Implementation of FFT Architecture using Various Adders

Mallika Verma, Ankur Bhardwaj

Abstract: In 1965 a technique called Fast Fourier Transform (FFT) was invented to find the Fourier Transform. This paper compares three architectures, the basic architecture/ non-reduced architecture of FFT, decomposed FFT architecture without retiming and decomposed FFT architecture with retiming. In each case, the adder used will be Ripple Carry Adder (RCA) and Carry Save Adder (CSA). A fast Fourier transform (FFT) calculates the discrete Fourier transform (DFT) or the inverse (IDFT) of a sequence. Fourier analysis transforms a signal from time to frequency domain or vice versa. One of the most burgeoning use of FFT is in Orthogonal Frequency Division Multiplex (OFDM) used by most cell phones, followed by the use in image processing. The synthesis has been carried out on Xilinx ISE Design Suite 14.7. There is a decrease in delay of 0.824% in Ripple Carry Adder and 6.869% in Carry Save Adder, further the reduced architecture for both the RCA and CSA architectures shows significant area optimization (approximately 20%) from the non-reduced counterparts of the FFT implementation.

Keywords: Area optimization, CSA, Decomposition, DFT, FFT, IDFT, Non- Reduced, OFDM, RCA

I. INTRODUCTION

The efficient and structured computation for the purpose of reducing the hardware requirements has been the field of interest of the designer from a very long time. Fourier transform converts a signal from time or space domain to frequency domain and vice versa. FFT is a way to evaluate the same result of DFT promptly, where DFT takes O(N^2)arithmetic operations for computation, FFT takes, only O(N log N) operations. The difference in speed can be humongous, especially where N may be in the thousands or millions. The Discrete Fourier Transform (DFT) deals with a finite set of data and can be implemented in computers by algorithms or even dedicated hardware. DFT has been utilised across numerous fields such as image processing, radar, voice processing, data compression and sonar systems.[1] The Fast Fourier Transform is an algorithm to find the DFT and IDFT. It produces the identical output preciselyas computing the DFT, however FFT is much faster. [2] Cooley-Tukey algorithm, most commonly used, is a divide and conquer algorithm which breaks a DFT into smaller DFTs having twiddle factors.

Manuscript published on 30 August 2019.
***Correspondence Author(s)**

Mallika Verma, Electronics and Communication, Jaypee Institute of Information Technology, Noida, Uttar Pradesh, India. Email: contact.mallikaverma@gmail.com
Ankur Bhardwaj, Electronics and Communication, Jaypee Institute of Information Technology, Noida, Uttar Pradesh, India. Email: ankur.bhardwaj@jiit.ac.in

The algorithm of two types, first - time-based (DIT) and, second - frequency-based (DIF) Fast Fourier Transform. Order of data in DIT FFT isfrom bit reversal in input to normal order in output, whereas DIF FFT is converse. In Radix-2 algorithm, a N point FFT is continuously split into smaller parts till two point FFT is obtained.[3]

II. OVERVIEW OF FFT ARCHITECTURE, RE-TIMING AND ADDERS USED

In Fast Fourier transform, a butterfly puts together the results of smaller DFTs into a larger DFT, or does the inverse. The name "butterfly" comes from the structure of the data-flow representation in the radix-2 case.[4]

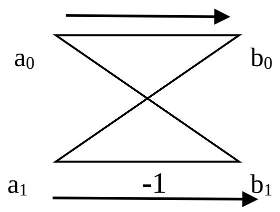


Figure 1: Signal Flow Graph with input a and output b

According to radix-2 Cooley–Tukey algorithm, the output- b0 and b1 can be written in terms of inputs- a0 and a1 by using the following equations:
b0 = a0 + a1
b1= a0 – a1
Where twiddle factors are not included.
Considering twiddle factor Wnk = $e^{2\pi ik/n}$, the output b0 and b1 are:
b0 = a0+ a1Wnk
b1= a0 – a1Wnk
Where k is an integer which depends on the part of the transform being computed.

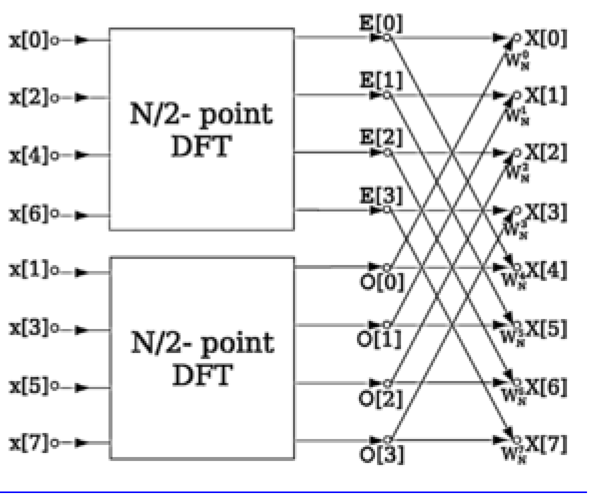


Figure 2: ADIT radix-2 FFT

The adders used in butterfly computation are Ripple Carry Adder and Carry Save Adder. A carry save adder calculates the sum of three or more n bit numbers in binary. It produces two outputs – Sum and Carry, a sequence of partial sum bits and a sequence of partial carry bits respectively.[5,6] On the other hand, a ripple carry adder takes two inputs, A0 and B0 with Cin initially as 0 for a full adder, the carry generated is fed as an input to the next full adder with inputs A1 and B1. The carry gets rippled till n stages and the final Cout along with the sum generated in each stage is the final output. As each carry is fed as input to the next full adder or is rippled, it is called a Ripple Carry Adder.[6]

