Volume 4, Issue 1
February 2015

ITEE Journal
Information Technology & Electrical Engineering

ISSN: - 2306-708X

A Constant Word-length Bit-Parallel Coordinate Rotation Digital Computer based Multiplier for Fixed-Point Digital Signal Processing Operations

[1]Burhan Khurshid, [2] Roohie Naaz Mir

[1]Department of Computer Sciences & Engineering, NIT, Srinagar

[2]Department of Computer Sciences & Engineering, NIT, Srinagar

E-mail: [1]burhan_07phd12@nitsri.net, [2]naaz310@nitsri.net

ABSTRACT

Fixed-point multiplication is frequently used in many DSP algorithms. This paper considers the design of a constant word-length bit-parallel fixed-point multiplier based on CORDIC algorithm. Traditional bit-parallel multipliers are designed using ripple-carry or carry-save logic. A comparative analysis of our implementation results against three widely used constant word-length bit-parallel fixed-point multipliers viz. ripple carry array multiplier, carry-save array multiplier and Bough-Wooley multiplier is presented in this paper. The comparison is made with respect to resource utilization, timing and power dissipation. The implementation is carried out for varying input word-lengths ranging from 4 to 32-bit parallel operands. Further, the implementation targets three different FPGA families viz. Spartan-6, Virtex-4, and Virtex-5. Our implementation achieves a reduction in resource usage by at least 30 %; increase in speed by at least 5 % and reduction in dynamic power dissipation by at least 20 %.

Keywords: *Fixed point arithmetic, DSP, ASIC, FPGA, CORDIC*

1. INTRODUCTION

Fixed point multipliers are frequently used in many digital signal processing (DSP) operations [1] [2] [3] [4]. They form an integral part of digital filters, an important class of Linear Time Invariant (LTI) systems designed to modify frequency response of the input signal. The goal of digital design is to maximize the performance while keeping the cost down [5]. In the context of general digital design, performance is measured in terms of the amount of hardware circuitry and resources required; the speed of execution (throughput and clock rate); and the amount of power dissipated. This demands for high speed, low area and low power realization of circuits used in these DSP systems [6] [7].

Traditional DSP implementations have focused mainly on processor-oriented solutions where the design process mainly consists of developing the necessary high-level code with some thought given to the underlying hardware architecture [8]. For high-speed applications some platform-based solutions such as application specific integrated circuits (ASIC) and structural ASICs have been used [2]. Recently field programmable gate arrays (FPGAs) have proven to be favored platform for VLSI design engineers. FPGAs offer many advantages over ASIC and programmable systems. The high speed and low power advantage of FPGAs over microprocessors is a sustainable trend for a wide variety of applications [9] [10] [11]. Some other advantages include design modifications post production, low non-recurring engineering (NRE) costs, re-configurable design approach etc. [12] [13].

CORDIC (COordinate Rotation DIgital Computer) [14] [15] is an iterative algorithm used for calculating various linear, trigonometric, hyperbolic and transcendental functions. The algorithm operates by rotating a vector, in linear, circular or hyperbolic coordinate systems, using only add and shift operations. CORDIC is unparalleled in its ability to encapsulate a diversity of math functions in one basic set of iterations [16]. Since the algorithm involves only add and shift operations it has very good hardware efficiency and a very minimal control overhead.

CORDIC algorithm has been applied to many different applications and has been used as a core arithmetic engine in many VLSI signal-processing implementations [17]. It has been used for computing the fast Fourier transform (FFT) [18] [19], the discrete cosine transform (DCT) [20], and the discrete Hartley transform [21]. A lot of work has focused on CORDIC based approaches for implementing various types of linear operations, including singular value decomposition (SVD) [22], Given's rotations [23], recursive least square (RLS) filtering [24] etc.

The rest of the paper is as follows. Section II discusses bit-parallel fixed-point multiplication and the traditional designs being used. Section III briefly discusses the CORDIC algorithm, its operating modes and how it can be used to perform constant word-length fixed point multiplication. Section IV discusses the hardware structures that are derived from the algorithm. Unrolled and pipelined structures are considered. Section V carries out the actual synthesis and implementation. A part of the section is dedicated to error analysis intended to compare the accuracy of the CORDIC approach. Conclusions are drawn in section VI and references are listed at last.

2. BIT-PARALLEL MULTIPLIERS

Bit-parallel multipliers process one whole word of the input sample each clock cycle and are ideal for high-speed applications. In this paper we consider three widely used bit-parallel fixed-point multipliers viz. parallel ripple-carry array (RCA) multiplier, parallel carry-save array (CSA) multiplier and Baugh-Wooley (BW) multiplier. The operands in each case are assumed to be represented in fixed-point 2's complement representation. Therefore, N-bit operands X and Y may be represented as:

$$X = x_{N-1}.x_{N-2}x_{N-3}\dots\dots x_1x_0 \quad (1)$$

$$Y = y_{N-1}.y_{N-2}y_{N-3}\dots\dots y_1y_0 \quad (2)$$

The most significant bit in each case is the sign bit with '0' denoting a positive number and '1' a negative number. The magnitude of these numbers lies in the range [-1, 1) and is given by:

$$X = -x_{N-1} + \sum_{i=1}^{N-1} x_{N-1-i}\, 2^{-i} \quad (3)$$

$$Y = -y_{N-1} + \sum_{i=1}^{N-1} y_{N-1-i}\, 2^{-i} \quad (4)$$

The value of the product P = X × Y is given by:

$$P = -p_{2N-2} + \sum_{i=1}^{2N-2} p_{2N-2-i}\, 2^{-i} \quad (5)$$

The product P may be represented as:

$$P = p_{2N-2}.p_{2N-3}p_{2N-4}\dots\dots p_1p_0 \quad (6)$$

In constant word length multiplication, the N-1 lower order bits in the product P are discarded, and the product is given by:

$$Z = -z_{N-1} + \sum_{i=1}^{N-1} z_{N-1-i}\, 2^{-i} \quad (7)$$

The constant word length product Z may be represented as:

$$Z = z_{N-1}.z_{N-2}z_{N-3}\dots\dots z_1z_0 \quad (8)$$

The product Z, therefore, is not a full-precision product [5]. Based on this concept of constant word length multiplication, three traditional approaches to bit parallel fixed point multiplication have been used. These are briefly discussed below:

2. 1. Parallel ripple-carry array multiplier

In parallel ripple-carry array multiplier, the carry is rippled to the adder to the left in the same row [5]. Thus, within a row each adder has to wait for the carry input to perform its computation. In other words there exists an intra-iteration constraint between any two adjacent adder nodes within a row, assuming there is no pipelining involved. Owing to this ripple-carry nature the critical paths involved are quite large which limits the speed of multiplication. Figure 1.a shows a 4-bit RCA fixed point multiplier.

2. 2. Parallel Carry-Save Array Multiplier

In the carry-save array multiplier, the carry outputs are saved and used in the adder in the next row. In this case, the partial product is replaced by a partial sum and a partial carry, which are saved and passed to the next row. The advantage of carry-save addition is that the additions at different bit positions in the same row are now independent of each other and can be carried out in parallel, which essentially speeds up the addition phase of each cycle, and hence speeds up the multiplication [5]. The addition of the partial sum and the partial carry at the last step is performed by a vector merging adder (VMA), which may be implemented either as a ripple-carry adder or a carry-save adder. Figure 1.b shows a 4-bit CSA fixed point multiplier.

2. 3. Bough-Wooley Multiplier

The difficulty of 2's complement multiplication lies in handling the sign bits of the multiplicand and multiplier. An efficient way to overcome this problem is provided by the Bough-Wooley multiplication algorithm. The algorithm is an efficient way to handle the sign bits and helps in designing regular multipliers using 2's complement operands. The Baugh-Wooley multiplication may be implemented as either a carry-ripple array or a carry-save array [5]. Figure 1.c shows a 4-bit BW multiplier implemented as a carry save array.

With 2's complement representation a correct final result is guaranteed even if there is an intermediate overflow [5]. The magnitude of the number in (8) will always be less than 1. Since the magnitude of the filtering coefficients that determine the frequency behavior in DSP filtering operations is always less than 1, this type of representation is appropriate.

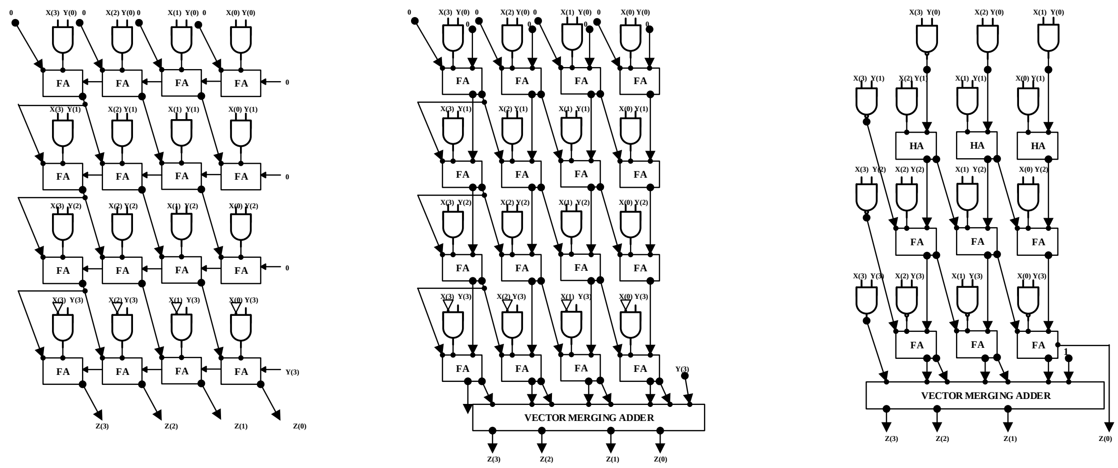

Figure 1 Constant word-length multipliers. a) RCA multiplier. b) CSA multiplier. c) BW multiplier.

