ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering
(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2014

Design and Implementation of Bi-Rotational CORDIC Algorithm

K.Surya Kumari [1], S.Jahnavi[2]

M.Tech, Asso. Professor, Dept of Electronics & Communication Engineering, Pragati Engineering College, Surampalem, AP, India[1]

M.Tech Student, Dept of Electronics & Communication Engineering, Pragati Engineering College, Surampalem, AP, India[2]

ABSTRACT: The CORDIC algorithm is a repetitive calculation approach ability of emerging different basic functions with a proper shift-and-add method Used to evaluate a large amount of functions. It contains no. of adder-sub tractors, shift registers with respect to complexity of operation. In this paper we present improved method of shifting by using an alternate scheme by increasing the no. of barrel shifters with increasing pre shifting method and Fault Tolerance in Bi Rotational CORDIC circuits Higher rate of accuracy in fixed and known rotations. The improvement in the fixed angle Rotation reducing the area- and Complexity in the application. From the basic architecture of cordic an Fixed angle rotation is implemented by vector rotation. the rotation of vectors uncontrolled by the circuit till all rotations are completed it will results large system gain and unpredictable angles for effective operation of known angles in this paper angle correction ,Quadrant correction and gain correction is implemented. the angle correction is selected by the initial vector bits of selection and the System gain is controlled by an external gain control mechanism of fixed system gain similar to the Normal cordic implementation.

KEYWORDS: CORDIC algorithm, rotation mode and vector mode, control CORDIC.

I. INTRODUCTION

THE Coordinate Rotation DIgital Computer (CORDIC) algorithm [1], [2] 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. Thus, the corresponding hardware can be implemented in very economic fashion. Subsequently, it has been applied for many performance demanding applications in digital signal processing (DSP), image processing, and video technology like fast Fourier transform (FFT) [3], [4], discrete Hartley transform (DHT) [4], [5], discrete cosine transform (DCT) [4], [6], discrete sine transform (DST) [4], Hough transform (HT) [7]–[9], [12], graphics application [10], [11], and motion vector estimation[12]. In essence, a CORDIC can be operated in two different modes: the rotation and the vectoring mode. In the former mode of operation, given a vector with initial coordinate (xo,yo) and a target rotation angle zo the objective is to compute the final coordinate(x1,y1) through a series of backward and forward rotation of the vector in an iterative manner. Digital signal processing (DSP) algorithms exhibit an increasing need for the efficient implementation of complex arithmetic operations. The computation of trigonometric functions, coordinate transformations or rotations of complex valued phasors is almost naturally involved with modern DSP algorithms. Popular application examples are algorithms used in digital communication technology and in adaptive signal processing. While in digital communications, the straightforward evaluation of the cited functions is important, numerous matrix based adaptive signal processing algorithms require the solution of systems of linear equations, QR factorization or the computation of Eigen values, eigenvectors or singular values. All these tasks can be efficiently implemented using processing elements performing vector rotations.

The Coordinate Rotation Digital Computer algorithm (CORDIC) gives the opportunity to calculate the desired functions in rather simple, and elegant way. CORDIC is a method for computing elementary functions using minimal hardware such as shift's, adds/subs and compares. CORDIC works by rotating the coordinate system through constant angles until the angle is reduces to zero. The angle offsets are selected such that the operations on X and Y are only shifts and adds.

All trigonometric functions can be computed using vector rotation. The CORDIC algorithm was developed by Volder in 1959. It rotates the vector, step by step, with a given angle. Additional theoretical work has been done by Walther in 1971. The main principle of CORDIC are calculations based on shift registers and adders instead of multiplications, what saves much hardware resources. CORDIC is used for polar to rectangular and rectangular to polar conversions and also for calculation of trigonometric functions, vector magnitude and in some transformations, like discrete Fourier transform (DFT) or discrete cosine transform (DCT). In particular case, the CORDIC algorithm is used in wireless LAN (WLAN) by receivers.

II. CORDIC ALGORITHM

All the trigonometric functions can be computed or derived from functions using vector rotations. The CORDIC algorithm provides an iterative method of performing vector rotations by arbitrary angles using only shift and add operations. The algorithm is derived using the general rotation transform.