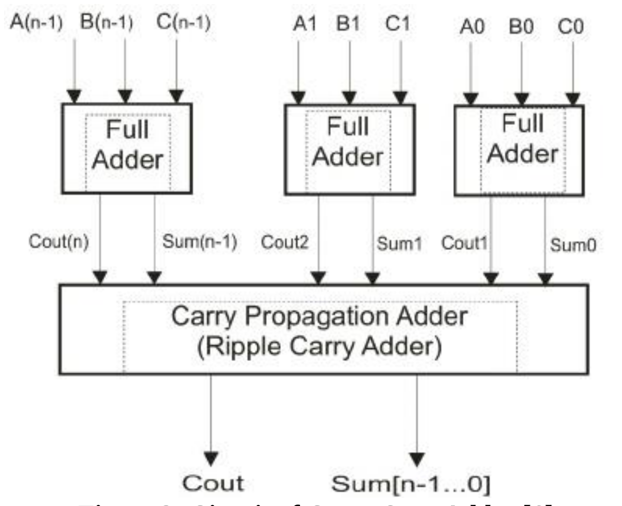


Figure 3: Circuit of Carry Save Adder [6]

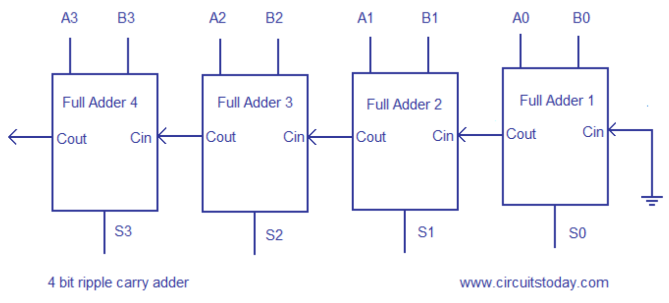


Figure 4: Circuit of Ripple Carry Adder [6]

Re-timing is used to shift the memory elements, latches or registers, in such a way that it's purpose remains the same in the circuit, thereby improving area, power characteristics and performance.[7] Re-timing is frequently used to minimise the clock period by searching for the minimum workable period by binary search. Further, it can be used to change a given synchronous circuit into a more effective circuit keeping different cost criteria in mind.[8]

III. SUGGESTED FFT ARCHITECTURE AND IMPLEMENTATION

Three FFT architectures build on Cooley-Tukey algorithm have been implemented.

A. *Basic Butterfly / Non- Reduced Architecture*

By using the divide and conquer approach, radix-2 FFT is obtained. Splitting the sequence x[n] into sequences A1 and A2, each of length $\frac{N}{2}$,

A1[n] = x[2n], samples are even

A2[n]= x[2n + 1], samples are odd

For n = 0 to (N/2 – 1)

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

$$= \sum_{\substack{n=0 \\ n\ even \\ so\ n=2m}}^{N-1} x[n]W_N^{kn} + \sum_{\substack{n=0 \\ n\ odd \\ so\ n=2m+1}}^{N-1} x[n]W_N^{kn} \quad (2)$$

$$= \sum_{m=0}^{\frac{N}{2}-1} x[2m]\, W_N^{k2m} + \sum_{m=0}^{\frac{N}{2}-1} x[2m+1]\, W_N^{k(2m+1)} \quad (3)$$

$$= \sum_{m=0}^{\frac{N}{2}-1} s1[m]W_{N/2}^{km} + W_N^k \sum_{\substack{n=0 \\ n\ odd \\ so\ n=2m+1}}^{N-1} s2[m]W_N^{km} \quad (4)$$

Because of recursion property

$$S1[k] = \sum_{n=0}^{\frac{N}{2}-1} s1[n]\, W_{N/2}^{kn} \text{ and } S2[k] = \sum_{n=0}^{\frac{N}{2}-1} s2[n]\, W_{N/2}^{kn} \quad (5)$$

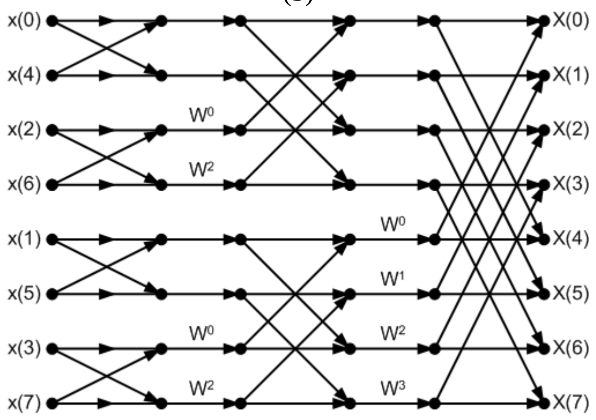


Figure 5: Signal flow graph of 8 point DIT-FFT

Output of each node is calculated by placing RCA and CSA at each step of node calculation to find the ultimate output.

