International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol.8, No.12 (2015), pp.247-258
http://dx.doi.org/10.14257/ijsip.2015.8.12.24

Research on the Design of CORDIC Vector Magnitude Calculator by using Model-Based Design

Jianying Cao, Xiaoxia Zheng and Yang Nie

Department of Physics, Jining Normal University, Inner Mongolia, China
514716928@qq.com

Abstract

Vector magnitude calculation is significant in the QR-algorithm by Coordinate Rotation Digital Computer (CORDIC), which is increasingly used in adaptive applications. However, its hardware implementation is extremely difficult. In this paper, a fixed point CORDIC system is constructed to compute the magnitude of a vector by using Model-Based Design. The simulation results show that the proposed method of CORDIC vector magnitude calculator is not only simple in structure and easy to implement, and can improve the speed of operation with good scalability.

Keywords: *Vector Magnitude; CORDIC; Model-Based Design; System Generator*

1. Introduction

CORDIC was originally presented by Jack E. Volder for the computation of trigonometric functions, multiplication and division in 1959 [1]. In 1971, Walther unified the circular system, linear system and hyperbolic system to an iterative equation, and proposed the algorithm of CORDIC [2]. In these years, not only a wide variety of applications of CORDIC have emerged, but also a lot of progress has been made in the field of algorithm design and development of architectures for hardware solutions to those applications.

The algorithm of CORDIC is widely used for efficient and low-cost implementation of a large class of applications, which include the generation of trigonometric, logarithmic and transcendental elementary functions; complex number multiplication, eigenvalue computation, matrix inversion, solution of linear systems and singular value decomposition for signal processing[3]. In a broad way, the algorithm of CORDIC is a kind of mathematical calculation method. It can be decomposed into a series of addition and subtraction and shift operations, so it is very suitable for hardware implementation. One significant application of CORDIC vector magnitude calculations is the QR-algorithm, and the hardware implementation of QR is a triangular array, and requires that an incoming vector is subject to rotation. These rotations are performed by a subset of cells within a QR array, and the angle of rotation is computed by CORDIC.

Although, the computation of vector magnitude is simplified by CORDIC in the QR-algorithm, however, the model of the algorithm is still a very complicated process. With the development of electronic technology, the concept of design is also developing rapidly. The method of Model-Based Design (MBD) [4-6] is a kind concept of system level design. In the MBD, a system model is at the center of the development process, from requirements development, through design, implementation, and testing. This method is more competent for the modeling and implementation of complex algorithms and is used in much motion control, industrial equipment, aerospace, and automotive applications.

The remainder of this paper is structured as follows. In Section 2, the principles of CORDIC operation are discussed, which are included the elementary ideas from coordinate transformation to rotation mode and vectoring mode. Realization architectures of CORDIC algorithms are expounded also in Section 2. In Section 3, the method of

ISSN: 2005-4254 IJSIP
Copyright ⓒ 2015 SERSC

Model-Based Design is introduced. A fixed point CORDIC system is constructed to compute the magnitude of a vector in Section 4. Finally, concluding remarks are drawn in Section 5.

2. The Basic Principle of CORDIC

In this Section, the basic principle of CORDIC is discussed and presented its iterative algorithm for different operating modes. At the end of this Section, the hardware implementation architecture of the three CORDIC algorithms is introduced.

2.1. Rotating Mode

As shown in Figure 1, the rotation of a two-dimensional vector $P = [x_P, y_P]$ through an angle α, to obtain a rotated vector $Q = [x_Q, y_Q]$ could be performed by the matrix product $Q = WP$, where W is the rotation matrix:

$$W = \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix} \tag{1}$$

By factoring out the cosine term in (1), Q can be rewritten as

$$\begin{cases} x_Q = \cos\alpha\,(x_P - y_P \tan\alpha) \\ y_Q = \cos\alpha\,(y_P + x_P \tan\alpha) \end{cases} \tag{2}$$

It can be seen that $\cos\alpha$ is just a length factor of the vector. If we now drop the $\cos\alpha$ term, as shown in Figure 2, then we have a pseudo-rotation [7]. Note that we have no mathematical justification for dropping the $\cos\alpha$ term. We drop it for convenience of implementation, and *R* can be rewritten as

$$\begin{cases} x_R = x_P - y_P \tan\alpha \\ y_R = y_P + x_P \tan\alpha \end{cases} \tag{3}$$

