary HAAR transform.

Based on study of the existing techniques for designing the multiplier-less architecture for signal processing transforms available in the literature, following research objectives are addressed in the thesis:

- 1) It proposes an efficient approach for the implementation of the Mixed Scaling and Rotation CORDIC algorithm and also improves its SQNR by weighted amplifying factors.
- 2) The second contribution provides Coefficient combined and shift and add implementation (CCSSI) based approach for the design of multiplier-less rotators for various transforms for multiple constant rotators as well.
- 3) Designing a unique tunable multiplier-less architecture for the design of sinusoidal as well as non-sinusoidal transforms.

1.3 Contribution of the thesis

This thesis addresses two major contribution in the architecture of multiplier-less algorithms. First, it provides a simple and efficient method to increase the SQNR for the Mixed Scaling Rotation CORDIC (Lin and Wu) approach. It then presents a combined architecture for the MSR CORDIC algorithm for designing the sinusoidal transforms. Figure 1.1 presents the block diagram of the overall framework for the first contribution.

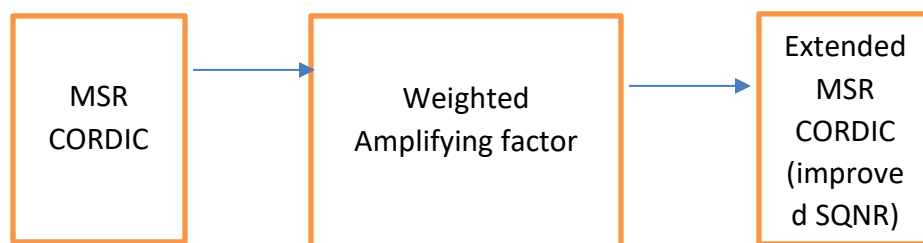


Figure 1.1 Block diagram of the MSR based system

The second technique presents designing of constant rotators that combines the coefficient selection and the shift and adds implementation in the design process. It takes into the consideration the tunable parameters viz. Signal space, maximum error that can be allowed and number of adders and based on these inputs it provides number of possible optimum coefficients

that satisfies the conditions. It also presents a multiplierless tunable architecture for the optimum coefficients generated which immensely reduce the latency which a multiplication based algorithms might have required. Emphasize is more on generation of optimum coefficients that is majorly evaluated by Canonic signed digit algorithm (CSD). Results shows that the framework works better than the existing algorithms present in the literature. The overall block diagram for the system is as shown in Figure 1.2 below.

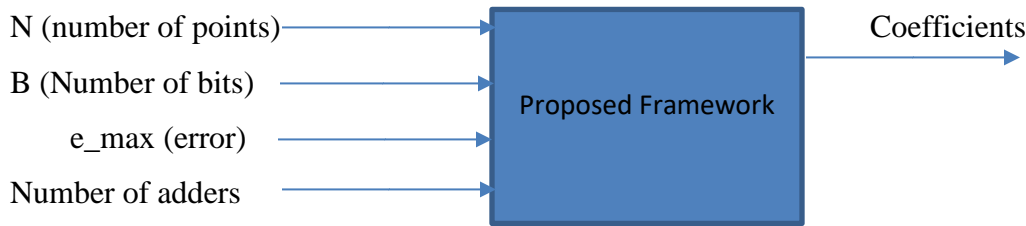


Figure 1.2 Block diagram of the CCSSI based proposed framework

1.4 Organization of the thesis

The thesis has five chapters followed by the references. The first chapter is the Introduction. Chapter 2 gives the literature review and background theory. Chapter 3 describes the proposed algorithm for first contributions. The Second Proposed Algorithm and Tunable Architecture is discussed in Chapter 4. At the end conclusion of the work with the future scope is discussed in Chapter 5 which is then followed by References.

Chapter 2 (Literature Survey): In this chapter study of various multiplication algorithms are discussed. Also, the bottleneck and research issues based on the study of different algorithms are highlighted.

Chapter 3 (Proposed Multiplierless Approach based on MSR CORDIC): This chapter presents proposed techniques for addressing the research issues of Mixed Scaling and Rotation CORDIC algorithm.

Chapter 4 (CCSSI based proposed multiplierless architecture): This chapter extends the existing CCSSI algorithm with for different angles and defines a tunable architecture for Coefficient combined scaling and shift and add implementation approach.

The first technique proposes a weighted amplifying factor based approach for improving the SQNR keeping the hardware same for the Mixed Scaling and Rotation CORDIC algorithm.

The Second technique addresses the issue of optimizing the number of coefficient in the CCSSI algorithm based on the tunable parameters, the considered parameters are signal space, maximum allowable error and number of adders. Also an architecture for the tunable multiplierless algorithm for sinusoidal as well as non-sinusoidal transform is addressed. Given number of points, number of adders, maximum error, and signal space it generates the coefficients for the signal processing transform. The results of the overall framework are presented and discussed.

Chapter 5 (Conclusion and Future scope): This chapter summarizes and presents the conclusions of the overall thesis. The scope of the future work in the domain is discussed.

Chapter 2

Literature Review and Background Theory

In the chapter below background on Rotators and different multiplierless multiplication algorithm is provided. Later in the chapter, study of various existing techniques for multiplierless architectures are discussed followed by the research issues of the existing techniques.

2.1. Rotators used in Hardware

A Rotation operation in the signal processing algorithms is a multiplication by a complex number whose magnitude is equal to one. Discrete Fourier transform (DFT) or Fast Fourier transform (FFT) (Ahmed and Rao, “Fast Fourier Transform”), Discrete sine or cosine transform (DCT) and also many filters such as IIR filters and FIR filters uses rotation operation in their algorithms (Garrido and Grajal).

A rotation of a complex number in a complex plane is carried out by multiplying a complex number with $e^{j\alpha}$. The new obtained complex number is at the angle α . Mathematically it can be written as

$$\begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (2.1)$$

where x and y are the current complex point $(x + jy)$ and X and Y are the point $(X + jY)$ obtained after rotation by angle α .

The values obtained from the above equation are scaled by a factor of K which remains fixed once the number of iterations are decided. But, the scaling factor inherent to this algorithm is an important drawback in this method.

For implementing this rotation on hardware either the CORDIC algorithm or complex multipliers are popular approaches found in the literature.

2.2. Standard CORDIC (Volder)

CORDIC stands for the Coordinate Rotation in Digital Computer. In the digital system, the CORDIC algorithm uses shift and add implementation for the micro rotation which is more simple and easy to implement on hardware whereas ,complex multipliers (Despain) uses real multipliers with the adders which are much complex to implement on the hardware.

To implement such algorithms a rotator is required in the hardware which performs rotation based on the given angle. There are two main types of rotators:

1. General Rotators
2. Constant Rotators

In the case of general rotators, it can carry out rotation based on any angle which is given as an input to the rotators. This type of rotator is usually implemented by the complex multipliers which consist of four real multipliers and two adders. The coefficient obtained by the rotation is simply multiplied with a signal as an input to these complex multipliers. On the other hand, in case of constant rotators, they are designed to rotate based on a fixed angle. The rotations are fixed by specific angles and these types of rotators are used by the CORDIC algorithms. Algorithms such as CORDIC use micro rotations by breaking angles that are easy to implement using constant rotators and multiplication is carried out using shifters and adders in the hardware.

This process tries to look for an optimal way from the hardware point of view, a way to ensure that the computation is as fast as possible with accepted accuracy so to make the architecture of the hardware as simple as possible. CORDIC is a simple shift-add iterative procedure to perform several computing tasks. It works in two modes: the rotation-mode and the vectoring-mode. In the Rotation-mode, CORDIC determines the components of a vector because of the rotation of a given vector by a certain angle. It is widely used for complex multiplications and graphic applications. Using vectoring-mode CORDIC, the magnitude as well as the phase angle of a planar vector are estimated from its component values.

The following equations demonstrate rotation of a vector with the help of CORDIC algorithm.

$$\begin{bmatrix} X_{i+1} \\ Y_{i+1} \end{bmatrix} = K_i \begin{bmatrix} 1 & -\tan\alpha_i \\ \tan\alpha_i & 1 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad (2.2)$$

$$\text{where, } \tan\alpha_i = 2^{-i}$$

Hence it brings,

$$X_{i+1} = k_i (x_i - d_i y_i 2^{-i}) \quad (2.3)$$

$$Y_{i+1} = k_i (y_i + d_i x_i 2^{-i}) \quad (2.4)$$

Where, d_i takes the values (+1, -1).

CORDIC suffers the drawback of large number of iterations (i), for example., to get the value of angle of 45 degree it takes on the fixed i number of iterations, whereas for the value $i=0$, the value of 45 degree can be reached.

2.3 Angle Recoding CORDIC (Hu and Naganathan)

The disadvantage for CORDIC algorithm is addressed in the second version which is named as Angle recoding CORDIC. In the equations of the CORDIC given below it tries d_i to take on the value of (+1,0,-1) which is nothing but allowing the CORDIC algorithm to take on the value of zero and then finding the minimum number of steps to reach a particular angle with the help of Greedy algorithm.

$$X_{i+1} = k_i (x_i - d_i y_i 2^{-i}) \quad (2.5)$$

$$Y_{i+1} = k_i (y_i + d_i x_i 2^{-i}) \quad (2.6)$$

Where, d_i takes the values (+1,0,-1).

2.4 Unified CORDIC (Walther)

CORDIC algorithms then can be computed with help of linear, circular, and hyperbolic trajectories. Depending upon their equation, trajectories of the vectors will be defined and generated by the successive iterations. Out of these three categories, circular and hyperbolic are more prevalent and widely used. Circular CORDIC is mainly used for the computation of sine/cosine functions, waveform generation, implementation of digital filters, transform computation, matrix calculations etc., whereas, Hyperbolic CORDIC is used for the computation of exponents and sinh/cosh functions, neural networks. CORDIC has a wide range of applications and is also used to simplify other basic and important algorithms like Eigen-value estimations, QR decomposition, phase and frequency estimations, singular value decomposition, synchronization in digital receivers, graphics processing and robot manipulation, both rotation and vectoring-modes. The hardware implementation of these applications requires more than one CORDIC processor operating in different modes and trajectories (Aggarwal et al.).

The following equations shows the Unified CORDIC with multiple trajectories. The parameter m determines the type of trajectory.

$$X_{i+1} = x_i - m d_i y_i 2^{-i} \quad (2.7)$$

$$Y_{i+1} = y_i + d_i x_i 2^{-i} \quad (2.8)$$

$m = 0, 1, -1$ for linear, circular and hyperbolic respectively.

As the equations of hyperbolic, circular, and linear CORDIC are different, therefore each will have a different architecture. The algorithm works on bringing all the different trajectories into a single combined architecture, thus it presents reconfigurable CORDIC (Co-ordinate Rotation Digital Computer) architectures which can be configured to operate either for circular or hyperbolic trajectories in rotation as well as vectoring-modes by manipulating equations in such a way that changing one value may change to either of the remaining two algorithms. It propose three reconfigurable CORDIC designs: a reconfigurable rotation mode CORDIC that operates either for circular or hyperbolic trajectory, a reconfigurable vectoring-mode CORDIC for circular and hyperbolic trajectories, and a generalized reconfigurable CORDIC that can operate in any of the modes for both circular as well as hyperbolic trajectories. The reconfigurable CORDIC can perform the computation of various trigonometric and exponential functions, logarithms (Hoang et al.; De Caro, Genovese, et al.; Sai and Hoang; Paul et al.; Johansson et al.; De Caro, Petra, et al.; Pineiro et al.; Juang et al.; Goel et al.), square-root, etc. of circular and hyperbolic CORDIC using either rotation-mode or vectoring mode of operation in one single circuit. The various applications include digital synchronizers, graphics processors, scientific calculators and many other applications. Because all the three architectures are implemented in one architecture and in a combined way, it saves a lot of area as well. In Unified CORDIC however, it is difficult to increase the region of convergence (Hu et al.) for all the three trajectories. For instance, the ROC for circular is in the range $[-99, 99]$ and for the hyperbolic trajectory the same comes down to 60.4 degrees. Also the scaling factor are different for the trajectories.

2.5 Reconfigurable CORDIC (Aggarwal et al.)

Unified CORDIC discussed in the sections above takes care of three different trajectories viz. Circular, Linear and Hyperbolic CORDIC operations by adding a parameter to the basic CORDIC equations. However, the major drawback it suffers from is in the region of convergence (ROC). The hyperbolic function cannot be defined of the entire range of ROC. Since the hyperbolic functions lack symmetry, the second order approximations of Taylor series can only get the ROC of 22.5° . The drawback for the ROC in Unified CORDIC is in changing the scaling operations using the second order Taylor series approximations, the algorithm imposes restrictions on the basic shift $i=4$ which limits the region of convergence to 7.16° however this can be increased to 22.5° by increasing the number of iterations.

Reconfigurable CORDIC proposes the CORDIC with 1) Reconfigurable Rotational mode with trajectories viz. Circular and Hyperbolic and extends the ROC for the entire range of computations. 2) Reconfigurable Vectoring mode with trajectories viz. Circular and Hyperbolic functions with extended ROC. It takes the third order Taylor series approximations for deriving the matrices for the CORDIC operation. Same set of elementary angles are used to derive both Circular and Hyperbolic operations. The pipelined structure (He and Torkelson, "Design and Implementation of a 1024-Point Pipeline FFT Processor"; Gorman and Wills; He and Torkelson, "Designing Pipeline FFT Processor for OFDM (de) Modulation"; Oh and Lim) proposed helps decrease the latency inculcated by the algorithm. However, the algorithm do not generate the accuracy levels without using multipliers. Also, the latency is increased for recursive design proposed.

2.6 Enhanced Scaling free CORDIC (Jaime et al.)

Enhanced Scaling Free CORDIC has been implemented successfully in wireless applications. This new enhancement and some further improvements in this method have obtained some architectures which are able to reach 35% lower latency and 36% reduction in area and power consumption compared to the original architecture. This scaling free CORDIC is mainly intended

for rotation mode but it can also be used for vectoring mode. It makes micro rotation following the same direction. In this algorithm, the approximations are made in the following form:

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots \quad (2.9)$$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots \quad (2.10)$$

This method also overcomes the calculation of the z factor that was performed in the standard CORDIC. A scaling free CORDIC iteration can be expressed as follows:

This brings down the enhanced CORDIC equations as

$$X_{i+1} = x_i \left(1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots \right) - y_i \left(x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots \right) \quad (2.11)$$

$$Y_{i+1} = y_i \left(1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots \right) + x_i \left(x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \dots \right) \quad (2.12)$$

Scaling free CORDIC method can be implemented on hardware by just adders and shifters. But the only problem with this system is its extremely small range of convergence.

2.7 EEAS CORDIC (Meher et al.)

In digital system, numbers need to be defined with a finite number of bits, which leads to quantization error. $\cos \alpha$ and $\sin \alpha$ in equation (2.1) can be written as C and S which are integers and can be implemented in a digital system. The range of this C and S which uses b bits is given as $[-2^{b-1}, 2^{b-1} - 1]$. This transforms the equation in a digital system as,

$$\begin{bmatrix} X_D \\ Y_D \end{bmatrix} = \begin{bmatrix} C & -S \\ S & C \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (2.13)$$

where, X_D and Y_D represents the point in the digital system, and C and S are obtained by a different method which is discussed further in this section.

As mentioned in the above section on Rotators used in Hardware, there are two types of rotators one is a general rotator and the second type of rotator is a constant rotator. CORDIC algorithm uses general rotations and also breaks down rotation angle into a series of k-micro rotations which can be represented as $\alpha_k = \pm \tan^{-1} 2^{-k}$. These angles use only two adders and calculate the rotation as

$$\begin{bmatrix} X_D \\ Y_D \end{bmatrix} = \begin{bmatrix} 2^k & -\delta_k \\ \delta_k & 2^k \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (2.14)$$

where, C is denoted as 2^k and S as δ_k . Here δ_k belongs to set $\{-1, 1\}$ which is required to determine the direction of rotation. There also exists a notion of scaling factor as for C and S which is denoted by R and this algorithm calculates scaling for each angle k as $R(k) = \sqrt{2^{2k} + 1}$.

The extended elementary angle set (EEAS) CORDIC algorithm (Meher et al.) is based on constant rotators. Constant rotators work on a fixed angle only due to which, this algorithm considers the elementary angles only, given by $\alpha_k = \tan^{-1} (\delta_k 2^{-a_k} + \gamma_k 2^{-b_k})$. The values δ_k & γ_k belongs to set $\{-1, 0, 1\}$ and a_k & b_k belongs to N. The value of $b_k > a_k$ and a value $c_k = b_k - a_k$, the rotation which uses four adders can be determined as

$$\begin{bmatrix} X_D \\ Y_D \end{bmatrix} = \begin{bmatrix} 2^{b_k} & -(\delta_k 2^{c_k} + \gamma_k) \\ (\delta_k 2^{c_k} + \gamma_k) & 2^{b_k} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (2.15)$$

2.8 Mixed Scaling and Rotation CORDIC (Lin and Wu)

The algorithm uses $2 * (I_k + J_k + 1)$ adders per micro rotation and each micro rotation is calculated as

$$\begin{bmatrix} X_D \\ Y_D \end{bmatrix} = \begin{bmatrix} \sum_{i=0}^{I_k-1} \delta_{ki} 2^{a_{ki}} & -\sum_{j=0}^{J_k-1} \gamma_{kj} 2^{b_{kj}} \\ \sum_{j=0}^{J_k-1} \gamma_{kj} 2^{b_{kj}} & \sum_{i=0}^{I_k-1} \delta_{ki} 2^{a_{ki}} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (2.16)$$

The major difference between the conventional CORDIC algorithm and EEAS CORDIC & MSR-CORDIC is that the algorithm provide a solution to compensate scaling which depends on the rotation angle.

Other approaches to reduce the number of micro rotation (Meher et al.) suggest selecting the subset of micro rotation in CORDIC and approximating the rotation angle to reduce rotation error.

