Intl J Engg Sci Adv Research 2015 Sep;1(3):223-227

Performance of the CORDIC Processors

Shalini Rai
J.K. Institute of Applied Physics &Technology
University of Allahabad, Allahabad (UP)
shaliniece_04@yahoo.com

Rajeev Srivastava
J.K. Institute of Applied Physics &Technology
University of Allahabad, Allahabad (UP)
rajeev_jk@rediffmail.com

***Abstract*— In this paper simulation and testing of bit parallel iterative, bit parallel unrolled and bit serial iterative CORDIC Processors on SPARTAN 3 FPGA using VHDL project navigator version 6.1 has been reported. Different trigonometric functions have been evaluated on each of the above architectures and a comparison of their speed of computation and error performance has been presented.**

***Keywords*— CORDIC; Trigonometric functions; Error Analysis**

I. INTRODUCTION

Speedy calculation of trigonometric functions by electronic hardware has been an active research field due to its need in truly demanding tasks. For a long time microprocessor- based systems have been used to perform these tasks, however the advent of reconfigurable logic VLSIs e.g. FPGAs has enabled higher speeds of dedicated hardware solutions at costs that are competitive with traditional software approach.

Several hardware efficient algorithms and correspondingly suitable architectures have been explored for the purpose. Among the existing hardware algorithms CORDIC (Coordinate Digital computer) [1, 2] is widely adopted because of its simplicity and speed efficiency. The implementations of CORDIC algorithms on FPGAs are known as CORDIC Processors. Various algorithms for FPGAs and Application Specific Integrated Circuits (ASICs) implementations of CORDIC have been reported [3-10]. The CORDIC algorithm has been applied to diverse applications such as real time navigation problems, in the 8087 math coprocessor [6], the HP-35 calculator, radar signal processor [12] and robotics.

It has also been proposed for computing in digital signal processing [3], Discrete Fourier [13], Discrete Hartley [14] and Chirp-Z transforms [15], filtering [13], Singular Value Decomposition [16], and solving linear system [17].

In this paper the error and speed performance of three CORDIC architectures, viz, bit parallel iterative, bit parallel unrolled and bit serial iterative architectures have been obtained after simulating them on SPARTAN 3 FPGA using VHDL project navigator version 6.1.

The rest of paper is organized in the following manner .Section II presents in brief the theory of CORDIC algorithm. Section III presents the CORDIC architectures, viz, bit parallel iterative, bit parallel unrolled and bit serial iterative. I/O formats are described in section IV. Error performance of the above architectures incurred in the evaluation of different functions are reported in section V. Section VI conclude the paper.

II. THEORY OF CORDIC ALGORITHM

A. The CORDIC Algorithm

CORDIC (Coordinate Rotation Digital Computer) is an iterative algorithm for the calculation of trigonometric functions [18,19]. The algorithm is based on the general vector rotation transform: the following transform equations rotate a vector in a Cartesian plane by an angle α

$$x' = x\cos\alpha - y\sin\alpha$$

$$y' = y\cos\alpha + x\sin\alpha$$

These equations can be rearranged as

$$x' = \cos\alpha\,[x - y\tan\alpha]$$

$$y' = \cos\alpha\,[y + x\tan\alpha]$$

Now, let us consider small rotations δi

Such that $\tan\delta i = \pm 2^{-i}$

And $ki = \cos\delta i = \cos(\tan^{-1} \pm 2^{-i})$

i.e $ki = 1/(1+2^{-2i})^{0.5}$

Further rotation by an arbitrary angle α can be obtained by performing a series of successively smaller elementary rotations by angle δi. These iterative rotations can be expressed as

$$x_{i+1} = k_i [x_i - y_i . d_i . 2^{-i}]$$

$$y_{i+1} = k_i [y_i + x_i . d_i . 2^{-i}]$$

$$k_i = \cos(\tan^{-1} 2^{-i}) = 1/(1+2^{-2i})^{0.5}$$

d_i = +1 for anticlockwise rotation

-1 for clockwise rotation

The above iterative operations can be easily implemented digitally by add and shift operations. The constant k i's cab be treated as system processing gain. The product of ki's represents the so called k factor [1].

$$n-1$$

$$K = \prod_{i=0} ki = 0.6073$$

This k can be calculated in advance and may be implemented by initializing the iterative rotations with a vector of length $\lceil k \rceil$.

