Optimization of time and area of a Cordic processor for Low power application

Savitha.S [1], Yogitha.S [2]
[1,2]Assistant Professor, Deparment of Electronics and Communication Engineering,
[1,2]Sri Shakthi Institute of Engineering and Technology, India.

Abstract— This paper presents an ideology of area and time efficient CORDIC algorithm that completely eliminates the scale-facto by the suitable selection of the order of approximation of Taylor series concept. The proposed CORDIC circuit attains the desired range of convergence and meets the accuracy requirement. An algorithm called Generalized micro-rotation selection technique has been implemented to redefine the elementary angles for reducing the number of CORDIC iterations. The proposed CORDIC processor provides the flexibility to change the number of iterations depending on the accuracy, area and latency requirements. The proposed Cordic processor was simulated in Modelsim6.1 and implemented on Xilinx Spartan XC2S500E device and Chipscope Pro in order to evaluate various parameters.

Keywords—coordinate rotation digital computer (CORDIC), cosine/sine, field-programmable gate array (FPGA).

I. INTRODUCTION

CORDIC (for **CO**ordinate **R**otation **DI**gital **C**omputer), also known as **Volder's algorithm**, is a simple and efficient algorithm to calculate hyperbolic and trigonometric functions, typically converging with one digit (or bit) per iteration. CORDIC is therefore also an example of digit-by-digit algorithms. CORDIC and closely related methods known as **pseudo-multiplication** and **pseudo-division** or **factor combining** are commonly used when no hardware multiplier is available (e.g. in simple microcontrollers and FPGAs), as the only operations it requires are addition, subtraction, bit shift and table lookup.The coordinate rotation digital computer (CORDIC) has established its popularity in several important areas of application, like generation of sine and cosine functions, calculation of discrete sinusoidal transforms like fast Fourier transform (FFT), discrete sine/cosine transforms (DST/DCT), householder transform (HT), etc.. Many variations have been suggested for efficient implementation of CORDIC with less number of iterations over the conventional CORDIC algorithm. The number of CORDIC iterations is optimized by greedy search at the cost of additional area and time for the implementation of variable scale-factor. Efficient scale-factor compensation techniques are proposed, which adversely affect the latency/throughput of computation. Two area-time efficient CORDIC architectures have been suggested, which involve constant scale-factor multiplication for adequate range of convergence (RoC). The virtually scale-free CORDIC also requires multiplication by constant scale-factor and relatively more area to achieve respectable RoC. The enhanced scale-free CORDIC in combines few conventional CORDIC iterations with scaling-free CORDIC iterations for an efficient pipelined CORDIC implementation with improved RoC. However, if used for recursive CORDIC architecture, combining two different types of CORDIC iterations degrades performance. In this paper, we propose a novel scaling-free CORDIC algorithm for area-time efficient implementation of CORDIC with adequate RoC. The proposed recursive architecture has comparable or less area complexity with other existing scaling-free CORDIC algorithms. Moreover, no scale-factor multiplications are required for extending the RoC to entire coordinate space, as required. propose a novel scaling-free CORDIC algorithm for area-time efficient implementation of CORDIC with adequate RoC. The proposed recursive architecture has comparable or less area complexity with other existing scaling-free CORDIC algorithms. Moreover, no scale-factor multiplications are required for extending the RoC to entire coordinate space, as required. The CORDIC algorithm operates either in, rotation mode or vectoring mode, following linear, circular or hyperbolic coordinate trajectories. In this paper, we focus on rotation mode CORDIC using circular trajectory.

