FPGA Implementation of Sine and Cosine Value using Cordic Algorithm

Samiha Aouissi, Mohamed Benouaret
Badji Mokhtar Annaba University
Automatic and Signals Laboratory (LASA)
Faculty of Engineering, BP: 12, Annaba, 23000, Algeria
e-mail : aouissi-samiha@hotmail.fr, mohamed.benouaret@gmail.com

Abstract— **Digital signal processing (DSP)algorithms exhibit an increasing need for the efficient implementation of trigonometric functions. The trigonometric functions like cosine and sine form the building block of certain transforms like Discrete Fourier Transform(DFT), Discrete Cosine Transform (DCT), and Fast Fourier Transform (FFT) [1]. For calculating such basic trigonometric functions, a hardware efficient algorithm is needed which will provide highest throughput and reduced latency. This paper presents the design and implementation of CORDIC algorithm for cosine and sine calculation on FPGA. Obviously, our aim is to perform this architecture to ensure the optimization design in terms of power, area and speed for efficient FPGA implementation. The circuits have been described in Very high speed integrated circuit Hardware Description Language (VHDL).**

Keywords- DSP; CORDIC algorithm ; cosine; sine;FPGA;vhdl

I. INTRODUCTION

CORDIC is an acronym for Coordinate Rotation Digital Computer. It is a method of calculating trigonometric functions, adapted to computers based on binary logic[2]. Given the prohibitive cost of multiplication in terms of area and frequency of operation, it is advantageous to limit the number to a minimum when designing. The algorithm for calculating trigonometric CORDIC calculates successive iterations by the cosine and sine functions using only multiplication or division by powers of two, which it translates into practice, shifts to the left or right .Field-programmable gate arrays (FPGAs) have also been suggested as an efficient replacement for application-specific integrated circuits (ASICs) and digital signal processors (DSPs) for applications that require a combination of high performance, low cost, and flexibility [3].

This paper is concerned with FPGA implementation for Sine and Cosine value using Cordic Algorithm, organized as follows. Section II, is reviewing the theoretical for CORDIC algorithm. In section III, FPGA design methodologies are explained Implementation of CORDIC algorithm. In section IV Results and discussion then synthesis, verifications and performance evaluation are illustrated. At last, conclusion is given in section V.

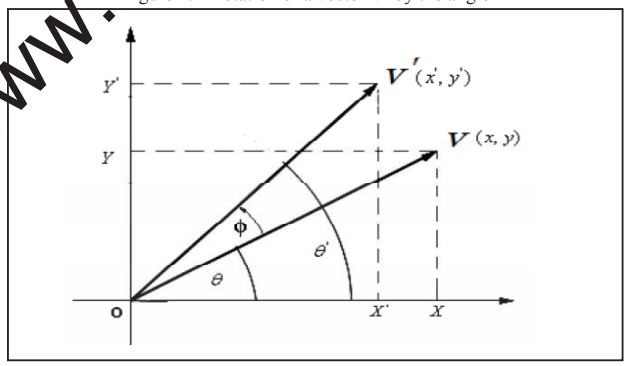
II. CORDIC OVERVIEW

The CORDIC algorithm was proposed by Volder [2] in 1959. Walther generalized this algorithm [4] for The three - coordinate system. Later some modifications have been presented [5]. The CORDIC algorithm provides an efficient mean of computing trigonometric functions by rotating a vector through some angle, specified by its coordinates[6].

First, consider the basic CORDIC algorithm as illustrated in fig.1

All the trigonometric functions can be evaluated from functions using vector rotations. The CORDIC algorithm provides an iterative method of doing vector rotations by certain angles using only shifts and adds operations. The algorithm is derived using the general rotation transform :

Figure 1. Rotation of a vector V by the angle Φ



$$X' = X\cos(\Phi) - Y\cos(\Phi) \quad (1)$$

$$Y' = Y\cos(\Phi) + X\cos(\Phi)$$

Where (X',Y') are the coordinates of the resulting vector after rotation of a vector with coordinates (X,Y) through an angle of Φ in the rectangular plane. These equations can be :

$$X' = \cos(\Phi).[X - Y\tan(\Phi)] \quad (2)$$

$$Y' = \cos(\Phi).[Y + X\tan(\Phi)]$$

Now, if the rotation angles are restricted such that : $\tan(\Phi) = 2^{-i}$, then the tangent multiplication term is reduced to a shift operation. Hence angles of rotation can be found by doing continuous smaller elementary rotations [7]. The above equation for rotation can be expressed as:

$$X_{i+1} = K_i.[X_i - Y_i\delta_i 2^{-i}]$$

$$Y_{i+1} = K_i.[Y_i + X_i\delta_i 2^{-i}] \quad (3)$$

Where, $k_i = \frac{1}{\sqrt{1+2^{-2i}}}$ and $\delta_i = \pm 1$ depending upon the earlier iteration. Taking away the scale constant from the equations yields a shift and add algorithm for vector rotation [8]. The product K_i approaches the value of 0.607312.

