IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p- ISSN: 2278-8735. Volume 7, Issue 1 (Jul. - Aug. 2013), PP 12-20
www.iosrjournals.org

Area Time Efficient Scaling Free Rotation Mode Cordic Using Circular Trajectory

N. Hima bindu M.Tech (VLSI Design), K. Geetha M.Tech, Assoc. Professor,
Dept of Electronics and Communication engineering Sri Krishna Devaraya Engineering College Gooty, Andhra Pradesh

Abstract: *This paper presents an area-time efficient Coordinate Rotation Digital Computer (CORDIC) algorithm that completely eliminates the scale-factor. Besides we have proposed an algorithm to reduce the number of CORDIC iterations by increasing the number of stages. The efficient scale factor compensation techniques are proposed which adversely effect the latency/throughput of computation. The proposed CORDIC algorithm provides the flexibility to manipulate the number of iterations depending on the accuracy, area and latency requirements. The CORDIC is an iterative arithmetic algorithm for computing generalized vector rotations without performing multiplications.*
Index Terms: *coordinate rotation digital computer (CORDIC), cosine/sine, field-programmable gate array (FPGA), most-significant-1, recursive architecture, Discrete Fourier Transform (DFT), Discrete Cosine transform (DCT), Iterative CORDIC, Pipelined CORDIC.*

I. Introduction

Year 2009 marks the completion of 50 years of the invention of CORDIC (Coordinate Rotation Digital Computer) by Jack E.volder. The beauty of CORDIC lies in the fact that by simple shift-add operations, it can perform several computing tasks such as the calculation of trigonometric, hyperbolic and logarithmic functions, real and complex multiplications, division, square-root and many others. The CORDIC is an entire-transfer computer, it contains a special arithmetic unit consisting of three shift registers, three adder- subtractor, and special interconnections. The CORDIC is applied in diverse areas such as signal and image processing, communication systems, robotics and 3-D graphics etc[1]-[3]. For applications where the angle of rotation is known in advance, a method to speed up the execution of the CORDIC algorithm by reducing the total number of iterations is presented. This is accomplished by using a technique called angle recoding. The proposed MVR-CORDIC algorithm (modified vector rotational CORDIC) will saves the 50% execution time in the iterative CORDIC structure, or 50%hardwarecomplexity in the parallel CORDIC structure compared with the conventional CORDIC scheme [4]-[6]. The corresponding architectures come for both rotation and vector modes and the other only for rotation mode to perform the scaling factor compensation in parallel with the classical CORDIC iterations. For fixed point arithmetic area and latency of the proposed implementation is compared with standard CORDIC [7]-[8]. The two area - time efficient CORDIC architectures have been suggested in [9]. In [11] the Coordinate Rotation Digital Computer (CORDIC) rotator is a well known and widely used algorithm within computers due to its way of carrying out some calculations such as trigonometric functions, many others. The new architecture which are able to reach a 35% lower latency and a 36% reduction in area and power consumption compared to the original scaling free architectures.

II. Brief Overview Of Cordic Algorithm

To evaluate trigonometric functions we have many approaches such as 1) Polynomial Approximations
2) Table lookup
3) CORDIC

1) Taylor Series

The Taylor series expansion for sine is:

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

This method is one of the oldest and most widely, but the problem associated with this method is, to get values of higher accuracies, higher order factorial and power has to be calculated. Moreover to implement this we would at least require a multiplier, divider, adder and a subtractor. For good accuracy it would be required to take each term in calculation till they become insignificant. Thus this approach has a lot of hardware requirements as well as it is slow.

2) Look up Table

The Lookup table approach involves storing values of sine and cosine at different angles. Based on the number of values stored, the lookup table can be big or small, but clearly, the smaller the lookup table, more is the error involved. The problem with a bigger lookup table is that it requires more memory and memory is expensive. Moreover the size of the Lookup table increases exponentially with the increase in the precision of the angle. Though this approach provides fast results it is very expensive to implement.

