IMPLEMENTATION OF FFT PROCESSOR ON FPGA USING VEDIC MULTIPLIER

Neha Narvekar , Saurabh Raikar, Prabhakar Salkar, Ashish Shirodkar
ECE Department, AITD, Mapusa-Goa.

Received: February 20, 2019 Accepted: March 26, 2019

ABSTRACT: *Many "Digital Signal Processing" applications uses "fast fourier transform". Common practice is to run FFT on Digital signal processor or Application specific IC. In this paper 8 point FFT is implemented using Radix 2 butterfly method. 3 stage pipelining is used to speed up the processing. Complex multiplication is done by incorporating complex Vedic multiplier which further enhances the speed.*

Key Words:

Introduction*:* In a given signal to see the frequency components present, we use FFT(Fast Fourier Transform) and IFFT(Inverse Fast Fourier Transform) algorithms. It also helps us to convert a signal from time domain to frequency domain. To attain low power utilisation, effective area consumption and fast execution many researchers uses FFT processor in wireless communication. For implementing FFT researchers utilize FPGA in the present times. Vital role is being played by FPGA in real time "Digital Signal Processing". To get 'N' point,'2' point FFT is being achieved by splitting the bigger structure into smaller units. Hence the algorithm is termed as "Radix-2 Algorithm". This paper consists of **8 point radix-2** butterfly unit using pipelined architecture(3 stages), Complex "Vedic Multiplier" and CLA ("Carry Look Ahead Adder")[1]. Butterfly diagram builds on the twiddle factor to create an efficient algorithm.

Radix-2 FFT Algorithm: The fundamental concept of split and rule is included in FFT, which requires splitting a "N-point DFT" into immediately smaller DFTs[10]. When we divide it into parts of two at each stage, it is known a "Radix-2 Algorithm". The "Decimation In Time" (DIT) "FFT Algorithm" depends on splitting (decimating) x(n) into smaller sequences and finding X(k) from the DFTs of these decimated sequences. The other algorithm is decimation in

frequency (DIF) which is derived by decimating the output sequence X(k) into smaller and smaller sub-sequences. Let us consider the decimation in time of N = 2^r point DFT given below: X(k) = $\sum_{n=0}^{N-1} x(n) W_N^{nk}$

We now split the original DFT into two DFTs 0f size N/2 each, one contain the odd numbered samples and the other containing the even numbered samples as shown below

X(k) = X1(k) + X2(k) (3.9)

$= \sum_{m=0}^{\frac{N}{2}-1} x(2m).W_N^{2mk} + \sum_{m=0}^{\frac{N}{2}-1} x(2m+1).W_N^{(2m+1)k}$ (3.10)

But $W_N^2 = W^{\frac{N}{2}}$, Now substituting in the equation(3.10) we get

$X(k) = \sum_{m=0}^{\frac{N}{2}-1} f1(m).W_{\frac{N}{2}}^{km} + W_N^k \sum_{m=0}^{\frac{N}{2}-1} f2(m).W_{\frac{N}{2}}^{km}$ (3.11)

= F1(k) + W_N^k .F2(k) where k = 0, 1...N − 1 (3.12)

where F1(k) and F2(k) are the n 2 point DFTs of the sequences f1(m) and f2(m) respectively[12].

Since F1(k) and F2(k) are periodic, with a period N/2, we have F1(k + N/2) = F1(k) and F2(k + N/2) = F2(k). In addition the factor $W_N^{k+\frac{N}{2}} = -W_N^k$. Hence equation(3.12) can be expressed as[12]

X(k) = F1(k) + W_N^k.F2(k) where k = 0, 1...N/2 − 1 (3.13) [13]

X(k + N 2) = F1(k) − W_N^k.F2(k) where k = 0, 1...N/2 − 1 (3.14) [13]

From the above equations we see that direct computation of F1(k) requires $\frac{N}{2}.\frac{N}{2}$complex computations, similarly F2(k) also requires$\frac{N}{2}.\frac{N}{2}+\frac{N}{2}$ complex multiplications[12].

