Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 05, May-2022

Low Power VLSI Implementation of Fast Fourier Transform

Toro Madhura Rahul
M.Tech VLSI Design,
VIT University
Vellore-632014, Tamil Nadu

***Abstract*— Power and speed are the two crucial design parameters in any integrated circuit. With increasing development in the field of VLSI, there is a need of high speed devices which consume less power. This project aims to design such a low power architecture for the FFT implementation. In this project FFT is implemented for a 32-point input sequence with Decimation in Time (DIT) for Radix-2 algorithm. The coding is done in Verilog using the ModelSim 10.5 and is synthesized using Intel Quartus Prime 18.0. ASIC performances for proposed architecture such as area, power, timing are verified using the Synopsys Design Compiler and FPGA performances such as LUTs, Slices, Flipflops are evaluated using the Xilinx ISE 14.7 Design Suite.**

Index Terms – FFT, Vedic Multiplier, Carry Select Adder, ASIC and FPGA, Synopsys DC, Xilinx ISE Design Suite

I. INTRODUCTION

The FFT is employed in the field of Digital Signal Processing (DSP) like filtering, spectral analysis and so on. The FFT plays vital role Digital communication such as digital video broadcasting, Orthogonal Frequency Division Multiplexing (OFDM) [11]. It is an algorithm to compute Discrete Fourier Transform (DFT) [11]. The Discrete Fourier Transform (DFT) is one of the most common operations in the field of signal processing. The algorithm for computing the DFT effectively is known as Fast Fourier Transform. FFT is used to transform a signal from time domain to frequency domain.

Discrete implementation of DFT has O(N2) complexity, which reduces to O(Nlog2N) through FFT. The Cooley-Tukey algorithm, usually called the Fast Fourier Transform, is a collection of algorithms for quicker calculation of the DFT. The FFT transforms a waveform into a series of sines and cosines at each frequency present in the original signal [7]. The Cooley-Tukey algorithm is also known as the Radix-2 algorithm and was shortly followed by the Radix-3, Radix-4, and Mixed Radix algorithms.

If an N – point DFT is implemented directly, the requirement of arithmetic operations is of the order of O(N2) that is N2 multiplications and N (N-1) additions. The order of arithmetic operations for FFT is O(NlogN) i.e. NlogN additions and (N/2)logN multiplications. Thus FFT implementation of DFT. is preferred [6]. FFT is made of complex addition and multiplication. Depending on inputs being real or complex, the structures of adders and multipliers are built. One of the key role in this calculation is the MAC unit. This unit consists of adders and multipliers.

II. LITERATURE SURVEY

A. DFT

The Discrete Fourier Transform is a very import a mathematical tool which is used for discrete-time signal processing. It is used to convert time domain signal into frequency domain. DFT can be used to compute the Z-transform for evenly spaced points on a circle for the given sequence. If the given sequence is finite then the DFT is used as the transform. DFT has many applications in digital signal processing such as linear filtering, spectrum analysis etc. DFT is very important to perform Fourier analysis in several applications [3]. The DFT is given as

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

where, k = 0, 1, 2, 3, ……N-1.
Here W_N is called as Twiddle Factor and it is expressed as the complex value phase factor, which is equal to Nth root of unity and this twiddle factor is given as:

$$W_N = e^{-2j\frac{n}{N}}$$

Hence X (k) can be written as:

$$X(k) = \sum_{n=0}^{N-1} x[n] W_N^{nK} \quad ; \quad 0 \le n \le N-1$$

There are lot of calculations in DFT and hence it requires a time efficient algorithm. Instead of using DFT if we use FFT algorithm [3] such as Decimation in Time (DIT) FFT with radix-2 algorithm, then there is reduction in the number of complex multiplications and additions to $\frac{N}{2}$log2N and Nlog2N respectively in order to calculate the DFT of a given sequence, x[n].

