ISSN 2047-3338

FPGA Implementation of a High Speed Multistage Pipelined Adder Based CORDIC Structure for Large Operand Word Lengths

Burhan Khurshid[1], Ghulam Mohd Rather[1] and Najeeb-ud-din Hakim[1]
[1]Department of ECE, National Institute of Technology, Srinagar, J & K, India

***Abstract*— CORDIC is an acronym for COordinate Rotation Digital Computer. It is a hardware-efficient, shift and add algorithm that is used in various digital signal processing applications for computing trigonometric, logarithmic, hyperbolic and other linear and transcendental functions. Traditionally the algorithm is implemented in hardware in two different styles: folded and unfolded. Unfolded structures are pipelined to increase the throughput of the structure. This paper implements the algorithm in normal unfolded style, but using multistage pipelined adders. The resulting structure is compared against the traditional unfolded and pipelined approaches and is shown to have an improved throughput. The structure has been coded in VHDL and implemented using Xilinx FPGA synthesis tool. The algorithm has been simulated for sine and cosine function evaluation. The simulations are carried out using Xilinx ISim tool.**

***Index Terms*—FPGA, Rotation Mode, Pipelined Adders and Unfolded Architecture**

I. INTRODUCTION

MANY of the algorithms used today in DSP require elementary functions such as trigonometric, inverse trigonometric, logarithmic, exponential, multiplication and division functions [1]. The traditional approach for implementing these functions was to look for some software solutions. These included the use of look-up tables, power series and polynomial expansion methods. These software based approaches, however, suffered from many drawbacks.

Look-up table methods, although fast, required large memory for high precision results. The use of power series was time consuming as it was too slow to converge to a desired precision.

With digital signal processing switching back to hardware efficient solutions [2], CORDIC came to picture. The CORDIC method of computation basically represents a compromise between the look-up table method and the power series method, wherein the precision of the result is preserved without having to use any considerable amount of on chip memory. With the advancement in VLSI technology, the design of high speed VLSI architectures for modern digital signal processing systems became a reality. This provided designers with significant impetus to map CORDIC algorithm into hardware structures. However, the frequent use of these architectures in modern DSP systems requires a rapid increase in performance accompanied by a reduction in cost and shorter time to market [1]. This is achieved by optimizing these structures for various performance parameters such as speed, power and area. One way of optimizing these structures is to incorporate such architectural modifications that enable them to be clocked at higher operating frequencies. The optimized structures are then implemented using suitable hardware platforms.

IC technology provides a variety of implementation formats for system designers. The implementation format defines the technology to be used, how the switching elements are organized and how the system functionality will be materialized. Perhaps the most advanced implementation platform used today is the reconfigurable platform. FPGAs are reconfigurable systems that are often used as co-processors to implement CORDIC architectures. Historically, FPGAs have been slower, less energy efficient and generally achieved less functionality than their fixed ASIC counterparts. This has provided enough scope for designers to optimize FPGA based structures for parameters like throughput, power, area etc.

The rest of the paper is organized in the following manner. Section II discusses the CORDIC algorithm and its operating mode for sine and cosine function evaluation. Section III discusses the folded and unfolded CORDIC architectures. Section IV discusses the pipelined addition. Section V provides the implementation and simulation results and based on the comparison metrics, performance evaluation of folded, unfolded, pipelined and proposed CORDIC architectures is carried out.

This work has been carried out in SMDP-II VLSI laboratory of the Electronics and Communication Engineering Department, of National Institute of Technology Srinagar, India. This SMDP – II VLSI project is funded by Ministry of Communication and Information Technology, Government of India. Authors are grateful to the Ministry for the facilities provided under this project.

B. Khurshid is with the Electronics and Communication Department National Institute of Technology, Srinagar, J & K, India (phone: 9797875163; e-mail: khurshid_burhan@yahoo.com).

G. M. Rather is with the Electronics and Communication Department National Institute of Technology, Srinagar, J & K, India.

