International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-5, January 2020

Implementation of 64-Bits Radix - 8 IFFT for Computation Speed by IDIF using Verilog

B. Anil Kumar, M. Naveen Reddy, Vamshi Kollipara, B. Rajesh

Abstract—Always technical designers choice includes algorithms, flowcharts, programming etc and the end users requires given input and application output. Based upon this view this paper focus on the advancement of Inverse Fast Fourier Transform(IFFT) by doing design and observing the performance analysis of 64 point IFFT, using Radix-8 algorithm. The algorithm is developed by Inverse Decimation In Frequency(IDIF) of IFFT, using Verilog as design entity and synthesis are performed in Xilinx. In this architecture the numbers of stages are reduced to 75%.

Index Terms-IFFT, IDIF, Verilog, XILINX

I. INTRODUCTION

A Inverse Fast Fourier transform (IFFT) is an efficient algorithm to compute the Inverse discrete Fourier transform (IDFT) and its inverse. There are many distinct IFFT algorithms involving a wide range of mathematics, from simple complex number arithmetic to group theory and number theory.

A IDFT decomposes a sequence of values into components of different frequencies. This operation is useful in many fields (see discrete Fourier transform for properties and applications of the transform) but computing it directly from the definition is often too slow to be practical. An IFFT is a way to compute the same result more quickly: computing a IDFT of N points in the naive way, using the definition, takes N(N-1), N^2 arithmetical operations, while an IFFT can compute the same result in only $N\log_8 N$, $N/2\log_8 N$ operations. The difference in speed can be substantial, especially for long data sets where N may be in the thousands or millions—in practice, the computation time can be reduced by several orders of magnitude in such cases, and the improvement is roughly proportional to N / log(N).

Manuscript published on January 30, 2020.
* Correspondence Author
B. Anil Kumar*, Assistant Professor,Dept of ECE,Malla Reddy Institute Of Engineering And Technology,Hyd.,TS, India.(Email: banilkmr301@gmail.com)
M.Naveen Reddy, B.Tech Student, Dept of ECE, Malla Reddy Institute Of Engineering And Technology, Hyd., TS, India (Email: maddirala.naveenreddy22@gmail.com)
VamshiKollipara, Assistant Professor,Dept of ECE,Malla Reddy Institute Of Engineering And Technology,Hyd.,TS, India..(Email: vamshiversatile@gmail.com)
B.Rajesh, Student, B.Tech Student, Dept of ECE, Malla Reddy Institute Of Engineering And Technology, Hyd., TS,India (Email: rajesh14111998@gmail.com)

This huge improvement made many IDFT based algorithms practical; IFFTs are of great importance to a wide variety of applications, from digital signal processing and solving partial differential equations to algorithms for quick multiplication of large integers.

The most well known IFFT algorithms depend upon the factorization of N, but there are IFFTs with O(N log N) complexity for all N, even for prime N. Many IFFT algorithms only depend on the fact that $e^{-2\prod j/N}$ is anth primitive root of unity, and thus can be applied to analogous transforms over any finite field, such as number theoretic transforms. Since the inverse IDFT is the same as the IDFT, but with the

opposite sign in the exponent and a 1/N factor, any IFFT algorithm can easily be adapted for it.

II. LITERATURE SURVEY

J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex Fourier series.

Mathematics of Computation, 1965. A fast algorithm for computing the Discrete Fourier Transform . (Re)discovered by Cooley &Tukey in 19651 and widely adopted there after has a long and fascinating history.Explained the design of pipelined structure from Radix-8 IFFT with IDIF algorithm using efficient butterfly structure. The different and dedicated structures for the 64 bit-width pipelined radix-8 IDIF butterfly structure are implemented. The main goal of this paper is to minimize the number of real multipliers of the architectures. This is done by varying the structure of the complex multipliers and applying them into the butterflies. These structures are widely used in fast and low power multiplier architectures. In pipelined Radix-8 IFFT structures have been developed with the help of Feed forward structures. Feed forward structure provides 16ns for performing 8-point IFFT.

