Journal of Theoretical and Applied Information Technology
20th December 2013. Vol. 58 No.2
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

MEMORY LESS Z PATH ELIMINATED VERY LOW POWER CORDIC BASED FFT PROCESSOR

[1]A. YASODAI, [2]Dr.A.V.RAMPRASAD
[1]Assistant Professor, Department of Electronics and Communication Engineering
Vickram College of Engineering, Enathi-630561, India
[1]yasodai0367@gmail.com
[2]Professor Department of Electronics and Communication Engineering
K.L.N College of Engineering, pottapalayam 630611, India
E-mail: [2]avramprasad2002@gmail.com

ABSTRACT

A complex number can be interpreted as a vector in imaginary plane. The vector rotation in the x/y plane can be realized by rotating a vector through a series of elementary angles. These elementary angles are chosen such that the vector rotation through each of them may be approximated easily with a simple shift and add operation, and their algebraic sum approaches the required rotation angle .This can be exercised by CORDIC algorithm in rotation mode. In this paper, we have proposed a pipelined architecture by pre computation of direction of rotation , radix-4 number representation, and the angle generator which has been processed in terms of hardware complexity , iteration delay and memory reduction. This CORDIC algorithm is implemented using signed digit arithmetic for the efficient implementation of rotational radix-4 CORDIC algorithm, eliminating z-path completely. Comparison of the proposed architecture with the available radix-2 and conventional radix-4 architectures is elaborated. The proposed algorithm also exercises an addressing scheme and the associated angle generator logic in order to eliminate the ROM usage for bottling the twiddle factors. The approached architecture for radix-4, 16-bit precision and N-point FFT was implemented on FPGA platform virtex 4 and is simulated and synthesized to validate the results (Xilinx ise 10.2.i). This contributes in the minimization of the dynamic power consumption of the proposed system to 28.52mW at 100MHz and 5.70mW at 20MHz for 16 point radix 4 FFT.

***Keywords**: CORDIC, memory less systems, FPGA, latency, Radix-4, speed, throughput, twiddle factor., FFT*

1. INTRODUCTION

Recently, signal processing systems are distinguished by a highly complex and real time operation. In order to live up to the real time requirements, FPGA-based fast hardware systems are affordable and provide a fair speedup. In recent times a system on chip (SOC) technology have current FPGAs to carry an entire system on a single chip. The FFT is a valuable constituent in most signal processing technologies. Analyzing its system, we understand that it presents serious demands of memory and the complex multipliers to perform rotations which call for very large area. Therefore, most architecture rather uses the CORDIC multipliers than the conventional multipliers.

Coordinate Rotation Digital Computer (CORDIC) algorithm is an iterative method utilizing simple shift and add operations, intended for the rapid evaluation of two dimensional vector rotations, and computation of trigonometric and transcendental functions [21]. Later, this algorithm was extrapolated for embedding hyperbolic functions [22]. As this algorithm draws growing attention in elementary function evaluation and signal processing applications [1,20], the VLSI realization has forced enormous interest and is practiced for hardware implementations. More of late, the betterments in the VLSI technology and the advent of EDA tools have broadened the applications of CORDIC algorithm to the field of biomedical signal processing [20], neural networks [19] and wireless communications[18] to mention a few.

The organization of the paper is as follows: In Section 2 , review of related works are discussed. In section 3, problem definition and solution is discussed. In section 4, basic CORDIC and FFT algorithms are discussed. In section 5, proposed

Journal of Theoretical and Applied Information Technology
20th December 2013. Vol. 58 No.2
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

methodology is discussed. In section 6, FFT with proposed methodology is discussed. In section 7, synthesis results are discussed.

2. REVIEW OF RELATED WORK

There has been several works in the literature associated with Radix-r CORDIC based FFT algorithm. Among them, a handful of significant researches are presented in this section. B. Lakshmi *et al.* in [1], presented a pipelined architecture using signed digit arithmetic for the VLSI efficient implementation of rotational radix-4 CORDIC algorithm, eliminating z-path completely. A detailed comparison of the their architecture with the available radix-2 architectures showed the latency and hardware improvement. Their architecture achieves latency improvement over the previously defined radix-4 architecture with a relatively small hardware overhead. Their architecture for 16-bit precision was implemented using VHDL and extensive simulations were performed to validate the results. The functionally simulated net list was synthesized for 16-bit precision with 90 nm CMOS technology library and the area-time measures were provided. Vachhani. L *et al.* in [2],presented two area-efficient algorithms and their architectures based on CORDIC. Their first algorithm eliminated ROM and required only low-width barrel shifters and their second algorithm eliminated barrel shifters completely. As a consequence, both the algorithms consume approximately 50% area in comparison with other CORDIC designs. Further, their algorithms were applicable to the entire range of angles.

Erdal Oruklu*et al.* in [3] presented a pipelined, reduced memory and low power CORDIC-based architecture for FFT implementation. Their algorithm utilized a new addressing scheme and the associated angle generator logic in order to remove any ROM usage for storing twiddle factors. As a case study, they implemented the radix-2 and radix-4 FFT algorithms on FPGA hardware. The synthesis results match the theoretical analysis and it was observed that more than 20% reduction was achieved in total memory logic. In addition, the dynamic power consumption was reduced by as much as 15% by reducing memory accesses. Francisco J. Jaime *et al.* in [4] presented an enhanced version of the scaling-free CORDIC. Their new enhancements was implemented and tested, which were able to reach a 35% lower latency and a 36% reduction in area and power consumption compared to the original scaling-free architecture. Amritakar Mandal *et al.* in [8] presented the design of pipelined architecture for the computation of Sine and Cosine values based on application specific CORDIC processor. Their design of CORDIC in the circular rotation mode gives a high system throughput by reducing latency in each individual pipelined stage. Saving area on silicon substrate was essential to their design of pipelined CORDIC and that was achieved through the optimization in the number of micro rotations. The computed quantization error was also minimized using required number of iterations. Ray, K.C. *et al.* in [9]presented hardware efficient and flexible architecture for computing Log Polar Transformation (LPT) for real time applications in image processing and pattern recognition. Purely pipelined CORDIC architecture was used to get high throughput. Their architecture was able to use with maximum frequency of 125 MHz with area 9.5 mm using 0.18 µm CMOS technology. Lakshmi. B *et al.* in [10] presented a low latency field programmable gate array implementation of an unfolded architecture for the implementation of rotational CORDIC algorithm. Their computational device was highly suitable for the implementation of customized hardware in portable devices where large parallelism and low clock rate were utilized to meet low power consumption requirement.

