International Journal of Applied Engineering Research
ISSN 0973-4562 Volume 10, Number 10 (2015) pp. 25851-25857

http://www.ripublication.com

Novel Approach of FFT Using CORDIC Algorithm and Analyzed With Chipscope Pro

K G V Nageswara Rao[1], B Murali Krishna[2], G L Madhumati[3]
[1]M.Tech Student, KL University, Vaddeswaram, Guntur, A.P.
[2]Asst.Professor, Department of ECE, KL University, Vaddeswaram, Guntur, A.P.
[3]Professor & Head of the ECE Department, DIET, Vijayawada,A.P.
E-mail: [1]kgvnrao.knr@gmail.com, [2]muralikrishna@kluniversity.in

Abstract

In this paper, high performance Fast Fourier Transform (FFT) processor is designed based on CORDIC algorithm and implemented on FPGA. CORDIC can increase the performance of FFT processor by replacing trigonometric functions using vector rotations. CORDIC algorithm uses shift-add procedure to calculate the trigonometric functions with this procedure it eliminates the memories for storing twiddle factor values. Based on DIF procedure FFT processor is designed by using verilog hard ware description language and it is dumped into spartan3E FPGA by using chipscope.

Keywords: Fast Fourier Transform, CORDIC Algorithm, FPGA.

Introduction

Fast Fourier Transform (FFT) is widely used in digital applications especially in communication systems. An FFT plays an important role in the communication systems, which occupies most of the hardware complexity. Due to huge demand in higher data rates in digital communications requires large-point FFTs such as 1024 ,2048,4096 etc. An FFT is designed with complex multiplier which is equal to two real adders and four real multipliers and ROM is for storing the twiddle factors. The Read Only Memory occupies most of the chip area, also consumes more power and reduces the speed because ROM size increases with large-point FFT's. Hence reduces the performance of FFT in terms of hardware consumption, power consumption, and performance.

The Co-ordinate Rotation Digital Computer [2] is a trigonometric algorithm, which makes simple to computes trigonometric functions by using vector rotations. This algorithm is used to compute trigonometric functions such as arctangent, cosine, sine, sinh, cosh, tanh, ln and exp. CORDIC algorithm is used in many applications because it uses only primitive operations such as shifts and additions to implement more complicated logics. CORDIC is a method which is alternate to replace the butterfly operation without using any conventional multiplier. CORDIC architecture is very robust and hardware efficient since it requires only add and shift operations, which is suitable for the twiddle factor multiplications in butterfly operations. Instead of storing the actual twiddle factor values in ROM, the CORDIC based FFT processor [3] requires only the twiddle factor angles to be stored in ROM. Additionally the CORDIC based butterfly architecture can be faster than the conventional multiplier based butterfly architecture in VLSI implementations [4]. This paper is organized as follows: Section-I gives the introduction of CORDIC algorithm and FFT. Section-II describes about the 16-point radix-2 FFT algorithm [5]. and Section-III describes about CORDIC algorithm. Section-IV gives the information about the results of FFT using Cordic algorithm[6].

16-Point Radix- 2 FFT

The Cooley-Tukey algorithm [7] is a most widely used FFT algorithm which was published in the year 1965. The idea of the FFT algorithm is to divide the N-point DFT into N/M point DFTs, if M=2 then it is split into two N/2 point DFT's [10]. This is known as the radix-2 point DFT. Similarly, we have radix 4, 8, 16 & …etc. The Cooley and Tukey presented the efficiency and simplicity of the DFT computation by divide and conquer approach. With this approach FFT computation was widely accepted. The idea of the Cooley–Tukey algorithm is to divide the larger FFT into smaller FFT's.

The N-point Fast Fourier transform (FFT) X (k) is defined as:

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{nk} \quad 0 \leq k \leq N-1 \text{----} \qquad (1)$$

where the twiddle factor W_N^{nk} is given by,

$W_N^{nk} = e^{-j2\pi nk/N}$

x(n) is the input sample and X(k) is output sample.

