V SULOCHANA , et al , International Journal of Computers Electrical and Advanced Communication Engineering [IJCEACE]™
Volume 1 , Issue 6, PP: 38 - 41 , SEPTEMBER 2014.

DEVELOPMENT OF 32-POINT FFT USING RADIX-4 IMPLEMENTATION

V.Sulochana 1*, K.Naga Lakshmi 2*

1. II.M.Tech (VLSI), Dept of ECE, AM Reddy Memorial College of Engineering & Technology, Petlurivaripalem.

2. Asst. Prof , HOD- Dept. of ECE, AM Reddy Memorial College of Engineering & Technology, Petlurivaripalem.

ABSTRACT

Now a day's users preferred digital electronic devices to meet their technical requirements of the system and it should be accurate and fast. Devices efficiency or accuracy depends on it's both intra and inter constraints based peripheral algorithms. This paper presents a development of FFT (Fast Fourier Transform), based on Decimation In Time (DIT) domain is called Radix-4 point DIT-FFT algorithm. Development of this entity is use VHDL design and for simulation Xilinx ISE. Computation and analysis of 256-bit, 64-point FFT is an efficient speed and implemented on FPGA Spartan -3E kit.

Keywords: DIFFFT, FPGA Spartan-3E kit.

I. INTRODUCTION

The designed system speed depends on their intra and inter peripherals the intra peripherals depends on designer choice and the inter peripherals depends on the users choice. The designer choice includes components and an algorithm, in that user's choice includes inputs, an external device etc. this paper has been focus on radix-2 algorithm which has more delay. To overcome this problem there is a need to modify in the algorithm. This paper deals extension of the existing algorithm called radix-4 algorithm mainly it focus on design and implementation of 256-bit 64-point DIT-FFT for FPGA kit. The simulation and synthesis is performed by Model-Sim ISE and Xilinx ISE Designs respectively, and we dumped on FPGA Spartan-3E for verification.

In this paper proposes and concentrate on the design of 32 and 64 point FFT and its performance analysis. The VHDL as a design entity the synthesis and stimulation is done on Xilinx ISE Design Suite 13.2.A DFT Decomposes a sequence of values into components of different frequencies. An FFT is a way to compute the same result but more quickly gets: Computing a DFT of N-points takes O(N2) Arithmetical operations while an FFT can compute the same DFT in only O (N.logN) operations. FFTs can decomposed using DFTs of even and odd points which is called a decimation in-time (DIT) FFT or they can decomposed using another approach which is called a Decimation-infrequency (DIF) FFT.

The Radix-2 algorithm mainly depends on various components and number of stages increases proportionally computations also increases. The 16-point FFT have complex multiplication and additions are both 32 and 64 respectively and for 32 point FFT these are 90 and 180 respectively. The inputs are more it became impossible to calculate by using Radix-2 algorithm. By this the number of inputs are more the design complexity also more.

II. Proposed Radix-4 system

This paper propos a system is based on radix-4 algorithm. The Radix-4 is another FFT algorithm which was surveyed to improve the speed of functioning by reducing the computation time; and this can be obtained by change the base to 4. The same number if base increases the power/index will decreases. For radix-4 the number of stages are reduced to 50% since $N=4^3$ ($N=4^M$) i.e. only 3 stages. Radix-4 is having four inputs and four outputs and it follows in-place algorithm. The following will explain the functioning of radix-4 and how the computational complexity is reduced.

Functioning of radix-4 algorithm :

The radix-4 DIT-FFT recursively partitions a DFT into four quarter-length DFTs of groups of every fourth time sample. The outputs of these shorter FFTs are reused to compute many outputs, thus greatly reducing the total computational cost. The radix-4 FFTs require only 75% as many complex multiplications as the radix-2 FFTs.

radix-4 decimation-in-time and decimation-in-frequency Fast Fourier transforms (FFTs) gain their speed by reusing the results of smaller, intermediate computations to compute multiple DFT frequency outputs. The radix-4 decimation-in-time algorithm rearranges the DFT equation into four parts: sums over all groups of every fourth discrete-time index n = [0, 4, 8 ... N - 4], n = [1, 5, 9 ... N - 3], n = [2, 6, 10 ... N - 2] and n = [3, 7, 11 ... N - 1], (This works out only when the FFT length is a multiple of four.) Just as in the radix-2 DITFFT, further mathematical manipulation shows that the length-N DFT can be computed as the sum of the outputs of four length-N/4 DFTs, of the even-indexed and odd-indexed discrete-time samples, respectively, where three of them are multiplied by so-called twiddle factors Wk N = e-(i 2лk/N), W2kN , and W3kN . Decimate/split the *N*-point input sequence into four sub sequences, $x(4n)$, $x(4n+1)$, $x(4n+2)$, $x(4n+3)$, $n = 0, 1, \ldots, N/4-1$.