B. FFT

The Fast Fourier transform (FFT) is a efficient algorithm to compute the Discrete Fourier Transform (DFT). FFT algorithm uses the concept of divide and conquer approach and it is done by decomposing the computation of DFT into smaller sequences of DFTs. This approach is useful in many areas but its direct calculation requires more time. The FFT

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 05, May-2022

algorithm is classified into two: Decimation In Time (DIT) and Decimation In Frequency (DIF) algorithms. In decimation in time, the input sequence is divided into smaller sequences and the outputs are combined in a specific pattern to obtain the intended DFT output. In the decimation in frequency approach, the frequency samples of the DFT are divided into smaller sub-sequences in a certain manner. DIT and DIF algorithms are further classified as: Radix-2, Radix-4, Radix-8, Split-Radix [3].

The decimation-in-time (DIT) radix-2 FFT recursively partitions a DFT into two DFTs that have length half of the total length of sequence such as even-indexed and odd-indexed time samples. The outputs of the shorter FFTs are used again to compute other outputs, thus greatly reducing the total computational cost [8].

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

The N-point data sequence is splitted into two $\frac{N}{2}$point data sequences F1(k) and F2(k), corresponding to the even-numbered and odd-numbered samples of x(n) respectively [4].

Since F1(k) and F2(k) are periodic, with period $\frac{N}{2}$, we have $F1\left(K+\frac{N}{2}\right)$= F1(k) and $F2(k+\frac{N}{2})$= F2(k). In addition, the factor $W_N^{k+\frac{N}{2}} = -WN_N^k$. Hence, we can write:

$$X(k) = F1(k) + F2(k)W_N^k \;;\; k = 0,1,2\ldots., \frac{N}{2}-1$$

$$X\left(k+\frac{N}{2}\right) = F1(k) - F2(K)W_N^k \;;\; k = 0,1,2,\ldots., \frac{N}{2}-1$$

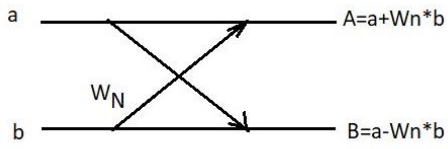

Fig.1. Basic Butterfly in DIT FFT Algorithm

Here we can see that the basic computational method is to take two complex numbers, i.e., the pair (a, b). Then do the multiplication of "b" and W_N, and then add and subtract the product value from "a" to form two new complex numbers (A, B). This basic calculation is called a butterfly [4].

C. Radix-2 FFT

The name Radix-2 is given because of its base is equals to 2 and the representation is 2^M, here M represents the stage. A 32-points FFT can be computed in 5 stages where each stage consists of 16 butterfly operations. Butterfly operation in a Radix-2 algorithm is that part of FFT computation, that provides the Fourier transform of two-point sequence in a simplified manner [3].

In this case 32-points DFT can be divided into sixteen 2-point DFTs. Then these can be combined to get eight 4-point DFTs. Further they are combined to get four 8-point DFTs. These are combined to get two 16-point DFTs. Finally, we get 32-point DFT.

