Sine/Cosine Generator Using Pipelined CORDIC Processor

R. Ranga Teja, P. Sudhakara Reddy, *IEEE, Member*

***Abstract* — At present, many of the computations in signal processing and wireless communication applications are linked with complex analysis of several functions. These complex functions are combination of sine and cosine terms that generally spread in the channel. Most of these functions can be split into elementary functions. In this paper we present a hardware efficient architecture by using CORDIC algorithm for the calculation of sine and cosine functions. This approach is simulated using ModelSim simulation software, synthesized using Xilinx ISE design suite and the proposed architecture is implemented on Xilinx FPGA target device i.e. SPARTAN 3E. Finally, the device utilization summary and timing reports are presented.**

***Index Terms*—CORDIC, FPGA, Pipelined processor, sine and cosine generator.**

I. Introduction

The solutions for the design of high speed VLSI architectures for real-time digital signal processing (DSP) algorithms are mapped from algorithm into hardware efficient architectures. With the advent of low cost, low power FPGA's; design of such architectures which would satisfy the performance requirements for the signal processing applications such as Three dimensional (3D) graphics, video/image/ signal processing systems has become simple.

Many of the DSP algorithms uses the calculation of elementary functions such as trigonometric, inverse trigonometric, logarithm, exponential, multiplication, and division functions which require high computational power. The commonly used software solutions for the digital implementation of these functions are table lookup method and polynomial expansions, requiring number of multiplication and additions/subtractions.

In 1959, Volder [1] has proposed a special purpose digital computing unit known as COordinate Rotation DIgital Computer (CORDIC). This algorithm was initially developed for trigonometric functions, using Givens rotation transform technique. The CORDIC algorithm computes 2D rotation using iterative equations employing shift and add operations which have simple architecture and consume less power. Walther has proposed a unified algorithm to compute rotation in circular, linear, and hyperbolic coordinate systems [2]. The CORDIC algorithm performs various elementary functions possible in rotation and vectoring mode of circular, linear, and hyperbolic coordinate systems [3][4]. CORDIC technique has been used in many applications, such as signal processing, linear transformations, digital filters and matrix based computations. Ultra low power systems can be efficiently developed by CORDIC [5][6]. More recently, the advances in the VLSI technology and the advent of EDA tools have extended the application of CORDIC algorithm to the field of biomedical signal processing, neural networks, software defined radio, and MIMO systems etc [7][8].

Manuscript received May 11, 2011.

R.Ranga Teja, Student, Dept.of ECE, Srikalahasteeswara Institute of Technology, Srikalahasthi, Andhra Pradesh, India. Ph:09985259705, E-mail:rangateja@gmail.com.

Dr.P.Sudhakara Reddy, Associate Professor, Dept.of ECE, Srikalahasteeswara Institute of Technology, Srikalahasthi, Andhra Pradesh, India. Ph:09703415601, E-mail:psr_vlsi_dsp@rediffmail.com.

II. Problem Statement and Definition

There is a real need of hardware efficient algorithms in the present generation of technologies because of the intense signal processing requirements needed by them. Thus, the current trend is back toward hardware efficient algorithms. Among all those unveiled, shift-and-add architectures commonly known as CORDIC have wide range of attractive features that can compute almost all elementary functions with simple architecture and further study on such algorithm could give more interesting results which can be best suited for all the present world applications. Many of those applications require elementary calculations like sin and cosine. Hence, in this paper we are presenting a sine/cosine generator using CORDIC algorithm and its performance reports. The pipelined architecture takes the angle/phase as input and gives both sine and cosine for the given input in predetermined number of micro rotations. These micro rotations are decided by the accuracy demanded by the application.

Section I gives brief introduction, section II defines the problem statement. The CORDIC algorithm and architecture are briefly studied in section III, The Simulation and Synthesis results are discussed in section IV and finally concluded in section V.

III. CORDIC Algorithm and Architecture

The CORDIC algorithm involves rotation of a vector ' v' on the *XY*-plane in circular, linear and hyperbolic coordinate systems depending on the function to be evaluated.