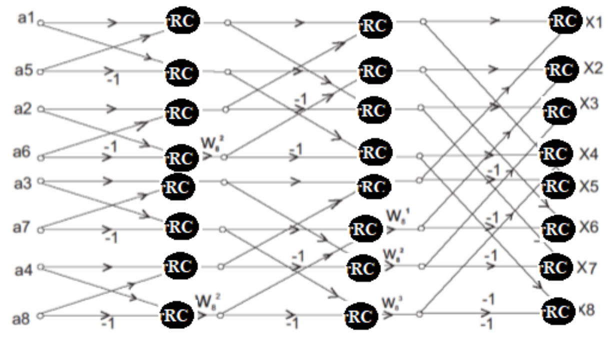


Figure 6: Signal flow graph of radix-2, 8 point DIT-FFT with Ripple Carry Adder

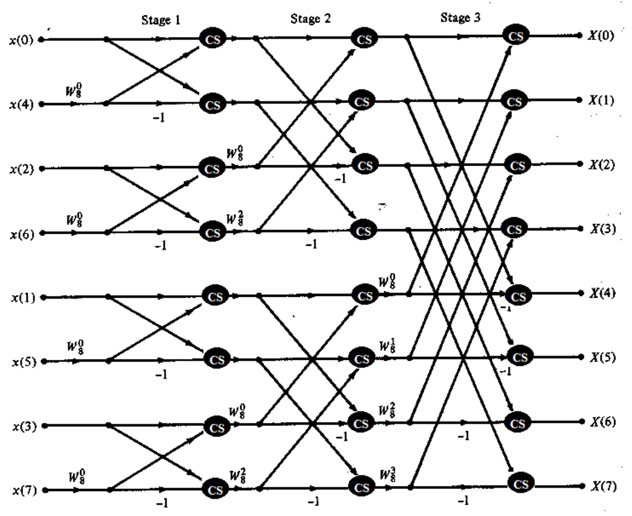


Figure 7: Signal flow graph of radix-2, 8 point DIT-FFT with Carry Save Adder

B. *(i) Reduced FFT Architecture without retiming*

Let us take a 8-point real valued sequence,
x(n)={x0 , x1 , x2 , x3 , x4, x5 , x6 , x7}
Where x(n) can be expressed as,
x(n)={m1, m2, m3, m4, m5, m6, m7, m8}
Using the equations of DFT,we obtain the DFT of x(n) as X(K),
Where
X(K)={X1, X2, X3, X4, X5, X6, X7, X8}
The architecture presented [9] reduces power and the area is optimized.
The outputs are Figure 6 are mapped are follows:
X1= X1R_{eal}+iX1$I_{maginary}$, and X1$I_{maginary}$=0, every time
X2= X2R_{eal}+iX2$I_{maginary}$,
X3= X3R_{eal}+iX3$I_{maginary}$,
X4= X4R_{eal}+iX4$I_{maginary}$,
X5= X5R_{eal} +iX5$I_{maginary}$, and X5$I_{maginary}$=0, every time
X6=X6R_{eal}+iX6$I_{maginary}$,
Where X6R_{eal}=X4R_{eal}, and X6$I_{maginary}$= -X4$I_{maginary}$
X7=X7R_{eal}+iX7$I_{maginary}$,
Where X7R_{eal}= -X3R_{eal}, and X7$I_{maginary}$ = -X3$I_{maginary}$,
And similarly,
X8=X8R_{eal}+iX8$I_{maginary}$,
Where X8R_{eal}=X2R_{eal}, and X8$I_{maginary}$= -X2$I_{maginary}$
Where twiddle factor X is: ±0.707 at consecutive stages

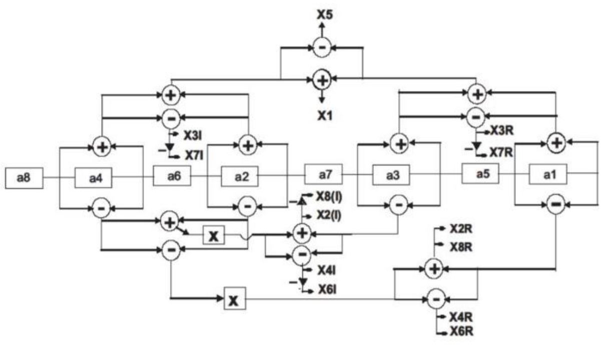


Figure 8: Without retiming: Decomposed FFT architecture, 8 point.[9]

The architecture shown in Fig. 8, is further improved by using RCA and CSA.

