International Journal of Scientific Research & Engineering Trends
Volume 7, Issue 4, July-Aug-2021, ISSN (Online): 2395-566X

Advanced Approach for CORDIC Architecture Using Carry Select Adder

Harsit Tiwari, Tarun Verma
Department of Electronics & Communication Engineering
Lakshmi Narain College Of Technology
Bhopal, (M.P.)
harshittiwari18@gmail.com

Abstract- We have developed an efficient CORDIC algorithm in this research, which uses many rotations to reduce the CORDIC rotation angle. Instead of employing a regular adder, this innovative CORDIC method employs an area efficient carry select adder (CSLA). In many data processing techniques, this adder can do quick arithmetic operations. Finally, several parameters such as area, power, and latency are evaluated, and the proposed approach reduces these when compared to the existing method.

Keywords- CORDIC, Carry Select Adder, Advanced approach.

I. INTRODUCTION

Digital Signal Processor (DSP) is one of the most essential processors in today's technology for speeding up operations and obtaining error-free results in hardware implementation. Multipliers and adders are the most important hardware units for filtering, convolution, compression, and other operations.

An efficient CORDIC algorithm was introduced to speed up the functioning of the DSP units. Trigonometric functions can be computed very precisely with minimal hardware and latency using this approach. For completing the above-mentioned tasks, the CORDIC digital computer has unique digital units such as shifters and adders [1].

Much architecture has been proposed throughout the years to take advantage of the CORDIC Algorithm. The number of rotation-extensions executed for each angle set became constant, and as a result, the scale factor employed in the method became constant, regardless of the operand. The necessity to compute the scaling factor [2] has been eliminated as a result of this effort. To achieve the best results in the implementation, an updated Scaling-Free CORDIC method was introduced for DSP applications.

The original CORDIC algorithm and improved scaling free CORDIC algorithms were developed on a microcontroller in this study, and a comparison of both algorithms was made, with the improved scaling free CORDIC proving to be faster and more accurate in computations [3]. An optimised CORDIC algorithm is one that overcomes the limitations of traditional CORDIC in terms of power, area, speed, and precision. The above-mentioned parameters are used to optimise the traditional CORDIC algorithm, which is then implemented on the FPGA [4].

For FFT computation, a new memory-saving CORDIC technique has been introduced. The size of the rotor will be minimised and the directions of the micro rotations of the rotors will be calculated for FFT computation of the input sequence [5]. FFT processor design was implemented effectively to reduce the amount of addition and multiplication operations using a mix of radix FFT and CORDIC algorithm. The twiddle factor in the FFT processor's butterfly computation was also reduced in complexity [6].

For simple and small area requirements, a constant coefficient quaternion multiplier is employed with the CORDIC algorithm; however more sparse iterations are necessary to achieve high accuracy results [7]. The number of functional units will be reduced using a Folded Pipelined Architecture for FFT with CORDIC Algorithm. Several butterfly units are present in the same column during the folding operation, which can be merged into a single butterfly unit.

II. RELATED WORK

Manupotisreenivasulu et al. [1] Up to the third and fourth stages, the suggested architecture for CORDIC adders, subtractors, and shifters are all replaced with multiplexers. All approaches are performed on Xilinx Spartan 3E (XC3S250E) and Xilinx Virtex6 FPGAs using an 8-bit and 16-bit Coordinate Rotation Digital Computer to achieve sine and cosine functions (XC6VLX240). CORDIC achieves a high operating frequency and requires less area for hardware implementation as compared to unrolled CORDIC MUXes.

A. Elnabawy et al. [2] The CORDIC-based neuron hardware implementation consumes 0.26 to 0.4 mW of power, but the PWL-based neuron and the original Izhikevich neuron hardware implementations consume 0.3

and 1.06 mW of power, respectively. The tradeoff between mistakes, power, and area is depicted in a Figure of Merit (FoM). When compared to the PWL-based neuron hardware implementation, the CORDIC-based model is determined to be preferable as an approximation method in terms of error, power, and area.

Dynamic CORDIC was proposed by P. Chou et al [3]. In 28 nm CMOS technology, the static and dynamic CORDIC are evaluated based on pre-layout simulation findings. The suggested dynamic CORDIC may deliver a 41 percent and 30 percent energy reduction compared to the static CORDIC at the average process corner for an IoT application with a 0.02 percent and 0.2 percent duty cycle.

P. A. Kumar et al. [4] the area, delay, and power consumption of serial and pipelined CORDIC architectures installed on a Cyclone IV E Device are compared. The area of a serial CORDIC architecture is small, whereas the latency of a pipeline CORDIC architecture is short. Graphic processors, digital synchronizers, real-time image processing, scientific calculators, and other devices use it.

Y. Xue et al. [5] And the Cyclone IV FPGA has already been used to model and implement this new way. The results demonstrate that the number of iterations has been significantly reduced, as has the amount of hardware resource consumption.

