Journal of Engineering and Applied Sciences 13 (4): 842-847, 2018
ISSN: 1816-949X

Implementation of Low Power CORDIC Algorithm

K. Nithin Asok, V. Nikhil, C. Sagin Manoj, K. Shyam Sunder,
V. Vijay Bhaskar Reddy and P. Rajeswari
Department of Electronics and Communication Engineering,
Amrita School of Engineering, University of Amrita, Amritapuri,
Amrita Vishwa Vidyapeetham, Coimbatore, India

Abstract: CORDIC (Coordinate Rotation Digital Computer) is a widely used algorithm for calculating complex trigonometric functions by using shift and add operations. The CORDIC algorithm makes use of an iterative method of solving complex trigonometric functions used in vector rotation. The latest trends in VLSI technology enhance different ways of implementing CORDIC architecture. CORDIC algorithm is a practical substitution for the utilization of multipliers and is a proficient calculation that can be used for any type of architectures. Power reduction is basic in all VLSI based applications with a specific end goal to expand the quality of the design. In this project, we have implemented both folded and unfolded radix 2 CORDIC architectures. The analysis of architectures is performed and verification in terms of power has been carried out.

Key words: Folded and unfolded radix 2 CORDIC, low power design, CORDIC architecture, verification, quality, design

INTRODUCTION

CORDIC (Coordinate Rotation Digital Computer) also known as Volder's algorithm is a quick, basic and effective calculation utilized for differing digital signal processing applications. CORDIC algorithm gives an iterative strategy for performing vector rotations by arbitrary points using only shift and add operations.

The fundamental idea of CORDIC is to break down the rotation angles into the weighted aggregate of set of predefined basic angles with the end goal that the rotation through each angle can be proficient by utilizing shift and add operations. In this present technology and limitations on power, energy consumption and latency problems if we are generating any trigonometric functions by using multiplier, adder or divider, then those architecture consumes more hardware and increase in computation time. Numerous applications have developed amid the previous 50 years of CORDIC algorithm and the CORDIC has gotten more consideration after a bound together approach is proposed for its execution and after that CORDIC based figuring has been used for scientific calculator applications. Due to the simplicity of the involved operations, the CORDIC algorithm is very well suited for VLSI implementation (Bhakthavatchalu *et al*., 2010). Regular usage of CORDIC have been programming focused. In any case, the advancement in the outline of the fast Very Large Scale Integration (VLSI) Models has furnished the designers to map the CORDIC Algorithm into architecture. The CORDIC algorithm figures 2D rotation utilizing the shift and add operations. Numerous adjustments have been proposed for CORDIC algorithm for high performance and minimal effort in the practical computation of a 2D vector rotation (Sarrigeorgidis and Rabaey, 2004).

Power analysis is essential for all VLSI based applications. For reduction of this problem, a low power radix 2 CORDIC is designed. We have carried out the verilog simulations and the power analysis for both the both the folded and unfolded radix 2 CORDIC architectures to check whether which is having the low power (Schmipfle and Simon, 2007).

CORDIC ALGORITHM AND ITS MODES OF OPERATIONS

The CORDIC algorithm gives an iterative strategy that includes the rotation of a vector in linear, circular or hyperbolic coordinate systems. The rotations are performed utilizing particular rotation angles. CORDIC strategies can be utilized in two modes, namely, the rotation mode and the vectoring mode. The rotation mode is utilized to perform the general rotation by a given angle X_i though the vectoring mode processes uncertain angle

Corresponding Author: K. Nithin Asok, Department of Electronics and Communication Engineering, Amrita School of Engineering, University of Amrita, Amritapuri, Amrita Vishwa Vidyapeetham, Coimbatore, India

Z_i of a vector by performing a limited number of small scale rotations. The three difference equations of the CORDIC algorithm for an iteration is given by:

$$X_{i+1} = X_i - m\sigma_i y_i \rho^{-S_{m,i}}$$

$$X_{i+1} = \sigma_i x_i \rho^{-S_{m,i}} + y_i$$

$$Z_{i+1} = z_i - \sigma_i a_{m,i}$$

Where:

σ_i = The direction of rotation
ρ = The radix of the number system
m = The choice of the circular (m = 1), linear (m = 0) and hyperbolic (m = -1) coordinate systems
$S_{m,i}$ = The integer shift sequence
$\sigma_i = \text{sign}(z_i)$ = For rotation mode
$\sigma_i = -\text{sign}(y_i)$ = For vectoring mode

In order to maintain a constant vector length, the obtained results have to be scaled by the scale factor:

$$K = \Pi_i K_i$$

Where:

k_i = The elementary scaling factor of the ith iteration
K = The resulting scaling factor after n iterations

The calculation of the scaling component and its compensation expands the equipment utilization relying upon the number system utilized as a part of the CORDIC. With proper estimation of x, y and z values both rotation and vectoring mode can be used to calculate frequently used elementary functions. In rotation mode, the input angle θ will be disintegrated utilizing a finite number of basic angles:

$$\theta = \sigma_0 a_0 + \sigma_1 a_1 + \ldots \sigma_{n-1} a_{n-1}$$

Where:

n = The number of small scale rotations
σ_i = The elementary angle for the ith iteration
σ_i = The direction of ith rotation
Z_0 = The angle accumulator instated with the input rotation angle

The direction of each vector is resolved to lessen the magnitude of the residual angle in the angle accumulator. The vectoring mode rotates the input vector through a predefined set of n elementary angles in order to diminish the y coordinate of the last vector to zero as nearly as could be expected under the circumstances and the direction of rotation in each emphasis will be based on the sign of the y coordinate.

CORDIC ARCHITECTURES

CORDIC architectures are essentially classified into folded and unfolded relying upon the equipment acknowledgement of the 3 iterative equations. Certain factors like carry/borrow propagate additions and some shifting operations makes the traditional CORDIC moderate for fast applications. These downsides were overcome by unfolding the iteration process so that each of the processing components perform same iteration (Yadav and Singh, 2013). Be that as it may, the unfolded design expends more power since, it is utilizing more shift and add operations. From the architecture of the CORDIC processor it is evident that the CORDIC core is the heart of the CORDIC processor. By copying the iterative equations of the CORDIC algorithm into equipment and time multiplexing all iterations into a single functional unit, we will get a folded architecture. For example, executing N multiplication operations utilizing a single multiplier relating to a folding factor of N and the request in which these multipliers are executed over a time of N cycles indicated by a folding set (Andraka, 1998). The decision of the folding set is basic to the execution of the folded structure since a suitable folding can prompt design with low power and area than others.

FOLDING OF RADIX 2 CORDIC

To design a low power CORDIC, we are time multiplexing many algorithm operations such as adders and multipliers into a single functional unit and this processing is called folding (Volder, 1959). In each iteration of unfolded architecture, 3 adders are used each serving the purpose of updating the x, y and z values, respectively. Hence, we have time multiplexed a single adder so as to serve the purpose of the 3 adders which are used in the unfolded architecture.

Now to time multiplex the adders, we are generating a single adder block and using multiplexers and latches at each clock cycles leaving the necessary folding delays, the inputs and outputs of the adder blocks are changed, so that, 3 operations can be carried out. At first clock cycle, the first inputs are fed into the adder so that the first operation can be carried out and stored in the output register. Similarly, the remaining two add/subtract operations can be carried out in the subsequent clock cycles. Even though the power consumed is less, there is a delay in getting the output. This is due to folding.

From Fig. 1, we can see that there is only one add/sub block used and it can carry out 3 operations at subsequent clock cycles.

COMPLETE IMPLEMENTATION OF UNFOLDED AND FOLDED RADIX 2 CORDIC

To implement the unfolded architecture, for each iteration, we need 3 adders and 2 barrel shifters and some registers wired in between them. The ZR values that is the angular values from the rotation mode are taken from the register already stored. In this way, we will have the output of each iteration. Then at last using IF ELSE loop, we will check in each iteration, the value becomes close to zero for the first time and then assign the value to the output register, respectively.

