IOSR Journal of Engineering (IOSRJEN) **www.iosrjen.org**
ISSN (e): 2250-3021, ISSN (p): 2278-8719
PP 68-72

FFT Design Using Model Sim

Ms. Sneha Kherde
Department of Electronics and Telecommunication Sipna College of Engineering and Technology (Sant Gadgebaba Amravati University) Amravati, Maharashtra
Kherde.sneha@gmail.com

Abstract: *FFT is used in digital communication and in digital signal processing and is widely used as a mathematical tool in different areas. Fast Fourier Transforms are algorithms for quick calculation of discrete fourier transform of data vector. In this paper, we had designed an 8-point FFT and its computation time. Here we had simulated and implemented Radix-2 8-point FFT in hardware using hardware language (VHDL) here time constraint is measured with the help of ModelSim SE.*

Index Terms: *DFT, FFT, FPGA, butterfly, radix.*

I. Introduction

The Fourier transform takes a signal in time domain (where each sample in the signal is associated with a time) and maps it, without loss of information, into the frequency domain. The discrete Fourier transform (DFT) is a one method of the Fourier transform, which is used in Fourier analysis. If a signal is modified in one domain, it will also be changed in the other domain. The term Fourier transform refers both to the frequency domain representation of the signal and the process that transforms signal to its frequency domain representation.. It transforms one function that is time into another that is frequency, so as to get discrete signals, hence called the DFT, of the original function. The DFT is to compute the sequence $\{X(k)\}$ of N complex-valued numbers given another sequence of data $\{x(n)\}$ of length N, according to the formula

$$X(k) = \sum_{n=0}^{N-1} x(n) W_N^{kn}, \qquad 0 \le k \le N-1$$

$$W_N = e^{-j2\pi/N} \qquad \text{-----(1)}$$

Where, W_N is twiddle factor or phase rotation factor. Twiddle factors referred to as the root-of-unity complex multiplicative constants in the butterfly operations of the FFT algorithm, used to recursively combine smaller discrete Fourier transforms.

We know that for each value of k,(N-1) complex addition and N complex multiplications are involved for computation of $X(k)$.. Similarly, to calculate all possible values of the DFT, N^2-N complex additions and N^2 complex multiplications are required [4][7].

A fast Fourier transform (FFT) is an efficient algorithm to compute the discrete Fourier transform (DFT). The computation of FFT can be performed in three stages.The Fast Fourier Transform is an enhanced computational algorithm to implement the Discrete Fourier Transform to an array of 2^N samples where, N is the length of samples. FFT's are algorithms for quick calculation of discrete Fourier transform of a data vector. The FFT is a DFT algorithm which reduces the number of computations needed for N points from O(N2) to O(N log N), where log is the base-2 algorithm. The ordering reduces the number of computations of the twiddle-factor values[12].

"DFT" is an method which contain mathematical function, re whereas "FFT" contain specific algorithms for computing DFT's. Figure-1 shows a FFT algorithm in radix-2 decimation in time for length 8-signal [2]. FFT designed in this paper created by using three phases, whose first phase can be discussed further and similarly the other two phases are created.