Another approach for designing rotators is based on optimizing the real constant multiplication which quantize $\cos \alpha$ and $\sin \alpha$ by b number of bits, the rotation is given as,

$$\begin{bmatrix} X_D \\ Y_D \end{bmatrix} = \begin{bmatrix} \lfloor 2^b \cos \alpha \rfloor & -\lfloor 2^b \sin \alpha \rfloor \\ \lfloor 2^b \sin \alpha \rfloor & \lfloor 2^b \cos \alpha \rfloor \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (2.17)$$

where, $[D]$ represents a rounding operation. Here, the multiplication of $[2^b \cos \alpha]$ and $[2^b \sin \alpha]$ with x and y is implemented by using shift and add operations.

One of the simple approaches is to use a canonic signed digit (CSD) (Garrido, Grajal, and Gustafsson) transform which was described in the above section on multiplierless multiplication. This approach provides less number of adders compared to convention binary multiplication. There also many other approaches that can further reduce the number of adders in multiplication which are known as single constant multiplication (SCM) and Multiple constant multiplication (MCM) (Voronenko and Püschel) techniques.

There exist many other approaches which are based on trigonometric identities (Philipov et al.; Arguello et al.; James; Garrido, Qureshi, Takala, et al.). These approaches and algorithms restrict the set of coefficients used for the rotation.

2.9 CORDIC II: A New improved Algorithm (Garrido, Källström, Kumm, et al.)

Unlike the prior CORDIC algorithms, The CORDIC II algorithm uses new angle sets to reach faster convergence. The basic CORDIC uses micro-rotations for the implementations of rotators which increases the latency of the algorithm. CORDIC II algorithm uses a combinations of Friend angles, Uniform scaled rotations (USR) and Nano rotations to reach to a particular angle. Friend angles are those angles a_i for which there exists a set of coefficient $P_i = C_i + j S_i$, whose angle is a_i , such that there magnitudes are same. USR uses the same scaling as the redundant CORDIC.

CORDIC II uses six rotation stages in pipeline that uses the one of the three angle sets.

Stage 1: This stage takes care of trivial angles $\pm 180^0$, $\pm 90^0$ and set the left over angles in the range from $\pm 45^0$. The Architecture of this takes two negators and four 2:1 multiplexers.

Stage 2: It uses friend angles and defines their architecture. It consists of five adders and seven 2:1 multiplexers.

Stage 3: This uses USR CORDIC. Its architecture consists of two adders and two 2:1 multiplexers.

Stage 4 & 5: These two stages use conventional CORDIC algorithm for rotations by angles 1.790° and 0.895° .

Stage 6: The sixth stage uses Nano rotations. It uses two adders, two shifters and scaling.

By using the proposed stages, the CORDIC II algorithm requires minimum number of adders than the conventional CORDIC used so far. However, CORDIC II algorithm defines sinusoidal rotators only all the six stages take care for circular rotation matrix.

2.10 Multiplication by two integers using the minimum number of adders (Dempster and Macleod, “Multiplication by Two Integers Using the Minimum Number of Adders”)

This method talks about the minimum adder graph algorithm which designs shifters and adder circuits that aid multiplication by integers using minimum number of adders. It considers the circuit as a graph made up of two input adders. It performs an exhaustive search of all possible graph topologies and produces two Tables out of which one contains the number of adders required to produce the circuit and other contains the partial products of the adders. Further, it also talks about reusing the repeating products which come up while multiplying two numbers. So, if an adder block has the same two inputs as before it doesn't need to calculate that again. Also, it takes care of cases in which one or two inputs of the adders are powers of 2. If that is the case, no extra adders are required as that can be achieved with shifters alone. Thus, the algorithm proposed claims the fewest adders required in a circuit.

2.10.1 Multiplier-less Constant Multiplication Algorithms

(Voronenko and Püschel)

There are many existing algorithms provided in the literature where the constant multiplication is carried out using a network of binary shifters and adders. Implementation on the hardware point of view binary multiplication is very easy and less costly since the hardware cost depends on the number of adders or subtractors used for the constant multiplication. Shifts in the hardware can be implemented by hardwiring. In literature, there are many algorithms available for the multiplierless constant multiplication such as canonic signed digit (CSD), single constant multiplication (SCM), multiple constant multiplications (MCM).

In this approach the binary multiplier can be represented by an equation as below,

$$C = \sum_{i=0}^n -c_i 2^i \quad (2.18)$$

Here in the above equation C is the output and c_i is the coefficient which belongs to set $\{0, 1\}$. Here, the number of adders required for the multiplication are determined by calculating the number of nonzero c_i in the above equation. Thus, in the signed digit notation, the number of adders used for the constant multiplication will be $N-1$ if there are total N nonzero c_i in the above equation.

In the concept of a canonic signed digit c belongs to different set which is $\{-1, 0, 1\}$ and it is represented as $\bar{1}$ in the binary representation of the number. Thus, by including -1 in the set, the number of adders used for the multiplication of constants can be reduced. The method for calculating the number of adders is the same as signed digits which are calculating the total number of nonzero c_i in the binary representation and subtract 1 from it which gives the number of adders. But before calculations, one needs to reduce every two consecutive 1's in the number with -1 representation. For instance, if 7 needs to be multiplied with a constant x i.e. required to find $7x$. The binary representation of 7 can be written as 111, and using shifters and adders x can be multiplied as $(x \ll 2 + x \ll 1 + x)$ which shows that 2 adders are required for multiplication of constant x with 7. But using canonic signed digit method multiplication can be performed using only one adder. hence substituting two consecutive 1's with -1 it can be done as given 11 and can

be written as $10\bar{1}$, by substituting in the same way in 7, it can be written as $100\bar{1}$ which shows that only 1 adder is required for the multiplication of x with 7. This can be represented in the form of adders and shifters as $(x \ll 3 - x)$.

The CSD transform as described above cannot be directly applied to a negative number. However, for finding CSD transform of a negative number, first take CSD transform of a positive number and then flip the sign of nonzero digits in the binary which gives CSD transform for a negative number. Thus using CSD transform number of adders and cost of hardware can be saved, however, it is not the optimal solution.

2.11 Combined Coefficient Selection and Shift and Implementation (CCSSI) (Garrido, Qureshi, and Gustafsson)

This approach uses the combined coefficient selection and shift-and-add implementation (CCSI) method and calculates the total number of coefficients obtained for different cases. The approach does not set any restriction to $C + j S$ selection it selects the best and efficient coefficient which is then used for multiplication with x and y using shift-and-add implementation.

It can solve two types of rotation problems SCR and MCR. The goal here is to find the optimal coefficient and the total number of coefficients with the given input angles based on N point DFT, word length i.e. b bits, maximum allowed error e_{max} and number of allowed adders that can be used, which is represented in the block diagram in the Figure-2.1 below.

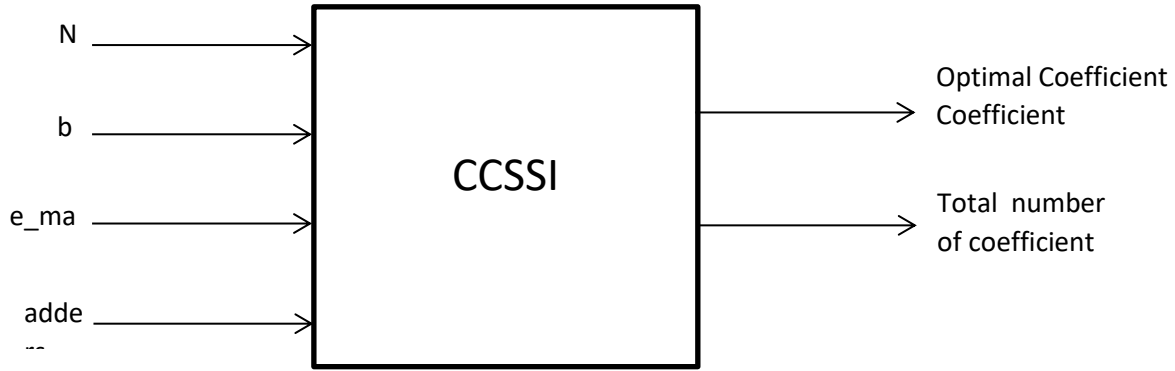


Figure 2.1: Block Diagram of the CCSSI based approach

An example to explain this approach is considered with word length to be $b = 5$, $e_{max} = 0.05$, angle = 14 & 38 and the number of allowed adders is 4.

- 1) Step-1: First complete design space which consists of all possible finite word length values as illustrated in Figure-2.2 (a) for our example. For this consider 2^{2b-2} different coefficient values for every angle in the provided set of angles.
- 2) Step-2: Narrow down the set of coefficient based on the angle $\delta = \sin^{-1} e_{max}$ from the angles taken into consideration. This is shown in the Figure-2.2 (b). Considering the coefficient in the range of angles $\alpha + \delta$ and $\alpha - \delta$, get the coefficient which has the rotation error less than the e_{max} .
- 3) Step-3: Under this step, the coefficients are further reduced based on the scaling. In the example considered, fixed scaling is used. Here the bound for reducing search space based on scaling is taken as $2 * R_{fixed} * e_{max}$, Figure-2.2 (c) illustrate the same.

- 4) Step-4: The number of adders required to implement each rotation is determined. Before that kernels are formed based on the remaining coefficient till Step-3. A kernel is the set of the coefficient for M angles. The next step is the number of adders are calculated for the kernel and then reduced set of kernel based on the maximum adder bound.

- 5) Step-5: Calculate the number of efficient coefficients obtained at the end.

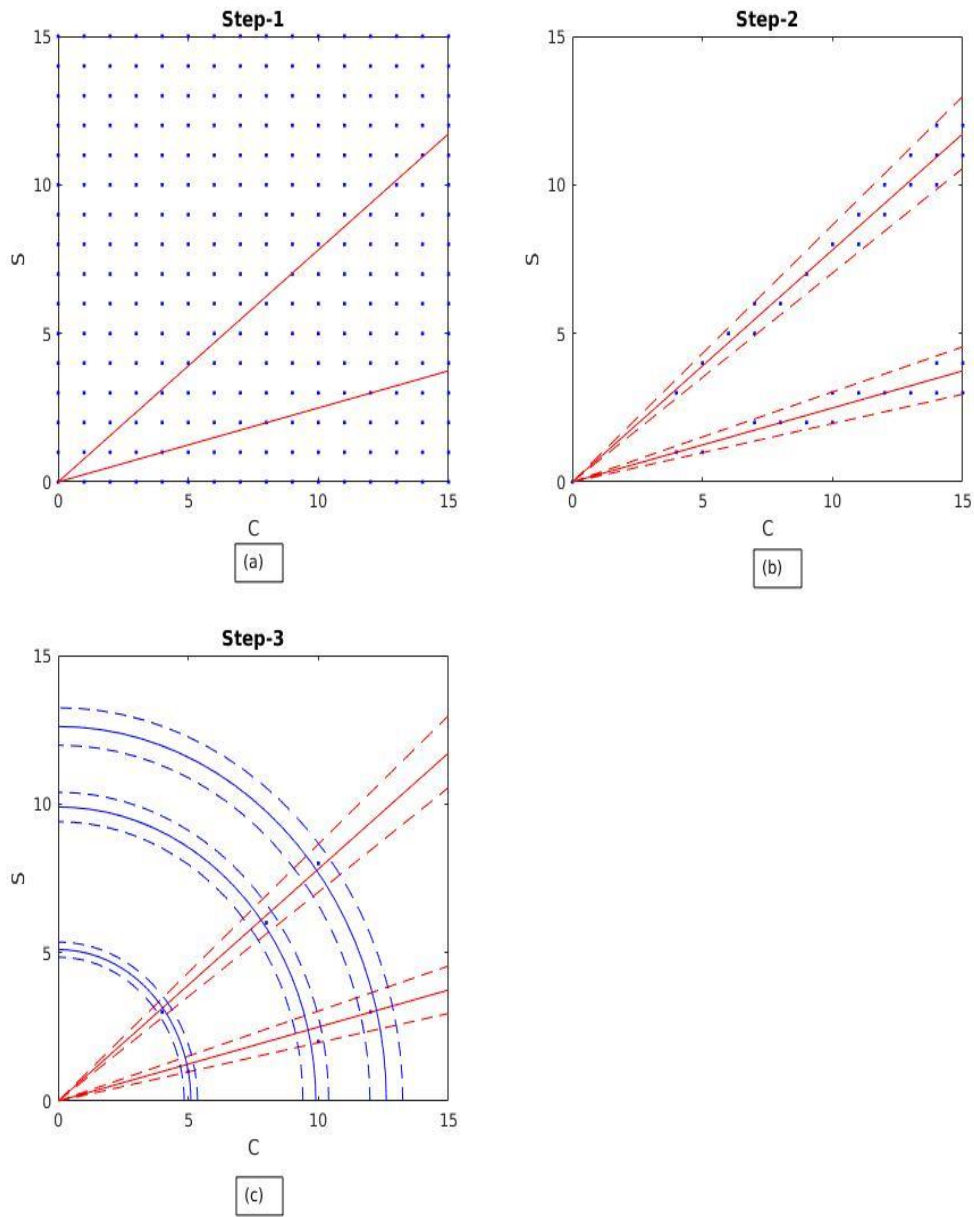


Figure 2.2: Steps for the proposed approach. (a) Initial design space with the required angles. Here are represented angle $\alpha = 14^\circ$ & 38° . (b) Reduced coefficient based on the delta angle. (c) Further reduction of coefficient based on the fixed scaling factor.

Summary

This Chapter discussed various techniques used for the multiplierless algorithms. CORDIC based approaches are well known for implementation of multiplierless architectures however, it carries certain drawbacks which increases the latency of the architecture and hence various versions of CORDIC are discussed. Unified CORDIC approach takes care of three different trajectories viz. Linear, Circular and Hyperbolic and hence it reduces the hardware overhead. However, the common region of convergence and scaling factor are major issues of it. Mixed Scaling Rotations (MSR) CORDIC is an efficient technique to determine the coefficients, however, the SQNR can be improved which can improve the performance of the algorithm

Coefficient Combined Selection and shift and Add implementation (CCSSI) approach gives more flexibility compared to the CORDIC based approaches and it also tries to optimize the hardware for particular tunable parameters like number of bits, number of adders and maximum allowable error. This helps in improving the latency of the architecture by minimizing the hardware required for a particular coefficient to be generated. MCR based coefficient selection pose an important bottleneck for CCSSI based algorithm as it is difficult to find common radius for the given tunable parameters. With CCSSI sinusoidal as well as non-sinusoidal coefficients also can be found.

Chapter 3

Proposed Approach Based On Mixed Scaling CORDIC

Co-ordinate Rotational Digital Computer (CORDIC) is an iterative arithmetic algorithm based on the principles of two dimensional geometry. The algorithm offers simple hardware implementation consisting of shift and add operations. It is suitable for the computation of trigonometric and hyperbolic functions, multiplication and division operations and logarithms. The simplicity of implementing these mathematical operations leads to its applications in Digital Signal Processing, such as Fast Fourier Transformation (FFT) (Heideman et al.; Ahmed and Rao, “Fast Fourier Transform”; James), Eigenvalue Decomposition, Singular Value Decomposition and QR factorization (Walther).

The iterative nature of the conventional CORDIC algorithm affects the speed of computation. Several algorithms have been proposed in the literature such as Angle Recording (AR), Fast CORDIC, Extended Elementary Angle Set (EEAS) , Modified Vector Rotational (MVR), Mixed Scaling Rotation (Meher et al.) amongst others to reduce the number of iterations. MSR-CORDIC can also be seen as the universal vector rotational CORDIC engine encompassing aforementioned algorithms. It significantly reduces the number of iterations thereby improving the speed and enhancing the signal-to-quantization-noise-ratio (SQNR) performance. It offers a unique feature of allowing intermediate vectors to have values other than unity by controlling the amplifying factor. The algorithm can be applied to the applications where the rotation angles are usually known beforehand.e.g. The twiddle factor in FFT (Ahmed and Rao, “Fast Fourier Transform”).

The main theme of the proposed approach lies in redefining the amplifying factor by introducing terms for representing the direction of the rotations. It is based on the principles of

geometry consisting micro-rotations with scaling and MSR-CORDIC algorithm. With redefined amplifying factor, the optimal parameters then can be calculated similar to (Lin and Wu) such that norm error and angle error are minimized at the same time. The main contribution lies in the fact that it provides higher SQNR performance while preserving the features of MSR-CORDIC.

A strong feature of the proposed algorithm is that it does not require additional hardware when compared to the existing MSR-CORDIC implementations.

3.1 Mixed Scaling and Rotation (MSR) CORDIC:

MSR-CORDIC algorithm is designed such that the rotations and scaling operations are performed at the same time. Unlike the conventional CORDIC, the MSR-CORDIC algorithm minimizes the errors in both the angle and norm. It also provides the feature of adjusting the range of the norm. These unique features of MSR-CORDIC provide better SQNR performance, global solution and reduction of round off noise. The algorithm 1 recalls the MSR-CORDIC scheme (Lin and Wu).

Various parameters are as follows: n denotes the n th iteration, N denotes the total number of iterations, $\eta_i(n); \mu_j(n) \in \{1, 0, -1\}$; $s_i(n); t_j(n) \in \{0, 1, \dots, S\}$, where S denotes the number of maximum shifts; I and J denotes the number of signed-power-of-two (SPT) terms of $x(n)$ and $y(n)$ respectively; θ_n is the n th elementary angle; $Z(n)$ is the accumulative angle, and $Z(0)$ is 0; \bar{p}_n denotes the product of the amplifying factors in the n th iteration, and \bar{p}_0 is 1. P denotes the scaling factor, and N_{spt} is denoted as the SPT term used which is the sum of I and J .

The design parameters $s_i(n)$, $t_j(n)$, $\eta_i(n)$ and $\mu_j(n)$ are selected such that the angle error $|Z(N) - \theta|$ and norm error $|1 - P|$ are minimized at the same time; where θ is the targeted angle.

