Design of an Optimized CORDIC for Fixed Angle of Rotation

M. S. Parvathy
PG student, VLSI & Embedded Systems, ECE Department
TKM Institute of Technology
Karuvelil P.O, Kollam, Kerala-691505, India

Anas A. S
Assistant professor, ECE Department
TKM Institute of Technology
Karuvelil P.O, Kollam, Kerala-691505, India

***Abstract*— The CORDIC algorithm is a technique for efficiently implementing the trigonometric functions and hyperbolic functions.Vector rotation through fixed and known angles has wide applications in various areas. But there is no optimized coordinate rotation digital computer (CORDIC) design for vector-rotation through specific angles. FPGA based CORDIC design for fixed angle of rotation provides optimization schemes and CORDIC circuits for fixed and known rotations with different levels of accuracy. For reducing the area and time complexities, a hardwired pre-shifting scheme in barrel-shifters of the CORDIC circuits is used. Pipelined schemes are suggested further for cascading dedicated single-rotation units and bi-rotation CORDIC units for high-throughput and reduced latency implementations. The fixed-point mean-squared-error of the CORDIC circuit is analyzed and reduced. This design offers higher throughput, less latency and less area-delay product. The fixed-angle CORDIC rotation would have wide applications in signal processing, games, animation, graphics and robotics. The coding is done in Verilog language, synthesized using Xilinx ISE 14.1 and simulated using ISim.**

***Keywords*— Coordinate rotation digital computer (CORDIC), digital signal processing (DSP) chip, VLSI**

I. Introduction

The Coordinate Rotation DIgital Computer (CORDIC) algorithm has been used for many years for efficient implementation of vector rotation operations in hardware. CORDIC or Coordinate Rotation Digital Computer is a simple and hardware-efficient algorithm for the implementation of various elementary, especially trigonometric functions. Instead of using Calculus based methods such as polynomial or rational functional approximation, it uses simple shift, add, subtract and table look-up operations to achieve this objective. By making slight adjustments to the initial conditions and the LUT values, it can be used to efficiently implement Trigonometric, Hyperbolic, Exponential functions, Coordinate Transformations etc. using the same hardware. Since it uses only shift-add arithmetic, VLSI implementation of such an algorithm is easily achievable.

The CORDIC algorithm was first proposed by Jack E Volder in 1959 [10], [11]. It is an iterative technique and consists of two modes of operation called rotation mode and vectoring mode. In either mode, the algorithm is rotation of an angle vector by a definite angle but in variable directions. This fixed rotation in variable direction is implemented through an iterative sequence of addition/subtraction followed by bit-shift operation. The final result is obtained by appropriately scaling the result obtained after successive iterations. CORDIC works by rotating the coordinate system through constant angles until the angle is reduces to zero.

CORDIC algorithm is very attractive for hardware implementation because less power dissipation, compactable & simple in design. Less power dissipation and compact because it uses only elementary shift-and-add operations to perform the vector rotation so it only needs the use of 2 shifter and 3 adder modules. Simplicity is due to elimination of multiplication with only addition, subtraction and bit-shifting operation. Either if there is an absence of a hardware multiplier (e.g. µC, µP) or there is a necessity to optimize the number of logic gates (e.g. FPGA) CORDIC is the preferred choice. However, the major disadvantage of the CORDIC algorithm are large number of iterations required for accurate results and thus the speed is low and time delay is high, power consumption is high in some architecture types ,whenever a hardware multiplier is available, e.g. in a DSP microprocessor

Rotation of vectors through a fixed and known angle has wide applications in robotics, graphics, games, and animation [3] [5]. Locomotion of robots is very often performed by successive rotations through small fixed angles and translations of the links. The translation operations are realized by simple additions of coordinate values while the new coordinates of a rotational step could be accomplished by suitable successive rotations through a small fixed angle which could be performed by a CORDIC circuit for fixed rotation. Similarly, interpolation of orientations between key-frames in computer graphics and animation could be performed by fixed CORDIC rotations. There are plenty of examples of uniform rotation starting from electrons inside an atom to the planets and satellites. A simple example of uniform rotations is the hands of an animated mechanical clock which perform one degree rotation each time. There are several cases where high-speed constant rotation is required in games, graphic, and animation. The objects with constant rotations are very often used in simulation, modelling, games, and animation. The contributions of this paper are as follows,

1. Algorithm for optimization schemes for reducing the number of micro-rotations and for reducing the complexity of barrel-shifters for fixed-angle vector-rotation.