In the case of folded architecture, the modules used to drive the output x, y and z values according to the corresponding inputs are same as the unfolded architecture except a folded adder will be defined here. Actual folding takes place in the folded adder module. In this module, three input pairs for which the calculation are to be done is given as input and output will be the outputs of the operation on the three pair of inputs. Here we first generate a add/sub block once at the start of the code and define a counter, so that we can give necessary folding delays before each level. Then at different values of the counter, the input registers and the output registers will be multiplexed with the generate add/sub block, so the three registers will receive the outputs at different clock cycles. Then as in the case of unfolded architecture, all the shifter modules along with the folded adder module will be wired and the inputs x, y and z values with their sign and the outputs will be updated x, y, z at each iteration. At last, using a IF ELSE loop, we will check in which iteration the value becomes close to zero and then assign the value to the output register, respectively.

SIMULATIONS AND RTL SCHEMATIC FOR UNFOLDED AND FOLDED RADIX 2 CORDIC

Output Values for the test inputs, x = 5, y = 0 and z = 65. For these inputs, we should get the output as x = 3.51 and y = 7.42. Figure 2 shows the unfolded radix 2 CORDIC simulation in Verilog.

Figure 2 represents the first iteration for unfolded radix 2 CORDIC algorithm. From Fig. 3, it is evident that 3 adders and 2 shifters are used in the architecture. Similarly in the case of folded radix 2 CORDIC, we are giving the same test inputs as in Unfolded CORDIC and the verilog simulation is shown in Fig. 4.

Figure 5 represents the first iteration of folded radix 2 CORDIC. Here, only 1 ALU block is present with some multiplexers and latches compared to the 3 ALU block present in the first iteration of the unfolded architecture.