Volume 4, Issue 1
February 2015

ITEE Journal
Information Technology & Electrical Engineering

ISSN: - 2306-708X

3. CORDIC ALGORITHM

CORDIC algorithm was first introduced by Volder [14] in 1959 as a technique for calculating the trigonometric functions required for real-time aircraft navigation. Since its introduction, the basic algorithm has been extended to evaluate a very rich set of functions from the one basic set of equations. Different versions of the CORDIC algorithm can be defined under the circular, hyperbolic, and linear coordinate systems [25]. These use a computation similar to that of the basic CORDIC algorithm, but can provide additional functions. The CORDIC engine can be either operated in the rotation mode or the vectoring mode. 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. In the vectoring mode, the CORDIC rotator rotates the input vector through whatever angle is necessary to align the resultant vector with the horizontal axis. The result of the vectoring operation is a rotation angle and the scaled magnitude of the original vector.

It is possible to capture the vectoring and rotation modes of the CORDIC algorithm in all three coordinate systems using a single set of unified equations. These equations are given as:

$$x_{i+1} = x_i - m\sigma_i y_i 2^{-i} \qquad (9)$$

$$y_{i+1} = y_i + \sigma_i x_i 2^{-i} \qquad (10)$$

$$z_{i+1} = \begin{Bmatrix} z_i - \sigma_i tan^{-1}(2^{-i}) \\ z_i - \sigma_i tanh^{-1}(2^{-i}) \\ z_i - \sigma_i(2^{-i}) \end{Bmatrix} \quad \begin{matrix} if\ m = 1 \\ if\ m = -1 \\ if\ m = 0 \end{matrix} \qquad (11)$$

Where,

$$m = \begin{Bmatrix} +1\ for\ circular\ coordinates \\ 0\ for\ linear\ coordinates \\ -1\ for\ hyperbolic\ coordinates \end{Bmatrix}$$

And the value of σ_i will determine the direction of rotation in the next iteration. For linear mode (m = 0) the CORDIC equations will be:

$$x_{i+1} = x_i \qquad (12)$$

$$y_{i+1} = y_i + \sigma_i x_i 2^{-i} \qquad (13)$$

$$z_{i+1} = z_i - \sigma_i(2^{-i}) \qquad (14)$$

After n iterations the resultant vectors will be:

$$x_n = x_s \qquad (15)$$

$$y_n = y_s + x_s z_s \qquad (16)$$

$$z_n = 0 \qquad (17)$$

If the initial value of *y*-vector (y_s) is chosen to be zero then after n iterations the *y*-vector will contain the product of *x* and *z* vectors. The accuracy of the results depends upon the number of iterations n.

4. CORDIC ARCHITECTURES

CORDIC algorithm can be implemented in a number of ways. A direct mapping of equations (12), (13) and (14) in hardware results in an iterative architecture. The iterative structure is bit-serial in nature resulting in slow structures. The iterative structure can be unrolled so that each of the n processing elements always performs the same iteration. Unfolded architectures have two advantages [26]. First, the shifters can be designed for fixed shifts, which means that they can be implemented in the wiring. Second, the ROM that holds the constant values for the *z*-branch need not be updated after every iteration. These constants can be hardwired instead of requiring storage space. The entire CORDIC processor is thus reduced to an array of interconnected adder- subtraction units as shown in figure 2.

The unrolled structure can be easily pipelined by placing registers along the feed forward cut sets indicated by dotted lines in figure 2. Pipelining the structure reduces the critical path along the *x*, *y* and *z* branches enabling it to be operated at higher clock speeds. Alternately, pipelining may also be utilized to reduce the power dissipation in a system. The dynamic power dissipation in a CMOS circuit is given by:

$$P = \alpha . C_{system} . V_{supply}^2 . f_{clk} \qquad (18)$$

Where,

α is the switching activity;

C_{system} is the total capacitance of the structure;

V_{supply} is the supply voltage and

f_{clk} is the clock frequency.

In a pipelined system the critical path is reduced such that the capacitance to be charged/discharged in a single clock cycle ($C_{critical}$) is reduced by some factor, say *M*. If the same clock speed, f_{clk}, is maintained then only a fraction of the original capacitance $\left(\frac{C_{critical}}{M}\right)$ is being charged/discharged in the same amount of time that was previously needed to charge/discharge the capacitance $C_{critical}$. In other words the supply voltage, V_{supply}, can be reduced to β V_{supply}, where β is a positive constant less than 1. The power consumption for the pipelined structure is thus given by;

$$P_{pip} = \alpha . C_{system} \beta^2 V_{supply}^2 f_{clk} \qquad (19)$$

$$P_{pip} = \beta^2 P \qquad (20)$$

The clock period is usually set equal to the maximum propagation delay in a circuit. Thus, for the original structure of figure 1;

$$T_{original} = \frac{C_{critical} V_{supply}}{k(V_{supply} - V_t)^2} \qquad (21)$$