N. Hakim is with the Electronics and Communication Department National Institute of Technology, Srinagar, J & K, India.

II. CORDIC ALGORITHM

The CORDIC algorithm provides an iterative method of performing vector rotations by arbitrary angles using only shift and add operations [3]. The algorithm, credited to Volder [4], is derived from the general (Givens) rotation transform:

$$x' = x \cos ø - y \sin ø \quad (1)$$

$$y' = x \sin ø + y \cos ø \quad (2)$$

This rotates a vector in a Cartesian plane by the angle $ø$. These can be rearranged so that:

$$x' = \cos ø \ [x - y \tan ø] \quad (3)$$

$$y' = \cos ø \ [y + x \tan ø] \quad (4)$$

The rotation angles are restricted so that, $\tan ø = \pm 2^{-i}$. This reduces the multiplication operation by the tangent term to simple shift operation. Any given target angle $ø$ can be decomposed into a sequence of smaller micro rotations. Thus $ø$ is decomposed as a sequence of elementary rotations:

$$ø = \Sigma \alpha_i \quad (5)$$

Using these basic ideas we have the basic iterative rotations as:

$$x_{i+1} = \cos \alpha_i \ [x_i - y_i \tan \alpha_i \ ] \quad (6)$$

$$y_{i+1} = \cos \alpha_i \ [y_i + x_i \tan \alpha_i \ ] \quad (7)$$

The rotation angles are restricted so that:

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

This assures that the multiplication by the tangent term is reduced to simple shifting operation.

$$x_{i+1} = [x_i - y_i \tan \alpha_i \ ] / (1 + \tan^2 \alpha_i)^{1/2}$$

$$y_{i+1} = [y_i + x_i \tan \alpha_i \ ] / (1 + \tan^2 \alpha_i)^{1/2}$$

Rearranging:

$$x_{i+1} = [x_i - y_i (\pm 2^{-i})] / (1 + 2^{-2i})^{1/2}$$

$$y_{i+1} = [y_i + x_i (\pm 2^{-i})] / (1 + 2^{-2i})^{1/2}$$

Or

$$x_{i+1} = K_i [x_i - y_i . d_i . 2^{-i}] \quad (8)$$

$$y_{i+1} = K_i [y_i + x_i . d_i . 2^{-i}] \quad (9)$$

Where,

$K_i = 1/(1+2^{-2i})^{1/2}$; known as scale constant.

$d_i = \pm 1$; known as decision function.

Removing the scale constant from the iterative equations yields a shift-add algorithm for vector rotation. The product of the K_i's can be applied elsewhere in the system or treated as part of a system processing gain. That product approaches 0.6073 as the number of iterations goes to infinity. Therefore, the rotation algorithm has a gain, *An*, of approximately 1.647. The exact gain depends on the number of iterations, and obeys the relation:

$$A_n = \Pi [1+2^{-2i}]^{1/2}$$

The angle accumulator adds a third difference equation to the CORDIC algorithm:

$$z_{i+1} = z_i - \alpha_i$$

$$z_{i+1} = z_i - d_i \tan^{-1} (2^{-i}) \qquad \{ \tan \alpha_i = \pm 2^{-i} \}$$

For a single CORDIC micro-rotation the resulting equations are:

$$x_{i+1} = x_i - y_i . d_i . 2^{-I} \quad (10)$$

$$y_{i+1} = y_i + x_i . d_i . 2^{-I} \quad (11)$$

$$z_{i+1} = z_i - d_i \tan^{-1} (2^{-i}) \quad (12)$$

The CORDIC rotator is normally operated in one of two modes. In rotation mode, the angle accumulator is initialized with the desired rotation angle. The rotation decision at each iteration is made to diminish the magnitude of the residual angle in the angle accumulator. The decision at each iteration is therefore based on the sign of the residual angle after each step. The CORDIC equations are:

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

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

$$z_{i+1} = z_i - d_i \tan^{-1} (2^{-i}),$$

Where,

$d_i = -1$ if $z_i < 0$

$+1$, otherwise

After n iterations we are provided with following results:

$$x_n = A_n [x_0 \cos z_0 - y_0 \sin z_0] \quad (13)$$

$$y_n = A_n [y_0 \cos z_0 + x_0 \sin z_0] \quad (14)$$

$$z_n = 0 \quad (15)$$

Setting the *y* component of the input vector to zero reduces the rotation mode result to:

$$x_n = A_n . x_0 \cos z_0 \quad (16)$$

$$y_n = A_n . x_0 \sin z_0 \quad (17)$$

By setting x_0 equal to $1/A_n$, the rotation produces the unscaled sine and cosine of the angle argument z_0.

III. CORDIC ARCHITECTURES

In general, CORDIC architectures can be broadly classified as folded and unfolded, based upon the hardware realization of the three iterative equations [5]. A direct duplication of equations 10, 11 and 12 into hardware results in folded architecture. Folded architectures have to be multiplexed in time domain so that all the iterations are carried out in a single functional unit. This provides a means for trading area for speed [6] in signal processing architectures. One of the widely used folded architectures is implementing the entire CORDIC core using a word serial design.

A. Folded word serial design

A folded word serial design [7], [8], also called iterative bit-parallel design is obtained simply by duplicating each of the three difference equations in hardware as shown in Fig. 1.

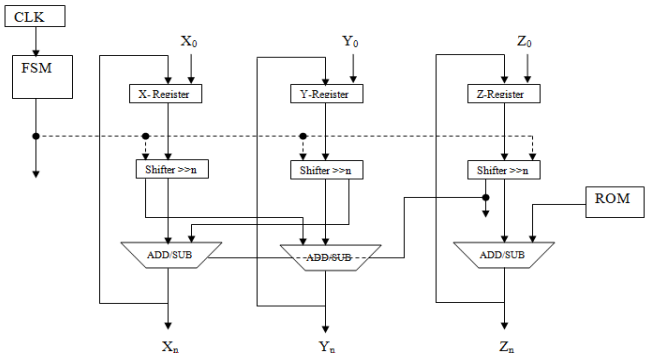


Fig. 1. A folded word serial CORDIC

Being a shift- add algorithm, each individual unit consists of an adder/subtractor unit, a shifter and a register for holding the computed values after each iteration. To start with, the initial values are fed into each branch via a multiplexer. The value in the *z* branch determines the operation of the adder-subtractor unit. Signals in the *x* and *y* branch pass through the shifter units and are then added to (or subtracted from) the unshifted signal in the opposite path. The *z* branch arithmetically combines the register values with the values taken from a lookup table whose address is changed according to the number of iteration. The result of this operation determines the nature of operation for the next iteration. After *n* iterations the results are directly read from the adder/subtractor units. A finite state machine is used to keep a track of shifting distances and the ROM addresses. Since the adder/subtractor unit and the shifters in each path are shared on time basis this conventional approach of implementing the CORDIC algorithm is not suitable for high speed applications [7]. Another disadvantage is with respect to the shift operations. When implemented in hardware the shifters have to change the shift distance with the number of iteration. For large number of iterations these require a high fan in and reduce the maximum speed for the application [7, 9].These shifters do not map well into FPGA architectures and if implemented require several layers of logic. The result is a slow design that uses large number of logic cells. In addition the output rate is also limited by the fact that the operation is performed iteratively and therefore the maximum output rate equals *1/n* times the clock rate, where *n* is the number of iterations.

B. Unfolded parallel design

The iterative nature of the CORDIC processor discussed above demands that the processor has to perform iterations at *n* times the data rate. The iteration process can be unfolded [8, 10] so that each of *n* processing elements always performs the same iteration. A direct application of the unfolding transformation is to design parallel processing architectures from serial processing architectures. At the word level, this means that word-parallel architectures can be designed from word-serial architectures [11]. An unfolded CORDIC processor is shown in Fig. 2.

