ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

Performance analysis and FPGA Implementation of Radix-2 and Radix-4 CORDIC

Prateek A Magdum, Bhagyalaxmi R Honnakasturi, J.M. Rudagi

Abstract*— Many Digital signal processing (DSP) applications are based on real time constraints. On account of this, conventional processors are not suitable for modern day DSP systems. Thus leading major issues pertaining to latency and throughput .In order to overcome issues and there by improvising in terms of performance, CORDIC is one such hardware efficient algorithm and its current trend of hardware intensive signal processing. It efficiently performs all elementary functions such as trigonometric, logarithmic, hyperbolic and exponential functions which are used in DSP systems. In this paper Radix-2 and Radix-4 CORDIC Architectures 16-bit, implemented on Xilinx 14.2FPGA platform, Simulated on ISim simulator and its results are analysed, verified and compared with MATLAB 2011b. Performance analysis of both the architectures has been done and power analysis is also discussed.

***Index Terms*—CORDIC, Cosine, Sine, FPGA, DSP.**

I. INTRODUCTION

J.E. Volder developed CO-ordinate Rotation Digital Computer (CORDIC) in 1959 to compute the rotation of two dimensional vectors [1]. Later Walther generalized this algorithm to compute logarithmic, exponential, division, hyperbolic and trigonometric functions [2]. CORDIC is an iterative algorithm for the calculation of the rotation of two dimensional vectors in linear, circular and hyperbolic coordinate systems. This rotation is carried out by a sequence of iterations. Each of this rotation over a prefixed elementary angle (micro rotation) is evaluated by means of addition and shift operations. The number of iteration of radix-2 CORDIC limits its architecture to use in high speed applications. Radix-4 CORDIC reduces the number of iterations which reduce latency and time of calculation. Antelo in the year 1996 proposed high radix CORDIC with faster implementation. This work mainly concentrated on Performance analysis of Radix-2 and Radix-4 CORDIC architectures and compares the latency of both.

In this paper, the organization of work as follows. Section II gives the basics of CORDIC algorithm. Section III describes Radix-2 CORDIC Architecture, block diagram, mathematical representation. Implementation of Radix-4 CORDIC Architecture has been described in Section IV. Section V gives simulation results, comparison plots and synthesis report. In the last section VI conclusion of this work has been discussed.

II. CORDIC ALGORITHM

The CORDIC algorithm is coordinate rotation in linear, circular and hyperbolic coordinate systems depending on which function is to be calculated. This is performed in the CORDIC algorithm by rotating a vector through a sequence of arbitrary angles whose algebraic sum approximates the desired rotation angle [1],[2]. These arbitrary angles have the property that vector rotation through each of them may be computed easily with a single shift and add operation. CORDIC operates in two modes: the rotation mode and the vector mode. In rotation mode, angle of rotation and coordinate components of original vector are given, where as in vector mode, only the coordinate of original components are given. Given angle, rotation mode is used to perform general rotation and to compute elementary operations such as trigonometric functions, multiplication, exponential, and hyperbolic functions depending on the coordinate system in which it is being rotated. The vectoring mode can be used to compute the angular argument of the original vector and to compute divisions, logarithmic functions. The number of micro rotations to be performed in both the modes depends on the application. In Cartesian plane rotating a vector by an angle θ can be arranged and equations are as follows

$$x' = \cos\theta\,[x - y\tan\theta] \quad (1)$$

$$y' = \cos\theta\,[y + x\tan\theta] \quad (2)$$

If the rotation angles are restricted so that $\tan(\theta) = \pm 2^{-i}$, the multiplication by the tangent term is condensed to a simple shift operation. Arbitrary angles of rotation are available by performing a series of consecutively smaller

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

micro rotation. If the decision at each iterations i, is which direction to relate rather than whether or not to rotate, then the cos(θ) term becomes a constant .The iterative rotation can now be expressed as [3]:

$$x_{i+1} = K_i\left[x_i + d_i 2^{-i} y_i\right] \quad (3)$$

$$y_{i+1} = K_i\left[y_i - d_i 2^{-i} x_i\right] \quad (4)$$

Where,
Ki= $1/(1+2^{-2i})^{1/2}$; known as scale constant.
di=±1; known as decision function.
Removing the scaling constant from the iterative equations yields a shift-add algorithm for vector rotation. The product of the K can be functional as part of a system processing gain or by initiating the rotating vector by the reciprocal of the gain of a certain number of iterations. The angle of a composite rotation is uniquely defined by the sequence of the directions of the micro rotations. That series can be represented by a decision vector. All possible decision vectors is an angular measurement system are based on set of binary arctangents. A favorable conversion method uses an additional adder-subtractor that holds the elementary rotation angles at each single iteration. The elementary angles can be expressed in any suitable angular unit either radians or degrees and are stored in small lookup table or it can be hardwired, depending on the implementation. The angle accumulator adds a third difference equation to the CORDIC algorithm

$$z_{i+1} = z_i - d_i \tan^{-1}(2^{-i}) \quad (5)$$

The CORDIC in rotation mode, a vector (x, y) is rotated by an angle θ. The angle accumulator is initialized with the desired rotation angle θ. The rotation decision per iteration is made to diminish the magnitude of the residual angle in the angle accumulator. Hence, the decision per iteration is based on the sign of the residual angle after each step. Normally, the angle accumulator may be eliminated, if the input angle is already expressed in the binary arctangent base.

III. RADIX-2 CORDIC ARCHITECTURE

Radix-2 architecture is as shown in Fig 1. The main pro of this type of architecture is that the barrel shifters are of fixed size and can be implemented in the wiring. Secondly, instead of requiring storage space that is ROM that holds the arbitrary angle values, need not to be restructured after each iteration because the constants can be hardwired. The LUT values for computing angle accumulator is distributed as constant to each adder in the angle accumulator chain so that the entire CORDIC processor is compact to an array of interconnected adder-subtraction units. Unlike other architectures there is no need of registers which avoids unfolded architecture strictly to behave like combinational circuit. The delay is favorable, but processing time is reduced as compared to other iterative structures. Thus produces speed required for faster applications.

The various components required for the Radix-2 CORDIC processor in unfolded style for implementation are ROM which stores the angle values $\tan^{-1}(2^{-i})$ where i is varied from 0 to 16 for 16-bit processor. There are barrel shifters required for shifting of the intermediate values Xi and Yi. The barrel shifters carry out a right shift which can be implemented using multiplexers. For next iteration for X,Y and Z computation there are addition/subtraction unit [7].
For rotation mode Radix-2 CORDIC

$$X_{i+1} = X_i - Y_i d_i 2^{-i} \quad (6)$$

$$Y_{i+1} = Y_i + X_i d_i 2^{-i} \quad (7)$$

$$Z_{i+1} = Z_i - d_i \tan^{-1}(2^{-i}) \quad (8)$$

Where σi=-1 for Zi<0. Else σi=1
After n iterations we get,

$$Xn = An\,[X0 \cos Z0 - Y0 \sin Z0] \quad (9)$$

$$Yn = An\,[Y0 \cos Z0 + X0 \sin Z0] \quad (10)$$

$$Zn = 0 \quad (11)$$

$$An = \prod_{i=0}^{n} \sqrt{(1 + 2^{-2i}} \quad (12)$$

For radix-2 CORDIC, processing gain is approximately K≈1.65.The major drawback of the conventional CORDIC algorithm is its comparatively high latency and low throughput due to the sequential nature of the iteration process with carry propagates addition and variable shifting in every iteration. To overcome these drawbacks, pipelined implementations are projected. On the other hand, the carry propagate addition remained a bottleneck for additional throughput enhancement. Two most important methodologies have been employed in order to increase the speed of CORDIC implementation. One reduces the delay of each iteration by adopting

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

redundant arithmetic to radix-2 CORDIC to reduce carry propagate addition. The other technique involves reducing the number of iterations by increasing the radix employed for the execution of CORDIC algorithm.

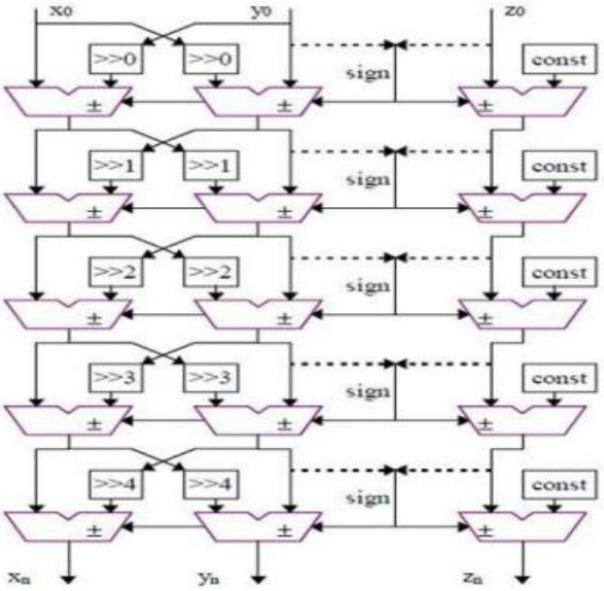

Fig 1: Radix-2 CORDIC Architecture [7].