Volume 4, Issue 1
February 2015

ITEE Journal
Information Technology & Electrical Engineering

ISSN: - 2306-708X

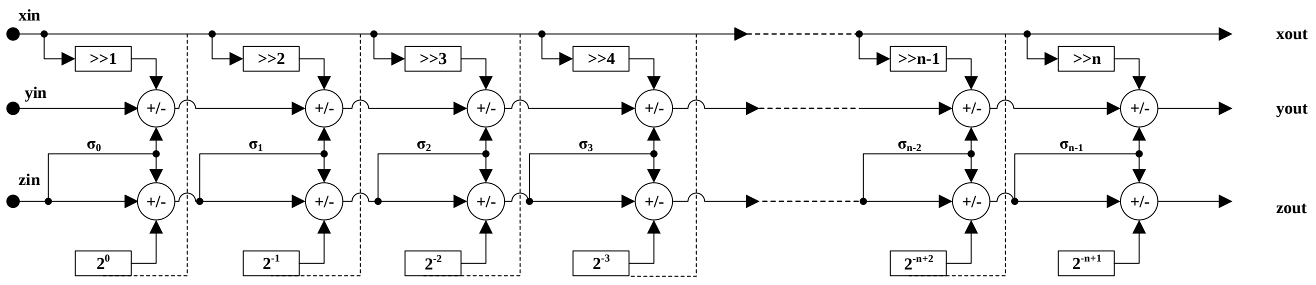

Figure 2 Unrolled CORDIC multiplier with dotted pipeline registers

For the pipelined structure;

$$T_{pip} = \frac{\frac{C_{critical}}{M}\beta V_{supply}}{k(\beta V_{supply} - V_t)^2} \tag{22}$$

Since the same clock speed is maintained for both the structures, the following quadratic equation can be used to find an appropriate value for β;

$$\frac{C_{critical}V_{supply}}{k(V_{supply} - V_t)^2} = \frac{\frac{C_{critical}}{M}\beta V_{supply}}{k(\beta V_{supply} - V_t)^2}$$

$$M(\beta V_{supply} - V_t)^2 = \beta(V_{supply} - V_t)^2 \tag{23}$$

5. SYNTHESIS AND IMPLEMENTATION

5. 1. Methodology

The implementation in this work is targeted at three different FPGA families viz. Spartan-6, Virtex-4 and Virtex-5. Only LX series has been considered as it is apt for general logic applications. The CORDIC engine is designed in twelve stages. Since most of the computations are performed in the initial stages, a 12- staged CORDIC ensures a good accuracy while maintaining a fast convergence. The implementation is carried for an input operand length varying from 4 to 32 bits. The parameters considered are resource utilization, timing and dynamic power dissipation. Resource utilization is considered in terms of on chip FPGA resources being used. Timing refers to the clock speed of a design and is limited by the setup time of the input / output registers, propagation and routing delay associated with the critical path, clock to output time associated with the flip flops and the skew between the launch (input) register and the capture (output) register. The digital clock managers (DCMs) inherent to modern FPGAs have been used to map the clocking resources. This ensures that there is no problem of skew in the mapped structure. Timing analysis is done to provide information about the speed/throughput of the system. Dynamic power dissipation is related to charging and discharging of node capacitances along the different switching elements. To ensure a fair comparison, similar test benches have been used for all the implemented designs i.e. the input statistics remain the same in each case. The initial design entry is done using VHDL. The constraints relating to the period and offsets are duly provided and a complete timing closure is ensured. The design synthesis, mapping and translation are carried out in Xilinx ISE 12.4 [27] and the simulator database is then analyzed for on-chip resources, throughput and timing metrics. Power metrics are obtained using Xpower analyzer.

5. 2. Experimental results

Table 1 gives a comparison of the on chip resources utilized by different multipliers for an input word-length of 16 bits on Spartan-6 device.

Table 1 Resource utilization for different multipliers

On-chip resource	**RCA**	**CSA**	**BW**	**CORDIC**
No. of Slice Registers	32	32	32	97
No. of slice LUTs	943	637	583	364
No. of occupied slices	361	253	206	125

Further analysis is carried out by implementing the different multiplier structures on different FPGA families for varying input word-lengths. The metrics obtained from the synthesizer database are plotted as a function of operand word-lengths and are presented in figures 3, 4 and 5. For simplicity we have considered only the occupied slices in each case. It is observed that there is a substantial reduction in the number of occupied slices for the CORDIC based multiplier. This is due to the inherent simple structure of the CORDIC algorithm that requires only additions and shift operations.