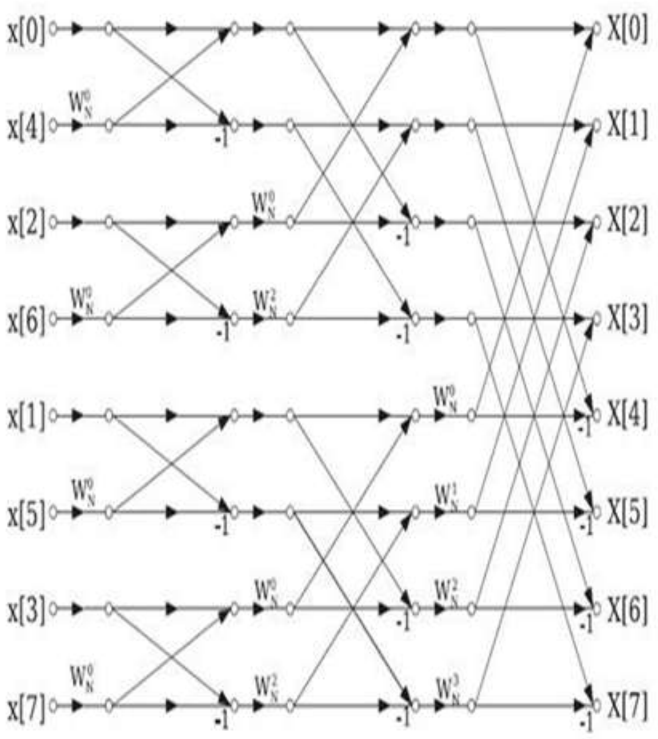


Figure 1: FFT algorithm in Radix-2 decimation in time

II. RADIX-2 DIT FFT ALGORITHM

With the help of field programmable gate arrays (FPGAs), one can provide hardware for application specific computation design. We can make the changes in designs in FPGA's and this can be implemented within a few hours, thus it result in savings in cost and design cycle [10].

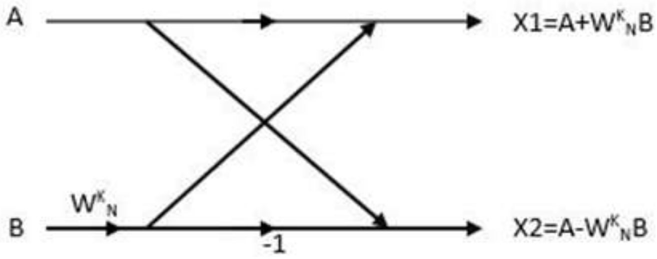


Figure 2: Butterfly for radix-2 DIT FFT

Figure-2, shows butterfly diagram for radix-2 DIT FFT. As shown in figure 1, is the first basic phase of FFT algorithm. There is the presence of complex number in twiddle factor therefore we need to use complex operation . Here from above diagram's equation, there are three blocks used that is complex addition, complex subtraction, complex multiplication. For any complex number multiplier design, the most critical part is the multiplication process. The complex multiplier (used for multiplication of twiddle factor) is shown in figure-3, where three simple multiplier, complimentor, adder has been used [9]. The implementation of a complex multiplier uses both the adders and complimentor needed for the FFT. The design of a FFT can be derived from the complex multiplier design [1].

In any multiplication operation, there are three major steps. For the first step, the partial products are generated. For the second step, the partial products are reduced to one row of final sums and carries. For the third step, the final sums and carries are added to generate the result