IV. RADIX-4 CORDIC ARCHITECTURE

The architecture is designed for 16 bit precision. The unscaled rotation and compensation part presented in this architecture is applicable for 16 bit precision. The proposed compensated architecture of the radix-4 CORDIC processor consists of unscaled CORDIC architecture and the K^{-1} computation architecture. Basically in this architecture there is a circular mode of operations for eight iterations. The scale factor compensation part also parallely operates to create the reciprocal of the scale factor; the two unscaled coordinate values are compensated by the previously calculated inverse of the scale factor [3],[6].

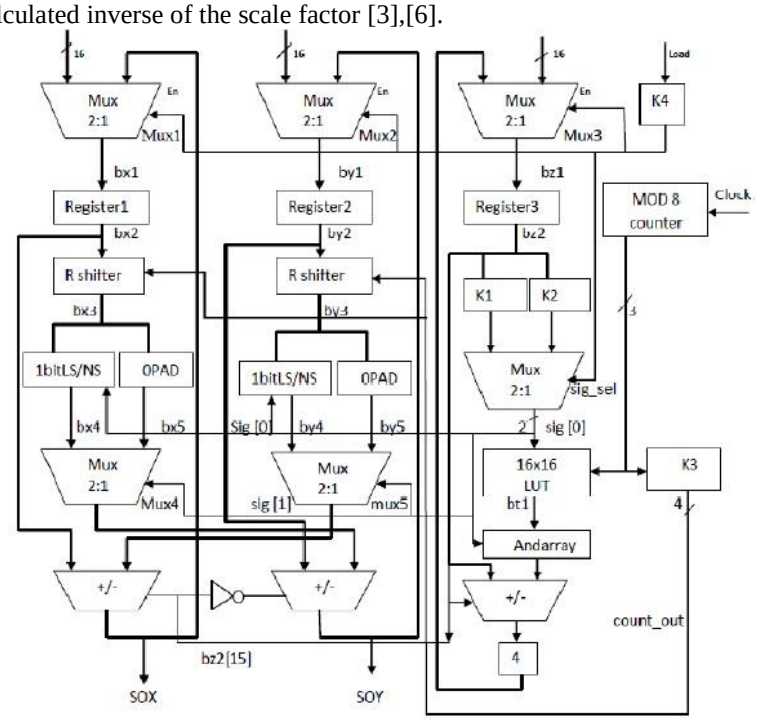

Fig 2. Unscaled Radix-4 CORDIC Architecture [6].

The unscaled CORDIC radix-4 architecture is as shown in Fig.2, consists of three processing paths for the x, y, z coordinates and a selection function path. During cycle 0, the 16 bit values of BX, BY, BZ are registered by means

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

of mux 1, 2, 3 and there by consuming an extra clock cycle to load the three values from outside of the CORDIC processor.

The block K4 function is to generate an extra delay of one clock period at the 0th count of MOD8 counter. The output values of the registers are bx1, by1, bz1. There are two hard right shifters Rshifter1 and Rshifter2. The count value at each clock edge is fed to the K3 block which is generating 4 bit values (count_out) at each clock edge. This 4 bit count_out will work as shift controller of all hard shifters. The barrel shifter carry out a right shift of d(d=2x times iteration) of the registered outputs, bx2, by2 and it will be incremented in every cycle filling the vacant left hand side with sign value. On the other hand the first 6 bit from the most significant side of register bz2 will go to the combinatorial blocks K1, K2. The K1, K2 blocks generate two different values of sigma simultaneously. Depending up on the count value of the MOD8 counter which is working at positive edge of the clock (system clock) an appropriate selection signal (sig_sel) is generated. The appropriate sigma will be selected through 2:1 multiplexer according to the value of sig_sel. The barrel right shifters outputs i.e. bx3 and by3 go to the block which are 1 bit left shifted or no shifted decided by the sig[0]. These block outputs are bx4 and by4 respectively. The zero padding blocks are dedicated to generate 16 bits of zero values. These block outputs i.e., bx5 and by5 are multiplexed with the bx4 and by4 respectively by two 2:1 multiplexers separately. The multiplexed outputs are selected by the sig [1] of the sigma [6].

