Volume 1, Issue 2, April 2014 ISSN: 2348-8565 (Online)

International Journal of Modern Engineering and Research Technology

Website: http://www.ijmert.org Email: editor.ijmert@gmail.com

An efficient FFT Processor Architecture with Optimised Area and Speed

Amit Kumar Mishra
M. Tech. Scholar,
Shri Ram Institute of Technology
Jabalpur, (M.P) [India]
Email: amit_2440@yahoo.co.in

Divayanshu Rao
Assistant Professor
Shri Ram Institute of Technology
Jabalpur, (M.P) [India]

Prof. Ravi Mohan
Associate Professor,
Department of Electronics & Communication Engg,
Shri Ram Institute of Technology,
Jabalpur, (M.P.) [INDIA]
Email: ravimohan7677@yahoo.co.in

ABSTRACT

It is required to design a very high-performance FFT processor to satisfy the desires of real time with low cost in many different systems. So paper presents a radix-2 FFT processor based implemented Field Programmable Gate Array (FPGA) which can work for Wireless Local Area Networks (WLAN) is proposed. Different than being stored in the conventional ROM, the twiddle factors in proposed FSM based FFT processor can be given directly. A unique and easy address mapping scheme proposed. The FFT processor has five different FSM stages. Finally, the pipelined 8-point proposed FFT processor can be completely implemented within only 637 slices only. As known the multiplication is the major part in FFT operation In this paper it is also proposed a new tree multiplication structure based architecture to design this Vedic multiplier. For the addition of partially generated products a new addition tree structure has been proposed. It has better speed in compare with Wallace, Booth, Modified Booth Wallace, Array, Karatsuba and Vedic Karatsuba Multiplier.

Keywords:— *Inverse Discrete Fourier Transform (IDFT), FPGA (Field Programmable Gate Array), Finite State Machine (FSM), Fast Fourier Transform (FFT)*

I. INTRODUCTION

FFT is a compulsory requirement when we require computation in periodic digital signal and FFT is very good in image processing, SAR (synthetic aperture radar), Radar, & multimedia applications. Our thesis work will enhance the functioning of these kinds of systems. We are developing and implementing a new fast and area efficient FFT Processor, proposed work will be faster multiplier then ref [1] and also we are expecting less dealy and less area FFT processor as compare to ref [2] [3] [4].

Basic radix-2 DIF-FFT algorithm butterfly which decimation is better than DIT-FFT so DIF-FFT is been used to design the system, a vedic-multiplication method, and a unique address mapping scheme which reduces delay and increases the speed of the system. The FSM based FFT architecture is proposed and each unit is also illustrated in work. The implementation of the 8-point FFT processor based on FPGA, use total 12 butterfly and each need one complex multiplication so total 12comlex multiplication required as know with conventional method of complex number multiplication it requires 4 normal multiplication so total 12x4 which is total 48 multiplication if our signals are of 16 bit long and in FFT it would be floating also so it is requires to make a floating multiplier of 16 bit.

We have studied all the references above and consider them as our base research papers for our research work and design proposed our architecture.

II. FFT Algorithm Used

Radix-2 FFT Algorithm The FFT algorithm can compute the Discrete Fourier Transform (DFT) effectively. Given a sequence {x(n)} of N complex numbers, we can compute its DFT, another sequence {X(k)} of N complex numbers, according to the following formula [6]

$$X(K) = \sum_{n=0}^{N-1} x(n)\, W_N^{nk}, K=0,1,..N-1$$

And according to the different way to decimate, it can be divided into two types, DIF (Decimation in Frequency) and DIT (Decimation in Time). The DIF algorithm is easier to design than DIT. And considering the finite word length effect, DIF has much more advantages than DIT, like reducing the noise, which is introduced by the multiplication when it is implemented with the fixed point [7] and reducing the complexity of the whole system. Consequently, we use the DIF algorithm to design radix-2 FFT module and most of current FFT processors are also based on this algorithm [8].