$$\begin{aligned} X(k) &= \sum_{n=0}^{N-1} x(n)e^{(-j2\pi nk/N)} \\ &= \sum_{n=0}^{\frac{N}{4}-1} x(4n)e^{(-j2\pi(4n)k/N)} + \\ &\quad \sum_{n=0}^{\frac{N}{4}-1} x(4n+1)e^{(-j2\pi(4n+1)k/N)} + \\ &\quad \sum_{n=0}^{\frac{N}{4}-1} x(4n+2)e^{(-j2\pi(4n+2)k/N)} + \\ &\quad \sum_{n=0}^{\frac{N}{4}-1} x(4n+3)e^{(-j2\pi(4n+3)k/N)} \end{aligned} \quad (1)$$

X(k) = {DFT N/4 [x(4n)] + Wk N DFT N/4 [x(4n+1)] + W2kN DFT N/4 [x(4n+2)] + W3kN DFT N/4 [x(4n+3)]} (2)

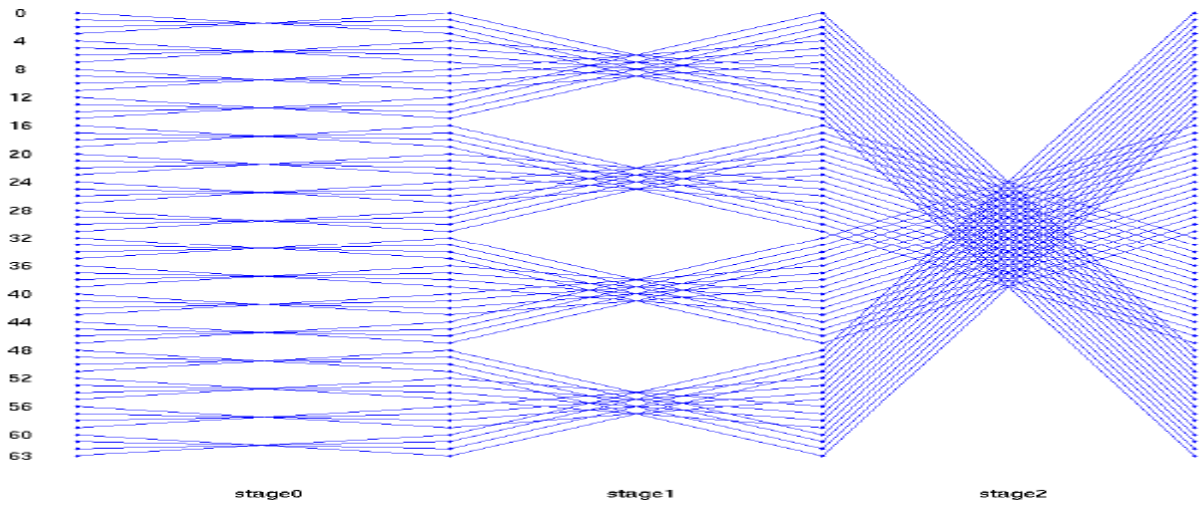

Fig.1. 64-point radix-4 DITFFT butterfly diagram

Fig1. It depicts a 64-point radix-4 DIT-FFT using the butterfly symbol represented as mathematical operations.

III. SIMULATION RESULTS

Fig.1. represent a 64-point radix-4 DIT-FFT butterfly diagram in which there are three stages. These stages are top radix, comutator , radix 8 and 4 basic operations.