(ii) Reduced Architecture with Retiming

A clock is fed to the state machine where in a single butterfly computes all the stages of the butterfly architecture.

Using an overlapped structure, re-arranging the input and recreating the stage a state machine is designed whose input values, computed values and output values are stored in a register. The state machine is made such that it is input controlled so that we can manipulate the inputs being fed to the butterfly to get the respective output without increasing the structural complexity. In the following architecture clock, reset, state_machine_start and inputs are fed to obtain the fast fourier transform of the desired inputs. The architecture has a single butterfly structure which performs all the operations, thereby decreasing delay.

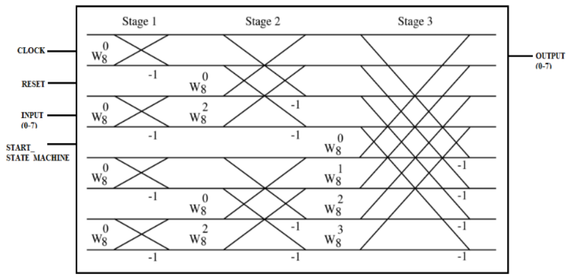


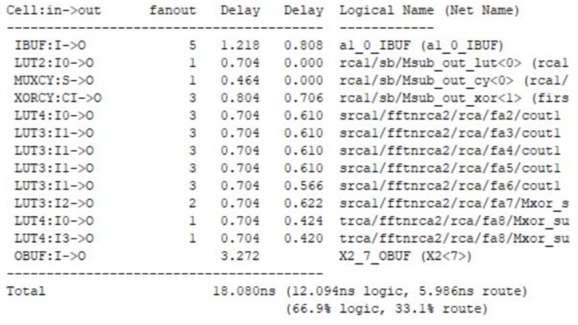
Figure 9: Digital circuit for FFT Architecture with Re-Timing

IV. RESULTS AND DISCUSSIONS

A. *Basic Butterfly/ Non-Reduced Architecture*

1. RCA

Table I: Delay



```
                              Gate    Net
Cell:in->out       fanout    Delay   Delay  Logical Name (Net Name)
----------------------------------------  ------------
 IBUF:I->O              5    1.218   0.808  a1_0_IBUF (a1_0_IBUF)
 LUT2:I0->O             1    0.704   0.000  rca1/sb/Msub_out_lut<0> (rca1
 MUXCY:S->O             1    0.464   0.000  rca1/sb/Msub_out_cy<0> (rca1/
 XORCY:CI->O            3    0.804   0.706  rca1/sb/Msub_out_xor<1> (firs
 LUT4:I0->O             3    0.704   0.610  srca1/fftnrca2/rca/fa2/cout1
 LUT3:I1->O             3    0.704   0.610  srca1/fftnrca2/rca/fa3/cout1
 LUT3:I1->O             3    0.704   0.610  srca1/fftnrca2/rca/fa4/cout1
 LUT3:I1->O             3    0.704   0.610  srca1/fftnrca2/rca/fa5/cout1
 LUT3:I1->O             3    0.704   0.566  srca1/fftnrca2/rca/fa6/cout1
 LUT3:I2->O             2    0.704   0.622  srca1/fftnrca2/rca/fa7/Mxor_s
 LUT4:I0->O             1    0.704   0.424  trca/fftnrca2/rca/fa8/Mxor_su
 LUT4:I3->O             1    0.704   0.420  trca/fftnrca2/rca/fa8/Mxor_su
 OBUF:I->O                   3.272          X2_7_OBUF (X2<7>)
----------------------------------------
Total                        18.080ns (12.094ns logic, 5.986ns route)
                                      (66.9% logic, 33.1% route)
```

Table II: Logic Utilization

Logic Utilization	Used	Available	Utilization
No. of slices	142	4656	3%
No. of 4 input LUTs	261	9312	2%