III. Methodology

For radix-2, N point FFT, each stage required N/2 butterflies. Proposed FFT shown in figure 1, suggest that if the combination of N/2 butterflies is taken then the FFT operations can be performed in 'm' number of cycles. This approach required a Control Unit for properly arranging the output from the previous stage and feed that into the next stage. It is also known as feedback algorithm.

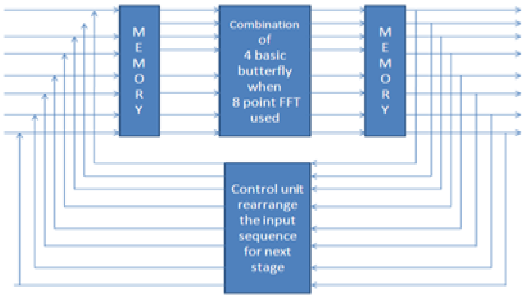

MEMORY
Combination of 4 basic butterfly when 8 point FFT used
MEMORY
Control unit rearrange the input sequence for next stage

Figure 1: *Proposed Fast Fourier Transform (FFT)*

Three-multiplication Method Consider two inputs Z1=(x1+iy1) and Z2=(x2 + iy2). On taking conventional multiplication of these inputs they gives the outputs as- Real Part(R) = (x1x2-y1y2) and Imaginary Part (I) = (x1y2+y1x2). On developing it requires four multipliers, one subtractor and one adder.

But if the two inputs are multiplied by the proposed approach the outputs are given as

Real Part(R) = x1(x2+y2)-y2(x1+y1) & Imaginary part(I) = x1(x2+y2)-x2(x1-y1)

Upon adding the above two terms (R and I) it gives the same value as simple multiplication. But implementation of R and I requires three multipliers and two adder and three subtractors (term x1(x1+x2) is counted once because it is repeating in real and imaginary part), so one multiplier is reduced on cost of one adder and two subtractor.

Proposed complex multiplication need one extra adder and two extra subtractors on the cost of one reduced multiplier.

A 16 bit adder need 16 Full adder and 16 bit subtractor need 16 Full adders with 16 XOR

gates. But one 16 bit multiplier needs 16x16=256 AND gate and 32x15=480 Full adder (for conventional multiplication) and this can be reducing maximum up-to 75% of conventional requirement even if advance multiplication techniques like Wallace, Vedic, booth etc. are used.

Therefore it can be concluded that still one adder and two subtractions is a better deal instead of using one 16 bit multiplier.

Suppose z1=3.25+3j and z2=7.5+1.17j are two inputs to be multiplied. The real and imaginary parts after their multiplication are found out as-

R=3.25(7.5+1.17)-1.17(3.25+3) => 3.25(8.67)-1.17(3.25) => 28.1775-3.8025=> 20.865

I=3.25(7.5+1.17)-7.5(3.25-3) => 3.25(8.67)-7.5(.25) => 28.1775-1,875=>26.3025

Let’s have the above example in binary form-

X1(3.25) =>000000000011.0100 and X2(7.5) => 000000000111.1000

Y1 (3) =>000000000011.0000 and Y2 (1.1875) => 000000000001.0011

X2+Y2 (8.6875) => 000000001000.1011

X1+Y1 (6.25) => 000000000110.0100

X1-Y1 (0.25) =>000000000000.0100

{X1*(X2+Y2)}(28.234375) => 00011100.00111100

{Y2*(X1+Y1)}(7.421875)=> 00000111.01101100

{X2*(X1-Y1)}(1.875)=> 00000001.11100000

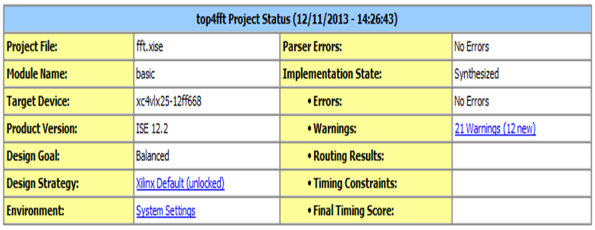
{X1*(X2+Y2) – Y2*(X1+Y1)}(20.8125) => 00010100.11010000

{X1*(X2+Y2) – X2*(X1-Y1)}(26.359375)=> 00011010.01011100

So the final Real part is R= 20.8125 and imaginary part is I= 26.359375.