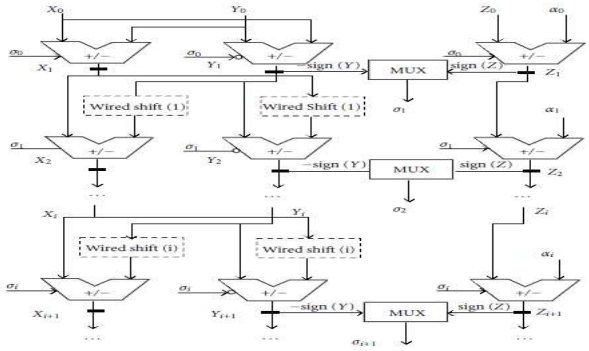


Fig. 2. Unfolded CORDIC design

Unfolding the CORDIC processor results in two significant changes. First, the shifter in each unit is of fixed shift i.e. it has to perform a constant shifting operation in each stage. Thus the shifter needs not to be updated as in the iterative structure. This makes their implementation in FPGAs quite feasible. Second, the unfolding process eliminates the use of ROM from the processor which was required to hold the constant angle values during each iteration. Those constants can be hardwired instead of requiring storage space. The entire CORDIC processor is thus reduced to an array of interconnected adder-subtraction units. The need for registers is also eliminated, making the unrolled processor strictly combinatorial. Another advantage of the unrolled design is that the processor can be easily pipelined [12] by inserting registers between the adder-subtraction units. In case of most FPGA architectures there are already registers present in each logic cell, so the addition of the pipeline registers has no additional hardware cost.

IV. PIPELINED ADDERS

A very useful implementation technique, especially for signal processing circuits, is pipelining. Consider a 128-bit adder made up of four 32-bit adders. A parallel (combinational) implementation is described in Fig. 3. The computation time (latency) of the circuit is roughly equal to 4.T, where T is the computation time of a 32-bit adder, so that the maximum sample rate of the input operands x and y is equal to 1/(4.T). The corresponding pipelined circuit is shown in Fig. 4: a register is inserted between the computation resources assigned to successive cycles, in such a way that a new addition can be started as soon as the first cycle of the preceding addition has been completed, that is, every T seconds. In this way the latency is still equal to 4.T. Nevertheless, the sample rate is equal to 1/T instead of 1/(4.T).