Further, the angle of composite rotation is uniquely defined by the sequence of the direction of the elementary rotations. This sequence can be represented by a decision vector .Thus the angle accumulator adds a third difference equation

$z_{i+1} = z_i - d_i (\tan^{-1} 2^{-i})$ to the CORDIC algorithm

B. *Operating modes of CORDIC Algorithm*

The CORDIC algorithm works in one of two modes. In The first mode, called rotation mode, in which rotates the input vector is rotated by a specified angle. The second mode, called vectoring mode, rotates the input vector to the x axis while recording the angle required making that rotation.

In the rotation mode

$$d_i = -1 \text{ if } z_i < 0$$
$$= +1 \text{ otherwise}$$

This provides the following result

$$x_n = A_n [x_0 \cos z_0 - y_0 \sin z_0]$$
$$y_n = A_n [y_0 \cos z_0 + x_0 \sin z_0]$$
$$z_n = 0$$
$$A_n = \prod_{n} (1+2^{-2.i})$$

In the vectoring mode

$d_i = +1$ if $y_i < 0$,

$= -1$ otherwise.

This provides the following result

$$x_n = A_n (x_0^2 + y_0^2)^{0.5}$$
$$y_n = 0$$
$$z_n = z_0 + \tan^{-1} (y_0/x_0)$$
$$A_n = \prod_{n} (1+2^{-2.i})$$

It may be noted that a judicious choice of initial values and mode of operation is needed for direct computation different trigonometric functions and transformation between polar and Cartesian Coordinates. For example, the rotation mode is suitable for calculating sine and cosine functions and polar to rectangular co-ordinates. Whereas for arctangent, vector magnitude calculation and Cartesian to polar transformations the vectoring mode of operation suitable.

III. CORDIC ARCHITECTURES AND PROCESSORS

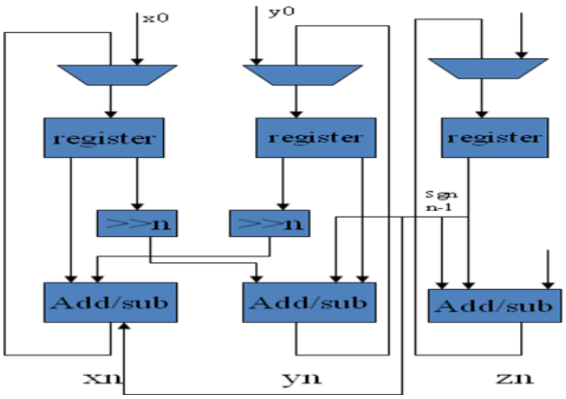
When implementing the CORDIC algorithm on FPGA one can choose between various design architectures and must balance circuit complexity with respect to performance. The implementation of the following three architectures and their performance is reported here.

(i)Bit Parallel iterative CORDIC Processor

(ii) Bit Parallel Unrolled CORDIC Processor

(iii) Bit Serial iterative CORDIC Processor

The Bit –Parallel CORDIC uses a single block of hardware circuits for all the n iterations. Therefore a latency of n clock cycles exists between the input and output data. Parallel evaluation of the input data bits is adopted. Thus the Bit Parallel implementation is faster than a serial evaluation of input data bits. Figure 1shows the general architecture of a Bit –parallel CORDIC processor. The x, y, z branches shows in the Figure 1 evaluate the basic equation of CORDIC algorithm. These branches calculate the (i+1)th value of x, y, z in the ith clock cycle. These (i+1)th values are the input to the next iterations.