M. Heidarpur et al. [6] the system's correctness, efficacy, and speed are verified by comparing the implementation outcomes to the original model and state-of-the-art. These comparisons show that the suggested neuromorphic system outperforms the competition in terms of performance and accuracy while being simple to construct and scale.

P. I. Puzyrev et al. [7] The outcome is a relationship between SDFR shift and CORDIC parameters, as well as an approximate expression that aids in obtaining assured SFDR for specified CORDIC parameters across the whole frequency range.

A. Sahoo et al. [8] The basic CORDIC algorithm is reviewed, and the folded word-serial and unfolded pipelined CORDIC architectures are implemented. The Xilinx Artix-7 FPGA is used to implement these designs. For designs, hardware utilisation and operation speed are tabulated. The accuracy of a 16 bit precision CORDIC design is determined by comparing it to the results of a Matlab simulation.

III. PROPOSED METHODOLOGY

The CORDIC (COordinate Rotation Digital Computing) technique is a space and time efficient method for computing the Sine and Cosine of a given angle. It can also be used to find the log, exponent, and square root of a number. Sine and cosine generation, vector magnitude, polar-cartesian conversions, and vector rotation are all common applications.

The CORDIC core is a 16-bit fixed-point CORDIC with a parameterized Verilog RTL code. This system takes a 17-bit angle in degrees (signed magnitude representation) as input and outputs 17-bit sine and cosine values. The time economy of this technique is due to the shift operation replacing the multiplication/division step. As a result, the only expensive operation remaining is addition. As a result, adders are at the core of CORDIC design.CLA (Carry Look-ahead Adder), RCA (Ripple Carry Adder), and a combination of CLA and RCA are used in the CORDIC core.

Small groups of CLAs are linked by rippling carry-out and carry-ins in a CLA-RCA combination. There are trade-offs between area and efficiency in all three systems. CLA-based CORDIC has the smallest area but is the slowest of the three CORDIC architectures, but segmented CLA-based CORDIC has the highest performance with area panelty.

- Fixed-point arithmetic using 16 bits.
- The input angle ranges from -45 to +45 degrees.
- Signed magnitude representation of the input angle value (1 sign bit, 8 integer bits, 8 fractional bits).
- 2's complement representation of output values
- Reset in synchrony.
- Verilog RTL code with parameterized sample synthesis script (For Synopsys DC).

We concentrated on developing the CORDIC basic theory and the Verilog implementation of parallel CORDIC blocks. We have included a full instruction in a.pdf file, which can be found below, due to the web page's lack of flexibility. This article shows how to implement two major CORDIC blocks in Verilog. There's also an example of using CORDIC iterations to calculate the magnitude and phase of a complex number. The download links for such codes are provided below. The CORDIC algorithm can deal with three different coordinate systems.

Circular \sLinear \sHyperbolic:
CORDIC was originally designed for circular co-ordinates, but it was later expanded to include linear and hyperbolic co-ordinates. There are two modes of operation for the CORDIC algorithm:

Rotation \sVectoring: Both options are feasible in each coordinate system. The CORDIC block evaluates several functions depending on the mode of operation.CORDIC can be used to calculate the following popular functions.

- Coordinates can be rotated at any angle.

- Conversion of coordinates to polar coordinates
- A complex number's magnitude and the angle between them
- An angle's sine and cosine
- An angle's hyperbolic sine and cosine
- operation of the division
- Operation of multiplication
- operation on the square root

CORDIC is utilised to optimise the hardware implementation in a number of well-known applications. The following are some of the most important application areas.The most direct application of CORDIC is in the creation of calculators that can perform several calculations with the same hardware.

CORDIC can be used to evaluate arithmetic functions in a variety of complex systems or algorithms. CORDIC is commonly used to calculate the reciprocal of a number or to evaluate division.The property of rotation of co-ordinates is utilised to create transform domains like FFT, Wavelet, Curvelet, and DCT, among others.

In vectoring mode, the CORDIC can easily find the magnitude or absolute of two values in circular co-ordinates. This characteristic is employed in a variety of applications, such as the factorization QR decomposition of matrices.CORDIC uses the same hardware to find the sine and cosine of an angle. Many DSP applications make use of this characteristic.CORDIC in parallel as seen, parallel CORDIC has a roughly identical design as pipelined CORDIC. The structure for iterative CORDIC is joined many times simultaneously in this approach. We can decompose each angle into sub-angles and propose the computation of these angles in parallel for any angle.

$$\theta = d_0\alpha_0 + d_1\alpha_1 + \ldots\ldots\ldots\ldots\ldots\ldots d_{n-1}\alpha_{n-1}$$

$$\alpha_i = \tan^{-1} 2^{-i}$$

The parallel design implements the shift add/sub function in parallel using clusters of shift add-sub levels with one clock cycle access time. The parallel architectures have even more inputs and outputs, as well as a higher throughput, but they are more capable than the iterative architectures.