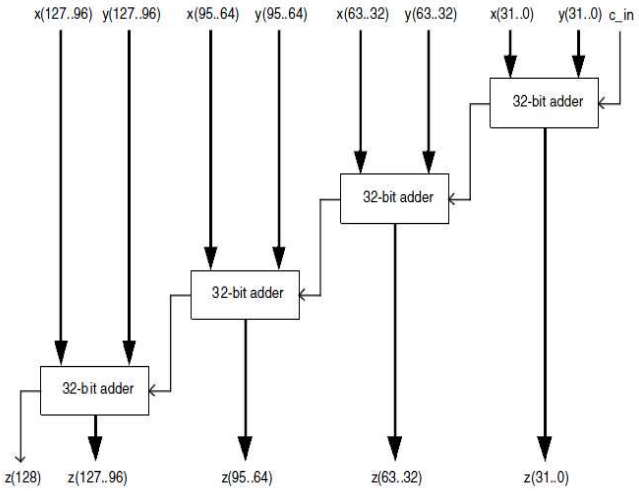


Fig. 3. Parallel Adder

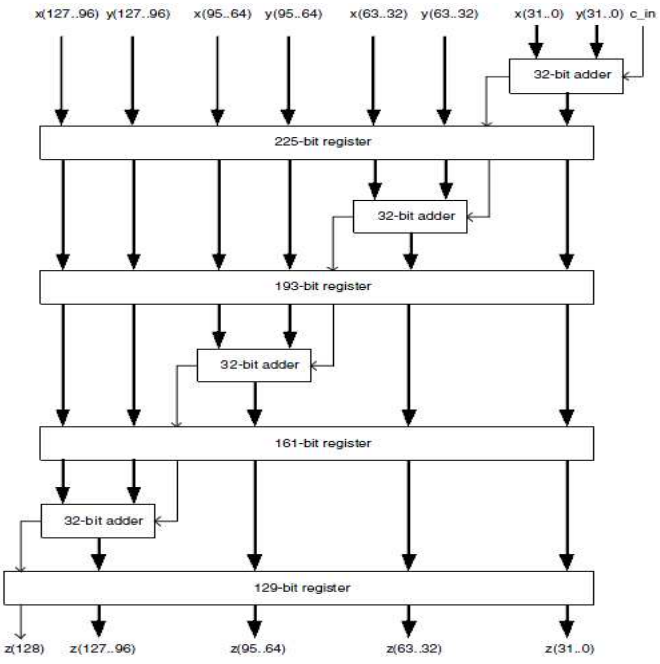


Fig. 4. Pipelined Adder

Most often, the basic cell of the field programmable gate arrays includes a flip flop so that the insertion of pipeline registers does not necessarily increase the total cost, computed in terms of used basic cells. The pipeline registers could consist of flip-flops not used in the non pipelined version.

V. IMPLEMENTATION AND RESULTS

A. Methodology

The proposed multistage pipelined adder based CORDIC processor is implemented in seven stages and for a word length of 32 and 64 bits. The initial design entry is done using VHDL. The design translation is carried out in Xilinx ISE 12.4 [13]. The simulator database is then analyzed for different performance parameters and logical conclusions are drawn. The core is implemented with the following synthesis description:

Platform: FPGA
Family: Virtex5
Target device: XC5VLX30
Package: FF324

Fig. 5 shows the generated RTL schematic of the proposed multistage pipelined adder based CORDIC for one iteration.

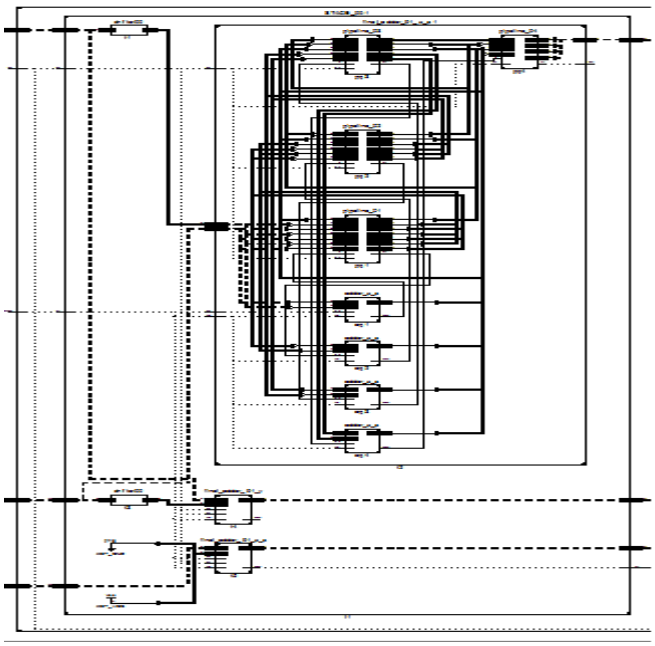


Fig. 5. RTL schematic of Proposed CORDIC

B. Simulations

The generated core has been simulated for sine and cosine functions by operating it in the rotation mode. Fig. 6 shows the simulated sine and cosine values of certain angles calculated using 32-bit proposed CORDIC.