III. IMPLEMENTATION OF CORDIC ALGORITHM

There are so many CORDIC architectures available in the literature. The pipelined architecture has an edge over others in terms of delay and throughput. Convergence is also quite good in this architecture [9]. (fig.2) It shows the different components of the implemented cordic algorithm , how they are interconnected as well as the various control signals. A number of modules have been incorporated, and each module is responsible for one elementary Calculates [10]. During every stage within the pipelined of Sine and Cosine architecture, only adders/sub tractors are used.

The proposed architecture of CORDIC algorithm the rotation range will be [0 π/2],continuous 8 iterations are performed and fixed point 24 bit representation is used for all the registers. Initially x=0.6073 and y=0. the angle by which the vector should be rotated at a particular iteration is fixed. At each iteration, the vector is rotated to the angle nearest to the required angle[1]. The nearest angle is selected from the angles available in the lookup table as shown below.

TABLE I. LOOKUP TABLE OF ANGLE

I	$2^{-i} = \tan\alpha_i$	$\alpha_i = \arctan(2^{-i})$	α_i in radian
0	1	45°	0.7854
1	0.5	26.565°	0.4636
2	0.25	14.063°	0.2450
3	0.125	7.125°	0.1244
4	0.0625	3.576°	0.0624
5	0.03125	1.7876°	0.0312
6	0.015625	0.8938°	0.0156
7	0.007812	0.4469°	0.0078

The implementation process has been divided into seven components which are a controller, a register_x, register_y, register_z, a read only memory (ROM) or LUTs , add_subb and a shifn.The values of the angles (rad) in floating point representation are listed in table 1. Precaution should be made to avoid any overflow which might occur in the adder.

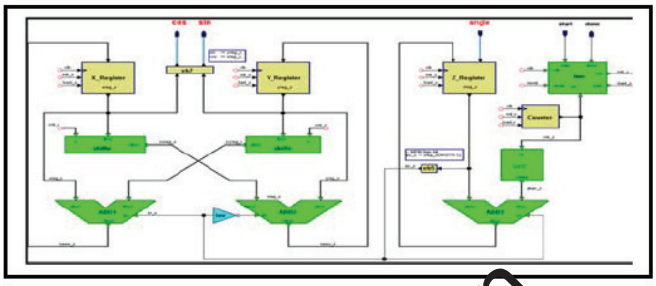


Figure 2. Serial Cordic Module in HDL Designer

For this reason, the angles which have been scaled up by 2^{22}, must be written on 2's complement using at least one extra bit than it is necessary to represent the biggest angles [11].The controller consists of a scaling accumulator and a time controller. This component is also responsible for the generation of the sampling signal (done) of the cordic algorithm and for the assertion of the signal which enables the last circuit of the algorithm to output the final result, the cosine and sine of angle if the value of counter is equal to seven. The controller is modeled by the following state machine (fig.3)

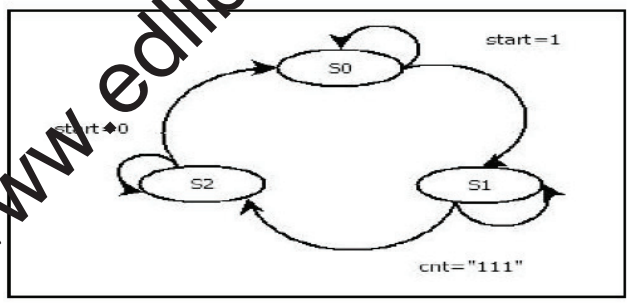


Figure 3. the state machine controller

- register_x is initialized to the value K_i = 9945 ($0.607*2^{14}$) after the precision , This component gives the cosine value at the end of processing Figure 4 is a bloc diagram generated by the RTL viewer of QuartusII.
- register_y is initialized to the value (0), and gives the sine value at the end of processing .