The odd and the even samples of the FFT are calculated using the DIF radix-2 FFT Algorithm [9].The Even samples of the FFT X[K] are given by the equation,

$$X(2r) = \sum_{n=0}^{N/2-1} \left(x(n) + x\left(n + \frac{N}{2}\right)\right) \text{-----} \qquad (2)$$

The odd samples of Fast Fourier Transform represented as X[k] and it is shown in equation below,

$$X(2r+1) = \sum_{n=N/2}^{N-1} \left(x(n) - x\left(n + \frac{N}{2}\right)\right) WNn \text{----} \qquad (3)$$

where r = 0,1,2,3……….(N/2 – 1)

The below figure describes the 16 point FFT diagram implemented using DIF method

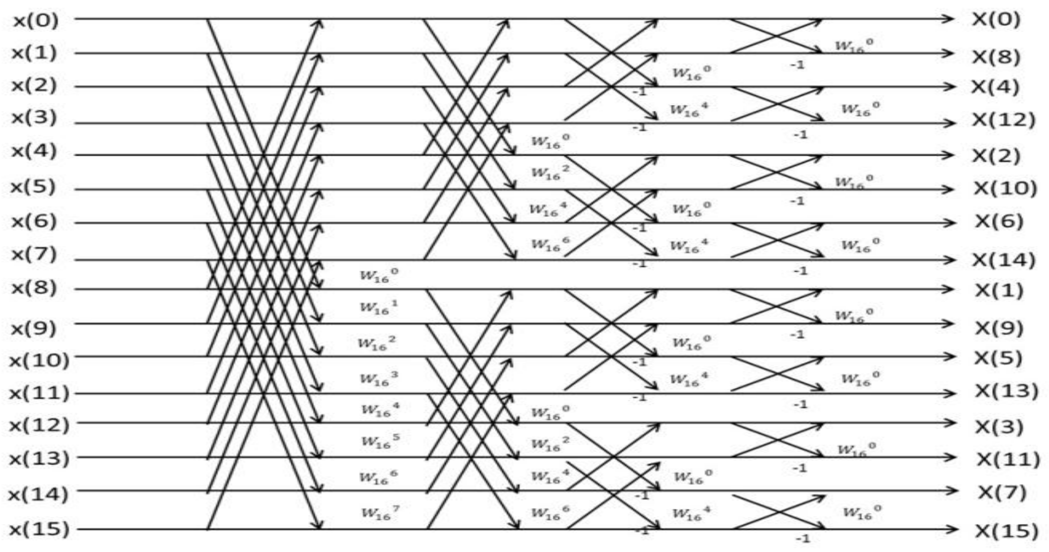


Figure 1: 16 point FFT Butterfly Architecture

CORDIC Algorithm

The CO-ordinate Rotation DIgital Computer (CORDIC) [8] is a efficient and high performance "shift-add" algorithm to compute the wide range of functions such as trigonometric equations, logarithmic, hyperbolic equations and linear. It is commonly used for the replacing the hardware multiplier. Vector rotations are used for converting the polar to rectangular and rectangular to polar conversion. It was first introduced by Jack Volder in 1949 based on the observation, that if we rotate a vector (1, 0) by an angle φ then its new vectors will be at { *sin* φ, cos φ} .

The vector rotation is defined as:

$x' = x.\cos\phi - y.\sin\phi$

$y' = y.\cos\phi + x.\sin\phi$

On rearrangement of the terms we get:

$x' = \cos\phi\ [\ x - y.\ \tan\phi\ ]$

$y' = \cos\phi\ [\ y + x.\ \tan\phi\ ]$

Still the equations are complex due to the presence of trigonometric functions. The rotation angles have been restricted to values such as $\tan\phi = \pm 2^{-i}$, the multiplication of the tangent have been simplified by using simple shift operations. Hence, the arbitrary angles can be computed by performing a series of rotations. On every rotation, the direction of rotations is done by obtaining the difference between the actual angle and the rotated angle. Mathematically the equation is given as shown below.

$X_{i+1} = K_i[x_i - y_i\ .\ d_i\ .\ 2^{-i}]$

$Y_{i+1} = K_i[y_i + x_i\ .\ d_i\ .\ 2^{-i}]$

Where $K_i = \cos(\tan^{-1} 2^{-i}) = 1/\sqrt{(1+2^{-2i})}$

$d_i = \pm 1$

The value of di is +1 if the obtained angle is greater than the current angle and is -1 if the current angle is greater. The value of Ki can be taken as 0.6073 when the iterations count is large.The CORDIC rotation and vectoring[8] are limited to the rotation angles between $-\pi/2$ and $\pi/2$. In the first iteration the tangent is limited to 2^0. For angles larger than $\pi/2$, requires an additional rotation.

