Volume 3, Issue 8, August 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Biquad Implementation of a FFT/IFFT an a FPGA Device

Jaspreet Kaur
ECE Department, PTU Jalandhar
India

Cherry
CSE Department, PTU Jalandhar
India

Dr.K.V.P. Singh
DIET KHARAR, PTU Jalandhar
India

Abstract: A FFT algorithm is characterized by a set of addition, subtraction and multiplication operations. In a block processing algorithm, when the data flow is continuous, data management becomes the most critical part in the hardware design. A control unit is necessary for clocking and synchronizing the different blocks of the FFT. When designing FFT architectures, there are a number of design specifications to be considered. This paper presents biquad structure or streaming for implementation of FFT/IFFT.

Key Words: Biquad, Discrete Fourier Transform, Fast Fourier Transform, FPGA, Fourier Transform.

I. Introduction

The analysis of real world signals is a fundamental problem for many engineers and scientists, especially for electrical engineers since almost every real world signal is changed into electrical signals by means of transducers, e.g., accelerometers in mechanical engineering, electrodes and blood pressure probes in biomedical engineering, seismic transducers in Earth Sciences, antennas in electromagnetics, and microphones in communication engineering, etc. Traditional way of observing and analyzing signals is to view them in time domain. Baron Jean Baptiste Fourier [1], more than a century ago, showed that any waveform that exists in the real world can be represented (i.e., generated) by adding up sine waves. Since then, we have been able to build (break down) our real world time signal in terms of (into) these sine waves. It is shown that the combination of sine waves is unique; any real world signal can be represented by only one combination of sine waves [2]. The Fourier transform (FT) has been widely used in circuit analysis and synthesis, from filter design to signal processing, image reconstruction, stochastic modeling to non-destructive measurements. The FT has also been widely used in electromagnetics from antenna theory to radio wave propagation modeling, radar cross-section prediction to multi-sensor system design. For example, the split-step parabolic equation method (which is nothing but the beam propagation method in optics) has been in use more than decades and is based on sequential FT operations between the spatial and wavenumber domains. Two and three dimensional propagation problems with non-flat realistic terrain profiles and inhomogeneous atmospheric variations above have been solved with this method successfully [3-5]. The principle of a transform in engineering is to find a different representation of a signal under investigation. The FT is the most important transform widely used in electrical and computer engineering.

The transformation from the time domain to the frequency domain (and back again) is based on the Fourier transform and its inverse, which are defined as

$$S(\omega) = \int_{-\infty}^{\infty} s(t) e^{-j2\pi f t} dt \qquad (1) \qquad (1.1a)$$

$$s(t) = \int_{-\infty}^{\infty} S(f) e^{j2\pi f t} df \qquad (1.1b)$$

Here, s(t), S(ω), and f are the time signal, the frequency signal and the frequency, respectively, and $j = \sqrt{-1}$. We, the physicists and engineers, sometimes prefer to write the transform in terms of angular frequency ω = 2π f, as

$$S(\omega) = \int_{-\infty}^{\infty} s(t) e^{-j\omega t} dt \qquad (2) \qquad (1.2a)$$

$$s(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} S(\omega) e^{j\omega t} d\omega \qquad (1.2b)$$

This, however, destroys the symmetry. To restore the symmetry of the transforms, the convention

$$S(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} s(t) e^{-j\omega t} dt \qquad (3) \qquad (1.3a)$$

© 2013, IJARCSSE All Rights Reserved

$$s(t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} S(\omega) e^{j\omega t} d\omega \quad (4)$$

(1.3b)

is sometimes used. The FT is valid for real or complex signals, and, in general, is a complex function of ω (or f).

A. *Discrete Fourier Transformation*

To compute the Fourier transform numerically on a computer, discretization plus numerical integration are required. This is an approximation of the true (i.e., mathematical), analytically-defined FT in a synthetic (digital) environment, and is called discrete Fourier transformation (DFT). There are three difficulties with the numerical computation of the FT:

- Discretization (introduces periodicity in both the time and the frequency domains)
- Numerical integration (introduces numerical error, approximation)
- Finite time duration (introduces maximum frequency and resolution limitations)

The DFT of a continuous time signal sampled over the period of T, with a sampling rate of Δt can be given as

$$S(m\Delta f) = \frac{T}{N} \sum_{n=0}^{N-1} s(n\Delta t) e^{-j2\pi m\Delta f\, n\Delta t} \quad (1.4)$$

(5)

where Δf=1/T, and, is valid at frequencies up to $f_{max} = 1/(2\Delta t)$.

II. FAST FOURIER TRANSFORM

The fundamental motivation for the FFT is the extremely high computational load of calculating the DFT. By exploiting the symmetry and periodicity properties of the twiddle factor, the number of required calculations is significantly reduced [1]. Many FFT algorithms exist today, but they are all derivatives of the work done by Cooley and Tukey in 1965 Their algorithm was so efficient in computation that it revolutionized digital signal processing . The highest efficiency is achieved when the sample length is a power of two [1].