Ayan Banerjee *et al.* in [11] presented a pipelined architecture using CORDIC for realization of transform domain equalizer. Transform domain equalizer had much faster convergence than its time domain counterpart for practical hardware realization having nonzero adaptation delay. Running DFT was employed as the transform, and CORDIC was used for realization of running DFT. Pipelining was applied throughout the architecture, which limited the critical path delay to the propagation delay of a single 16 bit adder for 16 bit arithmetic. For N tap equalizer, primary clock speed was N times of the sample clock speed, so that on arrival of each sample, the computation of whole transform and weight update was possible. Their architecture, hardware complexity was reduced by fully utilizing the pipeline without using parallel structures. The adaptation delay was only 2 sample clock periods resulting in fast convergence. Their architecture was suitable for VLSI implementation with primary clock speed limited by the binary adder propagation delay which could be as low as 2 ns in the present state-of-the-art technology.

3. PROBLEM DEFINITION AND SOLUTION

The primary intention of my research is to design an efficient implementation of rotational radix-4 CORDIC based FFT processor. In the Literature, a lot of works have been presented on VLSI architecture for CORDIC-based implementation. In recent years, the CORDIC algorithm based architectures for FFT have received a significant interest. Accordingly, the works presented in [3,5,7,15,16,17,18], describe the Multiplier less FFT architectures using CORDIC algorithm. The architecture in [3] is reduced memory and low power design which utilizes an addressing scheme and the associated angle generator logic in order to remove any ROM usage for storing twiddle factors. My work is based on the idea for efficient implementation of rotational radix-4 CORDIC algorithm, eliminating z-path completely in [3].The Radix-4 takes more time to select from among the five rotation direction values, and to select an appropriate angle out of five elementary angles [1]. To overcome this, I have decided to implement an CORDIC architecture which;

1) Pre compute all the direction of rotations for the input angle using the linear relation between the rotation angle and the constructed binary representation of directions of micro rotations.
2) Thus by eliminating the z-path in the hardware implementation of micro rotation.
3) Angle generator logic is introduced to reduce memory requirements to store twiddle factor values.

The advantages of this implementation will be:

1) Pre computation of direction of rotations for the input angles reduces the iteration delay, and power
2) Eliminates the z-path which improves the hardware complexity but with additional hardware required to pre compute the direction of rotation.

4. RADIX-4-CORDIC and FFT ALGORITHM

Generally, N point Discrete Fourier Transform is given by

$$X(k) = \sum_{i=0}^{N-1} x(i) * W_N^{ik} \qquad (1)$$

Where $k = 0,1,\ldots,N-1$ and $W_N^{ik} = e^{-j2\pi ik/N}$ is the “twiddle factor”. There will be $\log_r N$ stages and each stage contains N/r butterfly operations, in an N-point FFT. Because the CORDIC-based butterfly can be fast, the FFT-butterfly operation can be realized by radix-r CORDIC algorithm. The foundation for CORDIC algorithm is a 2D vector rotation in the xy-plane (see Figure.1). The vector rotation can be realized by rotating a vector through a series of elementary angles. These elementary angles are chosen such that the vector rotation through each of them may be approximated easily with a simple shift and add operation, and their algebraic sum approaches the required rotation angle [21]. The CORDIC algorithm can be exercised in two different modes, viz. Rotation mode and Vectoring mode.

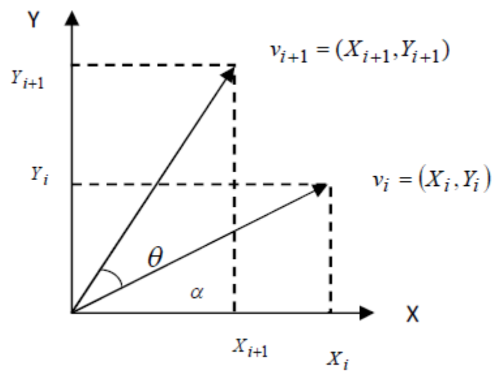


Figure.1. A 2D- Vector Rotation Using Cordic Algorithm.

Journal of Theoretical and Applied Information Technology
20th December 2013. Vol. 58 No.2
© 2005 - 2013 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

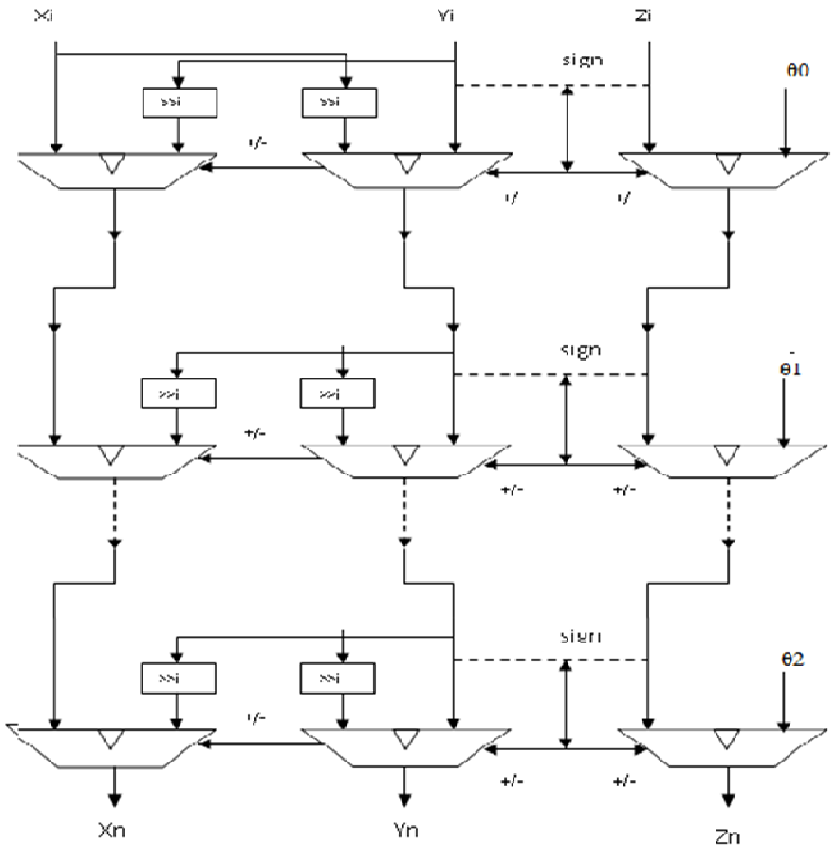


Figure.2. Basic Representation of Pipelined Cordic Processor.