3) Cordic Algorithm

CORDIC is an acronym for Coordinate Rotation Digital Computer introduced by Jack E. Volder. It is an iterative algorithm capable of calculating trigonometric and various other functions. In this algorithm with the help of an adder/subtractor, a small look up table and a shifter the trigonometric functions can be calculated very easily. The advantage that Cordic offers over other algorithms are that it does not require multiplication or division blocks, instead it works only with a shifter, adder/subtractor and a small lookup table. This reduces the hardware requirement drastically and provides reasonably good speed.

Many variations have been suggested for efficient implementation of CORDIC with less number of iterations over the conventional CORDIC algorithm [4]–[11]. The number of CORDIC iterations are optimized in [4]–[6] by greedy search at the cost of additional area and time for the implementation of variable scale-factor. In [7] and [8] efficient scale-factor compensation techniques are proposed, which adversely affect the latency/throughput of computation. Two area-time efficient CORDIC architectures have been suggested in [9], which involve constant scale-factor multiplication for adequate range of convergence (RoC). The virtually scale-free CORDIC in [10] also requires multiplication by constant scale-factor and relatively more area to achieve respectable RoC. The enhanced scale-free CORDIC in [11] combines few conventional CORDIC iterations with scaling-free CORDIC iterations for an efficient pipelined CORDIC implementation with improved RoC. However, if used for recursive CORDIC architecture, combining two different types of CORDIC iterations, degrades performance.

The low complexity technique for eliminating the scale factor is the use of Taylor series expansion. The Scaling-Free CORDIC and modified scale-free CORDIC are techniques based on Taylor series approach.

The former suffers from low range of convergence (RoC) which renders it unsuitable for practical applications, while the latter extends the RoC but introduces predictable but constant scale-factor of $1/\sqrt{2}$. The other hardware efficient architectures require scale-factor compensations to extend the range of convergence to the entire coordinate space.

Sequential/Iterative CORDIC

It requires Maximum number of Clock Cycles to calculate output, Minimum Clock Period per iteration, Variable Shifters do not map well on certain FPGA's due to high Fan-in.

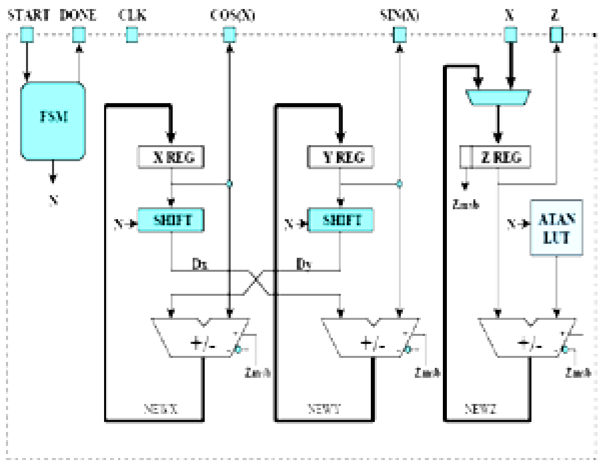

Fig:1. Variable Shifters

Parallel/Cascaded CORDIC:

It has Combinational circuit More Delay, but processing time is reduced as compared to iterative circuit. Shifters are of fixed shift, so they can be implemented in the wiring. Constants can be hardwired instead of requiring storage space.