Algorithm 1 MSR-CORDIC Scheme

1: for $n: =1$ to N

2: Perform micro-rotations and scaling

$$\begin{bmatrix} x(n) \\ y(n) \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^I \eta_i(n) 2^{-s_i(n)} & \sum_{j=1}^J \mu_j(n) 2^{-t_j(n)} \\ \sum_{j=1}^J \mu_j(n) 2^{-t_j(n)} & \sum_{i=1}^I \eta_i(n) 2^{-s_i(n)} \end{bmatrix} \begin{bmatrix} x(n-1) \\ y(n-1) \end{bmatrix} \quad (3.1)$$

3: Calculate elementary angle

$$\theta_n = \tan^{-1} \left(\frac{\sum_{j=1}^J \mu_j(n) 2^{-t_j(n)}}{\sum_{i=1}^I \eta_i(n) 2^{-s_i(n)}} \right) \quad (3.2)$$

4: Update accumulation angle

$$Z(n) = Z(n-1) + \theta_n \quad (3.3)$$

5: Amplifying factor in the n th rotation

$$p_n = \sqrt{\left(\sum_{i=1}^I 2^{-s_i(n)} \right)^2 + \left(\sum_{j=1}^J 2^{-t_j(n)} \right)^2} \quad (3.4)$$

6: Product of the amplifying factor in the n th rotation

$$\overline{p}_n = \overline{p}_{n-1} \times p_n \quad (3.5)$$

7: end

8: Scaling factor

$$P = \prod_{n=1}^N p_n \quad (3.6)$$

$|1 - P|$ are minimized at the same time; where θ is the targeted angle.

3.2 The Proposed Enhanced MSR Scheme

Given a rotation angle θ and vector $[x, y]^T$, the resultant vector $[x', y']^T$ can be computed as follows:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (3.7)$$

The resultant angle can be decomposed of multiple angles by using the concept of micro-rotations. Hence, (3.7) gets modified to

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \left(\prod_{i=1}^N \begin{bmatrix} \cos \theta_n & -\sin \theta_n \\ \sin \theta_n & \cos \theta_n \end{bmatrix} \right) \begin{bmatrix} x \\ y \end{bmatrix} \quad (3.8)$$

Where, $\theta = \sum_{n=1}^N \theta_n$

N is the total number of rotations and θ_n is the n th rotation angle.

The principal theme of the proposed work is to replace (3.8) by scaled products such that

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \left(\prod_{i=1}^N \begin{bmatrix} v_n \cos \theta_n & -v_n \sin \theta_n \\ v_n \sin \theta_n & v_n \cos \theta_n \end{bmatrix} \right) \begin{bmatrix} x \\ y \end{bmatrix} \quad (3.9)$$

Where, v_n is the scaling factor,

$$\theta = \sum_{n=1}^N \theta_n \quad (3.10)$$

And

$$\prod_{n=1}^N v_n = 1 \quad (3.11)$$

It can be observed from (3.10) that scaling operation is performed in parallel to the micro-rotation. It is also important to note that the intermediate vectors may have norms other than unity. Though, to retain the norm of the original vector, multiplication of all the scaling factors shall be unity. The trivial equality $\cos^2 \theta + \sin^2 \theta = 1$ and (3.12) can be utilized to produce,

$$\prod_{n=1}^N (\sqrt{(v_n \cos \theta_n)^2 + (v_n \sin \theta_n)^2}) = 1 \quad (3.12)$$

Further, it can be observed that there is a stark similarity between the structure of MSR-CORDIC and the method of micro-rotations coupled with scaling as used in (3.10) – (3.12).

The proposed novel MSR-CORDIC algorithm uses this concept in redefining the original MSR-CORDIC. Equation (3.1) from MSR-CORDIC is analogous to equation (3.9). Based on (3.13) the amplifying and the scaling factors defined in the algorithm 1 are modified as

$$v_n = \sqrt{\left(\sum_{i=1}^I \eta_i(n)2^{-si(n)}\right)^2 + \left(\sum_{j=1}^J \mu_j(n)2^{-tj(n)}\right)^2} \quad (3.13)$$

$$V = \prod_{n=1}^N v_n \quad (3.14)$$

The redefined amplifying factor contains additional terms $\eta_i(n)$ and $\mu_j(n)$. All the remaining equations (3.1) – (3.3) and (3.5) – (3.6) remain the same. The summary of the same is given in algorithm 2.

The proposed new MSR-CORDIC preserves all the features provided by the MSR-CORDIC. It can be adopted for both normal and generalized MSR-CORDIC schemes. Further, it is important to note that there is no need of any additional adders or shifters in comparison to the conventional MSRCORDIC schemes. The boundary condition in the proposed scheme remains the same as of the classical scheme, i.e. $v_{upper} = p_{upper}, v_{lower} = p_{lower}$. With the same hardware complexity, the proposed MSR-CORDIC provides greater SQNR performance.

3.3 The Proposed Algorithm

Algorithm 2 Proposed MSR-CORDIC Scheme with weighted amplifying factors

1: for $n: =1$ to N

2: Perform micro-rotations and scaling using equation

$$\begin{bmatrix} x(n) \\ y(n) \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^I \eta_i(n) 2^{-si(n)} & \sum_{j=1}^J \mu_j(n) 2^{-tj(n)} \\ \sum_{j=1}^J \mu_j(n) 2^{-tj(n)} & \sum_{i=1}^I \eta_i(n) 2^{-si(n)} \end{bmatrix} \begin{bmatrix} x(n-1) \\ y(n-1) \end{bmatrix} \quad (3.15)$$

3: Calculate elementary angle using equation

$$\theta_n = \tan^{-1} \left(\frac{\sum_{j=1}^J \mu_j(n) 2^{-tj(n)}}{\sum_{i=1}^I \eta_i(n) 2^{-si(n)}} \right) \quad (3.16)$$

4: Update accumulation angle using equation

$$Z(n) = Z(n-1) + \theta_n \quad (3.17)$$

5: *Weighted Amplifying factor in the nth rotation*

$$v_n = \sqrt{\left(\sum_{i=1}^I \eta_i(n)2^{-si(n)}\right)^2 + \left(\sum_{j=1}^J \mu_j(n)2^{-tj(n)}\right)^2} \quad (3.18)$$

6: *Product of the amplifying factor in the nth rotation*

$$\overline{v}_n = \overline{v}_{n-1} X v_n$$

Where $v_0 = 1$

(3.19)

7: *end*

8: *Scaling factor*

$$V = \prod_{n=1}^N v_n$$

(3.20)

3.4 Results and Discussions

A comparison of the proposed scheme and MSR-CORDIC in terms of the SQNR outcome is presented in this section. It is shown that the proposed scheme results in better SQNR performance. The constraints are fixed in the same way as outlined in (Lin and Wu) for the purpose of simplicity and fairness. An exhaustive search for each type of constraint is carried out to generate 512 distinct parameters sets of $s_i(n)$, $t_j(n)$, $\eta_i(n)$ and $\mu_j(n)$.

MSR-CORDIC offers two sets of schemes, namely, Normalized MSR-CORDIC and Generalized MSR-CORDIC. A comparison between the proposed scheme and MSR-CORDIC is presented in Fig.1 with NSPT = 4. Normalized scheme takes any one set of (I; J) that satisfies $I + J = 4$, i.e. (4; 0), (0; 4), (1; 3), (3; 1) and (2; 2). It is important to note that sets (4; 0) and (0; 4) offer only scaling operation and hence they are not considered for the comparison. The SQNR performance of (1; 3) and (3; 1) are same. Hence, only unique combinations such as (1; 3) is taken into account for the simulations. Unlike the normalized scheme which has the fixed choice of combinations for (I; J), Generalized scheme selects the combination which minimizes the angle error $|Z(N) - \theta|$ and norm error $|1 - V|$ the most at the same time.

The following observations can be made from Figure 3.1:

- 1) For the proposed method similar to the MSR-CORDIC, the generalized scheme offers the better SQNR performance when compared to the Normalized scheme. Furthermore, the SQNR performance of (2; 2) is higher when compared to (1; 3).
- 2) The plot shows that the proposed scheme offers higher SQNR performance for both the schemes when compared with that provided by corresponding MSR CORDIC counterparts.

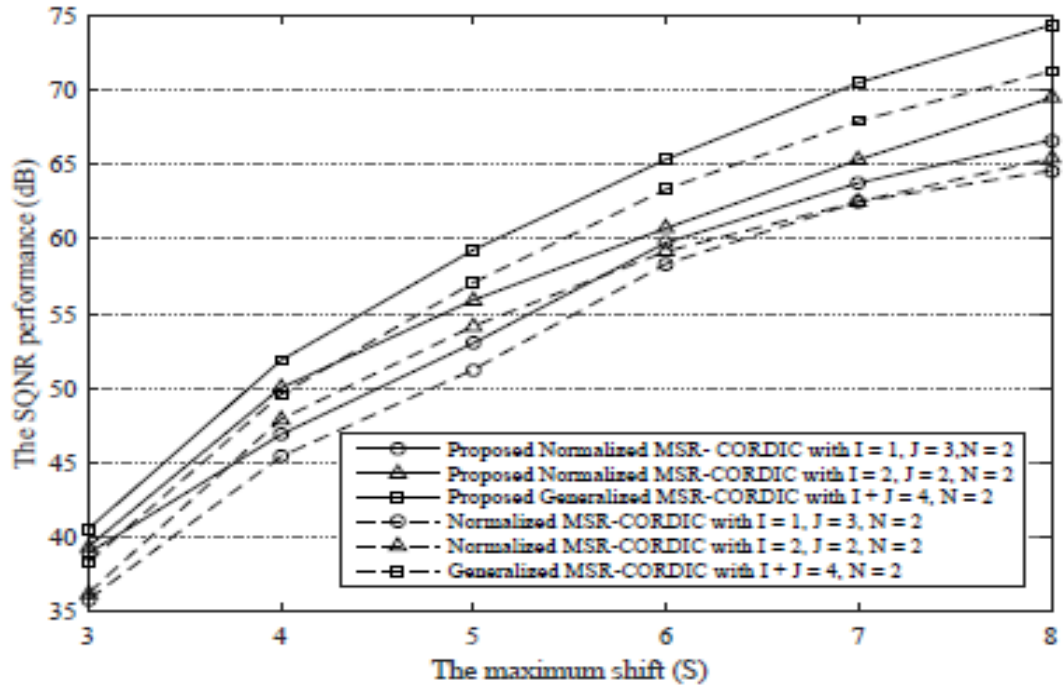


Figure.3.1 SQNR Comparison between MSR_CORDIC and the proposed scheme

The scaling factor in the conventional CORDIC algorithm is fixed as per the number of iterations. Though, with other algorithms such as Angle recoding, MVR and EEAS (Meher et al.), the scaling factor changes with every iteration. This leads to higher round-off noise error and hence deteriorates SQNR performance. The word length can be defined if range of the scaling factor is known a priori. Thus, the round-off noise can be reduced.

MSR-CORDIC allows to determine the range for the scaling factor such that $p_{upper} < \overline{p_n} < p_{lower}$ holds true. The parameter p_{lower} is fixed as $1/p_{upper}$ as per the boundary constraint explained in (Lin and Wu) and the same holds true for $\overline{v_n}$.

The analysis of SQNR performance with the change in scaling factor is depicted in Figure. 3.2. The parameters are selected as NSPT = 3, N = 3 and NSPT = 4, N = 2 for Figure. 3.2 (a) and Figure. 3.2 (b) respectively. Following can be observed from the plot:

- 1) Similar to MSR-CORDIC, the proposed scheme saturates when v_{upper} is 1.5.
- 2) The proposed scheme has better SQNR performance for the same parameters. When v_{upper} value reaches 1.3, the SQNR performance of the proposed scheme is better than the saturated SQNR value of MSR-CORDIC.

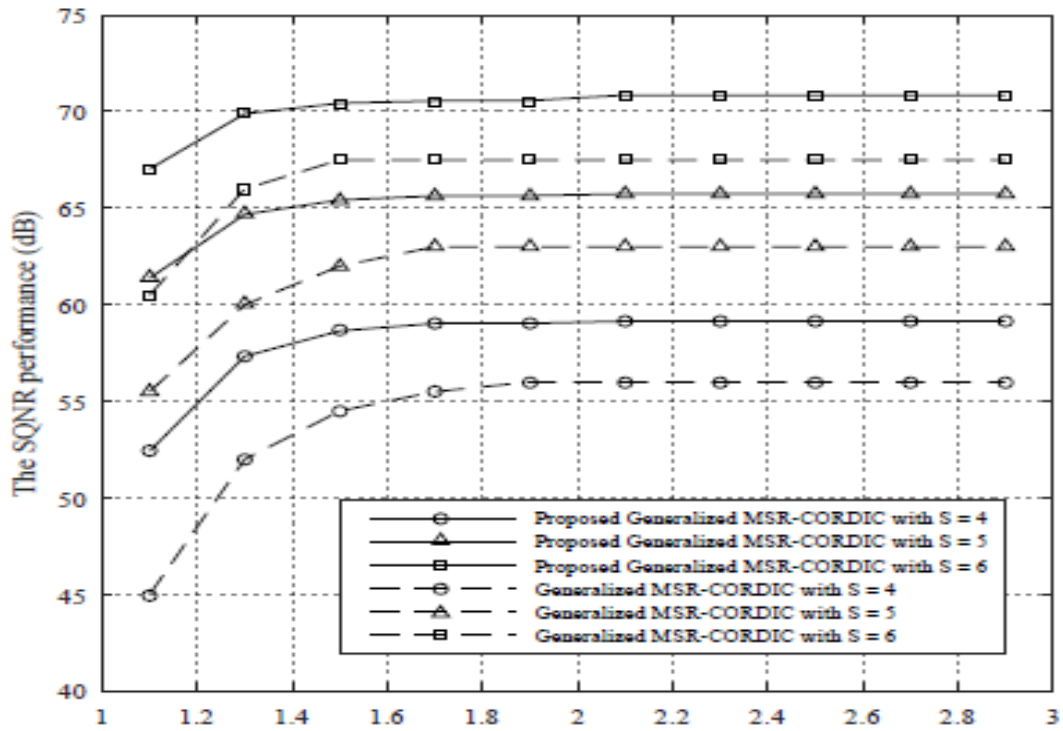


Figure 3.2 (a) Comparing the relationship between SQNR performances and scaling factor value of v_{upper} in generalized scheme of MSR-CORDIC and proposed MSR-CORDIC NSPT = 3 and N = 3.

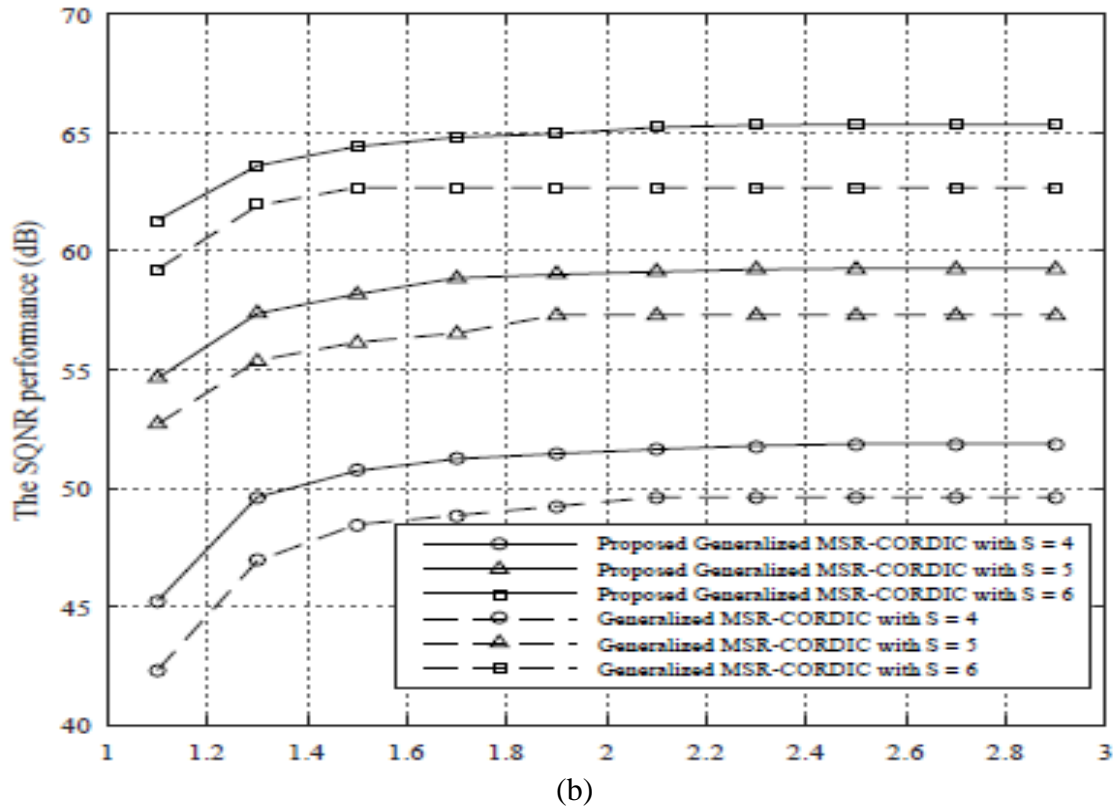


Figure 3.2 (b) Comparing the relationship between SQNR performances and scaling factor value of v_{upper} in generalized scheme of MSR-CORDIC and proposed MSR-CORDIC. NSPT = 4 and N = 2.

The analysis of the SQNR performance with different NSPT is shown in Figure 3.3. For a comparison of both, conventional and proposed MSR-CORDIC schemes, the parameters are selected as NSPT = 3 and NSPT = 4 with N = 2. It can be observed that the performance of the higher NSPT term is better in both the schemes. Also, the SQNR performance of the proposed scheme is better when compared with the same NSPT term of the other scheme.