The LUT holds the 16 step angles such that angles are expressed as $\tan^{-1}(\sigma_i \gamma^{-i})$ where σ_i {1, 2} and i corresponds to iteration being evaluated. For a particular iteration an angular value will be chosen from a particular location out of 16 locations. Out of corresponding 4 bit address value the first 3 bits from the most significant side are the MOD8 counter output. The least significant one is the sig [0] of the sigma. The16 bit stored value (bt1) from the Look Up Table will passes through an And-array block. The output of this block is either zero or 'bt1' itself. This block is controlled by the sig [1] of sigma. All the registers output i.e., bx2, by2, bz2 are either added or subtracted from the outputs of mux4, mux5 and And-array. The most significant bit of the angular error i.e.(bz15) will take the decision about addition or subtraction operation of these three adder/subtractor blocks. The uncompensated values of SOX, SOY can be achieved in every clock edge of system clock from the unscaled CORDIC architecture.

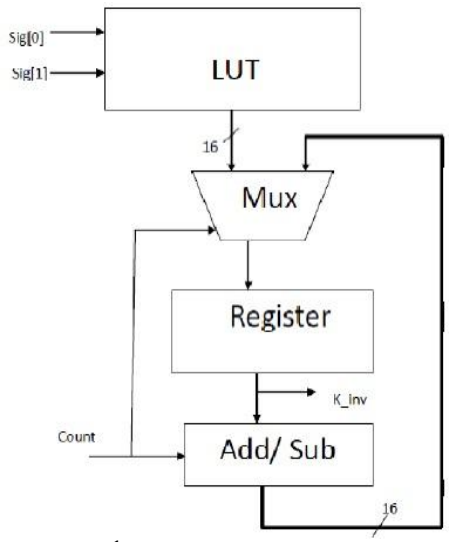

Fig 3: K^{-1} Computation Architecture [6]

Fig.3 shows K^{-1} Computation block. The scale factor is calculated in parallel with X, Y and W coordinates [6]. It requires nine cycles (i.e. n/4+1). The generated values of sigma are stored at iterations i=0 to n/4. These values of sigma with sigma at i=0 forming the most significant bits, form an addresses for ROM (LUT). The LUT stores probable values inverse of scale factor for the different combination of sigma i.e. $\sigma \in \{0, 1, 2\}$ and iterations $i \in \{0, 1, 2, \ldots n/4\}$.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

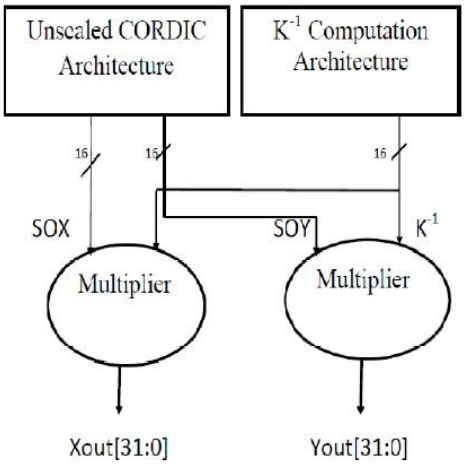

Fig 4: Scaled Radix-4 CORDIC Architecture [6].

The two architectures i.e. unscaled CORDIC architecture and K^{-1} computation architecture are clubbed together to produce appropriate result of the scaled radix-4 CORDIC processor [3]. The scaled Radix-4 CORDIC Architecture is as shown in the Fig.4. The unscaled architecture's output i.e. SOX, SOY can come at each positive edge of the system clock and the output of the K^{-1} is also comes at each clock edge. The output of unscaled architecture and the output of the K^{-1} computation architecture are multiplied by the two 16x16 multiplier. The inputs to the first multiplier are SOX and K^{-1} where as SOY and K^{-1} are the inputs to the second multiplier. Finally the two multiplier outputs will be the scaled versions of the X and Y coordinates, designated as Xout and Yout. These two compensated coordinate values will be each of 32 bit in word length [11][12].

