FPGA Implementation of Fast Fourier Transform

[1] Vanmathi.K, [2] Sekar.K, [3] Remya Ramachandran

1. Associate Professor, Department of EEE, Hindusthan College of Engineering and Technology, Coimbatore, Tamil Nadu.
2. Assistant Professor, Department of EEE, Hindusthan College of Engineering and Technology, Coimbatore, Tamil Nadu.
3. PG Scholar, Department of EEE, Hindusthan College of Engineering and Technology, Coimbatore, Tamil Nadu. rcremya471@gmail.com

Abstract - The Fast Fourier Transform is one of the most widely used digital signal processing algorithms. It is used to compute the Discrete Fourier Transform and its inverse. As a result, these are widely used for many applications in engineering, science, and mathematics which include areas such as: communications, signal processing, instrumentation, biomedical engineering, numerical methods, sonics and acoustics, and applied mechanics. It is described as the most important numerical algorithm of our lifetime. The number of applications for this transform continues to grow. The Decimation-In-Time radix-2 FFT using butterflies is designed. The butterfly operation is faster. The outputs of the shorter transforms are reused to compute many outputs, thus the total computational cost becomes less. The 16 bit and 32 bit inputs are synthesized using Verilog. The logic utilization obtained from the design summary of 16 and 32 bit radix-2 DIT FFT can be compared. The utilization factor increases as the number of bits increases. The design is developed using hardware description language Verilog on Xilinx 14.2 xc3s500E. The spartan3-tyro plus is used as hardware to implement the complex FFT values.

Keywords – Fast Fourier Transform (FFT), Decimation in Time (DIT), Field Programmable Gate Array (FPGA)

I. INTRODUCTION

A Fast Fourier Transform (FFT) is an algorithm to compute the Discrete Fourier Transform (DFT) and its inverse. A Fourier transform converts time (or space) to frequency and vice versa; an FFT rapidly computes such transformations. As a result, FFTs are widely used for many applications in engineering, science, and mathematics. FFTs have been described as the most important numerical algorithm of our lifetime. The signal processing applications need high throughput, so the aim of the fused elements is to reduce the circuit area first and then to reduce the delay [12]. The reduction of FFT complexity is crucial to minimize the power consumption and the amount of hardware [14].

A new efficient FFT algorithm for OFDM/DMT applications can achieve higher processing rate and better efficiency than the conventional algorithm and is very suitable for the OFDM/DMT applications such as the WLAN, DAB/DVB, and ADSL/VDSL systems [6]. Several communication systems need medium resolution (9–12 bits) analog-to-digital converters (ADCs) with bandwidths in the tens of MHz range [1]. The main reason for its widespread use is the existence of efficient techniques for its computation.

Field Programmable Gate Arrays (FPGAs) are programmable semiconductor devices that are based around a matrix of Configurable Logic Blocks connected through programmable interconnect. The speeds can be very fast, and multiple control loops can run on a single FPGA device at different rates. FPGAs can be programmed to the desired application or functionality requirements. Xilinx FPGAs allow for field upgrades to be completed remotely, eliminating the costs associated with re-designing or manually updating electronic systems. For implementing reconfigurable, industrial electronics systems FPGAs are commonly used. [11]. The FPGA implementation of a common FFT can perform transforms over finite and infinite fields.

This work presents the implementation of the radix-2 Decimation in Time (DIT) FFT in an FPGA using HDL. The design summary of 16 and 32 bit FFT input numbers are compared. The spartan3-tyro plus in the FPGA Spartan 3E family is used as hardware to implement the complex FFT values.

II. RADIX-2 DIT FFT

A. Fast Fourier Transform (FFT)

The proposed method is based on radix-2 FFT computed using butterfly diagrams. The DIT radix-2 FFT recursively partitions a DFT into two half-length DFTs of the even-indexed and odd-indexed time samples. The outputs of these shorter FFTs are reused to compute many outputs, thus greatly

reducing the total computational cost. This FFT is then implemented in FPGA.

Cooley and Tukey first introduced the concept of FFT by efficiently making use of symmetry and periodicity properties of the twiddle factors for its significant computational reduction [2]. The FFT and inverse FFT algorithms are the important processing blocks used for data conversion from time-to-frequency domain (FFT) and from frequency-to-time domain (IFFT) [4]. In most digital circuits, the modulators and demodulators can be implemented using simple FFT blocks. FFT processor's effectiveness plays an important role in optimizing system performance. It is able to perform flexible-size FFTs which a processor is desired [5].