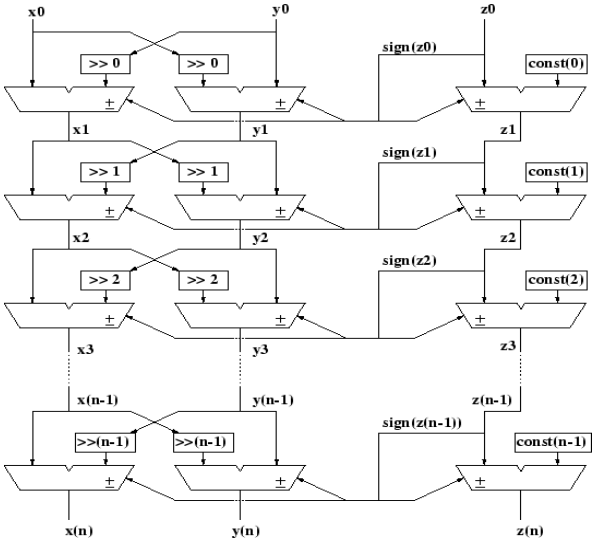
No. of bits =10

Fig. 1. Architecture of Bit Parallel iterative CORDIC Processor

1) Bit Parallel Unrolled CORDIC Processor: The unrolled CORDIC is a pipelined implementation of Bit Parallel CORDIC. Each iterations of the CORDIC algorithm is performed in separate hardware cells that are pipelined. A latency of n clock cycles exist between the input and output values. Figure 2 show the general architecture of unrolled CORDIC. The iterations are separated into different stages of the pipeline. Barrel shifters are not required because each stage shifts left by a constant by hardwiring the signals. The ROM that stores the arc tangent values is replaced by hardwired constant signals. The pipeline stores the output of each stage at the end of every clock cycle.

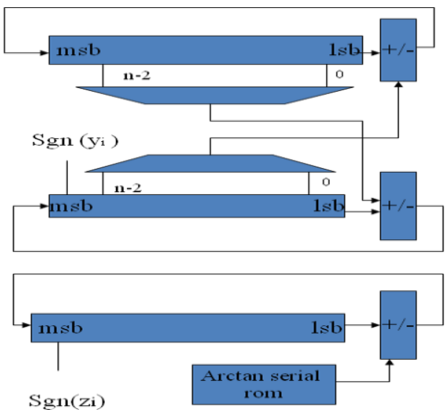
2) Bit Serial Iterative CORDIC: Both, the unrolled and the iterative bit parallel designs, show disadvantages in terms of complexity and path delays going along with large numbers of cross connections between single stages. This complexity is reduced bit serial iterative architecture because the processing is done bit by bit in serial mode. However, in this case the total processing time is increased and Throughput becomes a function of Clock rate/(number of iteration* word width).

A general architecture of bit serial iterative CORDIC Processor is shown in fig 3. In this architecture the bit serial adder subtractor component is implemented as a full adder where the subtraction is performed by adding the 2's complement of the sign bit of the angle accumulator. The shift by i operation is realized by the bit i-1 from its right end in the serial shift registers. A multiplexer is used to change position according to the current iteration. The initial values x0, y0, z0 are fed by parallel in parallel out, parallel in and serial out shift registers.



No. of bits =10

Fig. 2. Architecture of Bit Parallel Unrolled CORDIC Processor



No. of bits =10

Fig. 3. Architecture of Bit Serial Iterative CORDIC Processor

IV. IMPLIMENTATION OF 10 BIT CORDIC PROCESSORS

Implementation of Bit Parallel iterative CORDIC Processor: Basic Blocks which is used for designing of Bit Parallel iterative CORDIC Processor on Xilinx Project navigator version 6.1.

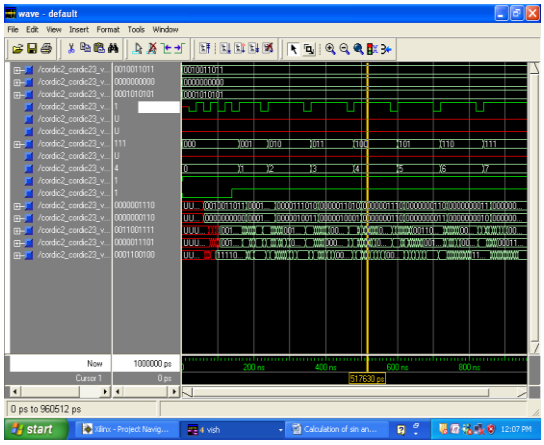
- 2*1 multiplexer, 10 Bit storage device
- Barrel shifter, Parallel full adder
- D flip flop

Testing and Performance

Rotation mode calculation of cos and sin of given angle: circular coordinate

Input values

x0<= 0010011011
y0<= 0000000000
z0<= 0001010101 (angle 30)



xn= 0011100000
yn= 0001111111
zn=11111111110

Error Analysis cos 30= .875 simulated value
cos 30 = .866 actual value
Error = (sim-actual/actual)*100 = (0.875000000-0.866)*100/0.866 = 1.03%
sin 30 = 0.496 simulated value
sin 30 = 0.5 actual value
Error = (0.496-0.5)*100/0.5= -0.8% , Processing Time= 900 ns= 0.9 µs

Implementation of Bit Parallel unrolled CORDIC Processor: Basic Blocks which is used for designing of Bit Parallel iterative CORDIC Processor on Xilinx Project navigator version 6.1.