Algorithm of Vedic Multiplier

The "Urdhva-Tiryakbyham Sutra"(One of the method of Vedic multiplication to do multiplication) uses "Vertical And Crosswise" as shown below in the "Line Diagram". So based on a "Line Diagram" an algorithm is expanded as described below[6].

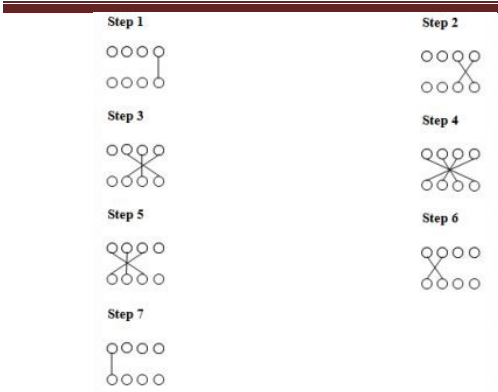


Fig:- Line Diagram [6].

Basic Butterfly Operation

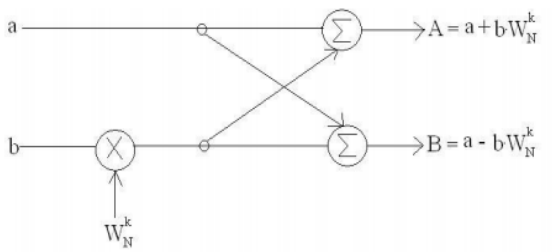


Figure 3.1: Basic Computation of FFT [2].

The butterfly is the most suitable mathematical schematic diagram with which "FFT's" are implemented. The above fig shows the basic computation performed at every stage. A complete 8 point FFT is shown in figure (3.2). Single butterfly operation involves four real multiplications and six real additions. Any DFT with N = 2^r points can be decimated "r" times, finally to form a summation of butterflies. There are log2N = k stages of computation[10]. Since each stage needs N/2 complex multiplications by the twiddle factor W_N^r and N complex additions, there are a total of $\frac{1}{2}$Nlog$_2$N complex multiplications and Nlog$_2$N complex additions[10].

n	Binary	Bit Reversed	N
0	000	000	0
1	001	100	4
2	010	010	2
3	011	110	6
4	100	001	1
5	101	101	5
6	110	011	3
7	111	111	7

Table: - Bit Reversal[11].

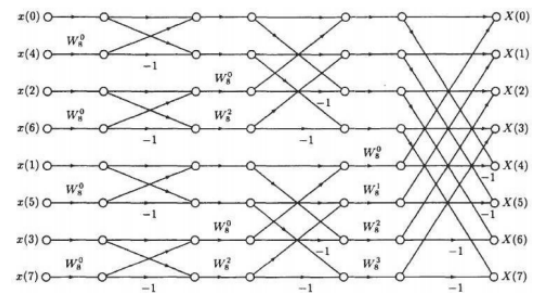
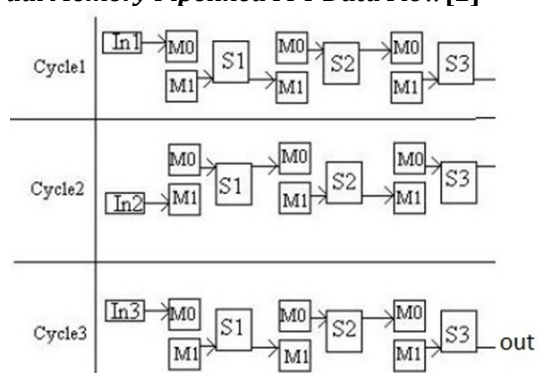


Figure 3.2: 8 point DIT FFT with bit reversed inputs [2].