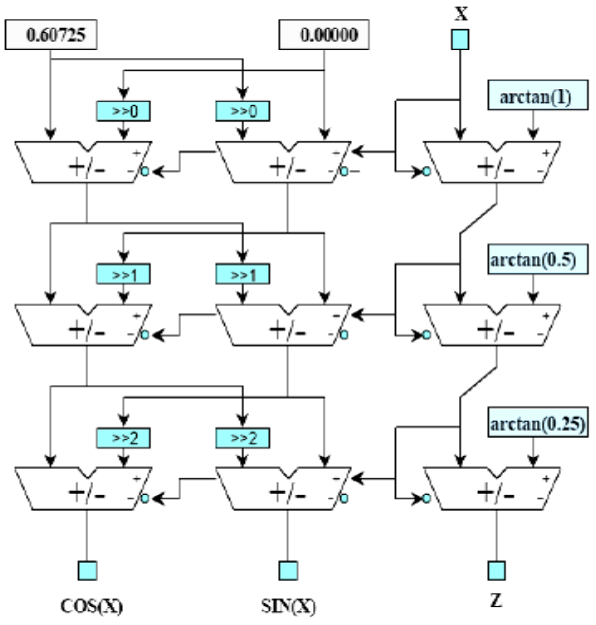

Fig.2. Parallel/Cascaded CORDIC

The key concept of CORDIC arithmetic is based on the simple and ancient principles of two-dimensional geometry. But the iterative formulation of a computational algorithm for its implementation was first described in 1959 by Jack E. Volder for the computation of trigonometric functions, multiplication and division. This year therefore marks the completion of 50 years of the CORDIC algorithm. Not only a wide variety of applications of CORDIC have emerged in the last 50 years, but also a lot of progress has been made in the area of algorithm design and development of architectures for high performance and low-cost hardware solutions of those applications. CORDIC-based computing received increased attention in 1971, by varying a few simple parameters; it could be used as a single algorithm for unified implementation of a wide range of elementary transcendental functions involving logarithms, exponentials, and square roots along with those suggested by Volder. During the same time, Cochran benchmarked various algorithms, and showed that CORDIC technique is a better choice for scientific calculator applications. The popularity of CORDIC was very much enhanced thereafter primarily due to its potential for efficient and low-cost implementation of a large class of applications which include: the generation of trigonometric, logarithmic and transcendental elementary functions; complex number multiplication, eigen value computation, matrix inversion, solution of linear systems and singular value decomposition (SVD) for signal processing, image processing, and general scientific computation.

The name CORDIC stands for Coordinate Rotation Digital Computer. Volder [Vold59] developed the underlying method of computing the rotation of a vector in a Cartesian coordinate system and evaluating the length and angle of a vector. The CORDIC method was later expanded for multiplication, division, logarithm, exponential and hyperbolic functions.

III. Pipelined Architecture

The principle of pipelining has emerged as a major architectural attribute of most present computer systems .Pipelining is one form of imbedding parallelism or concurrency in a computer system. It refers to a segmentation of a computational process (say, an instruction) into several sub processes which are executed by dedicated autonomous units (facilities, pipelining segments)

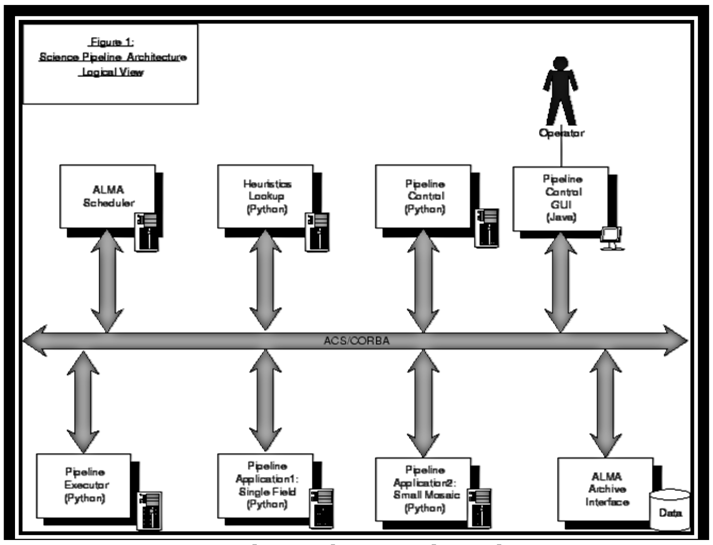
Figure 1:
Science Pipeline Architecture
Logical View
Operator
ALMA Scheduler
Heuristics Lookup (Python)
Pipeline Control (Python)
Pipeline Control GUI (Java)
ACS/CORBA
Pipeline Executor (Python)
Pipeline Application1: Single Field (Python)
Pipeline Application2: Small Mosaic (Python)
ALMA Archive Interface
Data
Fig.3.Pipe line architecture logical view