2. Shift-add operations for corresponding scaling circuits.

3. A hardware pre-shifting scheme is suggested for reduction of barrel-shifter complexity.

4. A cascaded pipelined circuit in which more than one CORDIC module in separate stages in a cascade. The cascade

of CORDIC modules is faster and involves less area-delay complexity.

5. An efficient CORDIC circuit using a pair of micro-rotations, and named as Bi rotation CORDIC.

II. SYSTEM ARCHITECTURE

This section presents reference CORDIC circuit, optimization of Cordic circuit, hardwired pre-shifting scheme to reduce the complexity of barrel shifter in reference Cordic circuit and multi-Stage Single-Rotation Cascaded Cordic Circuit.

A. Reference Cordic Circuit

Reference CORDIC circuit for vector rotation with initial vector (X, Y) rotate through an angle θ to obtain final vector (X', Y') through successive iterations is shown below. The rotation-mode CORDIC algorithm to rotate a vector U= [U_x U_y] through an angle θ to obtain a rotated vector V= [V_x V_y] is given by

$$(U_x)_{i+1} = (U_x)_i - \sigma_i (U_y)_i\, 2^{-i} \quad (1)$$

$$(U_y)_{i+1} = (U_y)_i + \sigma_i (U_x)_i\, 2^{-i} \quad (2)$$

$$\Phi_{i+1} = \Phi_i - \sigma_i \tan^{-1}(2^{-i}) \quad (3)$$

such that when n is sufficiently large

$$\begin{bmatrix} V_x \\ V_y \\ \Phi \end{bmatrix} \longleftarrow T \begin{bmatrix} (U_x)_n \\ (U_y)_n \\ 0 \end{bmatrix} \quad (4)$$

where $\sigma_i = -1$ **if** $\Phi_i < 0$ and $\sigma_i = 1$ otherwise, and K is the scale-factor of the CORDIC algorithm, given by,

$$K = \prod_{i=0}^{n-1} (1 + 2^{-2i})^{1/2} \quad (5)$$

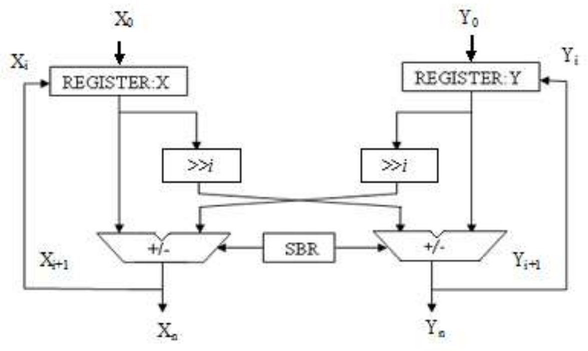

Fig. 1. Reference CORDIC circuit for fixed Rotations

A reference CORDIC circuit for fixed rotations according to (1) and (2) is shown in Fig. 1. X_0 and Y_0 are fed as set/reset input to the pair of input registers and the successive feedback values X_i and Y_i at the iteration are fed in parallel to the input registers. Note that conventionally we feed the pair of input registers with the initial values X_0 and Y_0 as the feedback values X_i and Y_i through a pair of multiplexers.

B. Optimization of Cordic Circuit

Optimization schemes are for reducing the number of micro-rotations and for reducing the complexity of barrel-shifters for fixed-angle vector-rotation. Optimization is done for both micro rotations and scaling with the help of algorithm.

