

**Military Technical College
Kobry El-Kobbah,
Cairo, Egypt**



**6th International Conference
on Electrical Engineering
ICEENG 2008**

A novel design and implementation of FPGA based 3D-CORDIC processor

By

Al-Homosy , G.M. * Abass, Y.M. ** Al-Kholy, S.A. * A.M.Rashed***

Abstract:

A new complete design and implementation of FPGA-based Three Dimensions CORDIC processor (3D-CORDIC) is introduced. Efficient mappings on FPGA have been performed leading to the fastest implementations. Simulation process have been performed for the proposed 3D-CORDIC processor using ModelSim SE tools of Mentor Graphics simulations and the MATLAB Software simulations, a good agreement of the proposed processor performance has been achieved. The 3D-CORDIC processor architecture has been implemented with 12 bit word-length in Xilinx Spartan-II series field programmable gates arrays (FPGA). The 3D-CORDIC processor use only 37 % of SLICES and 52 % of IOBs with maximum clock frequency 116 MHz, which is suitable for many CORDIC processor applications.

Keywords:

FPGA, CORDIC, VHDL, 3D-CORDIC PROCESSOR

* Physics & Mathematics Dept, Suez Canal University

** Physics Dept, Suez Canal University

*** Systems & Computers Dept, Al-Azhar University

1. Introduction:

The well known CORDIC algorithm, which has been applied with a great success to the hardware implementations of many signal processing tasks, e.g. sine and cosine generation, vector rotation, coordinate transformation, and linear system solving, is suitable for the implementation of 3-D vector interpolation [1]. In CORDIC, only simple shifters and adders are needed, which can be realized by the use of reconfigurable hardware platforms, especially by FPGA [2].

While hardware-efficient solutions often exist, the dominance of the software systems has kept those solutions out of the spotlight. Among these hardware-efficient algorithms is a class of iterative solutions for trigonometric and other transcendental functions. The trigonometric functions are based on vector rotations, while other functions such as square root are implemented using an incremental expression of the desired function. The trigonometric algorithm is called CORDIC, an acronym for (Coordinate Rotation Digital Computer). The incremental functions are performed with a very simple extension to the hardware architecture, and while not CORDIC in the strict sense, are often included because of the close similarity. The CORDIC algorithms generally produce one additional bit of accuracy for each iteration.

Volder [3] 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. Volder's algorithm is derived from the general equations for the step by step vector rotation. All the evaluation procedures in CORDIC are computed as a rotation of a vector in three different coordinates systems (circular coordinate, linear coordinate and hyperbolic coordinate) with an iterative unified formulation.

The CORDIC method was later expanded for multiplication, division, logarithm, exponential and hyperbolic functions. The various function computations were summarized into a unified technique in Walter [4]. The multiplications in CORDIC calculations are replaced by calculation based on shift registers and adders, 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). A broad field of digital signal processing applications find a valuable kernel in the generalized CORDIC algorithm proposed by Walter.

The CORDIC-algorithm provides an iterative method of performing vector rotations by arbitrary angles using only shift and add operations. The CORDIC arithmetic technique makes it possible to perform two dimension rotations using simple hardware components. Its use was originally intended for solving navigation problems, by calculating some elementary functions [5] and as a useful kernel for many digital signal

processing tasks [6] [7].

Many researches have been developed for the overall quantization error (OQE) and the optimization of a CORDIC processor [8] [9]. Most of today's efforts at direct compilation from a high-level language to FPGAs target very simple arithmetic units such as adders, multipliers, shifters, etc. Instead, more complex arithmetic units such as CORDICs coupled with various alternatives of number representations should be targeted by higher-level compilers to exploit the full potential of reconfigurable computing [8] [10].

More specifically, there are a number of digital signal processing (DSP) applications using the CORDIC-based hardware such as modulation [11], digital filtering [12], and Fast Fourier Transforms (FFTs) [13] [14]. Modified CORDIC algorithms have also been proposed in order to overcome the disadvantages of the conventional algorithm such as low computational speed and high complexity [15] [16] [17].

In this paper, the architecture of 3D-CORDIC Processor based on the CORDIC algorithm is proposed. It is suitable for Field Programmable Gate Arrays (FPGA) implementation in terms of the computational complexity. One objective of this paper is to show a possible direction for high-level compilation to 3D-CORDIC Processor. The remainder of the paper is organized as follows: The basics of CORDIC algorithm technique are discussed in Section 2. The new 3D-CORDIC Processor technique is introduced in Section 3. Design, Simulation and implementation of 3D-CORDIC processor are introduced in section 4. The MATLAB Ver. 7 is used to simulate 3D-CORDIC processor is given in section 5, Finally, discussion and conclusion are given in section 6.

2. THE BASICS OF CORDIC ALGORITHM:

The CORDIC is an algorithm performing a sequence of iteration computations by the use of coordinate rotation [3] [4] [18]. It can be used to generate important elementary functions by using only simple adders and shifters.

The basic CORDIC iteration equations are given by:

$$X_{i+1} = X_i - m d_i (2^{-s(m,i)}) Y_i \quad (2.1)$$

$$Y_{i+1} = Y_i + d_i (2^{-s(m,i)}) X_i \quad (2.2)$$

$$Z_{i+1} = Z_i + d_i \alpha_{m,i} \quad (2.3)$$

where m denotes the circular ($m = 1$), linear ($m = 0$) or hyperbolic ($m = -1$) coordinate system [19] [20], $i = 0, 1, 2, \dots, n-1$,

$$s(m,i) = \begin{cases} 0, 1, 2, 3, 4, 5, \dots, & m = 1 \\ 1, 2, 3, 4, 5, 6, \dots, & m = 0 \\ 1, 2, 3, 4, 5, 6, \dots, & m = -1 \end{cases}$$

$$\text{and } \alpha_{m,i} = m^{-1/2} \tan^{-1}[\sqrt{m} 2^{-s(m,i)}] \quad (2.4)$$

The rotation $d_i = \text{sign}(Z_i)$ for the rotation mode ($Z_n \rightarrow 0$);
 $d_i = -\text{sign}(X_i) \cdot \text{sign}(Y_i)$ for the vectoring mode ($Y_n \rightarrow 0$).