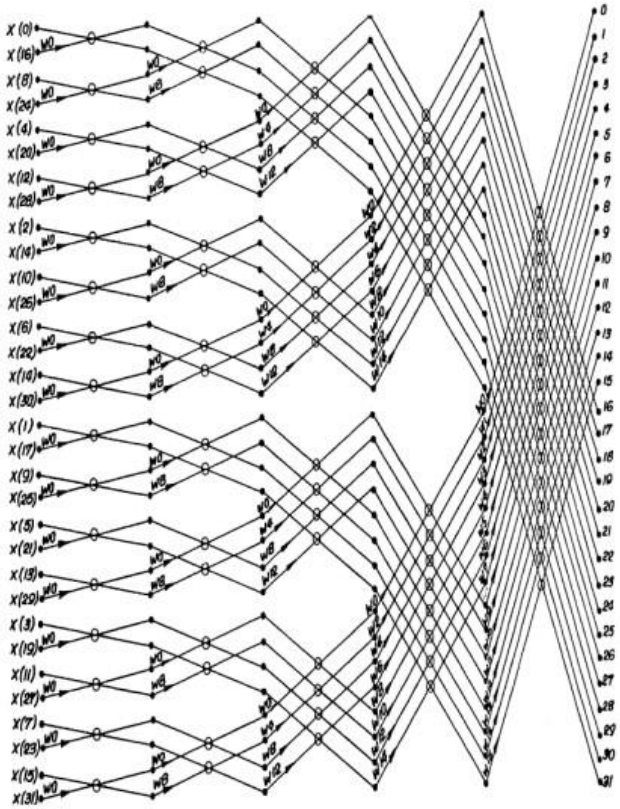

Fig.2. 32-Point DIT FFT with Radix-2

III. VEDIC MULTIPLIER

The Vedic Multiplier is a high speed multiplier when compared to conventional multiplier [1]. The Vedic multiplier is applied in all types of number schemes. Consider two 2-bit binary numbers (a1, a0) and (b1, b0). These two numbers are multiplied. The results after multiplication process of binary numbers give 4-bit of output. Vedic mathematics has 16 formulae for performing arithmetic calculation. Urdhva Tiryakbahayam Sutra is used for the multiplication. Its literal means "Vertical and Cross-wise" which enhances the speed of multiplication operation [4].

Algorithm:

Step-1: Multiply Least Significant Bit (LSB) of the multiplicand and the multiplier vertically, which gives the final outcome of the LSB.

Step-2: Multiply the LSB of multiplicand with the MSB of multiplier and multiply the MSB of multiplicand with the LSB of the multiplier (crosswise). Finally add the products. This step gives the second bit of final output.

Step-3: Multiply the MSB of the multiplicand and the multiplier(vertically). The product is added to the previous carry to achieve the step 2 additional process. The sum and

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 05, May-2022

carry that are obtained, are considered as the third and fourth bit of the final output. The Fig.3. shows the exact multiplication operation for 2-bit numbers.

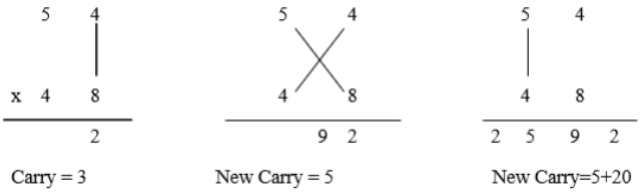

Fig.3. Multiplication of numbers 54 and 48.

E. Reversible Logic

A circuit that is able to reverse its output to get back its original input is termed as reversible logic circuit. In this logic there is no information loss, since all the information is present within the circuit and only need to be reversed, to get it recovered. Hence a reversible circuit is used to reduce the power dissipation. Reversible logics provide 1-1 mapping between input output combination making easy to retrace the input from output. There are specific gates designed called as reversible gates designed from logic gates, which has same number of input and output. Examples of Reversible gates: CNOT, Feynman, Peres, HNG, TR etc. The basic requirement of a reversible circuit is to have a one-to-one correspondence between inputs and outputs. Also, each input state must correspond to a unique output state. If the above requirement is fulfilled then an inverse circuit can be designed to make the circuit reversible.

Reversible Gates:

Peres Gate: It is a 3x3 reversible entryway which means, its 3 sources of input and 3 outputs. The output is given as X=A, Y=AxorB, Z=ABxorC. Peres gate has the Quantum cost of five.

Feymen Gate: Feynman gate is a 2x2 reversible gate. The input vector is I (A, B) and the output vector is O(P, Q) and outputs are defined by P=A, Q=A · B. Quantum cost of a Feynman gate is 1**.**

HNG Gate: It is a 4x4 bit gate and its quantum cost is six. The input vectors are I= (A, B, C, D) and output vectors are O= A,B,A(XOR)B(XOR),C, A(XOR)B).C(XOR)AB(XOR)D. It is used for designing full adders. HNG gate produces sum and carry in the same and thereby reduces the gate count and garbage outputs.