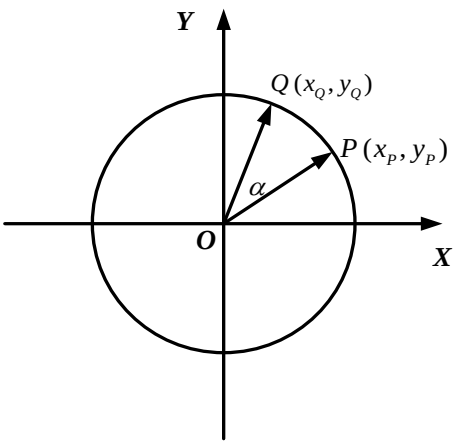


Figure 1. Rotation of Vector

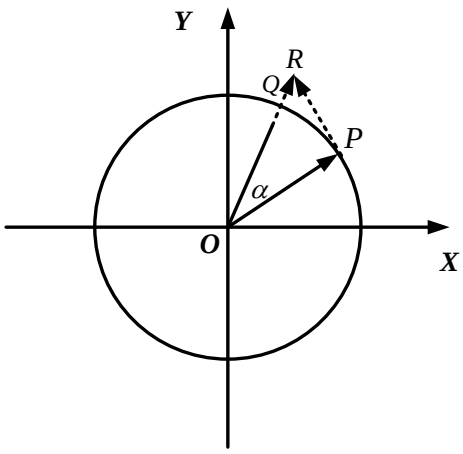


Figure 2. Pseudo-rotation of Vector

For pseudo-rotation, CORDIC performs α by a certain number of micro-rotations through angle $\Delta\alpha_i$, where

 Copyright ⓒ 2015 SERSC

$$\alpha = \sum_{i=0}^{N-1} \sigma_i \Delta\alpha_i \text{, and } \sigma_i = \pm 1 \tag{4}$$

in this way, a rotation is decomposed into a series of micro-rotations. The vector in iteration i is rotated by some angle α_i and the next rotation is made by an angle $\Delta\alpha_i$ that brings the vector to new angle α_{i+1}, then:

$$\begin{cases} x_{i+1} = x_i - y_i \tan\alpha_i \\ y_{i+1} = y_i + x_i \tan\alpha_i \end{cases} \tag{5}$$

To avoid multiplication in computing, the α_i is set as:

$$\alpha_i = \tan^{-1}(d_i 2^{-i}) \tag{6}$$

Where $d_i \in \{-1, 1\}$, then (5) can be rewritten as

$$\begin{cases} x_{i+1} = x_i - d_i y_i 2^{-i} \\ y_{i+1} = y_i + d_i x_i 2^{-i} \end{cases} \tag{7}$$

It can be seen that every time the micro rotation only need to add, subtract and shift operation. At this stage we introduce a third equation called the angle accumulator, which is used to keep track of the accumulative angle rotated at each iteration:

$$z_{i+1} = z_i - d_i \tan^{-1} 2^{-i} \tag{8}$$

At this point, the iterative equation of the circular system of CORDIC can be expressed as:

$$\begin{cases} x_{i+1} = x_i - d_i y_i 2^{-i} \\ y_{i+1} = y_i + d_i x_i 2^{-i} \\ z_{i+1} = z_i - d_i \tan^{-1} 2^{-i} \end{cases} \text{ and } d_i = \begin{cases} +1 & z_i \geq 0 \quad \text{for positive rotation} \\ -1 & z_i < 0 \quad \text{for negative rotation} \end{cases} \tag{9}$$

After n ($n \to \infty$) rotation, the final result is

$$\begin{cases} x_n = A_n(x_0 \cos z_0 - y_0 \sin z_0) \\ y_n = A_n(y_0 \cos z_0 + x_0 \sin z_0) \\ z_n = 0 \\ A_n = \prod_{i=0}^{n-1} \sqrt{1 + 2^{-2i}} \end{cases} \tag{10}$$

Where $n \to \infty$, $A_n \approx 1.646760258$. If $x_0 = 1/A_n$ and $y_0 = 0$, z_0 is the target rotation angle, the sine and cosine functions of rotating angle can be obtained by iterative according to (11).

$$\begin{cases} x_n = \cos z_0 \\ y_n = \sin z_0 \\ z_n = 0 \\ A_n = \prod_{i=0}^{n-1} \sqrt{1 + 2^{-2i}} \end{cases} \tag{11}$$

Copyright ⓒ 2015 SERSC

The sine function of rotating angle can be implemented with 12 bit fixed-point by the algorithm of CORDIC, and simulation results are shown in Figure 3 and Figure 4.

