Jupiter Publications Consortium

Vol. 2 Iss.4, pp. 81-84, 29th Feb, 2017
ISSN Print : 2454-1435 © 2017, IJRMMAE
ISSN Online : 2454-1443 © 2017, IJRMMAE
http://www.ijrmmae.in

International Journal of Research in Mechanical, Mechatronics and Automobile Engineering (IJRMMAE)

Implementation of A FFT Using High Speed and Power Efficient Multiplier

Dr.R.Sivakumar
Department of Electrical and Electronics Engineering,Shadan College of Engineering and Technology HYD,T.S,INDIA
Received 2, February 2017 | Accepted 29, February 2017

Abstract

Fast Fourier Transform (FFT) is used to convert a sign from Time area to frequency area and this is needed so that you can view the frequency components existing in the signals. A Fourier Transform converts a wave in the time area to the frequency domain. An FFT is an algorithm that pace up the calculation of a DFT. In core, an FFT is a DFT for speed. The whole purpose of an FFT is to pace up the calculations. The Decimation- In-Time radix- 2 FFT the use of butterflies has designed. The butterfly operation is faster. The outputs of the shorter transforms are reused to compute many outputs; as a result the total computational price will become less. An FFT is a very environment friendly DFT calculating algorithm. For the graph of FFT the some of the distinctive modules are used primarily Adders and Multipliers plays an essential function in the sketch of FFT. The average performance of the FFT is based on the throughput of the Multiplier. Here the multiplier with AHT is used to minimize the strength consumption and to make bigger the velocity of the FFT.
Keywords : Decimation in Time (DIT), Adaptive Hold Technique (AHT), Fast Fourier Transform (FFT), Discrete Fourier Transform (DFT).

I. Introduction

The Fast Fourier Transform (FFT) is a discrete Fourier radically change algorithm which reduces the quantity of computations needed for N factors from 2N6 to 2N log N, the place log is the base-2 logarithm. FFTs have been first discussed by Cooley and Tukey (1965), although Gauss had actually described the imperative factorization step as early as 1805 (Bergland 1969, Strang 1993). A discrete Fourier transform can be computed the usage of an FFT via potential of the Danielson- Lanczos lemma if the wide variety of factors N is a strength of two. If the wide variety of factors N is not a electricity of two, a seriously change can be performed on sets of factors corresponding to the top factors of N which is slightly degraded in speed. An efficient real Fourier radically change algorithm or a quick Hartley radically change (Bracewell 1999) gives a in addition expand in velocity by way of approximately a component of two. Base-4 and base-8 fast Fourier transforms use optimized code, and can be 20-30% quicker than base-2 speedy Fourier transforms. Which potential a 1024 sample FFT is 102.4 instances faster than the “straight” DFT. For larger numbers of samples the velocity gain improves. Prime factorization is sluggish when the elements are large, but discrete Fourier transforms can be made quick for N=2, 3, 4, 5, 7, 8, 11, 13, and sixteen using the Winograd radically change algorithm. Fast Fourier seriously change algorithms commonly fall into two classes: decimation in time, and decimation in frequency. The Cooley-Tukey FFT [2] algorithm first rearranges the enter factors in bit-reversed order, and then builds the output transform (decimation in time). The Sande-Tukey algorithm (Stoer & Bulirsch 1980) first transforms, then rearranges the output values (decimation in frequency). This paper is based totally carried out in the Decimation in Time class. Before going in element with DIT FFT here are the primary terms to apprehend the FFT

Jupiter Publications Consortium

Vol. 2 Iss.4, pp. 81-84, 29th Feb, 2017
ISSN Print : 2454-1435 © 2017, IJRMMAE
ISSN Online : 2454-1443 © 2017, IJRMMAE
http://www.ijrmmae.in

International Journal of Research in Mechanical, Mechatronics and Automobile Engineering (IJRMMAE)

Implementation i.e. Danielson-Lanczos Lemma (D-L Lemma) is required for long equation

A. Danielson-Lanczos Lemma (D-L Lemma)

The Danielson-Lanczos Lemma (D-L Lemma) equation for FFT is given as

$$F(n) = \sum_{k=0}^{N-1} x(k) e^{\frac{-j2\pi kn}{N}}$$

$$= \sum_{k=0}^{\frac{N}{2}-1} x(2k) e^{\frac{-j2\pi kn}{\left(\frac{N}{2}\right)}} + w_N^n \sum_{k=0}^{\frac{N}{2}-1} x(2k+1) e^{\frac{-j2\pi kn}{\left(\frac{N}{2}\right)}}$$

Here the DFT is broken up into two summations of half the size of the original. The first summation is the "even terms"

$$\sum_{k=0}^{\frac{N}{2}-1} x(2k) e^{\frac{-j2\pi kn}{\left(\frac{N}{2}\right)}}$$

$$w_N^n \sum_{k=0}^{\frac{N}{2}-1} x(2k+1) e^{\frac{-j2\pi kn}{\left(\frac{N}{2}\right)}}$$

W is the "twiddle factor".