To explain the basic concept of CORDIC, consider a Two-Dimensional Euclidean Space in figure (2.1). Let X_i and Y_i be the X and Y coordinates of the vector OP with magnitude R_i . This vector is rotated through an angle ϕ to form the new vector OQ. The rotation is not a pure vector rotation but a motion of vector OP along the tangent of the circle formed by OP as radius at the point P. Then the resultant vector will have a magnitude given by $R_i \sqrt{1 + \tan^2 \phi}$ which is $R_i \sec \phi$. The coordinates of Q (X_{i+1}, Y_{i+1}) can be expressed as follows [21]:

$$\begin{aligned} X_{i+1} &= X_i - Y_i \tan(\phi) \\ Y_{i+1} &= Y_i + X_i \tan(\phi) \end{aligned} \quad (2.5)$$

Taking into consideration the direction of rotation (s), $s=+1$ for anticlockwise rotation and $s=-1$ for clockwise rotation, the Equation (2.5) can be expressed as

$$\begin{aligned} X_{i+1} &= X_i - s_i Y_i \tan(\phi) \\ Y_{i+1} &= Y_i + s_i X_i \tan(\phi) \end{aligned} \quad (2.6)$$

In CORDIC algorithm the angle of rotation is achieved by a series of micro-rotation. In simple words the input angle is decomposed into small micro-angles that take values of $\tan^{-1}(2^{-i})$, where i take values from 0 to n .

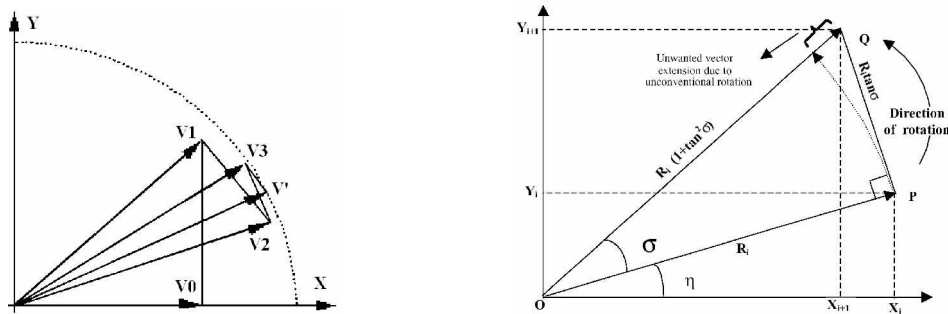


Figure (2.1) Iterative vector rotation, initialized with V_0

The scale factor $\{ k_{m,i} = \cos(\tan^{-1}(\phi)) = 1 / (1 + m d_i^2 2^{-2s(m,i)})^{1/2} \}$ in the i -th iteration, as shown in figure(2.1). After n iterations, the product of all the scale factors is as follows [22].

$$K_n = \prod_{i=0}^n k_i = \prod_{i=0}^n 1/(1 + m d_i^2 2^{-2s(m,i)})^{1/2} = \prod_{i=0}^n 1/(1 + m 2^{-2s(m,i)})^{1/2}$$

where the rotation direction is defined by $d_i = \{-1, +1\}$ and the scale factor is:

$$K_n \rightarrow 1.6476 \text{ as } n \rightarrow \infty \text{ or } 1/K_n \rightarrow 0.607 \text{ as } n \rightarrow \infty$$

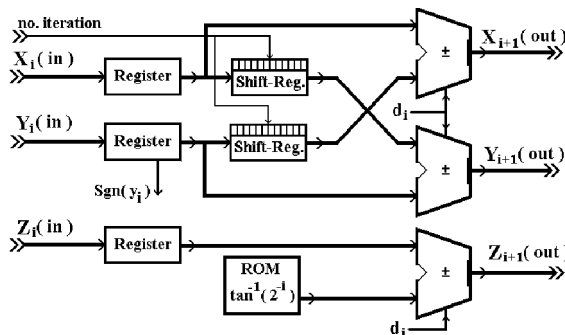
For data of **B**-bits wordlength, no more than **B** iterations need be performed, i.e., $n \leq B$. In addition, the final values, $X(n)$ and $Y(n)$, need to be scaled by an accumulated scaling factor K_n . The resulting vector $V'(X(n)/K, Y(n)/K)$ is the unit vector as shows in figure(2.1).

3. A New 3D-CORDIC Processor Based on 2D-CORDIC Design:

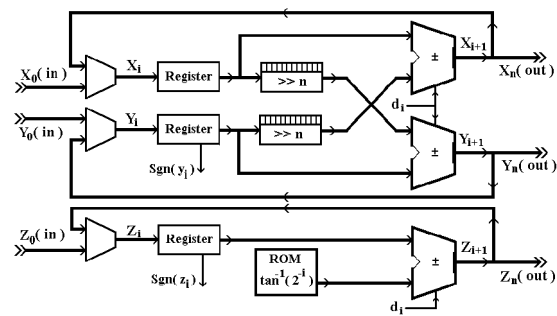
There are a different techniques to implement a CORDIC processor [8]. The ideal architecture depends on the speed versus area tradeoffs in the intended application. In the a new design of the CORDIC algorithm the 3D Cartesian Coordinate (X_o, Y_o, Z_o) are converted to Polar Coordinate ($R_{final}, \alpha, \gamma$), as shows in figure(3.1).

The CORDIC structure as described in equations (2.1, 2.2 and 2.3) is represented by the schematics as shown in figure(3.2), An iterative CORDIC architecture can be obtained simply by duplicating each of the three difference equations in hardware as shown in figure(3.2). The decision function, σ_i , is driven by the sign of the y or z register depending on whether it is operated in rotation mode or vectoring mode. In operation, the initial values are loaded via multiplexers into the x, y and z registers. Then on each of the next n clock cycles, the values from the registers are passed through the shifters and adder-subtractors and the results placed back in the registers. The shifters are modified on each iteration n to cause the desired shift for the iteration [19] [23].

The ROM address is incremented on any iteration so that the appropriate elementary angle value is presented to the z Adder. On the last iteration, the results are obtained directly from the Adder. Obviously, a simple state machine is required to keep track of the current iteration, and to select the degree of shift and ROM address for any iteration [24].



One Iterative CORDIC structure



N-Iterative CORDIC algorithm structure

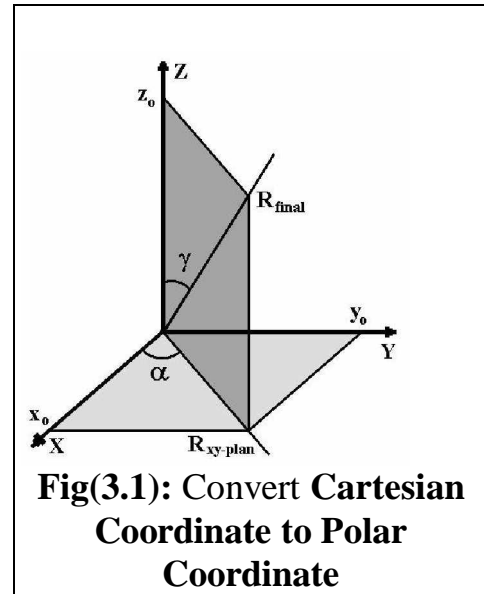
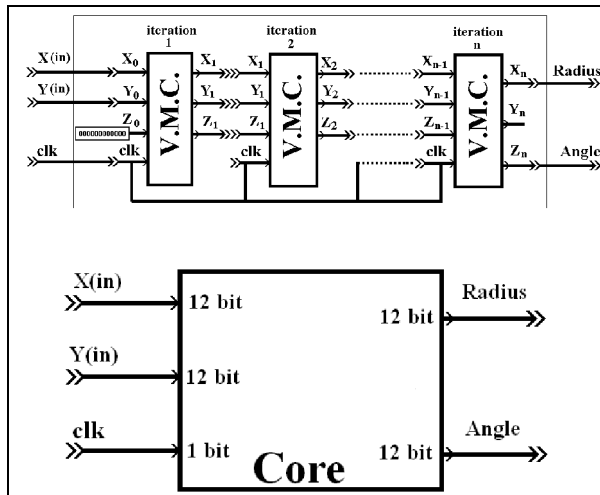


Figure (3.2) : Block Diagram of a Parallel CORDIC Architecture.

