Calculation of Sine and Cosine of an Angle using the CORDIC Algorithm

KAVYA SHARAT
Final year Student
Department of Electronics and Communication
RV College of Engineering
Bangalore-560059

Dr.B.V.UMA
Professor
Department of Electronics and Communication
RV College of Engineering
Bangalore-560059

SAGAR D.M
Final year Student
Department of Electronics and Communication
RV College of Engineering
Bangalore-560059

Abstract: **With increasing on chip complexities the on chip area is a major concern. Today users desire every gadget to be small enough, particularly the hand held systems. CORDIC is one such algorithm which serves this purpose. CORDIC algorithm has become a widely used approach to elementary function evaluation when silicon area is a primary concern. CORDIC is more economical than DSP algorithms both in terms of area and power consumption. This paper presents how to calculate sine and cosine values of the given angle using CORDIC algorithm. A brief description of the theory behind the algorithm is also given. Summary of CORDIC synthesis results based on Xilinx FPGAs is given. The system simulation was carried out using Xilinx ISE Design Suite12.1. The system is implemented using Virtex5 XC5VF70T FPGA with Xilinx ISE12.1 and Verilog Hardware Description Language.**

Keywords: **CORDIC, sine, cosine, FPGA, synthesis**

I. INTRODUCTION

Due to rapid advances made in VLSI technology, there is an entirely different approach towards computer design for real time applications. Many of the DSP applications try to closely simulate real time images. Speed, clarity and resemblance to real time objects are some of the issues to be addressed to achieve this goal. Trigonometric function calculation is one of the primary tasks performed in DSP applications. For a long time microprocessor – based systems have been used to perform this task. Software algorithms used by processors do not meet the highly demanding needs of all DSP tasks. So, hardware algorithms are used to perform such tasks. Instructions for performing a task are hardwired into the processor itself, i.e., the program is built right into the microprocessor circuit itself [1].

One such algorithm is CORDIC algorithm [2].This algorithm was initially developed for trigonometric functions, using Givens rotation transform technique. The CORDIC algorithm generally produces one additional bit of accuracy for each iteration [3]. Using a prescribed sequence of conditional additions or subtractions, the CORDIC arithmetic unit can be designed to solve the iterative equations.

By making slight adjustments to the initial conditions and the LUT values, it can be used to efficiently implement trigonometric functions etc. using the same hardware. Since it uses only shift-add arithmetic, the VLSI implementation of such an algorithm is easily achievable.

Section II gives brief overview of methods to compute trigonometric functions. The CORDIC algorithms briefly studied in section III and architecture in section IV. The simulation and synthesis results are discussed in section V and finally future scope and conclusions in section VI.

II. EVALUATION OF TRIGONOMETRIC FUNCTION

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

A. Look up Table

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

B. Taylor Series

The Taylor series expansion for sine is:

$\sin x = x - x^3/3! + x^5/5! - x^7/7! + \cdots$

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

C. CORDIC Algorithm

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

III. ALGORITHM

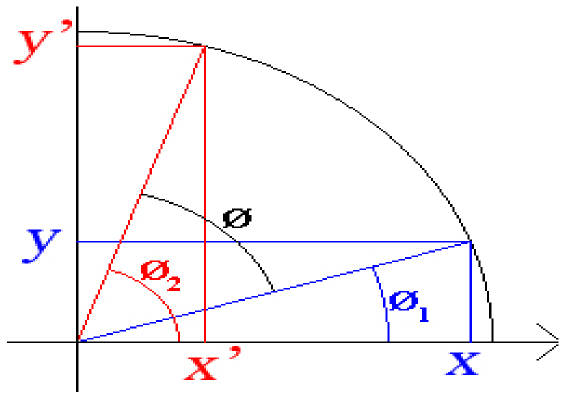


Figure 1. Rotation of a point by an angle□

In Figure1, the diagonal blue line is at an angle □1above the horizontal. The diagonal red line is actually the blue line rotated anti-clockwise by an angle □. The new X and Y values are related to the old X and Y values as follows

y (1)

For CORDIC, □2is the final angle whose sine or cosine we want to calculate andthe initial angle □1is set to a convenient value such as 0. Rather than rotating from □1to □2 in one full sweep, we move in steps with careful choice of step values. Rearranging (1) gives us:

In CORDIC, we restrict the angle of rotation such that tan .Thus multiplication by tan term can be replaced by simple shift operation .If i be the decision at each iteration so as to which direction to rotate, then the iterative rotation can be expressed as

where,

$$K_i = \cos(\alpha_i) = \cos(\tan^{-1}(2^{-i}))$$
$$d_i = \pm 1$$

is the scaling factor. The product of depends on the number of iterations performed before arriving at the final result and approaches 0.6073 as the number of iterations reaches infinity. Therefore, the rotation algorithm has a gain, ≈ 1.65. The exact gain depends on the number of iterations, and follows the following equation:

(4)

The angle of a composite rotation is realized by the sequence of the directions of the elementary rotations.An adder-subtractor is used that accumulates the elementary rotation angles post iteration. The angle accumulator adds a third difference equation to the CORDIC algorithm:

(5)

when the angle is in the arctangent base, this extra element is not needed.

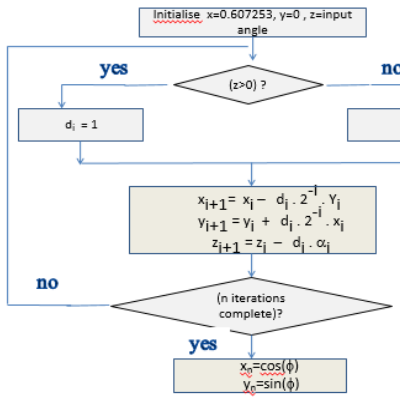

Initialise x=0.607253, y=0 , z=input angle
yes
(z>0) ?
no
$d_i = 1$
$d_i = -1$
$x_{i+1} = x_i - d_i . 2^{-i} . y_i$
$y_{i+1} = y_i + d_i . 2^{-i} . x_i$
$z_{i+1} = z_i - d_i . \alpha_i$
no
(n iterations complete)?
yes
$x_0=\cos(\phi)$
$y_0=\sin(\phi)$

Figure 2. Evaluation of sine and cosine of an angle

Figure 2 shows the algorithm to find sine and cosine of any angle using equation 5 and few conditional statements.

A. Rotation Mode

One mode of operation, called rotation mode, rotates the input vector by a specified angle. Here, the angle accumulator is initialized with the desired rotation angle. The rotation decision based on the sign of the residual angle and is such that the magnitude of the residual angle in the angle accumulator diminishes at each iteration. If the input angle is already expressed in the binary arctangent base, the angle accumulator is not needed. The equations for this are:

(5)

Where

if

+1 otherwise

Then,

$$x_n = A_n[x_0.\cos z_0 - y_0.\sin z_0]$$
$$y_n = A_n[y_0.\cos z_0 + x_0.\sin z_0]$$
$$z_n = 0$$
$$A_n = \prod n\sqrt{(1+2^{-2i})} \qquad (6)$$

B. Evaluation of Sine and Cosine using CORDIC

In rotational mode the sine and cosine of the input angle can be computed simultaneously [6], [7]. Setting the y component of the input vector to zero reduces the rotation mode result to:

$$x_n = A_n.x_0.\cos z_0$$
$$y_n = A_n.y_0.\sin z_0 (7)$$

If x_0 is equal to $1/A_n$, the rotation produces the unscaled sine and cosine of the angle argument, z_0 . Very often, the sine and cosine values modulate a magnitude value. Using other techniques like an LUT requires a pair of multipliers to obtain the required modulation. The algorithm performs the multiply as part of the rotation operation, and therefore eliminates the need for a pair of explicit multipliers. The output of the CORDIC rotator is scaled by the rotator gain. If the gain is not acceptable, a single multiply by the reciprocal of the gain constant placed before the CORDIC rotator will yield unscaled results.