The CORDIC algorithm performs a planar rotation. Graphicall y, planar rotation means transforming a vector (Xi, Yi) into a new vector (Xj, Yj).

$$X = r\cos\theta \quad ,Y = r\sin\theta \qquad (1)$$

$$V' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x\cos\phi - y\sin\phi \\ y\cos\phi + x\sin\phi \end{bmatrix} \qquad (2)$$

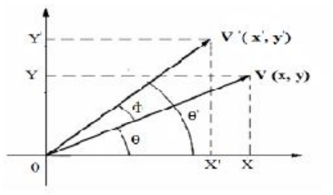


Fig 1 Rotation of a Vector V by the angle ϕ [3]

V came into picture after anticlockwise rotation by an angle Ǿ . From Fig.1, it can be observed

$$\theta' - \theta = \phi \qquad (3)$$

$$OX' = x' = r\cos\theta' \qquad (4)$$

$$= r\cos(\phi + \theta) \qquad (5)$$

$$= (r\cos\theta).\cos\phi - (r\sin\theta).\sin\phi \qquad (6)$$

Using Fig 2

$$OX' = x' = x\cos\phi - y\sin\phi$$

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering
(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2014

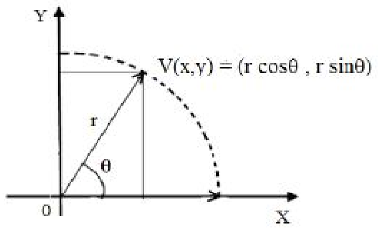


Fig 2 Vector V with magnitude r and phase θ

$$\mathrm{OY}' = y' = r\sin\theta' \quad (7)$$

$$= x\sin\phi + y\cos\phi \quad (8)$$

V′ after anticlockwise rotation of vector V by angle Ǿ is

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \quad (9)$$

It is well known that the rotation matrix

$$R(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \quad (10)$$

will rotate a vector $\begin{bmatrix} x \\ y \end{bmatrix}$, anticlockwise by radians in two dimensional spaces [6]. If this rotation matrix is applied to the initial vector $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$, the result will be vector co-ordinates of $\begin{bmatrix} \cos\phi \\ \sin\phi \end{bmatrix}$. It is easily seen that CORDIC method could be applied to calculate the functions $\sin(\phi)$ and $\cos(\phi)$ by in a program-like style:

```
For n=0 to [inf]
    If (Z(n) >= 0) then
    Z(n + 1) := Z(n) – atan(1/2^n);
    Else
    Z(n + 1) := Z(n) + atan(1/2^n);
    End if;
    End for;
```

In CORDIC processing, a bulk scale factor is generated that needs to be compensated. As long as all of the iterations allowed by a certain word length are carried out, the scale factor remains a constant and can be compensated with minimal hardware. However, in principle, the vector rotation for the majority of the angles lying in the coordinate space can be carried out using a smaller number of iterations by optimally choosing appropriate elementary rotation steps. This essentially requires by passing or repeating some of the CORDIC iterations as was proposed in [13] and [25]. However, in such a case, the scale factor no longer remains constant or predictable. The situation is much worse when

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering
(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2014

the CORDIC works in an embedded system where other circuitry of that system (a typical example is an OFDM synchronizer for IEEE 802.11(a) standard [15]) generates data and supplies it to the CORDIC. In such a case, the CORDIC has no *a priori* information about the actual angle of rotation and, hence, the scale factor is completely unpredictable. For its compensation, hardware circuitry is needed that is as complex as the original CORDIC itself. To the best of our knowledge, there is no existing scheme reported to date that can keep the scale factor constant and predictable while at the same time adaptively executing the minimum number of elementary rotation steps However, the former approach results in an overhead in hardware while the latter increases the total number of CORDIC iterations.

applying successive rotations to the initial vector $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$.It is possible to modify the rotation matrix by bringing a $\cos(\phi)$ term out of the matrix. Then [6]

$$R(\phi) = \cos(\phi)\begin{bmatrix} 1 & \tan(\phi) \\ \tan(\phi) & 1 \end{bmatrix} \qquad (11)$$

The multiplication by the tangent term can be avoided if $\tan(\phi) = 2^{-i}$. In digital hardware, this denotes a simple shift operation and since $\cos(\phi) = \cos(-\phi)$ is constant for a fixed number of iterations.