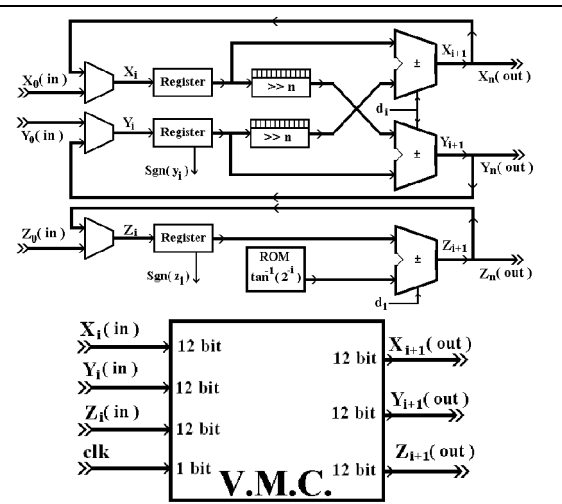
To determine the location of any point in XY-plane as (x_o, y_o) , reading (x_o, y_o) in X-Y plane then we must be able to specify the location of sign (x_o) and sign (y_o) then recording them in a register 2-bit which called Select register (SEL(2bit)). Table (3.1) shows the Select register related to different of sign (x_o) and sign (y_o) .

The Vector Mode CORDIC unite (VMC) as shown in figure (3.3), VMC unite are used to calculate the $x_{n(x-y \text{ plane})}$ and $z_{n(x-y \text{ plane})}$ by input positive value of the x_o and y_o to register $x_o(\text{in})$, $y_o(\text{in})$ respectively. In the new design the number of iterations $n = 12$ and the CORE shown in figure(3.4), which includes contents of 11 iterative of VMC unit .

Table(3.1) The Select Register related to different of sign(x_o) and sign(y_o)		
sign(x_o)	sign(y_o)	SEL(2 bit)
+ve	+ve	00
- ve	+ve	01
- ve	- ve	11
+ve	- ve	10



Figure(3.4) Vector mode of n=11 Iterative CORDIC Structure



Figure(3.3) Vector mode of one Iterative CORDIC Structure

The output values ($X_{n(x-y-plane)}$, $Z_{n(angel)}$) in the new design are modified and accurate as follows [19] :

1- CORE by correcting value of $x_{n(x-y-plane)}$ by using equation:

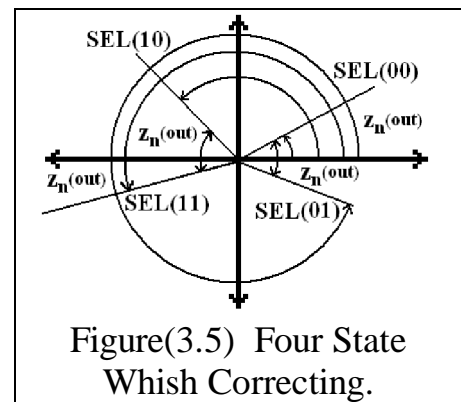
$R_{(x-y-plane)} = X_{n(x-y-plane)}/K$ Where $R_{(x-y-plane)}$ is radius of XY-plan, as shown in figure (3.1). The K factor is constant ($K \approx 1.6468$), the value of $1/K$ is divided to different values as shown in equation(3.1) :

$$\frac{1}{K} = \frac{1}{1.6468} = \frac{1}{2^1} + \frac{1}{2^3} - \frac{1}{2^6} - \frac{1}{2^9} \quad (3.1)$$

$$\therefore R_{(x-y-plan)} = \frac{X_{(x-y-plan)}}{K} = X_{(x-y-plan)} \left(\left(\frac{1}{2^1} + \frac{1}{2^3} \right) - \left(\frac{1}{2^6} + \frac{1}{2^9} \right) \right) \quad (3.2)$$

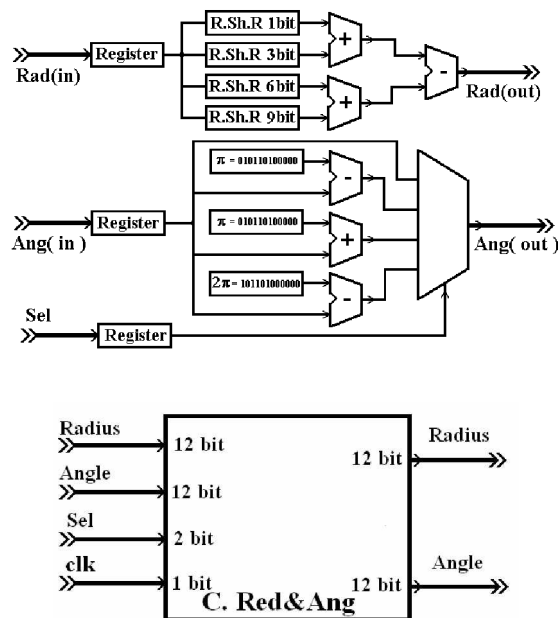
Now, $R_{(x-y-plane)}$ value is corrected by using right shift register process by 1-bit, 3-bit, 6 bit and 9-bit from $x_{n(x-y-plane)}$ according to equation(3.2). After that, adding and subtractions processes is shown equation (3.2).

2- A correction in the angle value (Alfa) become possible according to it position in xy-plane, if the x, y are positive together (SET = 00) the angle will be equal to $z_n(out)$, if the SET = 10 the angle will be equal to $(\pi - z_n(out))$, but if the select (SET = 11) the value of the angle will be equal to $(\pi + z_n(out))$ and finally if the select (SET = 01) the angle is equal to $(2\pi - z_n(out))$, as shown in figure(3.5).