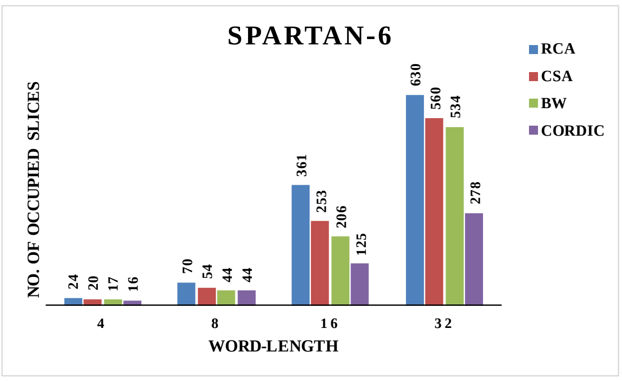

Figure 3 Resource utilization on Spartan-6 FPGA

Volume 4, Issue 1
February 2015

ITEE Journal
Information Technology & Electrical Engineering

ISSN: - 2306-708X

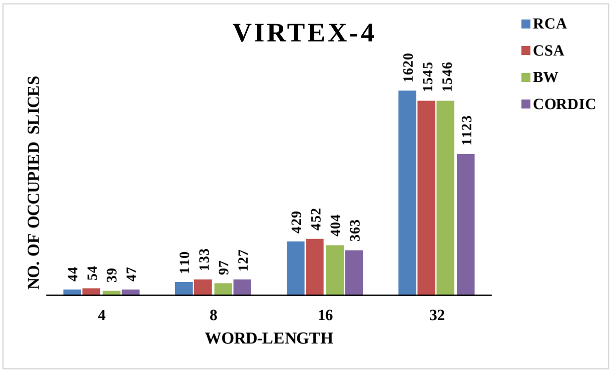

Figure 4 Resource utilization on Virtex-4 FPGA

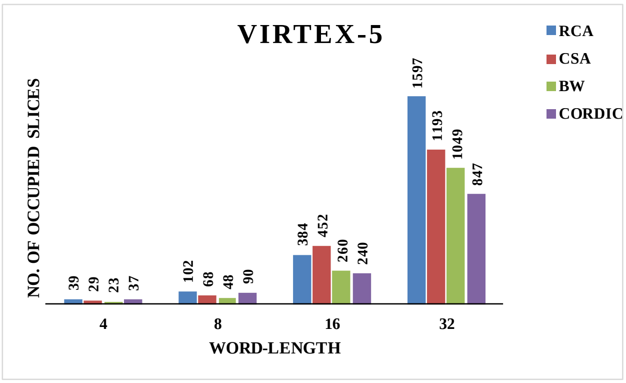

Figure 5 Resource utilization on Virtex-5 FPGA

Table 2 provides a comparison of the maximum achievable clock rates post implementation for a word length of 16 bits. The target family is Spartan-6. The CORDIC based multiplier has a better timing closure in terms of the relationship between an external clock pad and its associated data-in or data-out pad. This is indicated by the offset-in and offset-out metrics from the timing database of the synthesizer. Also, the delays associated with the critical path is quite low in CORDIC and is primarily limited by the adder logic delay and the associated interconnects. Note that the analysis is done for a CORDIC multiplier that is implemented in twelve stages. A further reduction in the combinational delay and the associated route is possible by reducing the number of stages in CORDIC. This, however, will affect the accuracy of the end results. Pipelining the structure results in the reduction of the critical path and thus high clock frequencies. Further analysis is carried out by plotting the maximum achievable speed against the operand word lengths for different target families. The results are shown in figures 6, 7 and 8. Again for simplicity only the maximum achievable speeds have been considered.

Table 2 Timing analysis for different multipliers on spartan-6

Timing Parameter	RCA	CSA	BW	CORDIC	PIPELINED CORDIC
Maximum frequency (MHz)	30.67	50.182	50.117	53.4	115.234
Minimum available offset-in (ns)	5.112	6.017	5.346	4.593	3.973
Minimum available offset-out (ns)	11.727	11.851	10.95	10.831	10.474