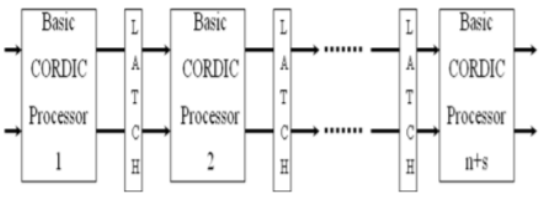

Fig 1. parallel architectures have even more inputs and outputs.

IV. RESULTS ANALYSIS

The proposed technique is simulated in MATLAB, and the full project was done on an I7 processor with 8 GB of RAM. Similarly, utilising the 180nm technology and RTL compiler, the CORDIC algorithm is used to compute area, power, and delay, with only shifters and adders and no multiplier used in the implementation. in terms of area, power, and latency, compares the performance of existing and new approaches.

Figure 1. compares the existing and suggested methods based on factors including area, power, and delay. All of the parameters listed above have been minimised, as evidenced by this study. Presents a graph comparing the current and planned system performance.

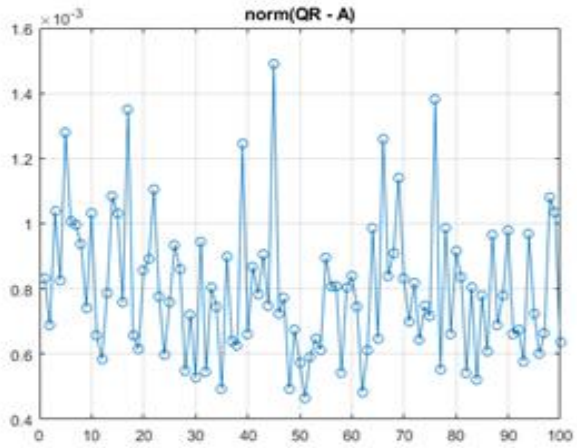

Fig 2. Compares proposed approach.

V. CONCLUSION

The CORDIC's design is thoroughly examined using various stages of micro rotation, which are employed to reduce the angle. The CORDIC architecture's adders are replaced with low-power, small-area CSLA adders.When compared to the present architecture, the performance metrics were thoroughly analysed, and we were successful in minimising the area, power, and delay.

REFERENCE

[1] Manupotisreenivasulu and T. Meenpal, "Efficient MUX Based CORDIC on FPGA for Signal Processing Application," 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), 2019, pp. 1-6, doi: 10.1109/ICECCT.2019.8869306.

[2] A. Elnabawy, H. Abdelmohsen, M. Moustafa, M. Elbediwy, A. Helmy and H. Mostafa, "A Low Power CORDIC-Based Hardware Implementation of

Izhikevich Neuron Model," 2018 16th IEEE International New Circuits and Systems Conference (NEWCAS), 2018, pp. 130-133, doi: 10.1109/NEWCAS.2018.8585485.

[3] P. Chou, Y. Fang, B. Chen, C. Liu, T. Lin and J. Wang, "Near-Threshold CORDIC Design with Dynamic Circuitry for Long-Standby IoT Applications," 2018 31st IEEE International System-on-Chip Conference (SOCC), 2018, pp. 250-253, doi: 10.1109/SOCC.2018.8618488.

[4] P. A. Kumar, "FPGA Implementation of the Trigonometric Functions Using the CORDIC Algorithm," 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS), 2019, pp. 894-900, doi: 10.1109/ICACCS.2019.8728315.

[5] Y. Xue and Z. Ma, "Design and Implementation of an Efficient Modified CORDIC Algorithm," 2019 IEEE 4th International Conference on Signal and Image Processing (ICSIP), 2019, pp. 480-484, doi: 10.1109/SIPROCESS.2019.8868732.

[6] M. Heidarpur, A. Ahmadi, M. Ahmadi and M. R. Azghadi, "CORDIC-SNN: On-FPGA STDP Learning with Izhikevich Neurons," 2020 IEEE International Symposium on Circuits and Systems (ISCAS), 2020, pp. 1-1, doi: 10.1109/ISCAS45731.2020.9180463.

[7] P. I. Puzyrev, K. V. Semenov and S. A. Zavyalov, "Spurious-Free Dynamic Range of CORDIC Based Digital Quadrature Demodulator," 2018 19th International Conference of Young Specialists on Micro/Nanotechnologies and Electron Devices (EDM), 2018, pp. 167-171, doi: 10.1109/EDM.2018.8434980.

[8] A. Sahoo and M. Panigrahy, "Hardware Implementation of CORDIC Algorithm," 2018 International Conference on Applied Electromagnetics, Signal Processing and Communication (AESPC), 2018, pp. 1-4, doi: 10.1109/AESPC44649.2018.9033343.