V. IMPLEMENTATION AND RESULTS

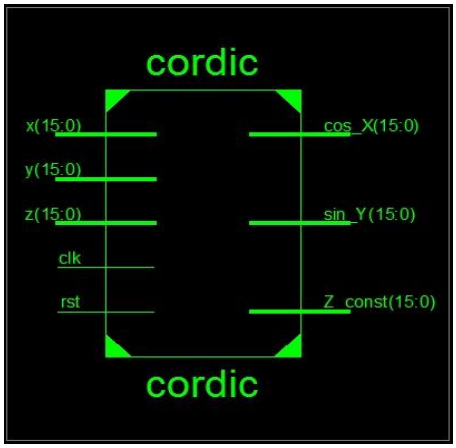

Fig 5: Top level RTL view of Radix-2 CORDIC

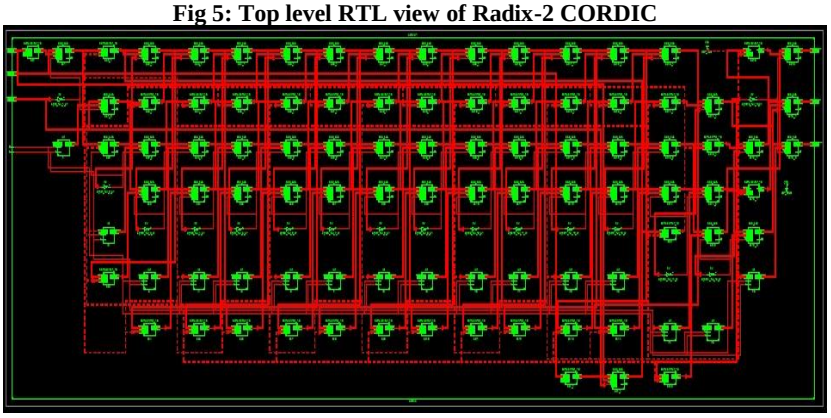

Fig 6: Schematic view of Radix-2 CORDIC

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

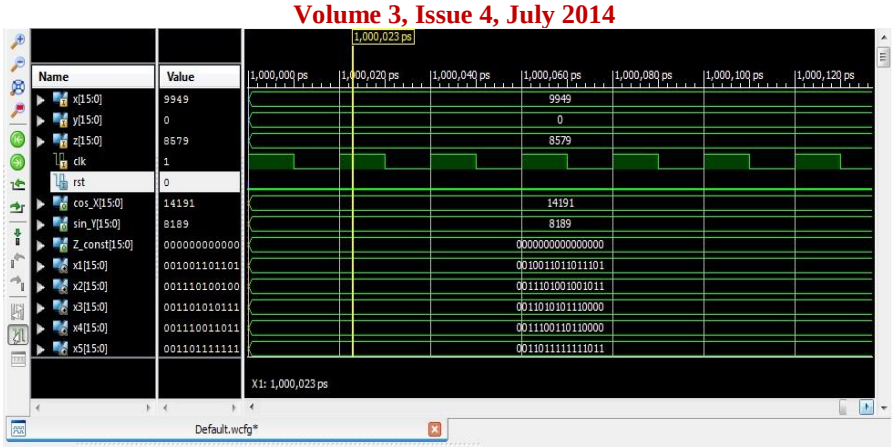

Fig 7. Simulation result of COSINE an SINE values for angle 30 degree.

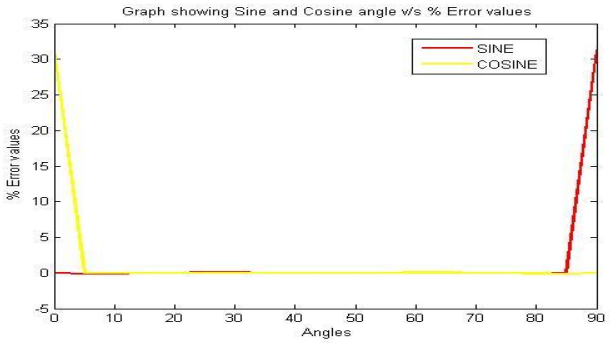

Fig 8:Comparison Graph showing Angle v/s % Error of Simulated and MATLAB values for Sine and Cosine.,Radix-2 CORDIC

