Area reduction in cordic processor using lookup table Method

R.SENTHAMIZHSELVI[1]
M.E (VLSI DESIGN)
Department Of Electronics and Communication Engineering
Srinivasan Engineering College.
Perambalur, Tamilnadu.
senthamizh.selvi08@gmail.com[1].

MR.M.SATHIYENTHIRAN[2]
Assistant professor
Department Of Electronics and Communication Engineering
Srinivasan Engineering College.
Perambalur,Tamilnadu.
sathiyen.ece@gmail.com[2].

Abstract -- An area–time efficient cordic algorithm that completely eliminates the scale factor. Look Up Table (LUT) is used to reduce the no.of flipflop's,slices and scaling factor. This cordic processor provides the flexibility to manipulate the number of iterations depending on the accuracy, area and latency requirements. Compared to the existing ,the proposed gives the lowest slice delay product. The proposed design has a considerable reduction in hardware when compared with other scaling free architectures. Fixed word length gives a better choice to choose the parameters. The proposed design implemented in Xilinx Spartan 2E device.

IndexTerms- CoordinateRotationDigital Computer (CORDIC), cosine/sine, FieldProgrammale Gate Array(FPGA), Look Up Table(LUT).

I. Introduction

The name CORDIC stands for Coordinate Rotation Digital Computer.Volder 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. The CORDIC method was later expanded for multiplication, division, logarithm, exponential and hyperbolic functions. The various function computations were summarized into a unified technique. The Coordinate Rotational Digital Computer (CORDIC) algorithm is an iterative technique that is capable of evaluating many basic arithmetic operations and mathematical functions. Examples of those that can be performed directly using the CORDIC technique are multiplication, division, square root, sine, cosine, inverse tangent, hyperbolic sine, hyperbolic cosine, and inverse hyperbolic tangent. The results of these basic functions can be processed further to obtain the functions of tangent, hyperbolic tangent, logarithms, and exponentials. Clearly this algorithm is a very powerful tool in areas where arithmetic function evaluation is heavily utilized such as robot control, Engineering graphics, and digital signal processing. However, a major shortcoming of CORDIC is the magnitude restriction that one must impose on its various input variables in order to guarantee that its output values converge. We introduce modifications to the algorithm which greatly ease these restrictions. We will emphasize the effects that these modifications have on a binary, fixed-point hardware implementation of the CORDIC algorithm.

II. Review of cordic and free its scaling implementation

The conventional CORDIC was first implemented by Volder, in 1959 [1]. The basic equations of the algorithm for circular coordinate system are shown below.

$$Xi+1 = [Xi - 2\text{-}i\ Yi] * Ki$$
$$Yi+1 = [Yi + 2\text{-}i\ Xi] * Ki$$
$$Zi+1 = Zi - \arctan(2\text{-}i) \qquad (1)$$

The above set of equations is considered for positive angle of rotation. If the angle is negative, the arithmetic signs get reversed. The index i represents the number of iteration of the unit since the number of iterations depends on the precision. Where i = 0 to N-1, N is the number of bits in the xy data path or the precision of the inputs. To eliminate the hardware required to compute the above constant after performing the algorithm, many have proposed alternate algorithms [3] – [6].

The work proposed in [3], with parallel compensation of scale factor, has shown two methods namely double rotation method and bit analysis method, compensates the scaling factor in parallel while carrying out the algorithm. Though the scale factor is compensated in parallel, additional hardware such as multiplexers and adders gets added in each stage of iteration making the architecture a bulky one. The one presented in [4], called MSR_CORDIC algorithm carries out computations fastly compared to conventional CORDIC but it

also has a drawback of additional shifters (2i+1 shifters) and adders which increase hardware.

The design proposed in [5], uses additional shifters and adders compared to conventional architecture. This design even depends on a parameter called basic shift, which limits the angle of rotation and more care has to be taken while mapping the angle to entire coordinate space. The above design is called modified virtually scaling free CORDIC. Another architecture using generalized micro-rotation selection is proposed in [6].

In this, they have approximated the angles of sin and cos using Taylor series expansion, as shown below.

$$\text{Sin } \alpha = \alpha - (3!)\text{-}1 \cdot \alpha 3 + (5!)\text{-}1 \cdot \alpha 5 \ldots$$
$$\text{Cos } \alpha = 1 - (2!)\text{-}1 \cdot \alpha 2 + (4!)\text{-}1 \cdot \alpha 4 \ldots \quad (2)$$

This recursive architecture though has better performance compared to that of others in same family has hardware overhead compared to conventional CORDIC. The advantage of this architecture is that it has lesser slice delay product compared to that of other scaling free architectures. In enhanced scaling free CORDIC proposed in [7], they have used radix 4 booth encoding to perform the algorithm. The disadvantage of this is that it performs rotation only in one direction. From this architecture, it is evident that even this has higher hardware compared to conventional CORDIC but the advantage lies in faster computation of the vector rotation.