The steps involved in optimizing the Elementary angle set of Micro-rotations are as follows:

- ε_Φ, maximum tolerable error between desired angle and approximated angle is given as an input.
- Optimization algorithm searches starts with single micro rotation
- If the micro-rotation that has smaller angle of deviation than ε_Φ cannot be found, the number of micro-rotations is increased by one.
- Optimization algorithm is run again.
- Based on the obtained micro-rotations, the parameters for scaling operation can be searched with the different objective function.

The steps involved in optimizing the scaling process are as follows:

- ε_k, maximum tolerable error between desired angle and approximated angle is given as an input.
- Optimization algorithm searches starts with single term of scaling.
- The number of scaling terms is increased by one until Δ_k is smaller than the given maximum deviation ε_k, where Δ_k is deviation of ratio of approximated scale factor to actual scale factor from 1.
- Optimization algorithm is run again.

For rotation of a vector through a known and fixed angle of rotation using a rotation-mode CORDIC circuit, a set of a small number of predetermined elementary angles {α_i, for $0 \leq i \leq m-1$}, where α_i = arctan ($2^{-k(i)}$) is the elementary angle to be used for the ith micro-rotation in the CORDIC algorithm and m is the minimum necessary number of micro-rotations. Meanwhile, it is well known that the rotation through any angle, $0 < \theta \leq 2\pi$ can be mapped into a positive rotation through $0 < \theta \leq \pi/4$ without any extra arithmetic operations. Hence, as a first step of optimization, we perform the rotation mapping so that the rotation angle lies in the range of $0 < \theta \leq \pi/4$. In the next step, we minimize the number of elementary angles in the set {α_i} according to the accuracy requirements. The rotation mode CORDIC algorithm of (1), (2) and (3) therefore, can be modified accordingly to have

$$\begin{bmatrix} (U_x)_{i+1} \\ (U_y)_{i+1} \end{bmatrix} = \begin{bmatrix} 1 & -\sigma_i\, 2^{-k(i)} \\ \sigma_i 2^{-k(i)} & 1 \end{bmatrix} \begin{bmatrix} (U_x)_i \\ (U_y)_i \end{bmatrix} \quad (6)$$

The set of micro rotations is optimized according to the algorithm which is described in section B. Since the elementary angles and direction of micro-rotations are predetermined for the given angle of rotation, the angle estimation data-path is not required in the CORDIC circuit for fixed and known rotations. Moreover, because only a few elementary angles are involved in this case, the corresponding control-bits could be stored in a ROM of few words.

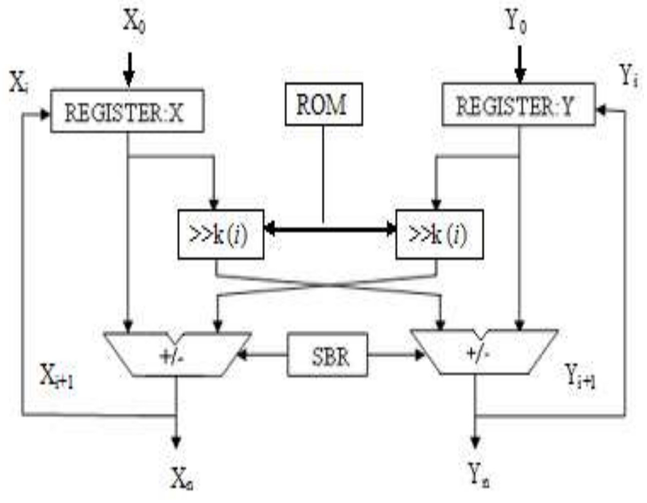

Fig. 2. Optimization of CORDIC circuit

The major contributors to the hardware-complexity in the implementation of a CORDIC circuit are the barrel-shifters and the adders.
The optimization of CORDIC circuit is shown in Fig. 2.

C. Hardwired Pre-Shifting Scheme

The barrel-shifter used in the optimization CORDIC circuit increases the complexity of the circuit. In order to avoid the complexity of barrel-shifter, a hardwired pre-shifting scheme is used. A barrel-shifter for maximum of S shifts for word-length L is implemented by [$\log_2(S+1)$] stages of demultiplexors, where each stage requires L number of 1:2 line MUXes. The hardware-complexity of barrel-shifter, therefore, increases linearly with the word-length and logarithmically with the maximum number of shifts. We can reduce the effective word-length in the MUXes of the barrel-shifters, and so also the number of stages of MUXes by simple hardwired pre-shifting.