III. EQUATIONS

The main reason for going with Inverse Fast Fourier transform is to reduce the complexity and make mathematical calculations easier compared to that of IDFT.The formulae what we use in IDFT fails for higher complex stages where the calculations become unperformable, So here in IFFT we go for higher complex stages with reduced number of calculations and complexity. The main difference between the calculations performed in IDFT and IFFT is logarithm.

In this paper we are trying to implement the IFFT using DIF with log base 8i.e.,Radix 8 structure implementing 64 bits of data.

General Equations

M-1

$X(K)=\sum x(n)WN$ ---------- (1)

N=0

N/2-1 N-1

$X(K)=\sum x(n)Wnkn+\sum x(n)Wnkn$ -----(2)

n=0 n=N/2

IV.BLOCK DIAGRAM

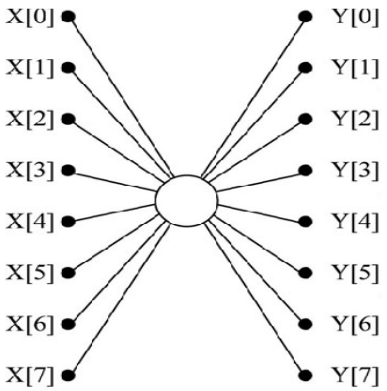


Figure 2 Butterfly Structure of RADIX-8

In the radix 8 IFFT first we align the data and it is given to the input buffer which controls the data and performs the distribution property which allows data to perform their operations in a sequence .The next block refers to the processing element i.e.,Radix 8 Butterfly structure ,Where the number of stages reduced to 75%.The next and main block of the structure is control signal which performs data control and processing control.The next block COEF ROM allows the processing element t perform the calculations and store the result temporarily for future years.

DESCRIPTION

Compare to that of radix 2 and radix 4 here in this paper we perform radix 8 operations by using the twiddle factors. The radix 8 butterfly structure helps us to carry out the complex calculations in the easier way.

In the butterfly structure of Radix 8 point IFFT we have 2 stages.The number of stages are obtained by reducing the complexity using the IFFT.The number of stages are obtained as follows below.

CALCULATIONS

To find the number of stages mathematically the equation is

n=logN8

N= number of samples

n= number of stages

n=log864

n=2log88

so,number of stages(n)=2.

S.no	Number of complex additions	Number of complex multiplication
N=64	IDFT N(N-1)=64(64-1) =4032	IDFT N^2=64*64=4096
N=64	IFFT $N\log_8 N=128\log_8 128$ =298.66	IFFT $N/2\log_8 N=128/2\log_8 128$ =149.33
N=128	IDFT N(N-1)=128(128-1) =16,256	IDFT N^2=128*128=16384
N=128	IFFT $N\log_8 N=128\log_8 128$ =298.66	IFFT $N/2\log_8 N=128/2\log_8 128$ =149.33

From the above table we can conclude that ,InIDFT for the higher stages the complexity increases where in IFFT the complexity is reduced .

V.PROCESS OF DECIMATION

First step of decimation is splitting a sequence in a smaller sequences.A sequence of 64 Bit can be splitted in 8 sequences of 4 blocks.Here on the first stage carries out the butterfly operation by applying the twiddle factor.

In the second stage the output from the 64 Bit IFFT is split into sequence of 8 equal parts.and the initial 32Bits are performed by addition and twiddle factor multiplication and then followed 32 Bits perform subtraction and multiplied with twiddle factor.

Twiddle factors :