II. CONVENTIONAL CORDIC ALGORITHM

In conventional CORDIC to obtain the rotated vector, the angle of rotation "θ" is decomposed into a sequence of fixed predefined elementary rotations with variable direction. The conventional rotation mode CORDIC estimates the (i+1)th intermediate rotated vector from the ith vector using circular trajectory as y_{i+1}

$$\begin{bmatrix} xi+1 \\ yi+1 \end{bmatrix} = K \begin{bmatrix} 1 & \mu itan\alpha i \\ \mu itan\alpha i & 1 \end{bmatrix} \begin{bmatrix} xi+1 \\ yi+1 \end{bmatrix} \qquad (1)$$

Where $K_i = \cos\alpha_i$
$\alpha_i = \tan^{-1}(2^{-i})$

The sine sequence $\mu_i \in \{1,-1\}$ is selected so that

$$\Theta = \sum_{i=0}^{b} \mu i * \alpha i \qquad (2)$$

Note that the range of convergence of this algorithm is limited to [-99.99°,99.99°], which can be extended to entire coordinate space using the properties of sine and cosine functions, using an extra iteration for full-range rotation. Scaling-free CORDIC was the first attempt to completely dispose of the scale-factor. However, the approximation imposes a restriction on the basic-shift1 i = [(b-2.585)/3]. For 16-bit data, the basic-shift=4 results in extremely low range of convergence. However, modified virtually adaptive scaling-free algorithm, extends the range of convergence over the entire coordinate space and introduces an adaptive scale-factor. The minimum possible permissible shifts in the CORDIC iteration have been termed as basic shift, which is equal to the number of right shifts in the first CORDIC iteration.

A. Rotation mode

CORDIC can be used to calculate a number of different functions. This explanation shows how to use CORDIC in rotation mode to calculate the sine and cosine of an angle, and assumes the desired angle is given in radians and represented in a fixed-point format. To determine the sine or cosine for an angle, the y or x coordinate of a point on the unit circle corresponding to the desired angle must be found. Using CORDIC, one would start with the vector V_0

$$\mathbf{V_0}= \begin{bmatrix}1\\0\end{bmatrix} \qquad (3)$$

In the first iteration, this vector is rotated 45 degree counterclockwise to get the vector V_i, Successive iterations rotate the vector in one or the other direction by size-decreasing steps, until the desired angle has been achieved. Step "i" size is arctan(2^{-i}) for i=0,1,2,3,.... [6]

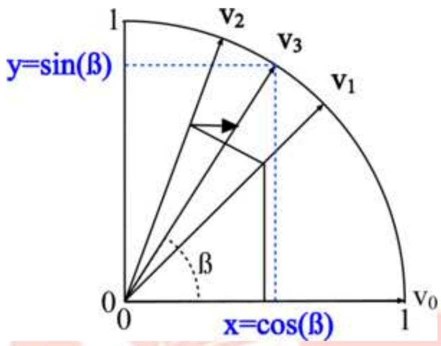

Fig. 1 An illustration of CORDIC algorithm in process

More formally, every iteration calculates a rotation, which is performed by multiplying the vector V_{i-1} with the rotation matrix R_i:

$$V_i=R_i V_{i-1} \qquad (4)$$

B. Vectoring mode

The rotation-mode algorithm described above can rotate any vector (not only a unit vector aligned along the *x* axis) by an angle between –90° and +90°. Decisions on the direction of the rotation depend on being positive or negative.

The vectoring-mode of operation requires a slight modification of the algorithm. It starts with a vector the *x* coordinate of which is positive and the *y* coordinate is arbitrary. Successive rotations have the goal of rotating the vector to the *x* axis (and therefore reducing the *y* coordinate to zero). At each step, the value of *y* determines the direction of the rotation. The final value contains the total angle of rotation. The final value of *x* will be the magnitude of the original vector scaled by *K*. So, an obvious use of the vectoring mode is the transformation from rectangular to polar coordinates.

III.PROPOSED ALGORITHM FOR SCALING FREE CORDIC

The proposed design is based on the following key ideas: 1) we use Taylor series expansion of sine and cosine functions to avoid scaling operation and 2) suggest a generalized sequence of micro-rotation to have adequate range of convergence (RoC) based on the chosen order of approximation of the Taylor series.

A.Representation of Micro-Rotations Using Taylor Series Approximation

Here, we study the impact of orders of approximation of Taylor series of sine and cosine functions on the micro-rotations to be used in CORDIC coordinate calculation.