The time involved in a barrel-shifter could also be reduced by hardwired pre-shifting, since the delay of the barrel-shifter is proportional to the number of stages of MUXes, and it also possible to reduce the number of stages by hardwired pre-shifting scheme. The hardwired pre-shifting scheme is shown in Fig. 3.

If *l* is the minimum number of shifts in the set of selected micro-rotations, we can load only the (L-*l*) more-significant bits (MSBs) of an input word from the registers to the barrel-shifters, since the *l* less significant bits (LSBs) would get truncated during shifting. The barrel-shifter, therefore, needs to implement a maximum of(s- *l*) shifts only, where s is the maximum number of shifts in the set of selected micro-rotations. The output of the barrel-shifters are loaded as the (L-*l*) LSBs to the add/subtract units, and the *l* MSBs of the corresponding operand of add/subtract unit are hardwired to 0. Therefore, the hardware-complexity of a barrel-shifter could be reduced by the hardwired pre-shifting approach.

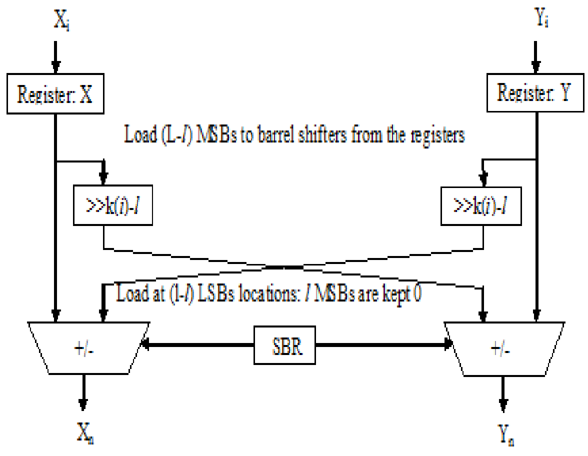
Xi
Yi
Register: X
Register: Y
Load (L-l) MSBs to barrel shifters from the registers
>>k(i)-l
>>k(i)-l
Load at (L-l) LSBs locations: l MSBs are kept 0
+/-
SBR
+/-
Xn
Yn

Fig. 3. Hardwired pre-shifting circuit

D. Multi-Stage Single-Rotation Cascaded Cordic Circuit

Multi-stage cascaded CORDIC circuit connects n number of single rotations in cascaded form. Here the initial vector values are provided to the first rotation module, whose output is provided as the input to the second rotation module and this goes on till the nth rotation module finally to obtain the final vector X_n and Y_n. The multi-stage single-rotation cascaded CORDIC circuit is shown in Fig. 4.

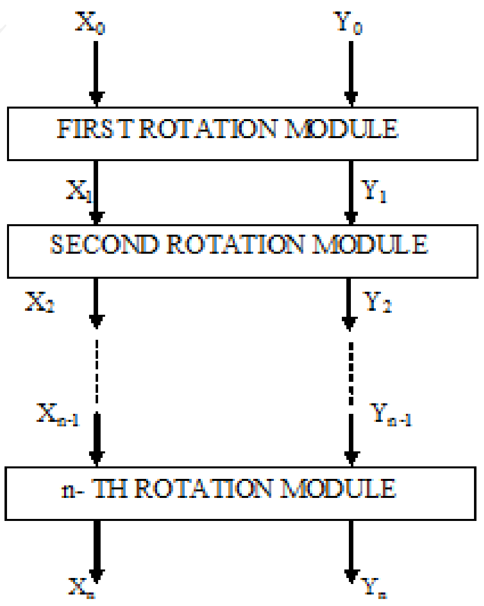

Fig. 4. Multi-stage single-rotation cascaded CORDIC circuit

Each stage of the cascaded design consists of a dedicated rotation-module that performs a specific micro-rotation. Each rotation-module consists of a pair of adders or subtractors (depending on the direction of micro-rotation which it is required to implement). Each of the adders/subtractors loads one of the pair of inputs directly and loads the other input in a pre-shifted form at (L-s(i)) LSB locations, where s(i) is the number of right-shifts required to be performed to implement the micro-rotation. The s(i) MSB locations are hardwired to be zero. The rotation-module in this case does not require input from SBR since each adder module always performs either addition or subtraction.