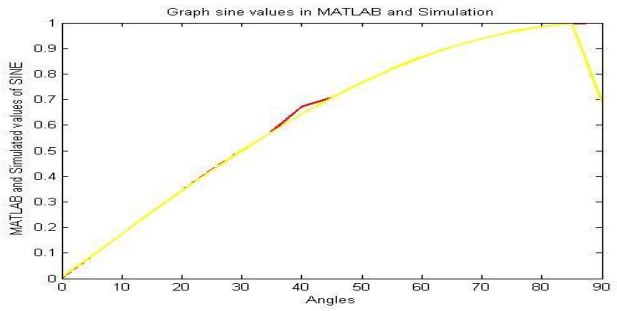

Fig 9: Comparison graph of MATLAB and Simulated error values for Sine, Radix-2 CORDIC

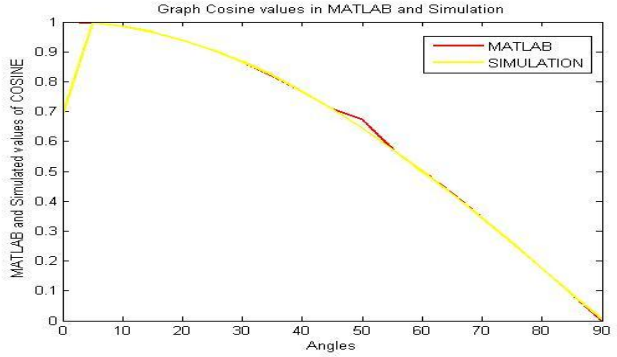

Fig 10: Comparison graph of MATLAB and Simulated error values for Cosin, Radix-2 CORDIC

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

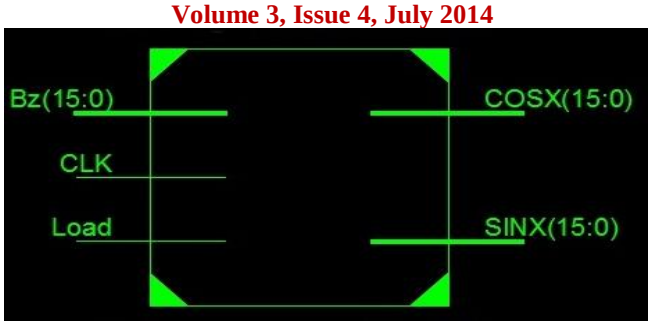

Fig 11: Top level RTL view of Radix-4 CORDIC

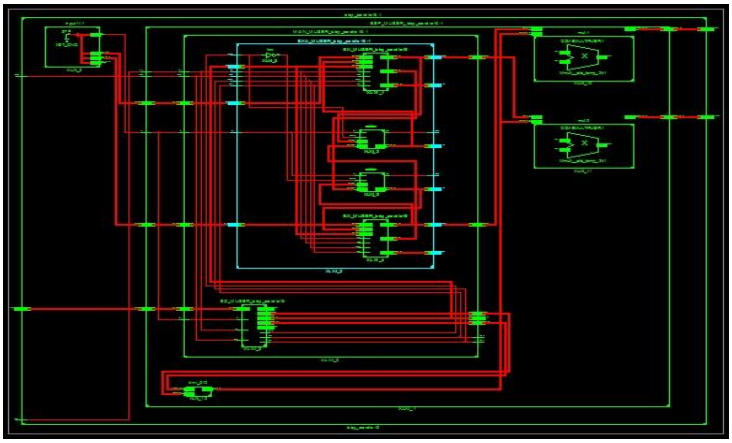

Fig 12: Schematic view of Radix-4 CORDIC.

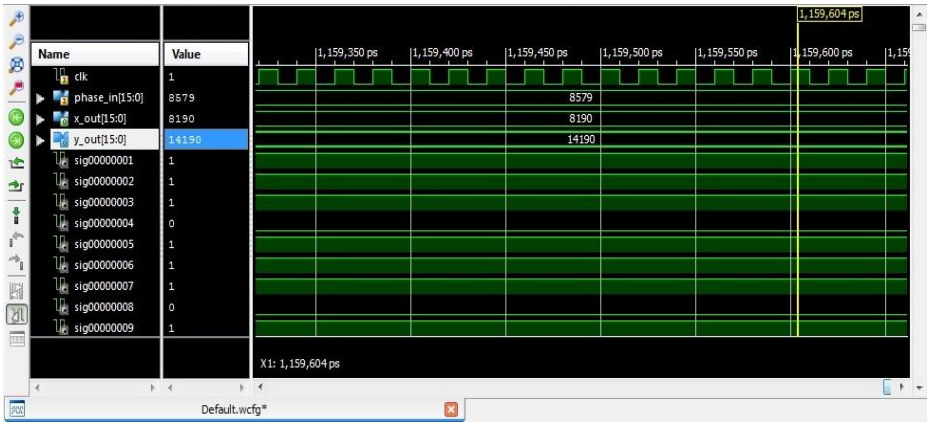

Fig13: Simulation result of COSINE an SINE values for angle 60 degree.