A further observation can be made that the hardware complexity of the proposed algorithm is the same as MSR CORDIC since scaling and micro-rotation equation for both the algorithm remains analogous. Hence, the proposed algorithm enhances the SQNR performance without adding hardware complexity. During the extensive and numerous simulations runs in addition to

those being reported, no instance could be found where the proposed scheme resulted in inferior SQNR performance than that afforded by the conventional MSR-CORDIC algorithm.

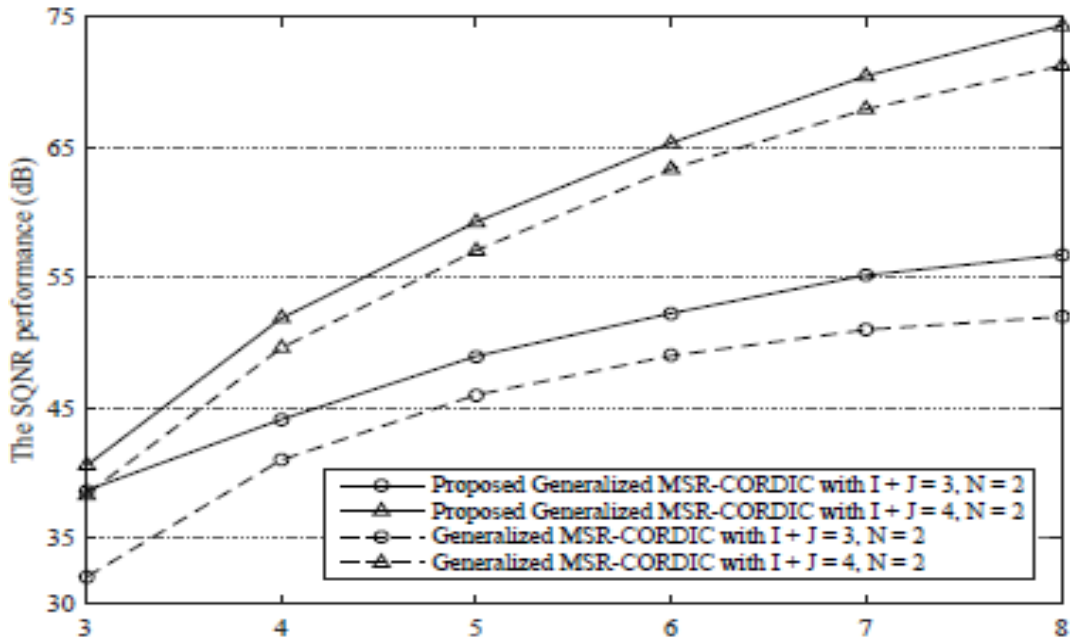


Figure 3.3 Analysis of MSR-CORDIC and proposed scheme for different combinations of NSPT.

Summary

An enhanced MSR-CORDIC algorithm is proposed in this chapter that employs weighted amplifying factors. The proposed algorithm can be implemented with both, Generalized and Normalized schemes. The algorithm provides better SQNR performance with no added hardware complexity. The results are compared with the existing MSR approach and it is found that the proposed generalized MSR CORDIC shows 6.6 % improvement for a given value of $I + J = 3$ and $N = 2$ in the SQNR compared to the existing MSR CORDIC algorithm.

Chapter 4

CCSSI based Proposed Approach

4.1 Proposed Approach based on CCSSI

The proposed approach uses the combined coefficient selection and shift-and-add implementation (CCSI) (Garrido, Qureshi, and Gustafsson) technique and calculates the total number of coefficient obtained for the different cases. This approach does not set any restriction to C+jS selection, it selects the best and efficient coefficient which is then used for multiplication with x and y using shift-and-add implementation. This approach is further extended to 64 point FFT and variation of the number of coefficient with the bits, Number of points and Number of adders is obtained. The design process which is followed for this work is as follow

Proposed approach can solve two types of rotation problems SCR and MCR. The goal here is to find the optimal coefficient and total number of coefficients with the given input angles based on N point transform, word length i.e. b bits, maximum allowed error e_{max} and number of allowed adders that can be used, which is represented in the block diagram in the Figure-4.1 below.

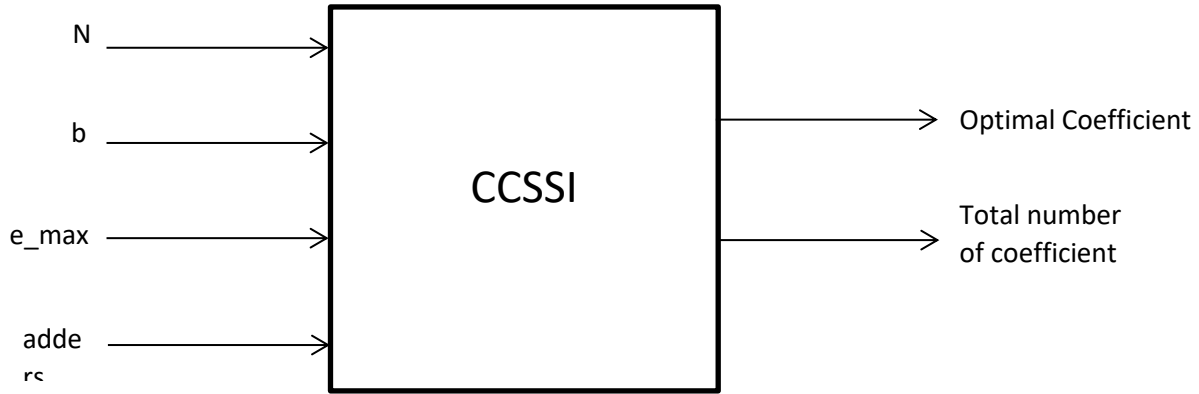


Figure 4.1: Block Diagram of proposed approach

An example to explain the proposed approach is considered where, word length is taken to be $b = 5$, $e_{max} = 0.05$, angle = 14 & 38 and number of allowed adders is 4.

Step-1: First complete design space which consists of all possible finite word length values as illustrated in Figure-4.2 (a). For this, 2^{2b-2} different coefficient are considered for every angle in the provided set of angles.

Step-2: Narrow down the set of coefficient based on the angle $\delta = \sin^{-1} e_{max}$ from the angles taken into consideration. This is shown in the Figure-4.2 (b). Considering the coefficient in the range of *angles* $\alpha + \delta$ and $\alpha - \delta$, get the coefficient which has the rotation error less than the e_{max} .

Step-3: Under this step, the coefficients are further reduced based on the scaling. In the example considered, fixed scaling is used. Here the bound for reducing search space based on scaling is taken as $2 * R_{fixed} * e_{max}$, Figure-4.2 (c) illustrate the same.

Step-4: The number of adders required to implement each rotation is determined. Before that kernels are formed based on the remaining coefficient till Step-3. A kernel is the set of the coefficient for M angles. The next step is the number of adders are calculated for the kernel and then reduced set of kernel based on the maximum adder bound.

Step-5: Calculate the number of efficient coefficients obtained at the end.

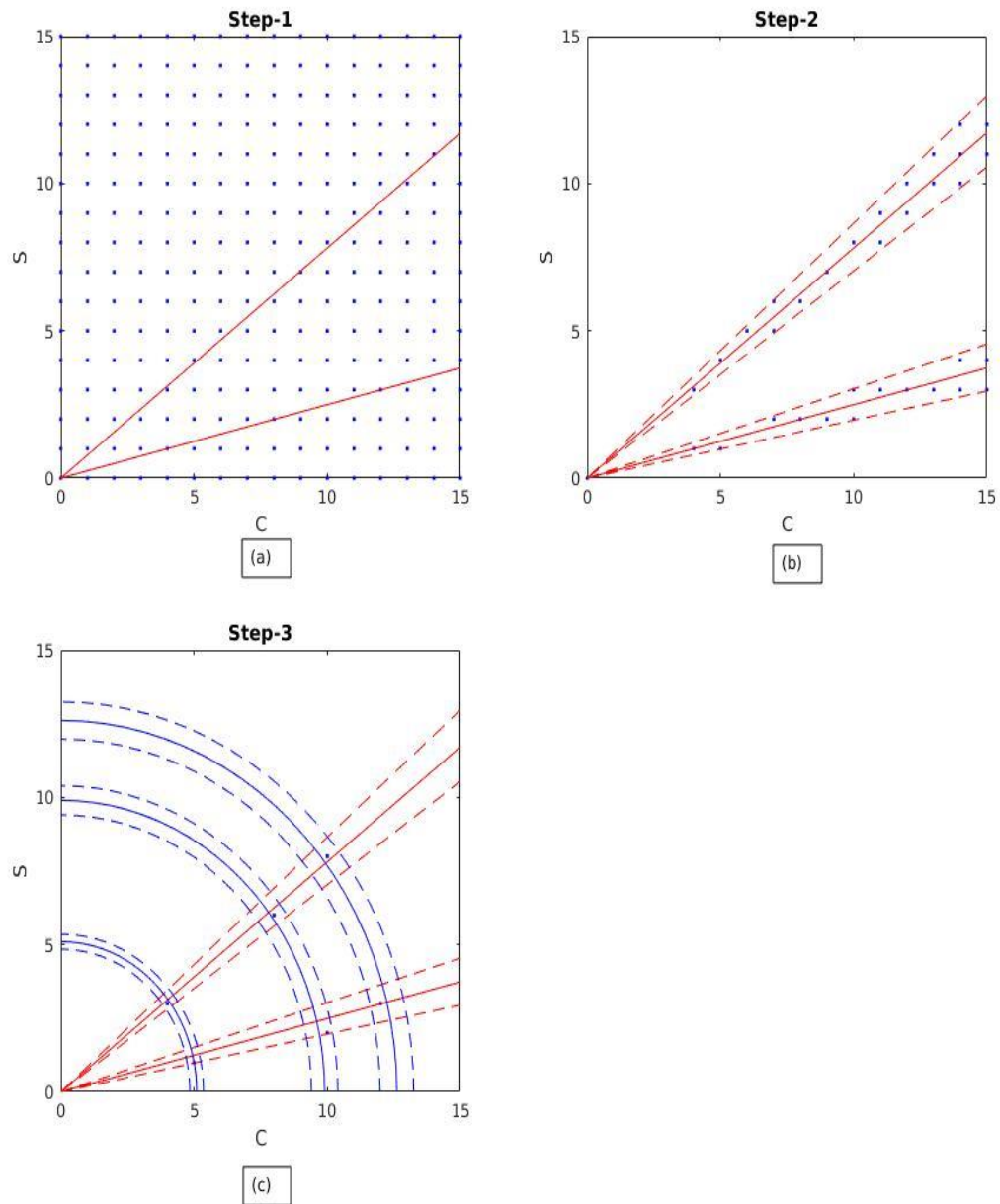


Figure 4.2: Steps for the proposed approach. (a) Initial design space with the required angles. Here are represented angle $\alpha = 14$ & 38 . (b) Reduced coefficient based on delta angle. (c) Further reduction of coefficient based on fixed scaling factor.

Table 4.1: MCR Remaining Kernel for $b = 5$ and $\text{max.error} = 0.05$ for two angles

| $b = 5, \text{max.error} = 0.05$ | | | |
|----------------------------------|------------|---------|-----------|
| Angle 1 | Angle 2 | R | Error |
| 14 | $10 + 10i$ | 14.0711 | 0.0050506 |
| 15 | $11 + 11i$ | 15.2782 | 0.018207 |
| 15 | $10 + 10i$ | 14.5711 | 0.029437 |
| 14 | $10 + 11i$ | 14.719 | 0.048849 |
| 14 | $11 + 10i$ | 14.719 | 0.048849 |
| 15 | $11 + 12i$ | 15.8296 | 0.052408 |

The Number of kernels is calculated with the algorithm described above for different parameters viz. number of bits (b) and maximum error and the observations are indicated in the Tables 4.1 to 4.8.

It is observed from the Table 4.1, that for maximum error bound of 0.05 and number of bits of space equals 5, 24 different coefficients are generated which can implement five angles. However it is important to note that all the different kernels generated are accumulated around the radius of 14 and 15 approximately which indicates that the CCSSI is suitable of generating the common radius used in the algorithms like Fast Fourier transform and other sinusoidal transforms.

To observe the effect of number of bits on the total number of coefficients generated and the error bound, The number of bits are from varied $b=5$ till $b=8$. Table 4.2, 4.3 and 4.4 depicts the results. As the number of bits or the word length increases, the number of coefficient increases as well in the given space and hence more coefficients are included.

Table 4.2: MCR Remaining Kernel for $b = 6$ and $\text{max.error} = 0.05$ for two angles

| b = 6, max.error = 0.05 | | | |
|-------------------------|------------|---------|----------|
| Angle 1 | Angle 2 | R | Error |
| 14 | $10 + 10i$ | 14.0711 | 0.005051 |
| 16 | $11 + 11i$ | 15.7782 | 0.014059 |
| 15 | $11 + 11i$ | 15.2782 | 0.018207 |
| 15 | $10 + 10i$ | 14.5711 | 0.029437 |
| 14 | $10 + 11i$ | 14.719 | 0.048849 |
| 14 | $11 + 10i$ | 14.719 | 0.048849 |

Table 4.3: MCR Remaining Kernel for $b = 7$ and $\text{max.error} = 0.05$ for two angles

| b = 7, max.error = 0.05 | | | |
|-------------------------|------------|---------|----------|
| Angle 1 | Angle 2 | R | Error |
| 14 | $10 + 10i$ | 14.0711 | 0.005051 |
| 16 | $11 + 11i$ | 15.7782 | 0.014059 |
| 15 | $11 + 11i$ | 15.2782 | 0.018207 |
| 15 | $10 + 10i$ | 14.5711 | 0.029437 |
| 14 | $10 + 11i$ | 14.719 | 0.048849 |
| 14 | $11 + 10i$ | 14.719 | 0.048849 |

Table 4.4: MCR Remaining Kernel for $b = 8$ and $\text{max.error} = 0.05$ for two angles

| b = 8, max.error = 0.05 | | | |
|-------------------------|------------|---------|----------|
| Angle 1 | Angle 2 | R | Error |
| 14 | $10 + 10i$ | 14.0711 | 0.005051 |
| 16 | $11 + 11i$ | 15.7782 | 0.014059 |
| 15 | $11 + 11i$ | 15.2782 | 0.018207 |
| 15 | $10 + 10i$ | 14.5711 | 0.029437 |
| 14 | $10 + 11i$ | 14.719 | 0.048849 |
| 14 | $11 + 10i$ | 14.719 | 0.048849 |

The results were extended for three, four and five different angles as well and Tables 4.5, 4.6, 4.7 and 4.8 shows the different coefficients found for five angles for radius equal to 15. Increasing the number of angles i.e. N, also be termed as number of points in transforms like FFT, the number of unique coefficients found for an error bound of 0.05 are closer to the radius of 15. For the Table 4.5 below the Coefficient combined and shift and add implementation algorithm is able to generate four angles closer to the radius and only Angle-5 needs to be taken outside of the error bound of 0.05. Hence, for number of points like $N=256,512$ or 1024 which is required in applications like OFDM. The algorithm can be found suitable for finding maximum number of coefficients for a given error bound.

Table 4.5: MCR Remaining Kernel for $b = 5$ and $\text{max.error} = 0.05$ for five angles

| b = 5, max.error = 0.05 | | | | | | |
|-------------------------|---------|---------|---------|----------|---------|----------|
| Angle-1 | Ange-2 | Angle-3 | Angle-4 | Angle-5 | R | Error |
| 14 | 14 + 3i | 13 + 5i | 12 + 8i | 10 + 10i | 14.0158 | 0.029014 |
| 15 | 15 + 3i | 14 + 6i | 13 + 9i | 11 + 11i | 15.5488 | 0.035295 |
| 15 | 15 + 3i | 14 + 6i | 12 + 8i | 11 + 11i | 15.0069 | 0.038974 |
| 15 | 15 + 3i | 15 + 6i | 13 + 8i | 11 + 11i | 15.5447 | 0.04121 |
| 15 | 14 + 3i | 14 + 6i | 12 + 8i | 10 + 10i | 14.7678 | 0.042369 |
| 15 | 14 + 3i | 14 + 6i | 13 + 8i | 11 + 11i | 14.9349 | 0.043767 |
| 14 | 15 + 3i | 13 + 6i | 12 + 8i | 10 + 10i | 14.6388 | 0.044977 |
| 15 | 15 + 3i | 13 + 6i | 12 + 8i | 10 + 10i | 14.6388 | 0.044977 |
| 14 | 14 + 3i | 13 + 6i | 11 + 8i | 10 + 10i | 13.9702 | 0.047274 |
| 15 | 14 + 3i | 14 + 6i | 13 + 8i | 10 + 11i | 14.8359 | 0.04767 |
| 15 | 15 + 3i | 14 + 6i | 13 + 8i | 10 + 11i | 14.8359 | 0.04767 |
| 15 | 14 + 3i | 14 + 6i | 13 + 8i | 11 + 10i | 14.8359 | 0.04767 |
| 15 | 15 + 3i | 14 + 6i | 13 + 8i | 11 + 10i | 14.8359 | 0.04767 |
| 15 | 14 + 3i | 14 + 6i | 12 + 8i | 10 + 11i | 15.01 | 0.048311 |
| 15 | 14 + 3i | 14 + 6i | 12 + 8i | 11 + 10i | 15.01 | 0.048311 |
| 15 | 14 + 3i | 13 + 6i | 12 + 8i | 10 + 11i | 14.7409 | 0.048529 |
| 15 | 15 + 3i | 13 + 6i | 12 + 8i | 10 + 11i | 14.7409 | 0.048529 |
| 15 | 14 + 3i | 13 + 6i | 12 + 8i | 11 + 10i | 14.7409 | 0.048529 |

| | | | | | | |
|-----------|-----------|-----------|-----------|------------|---------|----------|
| 15 | $15 + 3i$ | $13 + 6i$ | $12 + 8i$ | $11 + 10i$ | 14.7409 | 0.048529 |
| 15 | $15 + 3i$ | $14 + 6i$ | $13 + 9i$ | $10 + 11i$ | 15.1043 | 0.049766 |
| 15 | $15 + 3i$ | $14 + 6i$ | $13 + 9i$ | $11 + 10i$ | 15.1043 | 0.049766 |