1- The **rotation mode** is committed to practice general rotation for the given angle and to compute elementary operations such as multiplication, exponential, trigonometric functions and hyperbolic functions counting on the coordinate system.
2- The **vectoring mode** is utilized to define the angular argument of the original vector and estimate the logarithmic and division functions.

The standard CORDIC system in circular coordinate system practices 'n' iterations for n-bit precision using radix-r number representation. For a given n-bit precision, the increase of radix-r cuts down the number of microrotations. For example, the radix-4 CORDIC system achieves latency reduction which is an extension of radix-2 CORDIC algorithm. Using radix-4 we achieve half the number of microrotations than that required in radix-2. The iteration equations of the radix-4 CORDIC algorithm at the $(i+1)^{th}$ step [1] under circular coordinate system acting in rotation mode are as follows:

$$X_{i+1} = X_i - \sigma_i * Y_i * 4^{-i} \tag{2a}$$

$$Y_{i+1} = \sigma_i * X_i * 4^{-i} + Y_i \tag{2b}$$

$$Z_{i+1} = Z_i - \tan^{-1}(\sigma_i * 4^{-i}) \tag{2c}$$

Where, the direction of rotation in each iteration is $\sigma_i \epsilon \{-2, -1, 0, 1, 2\}$, and an elementary angle $\tan^{-1}(\sigma_i * 4^{-i})$ .The realization of Z_{i+1} in Eq. (2) increases the length of the vector.

To uphold the expected value of the vector, $(i+1)^{th}$ coordinates must be multiplied by the scale factor. The scale factor depends on the values of σ_i and must be estimated for each rotation angle.

$$K^{-1} = \prod_{i \ge 0} k_i^{-1} = \prod_{i \ge 0} \sqrt{\left(1 + \sigma_i^2 * 4^{-2i}\right)} \tag{3}$$

5. PROPOSED METHODOLOGY

A pipelined FFT architecture using radix-4 CORDIC with prediction block is proposed to reduce iteration delay, eliminate the z-path and also to cut down the memory required to store the twiddle factor angle. The number of iterations is reduced by using radix-4 number system and iteration delay is reduced by radix-4 signed digit arithmetic and pre computation of direction of micro rotation. The radix-4 takes more time to select from among the five rotation direction values, and to select an appropriate angle out of five elementary angles, which increases the iteration delay compared to the radix-2. Nevertheless, the proposed CORDIC architecture in rotation mode is designed to overcome this issue by pre compiling all the direction of rotations for the generated input angle utilizing the linear relation between the rotation angle and the framed binary representation of directions of micro rotations [14]. The proposed architecture comprises of four basic building blocks viz., the angle generator, the *x/y* path, the σ-prediction block, and the scale factor computation block as shown in Figure 3. The angle generator block produces the sequence of angles in regular and increasing order with the help of an accumulator and control logic. The σ-prediction block predicts all the σ_i values in radix-4 interpretation for the generated target angle θ, prior to the true CORDIC rotation starts iteratively in the *x/y* path. During every clock cycle, the σ_i values are buffered and sent sequentially to the corresponding stages of *x/y* path.

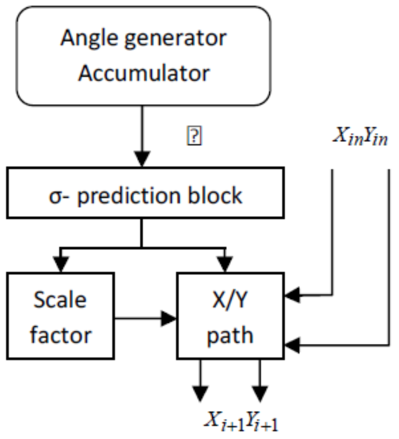


Figure.3. *Block Schematic Of The Proposed System.*

Proposed system:

1) θ - (Angle) generator block:-

The key operation of FFT is as shown in Eq. (1), $X(k)=\sum_{i=0}^{N-1} x(i) * W_N^{ik}$ .This is equivalent to Rotate $x(i)$ by an angle $\left(-j\frac{2\pi}{N}ik\right)$, which can be realized very well by the CORDIC algorithm. With simple shift and add operations, CORDIC-based butterfly can be fast. As we know that, an FFT processor needs to store the twiddle factors in memory. But, the CORDIC-based FFT doesn't have twiddle factors but needs a memory bank to store the rotation angles.

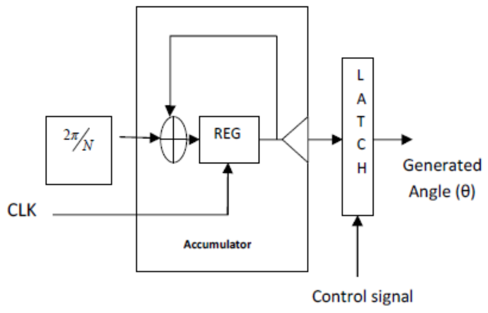


Figure.4. θ *- Generator Of The Proposed System*

Using a addressing scheme presented in [3], the twiddle factor angles generated by a simple accumulator, follow a regular and increasing order. For example, in 65536-point radix-4 FFT, it can be seen that twiddle factor angles are sequentially increasing, and every angle is a multiple of the basic angle $\left(\frac{2\pi}{N}\right)$, which is $\left(\frac{\pi}{32768}\right)$.For $(\log_4 65536)=8$ different FFT stages, the angles increase always one step per clock cycle. This function can be realized by the angle generator circuit which is composed of an accumulator, and an output latch, as shown in Figure.4. The accumulator output depends on the Control signals those enable or disable the latch which is simple to realize, as it is based on the present FFT butterfly stage and RAM address bits $b_{n-1}\ldots b_2b_1b_0$ [3]. Usually, for an N-point FFT, an $(n-1)=7$ -bit butterfly counter-B $(b_{n-2}b_{n-3}\ldots b_1b_0)$ will satisfy the address sequences and the control logic of the angle generator. At each stage 's', the RAM address bits are rotated right by s-bits of butterfly counter B $(b_{s-1}b_{s-2}\ldots b_1b_0b_{n-2}b_{n-3}\ldots b_s)$ and the control logic of the latch is driven by the sequence of the pattern; $(b_{n-2}b_{n-3}\dots b_s 0\dots 0)$ (s "0").