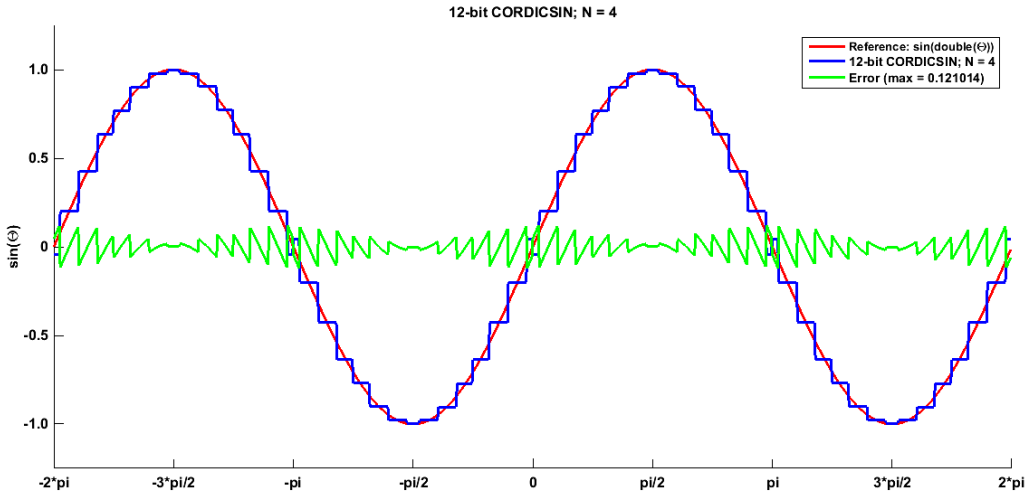


Figure 3. The *sin* Function by CORDIC, Iteration Number N=4

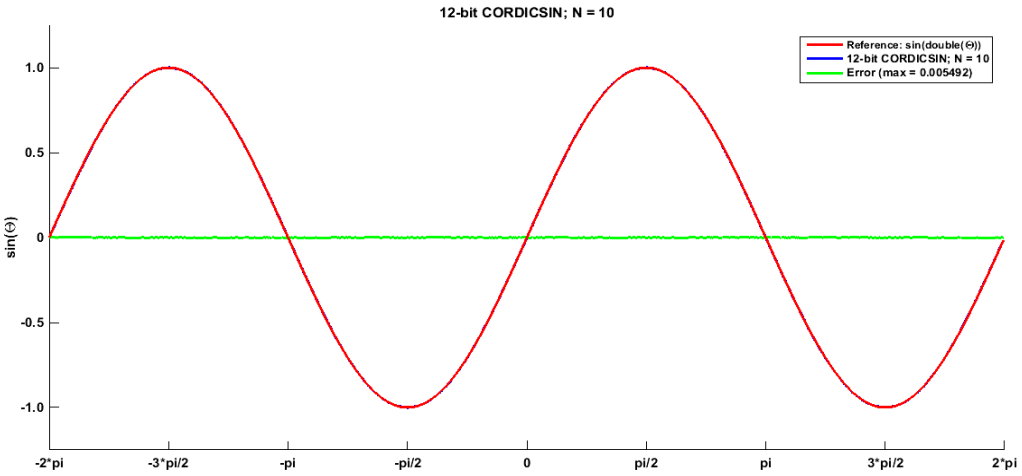


Figure 4. The *sin* Function by CORDIC, Iteration Number N=10

2.2. Vectoring Mode

In vectoring mode, the CORDIC algorithm is mainly used to realize the conversion of rectangular coordinates to polar coordinates [8]. In rotation mode, we drive x towards 0, but we drive y towards 0 in vectoring mode. In order to achieve this goal, the direction of rotation is determined by y symbol in the process of iteration and the initial vector rotation to the positive axis of X finally. The diagram of vector rotation is shown in Figure 5, and the corresponding iterative equation is defined as

$$\begin{cases} x_{i+1} = x_i - d_i y_i 2^{-i} \\ y_{i+1} = y_i + d_i x_i 2^{-i} \\ z_{i+1} = z_i - d_i \tan^{-1} 2^{-i} \end{cases} \quad \text{and} \quad d_i = \begin{cases} +1 & y_i \le 0 \\ -1 & y_i > 0 \end{cases} \tag{12}$$