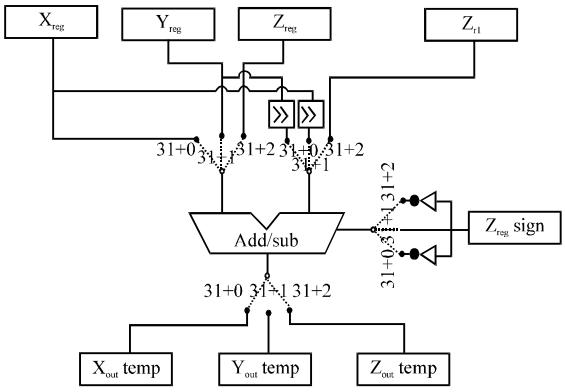

Fig. 1: Folded radix 2 CORDIC architecture

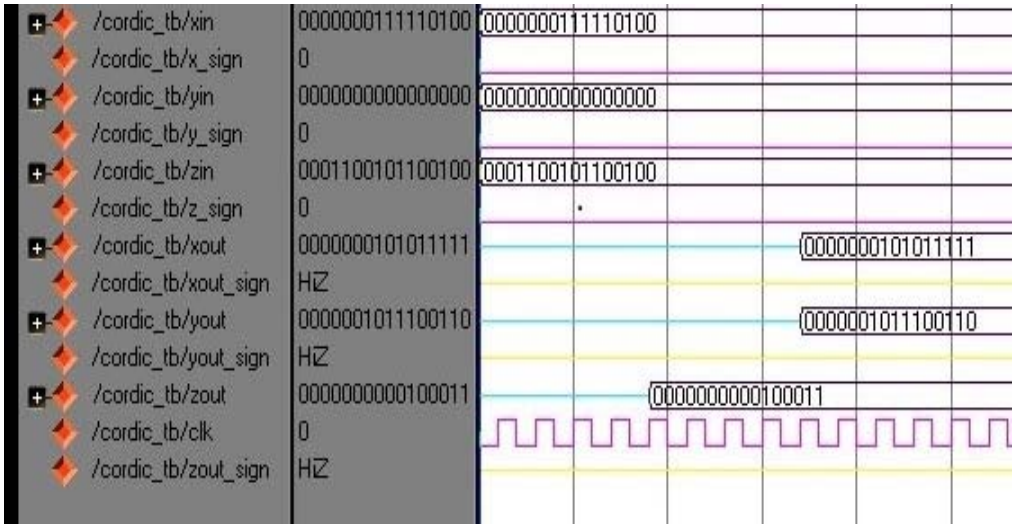

Fig. 2: Verilog simulation for unfolded radix 2 CORDIC

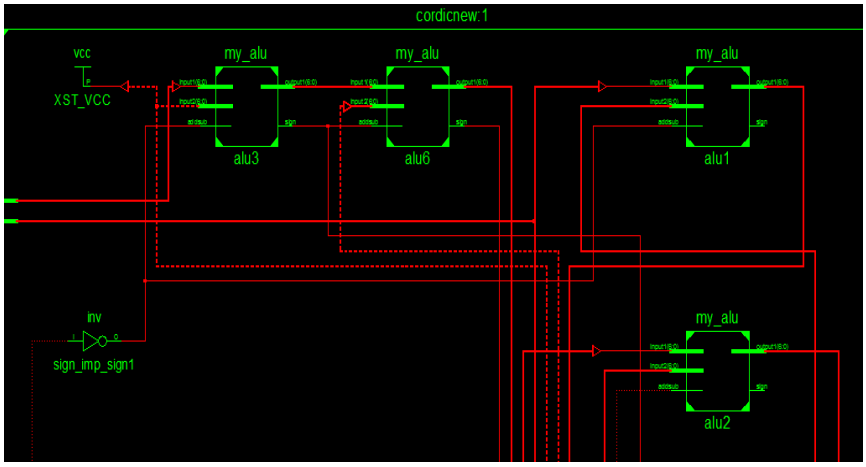

Fig. 3: RTL schematic for unfolded radix 2 CORDIC

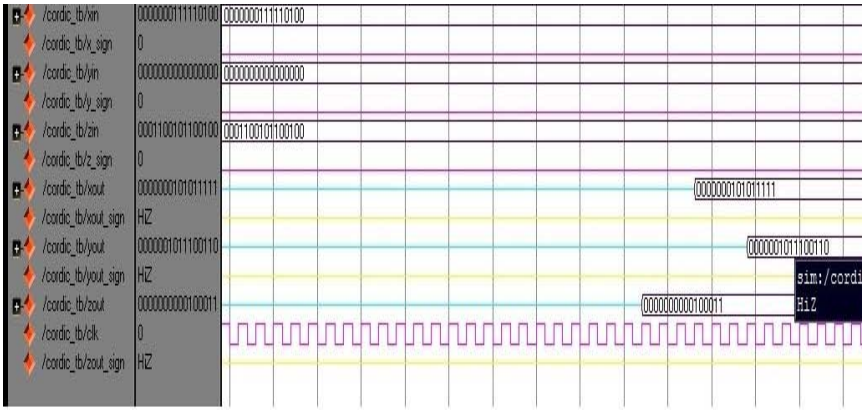

Fig. 4: Verilog simulation for folded radix 2 CORDIC

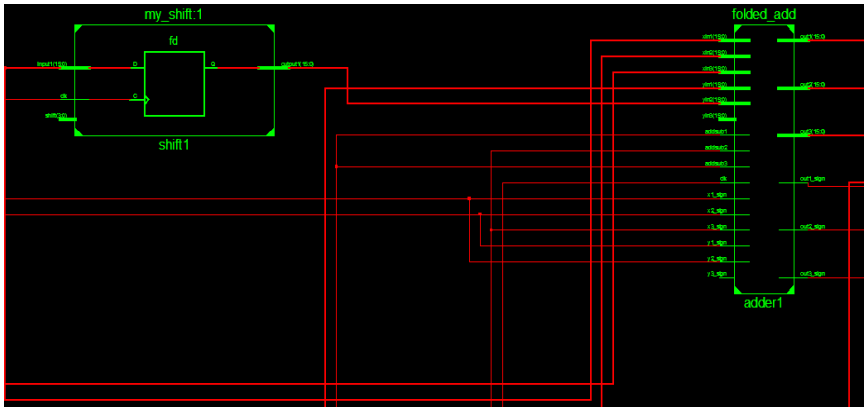

Fig. 5: RTL schematic for folded radix 2 CORDIC

POWER ANALYSIS FOR FOLDED AND UNFOLDED RADIX 2 CORDIC

Power analysis for both architectures is done to check which one is having the low power. From Fig. 6 and 7, we can see that the total power and the logic used in the architecture is considerably less for the folded radix 2 CORDIC. So, folding of the radix 2 CORDIC contributes to the low power CORDIC algorithm.