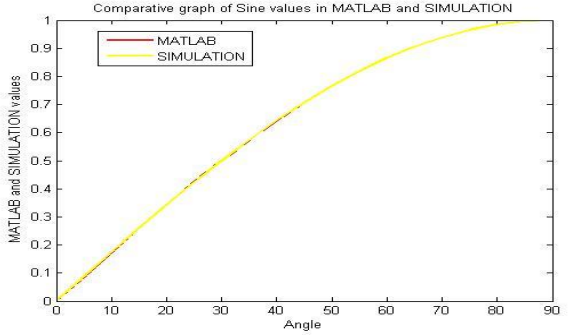

Fig 14: Comparison graph of MATLAB and Simulated error values for Sine, Radix-4 CORDIC

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

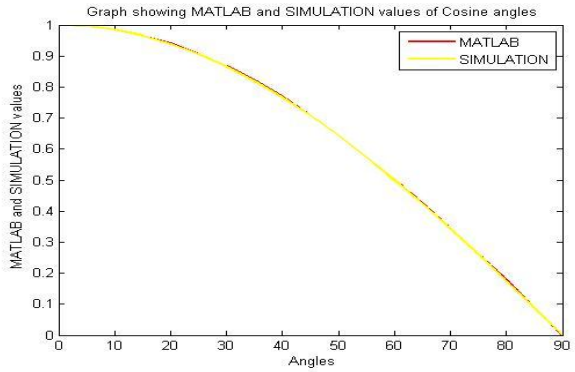

Fig 15: Comparison graph of MATLAB and Simulated error values for Cosine, Radix-4 CORDIC

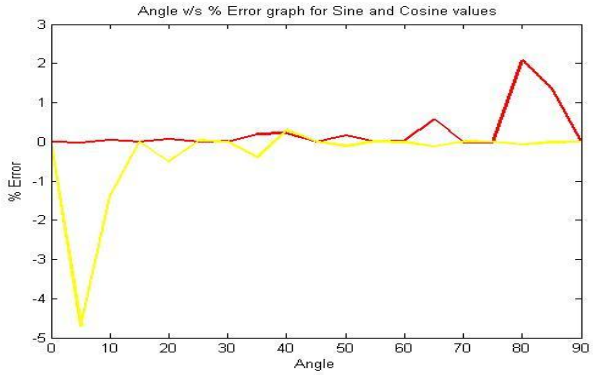

Fig 16: Graph showing Angle v/s % Error of Simulated and MATLAB values., Radix-4 CORDIC

Comparative Study:

Table I. Comparative Analysis for Radix-2 and Radix-4 architecture

Parameters	Radix-2	Radix-4
Number of Slices	1 out of 93120	72 out of 93120
Number of fully used LUT-FF Pairs	0 out of 966	71 out of 46560
Number of Slice LUTs	965 out of 46560	280 out of 46560
Number of bonded IOBs	98 out of 240	50 out of 240
Number of BUFG/BUFGCTRLs	1 out of 32	1 out of 32
Speed- MHz	22.18 MHz	210.643 MHz
Power (W)	1.293	1.293
Bit Precision	16	16

VI. CONCLUSION

CORDIC is a powerful algorithm, and a popular algorithm of alternative when it comes to various Digital Signal Processing applications. Implementation of a CORDIC based processor on FPGA gives us a powerful mechanism of implementing complex computations on a platform that provides a lot of resources and flexibility at a relatively lesser cost. Further, since the algorithm is straightforward and efficient, the design and VLSI implementation of a CORDIC based processor is easily achievable. A 16 bit Radix-2 and Radix-4 CORDIC architecture is designed and implemented on Xilinx FPGA (Virtex6 – xc6vlx75t-1ff484) and analysis of the CORDIC algorithm is done on the MATLAB (R2011b version 7.13). The comparative study for Radix-2 and Radix-4 CORDIC has been

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

carried out and shown in Table I. The Radix-2 architecture operates at the clock frequency of 22.18 MHz and Radix-4 operates at the clock frequency of 210.643 MHz with the power consumption of 1.293 W. The Radix-4 architecture operates with increased speed, reduced area and operating power as compared with Radix-2 architecture.

