IJIRSET
International Journal of Innovative Research in
SCIENCE | ENGINEERING | TECHNOLOGY

e-ISSN: 2319-8753 | p-ISSN: 2347-6710

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 13, Issue 2, February 2024

ISSN INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.423

9940 572 462 | 6381 907 438 | ijirset@gmail.com | www.ijirset.com

International Journal of Innovative Research in Science, Engineering and Technology (IJIRSET)

|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 13, Issue 2, February 2024 ||

| DOI:10.15680/IJIRSET.2024.1302058 |

FPGA Implementation of 8-Point DIT FFT in Combinational Logic using Xilinx 14.7

Anitha V, Swathi A, Sujitha M, Nargis J

Department of Electronics and Communication Engineering, Government College of Engineering, Bargur, krishnagiri, Tamil Nadu, India

ABSTRACT: This project presents the FPGA implementation of an 8-point Decimation in Time (DIT) Fast Fourier Transform (FFT) using Verilog in the Xilinx environment. The project involves breaking down the FFT algorithm into smaller stages and implementing them in Verilog. It includes simulation, synthesis, and verification stages using Xilinx tools. The implementation demonstrates the feasibility of using FPGAs for real-time FFT computations, contributing to the field of digital signal processing.

KEYWORDS: FPGA, Verilog, FFT, DIT, Xilinx, Butterfly units, Synthesis and Simulation.

I. INTRODUCTION

The Fast Fourier Transform (FFT) algorithm is a cornerstone in digital signal processing, facilitating the efficient computation of the Discrete Fourier Transform (DFT) for a sequence of complex numbers. Its applications span across various domains including communication systems, image processing, audio signal analysis, and many others. Among the different FFT algorithms, the radix-2 decimation in time (DIT) FFT is particularly notable for its simplicity and efficiency.In this project, we focus on the hardware implementation of a radix-2 8-point DIT FFT using Field-Programmable Gate Arrays (FPGAs) and Verilog hardware description language. The radix-2 FFT algorithm decomposes the DFT computation into smaller sub-problems, allowing for a divide-and-conquer approach that significantly reduces computational complexity.

1.1 Radix-2 8-Point DIT FFT: Overview

The Radix-2 8-point DIT FFT algorithm transforms an input time-domain sequence x(n) of length 8 into a frequency-domain sequence X(k). In the Decimation in Time (DIT) algorithm, the time-domain sequence is successively divided into smaller sub-sequences, enabling efficient computation of the FFT.In the DIT Radix-2 FFT, the time-domain sequence x(n) undergoes a decimation process, wherein it is split into 2-point sequences. For each 2-point sequence, a 2-point Discrete Fourier Transform (DFT) is computed. Subsequently, from the results of the 2-point DFTs, the 4-point DFT can be calculated. This process continues recursively until the desired 8-point DFT is obtained.
The DIT FFT algorithm is characterized by its ability to realize the N-point DFT from two N/2-point DFTs, which in turn are computed from two N/4-point DFTs, and so forth. This recursive decomposition significantly reduces the computational complexity of the FFT.The algorithm's efficiency lies in its reduced number of computations and processing time compared to direct computation methods. By splitting the input sequence into smaller and smaller subsequences, the DIT FFT algorithm achieves efficient frequency domain transformation.Radix-2 8-point DIT FFT algorithm serves as a powerful tool for transforming time-domain signals into their frequency-domain representations. Its recursive decomposition approach and efficient computation make it well-suited for real-time signal processing applications.

1.2 IEEE 754 Single Precision Floating Point Standard

We have implemented a radix-2 8-point Decimation in Time (DIT) Fast Fourier Transform (FFT) architecture using IEEE 754 single precision floating point representation for inputs, intermediate computations, and outputs. Our project primarily consists of fundamental modules for floating point addition, subtraction, and multiplication, which are utilized across multiple stages of the DIT FFT algorithm.IEEE 754 single precision format includes sign bit, exponent, and fraction bits, providing a standardized representation for floating point numbers.
A single precision floating-point number consists of 32 bits, organized into three parts:

|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 13, Issue 2, February 2024 ||

| DOI:10.15680/IJIRSET.2024.1302058 |

1 bit for the sign (positive or negative)
8 bits for the exponent
23 bits for the significand (also known as the mantissa or fraction)
Sign Bit: The leftmost bit (bit 31) represents the sign of the number. 0 indicates a positive number, and 1 indicates a negative number.
Exponent: The next 8 bits (bits 30 to 23) represent the exponent of the number. The exponent is biased by 127, meaning that an exponent of 0 is represented by 127, and an exponent of 1 is represented by 128, and so on. This allows both positive and negative exponents to be represented.
Significand: The remaining 23 bits (bits 22 to 0) represent the significand of the number. The significand is normalized such that the leading bit is always assumed to be 1 (except for denormalized numbers and special cases like zero and infinity). Therefore, only the fractional part beyond the leading 1 is stored explicitly.
Arithmetic operations, including addition, subtraction, multiplication, and division, are performed following specific rules defined by the IEEE 754 standard to ensure accuracy and consistency across different platforms and implementations.

