International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-1 Issue-5, October 2012

Implementation of Coordinate Rotation algorithm for Hardware Multipliers

Swathi Gera, M.Ashok Kumar, K.V.Ramana Rao

Abstract: ***Most of the hardware algorithms exist to handle the hardware intensive signal processing problems. Among these algorithms is a set of shift-add algorithms collectively known as CORDIC for computing a wide range of functions including certain trigonometric, hyperbolic and logarithmic functions. Apart from this it can handle linear functions. Even though numerous articles covering various aspects of CORDIC algorithms, very few surveys concentrate on implementation in FPGAs. CORDIC (Coordinate Rotation Digital Computer) is an algorithm for computing transcendental functions like sine, cosine and arctangent. The method can also be easily extended to compute square roots as well as hyperbolic functions. The algorithm works by reducing the calculation into a number of micro-rotations for which the arctangent value is precomputed and loaded in a table. This method reduces the computation to addition, subtraction, compares, and shifts.Various digital architectures were proposed and compared, including low-cost sequential and high performance pipelined solutions. Fixed point and floating point arithmetic was considered. The concepts were implemented in VHDL, verified and synthesized with Xilinx tools. Selected approach was physically implemented and tested.***

Index Terms: CORDIC, sine, cosine, FPGA, synthesis. SVD, digital, hardware, VHDL, FPGA

I. INTRODUCTION

In 1959 Jack E. Volder described the Coordinate Rotation Digital Computer or CORDIC for the calculation of trigonometric functions, multiplication, division and conversion between binary and mixed radix number systems. The CORDIC- algorithm provides an iterative method of performing vector rotations arbitrary angles using only shift and add. Processing of matrices, especially inversion remains a key challenge for contemporary computing machines. Very smart algorithms were proposed many years ago, by the scientists who expected rapid development of digital hardware in the future. Many of those solutions were presumed to work on futuristic parallel devices. CORDIC are good examples here [1-3]. Eventually recent years have brought the long expected rapid development of digital hardware and growth of programmable logic devices complexity. This paper describes a study of hardware implementation and theory on CORDIC modules. The authors focus on comparison of architecture variants in the context of resource allocation, speed and accuracy. Similar works may be found in contemporary literature [8] showing growing interest in practical use of achievements of great mid 20-th century mathematicians.

Revised Manuscript Received on 30 October 2012
*****Correspondence Author(s)
Swathi Gera*, Department of ECE, JNTUK University, Pydah College of Engineering and Technology, Vishakhapatnam, India.
M.Ashok Kumar, Asst. Professor, Department of ECE, JNTUK University, Pydah College of Engineering and Technology, Vishakhapatnam, India.

II. CORDIC THEORY

To rotate the coordinate (X i ,Y i) to the point (Xj,Yj) We use the following matrix calculation.

$$\begin{bmatrix} X_j \\ Y_j \end{bmatrix} = \begin{bmatrix} \cos\Theta & -\sin\Theta \\ \sin\Theta & \cos\Theta \end{bmatrix} * \begin{bmatrix} X_i \\ Y_i \end{bmatrix} \quad (1)$$

Equation 1 can be modified by factoring a cos➔ leading to the following:

$$\begin{bmatrix} X_j \\ Y_j \end{bmatrix} = \cos\Theta * \begin{bmatrix} 1 & -\tan\Theta \\ \tan\Theta & 1 \end{bmatrix} * \begin{bmatrix} X_i \\ Y_i \end{bmatrix} \quad (2)$$

Any angle Θ in the first quadrant can be represented by the following equation.

$$\Theta_n = \sum_{n=0}^{\infty} S_i * \arctan\left(1/2^n\right) \quad (3)$$

where Si = 0 or 1

For example, the angle 0 is computed by letting S_i in equation (3) be simply Si =0,0,0,0... . The angle Θ =π/2 is computed by letting S i =1,1,1,1... . π /4 would be 1,0,0,0... , and so on. In practice, due to the accuracy limits of the digital representation, the summation can be limited to the number of bits representing the value, or fewer. The arctan(1/2)^n quickly approaches 0.

Note that equation 3 can also be constructed such that S_i = -1 or 1. For example, the angle π /4 would be computed with the sequence 1,1,−1,−1,−1,... . The advantage of this method will become apparent. This results in the following equation for $\tan\Theta_n$.

$$\tan\Theta_n = S_n * (1/2^n) \quad (4)$$

Turning our attention back to equation (2), and given that the angle Θ can be represented as a weighted sum (3), we can represent the rotation from (X i ,Y i) to (X j ,Y j) as a sequence of smaller rotations where the n+1 rotational coordinate is derived from the n th coordinate using the following equation

$$\begin{bmatrix} X_{n+1} \\ Y_{n+1} \end{bmatrix} = \cos\Theta_n * \begin{bmatrix} 1 & -\tan\Theta_n \\ \tan\Theta_n & 1 \end{bmatrix} * \begin{bmatrix} X_n \\ Y_n \end{bmatrix} \quad (5)$$