2) X/Y Path of the proposed Architecture:

The basic structure of proposed design for any radix butterfly operation of FFT processor is shown in Figure. 4. The proposed architecture requires $\left(\frac{n}{2}\right)=8$ micro rotation stages to implement x/y coordinates. The final x/y coordinates (see Figure.4) are scaled using the scale factor computed in parallel with the micro rotations, which requires adder/sub tractor, 2:1-multiplexer and an AND gate (see Figure. 7). Then the RBC converts the signed digit representation of the Radix-4 into binary form. Each micro rotation stage is designed by realizing the iteration equations,

$$X_{i+1} = X_i - \sigma_i * Y_i * 4^{-i}$$

$$Y_{i+1} = \sigma_i * X_i * 4^{-i} + Y_i$$

The Sign Digit adder/sub tractor is controlled by the sign of σ_i value (see Figure.5). This σ_i values are realized by the prediction block (see Figure.5).

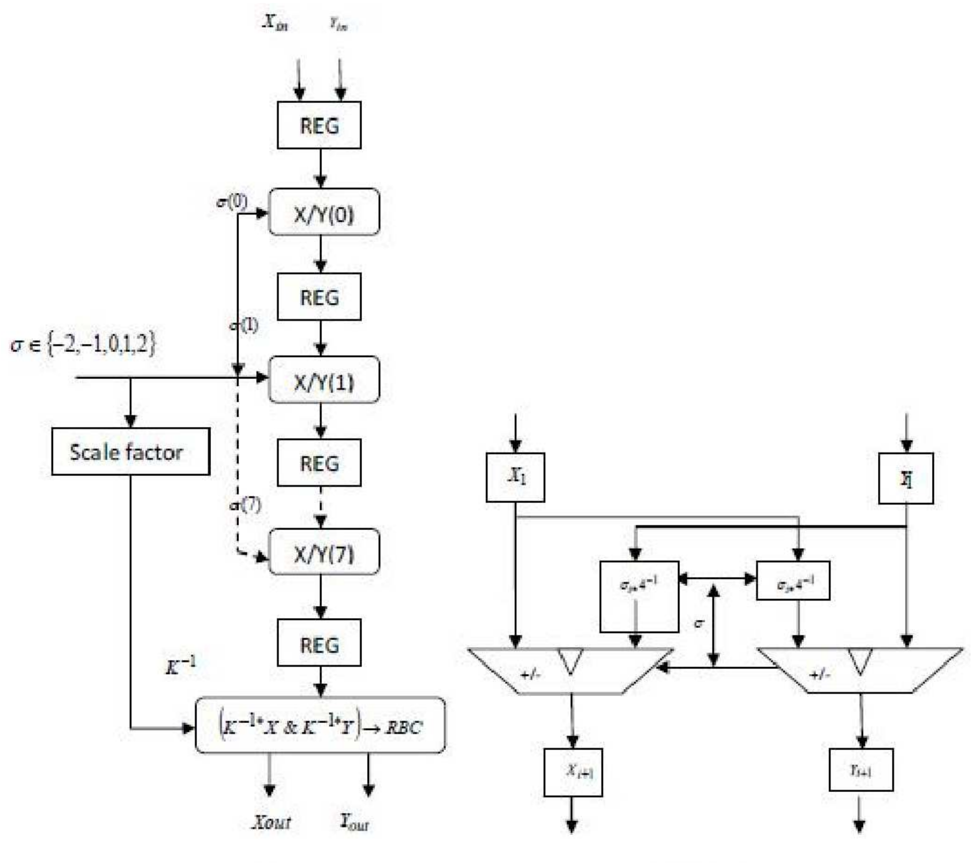


Fig.5a Fig.5b

Figure.5. a) X/Y path of the system, b) realization of X/Y path.

X/Y Hardware realization:

The x/y block of the proposed system for each stage is composed by combination of a pair of signed binary adders, pair of registers at input and output and a pair of right-shift registers(see Figure. 5b). For given angle θ, the input vector $v_i = (X_i, Y_i)$ is rotated to an output $v_{i+1} = (X_{i+1}, Y_{i+1})$ with σ_i , a predetermined direction vector, which assumes the direction of rotation.

Prediction block

The σ-prediction block requires pipelined implementation to compute the direction of micro rotations for the given (generated by the Angle generator) target angle. Realizing, a linear relation between the rotation angle and the binary representation of directions of micro rotations [1] given by

$$d = 0.5*\theta + 0.5*c_1 + \delta$$

$$c_1 = 2 - \sum_{1=0}^{\infty}\left(2^{-i} - \tan^{-1}(2^{-i})\right)$$

Where

$$\delta = \sum_{i=0}^{n/3}(d_i * \in_i)\, and \in_i = 2^{-i} - \tan^{-1}(2^{-i})$$