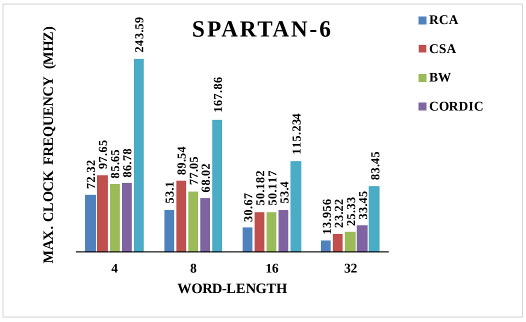

Figure 6 Maximum operating frequency on Spartan-6

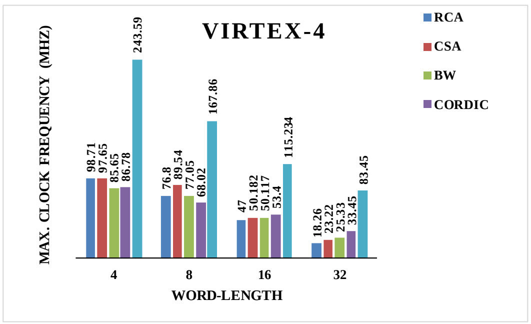

Figure 7 Maximum operating frequency on Virtex-4

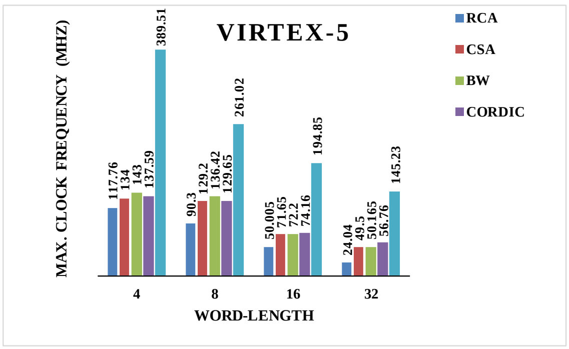

Figure 8 Maximum operating frequency on Virtex-5

Finally static and dynamic power dissipation for different structures is considered. Because an FPGA is programmable, it is only natural to look into minimizing the power dissipated. The static power dissipation in an FPGA consists of the device static power dissipation and the design static power dissipation. Although design static power is a very small percentage of the dynamic power dissipated, the device static power is device specific and is quite high. The dynamic power dissipation is a function of the input voltage (V^2), the clock frequency (f_{clk}), the switching activity (α), the total capacitance seen by a particular node (C_L) and the number of elements used (σ). The analysis was done for a constant supply voltage and at maximum operating frequency for each structure. To ensure a reasonable comparison the test vectors provided during post route simulation were selected to represent the worst case switching activity for data coming into the multiplier block. Same test benches were used for all the synthesized structures. The design node activity from the simulator database along with the power constraint file (PCF) was used for power analysis in the Xpower analyzer tool. Table 3 shows the power dissipated in various resources for operand length of 16 bits. The targeted device is Spartan-6.

The power dissipated in the clocking resources varies with the clock activity (clock frequency) as provided in the PCF. However, the capacitance C_L, which needs to be driven at each toggling node, varies with the type, fan-out, and capacitance of the logic and routing resources used in the design. The CORDIC multiplier uses fewer resources and has a reduced fan-out of non-clocking nets. This is indicated in table 4 where the average fan-out of non-clocking nets for different multipliers has been enlisted for a 16-bit operand word-length. The reduction in on-chip resources being utilized also leads to a reduction in the power dissipated in the logic. The reduction in the power dissipation in the signals and I/Os is indicative of the fact that the CORDIC based multiplier also tends to relax the signal transition rates for the duration of operation.

Table 3 Power dissipation on spartan-6 device

FPGA resource	**Power dissipated (mW)**				
	RCA	**CSA**	**BW**	**CORDIC**	**PIPELINED CORDIC**
Clock	0.9	1.24	1.22	1.98	3.51
Logic	2.34	1.14	1.71	0.8	0.64
Signals	4.41	2.66	3.29	1.92	0.84
I/Os	6.01	5.44	4.99	3.51	2.45
Dynamic	13.66	10.48	11.21	8.21	7.44
Quiescent	17.6	17.55	17.56	17.45	12.38
Total	31.26	28.03	28.77	25.66	19.82

Table 4 Average fan-out of non-clocking nets

Multiplier design	**Average fan-out of non-clock nets**
RCA	5.52
CSA	4.75
BW	4.78
CORDIC	4.18
PIPELINED CORDIC	3.76

For pipelined CORDIC the structure may be operated at the original frequency if the application demands a reduction in power dissipation. An appropriate value of β can be obtained from (23) and the structure may be operated at a reduced voltage. The values indicated in table 3 are derived for a pipeline depth of 11 (i.e. M = 11) and a nominal frequency of the basic CORDIC multiplier (i.e. 53.4 MHz). This results in a β of 0.15, after applying threshold corrections. The pipelined structure when operated at the reduced voltage results in reduced static power dissipation as indicated in table 3. Further analysis is carried out by plotting the total power dissipation as a function of input word-length for different multiplier structures and for different FPGA families. The results are shown in figures 9, 10 and 11.