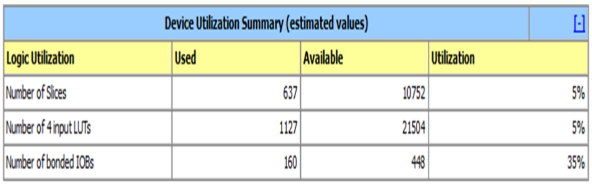
IV. RESULTS

Table 1: Top 4 FFT Project Status



top4fft Project Status (12/11/2013 - 14:26:43)			
Project File:	fft.xise	**Parser Errors:**	No Errors
Module Name:	basic	**Implementation State:**	Synthesized
Target Device:	xc4vlx25-12ff668	**• Errors:**	No Errors
Product Version:	ISE 12.2	**• Warnings:**	21 Warnings (12 new)
Design Goal:	Balanced	**• Routing Results:**	
Design Strategy:	Xilinx Default (unlocked)	**• Timing Constraints:**	
Environment:	System Settings	**• Final Timing Score:**	

Table 2: Device Utilization Summary



Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	637	10752	5%
Number of 4 input LUTs	1127	21504	5%
Number of bonded IOBs	160	448	35%

V. CONCLUSION

In this thesis report, different DIF-FFT architectures namely pipelined and area efficient architecture have been discussed and a new architecture which is based upon the feedback mechanism has been proposed. This proposed architecture is designed in such a way, so that it uses much lesser number of basic butterflies as compared to conventional FFT. Along with this, the total numbers of multipliers that are required for multiplication part in FFT are also reduced by a new complex multiplication technique.

Apart from these modifications, a new multiplier called Vedic multiplier is also introduced and is integrated with the proposed FFT. Earlier, the conventional multiplier that was used with FFT used to consume much time in producing the result but this Vedic multiplier performs the multiplication operation by the different method which is called Urdhva Triyagbhyam method and produces the result in a lesser time. Since

Vedic multiplier is known for producing the results it a much lesser time as compared to normal multiplier therefore integrating such multiplier with FFT helped to increase the speed to a new level

REFERENCES:

[1] "High Speed Energy Efficient ALU Design using Vedic Multiplication Techniques" M. Ramalatha, Senior Member, IEEE, K. Deena Dayalan, Member, IEEE, P. Dharani, Member, IEEE, S. Deborah Priya, Member, IEEE, ACTEA 2009, July 15-17, 2009 Zouk Mosbeh, Lebanon, 978-1-4244-3834-1/09/ © 2009 IEEE.

[2] "A Generalized Mixed-Radix Algorithm for Memory-Based FFT Processors", Chen-Fong Hsiao, Yuan Chen, Chen-Yi Lee, pp 1549-7747 © 2010 IEEE.

[3] J. A. C. Bingham, "Multicarrier modulation for data transmission: an idea whose time has come," IEEE Communication Magazine, vol. 28, no. 5, pp. 5-14, May 1990.

[4] J. Palicot and C. Roland, "FFT: A basic function for a reconfigurable receiver," 10th International conference on Telecommunications, vol. 1, pp. 898-902, March 2003.

[5] Min Jiang, Bing Yang, Yiling Fu, et al., "Design Of FFT processor with Low Power complex multiplier for OFDM-based high-speed wireless applications," International Symposium on Communications and Information Technology, vol. 2, pp. 639-641, Oct. 2004.

[6] Kai Zhong, Hui He, and Guangxi Zhu, "An ultra-high speed FFT processor," International Symposium on Signals, Circuits and Systems, vol. 1, pp. 37 -40, July 2003.

* * * * *