Table 4.6: MCR Remaining Kernel for $b = 6$ and $\text{max.error} = 0.05$ for five angles

| b = 6, max.error = 0.05 | | | | | | |
|-------------------------|-----------|-----------|-----------|------------|---------|----------|
| Angle-1 | Ange-2 | Angle-3 | Angle-4 | Angle-5 | R | Error |
| 14 | $14 + 3i$ | $13 + 5i$ | $12 + 8i$ | $10 + 10i$ | 14.0158 | 0.029014 |
| 15 | $15 + 3i$ | $14 + 6i$ | $13 + 9i$ | $11 + 11i$ | 15.5488 | 0.035295 |
| 15 | $15 + 3i$ | $14 + 6i$ | $12 + 8i$ | $11 + 11i$ | 15.0069 | 0.038974 |
| 15 | $15 + 3i$ | $15 + 6i$ | $13 + 8i$ | $11 + 11i$ | 15.5447 | 0.04121 |
| 15 | $14 + 3i$ | $14 + 6i$ | $12 + 8i$ | $10 + 10i$ | 14.7678 | 0.042369 |
| 15 | $14 + 3i$ | $14 + 6i$ | $13 + 8i$ | $11 + 11i$ | 14.9349 | 0.043767 |
| 14 | $15 + 3i$ | $13 + 6i$ | $12 + 8i$ | $10 + 10i$ | 14.6388 | 0.044977 |
| 15 | $15 + 3i$ | $13 + 6i$ | $12 + 8i$ | $10 + 10i$ | 14.6388 | 0.044977 |
| 14 | $14 + 3i$ | $13 + 6i$ | $11 + 8i$ | $10 + 10i$ | 13.9702 | 0.047274 |
| 15 | $14 + 3i$ | $14 + 6i$ | $13 + 8i$ | $10 + 11i$ | 14.8359 | 0.04767 |
| 15 | $15 + 3i$ | $14 + 6i$ | $13 + 8i$ | $10 + 11i$ | 14.8359 | 0.04767 |
| 15 | $14 + 3i$ | $14 + 6i$ | $13 + 8i$ | $11 + 10i$ | 14.8359 | 0.04767 |
| 15 | $15 + 3i$ | $14 + 6i$ | $13 + 8i$ | $11 + 10i$ | 14.8359 | 0.04767 |
| 15 | $14 + 3i$ | $14 + 6i$ | $12 + 8i$ | $10 + 11i$ | 15.01 | 0.048311 |

| | | | | | | |
|----|-----------|-----------|-----------|------------|---------|----------|
| 15 | $14 + 3i$ | $14 + 6i$ | $12 + 8i$ | $11 + 10i$ | 15.01 | 0.048311 |
| 15 | $14 + 3i$ | $13 + 6i$ | $12 + 8i$ | $10 + 11i$ | 14.7409 | 0.048529 |
| 15 | $15 + 3i$ | $13 + 6i$ | $12 + 8i$ | $10 + 11i$ | 14.7409 | 0.048529 |
| 15 | $14 + 3i$ | $13 + 6i$ | $12 + 8i$ | $11 + 10i$ | 14.7409 | 0.048529 |
| 15 | $15 + 3i$ | $13 + 6i$ | $12 + 8i$ | $11 + 10i$ | 14.7409 | 0.048529 |
| 15 | $15 + 3i$ | $14 + 6i$ | $13 + 9i$ | $10 + 11i$ | 15.1043 | 0.049766 |
| 15 | $15 + 3i$ | $14 + 6i$ | $13 + 9i$ | $11 + 10i$ | 15.1043 | 0.049766 |

Table 4.7: MCR Remaining Kernel for $b = 7$ and $\text{max.error} = 0.05$ for five different angles

| $b = 7, \text{max.error} = 0.05$ | | | | | | |
|----------------------------------|-----------|-----------|-----------|------------|---------|----------|
| Angle-1 | Angle-2 | Angle-3 | Angle-4 | Angle-5 | R | Error |
| 14 | $14 + 3i$ | $13 + 5i$ | $12 + 8i$ | $10 + 10i$ | 14.0158 | 0.029014 |
| 15 | $15 + 3i$ | $14 + 6i$ | $13 + 9i$ | $11 + 11i$ | 15.5488 | 0.035295 |
| 15 | $15 + 3i$ | $14 + 6i$ | $12 + 8i$ | $11 + 11i$ | 15.0069 | 0.038974 |
| 15 | $15 + 3i$ | $15 + 6i$ | $13 + 8i$ | $11 + 11i$ | 15.5447 | 0.04121 |
| 15 | $14 + 3i$ | $14 + 6i$ | $12 + 8i$ | $10 + 10i$ | 14.7678 | 0.042369 |
| 15 | $14 + 3i$ | $14 + 6i$ | $13 + 8i$ | $11 + 11i$ | 14.9349 | 0.043767 |
| 14 | $15 + 3i$ | $13 + 6i$ | $12 + 8i$ | $10 + 10i$ | 14.6388 | 0.044977 |
| 15 | $15 + 3i$ | $13 + 6i$ | $12 + 8i$ | $10 + 10i$ | 14.6388 | 0.044977 |
| 14 | $14 + 3i$ | $13 + 6i$ | $11 + 8i$ | $10 + 10i$ | 13.9702 | 0.047274 |
| 15 | $14 + 3i$ | $14 + 6i$ | $13 + 8i$ | $10 + 11i$ | 14.8359 | 0.04767 |
| 15 | $15 + 3i$ | $14 + 6i$ | $13 + 8i$ | $10 + 11i$ | 14.8359 | 0.04767 |
| 15 | $14 + 3i$ | $14 + 6i$ | $13 + 8i$ | $11 + 10i$ | 14.8359 | 0.04767 |

| | | | | | | |
|----|---------|---------|---------|----------|---------|----------|
| 15 | 15 + 3i | 14 + 6i | 13 + 8i | 11 + 10i | 14.8359 | 0.04767 |
| 15 | 14 + 3i | 14 + 6i | 12 + 8i | 10 + 11i | 15.01 | 0.048311 |
| 15 | 14 + 3i | 14 + 6i | 12 + 8i | 11 + 10i | 15.01 | 0.048311 |
| 15 | 14 + 3i | 13 + 6i | 12 + 8i | 10 + 11i | 14.7409 | 0.048529 |
| 15 | 15 + 3i | 13 + 6i | 12 + 8i | 10 + 11i | 14.7409 | 0.048529 |
| 15 | 14 + 3i | 13 + 6i | 12 + 8i | 11 + 10i | 14.7409 | 0.048529 |
| 15 | 15 + 3i | 13 + 6i | 12 + 8i | 11 + 10i | 14.7409 | 0.048529 |
| 15 | 15 + 3i | 14 + 6i | 13 + 9i | 10 + 11i | 15.1043 | 0.049766 |
| 15 | 15 + 3i | 14 + 6i | 13 + 9i | 11 + 10i | 15.1043 | 0.049766 |

Table 4.8: MCR Remaining Kernel for $b = 8$ and $\text{max.error} = 0.05$ for five angles

| b = 8, max.error = 0.05 | | | | | | |
|-------------------------|---------|---------|---------|----------|---------|----------|
| Angle-1 | Angle-2 | Angle-3 | Angle-4 | Angle-5 | R | Error |
| 14 | 14 + 3i | 13 + 5i | 12 + 8i | 10 + 10i | 14.0158 | 0.029014 |
| 15 | 15 + 3i | 14 + 6i | 13 + 9i | 11 + 11i | 15.5488 | 0.035295 |
| 15 | 15 + 3i | 14 + 6i | 12 + 8i | 11 + 11i | 15.0069 | 0.038974 |
| 15 | 15 + 3i | 15 + 6i | 13 + 8i | 11 + 11i | 15.5447 | 0.04121 |
| 15 | 14 + 3i | 14 + 6i | 12 + 8i | 10 + 10i | 14.7678 | 0.042369 |
| 15 | 14 + 3i | 14 + 6i | 13 + 8i | 11 + 11i | 14.9349 | 0.043767 |
| 14 | 15 + 3i | 13 + 6i | 12 + 8i | 10 + 10i | 14.6388 | 0.044977 |
| 15 | 15 + 3i | 13 + 6i | 12 + 8i | 10 + 10i | 14.6388 | 0.044977 |
| 14 | 14 + 3i | 13 + 6i | 11 + 8i | 10 + 10i | 13.9702 | 0.047274 |
| 15 | 14 + 3i | 14 + 6i | 13 + 8i | 10 + 11i | 14.8359 | 0.04767 |
| 15 | 15 + 3i | 14 + 6i | 13 + 8i | 10 + 11i | 14.8359 | 0.04767 |

| | | | | | | |
|----|---------|---------|---------|----------|---------|----------|
| 15 | 14 + 3i | 14 + 6i | 13 + 8i | 11 + 10i | 14.8359 | 0.04767 |
| 15 | 15 + 3i | 14 + 6i | 13 + 8i | 11 + 10i | 14.8359 | 0.04767 |
| 15 | 14 + 3i | 14 + 6i | 12 + 8i | 10 + 11i | 15.01 | 0.048311 |
| 15 | 14 + 3i | 14 + 6i | 12 + 8i | 11 + 10i | 15.01 | 0.048311 |
| 15 | 14 + 3i | 13 + 6i | 12 + 8i | 10 + 11i | 14.7409 | 0.048529 |
| 15 | 15 + 3i | 13 + 6i | 12 + 8i | 10 + 11i | 14.7409 | 0.048529 |
| 15 | 14 + 3i | 13 + 6i | 12 + 8i | 11 + 10i | 14.7409 | 0.048529 |
| 15 | 15 + 3i | 13 + 6i | 12 + 8i | 11 + 10i | 14.7409 | 0.048529 |
| 15 | 15 + 3i | 14 + 6i | 13 + 9i | 10 + 11i | 15.1043 | 0.049766 |
| 15 | 15 + 3i | 14 + 6i | 13 + 9i | 11 + 10i | 15.1043 | 0.049766 |

The coefficients with common kernels are considered in the Tables, however, the number of coefficients found, increases with increased number of bits. This is done to ensure that the kernels can also be generated with the help of number of bits=5. For transforms like DFT, this can be very useful as the number of kernels can be found out for single radius barring few points. It is very efficient in determining the kernels for non-sinusoidal transform with different scaling.

4.2 Coefficient selection for Non-sinusoidal transforms:

Sinusoidal transforms like DFT require a common radius for the coefficient as their trajectory is circular. However, this is not the case in terms of non-sinusoidal transforms where the coefficients can be from different radius at different points. CCSSI also helps in getting the coefficients for non-sinusoidal transforms like HAAR, where the trajectory is not circular. The fundamental aim is to find the coefficients at different radius with minimum possible error.

The CCSSI algorithm was implemented for these cases and the results are indicated in Table 4.9, 4.10 below. The Table 4.9 and 4.10 is implemented indicating values for N= 8, 16.

For Number of points (N) = 8, b = 5 bits and max error bound of 0.05, 35 unique coefficients can be found for two different angles. R indicates the values of radius where the points, given the conditions, are determined. With eight bits the number of coefficient is taken as one, however the max error bound is reduced in the order 10^{-2} , which immensely improves the accuracy.

The analysis of the same is carried out and the Table 4.10 shows the results for N = 16. As the number of points increases, the number of unique coefficients also increases as seen in the Tables. It is observed that Coefficient combined selections and shift and add implementation approach can also be used to determine the coefficients for given error bound for non-sinusoidal transforms.

Table 4.9: MCR Remaining Kernel for N = 8

| N = 8 | | | | | | | |
|-----------------|---------|----------|----------------------------|-----------|---------------|-------------------------------------|---|
| b (bits) | Angle 1 | Angle 2 | Min_er ror at Radius | Min_Error | Max_ Error | Number of unique coefficients | R |
| 5 | 7 | 5 + 5i | 7.0355 | 0.0050506 | 0.05 | 35 | 3,4,6,7 ,8,9,10 ,11,12, 13,14, |
| | 14 | 10 + 10i | 14.0711 | 0.0050506 | | | |
| | 10 | 7 + 7i | 9.9497 | 0.0050506 | | | |
| 6 | 24 | 17 + 17i | 24.0208 | 0.0008666 | 0.005 | 4 | 17,24, 27,31 |
| | 17 | 12 + 12i | 16.9853 | 0.0008666 | | | |

| | | | | | | | |
|----------|----|------------|---------|-----------|--------|----|--|
| 7 | 41 | $29 + 29i$ | 41.0061 | 0.0001487 | 0.0009 | 21 | 17,24, 34,41, 48,51, 58,65, 68,72, 85,89, 92,96, 102,10 6,109, |
| | 58 | $41 + 41i$ | 57.9914 | 0.0001487 | | | |
| 8 | 99 | $70 + 70i$ | 98.9975 | 2.55e-05 | 0.0001 | 1 | 99 |

Table 4.10: MCR Remaining Kernel for N = 16

| N = 16 | | | | | | | | |
|-----------------|---------|------------|------------|---------------------|-----------|-----------|-------------------------------|--|
| b (bits) | Angle 1 | Angle 2 | Angle 3 | Min_error at Radius | Min_Error | Max_Error | Number of unique coefficients | R |
| 5 | 13 | $12 + 5i$ | $9 + 9i$ | 12.8653 | 0.0106 | 0.05 | 123 | 7,8,9,10, 11,12, 13,14, 15.... |
| | | | | | | | | |
| 6 | 24 | $22 + 9i$ | $17 + 17i$ | 23.8856 | 0.0065 | 0.02 | 5 | 13,24, 26,28, 31 |
| | | | | | | | | |
| 7 | 105 | $97 + 40i$ | $74 + 74i$ | 104.7833 | 0.0020 | 0.007 | 318 | 24,41, 42, 44,48, 58,..... . |

| | | | | | | | | |
|---|-----|-----------|----------|----------|--------|-------|----|---------------------------|
| | | | | | | | | |
| 8 | 120 | 111 + 46i | 85 + 85i | 120.0784 | 0.0010 | 0.004 | 38 | 65,67, 79,81, 83... |
| | | | | | | | | |

4.3 Coefficient selection based on the number of adders

This section provides design of the rotators based on the steps indicated above. Also, this section provides an overview on how number of adders are calculated. Furthermore, the analysis work and simulations are carried out in MATLAB. First part will be calculation of number of adders for the kernel based on given method in (Garrido, Qureshi, and Gustafsson).

First calculate number of adders for single constant rotation which is describe below, If the Rotation coefficient is give as $P = C + j S$ the total number of adders for P is given as

$$AR(P) = 2 \cdot AM(C, S) + 2 \quad (4.1)$$

Where $AM (C, S)$ is number of adders used for multiplying C and S with constant and it is calculated based on CSD (Aksoy et al.) approach. Similarly if P is just real number or imaginary number then the total number of adder for P is given as

$$AR(P) = 2 \cdot AM(C) \quad (4.2)$$

Or

$$AR(P) = 2 \cdot AM(S) \quad (4.3)$$

CSD approach is used to compute $AM(C)$ and $AM(S)$. These cases are considered in order to make a better use of adders and design simple rotators. SCR (Garrido, Qureshi, and Gustafsson) technique is used for calculation of adders, Using it calculate number of adders for every coefficient in the kernel for each angle and then maximum number of adders are evaluated. Number of adders for kernel can be represented as

$$AK = \max\{AR(P_i)\} \quad (4.4)$$

Table 4.11: Remaining Kernel for four Adders

| Adder = 4 | | | |
|-----------|------------|------------|---------------|
| N Point | Parameters | | |
| N=8 | b | Min_err | Max_err Bound |
| | 4 | 0.0050506 | 0.08 |
| | 5 | 0.0050506 | 0.05 |
| | 6 | 0.00086655 | 0.02 |
| | 7 | 0.00086655 | 0.01 |
| | 8 | 0.00041386 | 0.005 |
| | 9 | 0.00041386 | 0.0025 |
| | 10 | 0.00041386 | 0.001 |
| | 11 | 5.34E-05 | 0.0005 |
| | 12 | | 0.0003 |
| | b | Min_err | Max_err Bound |
| N=16 | 4 | 0.054689 | 0.08 |
| | 5 | 0.042992 | 0.05 |
| | 6 | 0.015633 | 0.02 |
| | 7 | Nan | 0.01 |
| | 8 | Nan | 0.005 |
| | 9 | Nan | 0.0025 |
| | 10 | Nan | 0.001 |
| | 11 | Nan | 0.0005 |
| | 12 | Nan | 0.0003 |
| | | b | Min_err |
| N=32 | 4 | Nan | 0.08 |
| | 5 | 0.046525 | 0.05 |
| | 6 | Nan | 0.02 |
| | 7 | Nan | 0.01 |

| | | | |
|------|----------|----------------|----------------------|
| | 8 | Nan | 0.005 |
| | 9 | Nan | 0.0025 |
| | 10 | Nan | 0.001 |
| | 11 | Nan | 0.0005 |
| | 12 | Nan | 0.0003 |
| | b | Min_err | Max_err Bound |
| N=64 | 4 | nan | 0.08 |
| | 5 | nan | 0.05 |
| | 6 | nan | 0.02 |
| | 7 | nan | 0.01 |
| | 8 | nan | 0.005 |
| | 9 | nan | 0.0025 |
| | 10 | nan | 0.001 |
| | 11 | nan | 0.0005 |
| | 12 | nan | 0.0003 |