An FFT is a way to compute the DFT more quickly. Computing the DFT of N points in the naive way using the definition takes O(N2) arithmetical operations while a FFT can compute the same DFT in only O(N log N) operations. The difference in speed can be enormous. The computation time improvement is proportional to N / log (N). FFT becomes a common operator for a terminal which supports several standards where many frequency domain algorithms are involved [3]. A Wireless Personal Area Network connection for uncompressed video at a small cost is obtained using the multiband OFDM UWB proposal [10]. There are two types of computational elements called butterfly units. DIT computational elements consist of complex multiplication(s) followed by a sum and difference network. Decimation In Frequency (DIF) computational elements consist of a sum and difference network followed by complex multiplication(s).

The DFT for a sequence of length N is defined by

$$X(k)=\sum_{n=0}^{N-1} x(n)e^{-j2\pi kn/N}, \quad k = 0, \ldots\ldots, N-1 \qquad (1)$$

Where x(0),...., x(N-1) be complex numbers [7].Using the twiddle factor, the DFT can be re-written as

$$X(k) = \sum_{n=0}^{N-1}(x(n)\omega_N^{kn}), \quad k = 0, \ldots\ldots, N-1 \qquad (2)$$

Direct computation of the DFT is basically inefficient because it does not exploit the symmetry and periodicity properties of the phase factor W_N. These properties are [8]:

$$\text{Symmetry property: } \omega_N^{k+N/2} = -\omega_N^k \qquad (3)$$
$$\text{Periodicity property: } \omega_N^{k+N} = \omega_N^k \qquad (4)$$

FFT uses butterfly operations for computations. For a basic radix-2 butterfly, the operation is defined as

$$y_0 = x_0 + x_1\omega^k \qquad (5)$$
$$y_1 = x_0 - x_1\omega^k \qquad (6)$$

where k is an integer depending on the part of the transform being computed [13].An IEEE floating-point adder accepts normalized numbers, supports all four IEEE rounding modes, and outputs the correctly normalized rounded sum/difference in the format required by the IEEE Standard [9].

B. 16 BIT RADIX-2 DIT FFT

The length-16, DIT FFT with the input data scrambled and output data in order of the 16 bit butterfly diagram is shown in Figure 1. The twiddle factor computation is done at each stages of butterfly diagram. This 16 bit radix-2 DIT FFT is designed and simulated in Spartan 3E using Verilog synthesis.

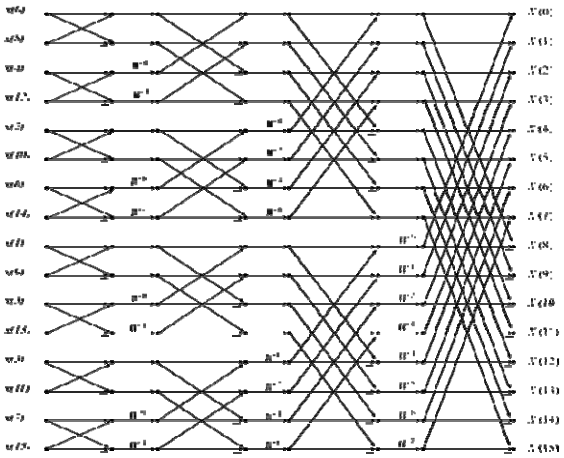

Figure 1: 16 Bit FFT Butterfly Diagram

C. 32 BIT RADIX-2 DIT FFT

The processing of the data is same as in 16 bit radix-2 DIT FFT. The only difference is that the length is 32. The number of butterfly unit for processing also increases. The complexity will increase. This is designed and simulated using Verilog synthesis in Spartan 3E.

III. SYNTHESIS RESULTS AND DISCUSSION

The 16 and 32 bit radix-2 DIT FFT are synthesized in Spartan 3E starter board as the evaluation development board. The family is Spartan 3E, the device used is xc3s500E, the package is FG320 and the speed is -4. The top level source type is HDL, the synthesis tool is XST(VHDL/Verilog), the simulator is Isim(VHDL/Verilog) and the VHDL source analysis standard is VHDL-93.

For 16 bit, the 16 bit binary numbers as 4 words in X0[3:0], X1[3:0], X2[3:0], X3[3:0] are given as the input and after the transformation using Verilog coding, the output is obtained as 4 words in Y0[3:0], Y1[3:0], Y2[3:0], Y3[3:0] consists of 16 bit binary numbers. The transformation is done with the help of butterfly diagrams. The result obtained after simulation in the Isim window is shown in figure 2.