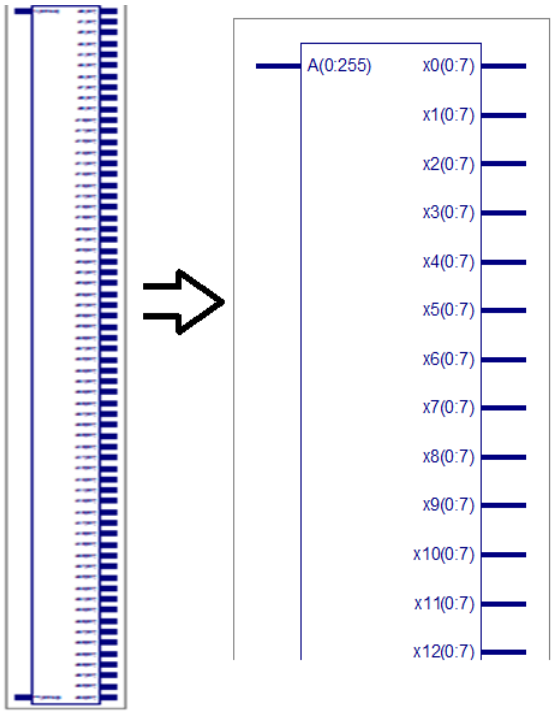

Fig.2. RTL view of 256-bits 64-points radix-4 DIT-FFT.

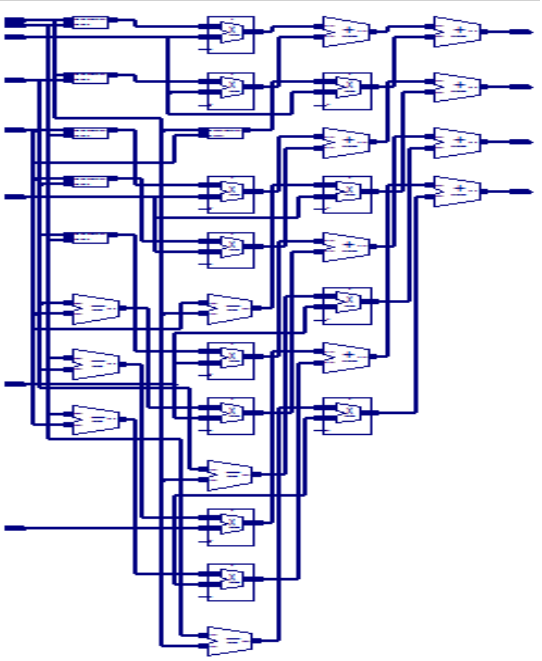

Fig.3. Internal diagram of even/Odd part.

The internal RTL view of top module is shown in fig. 2 in which 256-bit input and 512-bit outputs are present. Internal structure of top module consists of four top butters which are equal in size and having same number of inputs. Internal view of top butter M2, in this module even and odd parts are present and a comutator is used to connect all these parts inside the M2, and this is common to other three modules M3,M4 and M5. Internal structure of comutator, Even and odd parts of top radix 8 of internal diagram of even/odd part consists of adder, subtractor and multiplier as an internal components, is shown in fig. 3.

IV. CONCLUSION

This paper presents a new high speed FFT architecture based on radix-4 algorithm. For that we use pipelined 256-bit 64-point radic-4 DIT-FFT can be implemented easily by using FPGA technology, offered probability by this algorithm. The above synthesis results of radix-4 64 point is understandable but the radix-4 having less delay then compared to radix-2. Comparing with radix-2 technique 75% of time is saved in radix-4 algorithm. In this proposed paper the time delay is reduced then automatically the system speed is increased.

REFERENCES

[1] Asmitha Haveliya, Amity University Lucknow, India "Design And Simulation Of 32-Point FFT Using Radix-2 Algorithm For FPGA Implementation" 2012 Second International Conference on Advanced Computing & Communication Technologies.
[2] J. W. Cooley and J. W. Tukey, "An Algorithm for Machine Calculation of Complex Fourier Series," Math. Comput., vol. 19, pp. 297–301, Apr. 1965
[3] Douglas L. Jones "Decimation-in-Time (DIT) Radix-2 FFT Algorithms" Connexions module: m12016
[4] J. G. Proakis and D. G. Manolakis, Digital Signal Processing Prentice-Hall, 1996.
[5] J. Rabaey,A. Chandrakasan, and B. Nikolic, Digital Integrated Circuits A Design Perspective. Prentice-Hall, 2003.
[6] K. Parhi, VLSI Digital Signal Processing Systems New York, NY, USA: John Wiley & Sons, 1999.
[7] S. Johansson, S. He, and P. Nilsson, "Wordlength Optimization of a Pipelined FFT Processor," in Proc. of 42nd Midwest Symposium on Circuits and Systems, Las Cruces, NM, USA, Aug. 8-11 1999.
[8] Douglas L. Jones "Radix-4 FFT Algorithms" Connexions module: m12027
[9] W. Li and L. Wanhammar, "A Pipelined FFT Processor," in IEEE Workshop on Signal Processing Systems, 1999, pp. 654–662.