Parallel CORDIC can be pipelined by inserting registers between the adders stages. In most FPGA architectures there are already registers present in each logic cell, so pipeline registers has no hardware cost. Number of stages after which pipeline register is inserted can be modeled, considering clock frequency of system. When operating at greater clock period power consumption in later stages reduces due to lesser switching activity in each clock period.1

IV. Proposed Algorithm For Scaling Free Cordic

The proposed design is based on the following key ideas: 1) we use Taylor series expansion of sine and cosine functions to avoid scaling operation and 2) suggest a generalized sequence of micro-rotation to have adequate range of convergence (RoC) based on the chosen order of approximation of the Taylor series.

A. Taylor Series Approximation of Sine and Cosine Functions

The Taylor expansions of sine and cosine of an angle “-” are given by

$$\sin \propto = (\propto) - (3!)^{-1} \propto^3 + (5!)^{-1} \propto^5 - \cdots$$
$$\cos \propto = 1 - (2!)^{-1} \propto^2 + (4!)^{-1} \propto^4 - \cdots$$

We have estimated the maximum error in the evaluation of sine and cosine functions for different order of approximations. Therefore, we choose third order of approximation for Taylor’s expansion of sine and cosine functions.

1) Representation of Micro-Rotations Using Taylor Series Approximation*:*

Here, we study the impact of orders of approximation of Taylor series of sine and cosine functions on the micro-rotations to be used in CORDIC coordinate calculation. Both theoretical and simulation results are discussed to confirm the appropriate selection of the order of approximation. Using different orders of approximation of sine and cosine functions in (2), we can have

$$x_{i+1} = \left(1 - \frac{\alpha_i^2}{2!}\right) x_i - \left(\propto_i - \frac{\alpha_i^3}{3!}\right) y_i$$
$$y_{i+1} = \left(1 - \frac{\alpha_i^2}{2!}\right) y_i + \left(\propto_i - \frac{\alpha_i^3}{3!}\right) x_i \qquad \textit{(1a)}$$
$$x_{i+1} = \left(1 - \frac{\alpha_i^2}{2!} + \frac{\alpha_i^4}{4!}\right) x_i - \left(\propto_i - \frac{\alpha_i^3}{3!}\right) y_i$$
$$y_{i+1} = \left(1 - \frac{\alpha_i^2}{2!} + \frac{\alpha_i^4}{4!}\right) y_i + \left(\propto_i - \frac{\alpha_i^3}{3!}\right) x_i \qquad \textit{(1b)}$$
$$x_{i+1} = \left(1 - \frac{\alpha_i^2}{2!} + \frac{\alpha_i^4}{4!}\right) x_i - \left(\propto_i - \frac{\alpha_i^3}{3!} + \frac{\alpha_i^5}{5!}\right) y_i$$
$$y_{i+1} = \left(1 - \frac{\alpha_i^2}{2!} + \frac{\alpha_i^4}{4!}\right) y_i + \left(\propto_i - \frac{\alpha_i^3}{3!} + \frac{\alpha_i^5}{5!}\right) x_i \qquad \textit{(1c)}$$

$$x_{i+1} = \left(1 - \frac{\alpha_i^2}{2!} + \frac{\alpha_i^4}{4!} - \frac{\alpha_i^6}{6!}\right) x_i - \left(\propto_i - \frac{\alpha_i^3}{3!} + \frac{\alpha_i^5}{5!}\right) y_i$$
$$y_{i+1} = \left(1 - \frac{\alpha_i^2}{2!} + \frac{\alpha_i^4}{4!} - \frac{\alpha_i^6}{6!}\right) y_i + \left(\propto_i - \frac{\alpha_i^3}{3!} + \frac{\alpha_i^5}{5!}\right) x_i \qquad (1d)$$