E. Scaling Optimization

The generalized expression for the scale-factor given by (5) can be expressed explicitly for the selected set of m_1 micro rotations as

$$K=\prod_{i=0}^{m_1-1}(1+2^{-2k(i)})^{-1/2} \qquad (7)$$

where $\mathbf{k(i)}$ **for** $0 \leq i < m_1$ is the number of shifts in the *i*th micro-rotation

An approximate scale-factor is the product of shift-add terms of form

$$K_A=\prod_{i=0}^{m_2-1}[1+\delta_i 2^{-s(i)}] \qquad (8)$$

where s(*i*) is the number of shifts performed for the *i*th iteration of scaling, $\delta_i = \pm 1$, and m_2 is maximum number of scaling iterations required for the approximation. The scaling circuit using hardwired pre-shifting scheme is shown in Fig. 5. The scaling circuit based on approximated scaling factor, Ka shown in Fig. 5. can use hardwired pre-shifting for minimizing barrel-shifter complexity.

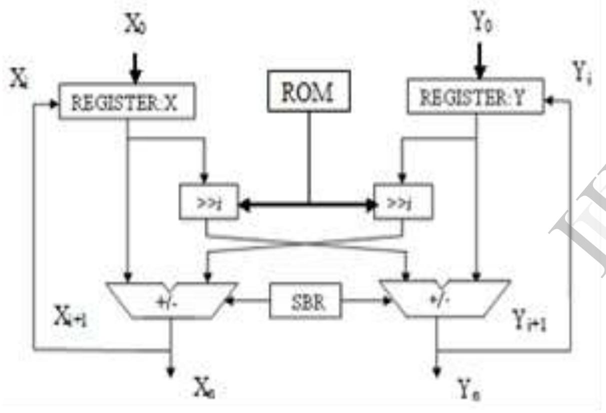

Fig. 5. Scaling circuit using hardwired pre- shifting scheme

III. Simulation Results

The design entry is modelled using Verilog in Xilinx ISE Design Suite 14.1 and the simulation of the design is performed using ISim from Xilinx ISE to validate the functionality of the design. Here CORDIC design for vector rotation through fixed and known angle is simulated.

A. Reference CORDIC circuit for single iteration

Here the initial vector (X_0, Y_0) which is rotated through a series of micro-rotations or iteration say, 1st iteration, 2nd iteration etc to obtain the final vector (X_n, Y_n).

Inputs - X = 0001 and Y = 0000
Shifted values- q11 = 0001 and q21 = 0001
Outputs – S1 = 0001 and S2 = 0001

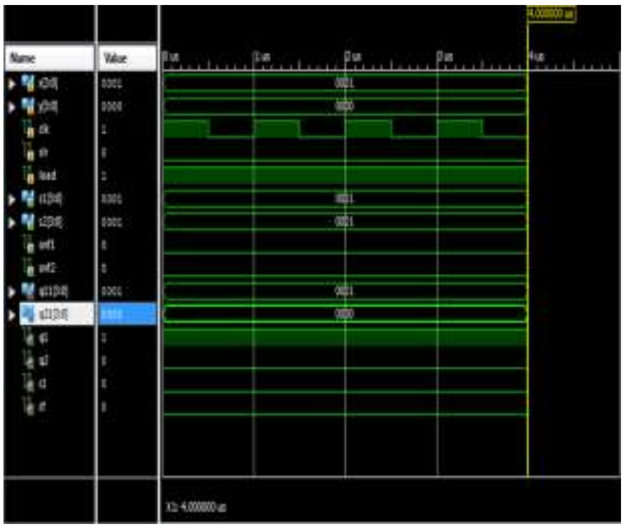

Fig. 6. Reference CORDIC circuit for 1st iteration

In Fig. 6. input X and Y is shifted for the iteration value corresponding to 45°(zero shift). The shifted value is then add/subtract to the input values according to sign bit values stored in SBR.