By substituting equation (4) into equation (5) we replace the tangent function with a divide by 2^n , which is a simple shift operation

$$\begin{bmatrix} X_{n+1} \\ Y_{n+1} \end{bmatrix} = \cos\Theta_n * \begin{bmatrix} 1 & -S_n * (1/2^n) \\ S_n * (1/2^n) & 1 \end{bmatrix} * \begin{bmatrix} X_n \\ Y_n \end{bmatrix} \quad (6)$$

By implementing the algorithm letting S_n take on the values -1 and 1, and by noting that

$$\cos(\Theta) = \cos(-\Theta) \quad (7)$$

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)

the $\cos(\Theta)$ term reduces to a constant K given by the equation

$$K=\prod_{n=0}^{\infty}\left(\cos\left(\arctan\left(1/2^{n}\right)\right)\right)\approx 0.607253 \qquad (8)$$

The constant K means that the CORDIC function as an overall gain by the factor 1/K which is often referred to as the CORDIC gain. Since this is a constant, it can be taken into account in other parts of the system.

The CORDIC algorithm can be implemented in hardware using three approaches: a sequential approach - the structure is unfolded in time, a parallel approach – the structure is unfolded in space or a combination of the two. In the first- sequential approach, arithmetic modules are shared by iterations. Intermediate results are fed back via the registers and the appropriate angles are delivered to arithmetic units by the muxes. Control is provided by iteration counter. Figure 1 shows the CORDIC sequential architecture.

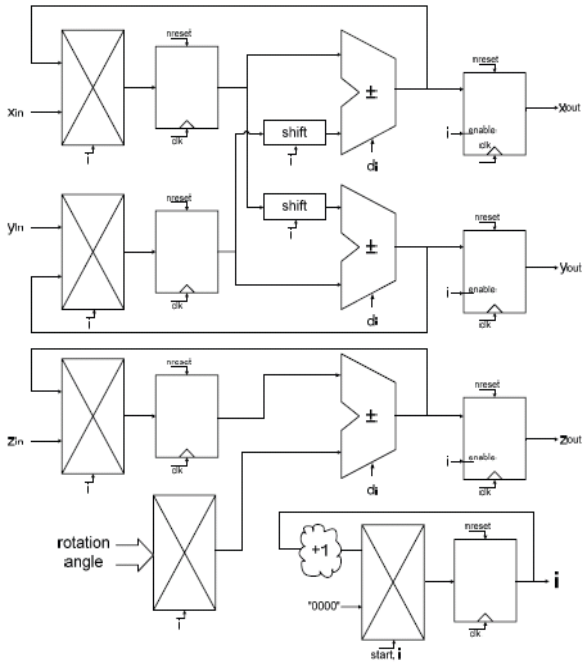

Figure 1. CORDIC - sequential architecture

Another concept is pipelined architecture presented in Fig. 2. Schematic shows a hardware providing 3 consecutive iterations. Arithmetic blocks are replicated for each iteration, thus the data flow may form a pipeline. This solution provides
much faster throughput but needs more hardware resources. Arithmetic is fixed point with 8-bit numbers coded in 2'complement *Figure 3 shows the CORDIC hardware implementation*

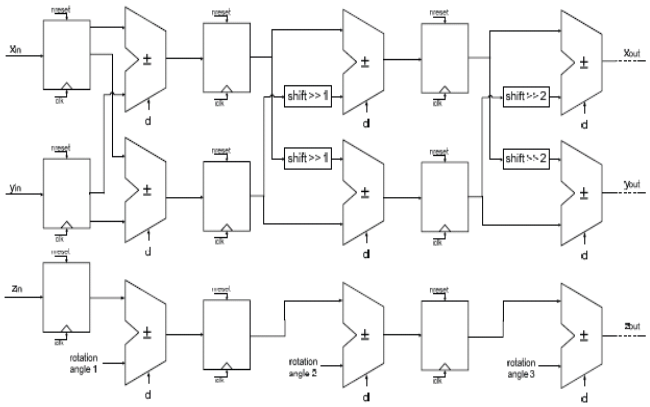

Figure 2. CORDIC – pipelined architecture

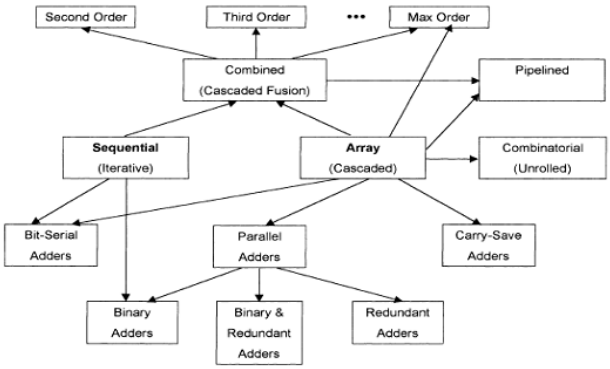

Figure 3: shows the CORDIC hardware implementation.