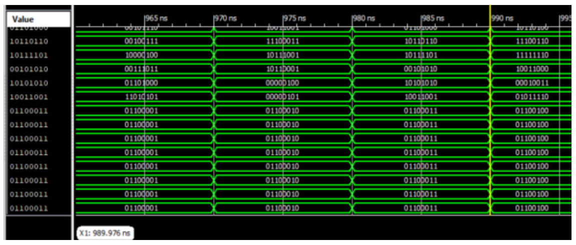


Figure 10: Simulation Results

2. CSA

Table III: Delay

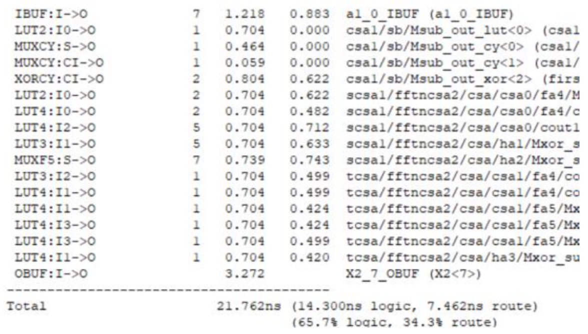

```
 IBUF:I->O              7   1.218   0.883  al_0_IBUF (al_0_IBUF)
 LUT2:I0->O             1   0.704   0.000  csal/sb/Msub_out_lut<0> (csal
 MUXCY:S->O             1   0.464   0.000  csal/sb/Msub_out_cy<0> (csal/
 MUXCY:CI->O            1   0.059   0.000  csal/sb/Msub_out_cy<1> (csal/
 XORCY:CI->O            2   0.804   0.622  csal/sb/Msub_out_xor<2> (firs
 LUT2:I0->O             2   0.704   0.622  scsal/fftncsa2/csa/csa0/fa4/M
 LUT4:I0->O             2   0.704   0.482  scsal/fftncsa2/csa/csa0/fa4/c
 LUT4:I2->O             5   0.704   0.712  scsal/fftncsa2/csa/csa0/coutl
 LUT3:I1->O             5   0.704   0.633  scsal/fftncsa2/csa/hal/Mxor_s
 MUXF5:S->O             7   0.739   0.743  scsal/fftncsa2/csa/ha2/Mxor_s
 LUT3:I2->O             1   0.704   0.499  tcsa/fftncsa2/csa/csal/fa4/co
 LUT4:I1->O             1   0.704   0.499  tcsa/fftncsa2/csa/csal/fa4/co
 LUT4:I1->O             1   0.704   0.424  tcsa/fftncsa2/csa/csal/fa5/Mx
 LUT4:I3->O             1   0.704   0.424  tcsa/fftncsa2/csa/csal/fa5/Mx
 LUT4:I3->O             1   0.704   0.499  tcsa/fftncsa2/csa/csal/fa5/Mx
 LUT4:I1->O             1   0.704   0.420  tcsa/fftncsa2/csa/ha3/Mxor_su
 OBUF:I->O                  3.272          X2_7_OBUF (X2<7>)
----------------------------------------
Total                      21.762ns (14.300ns logic, 7.462ns route)
                                    (65.7% logic, 34.3% route)
```

Table IV: Logic Utilization

Logic Utilization	Used	Available	Utilization
No. of slices	175	4656	3%
No. of 4 input LUTs	320	9312	3%

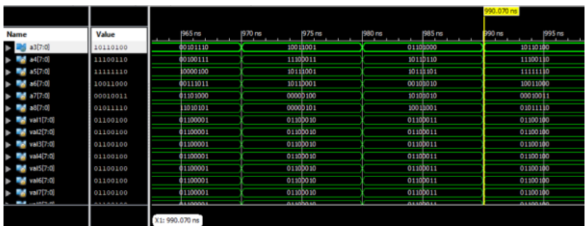


Figure 11: Simulation Results

B. *Reduced FFT Architecture Without Re-Timing*

1. RCA

Table V: Delay

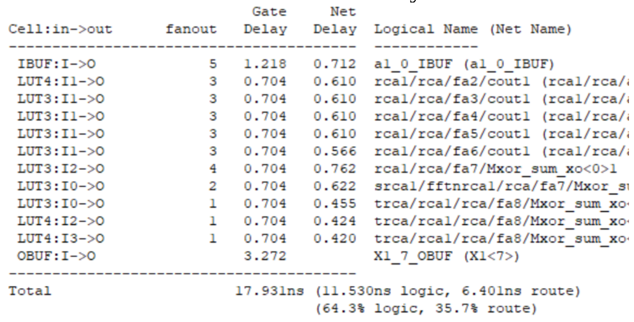

```
                                Gate     Net
Cell:in->out      fanout   Delay    Delay   Logical Name (Net Name)
----------------------------------------  ------------
 IBUF:I->O              5   1.218   0.712  al_0_IBUF (al_0_IBUF)
 LUT4:I1->O             3   0.704   0.610  rcal/rca/fa2/coutl (rcal/rca/
 LUT3:I1->O             3   0.704   0.610  rcal/rca/fa3/coutl (rcal/rca/
 LUT3:I1->O             3   0.704   0.610  rcal/rca/fa4/coutl (rcal/rca/
 LUT3:I1->O             3   0.704   0.610  rcal/rca/fa5/coutl (rcal/rca/
 LUT3:I1->O             3   0.704   0.566  rcal/rca/fa6/coutl (rcal/rca/
 LUT3:I2->O             4   0.704   0.762  rcal/rca/fa7/Mxor_sum_xo<0>1
 LUT3:I0->O             2   0.704   0.622  srcal/fftnrcal/rca/fa7/Mxor_s
 LUT3:I0->O             1   0.704   0.455  trca/rcal/rca/fa8/Mxor_sum_xo
 LUT4:I2->O             1   0.704   0.424  trca/rcal/rca/fa8/Mxor_sum_xo
 LUT4:I3->O             1   0.704   0.420  trca/rcal/rca/fa8/Mxor_sum_xo
 OBUF:I->O                  3.272          X1_7_OBUF (X1<7>)
----------------------------------------
Total                      17.931ns (11.530ns logic, 6.401ns route)
                                    (64.3% logic, 35.7% route)
```