It is important to note that the FFT is mathematically equivalent to the DFT, not an approximation. As a result, all of the properties and strengths of the DFT hold true for all FFT algorithms. Most of the weaknesses also apply to FFT algorithms. The FFT does reduce the computational requirements and the amount of quantization noise error due to the high computational load of the DFT.

The high computational load of the DFT is due to the N^2 terms in these equations. The FFT algorithms significantly reduce the required computations. All FFT algorithms have a computational load of

Number of Adds = C_1*N*LOG_2 (N)

Number of Multiplies = C_2*N*LOG_2 (N)

Where C_1 and C_2 are constants.

The radix-2 Cooley-Tukey algorithm requires approximately 3*N*LOG_2 (N) adds and 2*N*LOG_2 (N) multiplies. Figure1 compares the computational load of this FFT algorithm with that of computing the DFT directly.

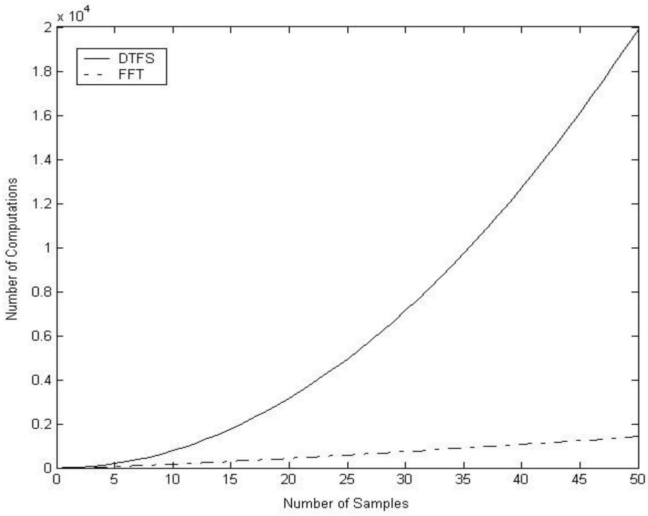


Figure 1: Computational Loads of DFT and FFT

As shown in figure1 there is a significant advantage to using the FFT over the DFT as the sample length (N) increases. For example, consider a sample length (N) of 32,768. Calculating the DFT directly would require 8,590,000,129 computations while calculating the FFT would require only 2,457,600. This means that if the calculation time for a computer to evaluate the FFT is 30 seconds, then it would require 29.13 hours to resolve the DFT.

III. BIQUAD STRUCTURE

Digital filters may be realized in hardware, software, or more often in firmware. Except for the most complex applications, they tend to be more expensive than analog filters, but they may be able to do things that analog filters cannot. Digital filters can be classified according to a wide range of characteristics, not all of which are mutually exclusive. The biquad is the preferred building block used to design digital filters by computer. Coefficients and structure are optimized for a specific performance Biquads are the preferred structure due to mathematical considerations.

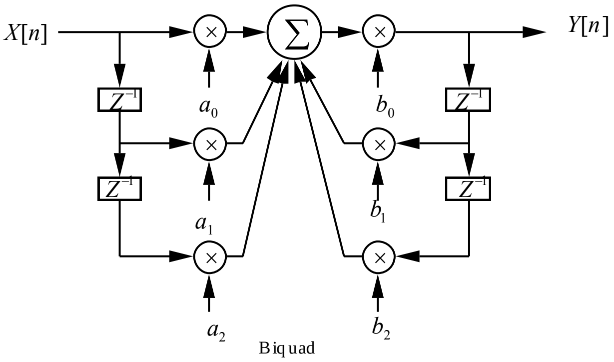


Figure 2: Biquad Implementation

The transfer function for the biquad if $b_0 = 1$ is given by:

$$\frac{Y[z]}{X[z]} = \frac{a_0 + a_1 z^{-1} + a_2 z^{-2}}{1 - b_1 z^{-1} - b_2 z^{-2}} \qquad (6)$$