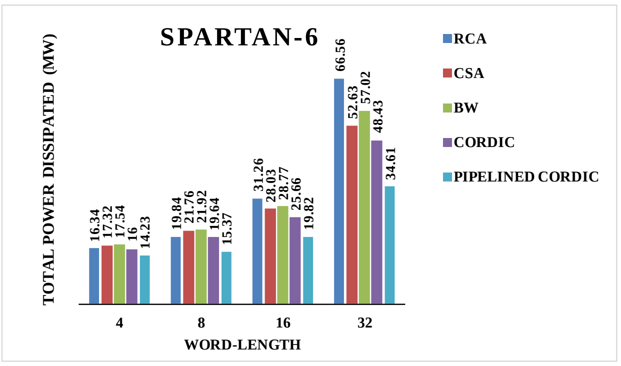

Figure 9 Power dissipation on Spartan-6

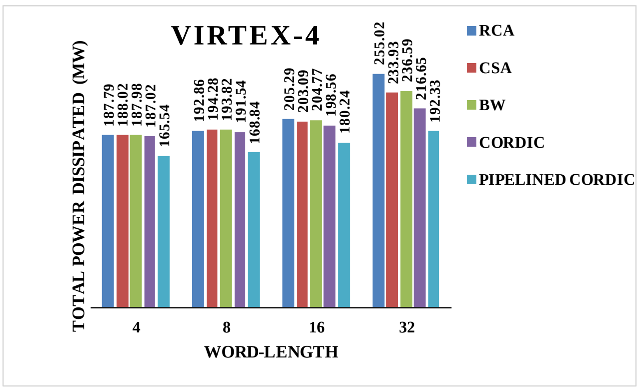

Figure 10 Power dissipation on Virtex-4

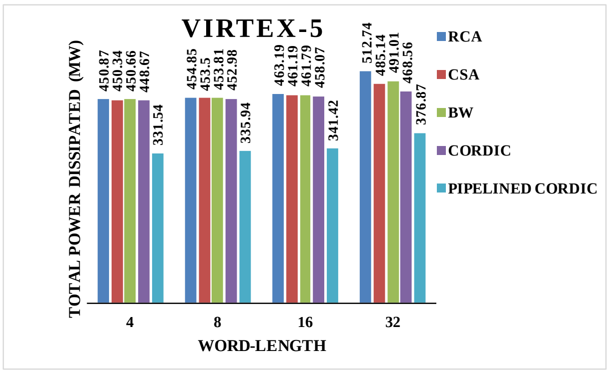

Figure 11 Power dissipation on Virtex-5

5. 3. Error analysis

CORDIC is an iterative algorithm and the accuracy of the end results depend on the number of iterative stages. However, for operands with magnitudes less than unity the error introduced is negligible. Figure 12 gives a comparison of the product values for a set of 16-bit input operands having same magnitude. The maximum error, when compared with the RCA multiplier is 3.1% and the average error is less than 3%. Note that the multiplier functionality is implemented in CORDIC by operating it in the linear mode and the results need not be scaled and can be directly read from the CORDIC engine. Thus there are no errors due to scaling. We have also plotted the percentage error variation against the operand word lengths. A set of random test vectors were generated for different word lengths and the average percentage error with respect to RCA multiplier was plotted against the word length. The results appear in the plot of figure 13.

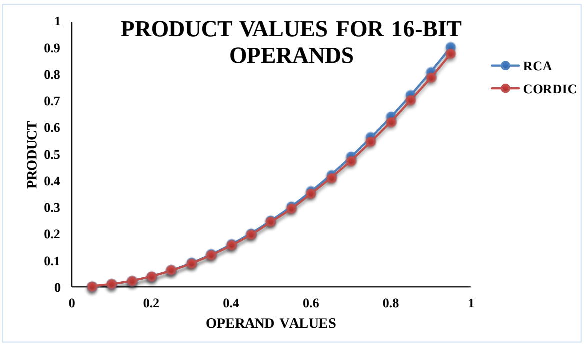

Figure 12 Product values for CORDIC multiplier

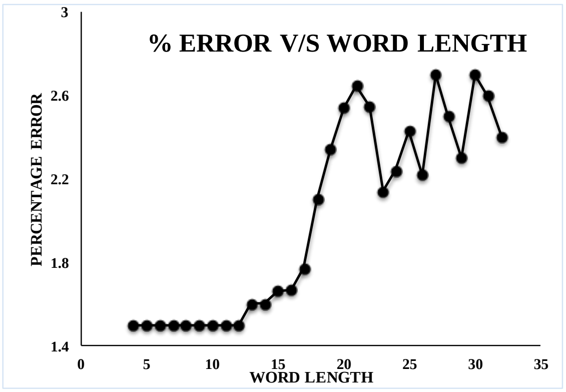

Figure 13 Percentage error for varying word lengths

6. CONCLUSIONS

This paper implemented a CORDIC based constant word length fixed-point multiplier by operating the CORDIC algorithm in the linear mode. The analysis and the experimental results carried out in this paper clearly indicate that a considerable improvement in performance is indeed achievable by using CORDIC based fixed-point multipliers. The results were compared against conventional fixed-point bit-parallel multipliers. Because of its simplicity in operation, the CORDIC based multiplier offers high-speed, low-power solution that is desirable in modern day DSP applications. Further, the structure can be easily pipelined for an increased operating frequency, thereby ensuring that the real-time throughput constraint that is inherent in many DSP systems is met. Alternately, the pipelined structure may be operated at reduced supply voltage resulting in low power hardware solutions.

REFERENCES

[1]. G. L. Narayan and B. Venkataramani, “ Optimization Techniques for FPGA based Wave Pipelined DSP Blocks,” IEEE Transc.Very Large Scale Integr. (VLSI) syst., vol. 13, No. 7, pp. 783-792, July 2005.

[2]. M. A. Ashour and H. I. Saleh, “An FPGA Implementation guide for some different types of Serial-Parallel Multiplier Structures,” Microelectronics Journal, vol. 31, pp. 161-168, 2000.

[3]. K. Compton, S. Hauck, “Reconfigurable Computing: A survey of Systems and Software,” ACM Computing Surveys, vol. 34, No. 2, pp. 171-210, June 2002.