Copyright ⓒ 2015 SERSC

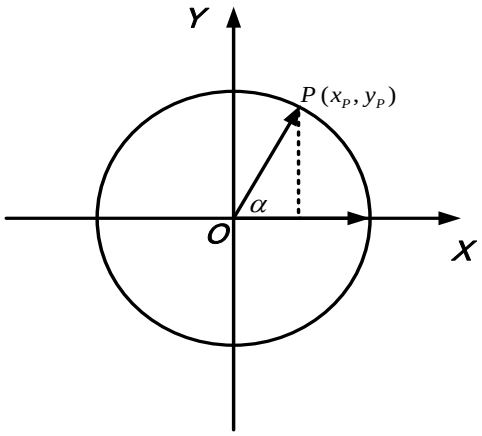


Figure 5. The Diagram of Vector Rotation

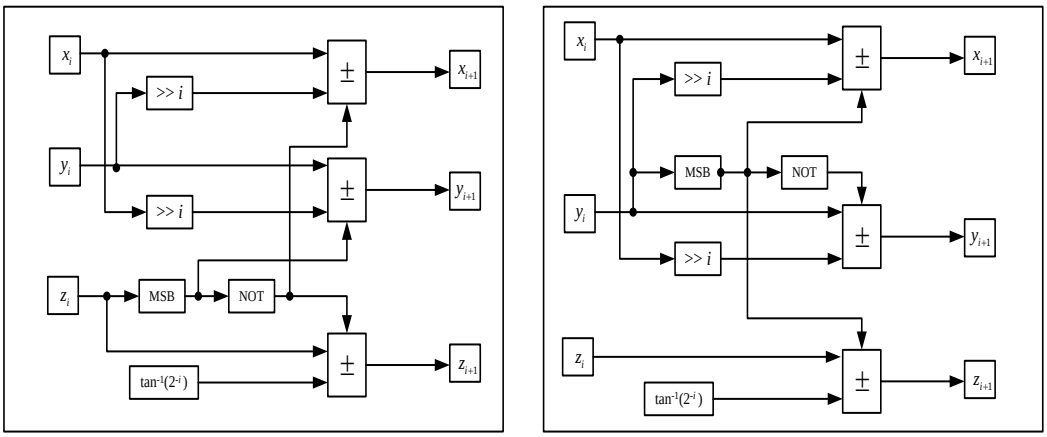
After n ($n \to \infty$) rotation, the final result is

$$\begin{cases} x_n = A_n \sqrt{x_0^2 + y_0^2} \\ y_n = 0 \\ z_n = z_0 + \tan^{-1}(y_0 / x_0) \\ A_n = \prod_{i=0}^{n-1} \sqrt{1 + 2^{-2i}} \end{cases} \tag{13}$$

Based on expression (13), if x is used for the real part of the complex, and y is used for the imaginary part of the complex, then the modulus of a complex number can be calculated.

2.3. The Hardware Structure of CORDIC

The hardware structure of the basic CORDIC unit can be derived from (9) and (12), and it is shown in Figure 6. It can be seen that the basic unit consists of three adder, one LUT and two shift register from Figure 6, which is the advantage of CORDIC.



(a) The Unit of Rotating Mode (b) The Unit of Vectoring Mode

Figure 6. The Processing Unit of Circular System of CORDIC

Copyright ⓒ 2015 SERSC

Since iterative processing units are the basic same, just shift amount and storage angle are different, so three hardware implementation architecture of CORDIC can be generalized, which include serial structure, parallel structure and parallel pipeline structure [9-10]. Three kinds of structure are illustrated in Figure 7.