To evaluate the frequency response, let $z = e^{j\omega T}$

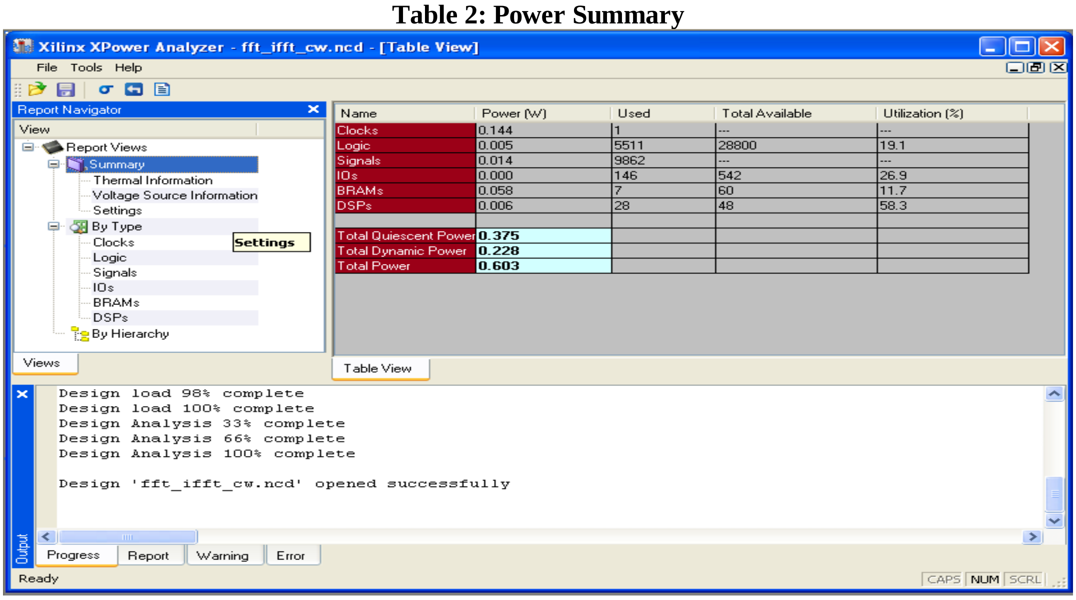
IV. IMPLEMENTATION OF FFT/ IFFT

The model for FFT and IFFT has been generated using system generated which was integrated with Matlab 2006b. The model has been simulated for Xilinx xc5vlx50t-1ff1136 device.

Table 1: Summary of resources utilized by the design

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	7046	28800	24%
Number of Slice LUTs	4542	28800	15%
Number of fully used LUT-FF pairs	4169	7419	56%
Number of bonded IOBs	146	480	30%
Number of Block RAM/FIFO	7	60	11%
Number of BUFG/BUFGCTRLs	1	32	3%
Number of DSP48Es	28	48	58%

Table 2: Power Summary



Name	Power (W)	Used	Total Available	Utilization (%)
Clocks	0.144	1	---	---
Logic	0.005	5511	28800	19.1
Signals	0.014	9862	---	---
IOs	0.000	146	542	26.9
BRAMs	0.058	7	60	11.7
DSPs	0.006	28	48	58.3
Total Quiescent Power	0.375			
Total Dynamic Power	0.228			
Total Power	0.603			

© 2013, IJARCSSE All Rights Reserved

The design has used 7046 Slice Registers, 4542 Slice LUTs, 4169 fully used LUT-FF pairs, 146 bonded IOBs, 7 Block RAM/FIFO, 1 BUFG/BUFGCTRLs and 28 DSP48Es out of totally available 28800 Slice Registers, 28800 Slice LUTs, 7419 fully used LUT-FF pairs, 480 bonded IOBs, 60 Block RAM/FIFO, 32 BUFG/BUFGCTRLs and 48 DSP48Es. Thus %age wise utilization comes out to be 24% for Slice Registers, 15% Slice LUTs, 56% for fully used LUT-FF pairs, 30% for bonded IOBs, 11% for Block RAM/FIFO, 3% for BUFG/BUFGCTRLs and 58% for DSP48Es. The summary of power utilization shows that the design will only use 603 mW of power also the estimated value of junction temperature is 31.2 degree centigrade.

V. Conclusion

The FFT/IFFT has been implemented using system generator tool from Xilinx. The results show that the design only uses 24% of available Slice Registers, 15% of available Slice LUTs, 56% of available fully used LUT-FF pairs, 30% of available bonded IOBs, 11% of available Block RAM/FIFO, 3% of available BUFG/BUFGCTRLs and 58% of available DSP48Es. Also the design only consumes 603 mw of power.

References

[1] Baron Jean Baptiste Fourier (http://bartleby.com/65/fo/Fouriers.html)
[2] HP, “The Fundamentals of Signal Analysis”, Hewlett Packard Application note 243, 1994
[3] L. Sevgi, F. Akleman, L. B. Felsen, “Ground Wave Propagation Modeling: Problem-matched Analytical Formulations and Direct Numerical Techniques”, IEEE Antennas and Propagation Magazine, Vol. 44, No.1, pp.55-75, Feb. 2002
[4] L. Sevgi, Complex Electromagnetic Problems and Numerical Simulation Approaches, IEEE Press – John Wiley and Sons, June 2003
[5] M. Levy, Parabolic equation methods for electromagnetic wave propagation, IEE, Institution of Electrical Engineers, 2000
[6] Jae Jeong Hwang Kamisetty Rao, Do Nyeon Kim. Fast Fourier Transform.
[7] David Renshaw Peter B. Denyer. VLSI signal processing:a bit-serial approach. 1985.

© 2013, IJARCSSE All Rights Reserved