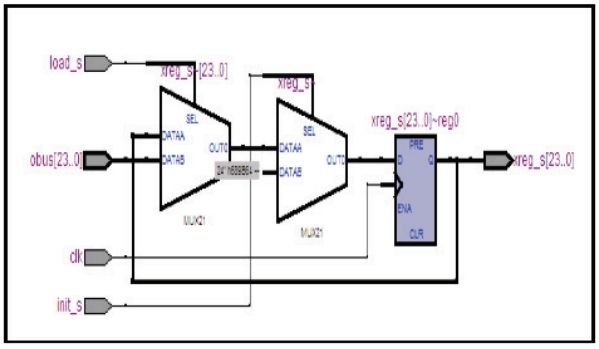


Figure 4. Schematic diagram for the register_x

The role of the ADD block is: addition or subtraction according to the value of most significant bit signal (Zreg) which is the signal (as), if (as) = 0 , δ_i is equal to "1"else δ_i is equal to "- 1" as the principle of expression (3).

The role of the shiftn block is: operation of the right shift (2^i) for the eight iteration.

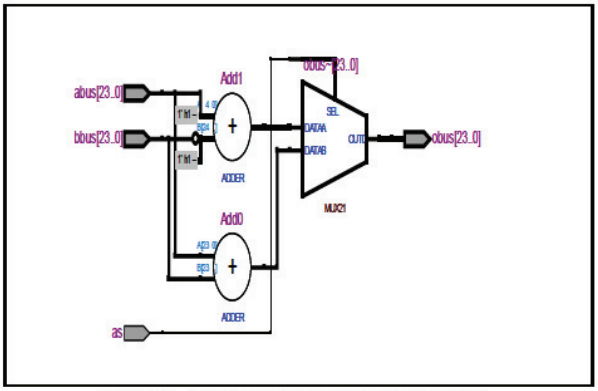


Figure 5. Schematic diagram for the block add

The delay to obtain the output (sine and cosine value) is function of the chosen resolution and number of iteration.

IV. RESULTS AND DISCUSSION

The high level description of the implementation was written in VHDL language. The simulations were run on Modelsim, and the results compared with those obtained from Matlab. The input data (angles) pattern chosen in the range [0, 6588397]. The Modelsim simulation result is shown in figure 6.

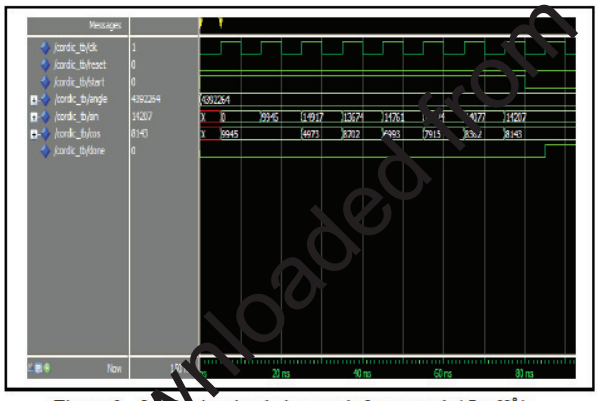


Figure 6. Modelsim simulation result for example(Φ =60˚)

The implemented cordic algorithm has a throughput of 8 clock cycles. In the example shown in Figure 6 the results simulation gives the cosine and sine of angle 60^0.the angle is converted to radians and multiply by 2^{22} to make full picture(4392264). At the end of the simulation, the result is divided by 9945 and multiply by 0.607 to obtain the correct cosine and sine (table 2).

TABLE II. EXAMPLE OF ANGLE (Φ =60)

Angle deg	*Angle rad*2^{22}*	*Cosine result*	*sine result*	*Cosine normalize*	*Sine Normalize*
60^0	4392264	8143	14207	0.5	0.86

V. HARDWARE SYNTHESIS

This section presents the hardware evaluation results of the proposed architecture for the implementation of cordic algorithm for sins and cosine value. The proposed architecture design was synthesized on a Quartus II based Altera's EP2C0F256C6 FPGA device of Cyclone II family. Fig. 7 shows the resource or component utilization of the proposed design implemented on above-mentioned FPGA kit in terms of Register, logic element and pins. The data consumed by register, logic element and pins are 1%, 2% and 50%, respectively, of total available resources. which was implemented on FPGA Programs were written in VHDL. Microsoft Excel was used for plotting the results.