$$R(\phi) = \cos(\phi_i)\begin{bmatrix} 1 & -2^{-i} \\ 2^{-i} & 1 \end{bmatrix} \qquad (12)$$

Equation (9) can be expressed as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \cos(\phi_i)\begin{bmatrix} 1 & -2^{-i} \\ 2^{-i} & 1 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} \qquad (13)$$

ϕ_i may be positive or negative depending upon whether the rotation is anticlockwise or clockwise. Equation (13) can be expressed as

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \cos(\phi_i)\begin{bmatrix} 1 & -d_i 2^{-i} \\ d_i 2^{-i} & 1 \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} \qquad (14)$$

The basic CORDIC equation can now be expressed [7] as

$$x_{i+1} = k_i(x_i - y_i d_i 2^{-i}) \qquad (15)$$

$$y_{i+1} = k_i(y_i + x_i d_i 2^{-i}) \qquad (16)$$

Angle accumulator is

$$z_{i+1} = z_i - d_i \tan^{-1} 2^{-i}$$

Or $$z_{i+1} = z_i - d_i \phi_i \qquad (17)$$

The architecture of the new CORDIC rotator can be derived by a suitable hardware mapping of the algorithm described above. For sake of clarity, the implementation of a 16-bit CORDIC rotator is described here as an example. All of the discussions presented in this section can be generalized for an n- bit CORDIC rotator as well.

The atan(1/2^i) is pre-calculated and stored in a table. [inf] is replaced with the required number of iterations, which is about 1 iteration per bit (16 iterations yield a 16bit result).

Where i denote the number of rotation required reaching the required angle of the required vector. $k_i = \cos(\arctan(2^{-i}))$ and $d_i = \pm 1$ (direction of rotation), the product of the k_i represents K-factor.

$$k = \prod_{i=0}^{n-1} k_i \tag{18}$$

Where $\prod_{i=0}^{n-1} k_i = \cos\phi_0 . \cos\phi_1 . \cos\phi_2 . \cos\phi_3 ..... \cos\phi_{n-1}$ (ϕ is the angle of rotation here for n times rotation). These ϕ_i are stored in the ROM of the CORDIC hardware as the look up table. k_i is the CORDIC gain. For 8-bit hardware CORDIC approximation method, the value of k_i as

$$k_i = \prod_{i=0}^{7} \cos\ \phi_i = \cos\ \phi_0 . \cos\ \phi_1 . \cos\ \phi_2 .... \cos\ \phi_7$$

$$= \cos 45^o . \cos 26.565^o . \cos 14.036^o ...... \cos 0.4469^o$$

$$= 0.6073 \tag{19}$$

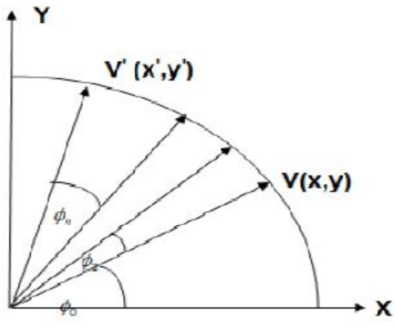


Fig 3 chaining multiple micro-rotation together

As shown in fig 3, multiple micro rotations may be chained together as the vector moves in discrete angular steps $(\phi_0, \phi_1, \phi_2, ---, \phi_n)$ from its initial position of $V(x, y)$ towards its final target position of $V'(x', y')$

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering
(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2014

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \cos\phi_0 . \cos\phi_1 \ldots . \cos\phi_n \begin{bmatrix} 1 & -\tan\phi_0 \\ \tan\phi_0 & 1 \end{bmatrix} \ldots \begin{bmatrix} 1 & -\tan\phi_n \\ \tan\phi_n & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

For 8 bit CORDIC hardware, cosine and sine of angle ϕ can be represented in matrix form by using equation (19)

$$\begin{bmatrix} \cos\phi \\ \sin\phi \end{bmatrix} = \begin{bmatrix} 1 & -\tan\phi_0 \\ \tan\phi_0 & 1 \end{bmatrix} \ldots \begin{bmatrix} 1 & -\tan\phi_7 \\ \tan\phi_7 & 1 \end{bmatrix} \begin{bmatrix} 0.6073 \\ 0 \end{bmatrix}$$