IV. DESIGN ARCHITECTURE & SYNTHESIS

A. 16-Bit VM

The proposed 16x16 Vedic multiplier is easily implemented using 8x8 Vedic multiplier and 16-bit Ripple Carry Adder. This makes the design of this proposed 16x16 multiplier very simple and efficient compare to conventional multipliers. All the multipliers are made using Peres, Feymen reversible gates. And RCA is design using HNG gates.

B. Carry Select Adder

In the proposed architecture, carry select adder is used instead of simply adding two 16-bit numbers. This helps to reduce the area and speed of calculation.

C. MAC Unit

This butterfly diagram is named as MAC in the design architecture. This unit consists of two complex inputs and twiddle factor. It computes complex addition and multiplication of the inputs and gives the output. • This unit is the basic butterfly unit that computes the addition and multiplication of the inputs.

- This unit consists of multipliers and adders.
- The inputs to this unit consist of complex input-1, input-2, complex twiddle input.
- The outputs consist of complex output-1, output-2.
- The obtained internal design of this unit is given in Fig.5.

D. Register

Registers are temporary storage locations that hold data. It is necessary to store the values of output after each stage.\\
There are two registers, at the input side and output side in order to save the inputs and outputs respectively. The registers have size of 32x2x8, 32 no. of inputs x 2(real and imaginary) x 8-bits.

E. MUX

MUX is used to select one input out of many inputs. In this design, we have 32-bit input which means there will be 5 stages of FFT operation. Hence the MUX used here is a 5x1. In stage-1, it selects inputs from register. The outputs of MAC are given as inputs to the MUX. Depending upon the stage, it calculates the output. After all stages output is calculated, the output is stored in the register.

Design:

The architecture is designed using two register blocks, multiplexer, MAC units.

- Inputs that will be entered are given to the Register-1 and saved.
- The MUX is used to select the inputs which goes to the MAC units. The outputs of the MAC units are given to the MUX until last stage output is obtained.
- The control unit is used to give the select signal to the MUX.
- The final output is saved in Register-2 and final output is obtained from this register.

The 16 butterfly units are used to compute outputs for all the stages.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 05, May-2022

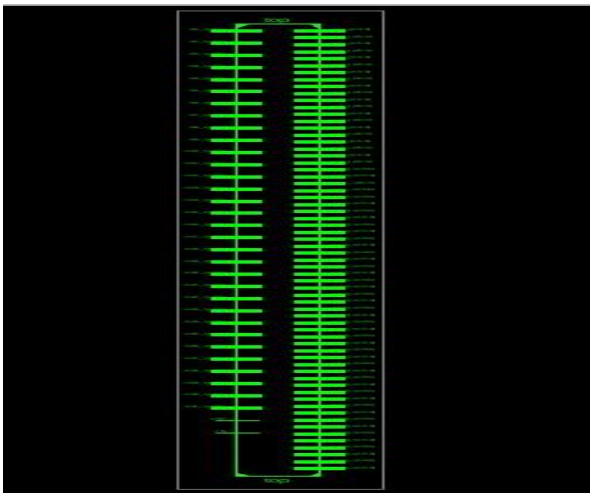

Fig.4. Top RTL

V. SIMULATION AND RESULTS

The performance of the proposed architecture was evaluated by ASIC and FPGA performances. The architecture has been implemented using Verilog language. Modelsim 10.5 tool is used to write a Verilog code and verifying the timing diagram. Xilinx ISE 14.7 Design Suite is used for evaluating FPGA performances like LUT, flipflop, slices, and frequency. Synopsys Design Compiler is used to calculate ASIC performance such as power, area, timing.

A. ASIC PERFORMANCE

The area, power, delay, and ADP is computed for this architecture. This proposed architecture output has been taken from both 32 nm technology. It is observed that the total power consumption in this architecture is 2.64 uW. The power report is shown in Fig.5.

The timing report is given in Fig.6. The data arrival time is 1.26 and data required time is 10.82. Hence slack has been met.

The area report is given in Fig.7. Total area required is 29684.8 unit sq.