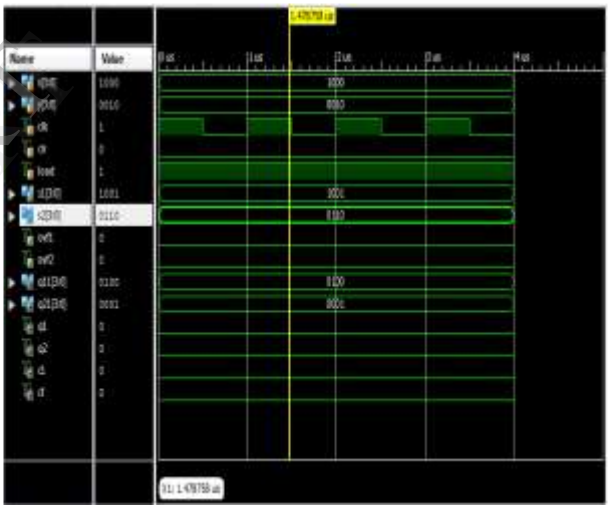

Fig. 7. Reference CORDIC circuit for 2nd iteration

In Fig. 7. the input X and Y is shifted for the iteration value corresponding to 26.6°(one shift). The shifted value is then add/subtract to the input values according to sign bit values stored in SBR.

Inputs - X = 1000 and Y = 0010
Shifted values- q11 = 0100 and q21 = 0001
Outputs – S1 = 1001 and S2 = 0110

B. Multi-stage single-rotation cascaded CORDIC circuit

Multi-stage cascaded CORDIC circuit connects 'n' number of single rotations in cascaded form Here the initial vector values are provided to the first rotation module, whose output is provided as the input to the second rotation module and this goes on till the nth rotation module finally to obtain the final vector X_n and Y_n. (X_n, Y_n) is the final vector obtained after vector rotation through n iterations from the initial vector (X_0, Y_0).

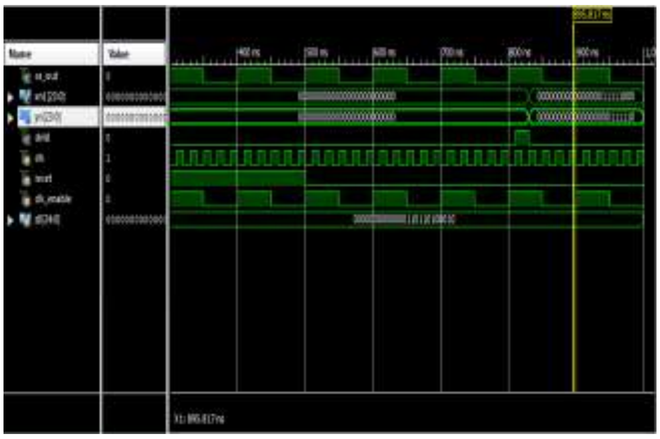

Fig. 8. Multi-stage single-rotation cascaded CORDIC circuit

In Fig. 8. the input initial values (X, Y) = (1, 0) is rotated by a fixed angle taken as 20°. For 20°, there are 7 numbers of iterations. The initial vector is rotated by the given angle to obtain the final vector (X_n, Y_n).

Outputs – X_n = 00000000000000011111000

and Y_n = 000000000000000000111110

Angle – Z_0 = 000000000000110110100010

C. *Hardwired pre-shifting scheme*

The barrel-shifter used in the optimization CORDIC circuit increases the complexity of the circuit. In order to avoid the complexity of barrel-shifter, a hardwired pre-shifting scheme is used. The barrel-shifter needs to implement a maximum of(s- *l*) shifts only, where s is the maximum number of shifts in the set of selected micro-rotations..