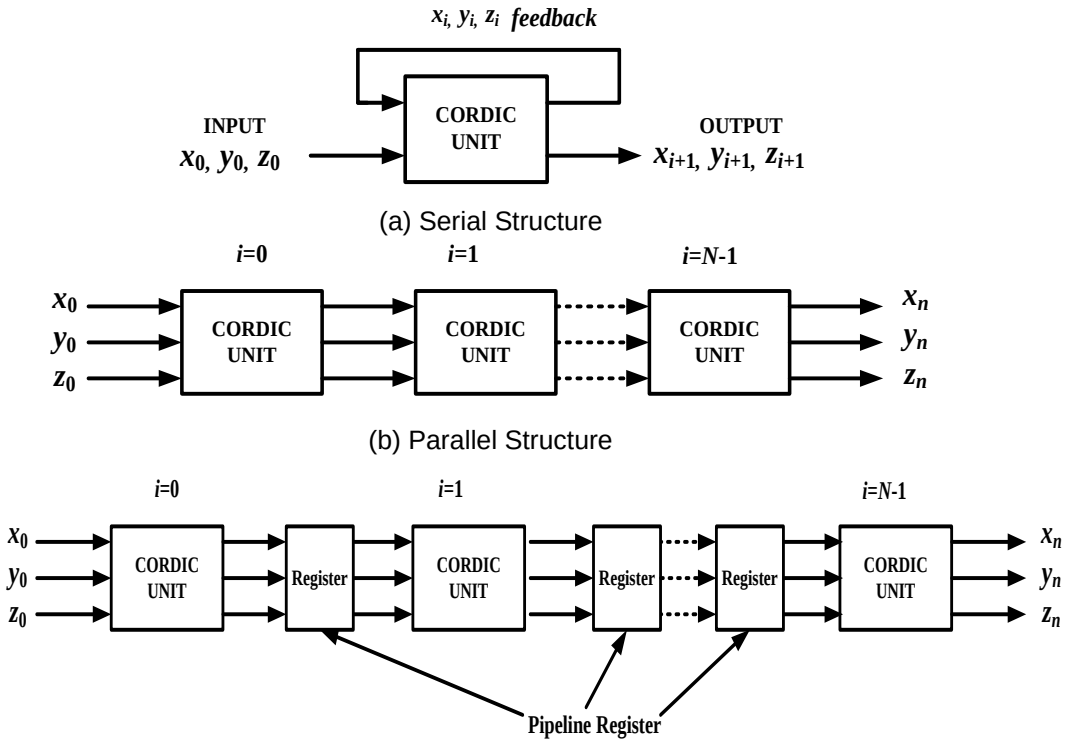


Figure 7. Three Hardware Implementation Architecture of CORDIC

The ideal CORDIC architecture depends on speed and area tradeoffs in the intended application. Alternative implementation options will be presented, and the cost and performance tradeoffs highlighted. In the serial structure, all CORDIC iterations performed using a single cell. This structure has the advantage of expanding fewer resources, but the timing control is complex and the processing speed of the system is low. The parallel structure is an array of cells, and it is an extension of the serial structure. The resource consumption of parallel structure is significantly increased, but the processing speed of the system is relatively fast. In order to improve the processing speed and to shorten the critical path, the pipeline register is added in the parallel structure, which is called parallel pipeline structure. Therefore, pipeline parallel structure has the fastest processing speed with consuming more resources.

3. The Introduction of Model-Based Design

With the improvement of the safety and reliability of the electronic products, the code amount of FPGA and embedded system is a trend of explosive growth. However, the development of the work flow in the initial form of artificial hand written code, and it has become increasingly unable to adapt to the millions of lines of code level complex system development. Therefore, the concept of MBD has been proposed [11].

MBD workflow is shown in Figure 8. The boxes represent different stages of design. The vertical arrows represent different stages of evolution. The right side of the curved

 Copyright ⓒ 2015 SERSC

arrow indicates the verification and validation process. With use of MBD related tools, it can ensure correct evolution in the design stage by continuous testing and verification in the evolution process.