We have used above equation for coordinate calculation for evaluating the best possible combination of approximation, which satisfies the accuracy and RoC requirements, with minimum possible hardware. Since the errors resulting are of very small order, we prefer to use them for coordinate calculation with minimum complexity.

B.Micro-Rotations Using Taylor Series Approximation and Factorial Approximation:

Although, we find that we can use Taylor series expansion with third order of approximation, with desired accuracy and RoC requirement cannot be used in the CORDIC shift-add iterations. The maximum percentage of error in sine and cosine values for both third order of approximation and factorial approximation is 0.0004% and 0.0168%, respectively, within the permissible CORDIC elementary angles range.

TABLE I
BIT REPRESENTATION OF ELEMENTARY ANGLES AND CORRESPONDING SHIFTS

Shift (s_i)	Elementary angle(α_i)	
	Decimal	16 bit hexadecimal
2	0.25	4000H
3	0.125	2000H
4	0.0625	1000H
5	0.03125	0800H

C.Determination of the Basic-Shift for a Given Order of Approximation of Taylor Series Expansion

One can find that: 1) the order of approximation of Taylor series expansion of sine and cosine functions determines the basic-shift to be used for CORDIC iterations, and 2) the basic-shift of CORDIC micro operation determines the range of convergence. The expressions for the basic-shifts, the first elementary angle of rotation(α_1)and RoC for different orders of approximations for different word-length of implementations are as follows:

Basic shift, $s=(b-\log_2(n+1)!)/(n+1)$ (5)

Where b is word length, and

$RoC=n_i x \alpha_i$ (6)

Where n_i is the number of micro rotation

TABLE II
COMPARISON OF APPROXIMATION ORDERS VERSUS ROC FOR VARIOUS BIT WIDTHS BASED ON (6)

Order of approximation	Basic shift		First elementary angle(Radians)		ROC for π=4 (Radians)	
	16 bit	32 bit	16 bit	32 bit	16 bit	32 bit
3	2	6	0.25	0.0156	1	0.06
4	1	5	0.5	0.0312	2	0.12
5	1	3	0.5	0.125	2	0.5

The values in Table II are derived from (5). We find with increase in the order of approximation, the basic-shift decreases, the first elementary angle of rotation increases and RoC is expanded. Very often inclusion of higher order terms does not have any impact on the accuracy for smaller word-lengths. The basic-shift for third order of approximation using (5), for 16-bit word-length is [2.854].The RoC(with basic-shift for 16-bit) is large enough to be mapped to the entire coordinate-space.

IV.GENERALISED MICRO-ROTATION SELECTION

In the proposed generalized micro-rotation sequence, we perform multiple iterations of basic-shift, followed by non-repetitive unidirectional iterations of the micro-rotations corresponding to other shift indices, to minimize the number of iterations and achieve adequate range of convergence.

A. Organization of Micro-Rotation Sequence

In the proposed scheme, we represent the rotation angle "θ" as

$$\theta = n_1.\ \alpha_s + \sum_{i=0}^{n2} \alpha_{si}$$
$$n = n_1 + n_2 \qquad (7)$$

where, α_s is the elementary angle corresponding to the basic-shift, α_{si} are elementary angles for other shifts, n1 and n2 are non-negative integers and a represents the total number of iterations. If we do not use any micro-rotation of angle and then n1 is zero, and n_2=n. On the other hand, if the desired angle of rotation "θ" is a multiple of as then n_2 is zero and n_1=n.

B. Defining the Elementary Angles

The elementary angles α_s and α_{si} given by

$$\alpha_{si} = 2^{-\alpha} \quad \text{and} \quad \alpha_{si} = 2^{-\alpha i} \qquad (8)$$

where, s is the basic-shift and s_i>s is the shift for ith iteration. For basic-shift =2, we can find α_s=7π/88 and for basic-shift =3, we can find α_s=7π/176. In Table II, we list the decimal and (0, 16) fixed point binary representation of the elementary angles corresponding to different shifts.