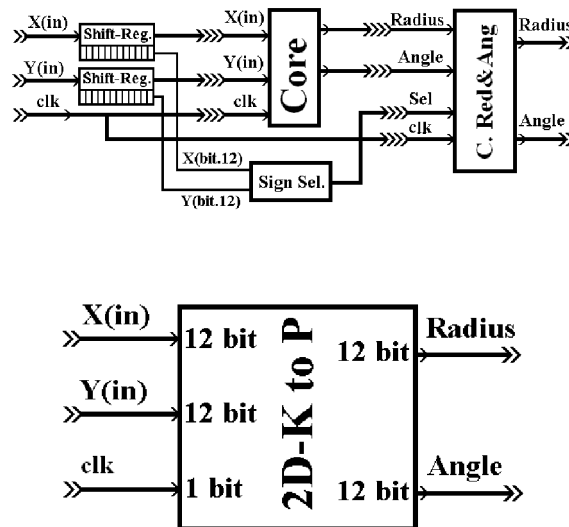


Figure(3.5) Four State Whish Correcting.

The correction process was made by block diagram of Correct Radius and Angle unit (C-Rad&Ang) as shown in figure (3.6). Figure (3.7) shows the block diagram of Two Dimensions Cartesian to Polar coordinate (2D-KtoP) or (2D CORDIC processor), which converts from two dimensional Cartesian coordinate to polar coordinate. From 2D-KtoP block diagram, radius $R_{(x-y-plane)}$ and angle α can be calculated [19].



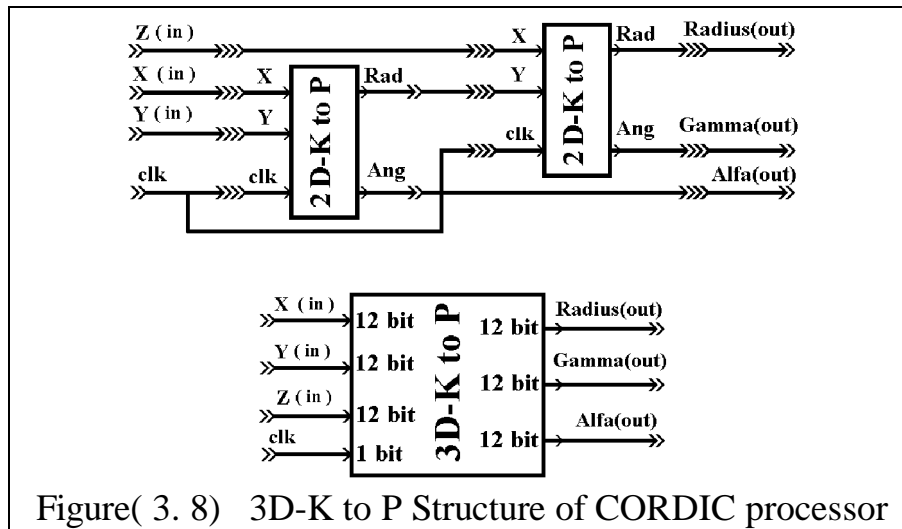
Figure(3.6) : Correct Radius and Angle Structure of CORDIC processor



Figure(3.7) : 2D-KtoP Structure of CORDIC processor

The radius $R_{(xy-plane)}$ and angle Alfa (α) can be calculated in the XY-plan, by using 2D CORDIC processor (2D-KtoP), at $X_{in}=X_o$ and $Y_{in}= Y_o$. And in $ZR_{(xy-plane)}$ -plan at $X_{in}=Z_o$ and $Y_{in}= R_{(xy-plane)}$, the radius R_{final} and angle Gamma (γ) is calculated, as shows in figure(3.2).

Figure (3.8) shows the block diagram of the Three Dimensions Cartesian to Polar coordinate (3D-KtoP) or (3D CORDIC processor).



4. FPGA IMPLEMENTAION AND RESULTS:

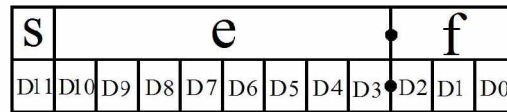
The VHDL (VHSIC hardware Description Language) is used to implement the algorithm and map it to FPGA [25]. The probability values are floating point numbers so we used the IEEE 754 standard to represent it.

The IEEE 754 standard represents the floating point numbers by three fields (sign (s), exponent (e), and mantissa (F)), Figure (4.1) shows the bit width alignment of the three fields for precession representation.

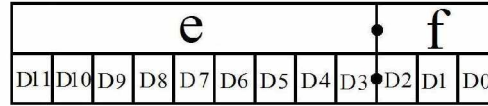
Sometimes much smaller bit widths than those specified in the IEEE 754 standard are sufficient to provide desired precision and occupy less resources than the full standard bit width implementation [26][27].

The Cartesian coordinates of a point in a 3-dimensional space (X_k , Y_k , Z_k) are contains 12-bit signed words. The 3D-CORDIC processor unit returns this point to the equivalent Polar coordinates (R , $\text{Alfa}(\alpha)$, $\text{Gamma}(\gamma)$), where red_p(out) is the radius (R), an_alfat_p(out) is the angle $\text{Alfa}(\alpha)$, an_gamma_p(out) is the angle $\text{Gamma}(\gamma)$.

As illustrated in figure (4.1), the sign field (S) is bit number D11 and is used to specify the sign of the number. Bits D10 down to D3 are the exponent field, this 8-bit quantity is a signed number represented by using integer value. Bits D2 down to D0 are used to store the binary representation of the floating point number. But in the angle values, as illustrated in figure(4.2). Not sign filed, bits D11 down to D3 are the exponent filed, and this 9-bit quantity is signed number represented by using integer value.



Figure(4.1): 12bit floating point format at dimensions value (X_{in} , Y_{in} , Z_{in} , Radius)



Figure(4.2): 12bit floating point format at angles value (Alfa, Gamma)

In the process of building the 3D-CORDIC processor, the 12-bits of word-length are used, the last bit referring to the sign bit at the numerical values, if the bit equal 0, the number is positive, or if the bit equal 1, the number is negative. The inputs and the output radius (x_{in} , y_{in} , z_{in} , $rad_p(out)$) are in following format as shown in figure (4.1) [19]:

The last bit referring to the sign bit (s)

The upper 8-bits represent the decimal value (e)

The lower 3-bits represent the fractional value (f)

While the output angles (Alfa and Gamma) represented in the following format as shown in figure (4.2):

The upper 9-bits represent the decimal value (e)

The lower 3-bits represent the fractional value (f)

No sign bit of angles (Alfa and Gamma), because the values of any angle are having positive value. According to the above angles (Alfa and Gamma) format the binary values of π -angle is equal to (010110100.000) and of 2π -angle is equal to (101101000.000).

The implementation of 2D-CORDIC processor generated from Xilinx FPGAs by using Spartan-II 2.5V [XC2S200E-5-PQ208C]. The design was coded in VHDL and successfully implementation in Xilinx Spartan-II FPGA.