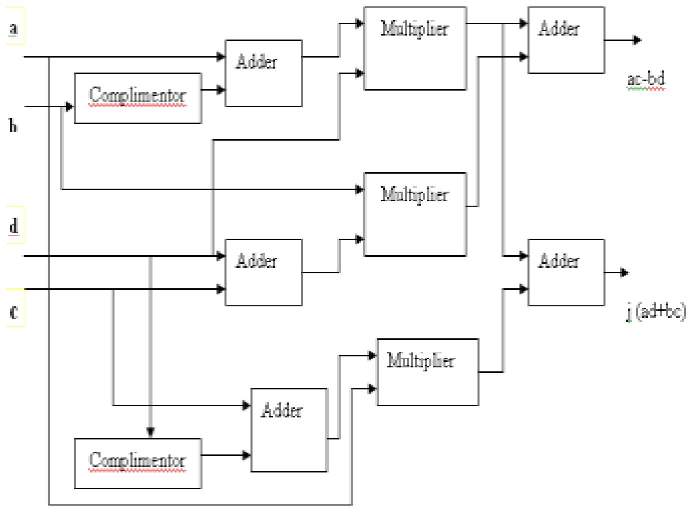


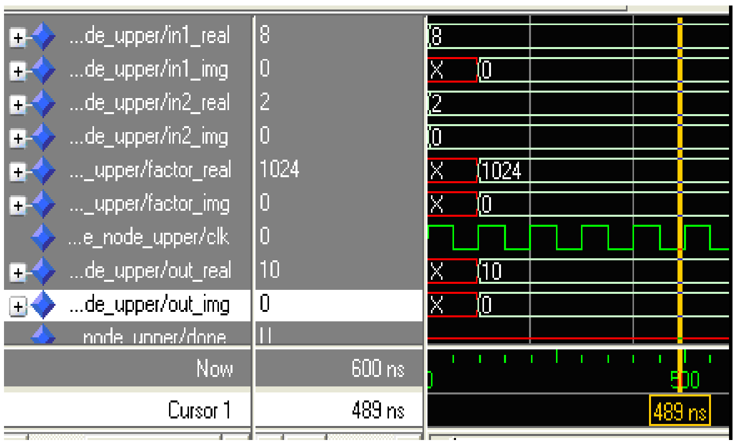
Figure 3: Block diagram of complex multiplier

Here the complex number multiplication can be divided into two main components known as real part and imaginary part as in [3]. This complex multiplier is shown by the following equation
(a + jb)(c+jd) = (ac – bd) + j(bc + ad)-------(2)
From equation 2, real part output is (ac – bd) and imaginary part output is (bc + ad). The real and imaginary parts of the output will be kept separate in design. Multiplier is an block which provide more hardware compared to those blocks which are used in FFT.

Equation 2 has the term (ac – bd) and (bc + ad) which can be modified as shown
(ac – bd) = a (c-d) + d (a-b)----------- (3)
(bc + ad) = b (c+d) + d (a-b)----------- (4)
By using above equations three multipliers and five adders are used, which reduces our hardware structure. Block diagram of complex multiplier using three multiplier is as shown in figure 3.
Multiplication has been done in such a way that “a” and “b” are always kept in multiplier and “c”,”d” are kept in multiplicand. By implementing this technique, several logic gates can be reduced. Complex multiplier finds extensive applications in areas like digital signal processing, communication etc. It is one of basic building block of digital system. The multiplier, adder and complementor are all built using half adders and full adders. Half adders find the sum of two binary numbers and provide a sum and a carry. A full adder is similar to the half adder however, it has an input for a carry bit. Full and half adders can be constructed using logic gates.
however, this is not the most efficient solution. This completes our first butterfly diagram refered in figure 2. The combination of these blocks can form complete FFT.

III. Simulation Result

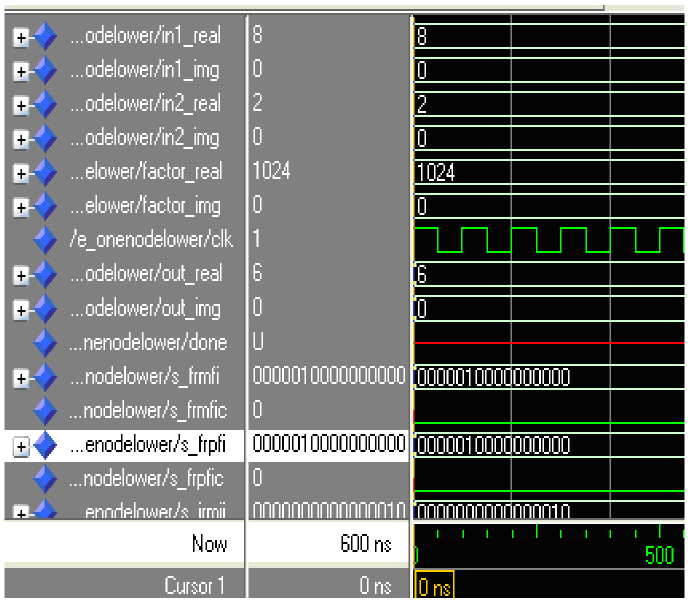
ModelSim SE (version 6.2c) has been used for simulation of VHDL code and same codes for hardware implementation can be synthesized using Xilinx ISE (version 6.2c). This paper has discussed about the DIT FFT algorithm. So as to reduce time of performance, complex multiplier with three multiplier is used. The output simulation result of X1 (Upper node) of radix-2 (figure-2) is as shown in figure-4. Here “in1_real” and “in1_img”is a first input with real and imaginary part respectively. The output term is also a complex value denoted by “out_real” and “out_img”.



```
...de_upper/in1_real      8
...de_upper/in1_img       0
...de_upper/in2_real      2
...de_upper/in2_img       0
..._upper/factor_real     1024
..._upper/factor_img      0
...e_node_upper/clk       0
...de_upper/out_real      10
...de_upper/out_img       0
...node_upper/done        U
Now                       600 ns
Cursor 1                  489 ns
```