Table VI: Logic Utilization

Logic Utilization	Used	Available	Utilization
No. of slices	115	4656	2%
No. of 4 input LUTs	213	9312	2%

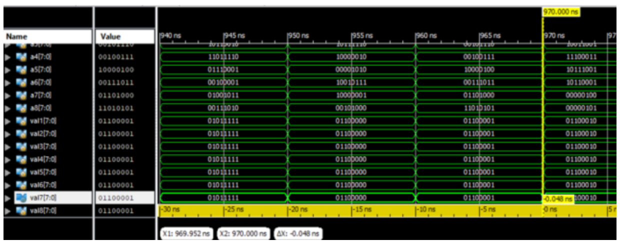


Figure 12: Simulation

2. CSA

Table VII: Delay

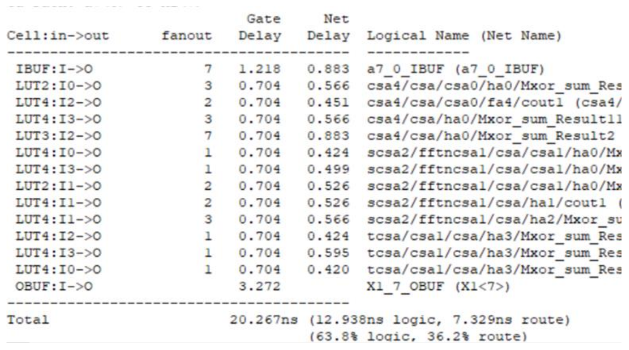

```
                                Gate     Net
Cell:in->out      fanout   Delay    Delay   Logical Name (Net Name)
----------------------------------------  ------------
 IBUF:I->O              7   1.218   0.883  a7_0_IBUF (a7_0_IBUF)
 LUT2:I0->O             3   0.704   0.566  csa4/csa/csa0/ha0/Mxor_sum_Res
 LUT4:I2->O             2   0.704   0.451  csa4/csa/csa0/fa4/coutl (csa4/
 LUT4:I3->O             3   0.704   0.566  csa4/csa/ha0/Mxor_sum_Result1l
 LUT3:I2->O             7   0.704   0.883  csa4/csa/ha0/Mxor_sum_Result2
 LUT4:I0->O             1   0.704   0.424  scsa2/fftncsal/csa/csal/ha0/Mx
 LUT4:I3->O             1   0.704   0.499  scsa2/fftncsal/csa/csal/ha0/Mx
 LUT2:I1->O             2   0.704   0.526  scsa2/fftncsal/csa/csal/ha0/Mx
 LUT4:I1->O             2   0.704   0.526  scsa2/fftncsal/csa/hal/coutl (
 LUT4:I1->O             3   0.704   0.566  scsa2/fftncsal/csa/ha2/Mxor_su
 LUT4:I2->O             1   0.704   0.424  tcsa/csal/csa/ha3/Mxor_sum_Res
 LUT4:I3->O             1   0.704   0.595  tcsa/csal/csa/ha3/Mxor_sum_Res
 LUT4:I0->O             1   0.704   0.420  tcsa/csal/csa/ha3/Mxor_sum_Res
 OBUF:I->O                  3.272          X1_7_OBUF (X1<7>)
----------------------------------------
Total                      20.267ns (12.938ns logic, 7.329ns route)
                                    (63.8% logic, 36.2% route)
```

Table VIII: Logic Utilization

Logic Utilization	Used	Available	Utilization
No. of slices	144	4656	3%
No. of 4 input LUTs	266	9312	2%

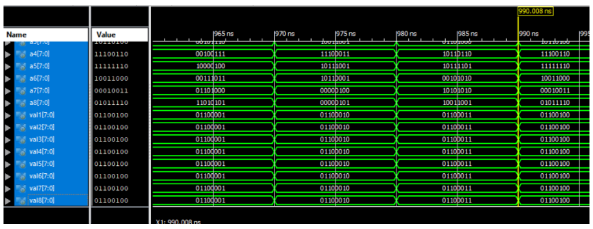
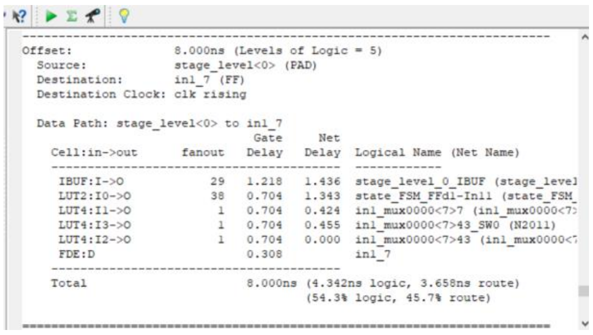