ACKNOWLEDGMENT

The authors would like to thank Prof. J.M.Rudagi for her constant support and guidance. The authors also thank the KLEs Dr.M.S.Sheshgiri College of Engineering and Technology for all the facilities provided. The Authors also express sincere gratitude towards all the teaching and non-teaching faculties and dear friends for their support.

REFERENCES

[1] J.E. Volder, "The CORDIC trigonometric computing technique", IRE Trans Electronic Computers 8(1959)330-334.

[2] J.S. Walther, "A unified algorithm for elementary functions", in: Proceedings of spring. Joint Computer Conference, 1971, pp. 379–385.

[3] Kaushik Bhattacharyya, Rakesh Biswas, Anindya Sundar Dhar, Swapna Banerjee "Architectural design and FPGA implementation of radix-4 CORDIC processor" Microprocessors and Microsystems 34 (2010) 96–101

[4] Pramod K. Meher et. Al "50 Years of CORDIC: Algorithms, Architectures, and Applications" IEEE transactions on circuits and systems—I: regular papers, vol. 56, no. 9, September 2009 pp 1893-1907.

[5] J M Rudagi and Dr shaila subbaraman, "FPGA Design, Implementation and Analysis of Trigonometric Generators using Radix 4 CORDIC Algorithm", International Journal of Microcircuits and Electronic. ISSN 0974-2204 Volume 4, Number 1 (2013), pp. 1-9

[6] J M Rudagi, Srikant, Basavaraj B Patil, Dr S Subbaraman, "Performance Analysis of Radix 4 CORDIC Processor in Rotation mode with Parallel Scale factor Computation", International Journal of Emerging Technology and Advanced Engineering. (ISSN 2250-2459, Volume 2, Issue 7, July 2012)

[7] J. M. Rudagi, Vinayak Dalavi, "Radix-2 CORDIC Method with Constant Scale Factor", International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering Vol. 2, Issue 7, July 2013.

[8] Srinivasa Murthy H N, Roopa M, "FPGA Implementation of Sine and Cosine Generators using CORDIC Algorithm", International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-1, Issue-6, November 2012.

[9] K. Hwang, "Computer Arithmetic Principles, Architecture and Design", Wiley, NewYork, 1979.

[10] Takafumi et al., "High Radix CORDIC algorithm for VLSI signal processing", in: IEEE Workshop on Signal Processing Systems (SIPS), November 1997, pp. 183–192

[11] M.G. Buddika Sumanasena, "A scale factor correction scheme for the CORDIC algorithm", IEEE Transactions on Computers 57 (8) (2008) 1148–1152.

[12] P. R. Rao et al., "High performance compensation technique for the radix-4 CORDIC algorithm", IEE Processing of computer and digital technique 149(5) (2002) 219-228.

AUTHOR BIOGRAPHY

Prateek A. Magadum received the B.E degree in Electrical and Electronics Engineering from HIT, Nidasoshi, India in 2009. He is currently pursuing M.Tech degree in VLSI Design and Embedded system from Visvesvaraya Technological University. His research interests are VLSI design and embedded systems.

Bhagyalaxmi R. Honnakasturi received the B.E. degree in Electronics and Communication Engineering from SDMCET, Dharawad, India in 2012. She is currently pursuing M.Tech degree in VLSI Design and Embedded system from Visvesvaraya Technological University. Her research interests are VLSI design and embedded systems.

ISSN: 2319-5967
ISO 9001:2008 Certified
International Journal of Engineering Science and Innovative Technology (IJESIT)
Volume 3, Issue 4, July 2014

J. M. Rudagi received the B.E. degree in Electrical Engineering from the Karnataka University, Dharwad, India, in 1991 and M. Tech in Digital Electronics and Advanced Communication from Manipal University, Manipal, India. She is currently working as an Associate professor in KLE Dr. M S Sheshgiri College of Engineering and Technology, Belgaum. She has 22 years of teaching experience. Her research interests are in low power VLSI design