III. IMPLEMENTATION RESULTS

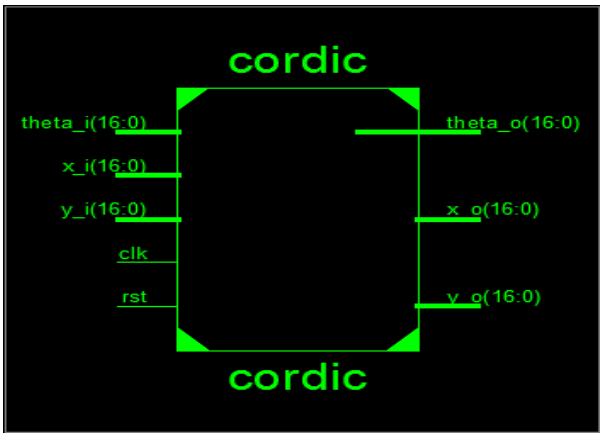

Fig. 4 shows the RTL block of the cordic algortihm

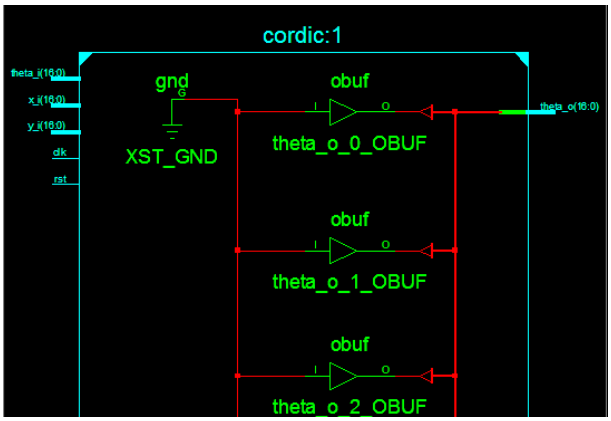

Figure 4 shows the technology schemeatic.

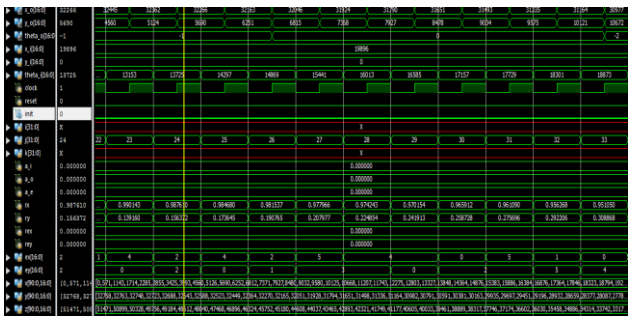

Figure 5:OUTPUT WAVEFORM

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

IV. CONCLUSIONS

This paper presents theoretical and practical aspects of implementing sine/cosine CORDIC-based generators in FPGAs. The main results can be summarized as follows: A trade-off speed/area will determine the right structural approach to CORDIC FPGA implementation for an application. Module count and operating speed depend significantly on the used synthesis tool. Simulation has shown that the redundant adder can improve the efficiency of CORDIC FPGA implementations for bit-lengths higher than 32-bit.

REFERENCES

1. C. Eckart, G. Young, “The approximation of one matrix by another of lower rank”, Psychometrika, vol. 1, no. 3, 1936.
2. J.E. Volder, “The CORDIC Trigonometric Computing Technique”, IRE Transactions on Electronic Computers, 1959.
3. G. Golub, W. Kahan, “Calculating the singular values and pseudoinverse of a matrix”, J. SIAM Numerical Analysis, Ser. B, Vol. 2, No. 2,1965, pp. 205-224.
4. R.P. Brent, F.T. Luk, C.F. Van Loan, “Computation of the singular value decomposition using mesh-connected processors”, Journal for VLSI Computer Systems, vol. 1, no. 3, 1985, pp. 243-270
5. J.R. Cavallaro, F.T. Luk, “CORDIC Arithmetic for a SVD Processor”. Journal for Parallel and Distributed Computing, vol. 5, 1988, pp. 271-290.
6. R. Andraka, “A Survey of CORDIC Algorithms for FPGA based computers”, in FPGA '98: Proc. of sixth international symposium on Field programmable gate arrays ACM/SIGDA, 1998, pp. 191-200.
7. F. Deprettere (ed.), “SVD and signal processing. Algorithms, applications and architectures”, Department of Electrical Engineering, Delft University of Technology, Elsevier Science Publishers B.V.,Amsterdam, 1988.
8. H. Wang, P. Leray, J. Palicot, “A CORDIC-based dynamically reconfigurable FPGA architecture for signal processing algorithms,” URSI 08, The XXIX General Assembly of the International Union of Radio Science, Chicago IL, 2008.
9. VHDL, IEEE Std No. 1076, 2000.
10. Xilinx ISE Web Pack, www.xilinx.com, 2009.[11] Floating-point arithmetic, IEEE Std No. 754, 2008.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)