Thus the architectures mentioned above for obtaining scale free CORDIC mostly have much hardware overhead compared to conventional CORDIC which makes the designers to concentrate on designing scale free architectures which have lesser or comparable hardware overhead to that of conventional CORDIC. Though latency is another issue in these designs, pipelined designs always have better latency compared to fully dedicated architectures.

III. Area reduction using lookup table method

Finding the value of floating point method and fixed point method using MATLAB. The value of floating point method is $6.7291e^{-004}$ and fixed point value is $6.6953e^{-004}$, in the fixed point deviation is lesser compare to floating point method.

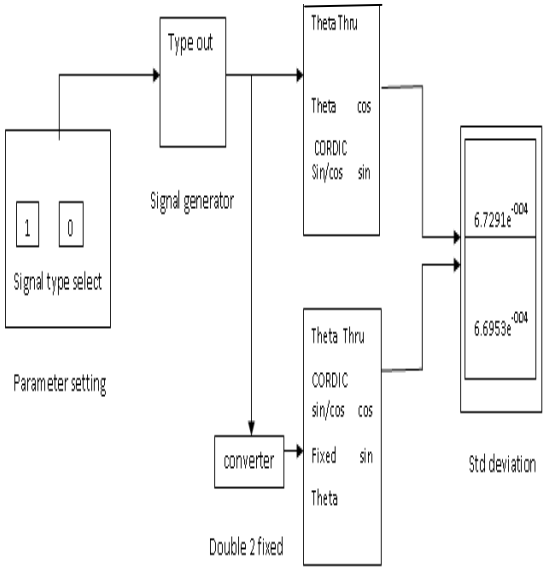

Fig.1. cordic fixed rotation sine/cosine calculation.

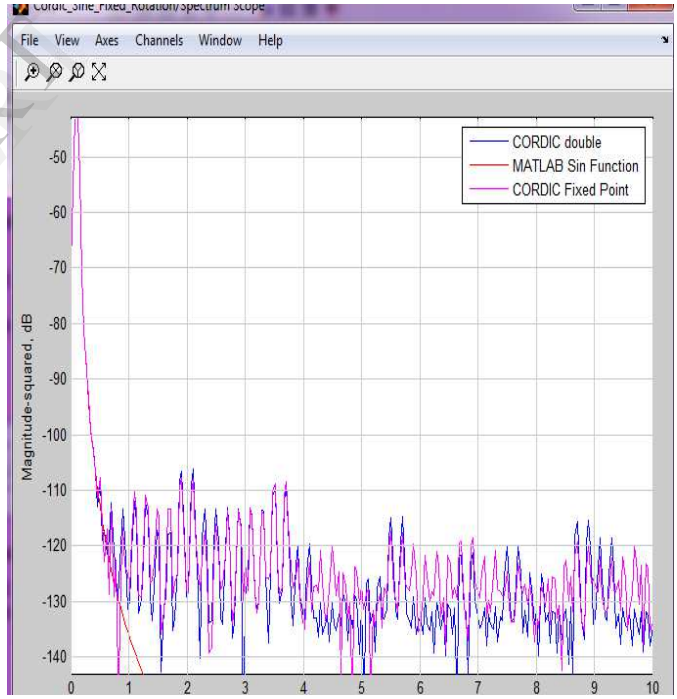

Fig.2. Frequency vs Magnitude-squared unit

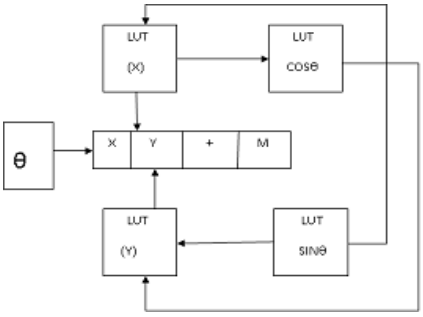

Fig.3.Look Up Table using block diagram

Advantage of this look up table method is gives the lesser slice delay product compared to other existing scaling free architectures. The block diagram for the proposed CORDIC architecture is shown in Fig.3. Multipliers and adders are needed in this method. The schematic of the top module of the proposed design is shown in Fig.4. The sub module,CORDIC_FP,in the figure is a 16 stages of CORDIC units.