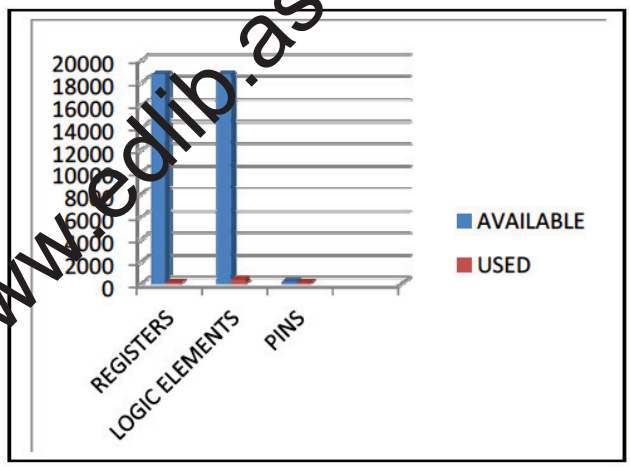


Figure 7. Resource utilization

TABLE III. PERIOD AND FREQUENCY UTILIZATION

Clock period	*7.642 ns*
Frequency	Restricted to 130.86MHZ

The result of power measurement for proposed architecture also has been measured and shown in table. 4. Hardware synthesis is done at C ° 38 ambient temperature. The total quiescent power consumed by the design was 0,07561Watt.

TABLE IV. POWER CONSUMPTION

Quartus II Version	*9.1 build 222 10/21/2009 SJ Web Edition*
Revision Name	cordic
Top-level Entity Name	Cordic_struct
Family	Cyclone II
Device	EP2C20F256C6
power models	Final
Total thermal power	75.61mw

CONCLUSION

Many features are added to the FPGAs with every generation, making it possible toper for computations at higher clock frequencies and in the most recent FPGAs, larger blocks aimed at digital signal processing (DSP) computations. 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 and 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 the 24-bit. The hardware description language (HDL) code is generated to structurally simulate and verify the functionality of the design.

REFERENCES

[1] Ms.Gadgil Amruta ,"low latency and high accuracy archtectures of cordic algorithm for cosine calculation on fpga", Cummins College of Engineering for Women Pune, India, 2008 IEEE

[2] Ramesh Bhakthavatchalu, Nisha Abdul Kareem "Comparison of Reconfigurable FFT Processor Implementation Using CORDIC and Multipliers", Dept of ECE Amrita Vishwa Vidyapeetham, 2011 IEEE.

[3] J. V. Volder, "The CORDIC trigonometric computing techniques", IRE Tranactions on Electron.Computing, vol. EC-8, pp. 330–334, Sept. 1959.

[4] J. S. Walther, "A unified algorithm for elementary functions",In Spring Joint Compuing. Conference, 1971, pp. 379–385.

[5] Roberto Sarmiento, F´elix Tobajas, Valent´ın de Monzie Nelson,"A CORDIC Processor for FFT Computation and Its Implementation Using Gallium Arsenide Technology", IEEE transactions on very large scale integration (VLSI) systems, vol. 6, no.1, march 1998.

[6] Honghao Zhang, Zhongjun Wang "Implementation of Frequency Offset Correction Using CORDIC Algorithm for 5 GHz W A N Applications",VLSI Circuit Design & Test Department Institute of Microelectronics , 2002 IEEE.

[7] G.Gopikiran"FPGA Implementation of Floatingpoint Rotation Mode CORDIC Algorithm", ECE Department National Institute of Technology,International Conference on Signal Processing, Communication, Computing and Networking Technologies (ICSCCN 2011).

[8] S. Younis, A. Al-Dweik1,"CORDIC based Architecture for Blind CFO Estimation in OFDM Systems", School of Electrical, Electronic and Computer Engineering, Newcastle University, U.K., 2011 IEEE.

[9] Magnus Nilsson, "FFT realization and implementation in fpga", Griffit University Ericsson Microwave System AB 2000/2001.

[10] Ramesh bhakthavatchalu, Parvathi Nair, "Analysis of Power Reduction Techniques Applied to Pipelined Parallel and Iterative CORDICDesign," Proceeding of International Conference on Network and Computer Science (ICNCS 2011).

[11] Saliha Harize, Mohamed Benouaret ,"A Simple and Efficient Scheme to Implement FIR Filters on FPGA" ,badji mokhtar university International Conference in khanechela ICEECA2012.