Design flow of FFT processor

The following design steps need to be taken before proceeding to design the FFT system. 1. FFT size
2. Selecting the data format
3. Selecting the algorithm
4. Selecting the architecture
5. Selecting the target device
6. Appropriate testing and verification

***Dual Memory Pipelined FFT Data Flow*[2]**



Total 8 samples are loaded in memory M0 in cycle1. Then next 8 samples are loaded in memory M1 while stage1 processes data from M0 and writes into next stage M0 in cycle2. In the 3rd cycle stage1 processes data from M1 and also stage2 processes data from M0 and new set of data is loaded in the M0 of first stage. In this way all the stages calculate data simultaneously and forms a pipeline. The input samples increases by this method. The final FFT will be obtained from the last stage, one output value per clock.

Design of Pipelined FFT Processor

This FFT design uses the dual memory pipelined architecture as explained in the above section.

The figure shown below is the schematic drawing of 1st stage processor of FFT.

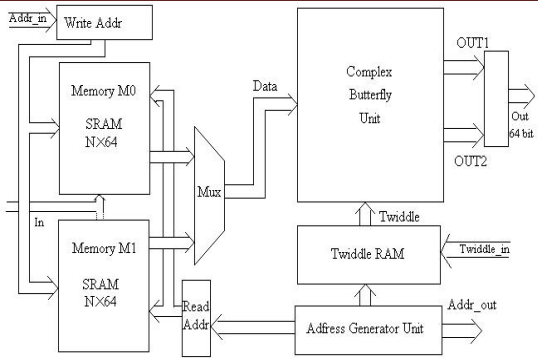


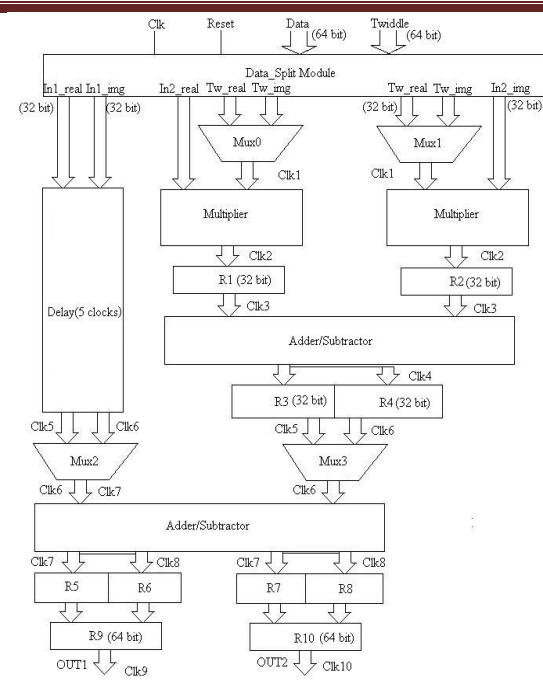
Fig: Single Stage of Pipelined FFT Design [2].

The design is based on three main processes.
1. Firstly Twiddle Coefficients are loaded into the internal memory of FPGA.
2. Secondly Data is loaded into the internal memory[14].
3. Then FFT is Computed.

Initially the system is being reset and the twiddle coefficients are stored in the Twiddle RAM and also at the same time in data memory M0 the incoming data samples are fed. Now by accessing data from M1 FFT processor begins its computation but zeros are being set to M1. When storing the entire data in M0 is done, all the location of M1 must have accessed by the FFT processor. Now M0 memory is being accessed once the FFT processor shifts to M0. Now the even stages is being computed with M1 memory and odd stages gets computed with M0 memory. At the output of the first stage the first processed data is available which takes 10 clock cycles after reset. The address of the data is available at the output of the stage. The stage done signal is claimed as a result of the synchronization of the storing of input data by the next stage which is also known as the previous stage output. The address generator produces the address for the memory to be approached. The address for the memory to be accessed is generated by the address generator. The memory of the next stage stores the output samples and the address for the same is provided by the address generator. At a period of time FFT calculation is being achieved,the alternate memory of the first stage is getting loaded with the new sets of data andtill FFT of a set of data points are calculated this process is being repeated.