B. FPGA PERFORMANCE

The performance parameters such as LUTs, the number of flip flops, slices, and operating frequency for FPGA device such as Vertex-6 Xc6vcx75t has been observed. Also the device utilization is observed. The following Fig.8. shows the device utilization.

```
  Cell Internal Power  = 216.8338 uW   (74%)
  Net Switching Power  =  76.7049 uW   (26%)
                         ---------
Total Dynamic Power    = 293.5387 uW  (100%)

Cell Leakage Power     =   2.3544 mW


                 Internal         Switching         Leakage           Total
Power Group      Power            Power             Power             Power   (   %    )
--------------------------------------------------------------------------------------
  (   0.00%)
  (   0.00%)
  (   0.00%)
  (  24.44%)
  (  64.21%)
  (   0.00%)
  (  11.35%)
--------------------------------------------------------------------------------------
Total          216.8338 uW       76.6962 uW     2.3544e+09 pW     2.6480e+03 uW
1
```

Fig.5. ASIC Power Report

```
Point                                                   Incr       Path
------------------------------------------------------------------------
clock clk2 (rise edge)                                  0.00       0.00
clock network delay (ideal)                             1.00       1.00
clkk/counter_reg_0_/CLK (DFFARX1_RVT)                   0.00       1.00 r
clkk/counter_reg_0_/Q (DFFARX1_RVT)                     0.12       1.12 f
clkk/U3/Y (NOR2X0_RVT)                                  0.07       1.19 r
clkk/U5/Y (AO22X1_RVT)                                  0.06       1.25 r
clkk/counter_reg_1_/D (DFFARX1_RVT)                     0.01       1.26 r
data arrival time                                                  1.26

clock clk2 (rise edge)                                 10.00      10.00
clock network delay (ideal)                             1.00      11.00
clock uncertainty                                      -0.15      10.85
clkk/counter_reg_1_/CLK (DFFARX1_RVT)                   0.00      10.85 r
library setup time                                     -0.03      10.82
data required time                                                10.82
------------------------------------------------------------------------
data required time                                                10.82
data arrival time                                                 -1.26
------------------------------------------------------------------------
slack (MET)                                                        9.56
```

Fig.6. ASIC Timing Report

```
Number of ports:                        15562
Number of nets:                         18307
Number of cells:                         8424
Number of combinational cells:           5108
Number of sequential cells:              1559
Number of macros/black boxes:               0
Number of buf/inv:                       2545
Number of references:                     103

Combinational area:                10631.097854
Buf/Inv area:                       4147.884354
Noncombinational area:             10972.921535
Macro/Black Box area:                 0.000000
Net Interconnect area:             8080.877623

Total cell area:                   21604.019389
Total area:                        29684.897012
```

Fig.7. ASIC Area Report

Target FPGA	Slice	LUT	Flipflop
Virtex6 Xc6vcx75t	4781/93120	9953/46560	1383/13351
Utilization (%)	5	21	10

Fig.8. FPGA Device Utilization

```
Timing Summary:
---------------
Speed Grade: -2

   Minimum period: 5.576ns (Maximum Frequency: 179.340MHz)
   Minimum input arrival time before clock: 2.627ns
   Maximum output required time after clock: 0.659ns
   Maximum combinational path delay: No path found
```