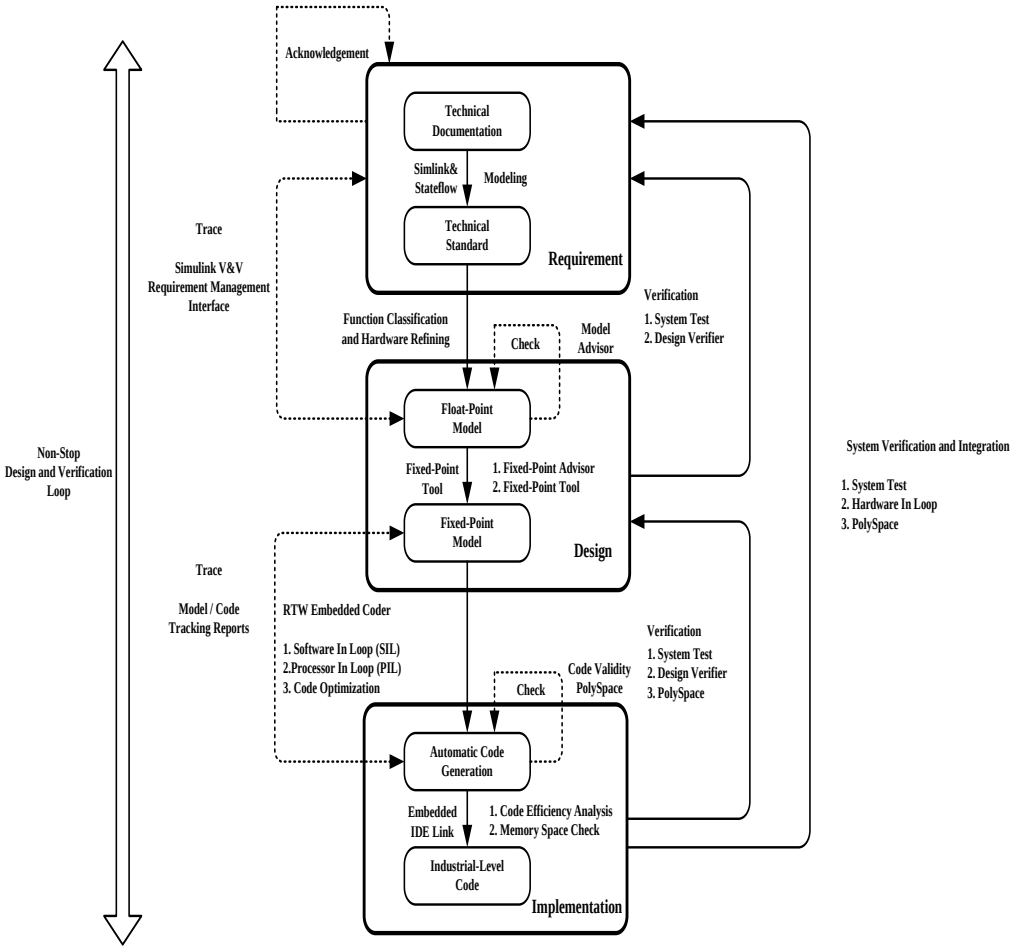


Figure 8. The Workflow Based on MBD

In a typical design flow a clean architectural idea cannot be captured as the engineer might hope, since the implementation intrudes into the design. This mixture of architecture and implementation has a particularly undesirable impact on portability, since the implementation detailed for one FPGA family will differ from those of another family. Even within speed-grades of the same FPGA family different pipelining will be required to meet the same clock frequency. Maintainability is reduced too, and this can dilute the benefits of an integrated tool flow.

System Generator is the development tool based on MBD, it is not independent software, but embedded in Simulink. The tool will automatically generate hardware description language (HDL) code, which is mapped to the FPGA device and the workflow is shown in Figure 9. Therefore, the designer can define a model representation of the system level design, and it is easy to convert the model design into a gate level representation.

Copyright ⓒ 2015 SERSC

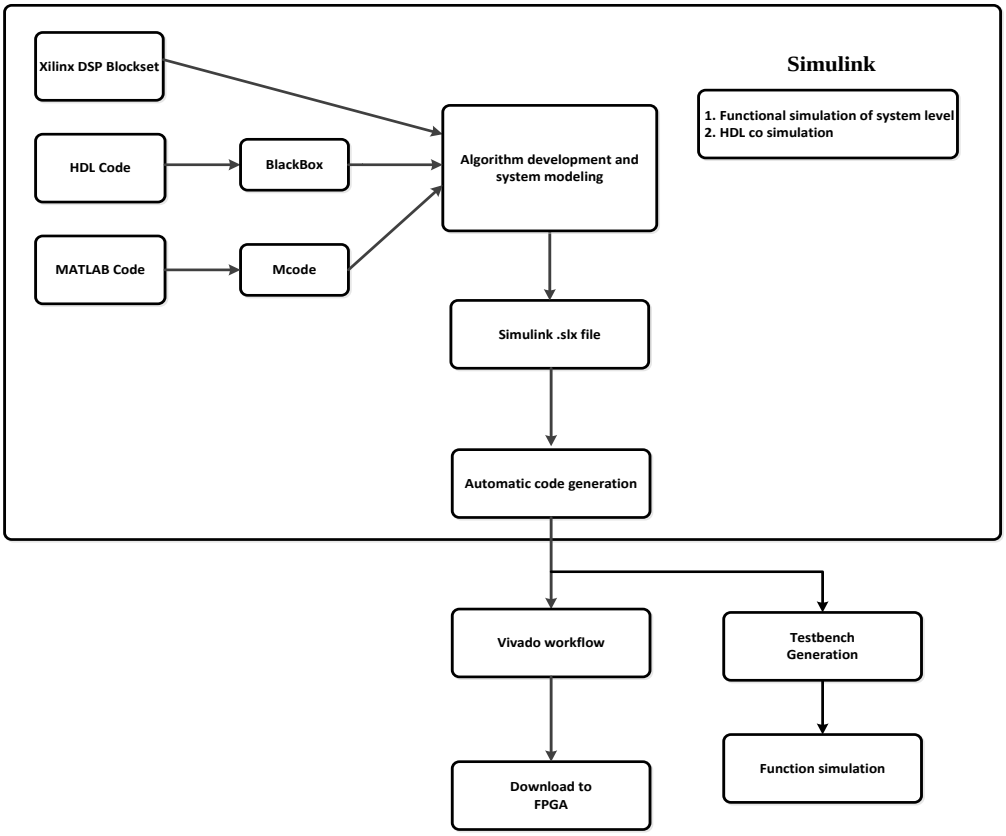


Figure 9. The Workflow based on System Generator

4. The Design of CORDIC Vector Magnitude Calculator

One significant application of CORDIC vector magnitude calculations is the QR-algorithm, which is increasingly used in adaptive applications. In this section we will design and verify a fixed point CORDIC system to compute the magnitude of a vector to a desired accuracy. For the purposes of this example, a modest level of precision is specified using MBD, but bear in mind that far greater accuracy is required for many practical applications.