[4]. R. Tessier, W. Burleson, “Reconfigurable Computing and Digital Signal Processing: Past, Present and Future,” Programmable Digital Signal Processors, Yu Wen Hue d, Marcel Dekker, pp. 147-186, 2002.

[5]. Keshab K. Parhi, "VLSI Digital Signal Processing Systems Design and Implementation," Wiley, 1999.

[6]. S. Shanthala and S. Y. Kulkarni, “VLSI Design and Implementation of Low Power MAC Unit with Block Enabling Technique,” European Journal of Scientific Research, ISSN 1450-216X, vol. 30, No. 4, pp. 620-630, 2009.

[7]. K. H. Chen, Y. H. Chen and Y. S. Chu, “A Versatile Multimedia Functional Unit Design using the Spurious Power Suppression Technique,” in Proc. IEEE Asian Solid-State Circuits conf., 2006, pp. 111-114.

[8]. Roger Woods, John McAllister, Gaye Lightbody and Ying Yi, “FPGA-based Implementation of Signal Processing Systems,” Wiley, 2008.

[9]. Z. Guo, W. Najjar, F. Vahid and K. Vissers, “A Quantitative Analysis of the Speed up Factors of FPGAs over Processors,” in Proc. Int. Symp. on FPGAs, ACM Press, 2004.

[10]. K. Underwood “FPGAs vs. CPUs: Trends in Peak Floating-Point Performance,” in Proc. Int. Symp. on FPGAs, ACM Press, 2001.

[11]. G. Stitt, F. Vahid and S. Nematbakhsh, “Energy Savings and Speed ups from Partitioning Critical Software Loops to Hardware in Embedded systems,” ACM Transc. Embedded Comput. Systems, vol. 3, pp. 218-232, 2004.

[12]. R. Tessier and W. Burleson, “Reconfigurable Computing for DSP: A Survey,” Journal of VLSI Signal Processing, vol. 28, pp. 7-27, 2001, Kluwer Academic Publisher.

[13]. T. J. Todman, G. A. Constantinides, S. J. E. Wilton, O. Mencer, W. Luk and P. Y. K. Cheung, “Reconfigurable Computing: Architecture and Design Methods,” in IEEE Proc. Comput. Digit. Tech., vol. 152, No. 2, March 2005.

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

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

[16]. S. Hauck and A. Dehon, “Reconfigurable Computing: The theory and Practice of FPGA-based Computing,” Morgan Kaufmann series, 2008.

[17]. D. H. Timmerman, B. J. Hosticka, G. Schimdt, “A Programmable CORDIC chip for Digital Signal Processing Applications,” IEEE Journal of Solid-State Circuits 26(9), September 1991.

[18]. A. M. Despain, “Very Fast Fourier Transform Algorithms for Hardware Implementation,” IEEE Transactions on ComputersC-28, May 1979.

[19]. A. M. Despain, “Fourier Transform Computers using CORDIC Iterations,” IEEE Transactions on Computers, 23 October 1974.

[20]. W. H. Chen, C. H. Smith, S. C. Fralick,” A fast Computational Algorithm for the Discrete Cosine Transform,” IEEE Transactions on Communications C-25, September 1977.

[21]. L. W. Chang, S. W. Lee, “Systolic Arrays for the Discrete Hartley Transform,” IEEE Transactions on Signal Processing 29(11), November 1991.

[22]. J. R. Cavallaro, F. T. Luk,” CORDIC arithmetic for an SVD processor,” Journal of Parallel and Distributed Computing 5, 1988.

[23]. C. M. Rader, “VLSI systolic arrays for adaptive nulling,” IEEE Signal Processing Magazine 13(4), July 1996.

[24]. B. Haller, J. Gotze, J. Cavallaro, ”Efficient Implementation of Rotation Operations for high-performance QRD-RLS filtering,” Proceedings of the International Conference on Application-Specific Systems, Architectures and Processors, July 1997.

[25]. D. Ercegovac, T. Lang, “Digital Arithmetic,” Morgan Kaufmann, 2004.

[26]. B. Khurshid, G. M. Rather and H. N. Shah, “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, March 2012.

[27]. http://www.xilinx.com

AUTHOR PROFILES

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. Currently he is pursuing his PhD in System design in the department of Computer Science and Engineering, NIT, Srinagar. His research interests include Reconfigurable architectures, Platform oriented solutions for arithmetic and DSP algorithms, Architectural and technology dependent optimizations targeted for FPGA platforms, etc. He has many publications in the related field and is a student member of IEEE. He is also a lifetime member of IETE.

Roohie Naaz Mir, received B.E. (Hons) in Electrical Engineering from University of Kashmir (India) in 1985, M.E. in Computer Science & Engineering from IISc Bangalore (India) in 1990 and Ph D from University of Kashmir, (India) in 2005. She is currently a Professor in the department of Computer Science & Engineering at NIT Srinagar, India. She is the co-author of many scientific publications in international journals and conferences. Her current research interests include reconfigurable computing, security and routing in wireless ad-hoc networks and sensor networks.