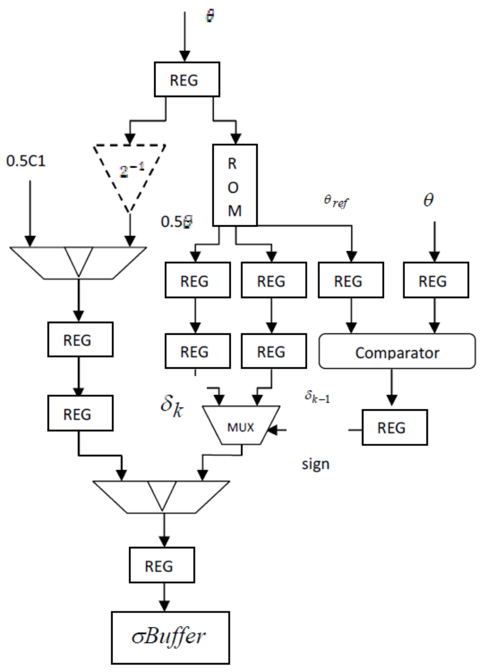


Figure.6 Realization Of Direction Of Rotation For An Angle Using $\sigma -$ *Prediction Unit.*

is utilized to predict the direction of rotations for a vector. The direction of micro rotations in binary form is represented by the variable 'd'. $(d - \theta)$ *and* θ has a linear relationship with negative slope being observed except for some θ value, called breakpoints [3]. There by computing'd' from θ using simple addition operation as shown in the Figure(6).The details of the various modules in the σ-pre computation block are as follows. For 16-bit precision, a $(32*48)$ ROM is selected to store these offset values [1]. The offset value δ is pre-computed for any input angle in the range $(-\pi/2, \pi/2)$ and the angle representing each of these δ is referred to as θ_{ref}, which coincides to the summation of the first $n/3$ micro rotations. Therefore, the target angle has to be greater than θ_{ref}. Each location of ROM holds a reference angle and two offset values (δ_k, δ_{k-1}). δ_k corresponds to the reference angle and (δ_{k-1}) corresponds to the previous reference angle. For the given input angle, the offset value is obtained from the ROM using $2:1$-multiplexer. Output of this multiplexer is controlled by the comparator, which generates sign signal '0' if $(\theta > \theta_{ref}\ forselecting\ \delta_k)$ else '1' if $(\theta < \theta_{ref}\ forselecting\ \delta_{k-1})$. The radix-4 arithmetic representation of $0.5c_1$ is added to 2's complement representation of 0.5θ and δ_{rom} using two adders to obtain $\sigma \in \{-2,-1,0,1,2\}$. The delay of each adder is $2t$ FA, where t FA is one full adder delay. These σ_i values are stored in a buffer to transfer them to the corresponding stages of x/y path. Thus, the direction of rotations is predicted before the x/y path initiates the computation of x/y coordinates and following into elimination of the z path.

Scale factor

The scale factor of the radix-2 CORDIC rotator is a constant but in case of radix-4 CORDIC rotator it isn't constant, this is observable from the equation below

$$K^{-1} = \prod_{i=0}^{n/4}\left(\frac{1}{\sqrt{1+\sigma_i^2 * 4^{-2i}}}\right)$$

where, $\sigma_i \in \{-2,-1,0,1,2\}$. Here, we compute K^{-1} only for first $\left(\frac{n}{4}+1\right) = 5$ iteration, as

$$K_i^{-1} = \frac{1}{\sqrt{1+4^{-2t}}}$$

becomes unity thereafter. For the present 16-bit implementation, it is adequate to approximate scale factor for the first 5- iterations only. Scale factor approximation is accomplished in parallel with the micro rotations by bottling all possible scale factors in memory [1]. From the expression above the σ_i^2 will have three values {0, 1, 4}.And so σ_i^2 can choose any one of the three values {0, 1, 4} in each iteration i, therefore for 16-bit precision actually it is required to store 3^5 possible scale factors in ROM. But, we can cut down the size of the ROM to $(27*16)$ using Taylor series expansion of the scale factor. For $i \geq \left[\frac{n}{8}+1\right]$ with 16-bit precision, the scale factors corresponding to the first 3-iterations are obtained from $(27*16)$ ROM. The 27 possible values for the scale factor K_2^{-1} corresponding to the first three iterations are computed as

$$K_i^{-1} = \frac{1}{\sqrt{1+\sigma_i^2 4^{-2t}}}$$

$$K_2^{-1} = \prod_{i=0}^{2} k_i^1$$

where, k_i^{-1} is the scale factor for the i^{th} iteration. Using these values, the scale factors for the remaining two iterations are evaluated using a combinational logic as shown in Figure.7. The scale factors for the k_3^{-1} and k_4^{-1} (remaining two iterations) are derived by executing the shift and subtraction operations over the scale factor retrieved from ROM. This is carried out by realizing the first two terms of their Taylor series expansions. And then, the scale factor is coded using the canonical signed digit representation to carry out the final multiplication over the x/y coordinates.[1].The proposed pipelined architecture for radix-4 CORDIC FFT algorithm is implemented using radix-4 signed digit arithmetic, eliminating z-path completely reducing the iteration delay. The hardware complexity is evaluated by determining number of additions and multiplications presented. Along with this memory reduction can be monitored.

Reduced memory:

For radix-4, $N=4^m$ -point, and $n=16$ -bit FFT, by using the θ generator, memory required [3] can be reduced to $\frac{(r-1)nN}{r}=786432$ out of $\frac{(3-1)nN}{r}=2883584$, 27.27%, whereas for radix-2 the memory can be reduced to 20% and conventional CORDIC require $\frac{5mN}{2}$ bits.

Iteration delay:

Iteration delay of a system increases, with the signed binary adder utilization and the hardware complexity. As the z-path is eliminated the iteration delay for that path is wiped out which is the effect of implementation of the prediction block. The prediction block with only few adders and multiplexers for the binary representation of the direction of micro rotations contributes to the reduction in iteration delay.

Therefore, the total delay of the proposed CORDIC system = 0.86941 ns.