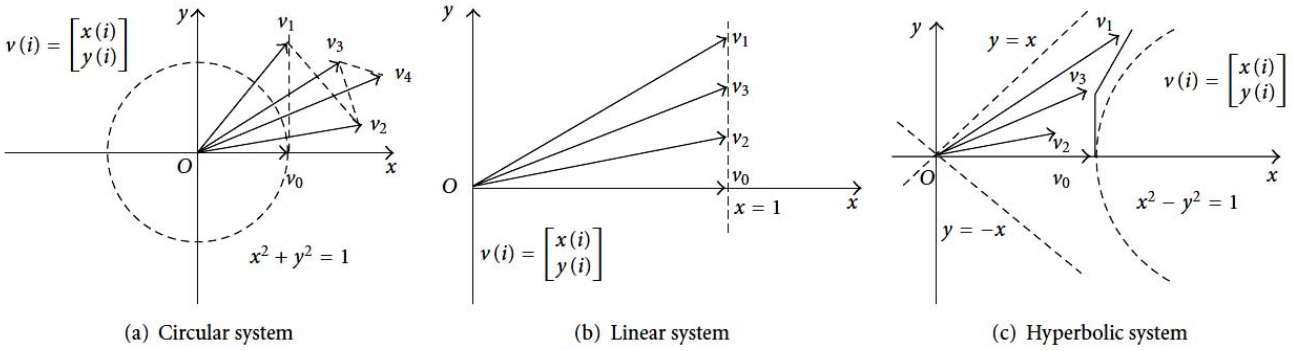
The conventional method of implementation of 2D vector rotation using Givens rotation transform is represented by the equations (1) and (2).

$$x_{out} = x_{in}\cos\theta - y_{in}\sin\theta \quad (1)$$

$$y_{out} = x_{in}\sin\theta + y_{in}\cos\theta \quad (2)$$

where (x_{in}, y_{in}) and (x_{out}, y_{out}) are the initial and final coordinates of the vector, respectively. The main principle of CORDIC algorithm is to implement every function in terms

of addition/subtraction and shifting. So, the multiplication terms in Givens rotation equations are removed by rearranging the equations (1) and (2). The single rotation is thus decomposed into micro rotations which are realized in terms of add/sub and shift operations. The generalized equations of the CORDIC algorithm for each micro iteration after rearranging givens rotation can be written as shown in equations (3), (4) and (5).



(a) Circular system (b) Linear system (c) Hyperbolic system

Fig. 1. Rotations in various coordinate systems

$$x_{i+1} = x_i - m\sigma_i y_i \rho^{-S_{m,i}} \quad (3)$$

$$y_{i+1} = \sigma_i x_i \rho^{-S_{m,i}} + y_i \quad (4)$$

$$z_{i+1} = z_i - \sigma_i \alpha_{m,i} \quad (5)$$

where α_i represents the direction of rotation, m represents the coordinate system in which the vector v is rotating circular$(m = 1)$, linear$(m = 0)$ and hyperbolic$(m = -1)$. $S_{m,i}$ is non decreasing integer shift sequence, ρ indicates the radix of number system in which CORDIC is implemented and $\alpha_{m,i}$ is the elementary rotation angle. The latter directly depends on $S_{m,i}$ through the relation (6)

$$\alpha_{m,i} = \frac{1}{\sqrt{m}} \tan^{-1}\left(\sqrt{m}\, \rho^{-S_{m,i}}\right) \quad (6)$$

The shift sequence $S_{m,i}$ depends on the coordinate system and the radix of number system. $S_{m,i}$ affects the convergence of the algorithm and *n* affects the accuracy of the final result. σ_i refers to the mode of CORDIC in which it is used. CORDIC can be used in two modes. One is rotation mode and other is vectoring mode. Equation (7) gives the value of σ_i in the CORDIC equations.

$$\sigma_i = \begin{cases} sign(z_i), & for\ rotation\ mode, \\ -sign(y_i), & for\ vectoring\ mode, \end{cases} \quad (7)$$