$W_N^K = e^{-j(2\Pi/N)K}$

For 64 points,the Twiddle factor is represented as,

N=64,

$W_8^0 = e^{-j(2\Pi/8)0} = 1$

$W_8^1 = e^{-j(2\Pi/8)1} = 0.7-0.7j$

$W_8^2 = e^{-j(2\Pi/8)2} = -j$

$W_8^3 = e^{-j(2\Pi/8)3} = -0.7-0.7j$

$W_8^4 = e^{-j(2\Pi/8)4} = -1$

$W_8^5 = e^{-j(2\Pi/8)5} = -0.7+0.7j$

$W_8^6 = e^{-j(2\Pi/8)6} = j$

$W_8^7 = e^{-j(2\Pi/8)7} = 0.7+0.7j$

$W_8^8 = e^{-j(2\Pi/8)8} = 1$

$W_8^9 = e^{-j(2\Pi/8)9} = 0.7-0.7j$

$W_8^{10} = e^{-j(2\Pi/8)10} = -j$

$W_8^{12} = e^{-j(2\Pi/8)12} = -1$

$W_8^{14} = e^{-j(2\Pi/8)14} = j$

$W_8^{15} = e^{-j(2\Pi/8)15} = 0.7+0.7j$

$W_8^{16} = e^{-j(2\Pi/8)16} = 1$

$W_8^{18} = e^{-j(2\Pi/8)18} = -j$

$W_8^{20} = e^{-j(2\Pi/8)20} = -1$

$W_8^{21} = e^{-j(2\Pi/8)21} = -0.7+0.7j$

$W_8^{24} = e^{-j(2\Pi/8)24} = 1$

$W_8^{25} = e^{-j(2\Pi/8)25} = 0.7-0.7j$

$W_8^{28} = e^{-j(2\Pi/8)28} = -1$

$W_8^{30} = e^{-j(2\Pi/8)30} = j$

$W_8^{35} = e^{-j(2\Pi/8)35} = -0.7-0.7j$

$W_8^{36} = e^{-j(2\Pi/8)36} = -1$

$W_8^{42} = e^{-j(2\Pi/8)42} = -j$

$W_8^{49} = e^{-j(2\Pi/8)49} = 0.7-0.7j$

Inputs at stage1:-

112,48,80,26,30,34,62,126,120,-5.656+64.707j,88j,-16.968+1.414j,-22,-60.802-1.414j,-54j,83.426-1.414j,116,52j,-84,-20j,26,90j-58,-122j,124,-42.42+1.414j,-92j,19.796+1.414j,-18,57.974-1.414j,50j,-80.598-1.414j,97.592-0.707j,58j,8j,41.032-0.707j,-98,-88.076+0.707j,-24+68j,40.076-48.076j,-124,66+56j,-54.662-0.707j,42j,86.662-0.707j,40,60j,76.968+0.707j,40+60j,76.968+0.707j,-52j,-99.592+39.592j,21.408-0.707j,-69j,-44.032-0.707j,-40+61j,5.248+0.707j,-88j,-37.248+0.707j,-64+56j,14,57.49-0.707j,-24-70j,-105.49-0.707j,-16

OUTPUTS AT STAGE 1:

56,56,56,56,-40,24,-8,56,64,64,64,64,-40,24,-8,56,60,60,60,60,-40,24,-8,56,68,68,68,68,-40,24,-8,56,58,58,58,58,-40,24,-8,56,66,66,66,66,-40,24,-8,56,62,62,62,62,-40,24,-8,56,70,70,70,70,-40,24,-8,56

INPUTS AT STAGE 2:

56,56,56,56,-40,24,-8,56,64,64,64,64,-40,24,-8,56,60,60,60,60,-40,24,-8,56,68,68,68,68,-40,24,-8,56,58,58,58,58,-40,24,-8,56,66,66,66,66,-40,24,-8,56,62,62,62,62,-40,24,-8,56,70,70,70,70,-40,24,-8,56

OUTPUTS AT STAGE 2:

x(n)=0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63

Bit Reversal process of 64 bit samples

Normal Data		Bit reversal Data
x(0)=000000	-	x(0)=000000
x(1)=000001	-	x(32)=100000
x(2)=000010	-	x(16)=010000
x(3)=000011	-	x(48)=110000
x(4)=000100	-	x(8)=001000
x(5)=000101	-	x(40)=101000
x(6)=000110	-	x(24)=011000
x(7)=000111	-	x(56)=111000
x(8)=001000	-	x(4)=000100
x(9)=001001	-	x(36)=100100
x(10)=001010	-	x(20)=010100
x(11)=001011	-	x(52)=110100
x(12)=001100	-	x(12)=001100
x(13)=001101	-	x(44)=101100
x(14)=001110	-	x(28)=011100
x(15)=001111	-	x(60)=111100
x(16)=010000	-	x(2)=000010
x(17)=010001	-	x(34)=100010
x(18)=010010	-	x(18)=010010
x(19)=010011	-	x(50)=110010
x(20)=010100	-	x(10)=001010
x(21)=010101	-	x(42)=101010
x(22)=010110	-	x(26)=011010
x(23)=010111	-	x(58)=111010
x(24)=011000	-	x(6)=000110
x(25)=011001	-	x(38)=100110
x(26)=011010	-	x(22)=010110
x(27)=011011	-	x(54)=110110
x(28)=011100	-	x(14)=001110
x(29)=011101	-	x(46)=101110
x(30)=011110	-	x(30)=011110
x(31)=011111	-	x(62)=111110
x(32)=100000	-	x(1)=000001
x(33)=100001	-	x(33)=100001
x(34)=100010	-	x(17)=010001
x(35)=100011	-	x(49)=110001
x(36)=100100	-	x(9)=001001
x(37)=100101	-	x(41)=101001
x(38)=100110	-	x(25)=011001
x(39)=100111	-	x(57)=111001
x(40)=101000	-	x(5)=000101
x(41)=101001	-	x(37)=100101
x(42)=101010	-	x(21)=010101
x(43)=101011	-	x(53)=110101
x(44)=101100	-	x(13)=001101
x(45)=101101	-	x(45)=101101
x(46)=101110	-	x(29)=011101
x(47)=101111	-	x(61)=111101
x(48)=110000	-	x(3)=000011
x(49)=110001	-	x(35)=100011
x(50)=110010	-	x(19)=010011
x(51)=110011	-	x(51)=110011
x(52)=110100	-	x(11)=001011
x(53)=110101	-	x(43)=101011
x(54)=110110	-	x(27)=011011
x(55)=110111	-	x(59)=111011
x(56)=111000	-	x(7)=000111
x(57)=111001	-	x(39)=100111
x(58)=111010	-	x(23)=010111
x(59)=111011	-	x(55)=110111
x(60)=111100	-	x(15)=001111
x(61)=111101	-	x(47)=101111
x(62)=111110	-	x(31)=011111
x(63)=111111	-	x(63)=111111

Discrete Fourier Transform, or simply referred to as DFT, is the algorithm that transforms the time domain signals to the frequency domain components. DFT, as the name suggests, is truly discrete; discrete time domain data sets are transformed into discrete frequency representation. In simple terms, it establishes a relationship between the time domain representation and the frequency domain representation. Fast Fourier Transform, or IFFT, is a computational algorithm that reduces the computing time and complexity of large transforms. IFFT is just an algorithm used for fast computation of the IDFT.

Applications of IFFT and IDFT

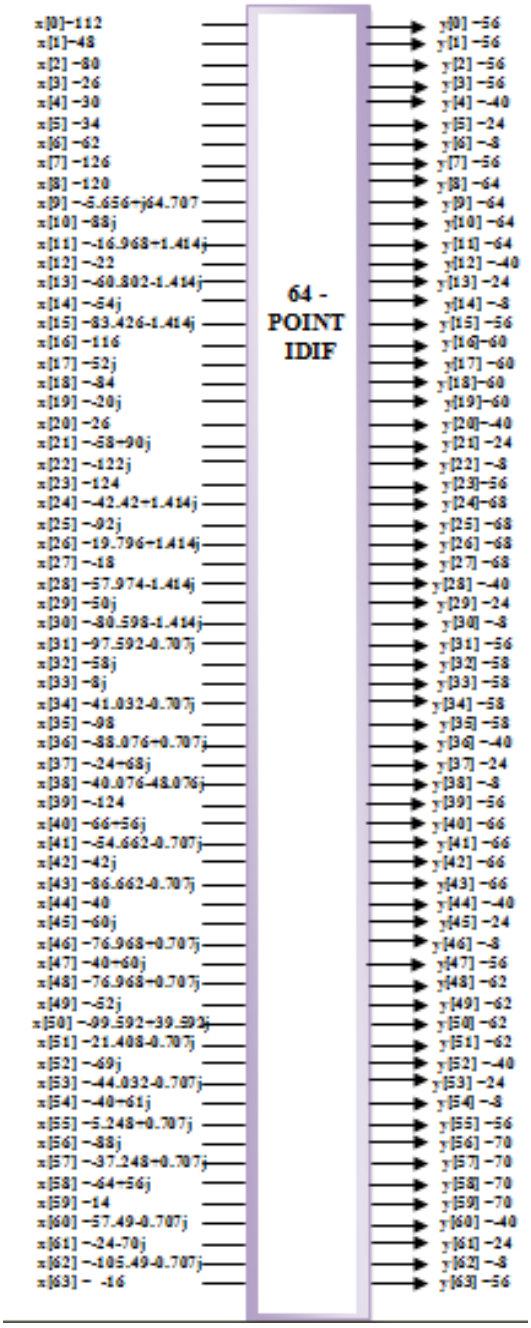
IDFT can be used in many digital processing systems across a variety of applications such as calculating a signal's frequency spectrum, solving partial differential applications, detection of targets from radar echoes, correlation analysis,