Table 4.12: Remaining Kernel for Six Adders

| Adder = 6 | | | |
|-----------|------------|----------------|----------------------|
| N Point | Parameters | | |
| | b | Min_err | Max_err Bound |
| N=8 | 4 | Nan | 0.08 |
| | 5 | 0.018207 | 0.05 |
| | 6 | 0.0018144 | 0.02 |
| | 7 | 0.00014868 | 0.01 |
| | 8 | 2.55E-05 | 0.005 |
| | 9 | 2.55E-05 | 0.0025 |
| | 10 | 4.38E-06 | 0.001 |
| | 11 | 7.51E-07 | 0.0005 |
| | 12 | | 0.0003 |
| | b | Min_err | Max_err Bound |
| N=16 | 4 | 0.043011 | 0.08 |
| | 5 | 0.010679 | 0.05 |
| | 6 | 0.010679 | 0.02 |
| | 7 | 0.0061425 | 0.01 |
| | 8 | 0.0039763 | 0.005 |
| | 9 | Nan | 0.0025 |
| | 10 | Nan | 0.001 |
| | 11 | Nan | 0.0005 |
| | 12 | | 0.0003 |
| | b | Min_err | Max_err Bound |
| N=32 | 4 | 0.05735 | 0.08 |
| | 5 | 0.029019 | 0.05 |

| | | | |
|------|----------|----------------|----------------------|
| | 6 | 0.01958 | 0.02 |
| | 7 | Nan | 0.01 |
| | 8 | Nan | 0.005 |
| | 9 | Nan | 0.0025 |
| | 10 | Nan | 0.001 |
| | 11 | Nan | 0.0005 |
| | 12 | | 0.0003 |
| | | | |
| | b | Min_err | Max_err Bound |
| N=64 | 4 | 0.071201 | 0.08 |
| | 5 | 0.037401 | 0.05 |
| | 6 | Nan | 0.02 |
| | 7 | Nan | 0.01 |
| | 8 | Nan | 0.005 |
| | 9 | Nan | 0.0025 |
| | 10 | Nan | 0.001 |
| | 11 | Nan | 0.0005 |
| | 12 | | 0.0003 |
| | | | |

Table 4.13: Remaining Kernel for Eight Adders

| Adder = 8 | | | |
|------------------|-------------------|----------------|----------------------|
| N Point | Parameters | | |
| N=8 | b | Min_err | Max_err Bound |
| | 4 | Nan | 0.08 |
| | 5 | 0.048849 | 0.05 |
| | 6 | Nan | 0.02 |
| | 7 | 0.0015501 | 0.01 |
| | 8 | 0.00014868 | 0.005 |
| | 9 | 4.38E-06 | 0.0025 |
| | 10 | 4.38E-06 | 0.001 |
| | 11 | 7.51E-07 | 0.0005 |
| 12 | | 0.0003 | |
| | | | |
| | b | Min_err | Max_err Bound |
| N=16 | 4 | Nan | 0.08 |
| | 5 | 0.037055 | 0.05 |
| | 6 | 0.0065304 | 0.02 |
| | 7 | 0.0043808 | 0.01 |
| | 8 | 0.0018509 | 0.005 |
| | 9 | 0.00050129 | 0.0025 |
| | 10 | 0.00049558 | 0.001 |
| | 11 | 0.00026925 | 0.0005 |
| | 12 | | 0.0003 |
| | | | |
| | b | Min_err | Max_err Bound |

| | | | |
|------|----------|----------------|----------------------|
| N=32 | 4 | Nan | 0.08 |
| | 5 | 0.029014 | 0.05 |
| | 6 | 0.0116 | 0.02 |
| | 7 | 0.0086775 | 0.01 |
| | 8 | Nan | 0.005 |
| | 9 | Nan | 0.0025 |
| | 10 | Nan | 0.001 |
| | 11 | Nan | 0.0005 |
| | 12 | | 0.0003 |
| | b | Min_err | Max_err Bound |
| N=64 | 4 | Nan | 0.08 |
| | 5 | 0.033444 | 0.05 |
| | 6 | 0.016457 | 0.02 |
| | 7 | Nan | 0.01 |
| | 8 | Nan | 0.005 |
| | 9 | Nan | 0.0025 |
| | 10 | Nan | 0.001 |
| | 11 | Nan | 0.0005 |
| | 12 | | 0.0003 |

Table 4.14: Remaining Kernel for Ten Adders

| Adder = 10 | | | |
|------------|------------|----------------|----------------------|
| N Point | Parameters | | |
| | b | Min_err | Max_err Bound |
| N=8 | 4 | Nan | 0.08 |
| | 5 | Nan | 0.05 |
| | 6 | Nan | 0.02 |
| | 7 | Nan | 0.01 |
| | 8 | Nan | 0.005 |
| | 9 | 0.00028495 | 0.0025 |
| | 10 | 9.16E-06 | 0.001 |
| | 11 | 4.38E-06 | 0.0005 |
| | 12 | | 0.0003 |
| | b | Min_err | Max_err Bound |
| N=16 | 4 | Nan | 0.08 |
| | 5 | Nan | 0.05 |
| | 6 | 0.0075697 | 0.02 |
| | 7 | 0.005827 | 0.01 |
| | 8 | 0.0010805 | 0.005 |
| | 9 | 0.00085848 | 0.0025 |
| | 10 | 0.0003629 | 0.001 |
| | 11 | 0.00011062 | 0.0005 |
| | 12 | | 0.0003 |

| | b | Min_err | Max_err Bound |
|------|----|-----------|---------------|
| N=32 | 4 | Nan | 0.08 |
| | 5 | Nan | 0.05 |
| | 6 | 0.014816 | 0.02 |
| | 7 | 0.0058553 | 0.01 |
| | 8 | 0.0032401 | 0.005 |
| | 9 | 0.0017499 | 0.0025 |
| | 10 | Nan | 0.001 |
| | 11 | Nan | 0.0005 |
| | 12 | | 0.0003 |
| | b | Min_err | Max_err Bound |
| N=64 | 4 | Nan | 0.08 |
| | 5 | Nan | 0.05 |
| | 6 | 0.017113 | 0.02 |
| | 7 | 0.0077183 | 0.01 |
| | 8 | 0.003861 | 0.005 |
| | 9 | Nan | 0.0025 |
| | 10 | Nan | 0.001 |
| | 11 | Nan | 0.0005 |
| | 12 | | 0.0003 |

Tables 4.11 to 4.14 shows the variation of number of adders from 4 to 10 and filtering the kernels from the steps found in the above sections. The Tables indicates the optimized kernels found for the given number of adders with the minimum and maximum error for N number of points. The maximum error was varied from 0.08 to 0.003 and kernels found with minimum error are indicated, based on this, the number of adders are determined. Nan indicates undetermined values. Table 4.15 shows the summary of the found kernel with other parameters. An example is considered to determine the tunable architecture.

An example of MCR with three angles to obtain the optimized coefficient for fixed scaling is taken here. The angles taken care are $\alpha_1 = 0^\circ$ $\alpha_2 = 22.5^\circ$ and $\alpha_3 = 45^\circ$. Given these angles as an input, the number of coefficients are reduced based on adders and scaling factors to design efficient rotators. The maximum allowable error and number of adders are considered as 0.05 and 6 respectively.

After performing the steps provided in the proposed approach the remaining coefficient are as shown in the Table 4.15. The coefficient can be selected based on the requirement. The kernel with the minimum rotation error i.e. 1.068×10^{-2} is at approximate radius 13 and also satisfies the

requirement of number of adders. However, coefficients which can be used with lesser number of adders than 6 are also available, but there always exists a trade-off between the rotation error and number of adders.

Table 4.15: MCR Remaining Kernel based on the proposed approach

| $\alpha_1 = 0^\circ$ | $\alpha_2 = 22.5^\circ$ | $\alpha_3 = 45^\circ$ | R | Err | Adder |
|----------------------|-------------------------|-----------------------|---------|------------------------|-------|
| 7 | 7+3i | 5+5i | 7.31 | 4.301×10^{-2} | 6 |
| 10 | 10+4i | 7+7i | 10.344 | 4.301×10^{-2} | 4 |
| 11 | 10+4i | 8+8i | 10.8474 | 4.299×10^{-2} | 4 |
| 13 | 12+5i | 9+9i | 12.8653 | 1.068×10^{-2} | 6 |

Table 4.16: Comparison of Multiplierless Rotator of various approaches with the Proposed approach.

| APPROACH | DESIGN SPACE | | | OPTIMIZATION PROBLEM | |
|--|-----------------------------------|----------------------------|-------------------|----------------------|-------------------|
| | Coefficient Selection | Shift-and-Add Optimization | Design Space Size | Scaling | Angle Set |
| General Rotators: angles not known a priori | | | | | |
| Conventional CORDIC (Volder) | Small | High (Direct) | Small | Uniform | General Rotations |
| Complex Multiplier (Chang and Parhi) | Small | Low | Small | Unity | General Rotations |
| Constant Rotators: angles known a priori | | | | | |
| Lifting Schemes (Chan and Yiu) | Small | Medium (CSD) | Small | Unity | SCR |
| EEAS CORDIC (Meher et al.) | Medium | High (Direct) | Small | Unity/Arbitrary | SCR |
| MRS-CORDIC (Lin and Wu) | Large | Medium (CSD) | Medium | Unity | SCR |
| CORDIC for Fixed Angles (Meher and Park) | Medium | High (Direct) | Medium | Unity/Arbitrary | SCR |
| Trigonometric Identities (CSD) (Voronenko and Püschel) | Medium | Medium (CSD) | Medium | Unity | MCR for FFT |
| Trigonometric Identities (SCM) (Thong and Nicolici) | Medium | High (MCM) | Medium | Unity | MCR for FFT |
| Base-3 Rotator (Chang and Parhi) | Medium | High (SCM, MCM) | Medium | Uniform | MCR for FFT |
| Rotator Using CSD (Gustafsson) | Small | Medium (CSD) | Small | Unity | MCR |
| Rotator Using MCM (Voronenko and Püschel) | Small | High (MCM) | Small | Unity | MCR |
| CCSSI (Garrido, Qureshi, and Gustafsson) | Maximum (Complete Freedom) | High (SCM, MCM) | Large | Any | SCR and MCR |
| Extended CCSSI (Proposed) | Maximum (Complete Freedom) | Medium (CSD) | Large | Any | Any |

Table 4.16 provides the comparison for the existing algorithms with the design space, angle set and Scaling. While CORDIC based techniques are wide but are only limited to Unity scaling and single constant rotation. The CCSSI based technique allows the flexibility in terms of parameters like Scaling, Single constant rotator and multiple constant rotator as well and also the degree of freedom allowed in space is an important advantage. This makes the Algorithm feasible to be considered for multiple radius and hence, multiple trajectory transforms as well.

4.4 Realization of Proposed Architecture

Realization of these coefficients in the form of shifters and adders is shown in the Figure-4.3 and combined realization of the kernel is shown in Figure-4.4. For combined realization, multiplexers are used to will determine the coefficient. In Figure-4.3 realization of the rotator for $\alpha = 22.5^\circ$ using $7 + 3i$ is shown, Combined realization for the First kernel is shown in Figure-4.4.

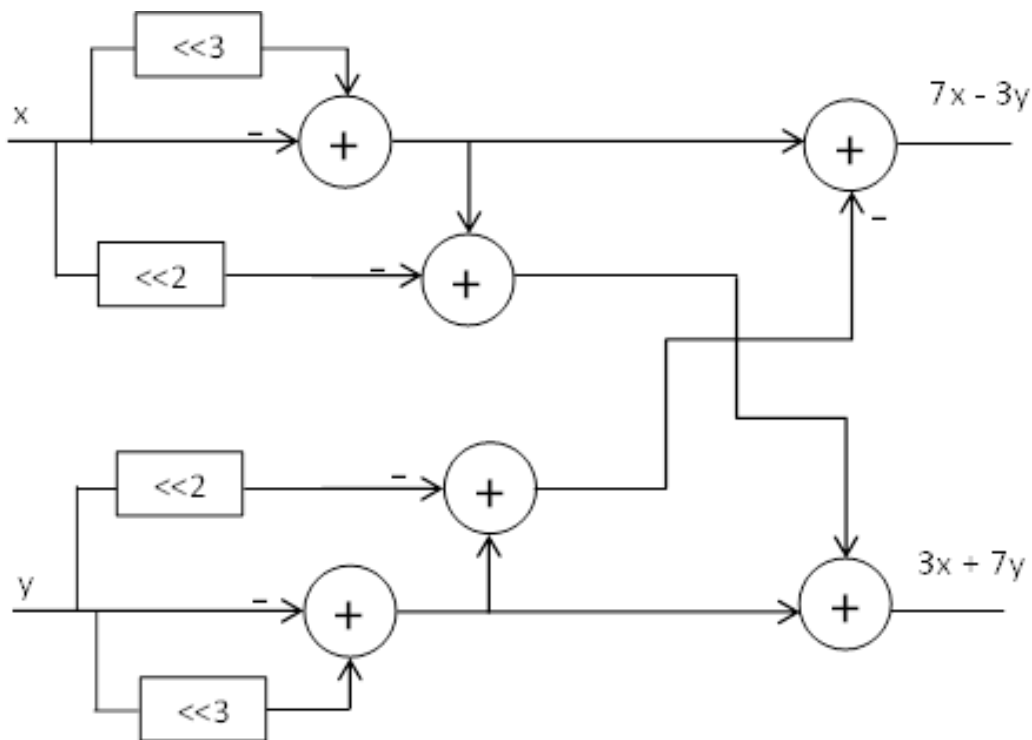


Figure 4.1: Realization of rotator for proposed framework ($\alpha = 22.5$ using $7 + 3i$)

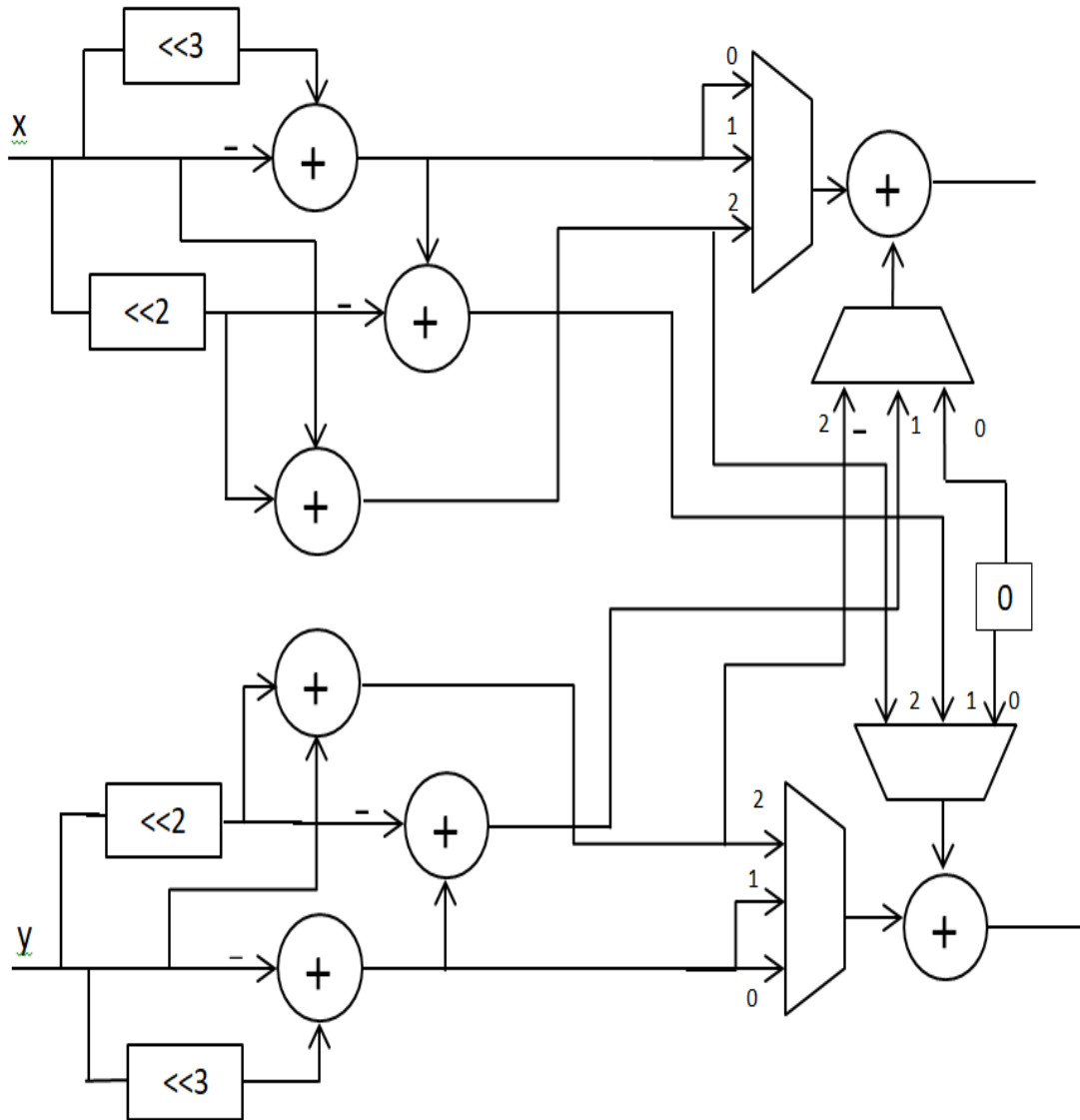


Figure 2.4: Realization of combined rotator design for kernel-1 from Table-4.15

Shift and add implementation of rotator is given in the above Figures.

The next section covers the results and conclusions obtained from this approach for various experiments carried out during this project.