1.3 Verilog HDL

Verilog HDL serves as the primary language for describing the digital circuits and system behaviour. We likely created testbenches to simulate the behavior of your 8-point FFT implementation. These testbenches generate input stimuli and monitor the output responses, enabling you to validate the correctness and accuracy. After simulation verification,the synthesis process ensures that your Verilog design meets the timing and resource constraints of the target FPGA platform. Then we used behavioral modeling to describe the functionality of the 8-point FFT algorithm using the DIT architecture.Verilog code outlines the sequential and parallel operations involved in computing the FFT, abstracting away details of the underlying hardware implementation.

II. MODULES

2.1 Floating Point Addition

Floating-point addition is a mathematical operation used to add two floating-point numbers represented in IEEE 754 standard format. It involves aligning the exponents of the numbers, adding or subtracting their significands, normalizing the result, and rounding it to fit within the specified precision. Overflow and underflow conditions must be handled to ensure accurate computation. Overall, floating-point addition is a fundamental operation in numerical computing, enabling precise representation and manipulation of real numbers.
The Verilog code we've designed encapsulates a floating-point addition module adhering to the IEEE 754 standard. Upon receiving two 32-bit floating-point numbers, the module undertakes a systematic process. Initially, it extracts the sign, exponent, and mantissa components from both input numbers, followed by aligning their exponents to prepare for addition. Through a series of logical operations, the module aligns the mantissas and performs addition using a dedicated 25-bit adder module. Special considerations are accounted for, including zero inputs, overflow, underflow, and rounding errors, ensuring precision and compliance with IEEE 754 standards. Ultimately, the module combines the resulting components to form the final 32-bit floating-point output, meticulously handling various scenarios to guarantee accurate floating-point addition.

2.2 Floating Point Subtraction

Floating-point subtraction is a crucial arithmetic operation essential for computing the difference between two floating-point numbers represented according to the IEEE 754 standard. In this standard, each floating-point number consists of a sign bit indicating positive or negative values, an exponent representing the scale of the number, and a significand containing the significant digits, including the fractional part. Subtraction in floating-point arithmetic involves several key steps. Initially, the exponents of the numbers being subtracted must be aligned to ensure they share the same scale. This alignment may necessitate shifting the significands and adjusting the exponents accordingly. Subsequently, the significands are subtracted, accounting for borrow during the subtraction process. Following subtraction, the result may require normalization to ensure the leading bit of the significand remains 1, thereby maintaining precision. Rounding may be necessary to accommodate the specified precision of the floating-point format, with various rounding modes available. Additionally, careful consideration is given to handling overflow and underflow conditions to ensure

|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 13, Issue 2, February 2024 ||

| DOI:10.15680/IJIRSET.2024.1302058 |

accurate representation of the result within the bounds of the floating-point format. Overall, floating-point subtraction is a complex yet fundamental operation critical for numerical computations requiring precise handling of real numbers.

2.3 Floating Point Multiplication

The Floating-point multiplication is a fundamental arithmetic operation crucial for computing the product of two floating-point numbers adhering to the IEEE 754 standard. Within this standard, each floating-point number encompasses a sign bit, an exponent denoting the scale of the number, and a significand embodying its significant digits, including the fractional component. The process of floating-point multiplication involves aligning the exponents of the multiplicand and multiplier to ensure their scales are compatible. This alignment necessitates appropriate shifts of the significands and adjustments to the exponents. Following exponent alignment, the significands are multiplied, and the result is adjusted to maintain precision and compliance with the floating-point format. Rounding may be applied to accommodate the specified precision and ensure the result fits within the confines of the floating-point representation. Like subtraction, floating-point multiplication requires consideration for potential overflow and underflow conditions, with strategies in place to address these scenarios and ensure accurate representation of the product. In summary, floating-point multiplication is a pivotal operation in numerical computing, enabling the precise computation of products while conforming to established standards for floating-point arithmetic.