The ideal outcome is to achieve the desired level of accuracy with the least hardware cost, and therefore knowledge of the precision offered by combinations of method is important. The magnitude of a vector can be computed by CORDIC.

$$|v| = \sqrt{x^2 + y^2} \tag{14}$$

It is important to consider the accuracy of the CORDIC algorithm when computing $|v|$. The appropriate parameters choose must be provided for a desired accuracy. However, the precision of a CORDIC calculation depends on two factors: the number of fractional bits, and the number of iterations. To compute the magnitude of a vector, CORDIC is used in the circular coordinate system and in vectoring mode. When calculating vector magnitude, the *y* output is expected to approach 0 and is not required. The *z* data path is not needed either.

The *x* output will then generate the following result:

$$x = K_n \sqrt{x^2 + y^2} \tag{15}$$

 Copyright ⓒ 2015 SERSC

Clearly the scaling factor K_n must be removed. This can be achieved by multiplying the *x* output by $1/K_n$. The value of K_n is dependent on the number of iterations which is known before hand and thus can be pre-computed according to:

$$K_n = \prod_{i=0}^{n-1} \sqrt{1 + 2^{-2i}} \tag{16}$$

The CORDIC algorithm is designed by System Generator, which compute vector magnitude with an accuracy of 10 effective fractional bits. It is assumed that the *x* and *y* inputs are constrained to ± 0.5, and the system model diagram is shown in Figure 10. The upper half of the whole model is floating point model, and the value of vector magnitude floating point is 0.3126. The lower half of the whole model is the fixed point model, and six processing unit is used to constitute a parallel structure for computing vector magnitude. The value of vector magnitude fixed point is 0.3124, and the comparison result of the floating point and fixed point is shown in Figure 11.

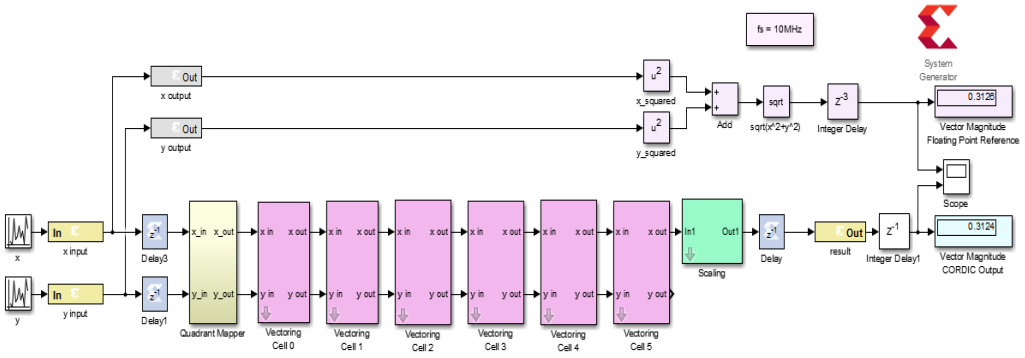


Figure 10. The Computing Model of Vector Magnitude by System Generator

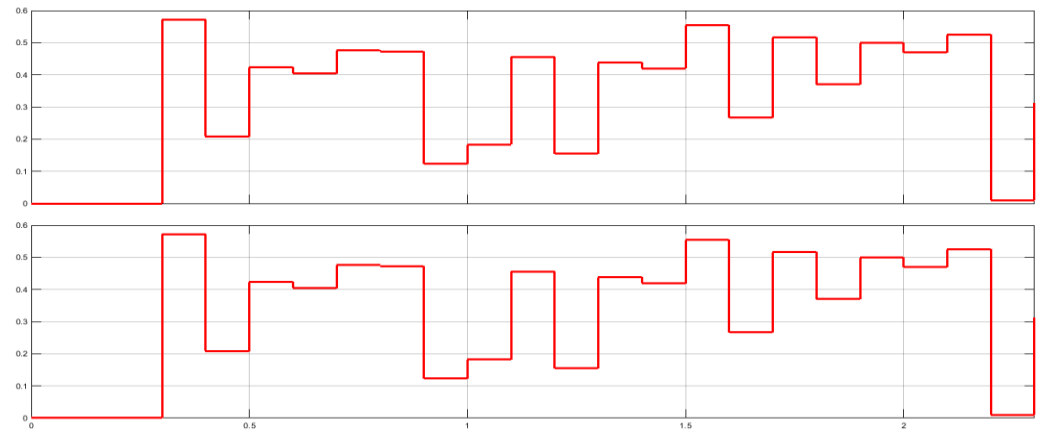


Figure 11. The Comparison Result of the Floating Point and Fixed Point

5. Conclusion

This paper completed the calculation of the vector magnitude using the CORDIC algorithm with MBD. In the MBD, users can complete a suitable design which could generate high reliability code for specification. The whole workflow of design starts from algorithm to realization can be done without written any hardware and software code. The