In rotation mode, the input angle θ (8) will be decomposed using a finite number of elementary angles

$$\theta = \alpha_0\sigma_0 + \alpha_1\sigma_1 + \ldots\ldots + \alpha_{n-1}\sigma_{n-1} \quad (8)$$

where *n* indicates the number of micro-rotations, α_i is the elementary angle for i^{th}iteration and σ_i is the direction of i^{th} micro-rotation. In rotation mode, z_0 is the angle accumulator initialized with the input rotation angle. The direction of vector in every iteration must be determined to reduce the magnitude of the residual angle in the angle accumulator. Therefore, the direction of rotation in any iteration is determined using the sign of the residual angle obtained in the previous iteration. Initially, the coordinates of initial vector are (x_{in}, y_{in}) and the micro rotations are performed such that the vector always moves closer toward required angle and decrease the value of the residual angle in angle accumulator. The coordinates of a vector obtained after n micro-rotations from initial position (x_{in}, y_{in}) are given by the equations (9), (10) and (11)

$$x_n = k(x_{in}\cos\theta - y_{in}\sin\theta) \quad (9)$$

$$y_n = k(x_{in}\sin\theta + y_{in}\cos\theta) \quad (10)$$

$$z_n \to 0 \quad (11)$$

CORDIC can be operated in different radix, generally powers of 2. The iteration equations of the radix-2 CORDIC algorithm in rotation mode of circular coordinate system at the $(i+1)^{th}$ step are obtained by using $\rho = 2$ in above equation and are given by the equations (12),(13) and (14)

$$x_{i+1} = x_i - \sigma_i y_i 2^{-i}, \quad (12)$$

$$y_{i+1} = \sigma_i 2^{-i} x_i + y_i, \quad (13)$$

$$z_i = z_i - \sigma_i \alpha_i \quad (14)$$

where $\alpha_i = \tan^{-1}(2^{-i})$ for circular coordinate system and 2^{-i} and $\tanh^{-1}(2^{-i})$ for linear and hyperbolic coordinate systems. σ_i refers to the direction of tracking/iteration and given by equation (15).

$$\sigma_i = \begin{cases} -1, & for\ z_i < 0, \\ 1, & otherwise, \end{cases} \quad (15)$$

In order to maintain a constant vector length, the obtained results have to be scaled by the scale factor k given by equation (16)

$$k = \prod_{i=0}^{n-1} \sqrt{1 + 2^{-2i}} \quad (16)$$

This constant factor changes with radix and for radix-2 CORDIC the constant factor, k ≈ 1.65.

There are number of ways to implement the CORDIC processor. The ideal architecture depends upon the speed verses area tradeoff's in the intended application. Simple among them is serial architecture. It is the direct solution for the CORDIC basic equations. In serial architecture only one

basic CORDIC block is present which contains three simple adder/Subtractors and two shifters with a ROM containing a look up table. ROM contains the fixed angle constants for the particular coordinate system in which CORDIC architecture is implemented. This architecture performs one micro rotation for every clock cycle. Thus, after n clock cycles the output is available. Since, it contains very less components involved in its architecture this type of architectures are used in area specific applications. The serial architecture is slow as it uses n clock cycles for every single calculation.