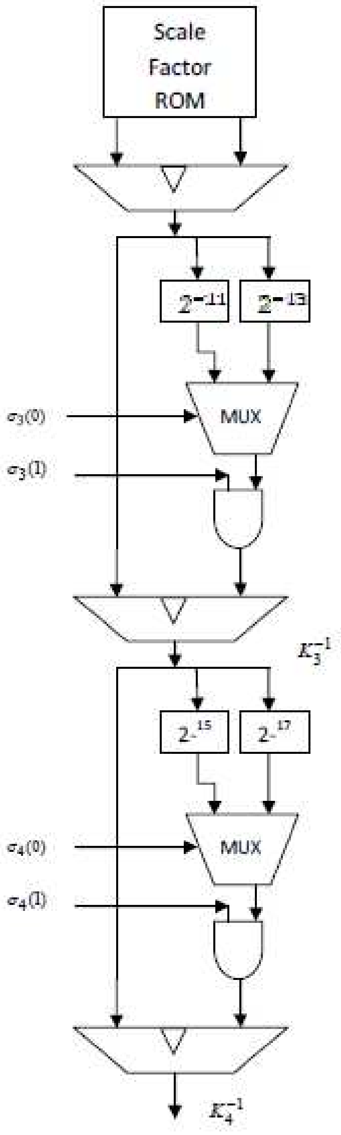


Figure.7. Scale Factor Computation Unit.

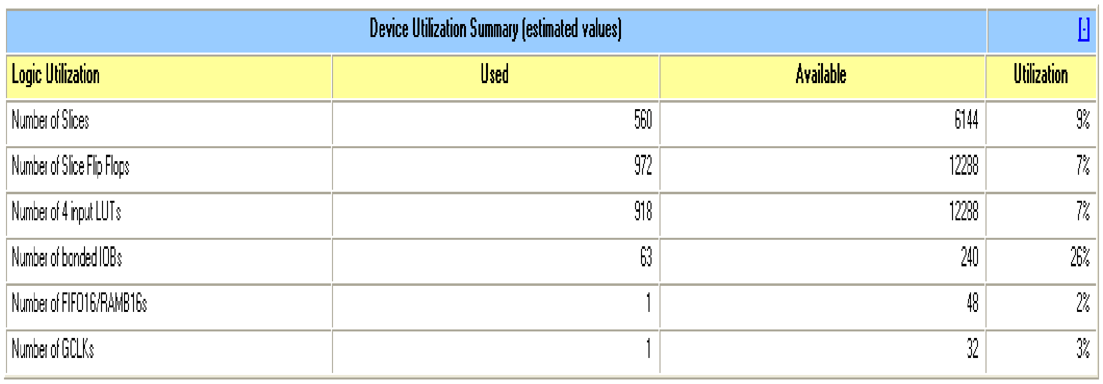
.

Hardware complexity:

Hardware complexity in case of conventional Radix-4 implementation is considered as the number of additions and multiplications in x/y path and z-path in terms of additions and multiplications , area which is $4*n^2=1024$ slices for x/y path and

$\frac{2.5 * n^2}{16} = 40$ slices for z-path. The hardware complexity of the proposed architectures in terms of total area/slices is 560slices for x/y path and zero for z-path, as it is eliminated. From Table (2), below it is evident that the complexity in the number of shift and add operations for radix-2 architecture is 2n, the complexity in number of stages of the proposed architecture is only n. This signifies hardware savings as hardwired shifts require more area. Comparing the hardware complexity of radix-2 architectures and the proposed architecture in terms of the z path, it is discovered that the proposed architecture wipes out the z path. Nevertheless, the proposed architecture demands additional hardware for the pre computation of direction of rotations and for the computation of variable scale factor as well. However, it may be noted that the overall hardware complexity of the proposed architecture is lower than all radix-2 architectures. This can be compared from the table(2).

Table.1. Logic Utilization Results Of The Proposed CORDIC X/Y-Path Architecture.



Device Utilization Summary (estimated values)			[-]
Logic Utilization	Used	Available	Utilization
Number of Slices	560	6144	9%
Number of Slice Flip Flops	972	12288	7%
Number of 4 input LUTs	918	12288	7%
Number of bonded IOBs	63	240	26%
Number of FIFO16/RAMB16s	1	48	2%
Number of GCLKs	1	32	3%

Table.2: Comparison Of The Proposed Architecture In Terms Of Area For Hardware Complexity.

Technique	**Radix-2[24]**	**Radix-4[23]**	**Proposed**
Additions(x/y)	2n=32	n=16	16
Multiplexers(x/y)	2n=32	n=16	16
Total Area(x/y)	$7n^2$=1792	$4n^2$=1024	5(7n)=560
Additions(z)	n=16	n/6	-
Multiplexers(z)	n=16	n/6	-
Total Area(z)	n^2=256	$2.5n^2/16$=40	-
Scale factor	Constant	variable	Variable
Scale factor Rom	-	$(3^{\left(\frac{n}{12}+1\right)} * n)$	$(3^{\left(\frac{n}{12}+1\right)} * n)$
Rotation H/W	$\lvert z \rvert$	Sign($\overline{w}$)	$(27 * 3n)$Rom

Table: 3. The Delay For The Proposed N-Point FFT Is Tabulated.

N-point FFT	Delay (ns)
256	0.290
1024	0.278
4096	0.290
65536	0.288

6. FFT WITH PROPSED METHODOLOGY

FFT with proposed methodology is shown in Figure. For radix 2 FFT it needs only four registers. But for radix -r FFT, it needs 2*r^2registers [3]. Using in –place computation technique, same registers can be used for both input & output in butterfly unit. Using a addressing scheme used in [3], the twiddle factor angles are generated by a simple accumulator, follows a regular increasing order. For example 65536 point radix-4 FFT, it can be seen that twiddle factor angles are sequentially increasing, every angle is a multiple of the basic angle 2Л/N , which is Л/32678 . It needs 8 different FFT stages (log $_4$ 65536) . The angle generation is shown in Figure 3.It comprises accumulator and an output latch. The accumulator output depends upon the control signals those enable or disable the latch which is simple to realize, as it based on the present butterfly FFT stage and RAM address bits b_n

…..$b_2b_1b_0$ [6]. Usually for an N- point FFT, an (n-1) , 7 bit butterfly counter–B (b_{n-2} b_{n-3}……b_1b_0) will satisfy the address sequences and the control logic of the angle generator. At each stage 's' , the RAM address bits are rotated right by s-bits of butterfly counter B ($b_{s-1}b_{s-2}$…….$b_1b_0b_{n-2}b_{n-3}$…b_3) and the control logic of the latch is driven by the sequence of the pattern $b_{n-2}b_{n-3}$…….bs0 ….0(s "0") .Z- path elimination reduces the delay and also reduces the additions , subtractions and multiplications in the z-path , it leads to reduced power consumption. Because of it's memory less logic to store twiddle factor angles, along Z- path eliminated CORDIC leads to very much reduced dynamic power consumption. Therefore proposed FFT structure is suitable for signal processing, biomedical and neural networks applications. Dynamic Power consumption by the proposed FFT at various frequencies is sketched below in the Frequency Vs Power graph.