Copyright ⓒ 2015 SERSC

results of the simulation show that the design of the fixed point is very close to the floating point design. This innovative design ideas and development process is not only the new trend of workflow, but also improves the efficiency and security of the generated code. In addition, this design process ensures that the design performance meets different requirements. It is suggested as a new design solution, which is particularly suitable for embedded hardware development. Therefore, MBD will become a new trend in the development of FPGA.

Acknowledgments

We would like to thank professor GE for stimulating discussions with respect to the topic of this paper and laboratory equipment. Moreover, we greatly appreciate the reviewers' comments that lead to an improved presentation of the results. This work was supported by Research Program of science and technology at Universities of Inner Mongolia Autonomous Region (NJZZ14288).

References

[1] J. E. Volder, "The CORDIC trigonometric computing technique", IRE Trans. Electron. Comput., vol. EC-8, **(1959)**, pp. 330-334.
[2] J. S. Walther, "A Unified Algorithm for Elementary Functions", Proc. of Spring Joint Computer Conference, **(1971)**, pp. 379-385.
[3] P. K. Meher, J. Valls, T. Juang, K. Sridharan and K. Maharatna, "50 years of CORDIC: algorithms, architectures and applications", IEEE Transactions on Circuits and Systems I, vol. 56, no. 9, **(2009)**, pp. 1893-1906.
[4] M. F. Costabile, D. Fogli, P. Mussion and A. Piccinno, "Visual interactive systems for end-user development: A model-based design methodology", IEEE Trans. Syst., Man, Cybern. A, Syst, Humans, vol. 37, no. 6, **(2007)**, pp. 1029-1046.
[5] S. SanchezSolano, M. BroxJimenez, E. delToro, P. BroxJimenez and I. Baturone, "Model-based design methodology for rapid development of fuzzy controllers on FPGAs", IEEE Trans. Ind. Informat, vol. 9, no. 3, **(2013)**, pp. 1361-1370.
[6] I. Skrjanc, "Evolving Fuzzy-Model-Based Design of Experiments With Supervised Hierarchical Clustering", IEEE Transactions, vol. 23, no. 4, **(2015)**, pp. 861-871.
[7] S. Wang, V. Piuri and J. E. E. Swartzlander, "Hybrid CORDIC algorithms", IEEE Transactions on Computers, vol. 46, no. 11, **(1997)**, pp. 1202-1207.
[8] S. F. Hsiao and J. M. Delosme, "Householder CORDIC algorithms", IEEE Transactions. on Computers, vol. 44, no. 8, **(1995)** pp. 990-1001.
[9] E. Antelo, J. Villalba, J. D. Bruguera and E. L. Zapatai, "High performance rotation architectures based on the radix-4 CORDIC algorithm", IEEE Transactions on Computers, vol. 46, no. 8, **(1997)** pp. 855-870.
[10] S. F. Hsiao, Y. H. Hu and T. B. Juang, "A memory-efficient and highspeed sine/cosine generator based on parallel CORDIC rotations", IEEE Signal Processing Letters, vol. 11, no. 2, **(2004)**, pp. 152-155.
[11] J. Liu, "Model based design workflow for FPGA compliance with DO-254 standard", Information Technology in Medicine and Education on (ITME), 2012 International Symposium on, vol. 2, **(2012)**, pp. 1026-1030.

Authors

Jianying Cao received the M.S. degree in Electronics and Communication Engineering from Inner Mongolia University, China. She is presently Lecturer in the department of physics at Jining Normal University, China. Her main research interests include Electronic Information and Physics Teaching Theory.

 Copyright ⓒ 2015 SERSC

Xiaoxia Zheng received the M.S. degree in College of Computer Science from Inner Mongolia University, China. She is presently Lecturer in the department of physics at Jining Normal University, China. Her main research interests include single-chip application technology, embedded teaching and research work.

Yang Nie is a Ph.D. in the Digital Engineering Center of Communication University, China. He is presently Lecturer in the department of physics at Jining Normal University, China. His main research interests include Compressive Sensing, High-performance DSP Algorithms and VLSI Architectures.

Copyright ⓒ 2015 SERSC

Copyright ⓒ 2015 SERSC