$$x_{i+1} = \left(1 - \frac{\alpha_i^2}{2!} + \frac{\alpha_i^4}{4!} - \frac{\alpha_i^6}{6!}\right) x_i - \left(\alpha_i - \frac{\alpha_i^3}{3!} + \frac{\alpha_i^5}{5!} - \frac{\alpha_i^7}{7!}\right) y_i$$
$$y_{i+1} = \left(1 - \frac{\alpha_i^2}{2!} + \frac{\alpha_i^4}{4!} - \frac{\alpha_i^6}{6!}\right) y_i + \left(\propto_i - \frac{\alpha_i^3}{3!} + \frac{\alpha_i^5}{5!} - \frac{\alpha_i^7}{7!}\right) x_i \qquad (1e)$$

We have used (1) for coordinate calculation for evaluating the best possible combination of approximation, which satisfies the accuracy and RoC requirements, with minimum possible hardware. In Fig. 1, we have plotted the error in magnitude estimated according to (1) (with respect to the corresponding built-in functions of MATLAB). Since Errors resulting from the five combinations (1a)–(1e) are of very small order, we prefer to use (1a) for coordinate calculation with minimum complexity.

2) Expressions for Micro-Rotations Using Taylor Series Approximation and Factorial Approximation:

Although, we find that we can use Taylor series expansion with third order of approximation (1a),with desired accuracy and RoC requirement, (1a)cannot be used in the CORDIC shift-add iterations. To implement (1a) by shift-add operations, we need to approximate the factorial terms by the power of 2values, replacing 3! by 2^3 in the (1a) we find

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} (1-(2!)^{-1}.\propto_i{}^2) & -(\propto_i - 2^{-3}\propto_i{}^3) \\ (\propto_i - 2^{-3}\propto_i{}^3) & (1-(2!)^{-1}.\propto_i{}^2) \end{bmatrix} \cdot \begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad (2)$$

In Fig. 1 only, we have plotted the error in magnitude using the approximated factorial values and exact factorial values after a CORDIC rotation for initial vector with coordinates X=1 and Y=1. The maximum percentage of error in sine and cosine values for both third order of approximation and factorial approximation is 0.0004% and 0.0168%, respectively, within the permissible CORDIC elementary angles range of $\left[0, \frac{7\pi}{88}\right]$ discussed.

3) Determination of the Basic-Shift for a Given Order of Approximation of Taylor Series Expansion:

One can find that: 1) the order of approximation of Taylor series expansion of sine and cosine functions determines the basic-shift to be used for CORDIC iterations, and 2) the basic-shift of CORDIC micro operation determines the range of convergence. The expressions for the basic-shifts, the first elementary angle of rotation ($\propto_1$) and RoCfor different orders of approximations for different word-length of implementations are as follows:

Basic shift $S = \left\lceil \frac{b - \log_2 (n+1)!}{(n+1)} \right\rceil$ *(3a)*

Where b is the word length

ROC=$n_1 . \propto_1$ *(3b)*

N is number of micro rotations

$\propto_1 = 2^{-s}$ (3c)

The values in Table I are derived from (3). We find with increase in the order of approximation, the basic-shift decreases, the first elementary angle of rotation increases and RoC is expanded. Very often inclusion of higher order terms does not have any impact on the accuracy for smaller word-lengths. The basic-shift for third order of approximation using (3a), for 16-bit word-length is [2.854].

TABLE I
COMPARISION OF APPROXIMATION ORDERS VERSUS ROC FOR VARIOUS BIT WIDTHS BASED ON(7)

Order of Approx.	Basic shift		First Elementary Angle (Radians)		RoC for n_1=4 (Radians)	
	16-bit	32-bit	16-bit	32-bit	16-bit	32-bit
3	2	6	0.25	0.01562	1	0.0625
4	1	5	0.5	0.03125	2	0.125
5	1	3	0.5	0.125	2	0.5

TABLE II BIT REPRESENTATION OF ELEMENTARY ANGLES AND CORRESPONDING SHIFTS

Shift (si)	Elementary angle(α_i)	
	Decimal	16-bit Hexa Decimal
2	0.25	4000
3	0.125	2000
4	0.0625	1000
5	0.03125	0800

In this paper, we propose a novel scaling-free CORDIC algorithm for area-time efficient implementation of CORDIC with adequate RoC. The proposed recursive architecture has comparable or less area complexity with other existing scaling-free CORDIC algorithms. Moreover, no scale-factor multiplications are required for extending the RoC to entire coordinate Space.