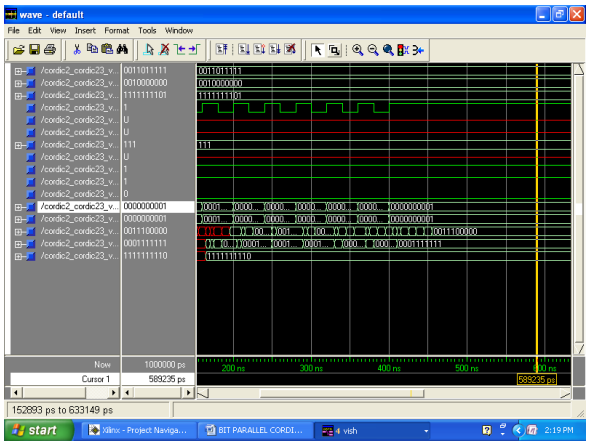
- 2*1 multiplexer
- parallel in parallel out, parallel in and serial out shift registers
- Parallel full adder,
- D flip flop

Testing and Performance

Rotation mode calculation of cos and sin of given angle: circular coordinate

Input values

x0=0010011011
y0=0000000000
z0=0001010101(angle=30)



xn= 0011100000
yn= 0001111111
zn=11111111110

Error Analysis cos 30= .875 simulated value
cos 30 = .866 actual value
Error = (sim-actual/actual)*100 = (0.875000000-0.866)*100/0.866 = 1.03%
sin 30 = 0.496 simulated value
sin 30 = 0.5 actual value
Error = (0.496-0.5)*100/0.5= -0.8%
Processing Time= 2870 ns= 2.8 *10^3 ns = 2.8 µs.

Implementation of Bit Serial CORDIC Processor

Basic Blocks which is used for designing of Bit Parallel iterative CORDIC Processor on Xilinx Project navigator version 6.1.

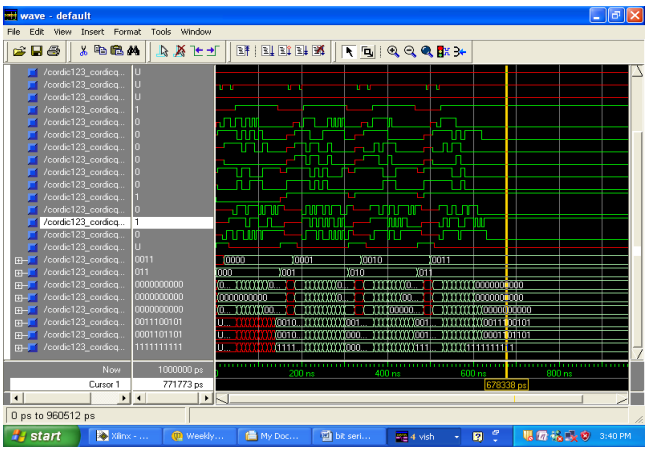
- 2*1 multiplexer
- parallel in parallel out, parallel in and serial out shift registers
- 10*1 multiplexer
- Serial full adder
- D flip flop

Testing and Performance

Rotation mode calculation of cos and sin of given angle: circular coordinate

Input values

x0<= 0010011011
y0<= 0000000000
z0<= 0001010101



xn=0011100101
yn= 0001101101
zn= 1111111111

Error Analysis cos 30= 0011100101 =0.894
simulated value, cos 30 = .866 actual val
Error=(sim-actual/actual)*100=(0.894-0.866)*100/0.866 = 3.2%
sin 30 = 0001101101= .42578125 simulated value
sin 30 = 0.5 actual value
Error = (0.42578125-0.5)*100/0.5= -14.86 % , Processing Time= 620 ns= 0.62 µs

V. RESULTS

Functions	Unrolled		Bit Parallel		Bit Serial	
	E	P	E	P	E	P
Cos	1.03	2.8	1.03	0.9	4	0.94
Sin	-0.8	2.8	-0.8	0.9	10	0.94
$\tan^{-1}$	-1.11	1.2	3.33	0.98	0.62	0.78
Multiply	0	1.8	-0.08	0.99	-4.6	0.96
Division	0	1.8	-4.6	0.99	-0.8	0.25
Cosh	-5	2.17	-7	0.895	-1.16	0.78
Sinh	2.8	2.17	0.44	0.895	8.5	0.78
$\tanh^{-1}$	2.43	2.26	-2.5	0.92	-0.185	0.95

E = ERROR IN PERCENTAGE

P = PROCESSING DELAY IN MICRO SECONDS

VI. CONCLUSIONS

In these Processors the fastest processor is Bit Serial CORDIC Processors. And then Bit Parallel iterative CORDIC Processors, very slow system is Unrolled CORDIC Processor. Percentage error is very small in Unrolled CORDIC Processor.