Fig.9. FPGA Timing Report

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 11 Issue 05, May-2022

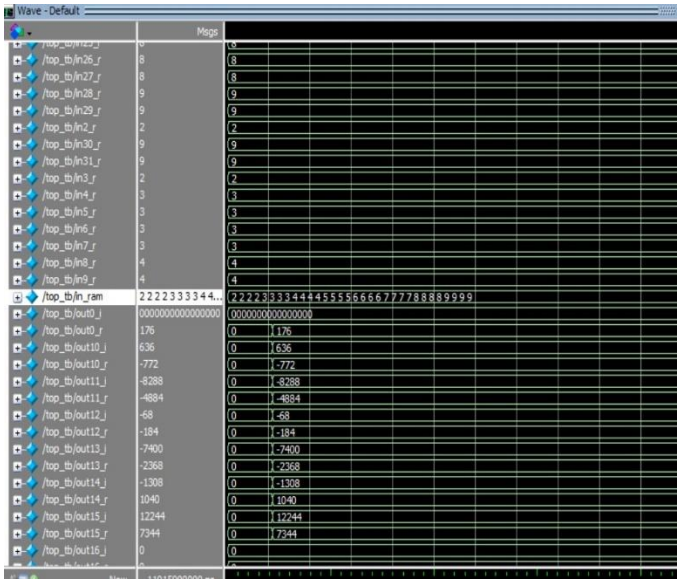

Fig.10. Simulation wave

VI. CONCLUSION

The proposed architecture has been implemented in Xilinx software by using Verilog code. Without using any normal digital adder, the FFT architecture has been designed which requires less access time and less area. In FPGA implementation, LUT, slices, flip flops, and frequency are improved in proposed architecture. ASIC 32 nm technology. The proposed architecture provides better FPGA and ASIC performance when compare to conventional methods.

In future work, Distributed Arithmetic based architecture and architecture level optimization can be used to further reduce the ASIC and FPGA results.

VII. REFERENCES

[1] Kausar Ali and Paresh Rawat. Vlsi architecture for fft using radix-2 butterfly of complex valued data. In 2017 International Conference on Computer Commu nication and Informatics (ICCCI), pages 1–5. IEEE, 2017.

[2] Asmita Haveliya. Design and simulation of 32-point fft using radix-2 algorithm for fpga implementation. In 2012 Second International Conference on Advanced Computing & Communication Technologies, pages 167–171. IEEE, 2012.

[3] Sudhanshu Mohan Khare and M Zahid Alam. Fpga based design and simulation of 32-point fft through radix-2 Dit algorithm.

[4] P Koti Lakshmi, B Santhosh Kumar, and Rameshwar Rao. Implementation of Vedic multiplier using reversible gates. In Computer Science Conference Proceedings in Computer Science & Information Technology (CS & IT) series, volume 5, pages 125–134, 2015.

[5] Weidong Li. Studies on implementation of low power FFT processors. Citeseer, 2003.

[6] Nooshin Mahdavi, Rozita Teymourzadeh, and Masuri Bin Othman. Vlsi imple- mentation of high speed and high resolution fft algorithm based on radix 2 for dsp application. In 2007 5th Student Conference on Research and Development, pages 1–4. IEEE, 2007.

[7] SIVA KUMAR Palaniappan. Design and implementation of radix-4 fast Fourier transform in ASIC chip with 0.18 µm standard CMOS technology. PhD thesis, MS thesis, Malaysia, 2008.

[8] Mayura Patrikar and Vaishali Tehre. Design and power measurement of different points fft using radix-2 algorithm for fpga implementation. In 2017 International conference of Electronics, Communication and Aerospace Technology (ICECA), volume 1, pages 190–195. IEEE, 2017.

[9] T Prabakaran and Mr R Arun Sekar. High performance reversible Vedic multiplier using cadence 45nm technology.

[10] S Josue Saenz, Susana Ortega Cisneros, Jorge Rivera Dominguez, et al. Fpga de- sign and implementation of radix-2 fast fourier transform algorithm with 16 and 32 points. In 2015 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC), pages 1–6. IEEE, 2015.

[11] Avinash Kumar Singh and Ashutosh Nandi. Design of radix 2 butterfly structure using Vedic multiplier and cla on xilinx. In 2017 Conference on Emerging Devices and Smart Systems (ICEDSS), pages 120–123. IEEE, 2017.

[12] Rajasekhar Turaka et al. Low power vlsi implementation of real fast fourier trans- form with dram-vm-cla. Microprocessors and Microsystems, 69:92–100, 2019.

[13] BV Uma, Harsha R Kamath, S Mohith, V Sreekar, and Shravan Bhagirath.Area and time optimized realization of 16 point fft and ifft blocks by using ieee 754 single precision complex floating point adder and multiplier. In 2015 International Conference on Soft Computing Techniques and Implementations (ICSCTI), pages 99 104. IEEE, 2015.