From the Figure.9 above it can be observed that, the power consumed by the FFT system increases with increase in frequency. And the comparison between the proposed and the existing FFT architecture in terms of the power consumed is tabulated. The table information indicates that the proposed technique consumes comparatively less power than those in [17] and [18] at 100MHz and 20MHz frequencies respectively. Due to pipeline architecture, delay for the various no. of FFT's are almost same. It is tabulated in table 3.

Table4: Comparison of the Power consumption between Proposed and Existing FFT designs.

Frequency @	**Proposed power**	**[17]Radix-2 power**	**[18] Radix-4 power**
100MHz	28.52mW	180mW	-
20MHz	5.70mW	-	24.71mW

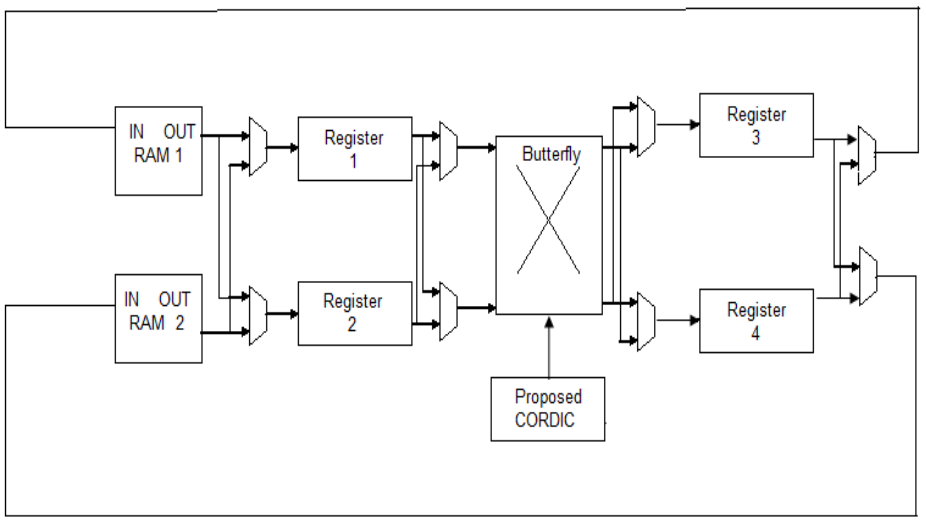
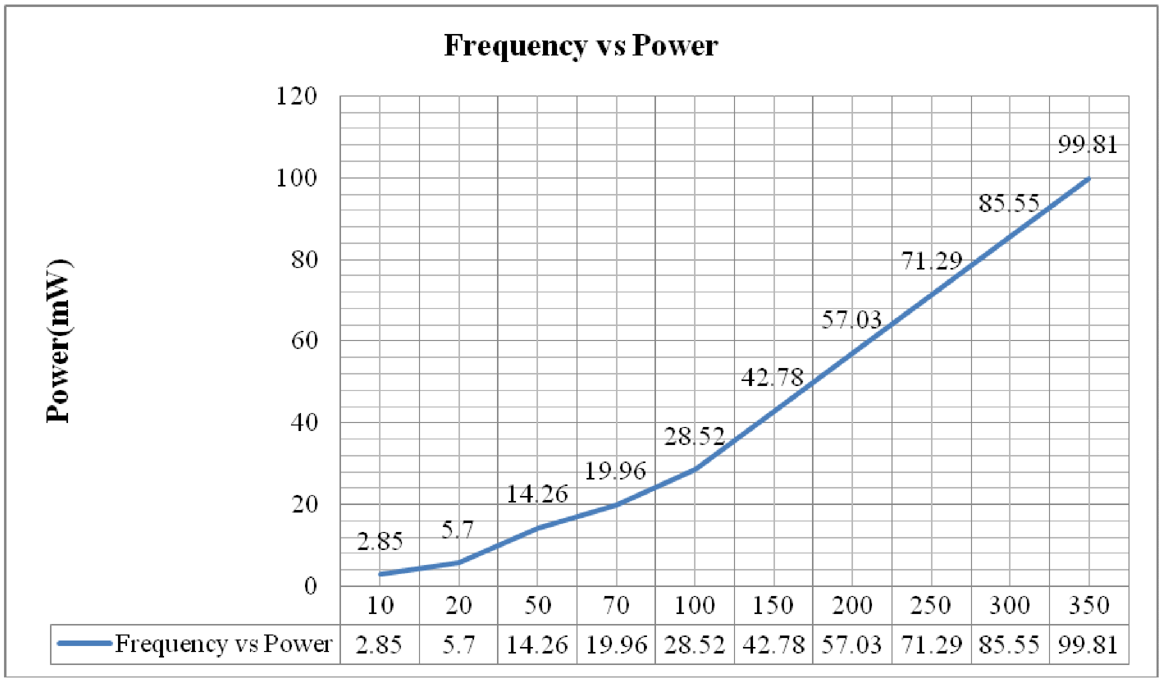


Figure. 8. RADIX-2 FFT with proposed CORDIC

7. RESULTS & CONCLUSION

The proposed design for 16 point radix-4 FFT algorithms have been realized using verilog and implemented on FPGA chip (virtex 4). It presents a pipelined architecture for the implementation of CORDIC algorithm exercising radix-4 number representation to halve the number of iterations. In addition to this, by pre compiling the direction of micro rotations, the iteration delay is reduced. Hence, the iteration delay is dependent on the x/y path. Moreover, the proposed architecture eliminates the z-path compared to conventional radix-4 architecture and reduces the memory usage by utilizing the angle generator block. The latency of the proposed architecture is n/2 clock cycles with through put rate of one valid result per eight clock cycles. The entire new proposed architecture operates at the clock rate of 450.564 MHZ. The proposed architecture shows hardware improvement along with reduced power consumption of 28.52mW at 100MHz and 5.70mW at 20MHz compared to radix-2 and other conventional radix-4 architectures respectively.