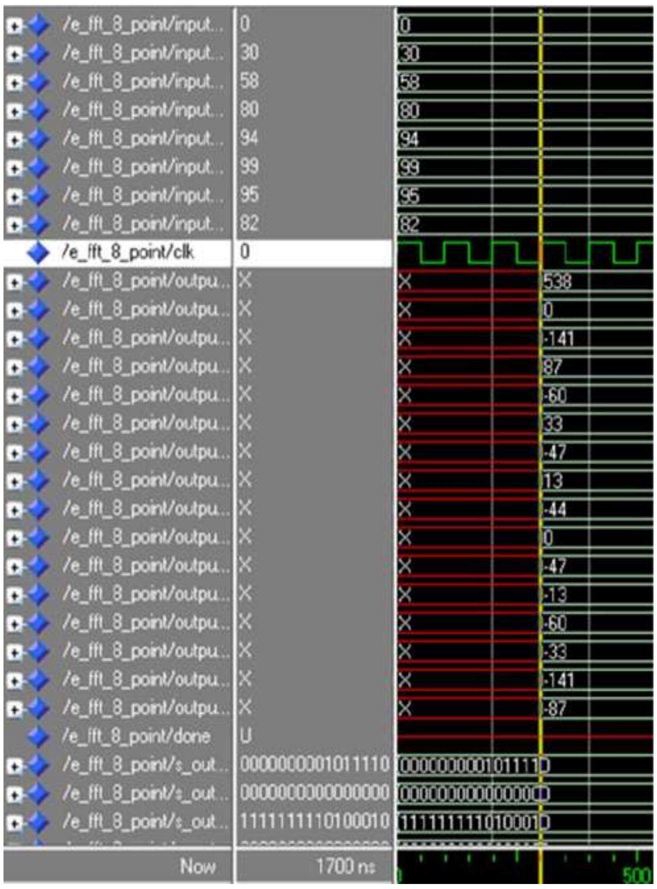
Figure 4: Simulation result of X1(upper node)

Similarly, output simulation result of X2 (lower node) of radix-2 (figure-2) is shown in figure-5. These are the result of 1st stage FFT as specified in Figure-1



```
...odelower/in1_real      8
...odelower/in1_img       0
...odelower/in2_real      2
...odelower/in2_img       0
...elower/factor_real     1024
...elower/factor_img      0
/e_onenodelower/clk       1
...odelower/out_real      6
...odelower/out_img       0
...nenodelower/done       U
...nodelower/s_frmfi      0000010000000000
...nodelower/s_frmfic     0
...enodelower/s_frpfi     0000010000000000
...nodelower/s_frpfic     0
...enodelower/s_irmii     0000000000000010
Now                       600 ns
Cursor 1                  0 ns
```