C. Generalized Micro-Rotation Sequence Identification

We identify the micro-rotations depending on the bit representation of the desired rotation angle in radix-2 system using most-significant-1 detector. For this we restrict the maximum rotation angle to π/4 radians as the entire coordinate space [0,2π], can be mapped to the [0.π/4] using octant symmetry of sine and cosine functions. Pseudo code for generating the micro-rotation sequence

```
Input: angle to be rotated ($\theta_i$)
Begin
M=Most-Significant-1 Location of $\theta_i$
if(M==15)then
α=0.25 radians
shift, $s_i$=2 and $\theta_{i+1}$= $\theta_i$- α
Else
  Shift, $s_i$=16-M
  $\theta_{i+1}$= $\theta_i$ with $\theta_i$[M]='0'
End
```

If the most-significant-1 location (M) of the rotation angle "θ" is smaller than the basic-shift "s_i", elementary angle of the basic-shift would be used for the CORDIC iteration. For a fixed word-length of N-bit, the shift(s_i) for the elementary angle is given by

$$s_i = N - M \tag{9}$$

D. Number of Iterations to Have Desired RoC

In this section, we decide on a suitable value of "α" for realizing rotations by angles in the range [0,π/4].The basic-shift for 16-bit word-length is "2.854". But the basic-shift should be an integer, so we design the iterations for both "2" and "3".With basic-shift =2[α_s=7pi/88], no more than three iterations of α_s are required; therefore, the maximum value of a_i is 3.The iterations corresponding to α_2 depend on the accuracy requirements. With various values of α_2 the accuracy varies and is different for "x" and "y" coordinates.

V. PROPOSED CORDIC ARCHITECTURE

The block diagram for the proposed CORDIC architecture is shown in Fig. 2. It makes use of the same stage for all the iterations for the coordinate calculations, as well as for the generation of shift values. The structure of each stage (shown in Fig. 2) consists of three computing blocks namely: the 1) shift-value estimation; 2) coordinate calculation; and 3) micro-rotation sequence generator. The combinatorial circuit for the evaluation of desired shift values is shown in Fig. 3; the coordinate calculation is implemented according to (6); the combinatorial circuit for generating the micro-rotation sequence is shown in Fig. 2.

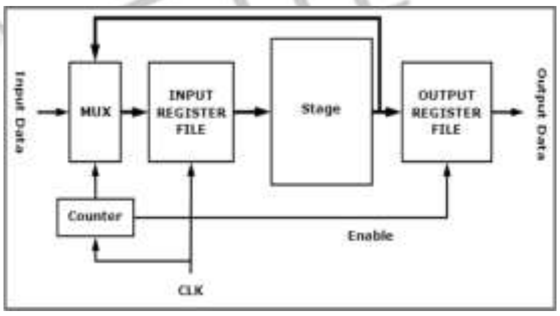

Fig.2 Proposed CORDIC processor

The number of iterations required in a CORDIC processor decides the rollover count of the counter. The rollover count is seven for basic shift=2 and ten for basic-shift =3. The expiry of the counter signals the completion of a CORDIC operation; depending on this signal, the multiplexer either loads a new data-set (rotation angle, initial value of "x" and "y") to start a fresh CORDIC operation, or recycles the output of the stage to begin a new iteration for the current CORDIC operation. The input and output register files act as latches for synchronization.

VI. FPGA IMPLEMENTATION

The proposed architecture is coded in VHDL and synthesized using Xilinx ISE9.2i to be implemented in Xilinx Spartan 3E (XC2S500EPQ208- 6) device. Slice-delay-product of the proposed architecture is compared with the existing CORDIC designs in Table IV; where, all designs are synthesized on Xilinx Spartan 3E XC2S500E device to maintain uniformity.

A .ChipScope Pro

The use of FPGAs allows for an alternative way to perform testing and debug activities which excludes the use of a logic analyzer. The method relies on incorporating specific circuit *cores* into the design, which can monitor the rest of the system, and send information to a host computer. The ChipScope Pro collection of IP cores accomplishes this task; by instantiating the Integrated Controller (ICON) and one or more Integrated Logic Analyzers (ILAs) into a design, any signals in the design can be sampled; the data thus captured can then be sent to a host computer for analysis.