Bit Parallel iterative CORDIC Processor is slow due to barrel shifter.

REFERENCES

[1] J.E. Volder, "The CORDIC Trigonometric Computing Technique," IRE Trans. Electronic Computers, vol. EC-8, no. 3, 1959,pp. 330–334.

[2] J.S. Walther, "A Unified Algorithm for Elementary Functions," in Proc. Spring. Joint Comput Conference, vol. 38, 1971, pp. 379–385.

[3] Y. Hu, "CORDIC-Based VLSI Architectures for Digital Signal Processing," IEEE Signal Processing Magazine, vol. 9, no. 3,July 1992, pp. 16–35.

[4] R. Andraka, "A Survey of CORDIC Algorithms for FPGAs," in Proceedings of the 1998 ACM/SIGDA Sixth International Symposium on Field Programmable Gate Arrays (FPGA '98), Monterey, CA, Feb. 22–24, 1998, pp. 191–200.

[5] H. Lin and A. Sips, "On-Line CORDIC Algorithms," IEEE Transactions on Computers, vol. 39, 1990, pp. 1038–1052.

[6] J.Dupratan J.-M. Muller, "The CORDIC Algorithm: New Result of Fast VLSI Implementation," IEEE Transactions on Computers, vol. 42, 1993, pp. 168–178.

[7] S.Wang,V.piuri, and E.Swartzlander, "Hybrid CORDIC Algorithm", IEEE Transactions on Computers, vol.46,1997,pp.1202-1207.

[8] D.Timmermann and S. Dolling, "Unfolded Redundant CORDIC VLSI Architectures with Reduced Area and Power Consumption," in Int. Conf. on VLSI 97, Brasilia, Sept. 1997.

[9] A.Wassatsch, S. Dolling, and D.Timmermann, "Area Minimization of Redundant CORDIC Pipeline Architectures," in IEEE International Conference on Computer Design (ICCD'98), Austin Texas, October 1998.

[10] JAVIER VALLS., "Evaluation of CORDIC Algorithms for FPGA Design," IEEE Journal of VLSI Signal Processing 32, 2002, pp. 207-222.

[11] Dupart, J. and Muller, J.M., "The CORDIC Algorithm: New Results for Fast VLSI Implementation," IEEE Transaction on Computers, Vol.42, pp. 168-178, 1993.

[12] Andraka, R.J., "Building a high Performance Bit serial Processor in an FPGA," Proceeding of Design of SuperCon '96, Jan 1996.pp 5.1-5.21.

[13] Deprettere, E., Dewidle, P., and Udo, R., "Pipelined CORDIC Architecture for Fast VLSI Filtering and Array Processing, Proc.ICASS'84, 1984, PP. 41.A.6.1-41.A.6.4.

[14] Hu, Y.H., and Naganathan, S., "A Angle Recording Method for CORDIC Algorithm Implementation", IEEE Transactions on Computers, vol.42,pp 99-102, January 1993.

[15] Hu,Y.H.,and Naganathan, S., "A novel Implementation Using a CORDIC Processors," IEEE Transaction on ASSP, vol. 38, pp .352-354, 1990.

[16] Sibul, L.H. and Fogelsangar, A.L., "Application of Coordinate Rotation Algorithm to Singular Value Decomposition,"IEEE Int, Symp. Circuits and Systems pp.821-824, 1984 .

[17] Ahmed, H.M., Delsome, J., and Morf, M., "Highly Concurrent Computing Structure for Matrix Arithmetic and Signal Processing," IEEE Comput. Mag., Vol.15, 1982, pp. 65 -82.

[18] Raj Kumar Tomar, Praveen Kumar Singh, Krishna Raj, "A Review of CORDIC Algorithms and Architectures with Applications for Efficient

[19] Designing", International Journal of Scientific & Engineering Research, Volume 4, Issue 8, AUGUST 2013, ISSN 2229-5518.