TABLE 1

FOR 8, 16, 24, 32 BIT CORDIC HARDWARE [8]

i	$2^{-i} = \tan\phi_i$	$\phi_i = \tan^{-1}(2^{-i})$	ϕ_i *in radians*
0	1	45°	0.7854
1	0.5	26.565°	0.4636
2	0.25	14.036°	0.2450
3	0.125	7.125°	0.1244
4	0.0625	3.576°	0.0624
5	0.03125	1.7876°	0.0312
6	0.015625	0.8938°	0.0156
7	0.0078125	0.4469°	0.0078
8	0.00390625	0.2238°	0.0039
9	0.001953125	0.1119°	0.0019
10	0.0009765625	0.05595°	0.00098
11	0.00048828125	0.02798°	0.00049
12	0.000244140625	0.01399°	0.00024
13	0.0001220703125	0.00699°	0.00012
14	0.00006103515625	0.00349°	0.000061
15	0.00003051757813	0.00175°	0.000031
16	0.00001525878906	0.000874°	0.0000152
17	0.000007629394531	0.000437°	0.0000076
18	0.000003814697266	0.0002186°	0.0000038
19	0.000001907348633	0.0001093°	0.0000019
20	0.0000009536743164	0.00005464°	0.00000095
21	0.0000004768371582	0.0000273°	0.00000048
22	0.0000002384185791	0.00001366°	0.00000024
23	0.0000001192092896	0.00000683°	0.00000012
24	0.00000005960464478	0.000003415°	0.0000000595
25	0.00000002980232239	0.000001708°	0.0000000298
26	0.00000001490116119	0.000000854°	0.0000000149
27	0.000000007450580597	0.000000427°	0.00000000745
28	0.000000003725290298	0.000000213°	0.00000000371
29	0.000000001862645149	0.000000107°	0.00000000187
30	0.0000000009313225746	0.0000000534°	0.00000000093
31	0.0000000004656612873	0.0000000267°	0.00000000047

For 16 bit CORDIC hardware, the value of k_i is

$$k_i = \prod_{i=0}^{15} \cos\phi_i = \prod_{i=0}^{15} \cos\phi_0 . \cos\phi_1 \ldots \cos\phi_{15} = 0.6073$$

For 24 bit CORDIC hardware, the value of k_i is

$$k_i - \prod_{i=0}^{23} \cos\phi_i - \prod_{i=0}^{23} \cos\phi_0 . \cos\phi_1 \ldots \cos\phi_{23} - 0.6073$$

For 32 bit CORDIC hardware, the value of k_i is

$$k_i - \prod_{i=0}^{31} \cos\phi_i - \prod_{i=0}^{31} \cos\phi_0 . \cos\phi_1 \ldots \cos\phi_{31} = 0.6073$$

CORDIC gain k_i is same for all the CORDIC hardware.

II. SCALING OPTIMIZATION AND IMPLEMENTATION

Optimized set of angle rotations and micro-rotations are discussed in this section

A. Fixed Rotation Angle scaling approach

The normalized equation for the scale-factor given by (2) can be articulated explicitly for the set of selected m1 micro rotations as

$$K = \prod_{i=0}^{m_1-1} \left[\left(1 + 2^{-2k(i)}\right)\right]^{-1/2}$$

where **k**(i) for 0≤i<m is the no. of shifts in the **i**th micro-rotation. Except for k(i)=0 (i.e., rotation by 45), by binomial expansion, any term can be written as

$$1 - \frac{x}{2} + \frac{3x^2}{8} - \frac{5x^3}{16} + \frac{35x^4}{128} - \frac{63x^5}{256} + \frac{231x^6}{1024} - \cdots$$

where $x = 2^{-2i}$.

III. SCALING IMPLEMENTATION FOR CORDIC

Micro-rotations and Scaling implemented either in the same circuit in interleaved manner or in two individual stages. The realization of scaling as well as the micro-rotation would however depend on desired level of accuracy, and the realization of scaling also depends on the realization of micro-rotations. Therefore, we discuss here the implementation of the scaling circuits with respect to different implementations of micro-rotations.