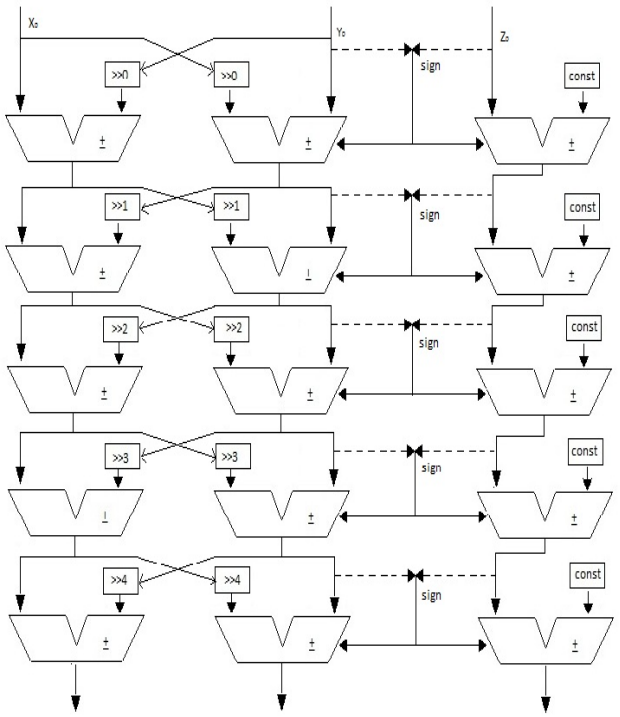


Fig. 2. Pipeline CORDIC processor

The second one is the pipelined structure. Pipelined CORDIC contains n number of CORDIC blocks which are cascaded. It contains fixed shift registers at each pipelined stage and performs fixed number of shifts every time. It contains registers at every stage to store the fixed angle for the particular micro rotation at each block in pipeline architecture. Thus, the first block always performs the first micro rotation i.e., 45 degrees. In the same way each block in this performs a single micro rotation. i.e., i^{th} stage performs the i^{th} micro rotation. This architecture is advantageous to serial architecture as it is fast and doesn't require a look up table. The number of blocks is dependent on the accuracy demanded by the application in which CORDIC is employed. The sign of z gives the direction of iteration for every stage as defined by the CORDIC equations and it can be clearly seen in the figure 2.

In this paper, the sine cosine generator we developed is a pipeline CORDIC with six cascaded stages each performing a specific micro-rotation. It operates in circular coordinate system in rotation mode. we use initial points of a vector as $\left(1/_k, 0\right)$ and input angle θ. Thus, the final coordinate equations become the required sine and cosine functions as represented by the equations (17),(18) and (19)

$$x_n = \cos\theta \quad (17)$$
$$y_n = \sin\theta \quad (18)$$
$$z_n \to 0 \quad (19)$$

IV. Simulation Results

This includes the simulation and synthesis of sine cosine generator implemented on the target device XILINX SPARTAN-3E. The sine cosine generator uses simple pipelined architecture based on CORDIC algorithm. Area and delay reports are also included for the corresponding target device. The CORDIC employed uses circular co-ordinate system and is operated in rotating mode. Hence, only phase is given as input and x, y values are given internally in the program or hardware.

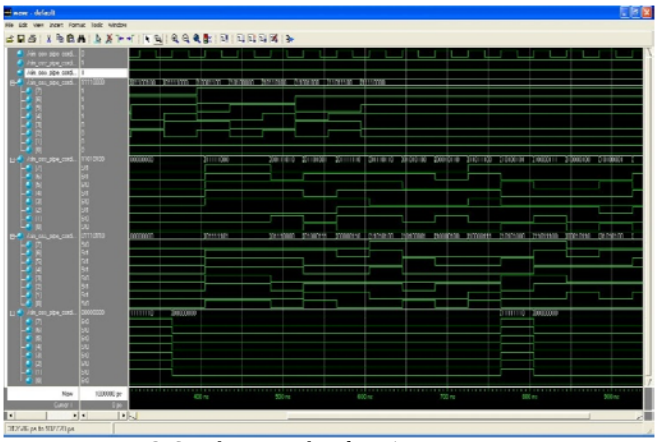


Fig. 3. Simulation results of sine/cosine generator

The code written in verilog is simulated using modelsim XE II/Evaluation 5.7g. After simulation of main block it has to be functionally verified. So, a test program with arbitrary inputs is written, figure.3 shows the compilation of test bench program by giving various test inputs to the master program.

As it is a pipelined architecture inputs can be given at every clock pulse and the cos and sine values for corresponding inputs will output after eight clock cycles as it. The three outputs for every angle input are sin function, cosine function and error (eps). eps indicates the proximity to the required angle tracked by CORDIC. In general eps indicates z_0.If the numbers of CORDIC blocks are increased in a pipeline architecture eps becomes less and approaches to zero.

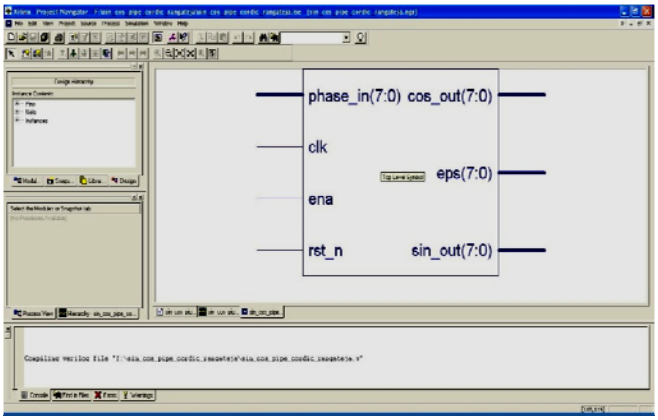


Fig. 4. XILINX window showing synthesized pin diagram

After the simulation of code in ModelSim the code is synthesized using XILINX ISE design suit and the figure 4 shows the corresponding pin diagram. The following table gives the device utilization summary of SPARTAN-3E when sine cosine generator using CORDIC algorithm is implemented on it.