Pseudo Code For Generating The Micro-Rotation Sequence

```
Input: angle to be rotated $\theta_i$
Begin
M=Most significant-1location ($\theta_i$)
If (M==15) then
α=0.25 radians
Shift,$s_i = 2 \; and \; \theta_{i+1} = \theta_i - \alpha$
else
shift,$s_i$=16-M
$\theta_{i+1} = \theta_i$ With $\theta_i$[M]='0'
END
```

V. Proposed Cordic Architecture

The block diagram for the proposed CORDIC architecture is shown in Fig. below. It makes use of the same stage for all the iterations for the coordinate calculations, as well as for the generation of shift values. The structure of each stage (shown in Fig. 5) consists of three computing blocks namely the 1) shift-value estimation; 2) coordinate calculation and 3) micro-rotation sequence generator.

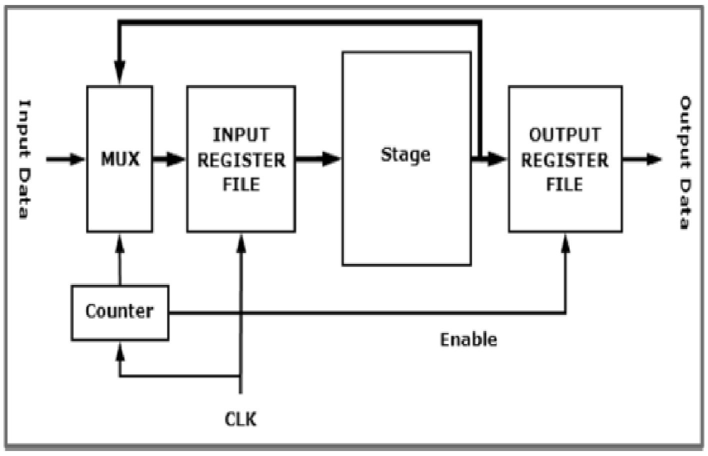

Fig.4. Recursive architecture of the proposed CORDIC processor.

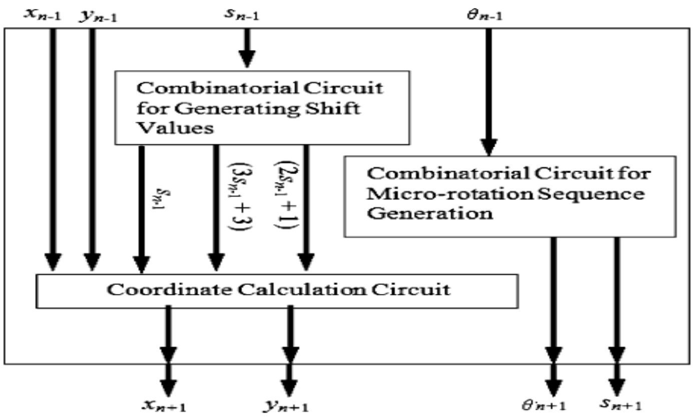
x_{n-1}
y_{n-1}
s_{n-1}
θ_{n-1}
Combinatorial Circuit for Generating Shift Values
s_{n-1}
$(3s_{n-1}+3)$
$(2s_{n-1}+1)$
Combinatorial Circuit for Micro-rotation Sequence Generation
Coordinate Calculation Circuit
x_{n+1}
y_{n+1}
θ_{n+1}
s_{n+1}

Fig. 5. Block diagram for the stage.