$x' = -d$; $y' = d.\ x$; $z' = z + d\ .\ \pi/2$ where $d = +1$ if $y<0$ or $d = -1$, otherwise

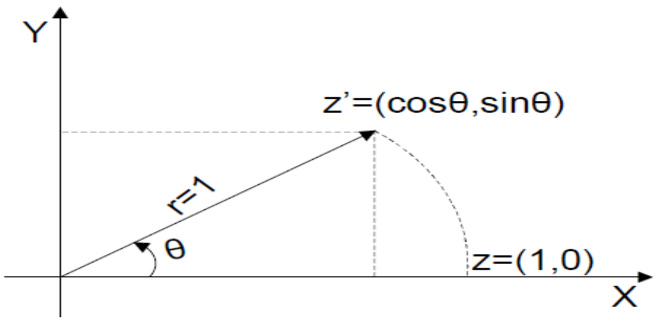


Figure 2: Vector Rotation

The CORDIC is operated in two modes. Rotating the input vector by an angle is called rotation mode. And rotating the input vector according to the x-axis meanwhile recording the angle which is required to make the rotation is called vectoring mode.

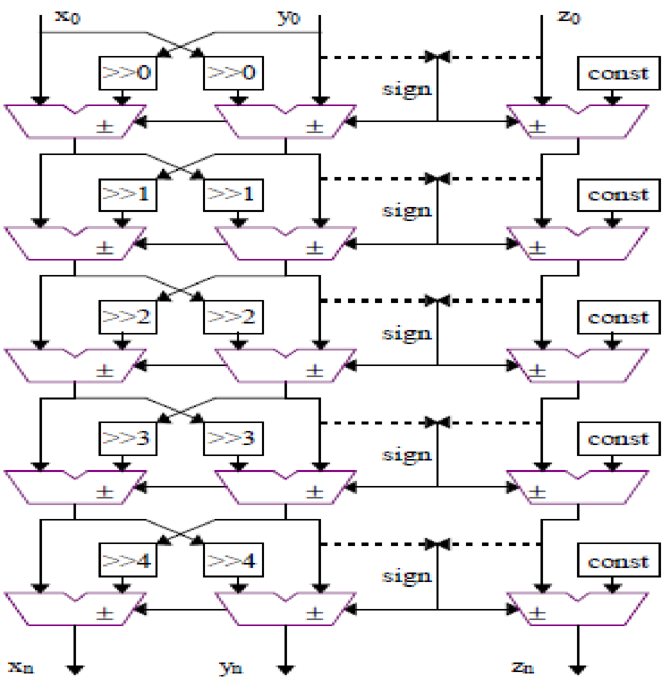


Figure 3: CORDIC Pipeline Unit

Stage wise twiddle factor angles which are required for 16-point FFT butterfly architecture [9] is presented in table 1.

Table 1: Rotational Angles required for four stages of 16 point FFT are given below

Twiddle factor angles			
Stage 0	Stage 1	Stage 2	Stage 3
0	0	0	0
$\pi/8$	0	0	0
$2\pi/8$	$2\pi/8$	0	0
$3\pi/8$	$2\pi/8$	0	0
$4\pi/8$	$4\pi/8$	$4\pi/8$	0
$5\pi/8$	$4\pi/8$	$4\pi/8$	0
$6\pi/8$	$6\pi/8$	$4\pi/8$	0
$7\pi/8$	$6\pi/8$	$4\pi/8$	0

Results

Hardware results obtained from Chip-scope pro analyzer for 4 point FFT is as shown below:

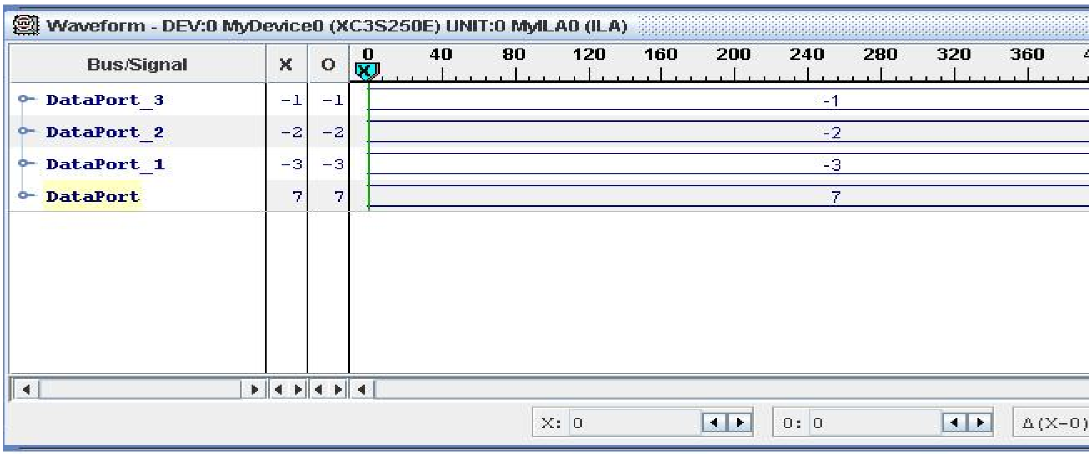
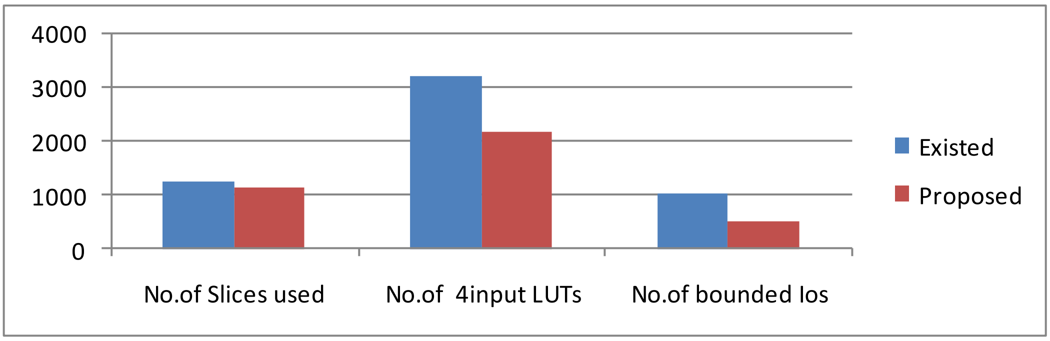


Figure 4: Chipscope Result for 4 point FFT

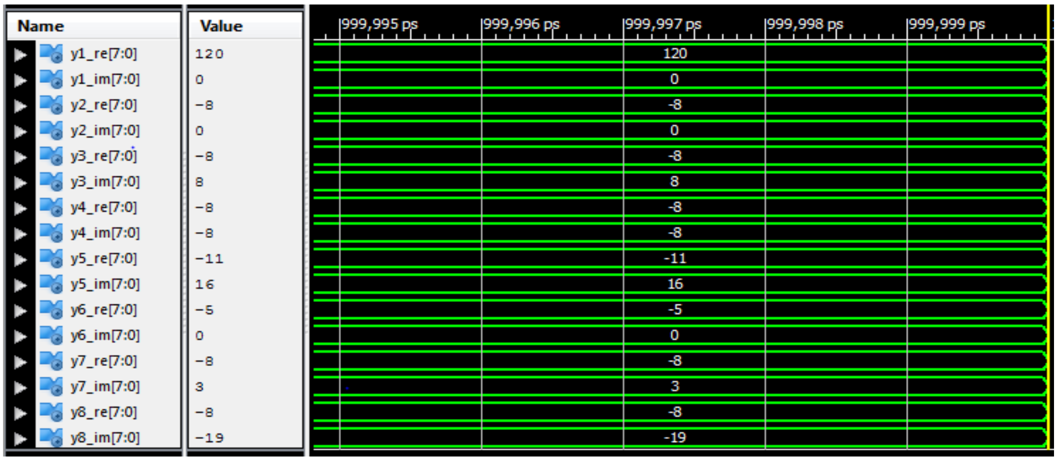
Table 2: Comparison Results of FFT with Cordic Algorithm and Conventional FFT