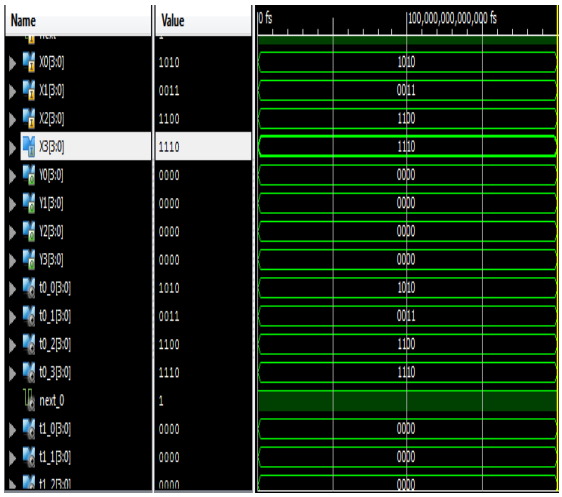

Figure 2: Simulation Output of 16 Bit Input DIT Radix-2 FFT

For 32 bit, the 32 bit binary numbers in x_r[31:0] and x_i[31:0] are given as the input of real and imaginary parts and after the transformation using Verilog coding, the output of real and imaginary parts are obtained in y_r[31:0] and y_i[31:0]. The transformation is done with the help of butterfly diagrams. The result obtained after simulation in the Isim window is shown in figure 3.

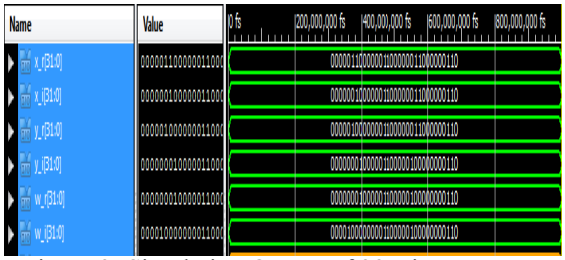

Figure 3: Simulation Output of 32 Bit Input DIT Radix-2 FFT

The 16 bit FFT DIT radix-2 contains 12 stages from stage 0 to stage 11as shown in the RTL schematic of figure 4. Each stage is interlinked. The output of the stage 0 is given as input to the stage 1. Its output is connected as input to stage 2 and so on. At the last stage, stage 11, the output of the whole process is obtained. The resistor transistor logic diagram shows the internal connections of the top dft with clock and reset connected in each stage.

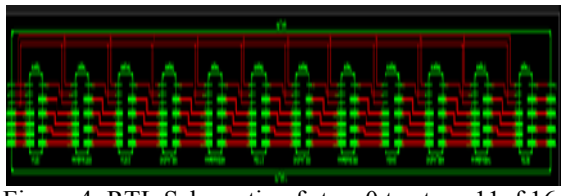
Figure 4: RTL Schematic of stage0 to stage 11of 16 Bit Input DIT Radix-2 FFT

The RTL Schematic of DFT top of 16 Bit Input DIT Radix-2 FFT is shown in figure 5. The inputs are given in the left side and outputs are obtained through the right side after processing the FFT values.

Figure 5: RTL Schematic of dft top of 16 Bit Input DIT Radix-2 FFT

The 16 and 32 bit synthesis results and design summary shows that the utilization factor is more for 32 bit radix-2 DIT FFT than 16 bit radix-2 DIT FFT. Table 1 shows the comparison of 16 and 32 bit logic utilization of xc3s500E for implementing this FFT algorithm. The 32 bit FFT has more logic utilization than 16 bit FFT. Thus as the number of FFT bits for processing increases the logic utilization also increases. The design can be done in this device since the percentage of logic utilization comes within 100%. The design cannot be able to done with the device, if the utilization of the device is more than 100%.

The logic utilization details of 32 bit shows that the number of slices is 8%, slice FF is 6%, 4 input LUTs is 7%, bonded IOBs is 15%, BRAMs is 60%,18*18 SIOs is 80% and Gclks is 4%. The utilization details of 16 bit shows that the number of slices is 6%, slice FF is 4%, 4 input LUTs is 5%, bonded IOBs is 15%, BRAMs is 50%,18*18 SIOs is 60% and Gclks is 4%. The device utilization is related to the complexity of the device. The complexity of the system increases as the device utilization increases. The 16 bit has low complexity with that of 32 bit. So as the number of bits is increasing, the complexity and the device utilization also increases.