The combinatorial circuit for generating the micro-rotation sequence is shown in Fig. 4. The number of iterations required in a CORDIC processor decides the rollover count of the counter. The rollover count is seven for basic shift =2 and ten for basic-shift =3.

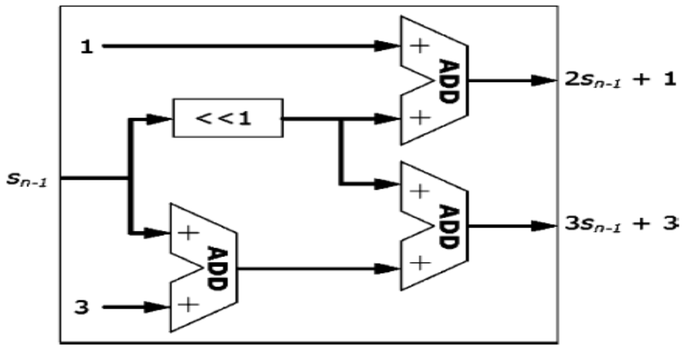

Fig. 6. Combinatorial circuit for generating the shift values.

The expiry of the counter signals the completion of a CORDIC operation; depending on this signal, the multiplexer either loads a new data-set (rotation angle, initial value of and “x”and”y”) to start a fresh CORDIC operation, or recycles the output of the stage to begin a new iteration for the current CORDIC operation. The input and output register files act as latches for synchronization.

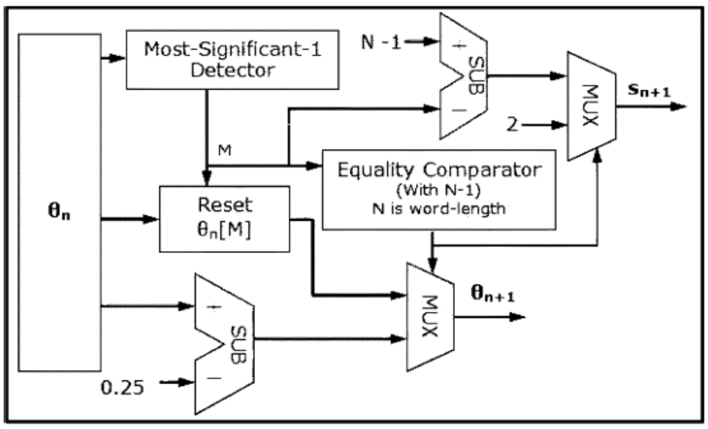
Most-Significant-1 Detector
N -1
SUB
MUX
S_{n+1}
2
M
Equality Comparator (With N-1) N is word-length
θ_n
Reset $\theta_n[M]$
MUX
θ_{n+1}
SUB
0.25

Fig 7. Micro-rotation sequence generation.

VI. Fpga Implementation

The proposed architecture is coded in Verilog and synthesized using Xilinx ISE9.2i to be implemented in Xilinx Spartan 2E (XC2S200EPQ208- 6) device. Slice-delay-product of the proposed architecture is compared with the existing CORDIC designs in Table III; where, all designs are synthesized on

Xilinx Spartan 2E XC2S200E device to maintain uniformity. The power dissipation of the proposed architecture for different clock frequencies is estimated by Xilinx XPower tool.

VII. Experimental Result And Discussion

TABLE III SLICE DELAY PRODUCT

Logic Utilization	Used	Available	Utilization
Number of Slices	958	5472	17%
Number of Slice Flip Flops	862	10944	7%
Number of 4 input LUTs	1749	10944	15%
Number of bonded IOBs	57	240	23%
Number of GCLKs	1	32	3%

Slice-delay-product of the proposed architecture is compared with the existing CORDIC designs in Table III is suggested to reduce the number of iterations for low latency implementation. The proposed CORDIC processor has 17% lower slice-delay product for identifying the micro-rotations.