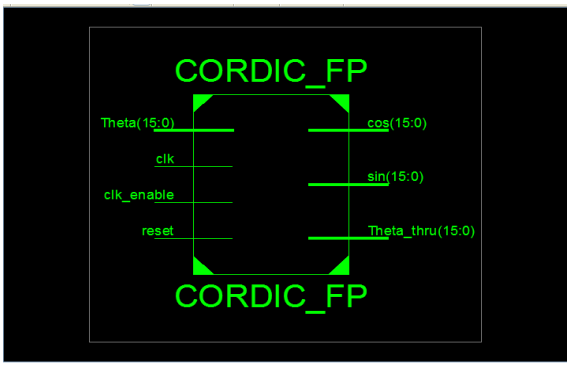

Fig.4.The schematic of the top module of the proposed design

IV. Results

TABLE 1. shows the device utilization summary when the design is implemented in Xilinx Spartan 2E FPGA.

TABLE 1.Device utilization summary

LOGIC UTILIZATION	USED	AVAILABLE	UTILIZATION
NUMBER OF SLICES	503	4656	10%
NUMBER OF FLIPFLOPS	798	9312	8%
NUMBER OF LUTS	984	9312	10%

Thus the slice delay product(SDP) can be given by

SDP=(No.of slices* worst case iterations)/freq. In MHz.

As the number of worst case iterations in proposed design is 16,the slice delay product comes to low compared to existing.

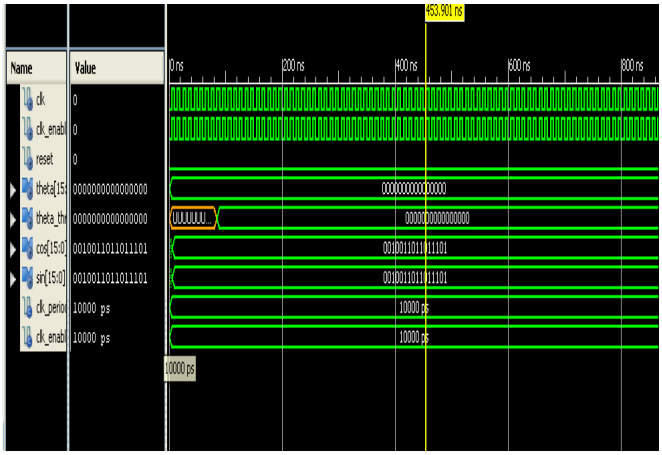

Fig.5.Xilinx simulation result.

Fig .5. shows the Xilinx simulation output of the CORDIC unit with the proposed scaling units. The corresponding outputs are represented in cos,sin,Theta-thru respectively.

V. Conclusion

A trade-off speed/area will determine the structural approach to CORDIC. The proposed algorithm provides scale-free solution for realizing vector-rotations using CORDIC, this is , not only meet the accuracy requirement,but also to attain the adequate range of convergence. This CORDIC processor gives the lowest slice delay product. The hardware requirement of CORDIC unit with proposed scaling units is less (or) comparable to other scalefree CORDIC architectures.

References

[1] J. E. Volder, “The CORDIC trigonometric computing technique,” IRETrans. Electron. Comput. vol. EC-8, pp. 330– 334, Sep. 1959.

[2] P. K. Meher, J. Walls, T.-B. Juang, K. Sridharan, and K. Maharatna, “50 years of CORDIC: Algorithms, architectures and applications,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 56, no. 9, pp. 1893–1907, Sep. 2009.

[3] J. Villalba, J. A. Hidalgo, E. L. Zapata, E. Antelo, and J. D. Bruguera, “CORDIC architectures with parallel compensation of scale factor,” Proc. Application Specific Array Processors Conf., pp. 258 – 269, Jul. 1995.

[4] Zhi-Xiu Lin and An-Yeu Wu, “Mixed- Scaling_Rotation CORDIC (MSR-CORDIC) Algorithm and Architecture for Scaling-Free High-Performance Rotational Operations,” Proc. Acoustics, Speech, and Signal Processing Conf., vol. 2, pp. 653 – 656, Apr. 2003.

[5] K. Maharatna, S. Banerjee, E. Grass, M. Krstic, and A. Troya, “Modified Virtually Scaling-Free adaptive CORDIC Rotator Algorithm and Architecture,” IEEE Trans. Circuits Syst. for Video Tech., vol. 5, no. 11, pp. 1463 – 1474, Nov. 2005.
[6] S. Aggarwal, P. K. Meher, and K. Khare, “Area- Time Efficient Scaling-Free CORDIC Using Generalized Micro- Rotation Selection,” IEEE Trans. VLSI Syst., vol. 20, no. 8, pp. 1542 – 1546, Aug. 2012.
[7] F. J. Jaime, M. A. Sanchez, J. Hormigo, J. Villalba, and E. L. Zapata, “Enhanced Scaling-Free CORDIC,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 7, pp. 1654 – 1662, Jul. 2010.
[8] M. G. Buddika Sumanasena, “A Scale Factor Correction Scheme for the CORDIC Algorithm,” IEEE Trans. Computers, vol. 57, no. 8, pp. 1148 – 1152, Aug. 2008.