Design of Building Blocks of FFT

***Complex Pipelined Floating Point Butterfly Unit*[2]**



Vedic Multiplier

Vedic mathematics is known for high speed compound mathematical circuit which is used in the field of image and signal processing [1]. To increase the speed of the Vedic multiplier a large number of multiplier strategies have been developed [1]. From the research it has been found that Vedic multiplier is one of the speedy and low power multiplier over traditional array and booth multipliers. Vedic multiplication is preferred for higher bit multiplication .The performance of Vedic multiplication technique is better than the Booth multiplication in terms of size and delay. The Vedic
Multiplier is being chosen accordingly to boost the overall presentation of "Butterfly Unit".

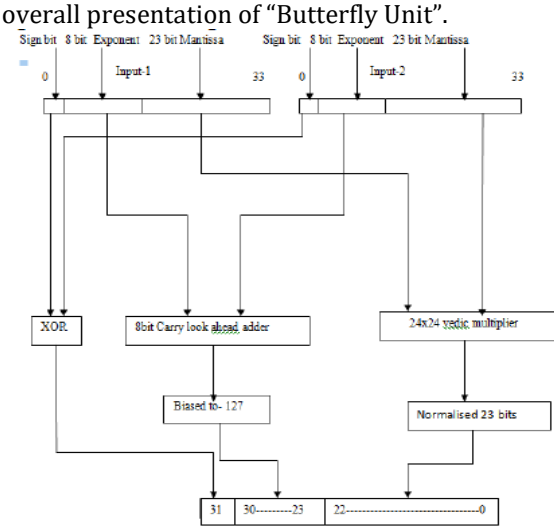


Fig: - 32×32 Bit Complex Vedic Multiplier [8].

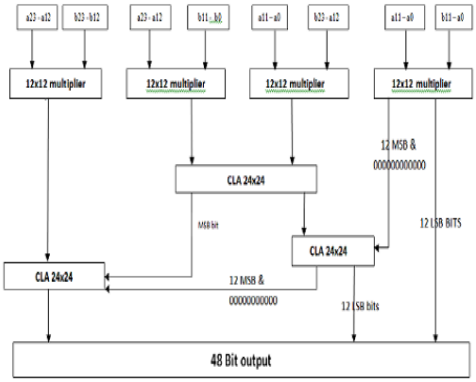


Fig: - 24×24 Bit Vedic Multiplier [8].

RESULT

The progress at the current stage is that we have successfully implemented the following things:

1. Generating any random signal and save it in text file through MATLAB.

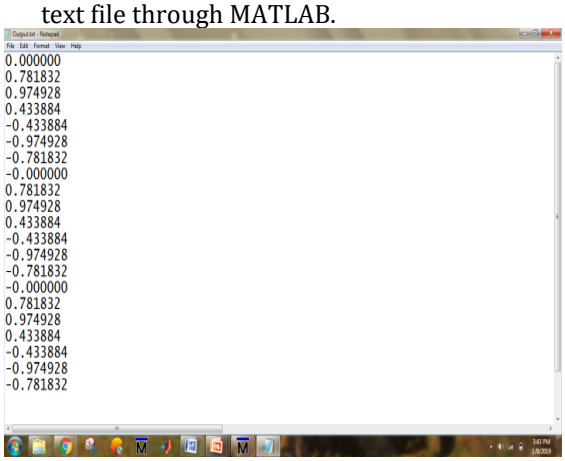


Fig: Simulation result

2. Implementation of 6 bit, 12 bit and 24 bit Vedic Multiplier.

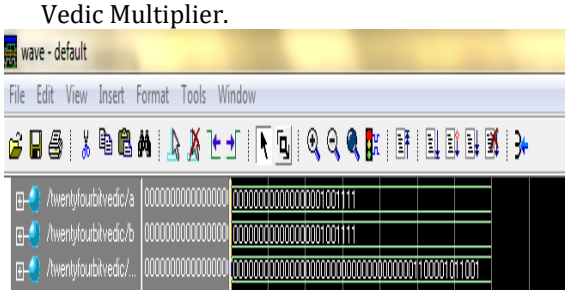


Fig: simulation result of 24 bit vedic multiplier

3. Implementation of 6 bit,8 bit,12 bit 24 bit CLA

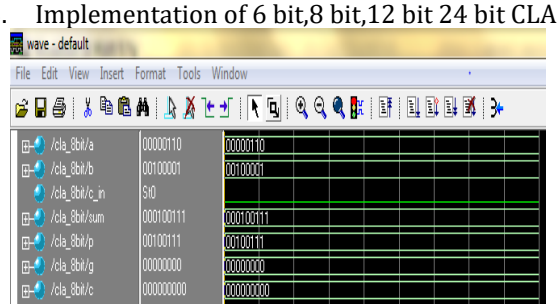


Fig: simulation result of 8 bit cla

4. IEEE 754(single precision) floating point multiplier implementation

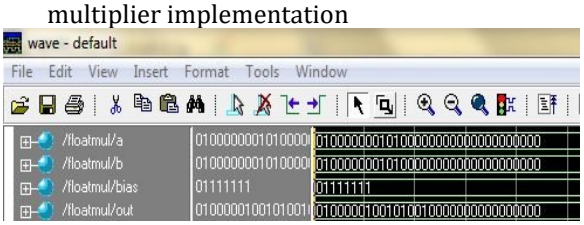


Fig: simulation result of 32bit multiplier

Conclusion and future work

We are planning to complete the coding part of our FFT processor using Pipelined architecture and Vedic Multiplier and later burning the code on a FPGA board.

ACKNOWLEDGEMENT

Firstly, praises and thanks to the God, the Almighty, for His blessings throughout our project work to complete the project successfully. We would like to express our deep and sincere credit to our project guide Mr. Laxmikanth bordekar, registrar/assistant professor, Agnel Institute of Technology and Design, our Laboratory Staff, our principal Dr. V. Mariappan for giving us the opportunity to do research and providing invaluable guidance throughout this work. He has helped us to carry out the project work. and to present the work as clearly as possible.

References

1. Avinash Kumar Singh and Ashutosh Nandi "Design of Radix 2 Butterfly structure using Vedic Multiplier and CLA on Xilinx" 3-4 March 2017/ IEEE
2. Laxmikant D. Bordekar, Gajanan S. Gawde and Dr. R. B. Lohani " Design of 1024 point pipelined FFT processor and Comparison of Software and FPGA Implementation of FFT"
3. P. Balasubramaniam , C. Dang , D.L. Maskell and K.. Prasad "Approximate Ripple Carry and Carry Look ahead Adders – A comparative study" 9-11 October 2017/ IEEE
4. Ankita Jain and Atush Jain " Design, Implementation & Comparison of Vedic Multipliers with conventional multipliers" 2017/IEEE
5. Josmin Thomas, R. Pushpanadan andJinesh s " Comparative study of performance Vedic Multiplier on the basis of adders" 19-20 December 2015/IEEE
6. AbhyarthanaBisoyi, MituBaral and Manoja Kumar Senapati "Comparison of 32-bit Vedic Multiplier with a conventional Multiplier" 2014/IEEE
7. YouTube tutorials
8. Swapnil Suresh Mohite ,Sanket Sanjay Nimbalkar , MadhavMakarandBhatkhande , Mrs. Rashmi Rahul Kulkarni "32 Bit Floating

Point Vedic Multiplier"2016/IOSR journal of VLSI and signal processing.
9. K.S. Thyagarajan."introduction to digital signal processing using MATLAB with Application to digital communications",springer Nature,2019.
10. Juan M Vilardy. "Design and implementation in VHDL code of the two dimensional fast Fourier transform for frequency filtering, convolution and correlation operations", journal of physics conference Series, 01/01/2011
11. vdocuments.mx
12. www.cmlab.csie.ntu.edu.tw
13. viplab.cs.nctu.edu.tw
14. www.ijrst.org