The VHDL code of proposed 3D-CORDIC is compiled by XILINX Foundation Series Express (ISE 4.2i) and simulated using ModelSim SE tools of Mentor Graphics [28], [29]. This model needs 1700 ns for processing the input values and presenting a result, this is a very fast implementation.

A simulation and results of the 3D-CORDIC processor with the inputs of table (4.1) is shown in figure (4.3).

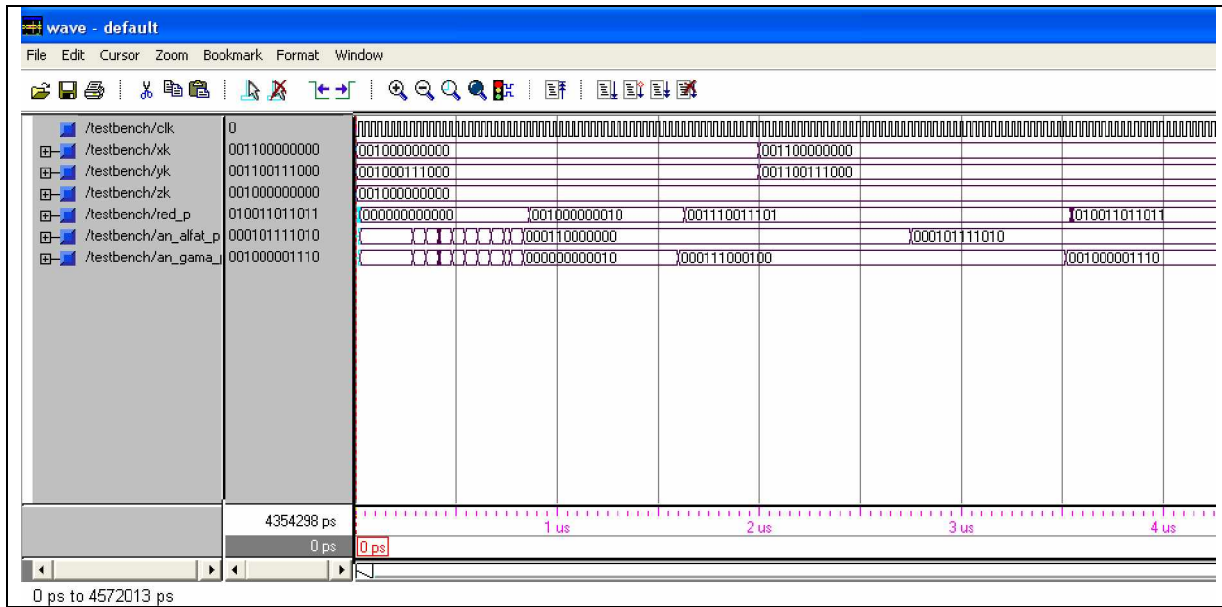


Figure (4.3) : Shows the Simulation and Results for 3D-CORDIC Processor

For the inputs operands:

$X_{in} = (001000000.000)_B = (+64.00)_D$, $Y_{in} = (001000111.000)_B = (+71.00)_D$, $Z_{in} = (001000000.000)_B = (+64.00)_D$ at time 0 sec, and the outputs is radius $= (001110011.101)_B = (+115.625)_D$, Alfa $= (000110000.000)_B = (48.00)_D$, Gamma $= (000111000.100)_B = (56.50)_D$, (in binary signed digit BSD from). At the time 1.7 μ sec, the input is $X_{in} = (001100000.000)_B = (+96.00)_D$, $Y_{in} = (001100111.000)_B = (+103.00)_D$, $Z_{in} = (001000000.000)_B = (+64.00)_D$ at time 2 μ sec, and the output is radius $= (010011011.011)_B = (+155.375)_D$, Alfa $= (000101111.010)_B = (47.25)_D$, Gamma $= (001000001.110)_B = (65.75)_D$ at the time 3.6 μ sec.

Table(4.1): Shows the Results for 3D-CORDIC Processor	
Input values	Output values
$X_{in} = (001000000.000)_B = (+64.00)_D$ $Y_{in} = (001000111.000)_B = (+71.00)_D$ $Z_{in} = (001000000.000)_B = (+64.00)_D$ wait after 2 μ s	Radius (R) $= (001110011.101)_B = (+115.625)_D$ Alfa (α) $= (000110000.000)_B = (48.00)_D$ Gamma (γ) $= (000111000.100)_B = (56.500)_D$
$X_{in} = (101000000.000)_B = (-64.00)_D$ $Y_{in} = (001000111.000)_B = (+71.00)_D$ $Z_{in} = (001000000.000)_B = (+64.00)_D$	Radius (R) $= (010011011.011)_B = (+155.375)_D$ Alfa (α) $= (000101111.010)_B = (47.25)_D$ Gamma (γ) $= (001000001.110)_B = (65.75)_D$

The experimental results and simulation are given in figure (4.3) and figure (4.4), which represent the simulation for 3D-CORDIC processor. Table (4.2) gives the summarized results of 3D-CORDIC Processor where the input is the step Clock (clk) and the three 12-bit signed as signed inputs (Xk, Yk, Zk) and outputs (rad_p, an_alfa_p, an_gama_p).