IV. ARCHITECTURE

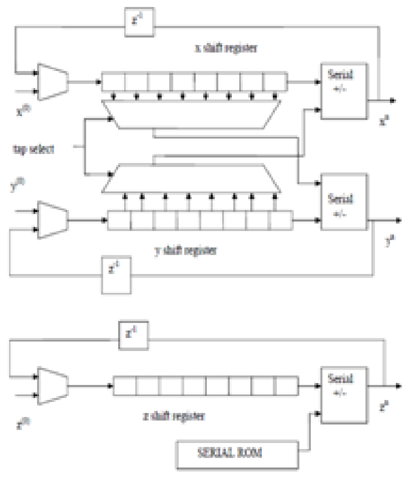


Figure 3. CORDIC Architecture for sine and cosine

The CORDIC algorithm requires approximately one shift add/ sub operation for each bit of accuracy. A CORDIC core implemented with sequential architectural configuration, implements these shift-add/sub operations serially, using asingle shift-add/sub stage and feeding back the output [4]. An iterative CORDIC core with N bit width has a minimum latency of N cycles. It takes at least N cycles to produce new output. The implementation size is directly proportional to the internal precision. To obtain sine and cosine values of a given angle z0, iterative structure takes the value of (x_0, y_0)as (1,0) in the first clock cycle. From the next clock cycle onwards it takes the feedback values and the operation continues till the required output is obtained. To get an N bit precise output, the structure requires iterating at least N times. Hence, it requires a minimum of N clock cycles for required output. This architecture offers advantages like less hardware complexity and low power consumption. But maximum number of clock cycles are required to calculate the output, thus calculation time is large. Also variable shifters do not map well on certain FPGAs

A. Advantages of CORDIC: [7]

- Minimum number of gates are needed for hardware implementation on an FPGA. Thus, hardware complexity is greatly reduced compared to other processors such as DSP multipliers. Hence, it is relatively simple in design.
- For hardware implementation, cost is less since only shift registers , adders and look – up table (ROM) are required.
- Delay involved during processing is comparable to that of a division or square-rooting operation.
- No multiplication and only addition, subtraction and bit-shifting operation ensures simple VLSI implementation.
- Either if there is an absence of a hardware multiplier (e.g. microcontroller, microprocessor) or there is a necessity to optimize the number of logic gates (e.g. FPGA), CORDIC is the preferred choice [4].

B. Applications

- CORDIC sine/cosine generators are used in satellite data processing systems for attitude determination.
- CORDIC modules has been proposed for calculation of Legendre Polynomials.
- Calculation of DFT, DHT, Chirp Z-transforms, filtering, Singular value decomposition and solving linear systems.
- In calculators , for calculation of transcendental functions.
- CORDIC can be used for linear functions , hyperbolic functions, square rooting , logarithms, exponentials

V. IMPLEMENTATION

In this paper, the FPGA implementation for calculating the sine and cosine values of given angle using CORDIC algorithm is presented for 8bits for the architecture shown in figure 3. The module was implemented on Virtex 5 FPGA by using Xilinx ISE Design Suite 12.1 and Verilog HDL and synthesized using the Xilinx ISE Design Suite. The following figure 4 shows the top-level RTL schematic of the Sine/Cosine generators.

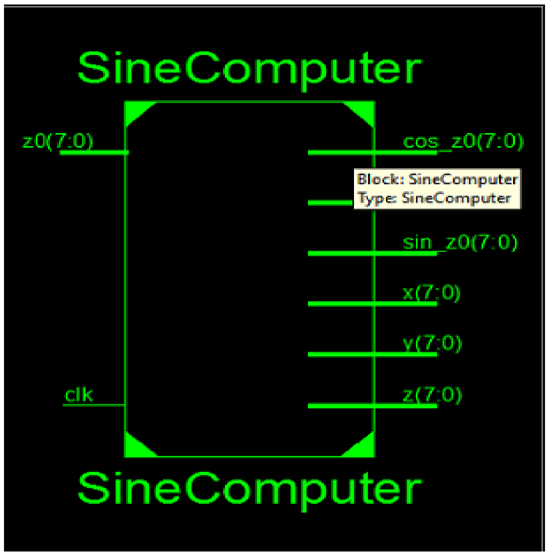


Figure 4.Top-level RTL schematic of the Sine/Cosine generators

Figure 5shows the simulation results for binary input angle z0 =45deg and binary outputs X(cos(z0)), Y(sin(z0)) in the form of waveform and their corresponding magnitude respectively.

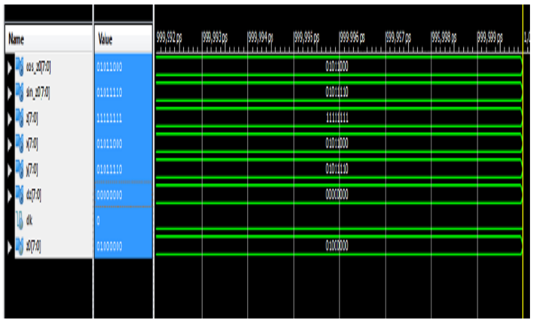
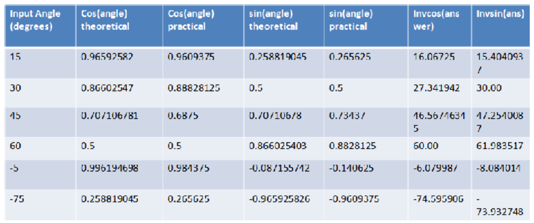


Figure 5.Sine/Cosine value for the given input angle Ain.