Figure 13: Simulation

C. *Reduced FFT Architecture With Re-Timing*

1. RCA

Table IX: Delay



```
--------------------------------------------------------------------------------
Offset:              8.000ns (Levels of Logic = 5)
  Source:            stage_level<0> (PAD)
  Destination:       inl_7 (FF)
  Destination Clock: clk rising

  Data Path: stage_level<0> to inl_7
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O            29   1.218   1.436  stage_level_0_IBUF (stage_level
     LUT2:I0->O           38   0.704   1.343  state_FSM_FFd1-In11 (state_FSM_
     LUT4:I1->O            1   0.704   0.424  inl_mux0000<7>7 (inl_mux0000<7>
     LUT4:I3->O            1   0.704   0.455  inl_mux0000<7>43_SW0 (N2011)
     LUT4:I2->O            1   0.704   0.000  inl_mux0000<7>43 (inl_mux0000<7
     FDE:D                     0.308          inl_7
    ----------------------------------------
    Total                      8.000ns (4.342ns logic, 3.658ns route)
                                       (54.3% logic, 45.7% route)

================================================================================
```

Table X: Logic Utilization

Logic Utilization	Used	Available	Utilization
No. of slice flip-slops	83	9312	1%
No. of 4 input LUTs	292	9312	3%
No. of occupied slices	155	4656	3%

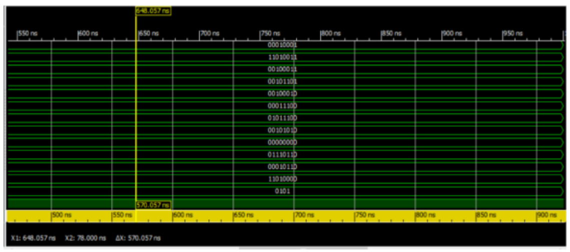
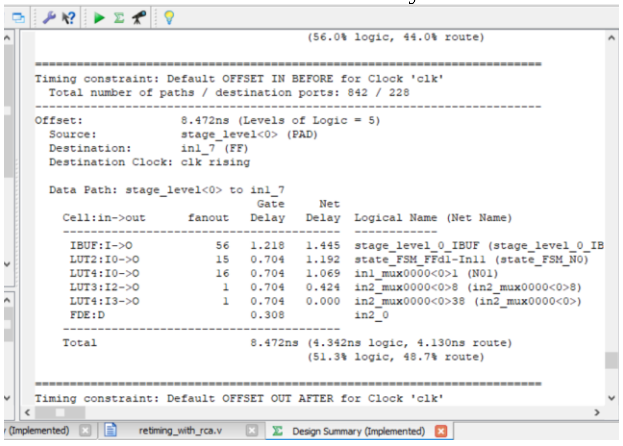


Figure 14: Simulation

2. CSA

Table XI: Delay



```
                                       (56.0% logic, 44.0% route)

================================================================================
Timing constraint: Default OFFSET IN BEFORE for Clock 'clk'
  Total number of paths / destination ports: 842 / 228
--------------------------------------------------------------------------------
Offset:              8.472ns (Levels of Logic = 5)
  Source:            stage_level<0> (PAD)
  Destination:       inl_7 (FF)
  Destination Clock: clk rising

  Data Path: stage_level<0> to inl_7
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     IBUF:I->O            56   1.218   1.445  stage_level_0_IBUF (stage_level_0_IB
     LUT2:I0->O           15   0.704   1.192  state_FSM_FFd1-In11 (state_FSM_N0)
     LUT4:I0->O           16   0.704   1.069  inl_mux0000<0>1 (N01)
     LUT3:I2->O            1   0.704   0.424  in2_mux0000<0>8 (in2_mux0000<0>8)
     LUT4:I3->O            1   0.704   0.000  in2_mux0000<0>38 (in2_mux0000<0>)
     FDE:D                     0.308          in2_0
    ----------------------------------------
    Total                      8.472ns (4.342ns logic, 4.130ns route)
                                       (51.3% logic, 48.7% route)

================================================================================
Timing constraint: Default OFFSET OUT AFTER for Clock 'clk'
```

Table XII: Logic utilization

Logic Utilization	Used	Available	Utilization
No. of slice flip-slops	90	9312	1%
No. of 4 input LUTs	312	9312	3%
No. of occupied slices	175	4656	3%

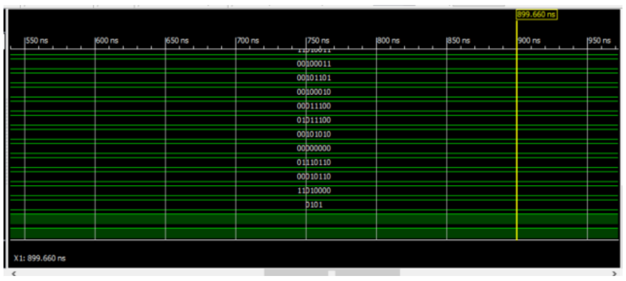


Figure 15: Simulation

V. CONCLUSION

A Verilog HDL code has been successfully simulated and synthesized on Xilinx ISE Design suite 14.7 for 8-point FFT with and without retiming.