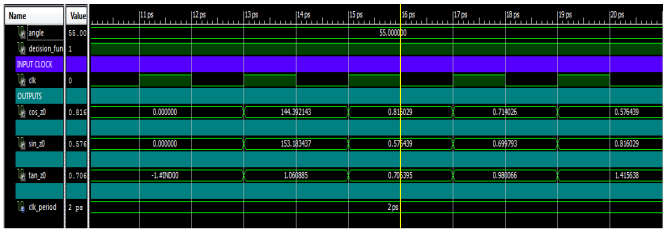


Fig. 6. Simulation result for 32-bit proposed CORDIC

C. Analysis and Results

The proposed structure is analyzed for different performance parameters. Table 1 gives the maximum operating frequency comparison for the 4-stage pipeline proposed, unfolded and pipelined structures for word lengths of 32 and 64 bits.

Table 1: Throughput comparison for 32 and 64-bit CORDIC

Parameter	CORDIC architectures.					
	Proposed (4-stage pipeline)		Unfolded (parallel)		Unfolded (pipelined)	
	32-bit	64-bit	32-bit	64-bit	32-bit	64-bit
Max. operating frequency	327.8 MHz	293.5 MHz	31.6 MHz	16.99 MHz	125.9 MHz	74.6 MHz

It is observed that when frequency response of the CORDIC structures is concerned, the proposed architecture has maximum operating frequency. The proposed structure can thus be clocked at high frequencies resulting in faster operations. The throughput can further be increased by increasing the number of pipeline stages of the adder unit. Table 2 below gives the throughput comparisons for 64-bit CORDIC using 8 pipeline stages in the adder unit.

Table 2: Throughput comparison for 64-bit CORDIC using 8 pipeline stages

Parameter	CORDIC architectures.		
	Proposed (8-stage pipeline)	Unfolded (parallel)	Unfolded (pipelined)
	64-bit	64-bit	64-bit
Max. operating frequency	325.850MHz	16.994 MHz	74.64 MHz

The proposed structure is analyzed for power consumption during operation. Since the structure is clocked at higher frequencies the power dissipation values are slightly greater than the conventional structures. Table 3 below gives the power comparison for the three structures. The power analysis has been carried out using Xpower analyzer tool

Table 3: Power comparison for 32-bit CORDIC

Instance (resource)	CORDIC architectures		
	Proposed (4-stage pipeline)	Unfolded (parallel)	Folded (pipelined)
power (clock)	36.97 mW	--	17.75 mW
power (logic)	20.28 mW	13.87 mW	9.20 mW
power (signals)	19.86 mW	11.01 mW	12.33 mW
power (IOs)	180.26 mW	196.07 mW	196.64 mW
power (leakage)/ quiescent	382.50 mW	382.30 mW	382.21 mW
dynamic power	257.3 mW	220.95mW	235.92 mW
total power dissipation	639.86 mW	603.25 mW	618.13 mW

Finally the three designs are analyzed for area consumption in terms of resource utilization and the results are tabulated in Table 4 below:

Table 4: Area comparison for 32-bit CORDIC

parameter	CORDIC architectures.		
	Proposed (4-stage pipeline)	Unfolded (parallel)	Folded (pipelined)
No. of Registers	1,579	--	678
No. of LUTs	2,376	1093	1006
No. of logic blocks used	1,705	1093	1006
No. of occupied Slices	989	589	336
No. of LUT Flip Flop pairs used	2,532	1093	1013
No. of bonded IOBs	194	193	194

As expected, the proposed structure is not an efficient user of logic since the introduction of pipeline stages requires additional registers to be inserted in between adders. However, since the implementation is FPGA based registers are already present in each logic cell and does not add to the overall cost of implementation.

VI. CONCLUSION

This paper proposed a multistage pipelined adder based CORDIC structure. The proposed structure was meant for modern day high speed and large operand DSP applications. The performance analysis of the proposed structure was carried out and the results were compared with the traditional pipelined and unfolded CORDIC architectures. The

implementation was targeted for FPGA devices. The proposed design uses the resources extensively but shows the best latency per sample and thus maximum throughput rate. It should be noted that the proposed structure will have no effect on the precision of the results, as the precision is the function of number of stages in the CORDIC core and not the approach used to implement the core. The paper provides a future scope for further improvements in the overall throughput by utilizing the same pipelined architecture for higher radix CORDIC algorithms. Also the structure may be optimized for power and area parameters by the repeated use of adders; this however will result in the decrease in the throughput of the overall structure.