TABLE I: THE ARRANGEMENT OF CHANNELS

1.	Number of Slices:	115 out of 960	11%
2.	Number of Slice Flip Flops:	188 out of 1920	9%
3.	Number of 4 input LUTs:	183 out of 1920	9%
4.	Number of bonded IOBs:	35 out of 108	32%
5.	Number of GCLKs:	1 out of 24	4%

The timing report includes the total time delay for the output to appear after giving input. This also includes detailed time delay for the logic and routing. i.e. 55% for logic and rest for the routing.

For Speed Grade: -4 minimum period required is 6.756ns which refer to a maximum frequency of 148.017MHz. The minimum input arrival time before clock and maximum output required time after clock are 5.404ns and 6.198ns.

V. CONCLUSION

In this paper, we developed a sine cosine generator, a common elementary trigonometric functions using one of the hardware efficient algorithm namely CORDIC. The sine cosine generator is targeted for SPARTAN 3E and it requires 11% of its total number of slices with a time delay of 6.76 ns in which 55% of delay is for logic and rest for routing. As the implemented design is a pipelined one, it is more efficient than bit serial approach and is more accurate and advantageous than bit serial architecture.

CORDIC is not only used in computation of elementary functions but also used for tracking of moving targets in space, distance between the multiple targets, reconfigurable systems, channel estimation computation at OFDM based Wireless communications, MIMO based video processing systems, etc. low power high speed CORDIC play effective role in future online computations.

REFERENCES

[1] J. E. Volder, "The CORDIC trigonometric computing technique," *IRE Transactions on Electronic Computers*, vol. 8, no. 3, pp. 330–334, 1959.

[2] J. S. Walther, "A unified algorithm for elementary functions," in *Proceedings of the AFIPS Spring Joint Computer Conference*, pp. 379–385, May 1971.

[3] R. Andraka, "A survey of CORDIC algorithms for FPGA based computers," in *Proceedings of the 6th ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA '98)*, pp. 191–200, February 1998.

[4] B.Lakshmi and A.S. Dhar "CORDIC Architectures: A Survey" in *Hindawi Publishing Corporation*, VLSI Design, Volume 2010, Article ID 794891, 19 pages.

[5] Ron Warner, Cort Lansenderfer "Low cost, Low power FPGAs Provide Ideal Platform for CORDIC Applications" in *lattice Semiconductor Corporation.*

[6] Jan Rabay and Konstatinos sarrigeorgidis "Ultra low power CORDIC processor for wireless communication algorithms" *Journal of signal processing 38,115-130, 2004 @ kluwer academic publishers. Manufactured in the Netherlands.*

[7] Jeongseon Euh, Jeevan Chittamuru, Wayne Burleson CORDIC VECTOR INTERPOLATOR FOR POWER-AWARE 3D COMPUTER GRAPHICS

[8] Y. H.Hu, "CORDIC-based VLSI architectures for digital signal processing," *IEEE Signal Processing Magazine*, vol. 9, no. 3, pp. 16–35, 1992.

R. Ranga Teja B.Tech, Department of Electronics and Communication Engineering, Sri Kalahasteeswara Institute of Technology, Srikalahasti, Chittoor District, Andhra Pradesh. He received B.Tech degree in Electronics and Communications Engineering from JNTU Ananthapur University. He was born on 1991 at Wanaparti in Andhra Pradesh. His area of research goes with VLSI and Digital Signal Processing.

Dr.P.Sudhakara Reddy, Associate Professor, Department of Electronics and Communication Engineering, Sri Kalahasteeswara Institute of Technology, Srikalahasti, Chittoor District, Andhra Pradesh. He received B.Tech degree in Electronics and Communications Engineering from N.B.K.R Institute of Science and Technology, Sri Venkateswara University, Tirupathi, Chittoor district ,Andhra Pradesh , M.Tech degree in Digital Systems and Computer Electronics Electronics from JNTU Hyderabad , Andhra Pradesh and Doctoral degree received from Sri Venkateswara University, Tirupathi, Chittoor strict , Andhra Pradesh, India. His area of research goes with VLSI Arrays for Signal/Image/Video Processing and Communications.