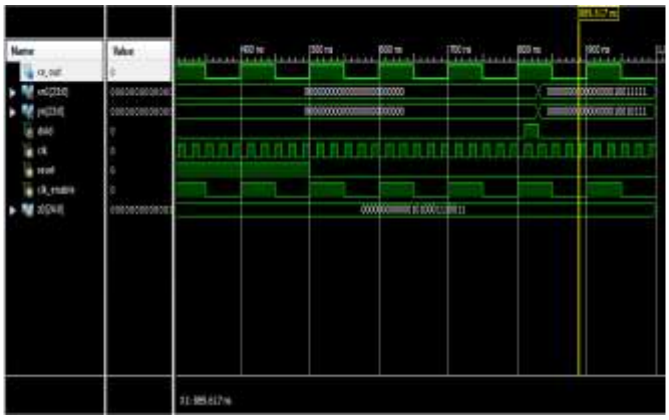

Fig. 9. Hardwired pre-shifting scheme

In this (L-*l*) MSBs are load to barrel shifters from the registers and load (l-*l*) LSBs locations by keeping *l* MSBs as 0 to adder/subtractor module. In Fig. 9. the input initial values (X, Y) = (1, 0) is rotated by a fixed angle taken as 30°. For 30°, there are 9 numbers of iterations. The initial vector is rotated by the given angle to obtain the final vector (X_n, Y_n).

IV. CONCLUSIONS

The Coordinate Rotation DIgital Computer (CORDIC) algorithm has been used for many years for efficient implementation of vector rotation operations in hardware. It is executed merely by table look-up, shift, and addition operations. In the rotation mode of CORDIC algorithm, given a vector with initial coordinate and a target rotation angle , the algorithm compute the final coordinate through a series of backward and forward rotation of the vector in an iterative manner. The project is intended to design an optimized circuit for vector rotation using CORDIC Algorithm. Optimization is performed for both Micro rotation and Scaling operation. The number of micro-rotations for rotation of vectors through fixed and known angles are optimized and several possible dedicated circuits are explored for rotation-mode CORDIC processing with different levels of accuracy

REFERENCES

[1] G.Sandhya, Syed Inthiyaz, Dr. Fazal Noor Basha, “Power and Area Optimization for Pipelined CORDIC Processor Architecture in VLSI,” *International Journal of Engineering Research and Applications (IJERA)*, Vol. 2, Issue 3, pp.1588-1595, May-Jun 2012.

[2] L. Vachhani, K. Sridharan, and P. K. Meher, “Efficient CORDIC algorithms and architectures for low area and high throughput implementation,” *IEEE Trans. Circuits Syst. II*, vol. 56, no. 1, pp. 61–65, Jan. 2009.

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

[4] S. Y. Park and N. I. Cho, “Fixed-point error analysis of CORDIC processor based on the variance propagation formula,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 3, pp. 573–584, Mar. 2005.

[5] T. Lang and E. Antelo, “High-throughput CORDIC-based geometry operations for 3D computer graphics,” *IEEE Trans. Comput.*, vol. 54, no. 3, pp. 347–361, Mar. 2005.

[6] K.Maharatna, S.Banerjee, E.Grass, M.Krstic, and A.Troya, “Modified virtually scaling free adaptive CORDIC rotator algorithm and architecture,” *IEEE Trans. Circuits Syst. for Video Technol.*, vol.15, no.11, pp.1463–1474, Nov. 2004.

[7] S. Y. Park and N. I. Cho, “Fixed-point error analysis of CORDIC processor based on the variance propagation formula,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 51, no. 3, pp. 573–584, Mar. 2004.

[8] Y. H. Hu and S. Naganathan, “An angle recoding method for CORDIC algorithm implementation,” *IEEE Trans. Comput.*, vol. 42, no. 1, pp.99–102, Jan. 1993.

[9] K. J. Jones, “2D systolic solution to discrete Fourier transform,” *IEE Proc. Comput. Digit. Techn.* vol. 136, no. 3, pp. 211–216, May 1989.

[10] J.S. Walther, “A unified algorithm for elementary functions,” in *Proc. 38th Spring Joint Comput. Conf.*, pp. 379–385, 1971.

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