B. Twiddle factor (W)

Twiddle factor is given by $W_N^n = e^{\frac{-j2\pi n}{N}}$.

C. Reverse Bit Format

A bit-reversal permutation is a permutation of a sequence of n items, the place n = 2k is a energy of two. It is described by using indexing the factors of the sequence by using the numbers from 0 to n-1 and then reversing the binary representations of each of these numbers. Each object is then mapped to the new position given with the aid of this reversed value. The bit reversal permutation is an involution, so repeating the identical permutation twice returns to the original ordering on the items.

D. Butterfly Diagram

The Butterfly design builds on the Danielson-Lanczos Lemma and the twiddle component to create an environment friendly algorithm. The Butterfly Diagram is the FFT algorithm represented as follows.

The basic easy 2 input butterfly mannequin is given in fig. 1.

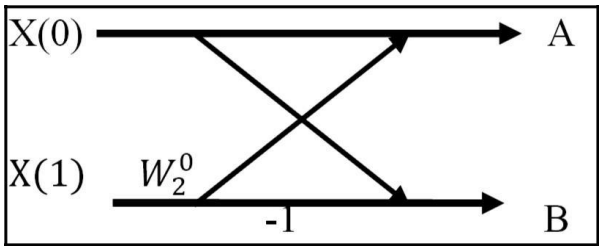

Fig. 1: 2 Point FFT (Butterfly Structure)

The mathematical illustration for the 2 input FFT is given as follows

Jupiter Publications Consortium

Vol. 2 Iss.4, pp. 81-84, 29th Feb, 2017
ISSN Print : 2454-1435 © 2017, IJRMMAE
ISSN Online : 2454-1443 © 2017, IJRMMAE
http://www.ijrmmae.in

International Journal of Research in Mechanical, Mechatronics and Automobile Engineering (IJRMMAE)

$$A = X(0) + W_2^0 * X(1)$$

$$B = X(1) - W_2^0 * X(0)$$

To perform the multiplication and addition duties for the implementation of FFT, the a range of sorts of multiplication tasks are used. The multiplication strategies are given beneath in detail.

II. Multiplers

A. Array Multiplier

The Array multiplier is nicely acknowledged due to its regular structure. Multiplier circuit is primarily based on add and shift algorithm. Each partial product is generated via the multiplication of the multiplicand with one multiplier bit. The partial product are shifted according to their bit orders and then added. The addition can be performed with regular lift propagate adder. N-1 adders are required the place N is the multiplier length.

			x	A3 B3	A2 B2	A1 B1	A0 B0	Inputs
			C	B0 x A3	B0 x A2	B0 x A1	B0 x A0	
		+	B1 x A3	B1 x A2	B1 x A1	B1 x A0		
		C	sum	sum	sum	sum		
	+	B2 x A3	B2 x A2	B2 x A1	B2 x A0			Internal Signals
	C	sum	sum	sum	sum			
+	B3 x A3	B3 x A2	B3 x A1	B3 x A0				
C	sum	sum	sum	sum				
Y7	Y6	Y5	Y4	Y3	Y2	Y1	Y0	Outputs

Fig. 2: Multiplication Process

The AM is a fast parallel multiplier and the multiplication process is as shown in

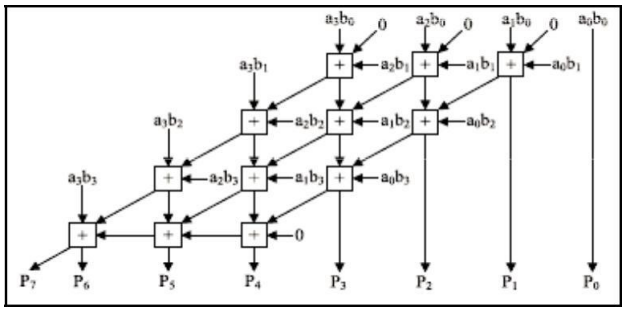

Fig. 3: Array Multiplier

Fig. 2 and fig. three indicates the block diagram of Array Multiplier. It consists of (n−1) rows of Carry Save Adder (CSA), in which every row includes (n − 1) Full Adder (FA) cells. Each FA in the CSA array has two outputs: (1) the sum bit goes down and (2) the elevate bit goes to the lower left FA. The closing row is a ripple adder for elevate propagation. The FAs in the AM are continually lively regardless of input. In the outcomes 16bit, 32bit array multipliers is designed and compared.