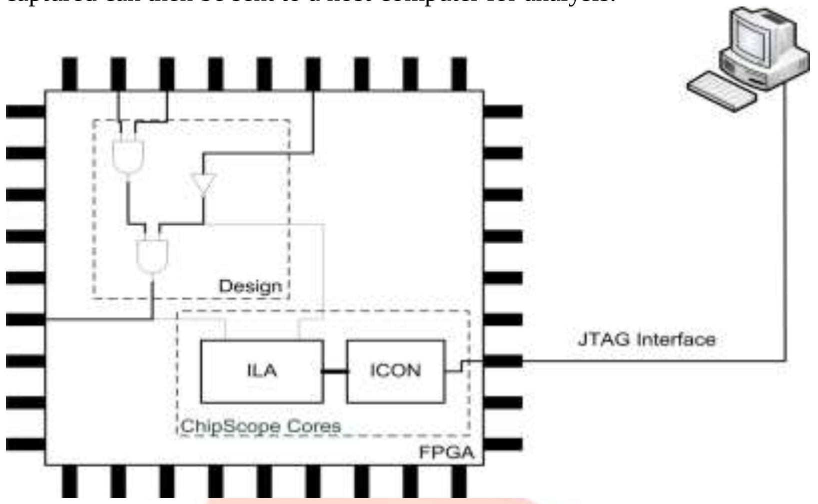

Fig.3 Testing and Debug of Digital Designs using ChipScope

VII. CONCLUSION

The proposed algorithm provides a scale-free solution for realizing vector-rotations using CORDIC. The order of Taylor series approximation is decided appropriately by the proposed algorithm, not only to meet the accuracy requirement but also to attain adequate range of convergence. The generalized micro-rotation selection technique is suggested to reduce the number of iterations for low latency implementation. Moreover, a high speed most-significant-1 detection scheme obviates the complex search algorithms for identifying the micro- rotations. The proposed CORDIC Processor has lower slice delay with penalty of increased slice
consumption on Xilinx Spartan 3E device.

REFERENCES

[1] J. E. Volder, “The CORDIC trigonometric computing technique,” *IRE Trans. Electron. Comput.*, vol. EC-8, pp. 330–334, Sep. 1959.
[2] K. Maharatna, A. S. Dhar, and S. Banerjee, “A VLSI array architecture for realization of DFT, DHT, DCT and DST,” *Signal Process.*, vol. 81,pp. 1813–1822, 2001.
[3] 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.
[4] C. S. Wu and A. Y. Wu, “Modified vector rotational CORDIC (MVRCORDIC) algorithm and architecture,” *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 48, no. 6, pp. 548–561, Jun. 2001.
[5] C.-S.Wu, A.-Y.Wu, and C.-H. Lin, “A high-performance/low-latency vector rotational CORDIC architecture based on extended elementary angle set and trellis-based searching schemes,” *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 50, no. 9, pp. 589–601, Sep. 2003.
[6] https://en.wikipedia.org/wiki/CORDIC
[7] Y. H. Hu and S. Naganathan, “An angle recoding method for CORDIC algorithm implementation,” *IEEE Trans. Comput.*, vol. 42, no. 1, pp. 99–102, Jan. 1993.
[8] M. G. B. Sumanasena, “A scale factor correction scheme for the CORDIC algorithm,” *IEEE Trans. Comput.*, vol. 57, no. 8, pp. 1148–1152, Aug. 2008.
[9] J. Villalba, T. Lang, and E. L. Zapata, “Parallel compensation of scale factor for the CORDIC algorithm,” *J. VLSI Signal Process. Syst.*, vol. 19, no. 3, pp. 227–241, Aug. 1998.
[10] L. Vachhani, K. Sridharan, and P. K. Meher, “Efficient CORDIC algorithms and architectures for low area and high throughput implementation,” *IEEE Trans. Circuit Syst. II, Exp. Briefs*, vol. 56, no. 1, pp. 61–65, Jan. 2009.
[11] 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.