Computing Sine and Cosine functions

Sine and Cosine can be calculated using the first CORDIC scheme which calculates:

$$[X_j, Y_j, Z_j] = [P(X_i \cos(Z_i) - Y_i \sin(Z_i)), P(Y_i \cos(Z_i) + X_i \sin(Z_i)), 0]$$

By using the following values as inputs

$$X_i = \frac{1}{P} = \frac{1}{1.6467} \approx 0.60725$$

$$Y_i = 0$$

$$Z_i = \theta$$

the core calculates:

$$[X_j, Y_j, Z_j] = [\cos\theta, \sin\theta, 0]$$

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering
(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2014

The input Z takes values from –180degrees to +180 degrees where:
0x8000 = –180degrees
0xEFFF = +80degrees
But the core only converges in the range –90degrees to +90degrees.

The other inputs and the outputs are all in the range of –1 to +1. The congregate constant P represented in this format results in:

$$Xi = 2^{15} \bullet P = 19898(dec) = 4DBA(hex)$$

Example:
Calculate sine and cosine of 30degrees.
First the angle has to be calculated:

$$360\deg \equiv 2^{16}$$

$$1\deg \equiv \frac{2^{16}}{360}$$

$$30\deg \equiv \frac{2^{16}}{360} \bullet 30 \approx 5461(dec) = 1555(hex)$$

The core calculates the following sine and cosine values for Zi=5461:
Sin : 16380(dec) = 3FFC(hex)
Cos : 28381(dec) = 6EDD(hex)
The outputs represent values in the –1 to +1 range. The results can be derived as follows:

$$2^{15} \equiv 1.0 \qquad\qquad 2^{15} \equiv 1.0$$

$$16380 \equiv \frac{1.0}{2^{15}} \bullet 16380 = 0.4999 \qquad\qquad 28381 \equiv \frac{1.0}{2^{15}} \bullet 28381 = 0.8661$$

	0 deg	30 deg	45 deg	60 deg	90 deg
Sin	0x01CC	0x3FFC	0x5A82	0x6EDC	0x8000
Cos	0x8000	0x6EDD	0x5A83	0x4000	0x01CC
Sin	0.01403	0.49998	0.70709	0.86609	1.00000
Cos	1.00000	0.86612	0.70712	0.50000	0.01403

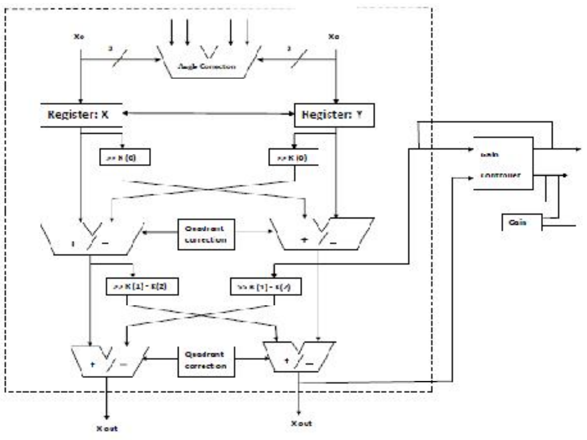
Where as the result should have been 0.5 and 0.8660.
Table 1: Sin/Cos outputs for some common angle

Although the core is very accurate small errors can be introduced by the algorithm (see example and results table). This should be only a problem when using the core over the entire output range, because the difference between +1 (0x7FFF) and –1 (0x8000) is only 1bit.

Generalized Implementation of Scaling

The shift and add circuits for scaling with respect to (7) is shown. The scaling circuit of Shift and add scaling circuit by hardwired pre-shifted loading can use hardwired pre-shifting for minimizing barrel-shifter difficulty and could be located after the CORDIC cell of Fig. 2 to perform micro-rotation and scaling in two separate stages. The generalized CORDIC circuit for fixed rotation to perform the micro-rotation and the scaling in interleaved manner in alternate cycles is shown. The circuit of Fig. 8 is similar to that of Fig. It involves only an additional line-changer circuit to change the path of unshifted (direct) input.