Figure 5: Simulation result of X2 (lower node)



```
/e_fft_8_point/input...   0                   0
/e_fft_8_point/input...   30                  30
/e_fft_8_point/input...   58                  58
/e_fft_8_point/input...   80                  80
/e_fft_8_point/input...   94                  94
/e_fft_8_point/input...   99                  99
/e_fft_8_point/input...   95                  95
/e_fft_8_point/input...   82                  82
/e_fft_8_point/clk        0
/e_fft_8_point/outpu...   X                   X   538
/e_fft_8_point/outpu...   X                   X   0
/e_fft_8_point/outpu...   X                   X   -141
/e_fft_8_point/outpu...   X                   X   87
/e_fft_8_point/outpu...   X                   X   -60
/e_fft_8_point/outpu...   X                   X   33
/e_fft_8_point/outpu...   X                   X   -47
/e_fft_8_point/outpu...   X                   X   13
/e_fft_8_point/outpu...   X                   X   -44
/e_fft_8_point/outpu...   X                   X   0
/e_fft_8_point/outpu...   X                   X   -47
/e_fft_8_point/outpu...   X                   X   -13
/e_fft_8_point/outpu...   X                   X   -60
/e_fft_8_point/outpu...   X                   X   -33
/e_fft_8_point/outpu...   X                   X   -141
/e_fft_8_point/outpu...   X                   X   -87
/e_fft_8_point/done       U
/e_fft_8_point/s_out...   0000000001011110    000000000101111
/e_fft_8_point/s_out...   0000000000000000    000000000000000
/e_fft_8_point/s_out...   1111111110100010    111111111010001
Now                       1700 ns             0                500
```

Figure-6: Input and Output of FFT

The simulation result of complete FFT that is the combination of 1st stage, 2nd stage and 3rd stage is as shown in figure-6. This is the combination of 16-bit real number and 16-bit imaginary number. Simulation of all these stages together gives reduction of time, we have reduced components of complex multiplier.

IV. Conclusion

In this paper, Model Sim SE 6.3f version is used to simulate the functionality of Fast Fourier Transform. The multiplier, adders and compliment unit were simulated, synthesized and implemented. As we have seen in this paper, firstly single butterfly is simulated, in which complex multiplier is also simulated and at the end complete FFT is shown. Hence in above result, time taken by each block in hardware language is shown.

References

[1]. Saad Bouguezel, M. Omair Ahmad,"IMPROVED RADIX-4 AND RADIX-8 FFT ALGORITHMS"IEEE. Department of Electrical and Computer Engineering Concordia University 1455 de Maisonneuve Blvd. West Montreal, P.Q., Canada.

[2]. Ali Saidi ," DECIMATION-IN-TIME-FREQUENCY FFT ALGORITHM" Motorola Applied Research, Paging and Wireless Data Group Boynton Beach.

[3]. Rizalafande Che Ismail and Razaidi Hussin "High Performance Complex Number Multiplier Using Booth-Wallace Algorithm" School of Microelectronic Engineering Kolej University Kejuruteraan Utara Malaysia.

[4]. J.G.Proakis and D.G.Manolakis,"Digital Signal Processing, Principles,algorithms and applications" Prentice Hall India Publication.

[5]. J.A.Hidalgo,V.Moreno-Vergara,O.Oballe, "A Radix-8 Multiplier Unit Design For Specfic Purpose",Dept of the Electronica, E.T.S.I.Industriales

[6]. C. S. Burrus and T. W. Parks,"DFT/FFT and Convolution Algorithms", New York,NY : John Wiley,1985.

[7]. A. Saidi , "Generalized FFT algorithm",Proc. ICC, pp.227-231, May 1993.

[8]. Robert Polge and Brooks Lawence, "Comparion of New Multiple Radix Fast fourier Number Theoetic Transform with FFT Algorithm in Terms of Performance and Hardware Cost", ECC Department, University of Alabama Huntsville, AL35899,IEEE : 744-749.

[9]. Keiichi Satoh and Jubee Taba, "Complex Multiplier Suited for FPGA Structure", in Proc.ITC-CSCC, 2008, P.341-344.

[10]. Xilinx Co."Xcell journel vol 58-59",2007 spring.

[11]. Y.N.Chang and K.K.Parhi,"An efficient pipelined FFT architecture,"IEEE Trans. Circuits Systems II, vol. 50, pp.322-352, June 2003.

[12]. Elements of digital signal processing by "Prof.N.SARKAR" Khanna publishers.