Table 1: Comparison of 16 and 32 bit logic utilization

Logic utilization	Used		Available		Utilization	
	16 bit	32 bit	16 bit	32 bit	16 bit	32 bit
No. of slices	285	403	4656	4656	6%	8%
No. of slice FFs	456	624	9312	9312	4%	6%
No. of 4 input LUTs	487	670	9312	9312	5%	7%
No. of bonded IOBs	36	36	232	232	15 %	15 %
No. of BRAMs	10	12	20	20	50 %	60 %
No. of MULT 18*18 SIOs	12	16	20	20	60 %	80 %
No. of GCLKs	1	1	24	24	4%	4%

IV. HARDWARE IMPLEMENTATION

The Spartan-3 EDK Board provides a powerful, self-contained development platform for designs targeting the new Spartan-3 FPGA from Xilinx. It features a 200K gate Spartan-3, on-board I/O devices, and 1MB fast asynchronous SRAM, making it the perfect platform to experiment with any new design, from a simple logic circuit to an embedded processor core.

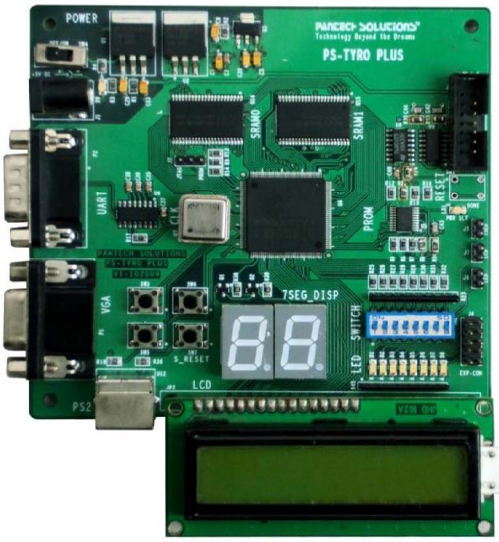

Figure 6: TYRO PLUS SPARTAN3 (EDK) Board Components placement top view

The FFT input complex numbers with real and imaginary parts are feeded in to the Spartan 3 tyro plus kit through coding after resetting the kit. The hardware FPGA Spartan family Spartan 3 tyro plus kit is shown in figure 6 is a 32 bit processor.

```
result is:
k,          Real Part              Imaginary Part
0           7.750000               0.000000
1           2.255817               -1.534301
2           -1.702563              2.314249
3           4.273728               -3.648580
4           -2.700000              2.770000
5           4.305338               -2.315199
6           -4.177437              1.914249
7           3.325116               -1.000920
8           -6.590000              0.000000
9           3.325116               1.000920
10          -4.177437              -1.914249
11          4.305338               2.315199
12          -2.700000              -2.770000
13          4.273728               3.648580
14          -1.702563              1.534301
15          2.255817               6.590000
```

Figure 7: Hardware Implementation Result

The inputs are given as 16 bit complex numbers. After the 12 stages of processing, the result is obtained as shown in figure 7. The output is obtained in personal computer display with real and imaginary parts in it.

V. CONCLUSION

In this work, the design of 16 and 32 bit radix-2 DIT FFT is synthesized using Xilinx. Since FFT is theoretically fast operation, the processing speed of the algorithm is high. The total computational cost of this is low as the shorter ones are reused to compute many outputs. The simulation of 16 and 32 bit radix-2 FFT is done and outputs are obtained in Isim window. The logic utilization details of 32 bit and 16 bit is compared. The logic utilization details of 32 bit shows that the number of slices is 8%, slice FF is 6%, 4 input LUTs is 7%, bonded IOBs is 15%, BRAMs is 60%,18*18 SIOs is 80% and Gclks is 4%. The utilization details of 16 bit shows that the number of slices is 6%, slice FF is 4%, 4 input LUTs is 5%, bonded IOBs is 15%, BRAMs is 50%, 18*18 SIOs is 60% and Gclks is 4%. The utilization factor of xc3s500E for synthesizing DIT radix-2 FFT increases as the number of bits for processing increases. The hardware FPGA in the Spartan family, Spartan 3 tyro plus is used to implement the 16 bit FFT complex values and the result is obtained in the personal computer.