VIII. Conclusion

The proposed algorithm provides a scale-free solution for realizing vector-rotations using CORDIC. The order of Taylor series approximation is decided appropriately by the proposed algorithm, not only to meet the accuracy requirement but also to attain adequate range of convergence. The generalized micro-rotation selection technique is suggested to reduce the number of iterations for low latency implementation. Moreover, a high speed most-significant-1 detection scheme obviates the complex search algorithms for identifying the micro-rotations. The proposed CORDIC processor has 17% lower slice-delay product with a penalty of about 13% increased slice consumption on Xilinx Spartan 2E device

References

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

[2] K. Maharatna, A. S. Dhar, and S. Banerjee, "A VLSI array architecture for realization of DFT, DHT, DCT and DST," Signal Process., vol. 81, pp. 1813–1822, 2001.

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

[4] C. S. Wu and A. Y. Wu, "Modified vector rotational CORDIC (MVRCORDIC) algorithm and architecture," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 48, no. 6, pp. 548–561, Jun. 2001.

[5] C.-S.Wu, A.-Y.Wu, and C.-H. Lin, "A high-performance/low-latency vector rotational CORDIC architecture based on extended elementary angle set and trellis-based searching schemes," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process, vol. 50, no. 9, pp. 589 601, Sep.2003.

[6] Y. H. Hu and S. Naganathan, "An angle recoding method for CORDIC algorithm implementation," IEEE Trans. Compute., vol. 42, no. 1, pp. 99–102, Jan. 1993.

[7] M. G. B. Sumanasena, "A scale factor correction scheme for the CORDIC algorithm," IEEE Trans. Compute., vol. 57, no. 8, pp. 1148–1152, Aug. 2008.

[8] J. Villalba, T. Lang, and E. L. Zapata, "Parallel compensation of scale factor for the CORDIC algorithm," J. VLSI Signal Process. Syst., vol. 19, no. 3, pp. 227–241, Aug. 1998.

[9] L. Vachhani, K. Sridharan, and P. K. Meher, "Efficient CORDIC algorithms and architectures for low area and high throughput implementation," IEEE Trans. Circuit Syst. II, Exp. Briefs, vol. 56, no. 1, pp. 61–65,Jan. 2009.

[10] K. Maharatna, S. Banerjee, E. Grass, M. Krstic, and A. Troya, "Modified virtually scaling-free adaptive CORDIC rotator algorithm and architecture,"IEEE Trans. Circuits Syst. Video Technol., vol. 11, no. 11,pp. 1463–1474, Nov. 2005.

[11] F. J. Jaime, M. A. Sanchez, J. Hormigo, J. Villalba, and E. L. Zapata "Enhanced scaling-free CORDIC," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 7, pp. 1654–1662, Jul. 2010.

[12] N. Takagi, T. Asada, and S. Yajima, "Redundant CORDIC methods with a constant scale factor for sine and cosine computation," IEEE Trans. on Computers, vol. 40, pp. 989–995,Sep. 1991.

[13] D. Timmermann, H. Hahn, and B. Hosticka, "Low latency time CORDIC algorithms," IEEE Trans. on Computers, vol. 41, pp. 1010–1015, Aug. 1992.

[14] J. Lee and T. Lang, "Constant-factor redundant CORDIC for angle calculation and rotation," IEEE Trans. on Computers, vol. 41, pp. 1016–1025, Aug. 1992.

[15] J. Duprat and J.-M. Muller, "The CORDIC Algorithm: New results for Fast VLSI Implementation," IEEE Transcation on Computers, vol. 42, pp. 168–178, Feb. 1993.

X. BIOGRAPHIES

N. Hima bindu [1] received B.Tech degree in Electronics & Communication Engineering from IFET College of engineering, villupuram, TN in 2011. She is now M.tech scholar in VLSI Design in Sri Krishna Devaraya Engineering college, Gooty, AP.

K.Geetha[2] received B.Tech, M.Tech degree in Electronics & Communication Engineering. She is having an experience of 10 years in the field of teaching, presently working as Associate Professor in Sri Krishna Devaraya Engineering College, Gooty, AP.