4.5 Results and Discussions

This section provides the results of the proposed approach which is obtained based on several observations. The experiments carried out are for the N point FFT where the values of N are considered from 8 to 64. Since the solution for $N = 4$ is trivial it is not taken into consideration. Also, word length (bits) is considered for the design space starting from 4 to 12 for better accuracy. In an N-point FFT, twiddle factor needs to be calculated to transform signal from one form to another. The twiddle factors are specific set of angles which are generated by dividing circumference of a circle in a complex plane into K equal parts. Eight angles can be found for 8 point by dividing $[0, 2\pi]$ into 8 equal parts at angles $[0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi, \frac{5\pi}{4}, \frac{3\pi}{2}, \frac{7\pi}{4}]$. Here, the only angle which belongs to $[0, \frac{\pi}{4}]$ is considered, since the points which belongs at an angle greater than $\frac{\pi}{4}$ can be generated by interchanging the values and changing signs of the corresponding points.

Based on these criterion, results are presented in the range $[0, \frac{\pi}{4}]$. For calculation of number of adders required for the coefficient, Canonic Signed Digit approach is considered.

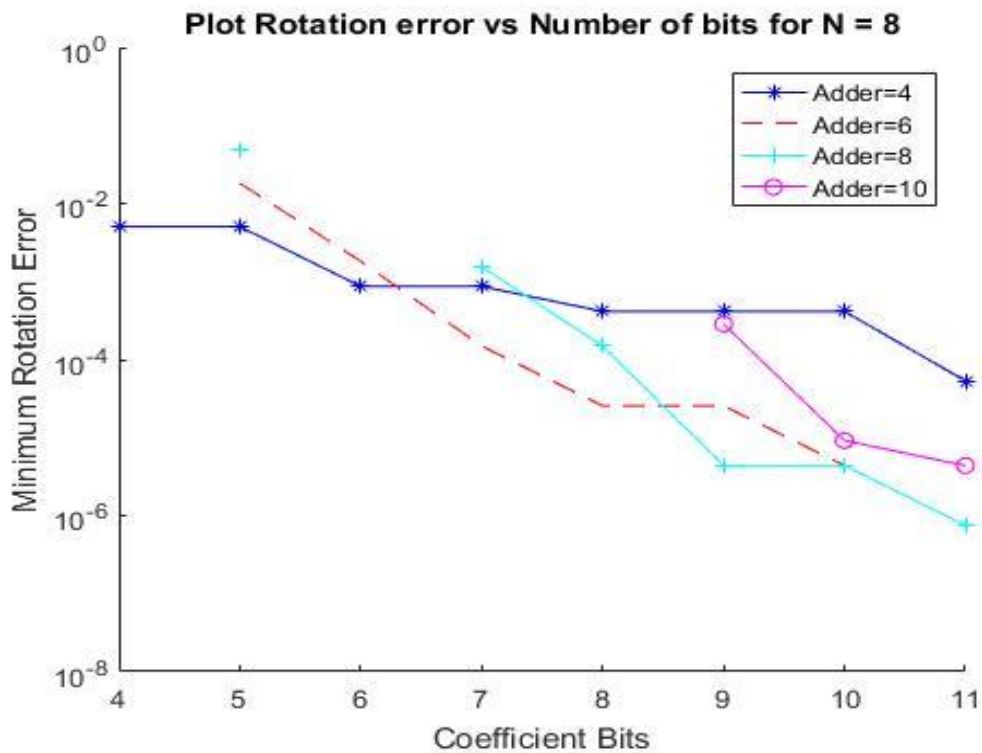


Figure 4.3: Graph representing trade-off between minimum Rotation Error vs. number of bits b.

First, the adder requirement is calculated using Canonic signed Digit for each coefficient in the kernel of m angles, then it is checked for the maximum number of adders required in that kernel.

Figures 4.5 to 4.8 presents the proposed results for the W_8 , W_{16} , W_{32} , and W_{64} . The results in the form of graph shows the trade-off between minimum rotation error and word length (b-bits). Number of minimum required adders and maximum allowed rotation error are considered to obtain the coefficient for the set of kernels. The optimum coefficient values are chosen based on the minimum rotation error and the number of adders.

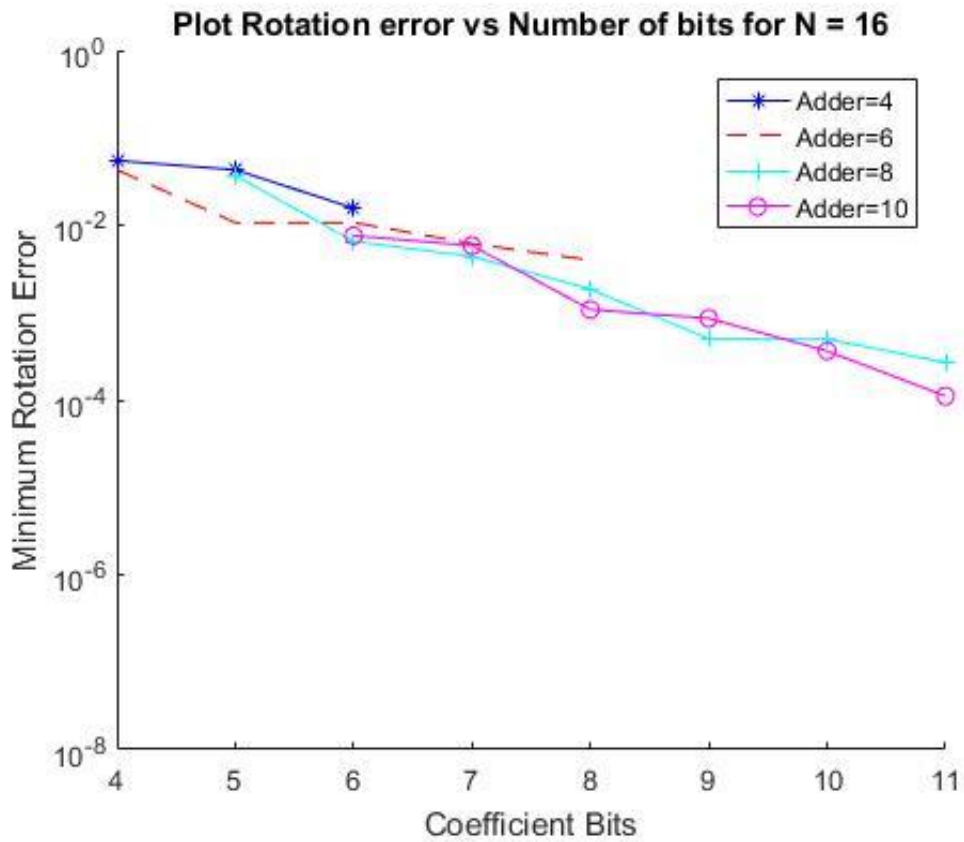


Figure 4.4: Graph representing trade-off between minimum Rotation Error vs. number of bits b.

Figures 4.5 and 4.6 depicts that the coefficients with minimum error around 10^{-6} to 10^{-4} for higher bits can be found with less number of adders are determined. When the number of points (N) is increased, the requirement for number of adders for multiplication also increases and less number of coefficients are obtained.

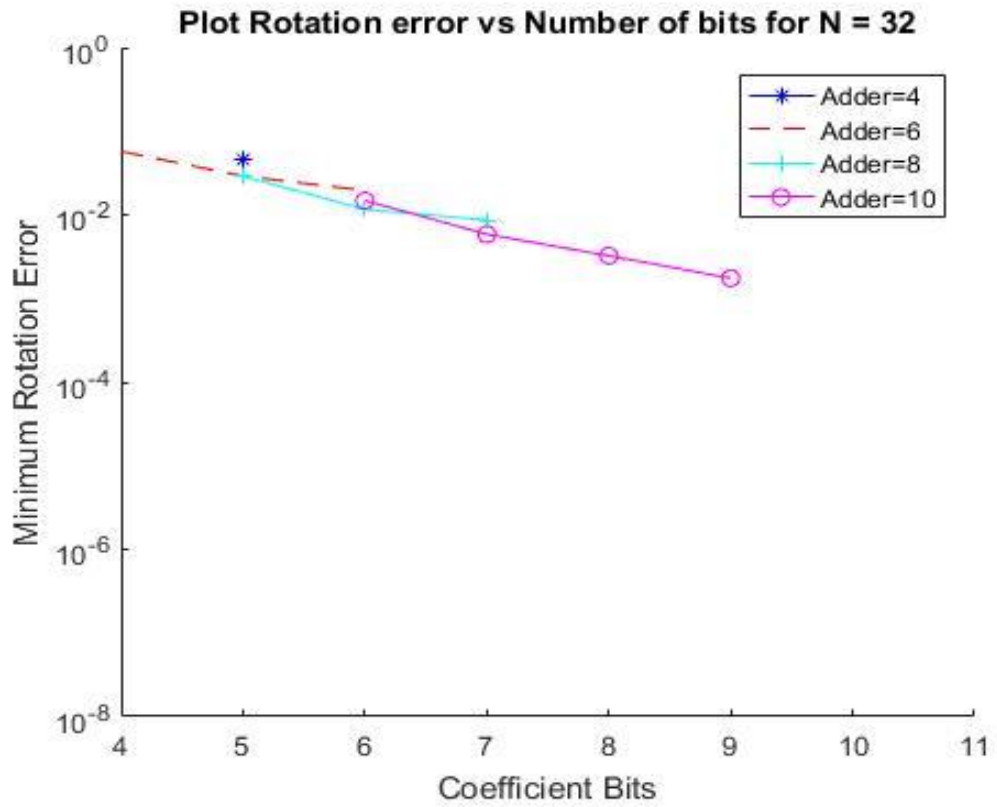


Figure 4.5: Graph representing trade-off between minimum Rotation Error vs. number of bits b .

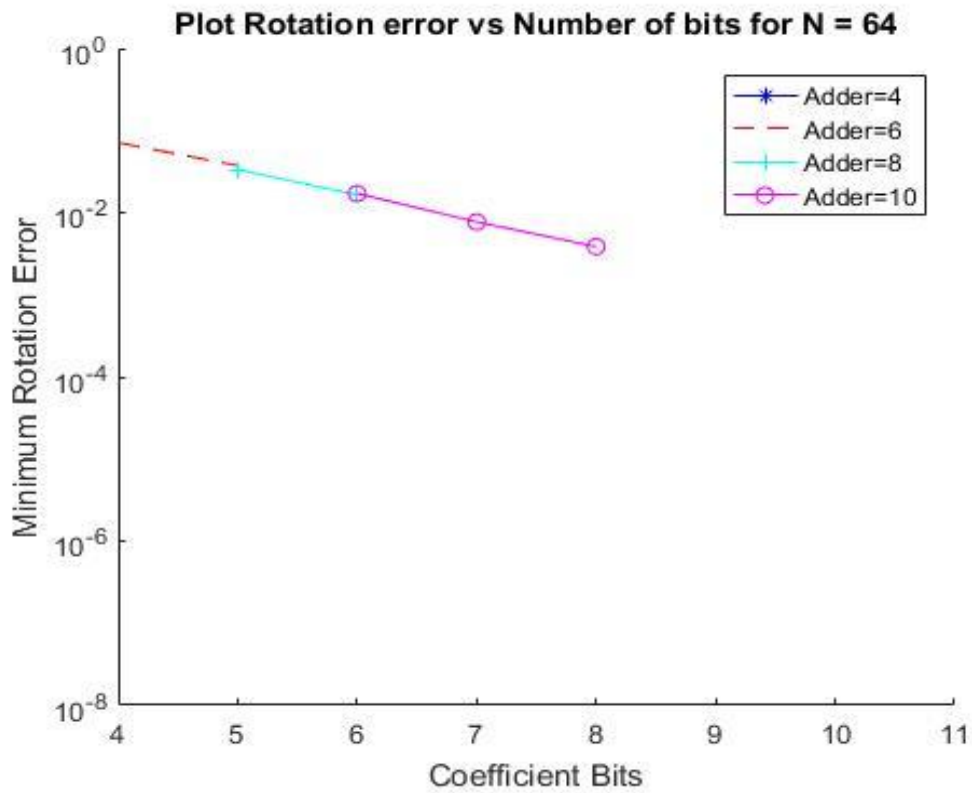


Figure 4.6: Graph representing trade-off between minimum Rotation Error vs. number of bits b .

Further, the number of unique coefficients obtained were calculated for different number of bits and various N points. This is summarized in the Table-4.18 which shows that with increase in number of bit for particular error bound and N point almost linear increase in number of unique coefficient is achieved.

Table 4.17: Number of unique coefficients obtained

| N Point | Parameters | | | |
|----------------|-------------------|----------------|-------------------------------------|----------------------|
| | b | Min_err | Total number of unique Coeff | Max_err Bound |
| N=8 | 5 | 0.0050506 | 19 | 0.05 |
| | 6 | 0.00086655 | 27 | 0.02 |
| | 7 | 0.00014868 | 57 | 0.01 |
| | 8 | 2.55E-05 | 101 | 0.005 |
| | 9 | 4.38E-06 | 231 | 0.0025 |
| | 10 | 4.38E-06 | 374 | 0.001 |
| | 11 | 7.51E-07 | 741 | 0.0005 |
| | 12 | 1.29E-07 | 1777 | 0.0003 |
| <hr/> | | | | |
| N=16 | b | Min_err | Total number of unique Coeff | Max_err Bound |
| | 5 | 0.010679 | 18 | 0.05 |
| | 6 | 0.0065304 | 15 | 0.02 |
| | 7 | 0.0043808 | 34 | 0.01 |
| | 8 | 0.0010805 | 69 | 0.005 |
| | 9 | 0.00050129 | 143 | 0.0025 |
| | 10 | 0.00021858 | 146 | 0.001 |
| | 11 | 0.00011062 | 287 | 0.0005 |
| 12 | 1.63E-05 | 1044 | 0.0003 | |
| <hr/> | | | | |
| N=32 | b | Min_err | Total number of unique Coeff | Max_err Bound |
| | 5 | 0.029014 | 34 | 0.05 |
| | 6 | 0.0116 | 15 | 0.02 |
| | 7 | 0.005413 | 26 | 0.01 |
| | 8 | 0.0029774 | 109 | 0.005 |
| | 9 | 0.0010565 | 143 | 0.0025 |
| | 10 | 0.0004414 | 54 | 0.001 |
| | 11 | 0.00028223 | 114 | 0.0005 |
| 12 | 0.00010229 | 841 | 0.0003 | |
| <hr/> | | | | |
| N=64 | b | Min_err | Total number of unique Coeff | Max_err Bound |
| | 5 | 0.033444 | 187 | 0.05 |
| | 6 | 0.016457 | 20 | 0.02 |
| | 7 | 0.0076126 | 37 | 0.01 |
| | 8 | 0.0033798 | 180 | 0.005 |
| | 9 | 0.0016573 | 255 | 0.0025 |
| | 10 | 0.00076875 | 14 | 0.001 |
| | 11 | 0.00038855 | 38 | 0.0005 |
| 12 | 0.00017898 | 1028 | 0.0003 | |

Summary

In this chapter the Coefficient combined selection and shift and add implementation approach was analyzed and further extended to different criteria. Firstly, analysis was done to find the number of coefficients that can be determined for a common radius given the number of bits. It is observed that as the number of bits and number of points are increased, it is possible to generate maximum number of coefficients for a common radius except few. This algorithm hence is suitable for determining the point for sinusoidal transform for large number of points. For $N = 2$ or 4 , other techniques can be used for determining the number of points. Also, the algorithm is tested for number of adders as a parameter. It is observed that when number of DFT points increases minimum rotation error for the kernel also increases because of increase in number of angles, overlapping coefficient increases and also the maximum rotation error changes for each kernel which leads to increase in minimum rotation error.

The algorithm is suitable for finding the kernels for the non-sinusoidal transforms as well as it does not set any restrictions on the trajectory. The overall multiplier less architecture for the algorithm is also proposed.

Chapter 5

Conclusions and Future Scope

5.1 Conclusions

In this thesis various approaches for multiplierless algorithms are proposed and investigated.

CCSSI based Approach:

The proposed approach uses the combined coefficient selection and shift-and-add implementation for the design of low complexity multiplierless constant rotators. This approach is further extended finding total number of unique coefficients which are obtained with the help of CCSSI method for different number of bits and N points. Experimental results based on this approach are provided. Significant low complexity and improvement are observed based on simulation with respect to different existing approach. The overall framework of the proposed approach is also presented. It is observed that the proposed framework presents tunable multiplierless architecture which is used to determine the coefficients for both sinusoidal as well as non-sinusoidal transforms. The tunable architecture can be used based on the applications so as to implement based on the number of adders, number of bits and number of points. With the proposed algorithm any coefficient can be generated based on the provided number of adders as an input. For instance, an angle of 22.5° is generated with the help of only 6 adders and 4 shifters. Experimental results shows that the architecture can be used for various applications as it is flexible enough depending on the parameters chosen.

MSR based Approach:

The proposed algorithm incorporates the technique of weighted amplifying factors. The proposed approach can be implemented with, both Generalized and Normalized schemes. The algorithm provides better SQNR performance with no added hardware complexity. The results are compared with the existing MSR approach and it is found that the proposed generalized MSR CORDIC shows 6.6 % improvement for a given value of $I + J = 3$ and $N=2$ indicated in Figure 3.3 in the SQNR compared to the existing MSR CORDIC algorithm. Mixed scaling and rotation algorithm is extended with weighted amplifying factors. The algorithm can be used to implement multiplierless architecture for sinusoidal transforms like Discrete Fourier transform.