REFERENCES

[1] B. Khurshid, G.M.Rather and N.Hakim, "Performance Comparison of Non-redundant and Redundant FPGA based Unfolded CORDIC Architectures,"in *International Journal of Electronics and Communication Technology,* vol. 3, issue 1 pp 85-89, Marcch 2012.

[2] B. Khurshid, G.M.Rather and N.Hakim, "Performance Analysis of CORDIC Architectures Targeted for FPGA Devices," in *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 2, issue 2 February 2012.

[3] J. E. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electronic Computing,* volume EC-8, pp 330 – 334, 1959.

[4] E. Deprettere, P. Dewilde, and R. Udo, "Pipelined CORDIC Architecture for Fast VLSI Filtering and Array Processing," *Proc. ICASSP'84*, 1984, pp. 41.A.6.1- 41.A.6.4.

[5] E. Meggitt, "Pseudo division and pseudo multiplication processes," *IBM Journal*, vol. 6, no. 2, pp. 210–226, 1962.

[6] C.H.Lin and A.Y. Wu, "Algorithm and Architecture for High-Performance Vector Rotational DSP Applications," *Regular IEEE Transactions: Circuits and Systems I,* Volume 52, pp 2385- 2398, November 2005.

[7] J.S. Walther, "A unified algorithm for elementary functions," *Proc. Spring. Joint Comp. Conf.*, vol. 38, pp. 379-385, 1971.

[8] M.D. Erecegovac and T. Lang, *Digital Arithmetic*, Elsevier, Amsterdam, the Netherlands, 2004.

[9] R.Andraka, "A survey of CORDIC algorithms for FPGA based computers," FPGA '98, in *ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, pp 191-200, 1998.

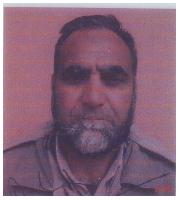
[10] Y.H. Hu, "Pipelined CORDIC architecture for the implementation of rotational based algorithm," in *Proceedings of the International Symposium on VLSI Technology, Systems and Applications*, p. 259, May 1985.

[11] K. K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation,* Wiley, 1999.

[12] A.A. De Lange, A.J. Van der Hoeven, E.F. Deprettere, and J. Bu, "An optimal floating-point pipeline CMOS CORDIC Processor," IEEE *ISCAS'88*, pp. 2043-47, 1988.

[13] ISE Simulator, Xilinx incorporation San Jose U.S.A, 2011.

Burhan Khurshid received the B.E. degree in Electronics and Communications Engineering from the Kashmir University, India, in 2008, the M.Tech degree in Communications and IT from National Institute of Technology, Srinagar, India in 2011. His research interests are in the field of Reconfigurable and DSP Design (system level) using VLSI.



G.M. Rather received the B.E. degree in Electronics and Communications Engineering from the Kashmir University, India, in 1981, the M. S. degree (1988) in Computer Communications and the Ph.D. degree (1997) from the Indian Institute of Sciences Bangalore India. He is currently Professor in the Department of EC E, N I T, Srinagar, India. His research interests are in the field of Communications and DSP Design using VLSI. He is a member of IETE India.

Najeeb-ud-din received the B.E. degree in Electronics and Communications Engineering from the Kashmir University, India, in 1985, the M. Eng. degree in Solid-state Electronics from the University of Roorkee, India, and the Ph.D. degree from the Indian Institute of Technology (IIT), Bombay, India, in 2003. He is currently Associate Professor in the Department of ECE, N I T, Srinagar, India. His research interests are in the field of SOI, CMOS Devices, Design, and Technology; in mixed-signal applications. He is a Senior Member of IEEE.