REFERENCES

[1] Ankesh. J, Muthusubramaniam. V and Shanthi. P, "Analysis and Design of a High Speed Continuous-time $\Delta\sum$ Modulator Using the Assisted Opamp Technique", *IEEE*

Journal of Solid-State Circuits, vol. 47, no. 7, pp. 1615-1625, July 2012.

[2] Chang W and Nguyen T.Q, "On the Fixed-Point Accuracy Analysis of FFT Algorithms", *IEEE Transactions on Signal Processing*, vol. 56, no. 10, pp.4673-4682, October 2008.

[3] Ghouwayel. A.A and Louet. Y, "FPGA Implementation of a Re-configurable FFT for Multi-standard Systems in Software Radio Context", *IEEE Transactions on Consumer Electronics*, vol. 55, no. 2, pp. 950-958, May 2009.

[4] Guan. X, Fei. Y, Lin. H, "Hierarchical Design of an Application-Specific Instruction Set Processor for High-Throughput and Scalable FFT Processing", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 20, no. 3, pp. 551-563, March 2012.

[5] Guichang. Z, Fan. X,and Alan. N.W, *"*A Power-Scalable Reconfigurable FFT/IFFT IC Based on a Multi-Processor Ring", *IEEE journal of solid-state circuits,* vol. 41, no. 2, pp. 483-495, Feb. 2006.

[6] Jung Y, Yoon H and Kim J, "New Efficient FFT Algorithm and Pipeline Implementation Results for OFDM/DMT Applications", *IEEE Transactions on Consumer Electronics*, vol. 49, no. 1, pp.14-20, February 2003.

[7] Lee Hand In-Cheol P, "Balanced Binary-Tree Decomposition for Area-Efficient Pipelined FFT Processing*", IEEE Transactions on Circuits and Systems—I: Regular Papers*, vol. 54, no. 4, pp.889-900, April 2007.

[8] Ramesh.B, Ammu. K, Pallavi. V and Meera.V, "Modified FPGA based Design and Implementation of Reconfigurable FFT Architecture", *IEEE symposium,* pp. 818-822, 2013.

[9] Seidel P.M and Even G, "Delay-Optimized Implementation of IEEE Floating-Point Addition", *IEEE Trans. Computers,* vol. 53, no. 2, pp. 97-113, February 2004.

[10] Sherrat R.S, Cadenas O and Goswami N, "A Low Clock Frequency FFT Core Implementation for Multiband Full-Rate Ultra-Wideband (UWB) Receivers", *IEEE Transactions on Consumer Electronics*, vol. 51, no. 3, pp. 798-802, August 2005.

[11] Stephen. M and Roger. W, *"*Power Efficient, FPGA Implementations of Transform Algorithms for Radar-Based Digital Receiver Applications", *IEEE Transactions on Industrial Informatics*, vol. 9, no. 3, pp. 1591-1600, Aug. 2013.

[12] Swartzlander. E.E, and Hani. H.M. Saleh, "FFT Implementation with Fused Floating-Point Operations", *IEEE Transactions on Computers*, vol. 61, no. 2, pp.284-288, February 2012.

[13] Tao. Y, Deyuan. G and Xiaoya. F, "A Multipath Fused Add-Subtract Unit for Digital signal Processing", Proc. *IEEE Midwest Symp. Circuits and Systems*, pp. 448-452, 2012.

[14] Yu-Heng. G.L, and Chien-In. H.C, "Dynamic Kernel Function FFT With Variable Truncation Scheme for Wideband Coarse Frequency Detection", *IEEE Transactions on Instrumentation and Measurement*, vol. 58, no. 5, pp. 1555-1562, May 2009.

K.VANMATHI received B.E and M.E Degrees in the years 1990 and 2004 respectively from PSG College of Technology, Coimbatore, Tamil Nadu, India. She has teaching experience of sixteen years. She is presently working as Associate Professor in the Department of Electrical and Electronics Engineering at Hindusthan college of Engineering and Technology. Her areas of interest are Linear machines, FEA of machines EMI and EMC issues. She has published many papers in National / International Conferences to her credit.

Remya Ramachandran received B.Tech Degree in Electronics and Communication Engineering in the year 2008 from Calicut University, Kerala, India. She is currently doing M.E. Degree in Applied Electronics under Anna University, Chennai, Tamil Nadu, India. She has published many papers in National / International Conferences to her credit. Her areas of interest are Signal Processing, Image Processing, Communication and VLSI.