VII. BI ROTATIONAL CORDIC

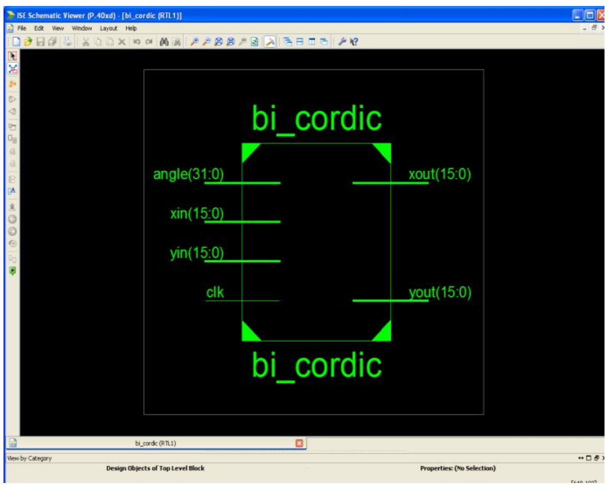


For Reducing Errors in Quadrant, angle and Gain correction Bi rotational Cordic is one of the techniques with extra hard ware implementation predefined gain and Quadrant correctional algorithms these are implemented form the basic cordic theory for the more accuracy in gain .The angle range should be -90 to +90 and all angles should be in the rage of first and fourth Quadrant of rotational implementation. These blocks improves the cordic fixe4d rotation with more accuracy .The accuracy can also be improved by the more no of shifting stages. also.

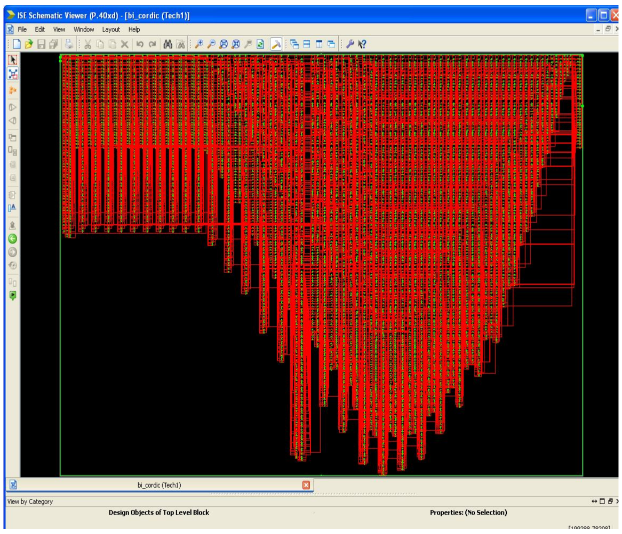
COMPLEXITY CONSIDERATIONS

We discuss here the hardware and time complexity of the proposed design. In the presented text we do not find similar work on CORDIC realization of known and fixed rotations. Therefore, we compare the proposed design with the conventional CORDIC design for the rotation of unknown angle. We have used the basic CORDIC processor in [3, Fig. 2] for the realization of conventional CORDIC. In addition, we have designed a reference architecture (see Fig. 1) for straight forward implementation of fixed rotations, and we have compared the complexities and speed performance of the proposed design with the conventional and reference design.

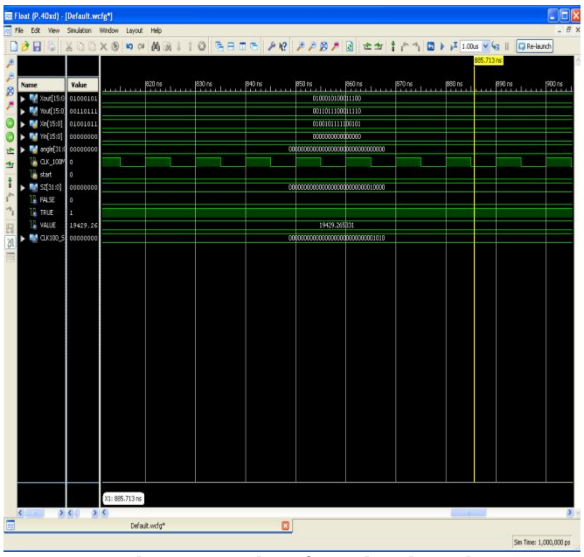
V. RESULTS AND ANALYSIS


ISE Schematic Viewer (P.40xd) - [bi_cordic (RTL1)]
File Edit View Window Layout Help
bi_cordic
angle(31:0)
xin(15:0)
yin(15:0)
clk
xout(15:0)
yout(15:0)
bi_cordic
bi_cordic (RTL1)
View by Category
Design Objects of Top Level Block
Properties: (No Selection)