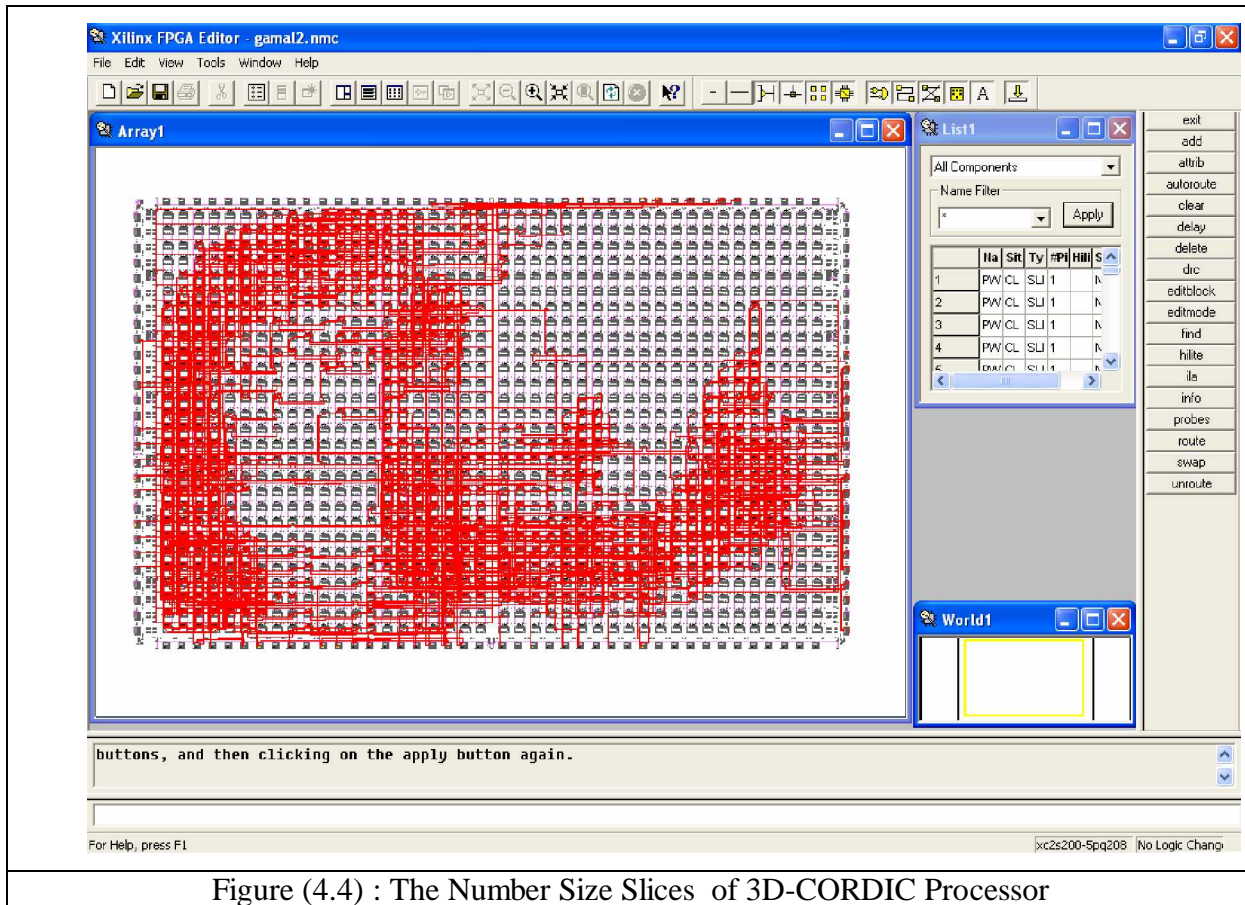


Table (4.3) and Table (4.4) summarizes the device utilization for Spartan-II (XC2S200E-5-PQ208C) FPGA with a speed grade of -5. This is shown synthesis results of 3D-CORDIC processor are found to running at a frequency of (116.618 MHz) with a total equivalent gate count of (21,626 gates).

Timing Summary:

Minimum period: 8.575ns (Maximum Frequency: 116.618MHz)

Minimum input arrival time before clock: 13.193ns

Maximum output required time after clock: 7.999ns

Maximum combinational path delay: No path found

Downloading and implementation our design gives the following: Device utilization summary:

Number of Resource	Used	Avail	Utilization
External GCLKIOBs	1	4	25%
External IOBs	73	140	52%
LOCed External IOBs	73	73	100%
SLICES	872	2352	37%
GCLKs	1	4	25%

Table(4.2). List of IO Ports for 3D-cordic processor			
Port	Width	Direction	Description
CLK	1	Input	System Clock
Xk	12	Input	X-coordinate input. Signed value
Yk	12	Input	Y-coordinate input. Signed value
Zk	12	input	Z-coordinate input. Signed value
red_p	12	Output	Radius output. Unsigned value.
an_alfat_p	12	Output	Angle (α) output. Unsigned value.
an_gama_p	12	Output	Angle (γ) output. Unsigned value.

Table(4.3) : Synthesis Results for 3D-CORDIC Processor				
Vendor	Family	Device	Resource usage	Max. Clock speed
Xilinx	Spartan-II	XC2S200E-5	872 slices	116.618 MHz

Design Summary

Number of Slices:	872 out of 2,352	37%
Number of Slices containing unrelated logic:	0 out of 872	0%
Number of Slice Flip Flops:	1,549 out of 4,704	32%
Total Number 4 input LUTs:	851 out of 4,704	18%
Number used as LUTs:	782	
Number used as a route-thru:	69	
Number of bonded IOBs:	72 out of 140	51%
Number of GCLKs:	1 out of 4	25%
Number of GCLKIOBs:	1 out of 4	25%
Total equivalent gate count for design:	21,626	
Additional JTAG gate count for IOBs:	3,504	

Table (4.4) : Synthesis Results of Xilinx XC2S200E-5-PQ208C FPGA			
Resource	Used	Available	Utilization
IOs	73	140-208	52.21%
Function generator 4 input LUTs	851	4704	18.09%
CLB Slice	872	2352	37.07%
Dffs or Latches (Flip Flops)	1549	4704	32.92%

The 3D-cordic processor and reconfigurable hardware implementation on FPGA are two important and promising technologies to scientific researches. In this paper we propose a 3D-CORDIC processor implemented using FPGA. The 3D-CORDIC processor is implemented on Xilinx Spartan-II ship (XC2S200E-5-PQ208C). A good agreement has been observed between the FPGA and MATLAB simulation. The 3D-CORDIC processor used in applications of the aconite position sewing of Robot Arm and efficient control in satellite and guidance technique. Also CORDIC implementation in VHDL has been used in angle calculation in wireless LAN receiver block. The results

show that, the 3D-CORDIC processor uses only 37 % of SLICES and 52% of IOBs which means that the system can be up grade to fit more user requirements without technology or even the FPGA ship. The maximum clock frequency is 116 MHz, Which is high sufficiently for many applications.

5. MATLAB SIMULATION:

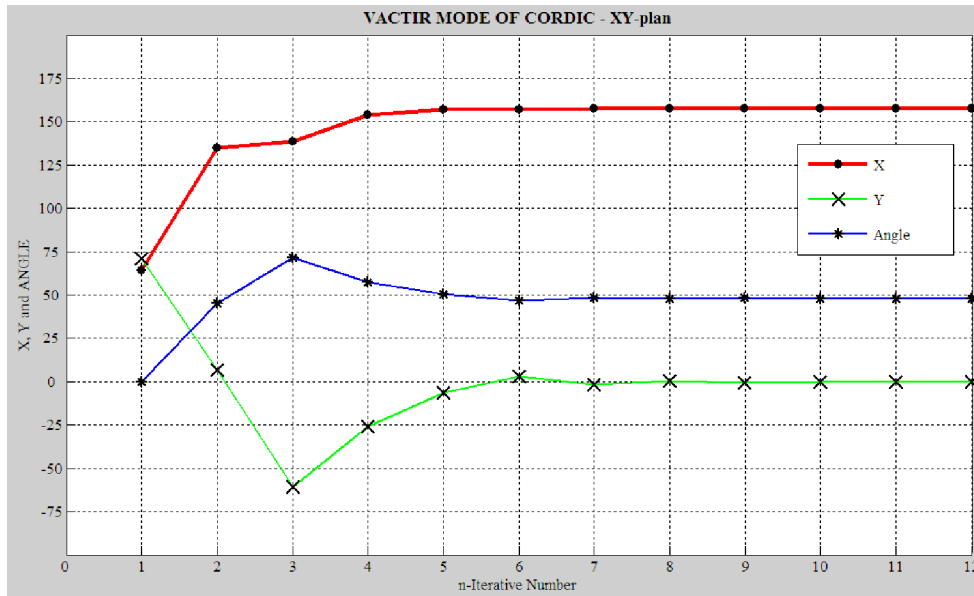
To compare FPGA simulation results with anther another simulation method, the MATLAB V.7 is used to simulate 3D-CORDIC processor [30].