2.4 Butterfly unit : 2 point

In the context of the first stage of an 8-point Decimation in Time (DIT) Fast Fourier Transform (FFT), the basic butterfly unit computes a 2-point Discrete Fourier Transform (DFT) using floating-point addition, subtraction, and multiplication, along with the application of twiddle factors. Here's a breakdown of how this basic butterfly unit operates:

1.Input - The basic butterfly unit takes two complex input values X_0 and X_1, represented as X_0 = x_0\re + jx_0\im and X_1 = x_1\re + jx_1_im , respectively. These represent the "top" and "bottom" elements of the butterfly unit.
2. Twiddle Factor - The twiddle factor W is a complex exponential coefficient that introduces the necessary phase shift for frequency domain transformations. W_8^0 the twiddle factor for the first element in an 8-point Discrete Fourier Transform (DFT) computation. In the context of a Radix-2 FFT, W_8 refers to the 8th root of unity, which can be expressed as a complex number e^{-j2pi8}.When k = 0 , W_8^0 simplifies to e^{-j0}, which is equal to 1. Therefore,W_8^0 equals 1.
3. Computation - The basic butterfly unit computes the following operations:
Y_0 = X_0 + W times X_1
Y_1 = X_0 - W times X_1 .
These computations involve floating-point addition, subtraction, and multiplication operations. The addition and subtraction operations compute the sum and difference of the input values, respectively. Multiplication by the twiddle factor applies the necessary phase shift to the input values.
4. Output - The resulting complex values Y_0 and Y_1 represent the outputs of the butterfly unit.(Y_0 \) contains the sum of the input values and the contribution from the twiddle factor.(Y_1 \) contains the difference of the input values and the contribution from the twiddle factor.

In summary, the basic butterfly unit in the first stage of an 8-point DIT FFT computes a 2-point DFT using floating-point arithmetic operations such as addition, subtraction, and multiplication, augmented by the application of twiddle factors. This unit is crucial for decomposing the input sequence into smaller DFTs, laying the groundwork for subsequent stages of the FFT algorithm.

2.5 Stage I

In stage 1 of a Radix-2 8-point Decimation in Time (DIT) Fast Fourier Transform (FFT), the input sequence is divided into four 2-point segments, and each segment undergoes a 2-point Discrete Fourier Transform (DFT) computation. However, since the twiddle factor \(W_8^0\) is equal to 1, no actual twiddle factor multiplication is performed in this stage. Instead, the computation involves simple butterfly operations. Here's an overview of stage 1:

1. Input Sequence - The input sequence of 8 points is initially arranged in appropriate order.

|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 13, Issue 2, February 2024 ||

| DOI:10.15680/IJIRSET.2024.1302058 |

2. Segmentation - The input sequence is divided into four 2-point segments: X_0 and X_4, X_2 and X_6, X_1 and X_5 and finally X_3 and X_7.
3. Butterfly Operations - For each 2-point segment, a butterfly operation is performed. Since the twiddle factor W_8^0 is 1, the computation simplifies to basic addition and subtraction.For example, for segment X_0, the butterfly operation computes:
Y_0 = X_0 + X_4
Y_1 = X_0 - X_4
Similarly, similar operations are performed for other segments also.
4. Output Sequence - After the butterfly operations, the output sequence consists of the resulting values Y_0, Y_1, Y_2 , Y_3 , Y_4, Y_5, Y_6 and Y_7.
This stage serves to decompose the input sequence into smaller DFTs, laying the foundation for subsequent stages of the FFT algorithm.

2.6 Stage II

In stage 2 of the Radix-2 8-point Decimation in Time (DIT) Fast Fourier Transform (FFT), the output sequence from stage 1, consisting of eight points y_0 to y_7 , is further processed. This stage utilizes twiddle factors W_8^0 = 1 and W_8^1 = -j , where j represents the imaginary unit. Here's an overview of stage 2:

1. Input Sequence - The output sequence from stage 1, consisting of eight points y_0 to y_7, is the input to stage 2.
2. Segmentation - The sequence is split into two pairs of 4-point segments: y_0 y_1 y_2 y_3 and y_4 y_5 y_6 y_7 .
3. Twiddle Factor Multiplication - Each segment pair undergoes multiplication by twiddle factors. For the first pair y_0 y_1 are multiplied by W_8^0 = 1 and y_2 y_3 are multiplied by W_8^1 = -j .This multiplication by W_8^1 introduces the required phase difference between the two segments.
4. Butterfly Operations - Each pair of segments undergoes butterfly operations, incorporating the effects of twiddle factors.For the first pair y_0 y_1 y_2 y_3 , the butterfly operation computes:
z_0 = y_0 + y_2
z_1 = y_1 + y_3
z_2 = y_0 - y_2 (-j) = y_0 + jy_2
z_3 = y_1 - y_3 (-j) = y_1 + jy_3
5. Output Sequence - After the butterfly operations, the resulting sequence consists of z_0 , z_1 , z_2 and z_3 for the first pair, and z_4 , z_5 , z_6 and z_7 for the second pair.