RTL Schematic of Cordic algorithm


ISE Schematic Viewer (P.40xd) - [bi_cordic (Tech1)]
File Edit View Window Layout Help
bi_cordic (Tech1)
View by Category
Design Objects of Top Level Block
Properties: (No Selection)

Technology Schematic of Cordic algorithm


Float (P.40xd) - [Default.wcfg*]
File Edit View Simulation Window Layout Help
Re-launch
885.713 ns
X1: 885.713 ns
Default.wcfg*
Sim Time: 1,000,000 ps

```
Name          Value
Xout[15:0]    01000101    0100010100011100
Yout[15:0]    00110111    0011011100011110
Xin[15:0]     01001011    0100101111100101
Yin[15:0]     00000000    0000000000000000
angle[31:0]   00000000    00000000000000000000000000000000
CLK_100M      0
start         0
SZ[31:0]      00000000    00000000000000000000000000010000
FALSE         0
TRUE          1
VALUE         19429.26    19429.265031
CLK100_S      00000000    00000000000000000000000000001010
```

Simulation Results of Cordic algorithm

VI. CONCLUSION

In this paper Control on BI CORDIC is implemented with Angle correction and Quadrant corrections .Accuracy can be further increased by these CORDIC algorithm techniques. Overshoot problem in original CORDIC has been overcome by angle selection scheme. Due to the increasing of no. of bits complexity of hardware also increases, which results in accurate simulation results. .

REFERENCES

[1] Volder J. E,"The CORDIC trigonometric computing technique", IRE Trans. Electronic Computing, Volume EC-8, pp 330 - 334, 1959.
[2] Terence K. Rodrigues and Eari E.Swartzlander ,Jr.Fellow,"Adaptive CORDIC :Using parallel Angle Recoding to Accelerate Rotations", IEEE Transactions on Computers, Vol .59,NO.4, April 2010
[3] Pramod K.Meher, Senior Member ,IEEE, Javier Valls, Member, IEEE, Tso- Bing Juang , Member, IEEE, K.Sridhan, Senior Member, IEEE, and Koushik Maharatna, Member , IEEE " 50 Years of CORDIC: Algorithms, Architectures, and Applications" , IEEE transactions on circuits and systems,VOL.56.NO.9,September, 2009.

ISSN (Print) : 2320 – 3765
ISSN (Online): 2278 – 8875

International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering
(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 11, November 2014

[4] Alan Sultan, " CORDIC: How Hand calculators Calculate" Integre Technical Publishing Co.,Inc. college Mathematics journal VOL.40 NO.2 , March 2009.
[5] Yu Hen Hu "CORDIC-Based VLSI Architectures for Digital signal processing", IEEE Signal processing-1053-5888/92/$3.00, 1992.
[6] Dr. Dobbs , Michael Pascale, "Using CORDIC methods for Computations in micro-controllers" ,1 September 2000.
[7] Y.H .Hu, "CORDIC based VLSI architecture for digital signal processing",IEEE signal processing Mag., pp 16-35, july 1992.
[8] Kia Bazargan , "CORDIC Algorithms" EE-5324-VLSI Design 2, University of Minnesto,Spring 2006.
[9] Tanya Vladimirova and Hans Tiggler,"FPGA implementation of sine and cosine generators Using the CORDIC Algorithm" surrey space centre ,University of Surrey , Guildford , Surrey , GU2 5XH.
[10] Eric Keller,"Dynamic circuit specialization of CORDIC processor", Nov 7, 2000.
[11] Quiang Gao, "Low complexity Arithmetic Implementation for DSP Radio Receiver" university of strathcyclde.(qiang.gao@eee.strath.ac.uk).
[12] Jean Duprat and Jean – Michel Muller " The CORDIC Algorithm :New Results for Fast VLSI implementation" IEEE Transactions on computers ,Vol 42.No.2,Feb 1993.