computing polynomial multiplication, spectral analysis, and more. IFFT has been widely used for acoustic measurements in churches and concert halls. Other applications of IFFT include spectral analysis in analog video measurements, large integer and polynomial multiplication, filtering algorithms, computing isotopic distributions, calculating Fourier series coefficients, calculating convolutions, generating low frequency noise, dense structured matrices, image processing, and more.

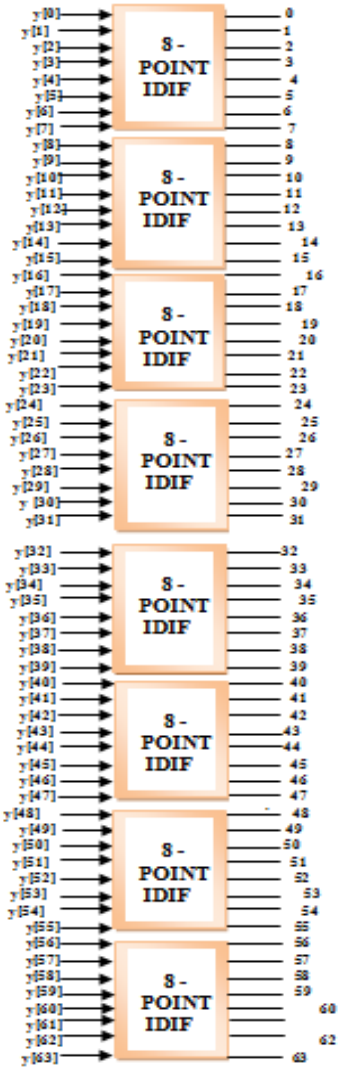
Summary of IFFT Vs. IDFT

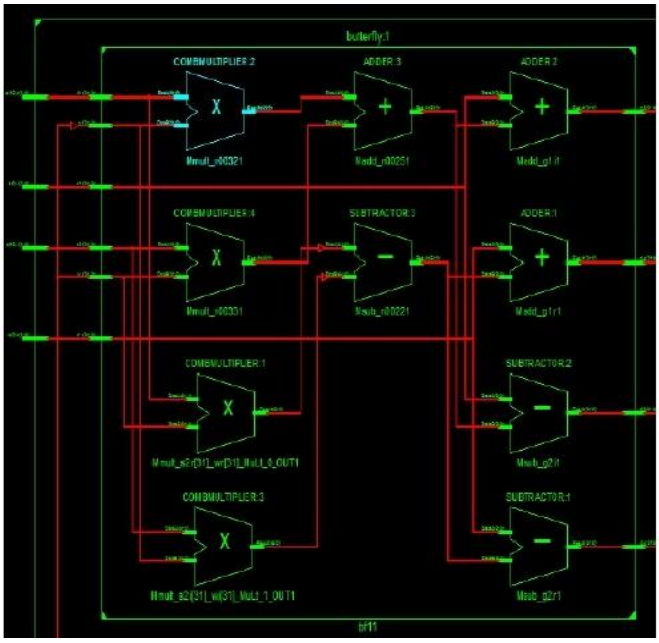
The Discrete Fourier Transform plays a key role in physics as it can be used as a mathematical tool to describe the relationship between the time domain and frequency domain representation of discrete signals. However, to reduce the computing time and complexity of large transforms, a more complex but less time-consuming algorithm such as the Fast Fourier Transform can be used. IFFT is an implementation of the IDFT used for used for fast computation of the IDFT. In short, IFFT can do everything a IDFT does, but more efficiently and much faster than a IDFT. It's an efficient way of computing the IDFT.Compare to that of radix 2 and radix 4 here in this paper we perform radix 8 operations by using the twiddle factors. The radix 8 butterfly structure helps us to carry out the complex calculations in the easier way. In the butterfly structure of Radix 8-point IFFT we have 2 stages. The number of stages are obtained by reducing the complexity using the IFFT. The number of stages are obtained as follows below.