From the synthesis report is can be observed that there is a significant decrease in delay for FFT architecture with re-timing, while comparing with basic FFT Architecture and decomposed FFT Architecture without re-timing, for both RCA(55.752% decrease) and CSA(61.069% decrease). However, in terms of area, it increases for RCA and remains the same for CSA when compared with basic FFT architecture and increases when compared with decomposed FFT Architecture without Re-Timing.

On the other hand, the decomposed FFT architecture without re-timing shows a significant decrease in terms of area (approximately 20%) for both RCA and CSA and a decrease in delay of 0.824% for RCA and 6.869% for CSA when compared with basic FFT architecture. The synthesis has been carried out on Xilinx SPARTAN 3 Series of FPGA with XC3S500E Device series. The results are favourable in terms of area and delay.

Table XIII: Percentage Increase/Decrease in area and delay while comparing basic architecture and reduced architecture without re-timing

Comparison Basis	Ripple Carry Adder	Carry Save Adder
Number of slices	19.014% Decrease	17.714% Decrease
Number of 4 input LUTs	18.391% Decrease	16.875% Decrease
Delay	0.824% Decrease	6.869% Decrease

Table XIV: Percentage Increase/Decrease in area and delay while comparing basic architecture and reduced architecture with re-timing

Comparison Basis	Ripple Carry Adder	Carry Save Adder
Number of slices	9.155% Increase	No increase or decrease
Number of 4 input LUTs	11.877% Increase	2.5% Decrease
Delay	55.752% Decrease	61.069% Decrease

ACKNOWLEDGMENT

It is author's privilege to express the most sincere gratitude to Mr. Ankur Bhardwaj, Department of Electronics and Communications, Jaypee Institute of Information Technology, Noida, under whose supreme and enlightened guidance the author has been able to accomplish this work. The author gratefully acknowledge her sincere gratitude to his illuminating guidance, whole-hearted co-operation right from beginning to its write-up and completion.

REFERENCES

1. Xie Yan-lin. "Design and Implementation of a scalable pipeline FFT Processor Based on FPGA" D.Xi Dian University, 2007.
2. V. Mangaiyarkarasi, Dr. C.Kumar Charlie Paul, "Performance analysis between Radix2, Radix4, Mixed Radix4-2 and Mixed Radix8-2 FFT", 2nd International Conference on Current Trends in Engineering and Technology, ICCTET'14
3. Chen, Y., Lin, Y.W. and Lee, C.Y., "A block scaling FFT/IFFT processor for WiMax applications," in Proc. 2nd IEEE Asian Solid-State Circuits Conf., Hangzhou, China, Nov.2006, pp. 203-206.
4. Alan V. Oppenheim, Ronald W. Schafer, and John R. Buck, Discrete-Time Signal Processing, 2nd edition (Upper Saddle River, NJ: Prentice Hall, 1989).
5. R.Mahalakshmi , Dr.T.Sasilatha, "A power efficient carry save adder and modified carry save adder using CMOStechnology", 2013 IEEE International Conference on Computational Intelligence and Computing Research, Enathi, 2013, pp. 1-5.
6. S. Nagaraj, G. M. S. Reddy and S. A. Mastani, "Analysis of different Adders using CMOS, CPL and DPL logic," 2017 14th IEEE India Council International Conference (INDICON), Roorkee, 2017, pp. 1-6.
7. Saxena, P., Pan, P., & Liu, C. L, "The retiming of single-phase clocked circuits containing level-sensitive latches". Proceedings Twelfth International Conference on VLSI Design, Goa, India, 1999, pp. 402-407
8. C. E. Leiserson and J. B. Saxe, "Retiming synchronous circuitry," Algorithmica, vol. 6, nos. 1–6, pp. 5–35, Jun. 1991.
9. Shashidhara. K. S, H.C. Srinivasaiah, "Low Power and Area efficient FFT architecture through decomposition technique", International Conference on Computer Communication and Informatics (ICCCI -2017), Jan. 05 – 07, 2017, Coimbatore, INDIA.

AUTHORS PROFILE

Mallika Verma was born at Delhi, India on July 26, 1997. She received her B. Tech degree from Jaypee Institute of Information Technology, Sector 62, Noida, Uttar Pradesh (June, 2019) and currently is pursuing her Masters degree in International Business from Queensland University of Technology, Brisbane, Australia.

Ankur Bhardwaj was born at Meerut, Uttar Pradesh, India on January 04, 1990. He received B.Tech degree from Gautam Buddh Technical University, Lucknow (June 2011), M.Tech degree from DTU Delhi (2013) and currently pursuing PhD degree from Jaypee Institute of Information Technology, Noida. He joined Jaypee Institute of Information Technology, Noida, in July 2013 as a lecturer. Presently he is an Assistant Professor in the Department of Electronics and Communication Engineering, Jaypee Institute of Information Technology, Sector 62, Noida.