The VHDL code in converted to a MATLAB code. The MATLAB simulation results of 3D-CORDIC processor, with the same inputs of FPGA foundation simulation are shown in table (5.1).

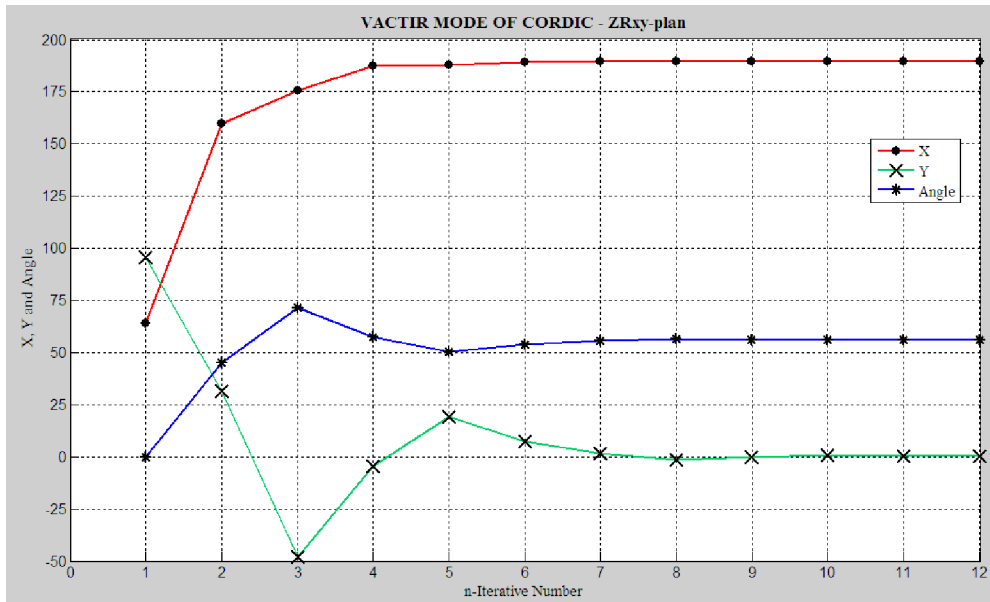
In figure(5.1) and figure(5.2) demonstrates the MATLAB simulation of X, Y, Z convergence after 12 iterations, for the first row value in table(5.1) for the 3D-CORDIC processor. The initial conditions in XY-plan are $x_1 = 64.000$, $y_1 = 71.000$ and $\text{Alfa}_1 = 0.000$, after 12 iterations, the registers contain: $x_{12} = 157.409$, $y_{12} = -0.092$, $\text{Alfa}_{12} = 48.002$, where the computed adjustment is $K \approx 1.646$. Thus, the final solutions are $\text{Alfa} = 48.002$ and Radius ($R_{\text{XY-plane}}$) = $x_{12} / K = 95.631$. The initial conditions in $ZR_{\text{XY-plane}}$ -plan are $x_1 = z_1 = 64.000$, $y_1 = R_{\text{XY-plane}} = 95.631$ and $\text{Gamma}_1 = 0.000$, after 12 iterations, the registers contain: $x_{12} = 189.495$, $y_{12} = 0.154$, $\text{Gamma}_{12} = 56.161$, the final solutions are $\text{Gamma} = 56.161$ and Radius (R_{final}) = $x_{12} / K = 115.125$.

The simulation by using MatLab v.7 of 3D-CORIC processor, the results of different values inputs are shown the table (5.1).

Table(5.1):	
Input values	Output values
$X_{\text{in}} = 64.000$	Radius (R) = 115.125
$Y_{\text{in}} = 71.000$	Angle (α) = 48.002
$Z_{\text{in}} = 64.000$	Gamma (γ) = 56.161
$X_{\text{in}} = 96.000$	Radius (R) = 154.795
$Y_{\text{in}} = 103.000$	Angle (α) = 46.994
$Z_{\text{in}} = 64.000$	Gamma (γ) = 65.601



Figure(5.1) : CORDIC Inverse Tangent Convergence at XY-Plan
 $\{ X(1) = X_{in}, Y(1) = Y_{in}, X(12) = K * R_{xy}, Y(12) \approx 0, Angle(12) = Alfa \}$



Figure(5.1) : CORDIC Inverse Tangent Convergence at ZR_{XY}-plan
 $\{ X(1) = Z_{in}, Y(1) = R_{xy}, X(12) = K * R_{final}, Y(12) \approx 0, Angle(12) = Gamma \}$

6. DISCUSSION AND CONCLUSION:

The 3D-cordic processor and reconfigurable hardware implementation on FPGA are two important and promising technologies to scientific researches. In this paper we propose a 3D-CORDIC processor implemented using FPGA. The 3D-CORDIC processor is implemented on Xilinx Spartan-II ship (XC2S200E-5-PQ208C). A good agreement has been observed between the FPGA and MATLAB simulation. The 3D-CORDIC processor used in applications of the aconite position sewing of Robot Arm and efficient control in satellite and guidance technique, and angle calculation in

wireless LAN receiver. The results show that, the 2D-CORDIC processor uses only 37 % of SLICES and 52% of IOBs which means that the system can be up grade to fit more user requirements without technology or even the FPGA ship. The maximum clock frequency is 116 MHz, Which is high sufficiently for many applications.