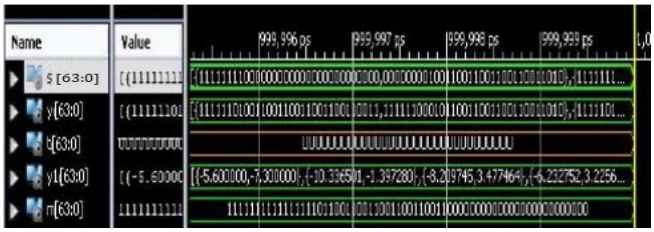
Block Diagram:

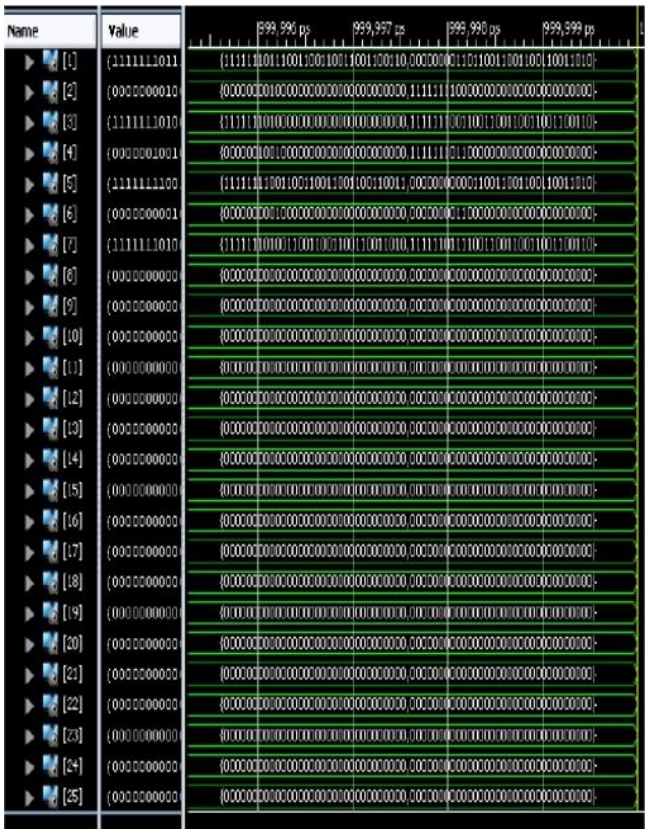


Block Diagram:



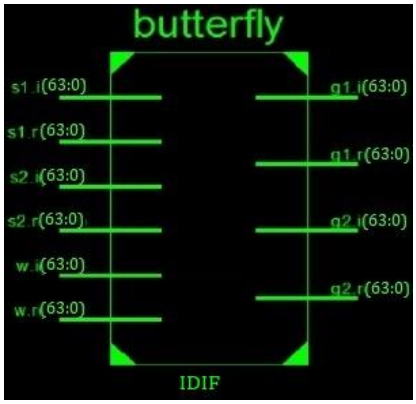




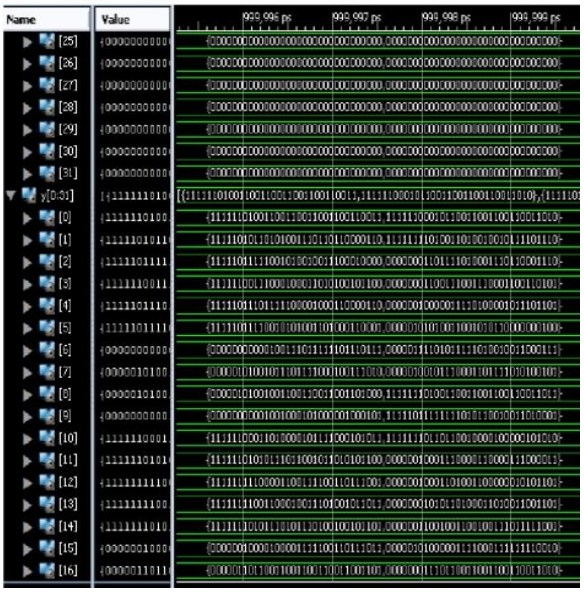


VI. SOFTWARE SIMULATION AND RESULTS

The proposed IFFT block of signal length 64 is been simulated and synthesized using the Xilinx Design Suite16.1. The RTL block thus obtained for the decimation intime domain radix -8Inverse Fast Fourier transform algorithm isshownThe RTL view of the butterfly structure obtained after thesimulation of the 64-point IFFT block, Decimation in timedomain is shown next and also the internal architectureofthe butterfly block is shown.



RTL Schematic:-



VII. CONCLUSION

This project describes the efficient use of VLSI for the implementation of radix 8 based IFFT architecture and the wave form result of the various stages has been obtained successfully. Compared to previous method the accuracy in obtained results has been increased with the help of efficient coding in VERILOG. The accuracy in results depends upon the equations obtained from the butterfly diagram and then on the correct drawing of scheduling diagrams based on these equations.The future scopes of this project are to implement the proposed IFFT architecture using Field-Programmable Gate Arrays (FPGAs) and also obtain the inverse Decimation In Frequency (IDIF) algorithm of IFFT.

REFERENCES

1 Bharat, K.; Broder, A. (1998): A technique for measuring the relative size and overlap of public Web search engines. Computer Networks, **30**(1–7), pp. 107–117.
2 Broder, A.; Kumar, R.; Maghoul, F.; Raghavan, P.; Rajagopalan, S.; Stata, R.; Tomkins, A.; Wiener, J. (2000): Graph structure in the Web. Computer Networks, **33**(1–6), pp. 309–320.
3 Chakrabarti, S. (2000): Data mining for hypertext: A tutorial survey. SIGKDD explorations, **1**(2), pp. 1–11.
4 SaadBouguezel, M. OmairAhmad,IMPROVED RADIX-4 ANDRADIX-8 IFFT ALGORITHMSIEEE. Department of Electrical andComputer Engineering Concordia University 1455 de MaisonneuveBlvd.West Montreal, P.Q., Canada.
5 Ali Saidi , DECIMATION-IN-TIME-FREQUENCY IFFTALGORITHM Motorola Applied Research, Paging and Wireless DataGroup Boynton Beach.
6 RizalafandeChe Ismail and RazaidiHussin High Performance ComplexNumber Multiplier Using Booth-Wallace Algorithm School of Microelectronic Engineering Kolej University KejuruteraanUtaraMalaysia.
7 J.G.Proakis and D.G.Manolakis, Digital Signal Processing,Principles,algorithms and applications Prentice Hall India Publication.
8 J.A.Hidalgo,V.Moreno-Vergara,O.Oballe, A Radix-8 Multiplier UnitDesign For SpecficPurpose,Dept of the Electronica ,E.T.S.I.Industriales
9 C. S. Burrus and T. W. Parks,IDFT/IFFT and Convolution Algorithms,NewYork,NY : John Wiley,1985.
10 A. Saidi , Generalized IFFTalgorithm,Proc. ICC,pp.227-231,May 1993.[8]

AUTHOR PROFILE

Mr. B. Anil Kumar , M.Tech working as Assistant professor ,in the Department of Electronics and communication , Mallareddy Institute of Engineering and Technology, Hyderabad. He studied B.Techin Electronics and Communications Engineering from JNTU college of Engineering, JNTUH, Hyderabad, Telangana and M.Techin VLSI SYSTEM DESIGN From Aurora college of engineering, JNTUH, yderabad,Telangana.

M. Naveen Reddy, is studying B.Tech in (Electronics & Communication Engineering) at Mallareddy Institute of Engineering & Technology (MRIET), Hyderabad. Telangana.

Mr. Vamshi Kollipara, M.Tech, working as Assistant professor ,in the Department of Electronics and communication , Mallareddy Institute of Engineering and Technology

B.Rajesh is studying B.Tech in (Electronics & Communication Engineering) at Mallareddy Institute of Engineering & Technology (MRIET), Hyderabad. Telangana.