References

- Aggarwal, Supriya, Pramod K. Meher, and Kavita Khare. "Concept, design, and implementation of reconfigurable CORDIC." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 24.4 (2015): 1588-1592.
- Ahmed, Nasir, and Kamisetty Ramamohan Rao. "Fast fourier transform." *Orthogonal Transforms for Digital Signal Processing*. Springer, Berlin, Heidelberg, 1975. 54-84.
- Ahmed, Nasir, and Kamisetty Ramamohan Rao. "Walsh-hadamard transform." *Orthogonal Transforms for Digital Signal Processing*. Springer, Berlin, Heidelberg, 1975. 99-152.
- Aksoy, Levent, Ece Olcay Güneş, and Paulo Flores. "Search algorithms for the multiple constant multiplications problem: Exact and approximate." *Microprocessors and Microsystems* 34.5 (2010): 151-162.
- Andersson, Rikard. "FFT hardware architectures with reduced twiddle factor sets." (2013).
- Andraka, Ray. "A survey of CORDIC algorithms for FPGA based computers." *Proceedings of the 1998 ACM/SIGDA sixth international symposium on Field programmable gate arrays*. 1998.
- Arguello, F., et al. "Parallel architecture for fast transforms with trigonometric kernel." *IEEE Transactions on Parallel and Distributed Systems* 5.10 (1994): 1091-1099.
- Bluestein, Leo. "A linear filtering approach to the computation of discrete Fourier transform." *IEEE Transactions on Audio and Electroacoustics* 18.4 (1970): 451-455.
- Chang, Yun-Nan, and Keshab K. Parhi. "High-performance digit-serial complex multiplier." *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* 47.6 (2000): 570-572.

- Chan, S. C., and P. M. Yiu. "An efficient multiplierless approximation of the fast Fourier transform using sum-of-powers-of-two (SOPOT) coefficients." *IEEE signal processing letters* 9.10 (2002): 322-325.
- Chen, Chuen-yau, and Cheng-yuan Lin. "High-resolution architecture for CORDIC algorithm realization." *2006 International Conference on Communications, Circuits and Systems*. Vol. 1. IEEE, 2006.
- De Caro, Davide, et al. "Accurate fixed-point logarithmic converter." *IEEE Transactions on Circuits and Systems II: Express Briefs* 61.7 (2014): 526-530.
- De Caro, Davide, Nicola Petra, and Antonio GM Strollo. "Efficient logarithmic converters for digital signal processing applications." *IEEE Transactions on Circuits and Systems II: Express Briefs* 58.10 (2011): 667-671.
- Dempster, Andrew G., and Malcolm D. Macleod. "Constant integer multiplication using minimum adders." *IEE Proceedings-Circuits, Devices and Systems* 141.5 (1994): 407-413.
- Dempster, Andrew G., and Malcolm D. Macleod. "Multiplication by two integers using the minimum number of adders." *2005 IEEE International Symposium on Circuits and Systems*. IEEE, 2005.
- Despain, Alvin M. "Fourier transform computers using CORDIC iterations." *IEEE Transactions on Computers* 100.10 (1974): 993-1001.
- Despain, Alvin M. "Very fast Fourier transform algorithms hardware for implementation." *IEEE Transactions on Computers* 5 (1979): 333-341.
- Garrido, Mario, Oscar Gustafsson, and Jesús Grajal. "Accurate rotations based on coefficient scaling." *IEEE Transactions on Circuits and Systems II: Express Briefs* 58.10 (2011): 662-666.
- Garrido, Mario, et al. "CORDIC II: a new improved CORDIC algorithm." *IEEE Transactions on Circuits and Systems II: Express Briefs* 63.2 (2015): 186-190.

- Gálvez, Mario Garrido. "Efficient hardware architectures for the computation of the FFT and other related signal processing algorithms in real time." Diss. Universidad Politécnica de Madrid, 2009.
- Garrido, Mario, and Jesus Grajal. "Efficient memoryless CORDIC for FFT computation." *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*. Vol. 2. IEEE, 2007.
- Garrido, Mario, et al. "Hardware architectures for the fast Fourier transform." *Handbook of Signal Processing Systems*. Springer, Cham, 2019. 613-647.
- Garrido, Mario, et al. "Multiplierless unity-gain SDF FFTs." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 24.9 (2016): 3003-3007.
- Goel, Garvit, Gaurav Mittal, and Abhijit R. Asati. "ROM based logic design for base-2 exponential and logarithm converter using fixed point number representation." *2016 International Conference on Inventive Computation Technologies (ICICT)*. Vol. 3. IEEE, 2016.
- Gorman, Steve F., and Jeffrey M. Wills. "Partial column FFT pipelines." *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* 42.6 (1995): 414-423.
- Gray, A., and J. Markel. "Digital lattice and ladder filter synthesis." *IEEE Transactions on audio and electroacoustics* 21.6 (1973): 491-500.
- Gustafsson, Oscar, and Fahad Qureshi. "Addition aware quantization for low complexity and high precision constant multiplication." *IEEE Signal Processing Letters* 17.2 (2009): 173-176.
- Garrido, Mario, Fahad Qureshi, and Oscar Gustafsson. "Low-complexity multiplierless constant rotators based on combined coefficient selection and shift-and-add implementation (CCSSI)." *IEEE Transactions on Circuits and Systems I: Regular Papers* 61.7 (2014): 2002-2012.
- Garrido, Mario, Jesús Grajal, and Oscar Gustafsson. "Optimum circuits for bit reversal." *IEEE Transactions on Circuits and Systems II: Express Briefs* 58.10 (2011): 657-661.

- Gorman, Steve F., and Jeffrey M. Wills. "Partial column FFT pipelines." *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* 42.6 (1995): 414-423.
- Gustafsson, Oscar, et al. "Simplified design of constant coefficient multipliers." *Circuits, Systems and Signal Processing* 25.2 (2006): 225-251.
- Gustafsson, Oscar. "A difference based adder graph heuristic for multiple constant multiplication problems." *2007 IEEE International Symposium on Circuits and Systems*. IEEE, 2007.
- Han, Wei, et al. "High-Performance Low-Power FFT Cores." *ETRI journal* 30.3 (2008): 451-460.
- He, Shousheng, and Mats Torkelson. "Design and implementation of a 1024-point pipeline FFT processor." *Proceedings of the IEEE 1998 Custom Integrated Circuits Conference (Cat. No. 98CH36143)*. IEEE, 1998.
- He, Shousheng, and Mats Torkelson. "Designing pipeline FFT processor for OFDM (de) modulation." *1998 URSI International Symposium on Signals, Systems, and Electronics. Conference Proceedings (Cat. No. 98EX167)*. IEEE, 1998.
- Heideman, Michael T., Don H. Johnson, and C. Sidney Burrus. "Gauss and the history of the fast Fourier transform." *Archive for history of exact sciences* 34.3 (1985): 265-277.
- Hoang, Van-Phuc, Xuan-Tien Do, and Cong-Kha Pham. "An efficient ASIC implementation of logarithm approximation for HDR image processing." *2013 International Conference on Advanced Technologies for Communications (ATC 2013)*. IEEE, 2013.
- Hsiao, Chen-Fong, Yuan Chen, and Chen-Yi Lee. "A generalized mixed-radix algorithm for memory-based FFT processors." *IEEE Transactions on Circuits and Systems II: Express Briefs* 57.1 (2010): 26-30.
- Hu, Yu Hen, and S. Naganathan. "An angle recoding method for CORDIC algorithm implementation." *IEEE Transactions on Computers* 42.1 (1993): 99-102.
- Hu, Xiaobo, Ronald G. Harber, and Steven C. Bass. "Expanding the range of convergence of the CORDIC algorithm." *IEEE Transactions on computers* 1 (1991): 13-21.

- Inguva, Sharath Chandra, and J. B. Seventline. "Enhanced CORDIC algorithm using an area efficient carry select adder." *2017 International Conference on Intelligent Sustainable Systems (ICISS)*. IEEE, 2017.
- Jaime, Francisco J., et al. "Enhanced scaling-free CORDIC." *IEEE Transactions on Circuits and Systems I: Regular Papers* 57.7 (2010): 1654-1662.
- James, D. "Quantization errors in the fast Fourier transform." *IEEE Transactions on Acoustics, Speech, and Signal Processing* 23.3 (1975): 277-283.
- Jo, Byung G., and Myung Hoon Sunwoo. "New continuous-flow mixed-radix (CFMR) FFT processor using novel in-place strategy." *IEEE Transactions on Circuits and Systems I: Regular Papers* 52.5 (2005): 911-919.
- Johansson, Kenny, Oscar Gustafsson, and Lars Wanhammar. "Implementation of elementary functions for logarithmic number systems." *IET Computers & Digital Techniques* 2.4 (2008): 295-304.
- Juang, Tso-Bing, Sheng-Hung Chen, and Huang-Jia Cheng. "A lower error and ROM-free logarithmic converter for digital signal processing applications." *IEEE Transactions on Circuits and Systems II: Express Briefs* 56.12 (2009): 931-935.
- Lee, Seungbeom, and Sin-Chong Park. "Modified sdf architecture for mixed dif/dit fft." *2007 IEEE International Symposium on Circuits and Systems*. IEEE, 2007.
- Li, Junwei, et al. "Study of CORDIC algorithm based on FPGA." *2016 Chinese Control and Decision Conference (CCDC)*. IEEE, 2016.
- Lin, Chih-Hsiu, and An-Yeu Wu. "Mixed-scaling-rotation CORDIC (MSR-CORDIC) algorithm and architecture for high-performance vector rotational DSP applications." *IEEE Transactions on Circuits and Systems I: Regular Papers* 52.11 (2005): 2385-2396.
- Liu, Hang, and Hanho Lee. "A high performance four-parallel 128/64-point radix-2 4 FFT/IFFT processor for MIMO-OFDM systems." *APCCAS 2008-2008 IEEE Asia Pacific Conference on Circuits and Systems*. IEEE, 2008.

- Loeffler, Christoph, Adriaan Ligtenberg, and George S. Moschytz. "Practical fast 1-D DCT algorithms with 11 multiplications." *International Conference on Acoustics, Speech, and Signal Processing*,. IEEE, 1989.
- Masram, Bharati Y., and P. T. Karule. "High speed 3D-DCT/IDCT CORDIC algorithm for DSP application." *European Journal of Advances in Engineering and Technology* 4.12 (2017): 941-950.
- Meher, Pramod K., et al. "50 years of CORDIC: Algorithms, architectures, and applications." *IEEE Transactions on Circuits and Systems I: Regular Papers* 56.9 (2009): 1893-1907.
- Meher, Pramod Kumar, and Sang Yoon Park. "CORDIC designs for fixed angle of rotation." *IEEE transactions on very large scale integration (VLSI) systems* 21.2 (2012): 217-228.
- Mishra, Ansuman, S. Sivanantham, and K. Sivasankaran. "Sine and cosine generator using CORDIC algorithm implemented in ASIC." *2015 Online International Conference on Green Engineering and Technologies (IC-GET)*. IEEE, 2015.
- Möller, Konrad, et al. "Reconfigurable constant multiplication for FPGAs." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 36.6 (2016): 927-937.
- Moroz, Leonid, Taras Mykytiv, and Martyn Herasym. "Improved scaling-free CORDIC algorithm." *2013 11th East-West Design and Test Symposium (EWDTS)*. IEEE Computer Society, 2013.
- Oh, Jung-Yeol, and Myoung-Seob Lim. "New radix-2 to the 4th power pipeline FFT processor." *IEICE transactions on electronics* 88.8 (2005): 1740-1746.
- Oppenheim, Alan V., and Clifford J. Weinstein. "Effects of finite register length in digital filtering and the fast Fourier transform." *Proceedings of the IEEE* 60.8 (1972): 957-976.
- Paul, Suganth, Nikhil Jayakumar, and Sunil P. Khatri. "A fast hardware approach for approximate, efficient logarithm and antilogarithm computations." *IEEE transactions on very large scale integration (vlsi) systems* 17.2 (2008): 269-277.

- Philipov, Ph, et al. "A parallel architecture for radix-2 fast fourier transform." *IEEE John Vincent Atanasoff 2006 International Symposium on Modern Computing (JVA'06)*. IEEE, 2006.
- Pineiro, J-A., Milos D. Ercegovac, and Javier D. Bruguera. "Algorithm and architecture for logarithm, exponential, and powering computation." *IEEE Transactions on Computers* 53.9 (2004): 1085-1096.
- Qureshi, Fahad, and Oscar Gustafsson. "Low-complexity constant multiplication based on trigonometric identities with applications to FFTs." *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* 94.11 (2011): 2361-2368.
- Qureshi, Fahad. *Optimization of rotations in FFTs*. Diss. Linköping University Electronic Press, 2012.
- Rabiner, Lawrence R., and Bernard Gold. "Theory and application of digital signal processing." *tads* (1975).
- Sai, Van-Thuan, and Van-Phuc Hoang. "An optimized implementation of logarithm hardware generator for digital signal processing." *2016 IEEE Sixth International Conference on Communications and Electronics (ICCE)*. IEEE, 2016.
- Sarode, Namrata, Rajeev Atluri, and P. K. Dakhole. "Mixed-radix and CORDIC algorithm for implementation of FFT." *2015 International Conference on Communications and Signal Processing (ICCSP)*. IEEE, 2015.
- Schulte, Michael J., and Earl E. Swartzlander. "Hardware designs for exactly rounded elementary functions." *IEEE Transactions on Computers* 43.8 (1994): 964-973.
- Shukla, Rohit, and Kailash Chandra Ray. "Low latency hybrid CORDIC algorithm." *IEEE Transactions on Computers* 63.12 (2013): 3066-3078.
- Swartzlander, Earl E., Wendell KW Young, and Saul J. Joseph. "A radix 4 delay commutator for fast Fourier transform processor implementation." *IEEE Journal of Solid-State Circuits* 19.5 (1984): 702-709.

- Tang, Aimei, et al. "CORDIC-based FFT real-time processing design and FPGA implementation." *2016 IEEE 12th International Colloquium on Signal Processing & Its Applications (CSPA)*. IEEE, 2016.
- Tang, Ping Tak Peter. *Table-lookup algorithms for elementary functions and their error analysis*. No. CONF-9106103-1. Argonne National Lab., IL (USA), 1991.
- Thong, Jason, and Nicola Nicolici. "Time-efficient single constant multiplication based on overlapping digit patterns." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 17.9 (2009): 1353-1357.
- Vaidyanathan, P. "Passive cascaded-lattice structures for low-sensitivity FIR filter design, with applications to filter banks." *IEEE transactions on circuits and systems* 33.11 (1986): 1045-1064.
- Volder, J. E. "IRE Trans. Electron. Comput." (1959): 330-334.
- Voronenko, Yevgen, and Markus Püschel. "Multiplierless multiple constant multiplication." *ACM Transactions on Algorithms (TALG)* 3.2 (2007): 11-es.
- Yang, Chia-Hsiang, Tsung-Han Yu, and Dejan Markovic. "Power and area minimization of reconfigurable FFT processors: A 3GPP-LTE example." *IEEE journal of solid-state circuits* 47.3 (2011): 757-768.
- Walther, John S. "A unified algorithm for elementary functions." *Proceedings of the May 18-20, 1971, spring joint computer conference*. 1971.
- Wanhammar, Lars. *DSP integrated circuits*. Elsevier, 1999.
- Wu, Cheng-Shing, and An-Yeu Wu. "Modified vector rotational CORDIC (MVR-CORDIC) algorithm and architecture." *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* 48.6 (2001): 548-561.
- Wu, Cheng-Shing, An-Yeu Wu, and Chih-Hsiu Lin. "A high-performance/low-latency vector rotational CORDIC architecture based on extended elementary angle set and trellis-based

searching schemes." *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing* 50.9 (2003): 589-601.

Multiplierless Tunable Architecture for Signal Processing Transforms

A Thesis Submitted to
Nirma University
In Partial Fulfilment of the Requirements for
The Degree of
Doctor of Philosophy
in
Technology & Engineering

By
Pratik Pravinkumar Trivedi
(13EXTPHDE95)



Electronics and Communication Engineering Department

Institute of Technology, Nirma University

Ahmedabad, Gujarat, India

February 2020

Nirma University
Institute of Technology

Conclusions and Future Scope

5.1 Conclusions

In this thesis various approaches for multiplierless algorithms are proposed and investigated.

CCSSI based Approach:

The proposed approach uses the combined coefficient selection and shift-and-add implementation for the design of low complexity multiplierless constant rotators. This approach is further extended finding total number of unique coefficients which are obtained with the help of CCSSI method for different number of bits and N points. Experimental results based on this approach are provided. Significant low complexity and improvement are observed based on simulation with respect to different existing approach. The overall framework of the proposed approach is also presented. It is observed that the proposed framework presents tunable multiplierless architecture which is used to determine the coefficients for both sinusoidal as well as non-sinusoidal transforms. The tunable architecture can be used based on the applications so as to implement based on the number of adders, number of bits and number of points. With the proposed algorithm any coefficient can be generated based on the provided number of adders as an input. For instance, an angle of 22.5° is generated with the help of only 6 adders and 4 shifters. Experimental results shows that the architecture can be used for various applications as it is flexible enough depending on the parameters chosen.

MSR based Approach:

The proposed algorithm incorporates the technique of weighted amplifying factors. The proposed approach can be implemented with, both Generalized and Normalized schemes. The algorithm provides better SQNR performance with no added hardware complexity. The results are

compared with the existing MSR approach and it is found that the proposed generalized MSR CORDIC shows 6.6 % improvement for a given value of $I + J = 3$ and $N=2$ indicated in Figure 3.3 in the SQNR compared to the existing MSR CORDIC algorithm. Mixed scaling and rotation algorithm is extended with weighted amplifying factors. The algorithm can be used to implement multiplierless architecture for sinusoidal transforms like Discrete Fourier transform.

5.2 Future scope

CCSSI based Approach:

The proposed algorithm can be extended by using SCM and MCM algorithms for multiplierless multiplication using shifters and adders. The coefficients at common radius for large variety of N can be incorporated for a given error bound. Machine learning algorithms can be used to find out common radius within the error bound. This will help the algorithm to generate coefficients with minimum number of bits and adders for given number of points.

Mixed scaling Rotation CORDIC based Approach:

A good future direction can be to perform a comparative analysis of the proposed scheme with the existing CORDIC based approaches and to study the possibility of analytically deriving optimal values of the parameters S , t , η and μ , thereby obviating the need for extensive parameter search. Also one can think of improving the latency required to generate the extensive search for the parameters given above. Architecture of the MSR CORDIC can be parallel so as to improve on the latency.