III. Experimental Results

Our experiments are conducted in a Windows operating system. The sketch of the various multipliers along with the proposed Multiplier are synthesized and simulated, and their synthesized outcomes are tabulated and compared. Figure – shows the RTL schematic for the FFT the usage of one of the ageing aware multiplier Technique. Table 1 is stuffed with the synthesis reports of the four bit multipliers and the Table 2 is given with the DIT-FFT synthesis reports. The implementation of DIT-FFT using traditional multipliers and the growing older aware multiplier are additionally completed the use of Verilog HDL in

Jupiter Publications Consortium

Vol. 2 Iss.4, pp. 81-84, 29th Feb, 2017
ISSN Print : 2454-1435 © 2017, IJRMMAE
ISSN Online : 2454-1443 © 2017, IJRMMAE
http://www.ijrmmae.in

International Journal of Research in Mechanical, Mechatronics and Automobile Engineering (IJRMMAE)

Xilinx14.1, and the simulations are located with ISE simulator. Xilinx Synthesizer is used to analyse the delay.

A. RTL Schematic of the Row Bypassing AHT Multiplier

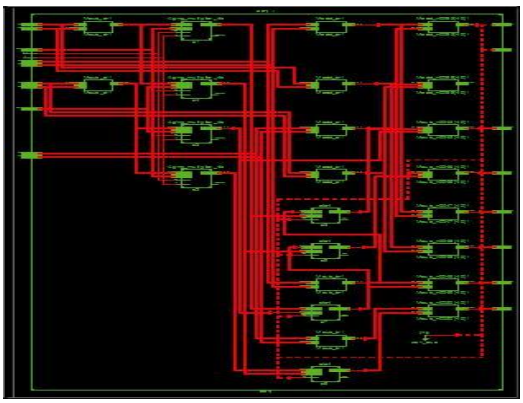

Fig. 8: RTL Schematic of ATH Row Bypassing Multiplier

IV. Future Scope

The implementation of radix-2, 4 point DIT-FFT is achieved the usage of Verilog HDL, as a future scope 8 point, 16 point, etc. DIT-FFTs can be applied with the assist of the getting old aware multiplier to get excessive pace and power environment friendly devices.

V. Conclusion

In this paper the implementation of high velocity FFT is designed to reduce the delay. The FFT which is applied using the AHT technique where as the AHT Multiplier has three important features. First, its Delay is very much less when compared to the other Traditional Multipliers. Second, it can provide dependable operations even after the aging impact occurs. The Razor flip-flops detect the timing violations and reexecute the operations the usage of two cycles. Last but no longer least, the AHT architecture is power environment friendly and it can also adjust the proportion of one-cycle patterns to minimize overall performance degradations due to the growing old effect. When the circuit is aged, and many blunders occur, the AHT circuit uses the second judging block to determine if an input is one cycle or two cycles and hence the timing mistakes can additionally be eliminated and can perform the error free operations. The proposed FFT is carried out using AHT structure in its multiplication system has a super gain in terms of Delay and hence, Adaptive Hold Technique can be noted as the dependable multiplier method which can be used in FFTs in harsh surroundings typically in aerospace applications etc.

References

[1] Lin, Chao, Yu-Hung Cho, and Yi-Ming Yang. "Aging-aware reliable multiplier design with adaptive hold logic." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*23.3 (2015): 544-556.

[2] Wang, Yuke, et al. "Novel memory reference reduction methods for FFT implementations on DSP processors." *IEEE Transactions on Signal Processing* 55.5 (2007): 2338-2349.

[3] Olivieri, Mauro. "Design of synchronous and asynchronous variable-latency pipelined multipliers." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 9.2 (2001): 365-376..

[4] Calimera, Andrea, Enrico Macii, and Massimo Poncino. "Design techniques for NBTI-tolerant power-gating architectures." IEEE Transactions on Circuits and Systems II: Express Briefs 59.4 (2012): 249-253.

[5] Paul, Bipul C., et al. "Impact of NBTI on the temporal performance degradation of digital circuits." IEEE Electron Device Letters 26.8 (2005): 560-562.