Parameter	Conventional FFT	FFT using CORDIC
Delay (ns)	33.02	14.22

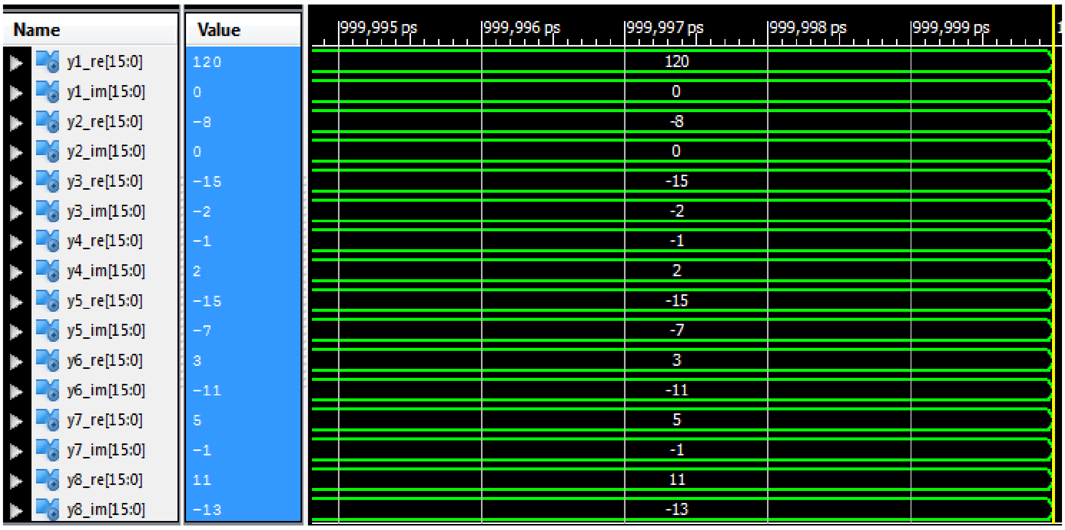
Chart of Comparison for Existing FFT with Proposed FFT



Simulation results of conventional 16-bit FFT



Simulation results of 16-bit FFT using Cordic algorithm



Conclusion

This paper describes the implementation of FFT processor based on CORDIC algorithm. Project was implemented by using Verilog HDL and also observed simulation results for both conventional FFT and CORDIC based FFT. It is proved that implementation of FFT processor using CORDIC algorithm as improved the performance which is shown in comparison table. The CORDIC based FFT algorithm is dumped into FPGA, spartan3E by using Chipscope analyzer and the results is shown in figure4. The FFT can also be implemented higher Radix Cordic algorithms and unfolded cordic algorithm.

References

[1] "A Unified View of CORDIC Processor Design", Application Specific Processors, Edited by Earl E.Swartzlander, Jr., Ch. 5, pp. 121-160, Kluwer Academic Press, November 1996.

[2] A Hybrid-Radix Approach for Efficient Implementation of Unfolded CORDIC Architectures for FPGA Platforms, 2014 International Conference on Signal Processing and Integrated Networks (SPIN)

[3] "An Angle Recoding Method for CORDIC Algorithm Implementation", IEEE Transactions on Computers, Vol. 42, pp. 99-102, January 1993

[4] "The CORDIC Algorithm: New Results for Fast VLSI Implementation," IEEE Transactions on Computers, Vol. 42, pp. 168-178, 1993

[5] "Verilog Implementation of 32 Point FFT Using Radix-2 Algorithm on FPGA Technology
" , (IOSR-JECE), .Volume 9, Issue 1

[6] "Efficient FPGA implementation of FFT/IFFT Processor",international journal of circuits, systems and signal processing, Issue 3, Volume 3, 2009

[7] James W. Cooley and John W. Tukey, An Algorithm for the Machine Calculation of Complex Fourier Series.

[8] "Pipelined CORDIC Architecture for Fast VLSI Filtering and Array Processing," Proc. ICASSP'84, 1984

[9] " Simple Computation of DIT FFT",ijarcsse, Volume 4, Issue 5, May 2014

[10] "Verilog Implementation of 32 Point FFT Using Radix-2 Algorithm on FPGA Technology" IOSR-JECE, Volume 9, Issue 1, Ver. II. e-ISSN:2278-2834.