	10	20	50	70	100	150	200	250	300	350
Frequency vs Power	2.85	5.7	14.26	19.96	28.52	42.78	57.03	71.29	85.55	99.81

Figure. 9. Frequency Vs Power Graph For 16 Point FFT

REFERENCES:

[1] B. Lakshmi, A.S. Dhar, "VLSI architecture for low latency radix-4 CORDIC," *Computers & Electrical Engineering*,Volume 37, Issue 6, November 2011, Pages 1032–1042.

[2]Vachhani, L., Sridharan, K., Meher, P.K., "Efficient CORDIC Algorithms and Architectures for Low Area and High Throughput Implementation," *Circuits and Systems II: Express Briefs, IEEE Transactions on* Jan. 2009, Volume: 56 , Issue: 1 Page(s): 61-65.

[3]ErdalOruklu, Xin Xiao &JafarSaniie, "Reduced Memory and Low Power Architectures for CORDIC-based FFT Processors," *Journal of Signal Processing Systems archive*, Volume 66, Issue 2, February 2012,Pages 129-134.

[4]Francisco J. Jaime, Miguel A. Sánchez, Javier Hormigo, Julio Villalba, and Emilio L. Zapata, "Enhanced Scaling-Free CORDIC," *IEEE Transactions on circuits and systems —I: regular papers*, Vol. 57, No. 7, July 2010.

[5]Chao Cheng and Keshab K. Parhi, "High-Through put VLSI Architecture for FFT Computation," *IEEE Transactions on circuits and systems —II: Express Briefs*, Vol. 54, No. 10, October 2007.

[6]Koushik Maharatna, Swapna Banerjee, Eckhard Grass, Milos Krstic, and Alfonso Troya, "Modified Virtually Scaling-Free Adaptive CORDIC Rotator Algorithm and Architecture," *IEEE Transactions on circuits and systems for video technology*, Vol. 15, No. 11, November 2005.

[7]M. Garrido and J. Grajal, "Efficient Memory less CORDIC for FFT computation," *Acoustics, Speech and Signal Processing, 2007.ICASSP 2007. IEEE International Conference on* April 2007, Vol. 2, Page(s): II-113- II-116.

[8]Amritakar Mandal, K. C. Tyagiand Brajesh Kumar Kaushik, "VLSI Architecture Design and Implementation for Application Specific CORDIC Processor," *ARTCOM '10 Proceedings of the 2010 International Conference on Advances in Recent Technologies in Communication and Computing*, Pages 191-193.

[9]Ray, K.C., Shukla, R. and Dhar, A.S., "CORDIC-based VLSI architecture for implementing Log Polar Transformation for real time applications," *Computing Communication and Networking Technologies (ICCCNT), 2010*

International Conference on 29-31 July 2010, Page(s): 1- 4.

[10]Lakshmi.B and Dhar. A.S, "FPGA implementation of a high speed VLSI architecture for CORDIC," *TENCON 2009 IEEE Region 10 Conference on* 23-26 Jan. 2009, Page(s): 1- 5.

[11] Ayan Banerjee, and AnindyaSundarDhar, "Pipelined VLSI Architecture using CORDIC for Transform Domain Equalizer," *Journal of Signal Processing Systems*, February 2012.

[12] Kaushik Bhattacharyya, Rakesh Biswas, Anindya SundarDhar and Swapna Banerjee, "Architectural design and FPGA implementation of radix-4 CORDIC processor," *Journal Microprocessors & Microsystems archive*, Volume 34 Issue 2-4, March, 2010, Pages 96-101.

[13]ElisardoAntelo, Julio Villalba, Javier D. Bruguera, and Emilio L. Zapata, "High Performance Rotation Architectures Based on the Radix-4 CORDIC Algorithm," *IEEE Transactions on computers*, Vol. 46, No. 8, August 1997.

[14] Martin Kuhlmann, Keshab K. Parhi, "P-CORDIC: A Pre computation Based Rotation CORDIC Algorithm," *EURASIP Journal on Applied Signal Processing*, vol.9, pp: 936-943, 2002.

[15] Guihua Liu and Quanyuan Feng, "ASIC Design of Low-power Reconfigurable FFT Processor," *7th international conference on ASCI*, pp: 44-47, IEEE conf-2007.

[16] ZhaoYutian, Erdogan Ahmet. T, Arslan Tughrul. S, "A novel low-power reconfigurable FFT processor," *IEEE International Symposium on Circuits and Systems*, vol.1, pp: 41-44, 2005.

[17]Hasan Mohammad Mahmudul, Arslan Tughrul S and Thompson John S, "A delay spread based low power reconfigurable FFT processor architecture for wireless receiver," *International Symposium on System-on-Chip*, pp: 135 – 138, 2003.

[18] Mohd ShamianZainal, Shingo Yoshizawa, and Yoshikazu Miyanaga, "Low Power FFT Design for Wireless Communication Systems," *International Symposium on Intelligent Signal Processing and Communication Systems*, IEEE conf-2008.

[19] Meng Qian, Applications of CORDIC Algorithm to Neural Networks in VLSI Design ,*IMACS 2006*,Beijing, China,pp:504- 508

[20]Ayan Benerjee, Swapna Banerjee, 'FPGA realization of a CORDIC based FFT processor for biomedical signal processing, Microprocessors and Microsystems, Feb 2011.pp 131-142

[21] Volder, J. (1959). The CORDIC trigonometric computing technique, *IEEE Transactions on Electronic Computers*EC-8.pp 130-134

[22] Walther JS. A unified algorithm for elementary functions. *In: Proceedings of AFIPS spring joint computer conference*. Atlantic City, NJ; 1971. p. 379–85.

[23]Antelo E, Villalba J, Bruguera JD, Zapata EL. High performance rotation architectures based on the radix-4 CORDIC algorithm. *IEEE Trans Comput* 1997; 46(8):855–70.

[24]Dawid H, Meyr H. The differential CORDIC algorithm: constant scale factor redundant implementation without correcting iterations. *IEEE Trans Comput* 1996; 45(3):307–18.