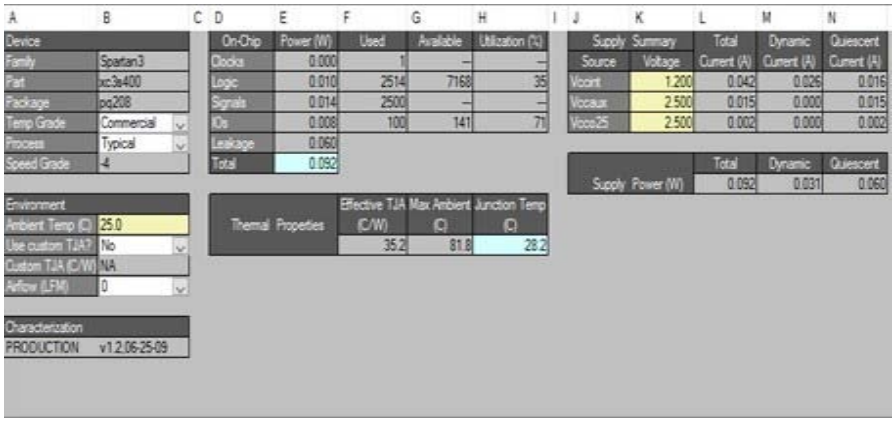

Fig. 6: Power analysis for unfolded radix 2 CORDIC

Fig. 7: Power analysis for folded radix 2 CORDIC

CONCLUSION

In present technology, everything is based on online like movies, shopping, learning, etc. To perform these applications, we need networking, internet of things and various other supportive systems and these systems are based on complex mathematical functions such as trigonometric functions. These functions can be easily calculated by using CORDIC algorithm. From this, it is evident that this project about implementing low power CORDIC algorithm has greater importance and lots of future scope.

The first and foremost objective is to implement an unfolded radix 2 CORDIC architecture. Verilog simulations up to 10 iterations for the above objective are done and RTL schematic of the iterations is obtained.

Folding of the radix 2 CORDIC falls under next objective and it is an important one since it reduces the silicon area time and decreases the power by multiplexing all algorithm operations into a single functional unit. Verilog simulations for the CORDIC architecture using folding adders is done and it can be generalized for all iterations and we observed by power analysis that implementing CORDIC by folding architecture consumes less power.

Even though, power consumed becomes less, there are some drawbacks such as latency delays. That is, the output will be received much greater than that usually taken by the unfolded architecture. Similarly, 10 iterations can be folded for even more better low power CORDIC.

REFERENCES

Andraka, R., 1998. A survey of CORDIC algorithms for FPGA based computers. Proceedings of the 1998 ACM-SIGDA Sixth International Symposium on Field Programmable Gate Arrays, February 22-25, 1998, ACM, Monterey, California, ISBN:0-89791-978-5, pp: 191-200.

Sarrigeorgidis, K. and J. Rabaey, 2004. Ultra low power CORDIC processor for wireless communication algorithms. J. VLSI. Signal Process., 38: 115-130.

Schmipfle, C. and S. Simon, 2007. Low power CORDIC implementation using redundant number representation. Master Thesis, Technical University of Munich, Munich, Germany.

Volder, J.E., 1959. The CORDIC trigonometric computing technique. IRE Trans. Electron. Comput., 8: 330-334.

Yadav, P. and K. Singh, 2013. A paper on CORDIC algorithm and its applications for current technology. Intl. J. Sci. Res., 1: 770-773.