Table 1. Sine and cosine values obtained for different angles



Input Angle (degrees)	Cos(angle) theoretical	Cos(angle) practical	sin(angle) theoretical	sin(angle) practical	Invcos(answer)	Invsin(ans)
15	0.96592582	0.9609375	0.258819045	0.265625	16.06725	15.404093 7
30	0.86602547	0.88828125	0.5	0.5	27.341942	30.00
45	0.70710678 1	0.6875	0.70710678	0.73437	46.5674634 5	47.254008 7
60	0.5	0.5	0.866025403	0.8828125	60.00	61.983517
-5	0.996194698	0.984375	-0.087155742	-0.140625	-6.079987	-8.084014
-75	0.258819045	0.265625	-0.965925826	-0.9609375	-74.595906	-73.932748

Table 1 shows practical and theoretical sine and cosine values for different input angles along with inverse for the sine and cosine obtained practically. Table 2 shows the synthesis report of the CORDIC algorithm for calculating sine and cosine of the given angle represented with 8bits.

Table 2.Synthesis report

Number of slice registers	79
Number of Slice LUT s	123
Number of fully used LUT - FF pairs	78
Number of bonded IOB	49
Maximum frequency	241.106 MHz
Minimum period	4.148 ns

VI. FUTURE SCOPE AND CONCLUSION

A. *Future scope*

- Angle recording technique can be used to reduce the number of iterations.
- Using Extended Elementary Angle Scheme.
- Optimisation of area for implementation on VLSI chips can be done by sharing of adders or using redundant adders. (CSA)
- Scaling free architectures [5]
- Pipelined Architecture can be used to improve speed.
- Iteration Number can be reduced by using High radix CORDIC. (e.g., Radix-4, Radix-8)

B. *Conclusions*

In this paper CORDIC algorithm has been successfully simulated for calculating the Sine and Cosine of an angle represented using 8 bits, on ISim simulator using the Verilog HDL programming language. This is different from the architecture proposed in [7] which deals with 8 bit representation of angles with distinction made between ALU and Control unit architectures .This system is implemented on Virtex 5(XC5VFX70T) FPGA using ISE Design Suite 12.1. The design is more efficient and consumes less resources and is less time intensive. Our system has a maximum frequency of 241.106 MHz with a minimum period of 4.148 ns. 123 slices of LUTs are used to implement the architecture.

VII. ACKNOWLEDGEMENTS

The authors would like to thank Prof.Uttarakumari, Head of the Electronics and Communication Department, for her guidance and also thank Prof. Satyanarayana Principal, R V College of Engineering, Bangalore, for his encouragement to carry out this project. The authors thank Dr.S.Sridhar, Professor and Dean, Cognitive & Central Computing Facility of R.V. College of Engineering, Bangalore, India for having communicated this paper to this journal.

REFERENCES

[1] V. Sharma, "FPGA Implementation of EEAS CORDIC based Sine and Cosine Generator," M. Tech Thesis, Dept. of Electronics and Communication Engineering, Thapar University, Patiala, 2009.

[2] J. Volder, "The CORDIC Trigonometric Computing Technique," IRE Transactions on Electronic Computing,Vol EC-8, Sept 1959, pp. 330-334.

[3] R. K. Jain, B. Tech Thesis, "Design and FPGA Implementation of CORDIC-based 8-point 1D DCT Processor" , NIT Rourkela, Rourkela, Orissa, 2011.

[4] Er. Manoj Arora, Er. R S Chauhan, Er.Lalit Bagga," FPGA Prototyping of Hardware Implementation of CORDIC Algorithm", International Journal of Scientific & Engineering Research, Volume 3, Issue 1, January-2012 1 ISSN 2229-5518

[5] P.Keerthi, ShaikJaffar, "Implementation of Efficiency CORDIC Algorithmfor Sine & Cosine Generation " , IOSR Journal of Electronics and Communication Engineering (IOSR-JECE) e-ISSN: 2278-2834,p- ISSN: 2278-8735. Volume 6, Issue 5 (Jul. - Aug. 2013), PP 49-56

[6] 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

[7] AmanChadha , DivyaJyoti and M. G. Bhatia ," Design and Simulation of an 8-bit Dedicated Processor for calculating the Sine and Cosine of an Angle using the CORDIC Algorithm "Proceedings of the 2011 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC) IEEE