REFERENCE:

- [1] J. Euh, J. Chittamuru, W. Burson, "CORDIC Based Interpolator for 3-D Graphics," *IEEE Workshop on Signal Processing Systems*, 2002, pp.240-245.
- [2] O. Mencer, L. Semeria, M. Morf, J. Delosme, "Application of Reconfigurable CORDIC Architecture," *The Journal of VLSI Signal Processing, Special Issue on Reconfigurable Computing*, March 2000.
- [3] J. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Trans. Electronic Computing*, Vol. EC-8, Sept 1959, pp. 330-334.
- [4] J.S. Walther, "A unified algorithm for elementary functions," *Spring Joint Computer Conference Proceedings*, Vol.38, 1971, Proc., pp. 379-385.
- [5] R.J. Linhardt, H.S. Miller, "Digit-by-Digit Transcendental Function Computation," *RCA, Rev.* 30 ,1969, pp. 209-247.
- [6] J. Giacomantone, H. Villagarcia, O. Bria, "A fast CORDIC Co-Processor Architecture for Digital Signal Processing Applications," *VI Congreso Argentino de Ciencias de la Computación (VI CACIC)*, October 2000.
- [7] Y.H. Hu, T.Y. Sung, "Efficient Implementation of the Chirp Z-Transform using a CORDIC processor," *IEEE Trans. ASSP*, Vol. 38, No. 2, Feb. 1990, pp. 352-354.
- [8] Hu. Yu Hen, "The quantisation effects of the CORDIC algorithm", *IEEE Trans. Signal Process.* 40 (4) (1992).
- [9] Alexander, S.W., Pfann, E., Stewart, R.W., "An Improved Algorithm For Assessing The Overall Quantisation Error In FPGA Based CORDIC Systems Computing A Vector Magnitude", *Electronic Edition* (link) BibTeX Volume 31, Number 2, pp. 87-93, 17/2/2006, Available online at: <http://www.sciencedirect.com>.
- [10] O. Mencer, M. Morf, M.J. Flynn, "PAM-Blox: High Performance FPGA Design for Adaptive Computing," *IEEE Symp. on FPGAs for Custom Computing Machines*, Napa Valley, CA, 1998. <http://umunhum.stanford.edu/PAM-Blox/>.
- [11] J. Vankka, M. Kosunen, I. Sanchis, K. Halonen, "A multicarrier QAM modulator," *IEEE Trans. Circuits Syst. II*, vol. 47, pp. 1–10, Jan. 2000
- [12] J. Ma, K. K. Parhi, G. J. Hekstra, and E. F. Deprettere, "Efficient implementations of pipelined CORDIC based IIR digital filters using fast orthonormal μ -rotations," *IEEE Trans. Signal Processing*, vol. 48, pp. 2712–2716, Sept. 2000.
- [13] A. M. Despain, "Fourier transform computers using CORDIC iterations," *IEEE Trans. Comput.*, vol. C-23, pp. 993–1001, Oct. 1974.

- [14] Y. H. Hu, "CORDIC-based VLSI architectures for digital signal processing," IEEE Signal Processing Mag., vol. 9, pp. 16–35, July 1992.
- [15] C. S. Wu, A. Y. Wu, "Modified vector rotational CORDIC (MVRCORDIC) algorithm and architecture," IEEE Trans. Circuits Syst. II, vol. 48, pp. 548-561, June 2001.
- [16] A. Y. Wu, C. S. Wu, "A unified view for vector rotational CORDIC algorithms and architectures based on angle quantization approach," IEEE Trans. Circuits Syst. I, vol. 49, pp. 1442–1456, Oct. 2002.
- [17] E. Antelo, J. Villalba, J. D. Bruguera, E. L. Zapata, "High performance rotation architectures based on the radix-4 CORDIC algorithm," IEEE Trans. Comput., vol. 46, pp. 855–870, Aug. 1997.
- [18] T. Lang, E. Antelo, "High-Throughput CORDIC Based Geometry Operations for 3D Computer Graphics," IEEE Transactions on Computers, Vol. 54. No. 3, March 2005, pp. 347-361.
- [19] G.M. Al-Homasy , Y.M. Abass, S.A. Al-Kholy, A.M. Rashed, " Design and Implementation of FPGA Based 2D-cordic processor," accepted for publication in Port-Said Engineering Research Journal (PSERJ), ISSN 1110-6603, 2008.
- [20] Jeanette F. Arrigo, Paul M. Chau, "Power Aware Attitude Computation During Rapid Rotational Motion," IEEE Transactions on Instrumentation and Measurement , Vol. 55, No. 1, February 2006, pp.63-69.
- [21] Bimal Gisuthan, Thambipillai Srikanthan, "Pipelining flat CORDIC based trigonometric function generators," microelectronics Journal 33, accepted 21 August 2001, 2002, pp.77-89. www.elsevier.com/locate/mejo
- [22] B. Heyne and J. Götze, "Cordic based algorithms for software defined radio (SDR) baseband processing," Advances in Radio Science 4, pp.179–184, 2006, www.adv-radio-sci.net/4/179/2006/
- [23] Tso-Bing Juang, Shen-Fu Hsiao, Ming-Yu Tsai, "Para-CORDIC: Parallel CORDIC Rotation Algorithm," IEEE Transactions on Circuits and Systems I, Vol. 51, No. 8, pp. 1515-1524, Aug. 2004.
- [24] Shen-Fu Hsiao, Yu-Hen Hu, Tso-Bing Juang, "A Memory-Efficient and High-Speed Sine/Cosine Generator Based on Parallel CORDIC Rotations," IEEE Signal Processing Letters, Vol. 11, No. 2, February 2004, pp.152-155
- [25] F.A. Elmisery, A.H. Khalil, A.E. Salama, H.F. Hammed, "A FPGA-Based HMM For A Discrete Arabic Speech Recognition System," ICM, Dec. 9/11/2003, Cairo, Egypt, pp. 322-325.
- [26] Nabeel Shirazi, Al Walters, Peter Athanas, "Quantitative Analysis of Floating Point Arithmetic on FPGA Based Custom Computing Machines," Proceeding of the IEEE Symposium on FPGAs for custom Computing Machines, pp.155-162, 1995.

- [27] Sang Yoon Park, Nam Ik Cho, "Fixed-Point Error Analysis of CORDIC Processor Based on the Variance Propagation Formula," IEEE Transaction on Circuits and Systems - I: Regular Papers, Vol. 51, No. 3, March 2004, pp. 573-584
- [28] Mentor Graphics, "Design Exploration and ModelSim SE Tutorial," Mentor Graphics Corporation 2001.
- [29] Joseph Kurian, A. Manavalan, "Performance evaluation and speed improvement of the seams VHDL-ams simulator using dynamic adjustment of the analog simulation interval," Master's thesis, University of Cincinnati, OH, 1998.
- [30] J. Ou, V.K. Prasanna, "PyGen: A MATLAB/Simulink based Tool for Synthesizing Parameterized and Energy Efficient Designs Using FPGAs," IEEE Inter. Symp. on Field-Programmable Custom Computing Machines (FCCM), 2004.
- [31] H.S. Kebbat, J.Ph. Blonde, F. Braun, "A new semi-flat architecture for high speed and reduced area CORDIC chip," Microelectronics Journal 37, (2006), pp. 181–187, www.elsevier.com/locate/mejo
- [32] Si Mahmoud Karabernou, Lounis Kessal, Fayçal Terranti, "Real-time FPGA implementation of Hough Transform using gradient and CORDIC algorithm," Image and Vision Computing, accepted 1 July 2005, Volume 25, Issue 6, 1 June 2007, Page 1032.