2.7 Stage III

In stage 3 of the Radix-2 8-point Decimation in Time (DIT) Fast Fourier Transform (FFT), the output sequence from stage 2, consisting of eight points z_0 to z_7 , is further processed. This stage utilizes twiddle factors W_8^0 , W_8^1 , W_8^2 , W_8^3 . Here's an overview of stage 3:

1. Input Sequence - The output sequence from stage 2, consisting of eight points z_0 to z_7, is the input to stage 3.
2. Segmentation - The sequence is split into a pair of 8-point segments: Z_0 Z_1 Z_2 Z_3 Z_4 Z_5 Z_6 and Z_7 .
3. Twiddle Factor Multiplication - Each segment pair undergoes multiplication by twiddle factors. For the second half multiplied by twiddle factors W_8^0 , W_8^1 , W_8^2 , W_8^3 .
4. Butterfly Operations - The input sequence undergoes butterfly operations to compute the 8-point DFT.The computation involves combining pairs of points according to the FFT algorithm.
5. Output Sequence - After the butterfly operations, the resulting sequence consists of Z_0 , Z_1 , Z_2 , Z_3 , Z_4 , Z_5 , Z_6 and Z_7 .

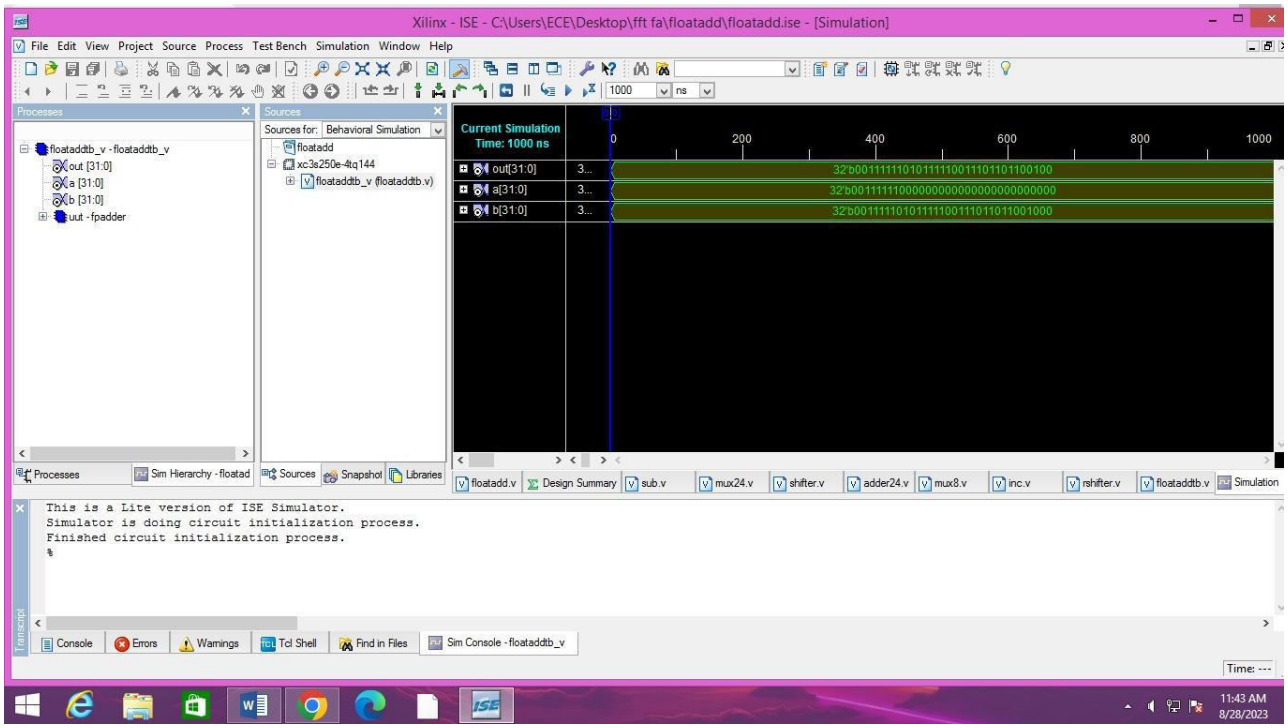
|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 13, Issue 2, February 2024 ||

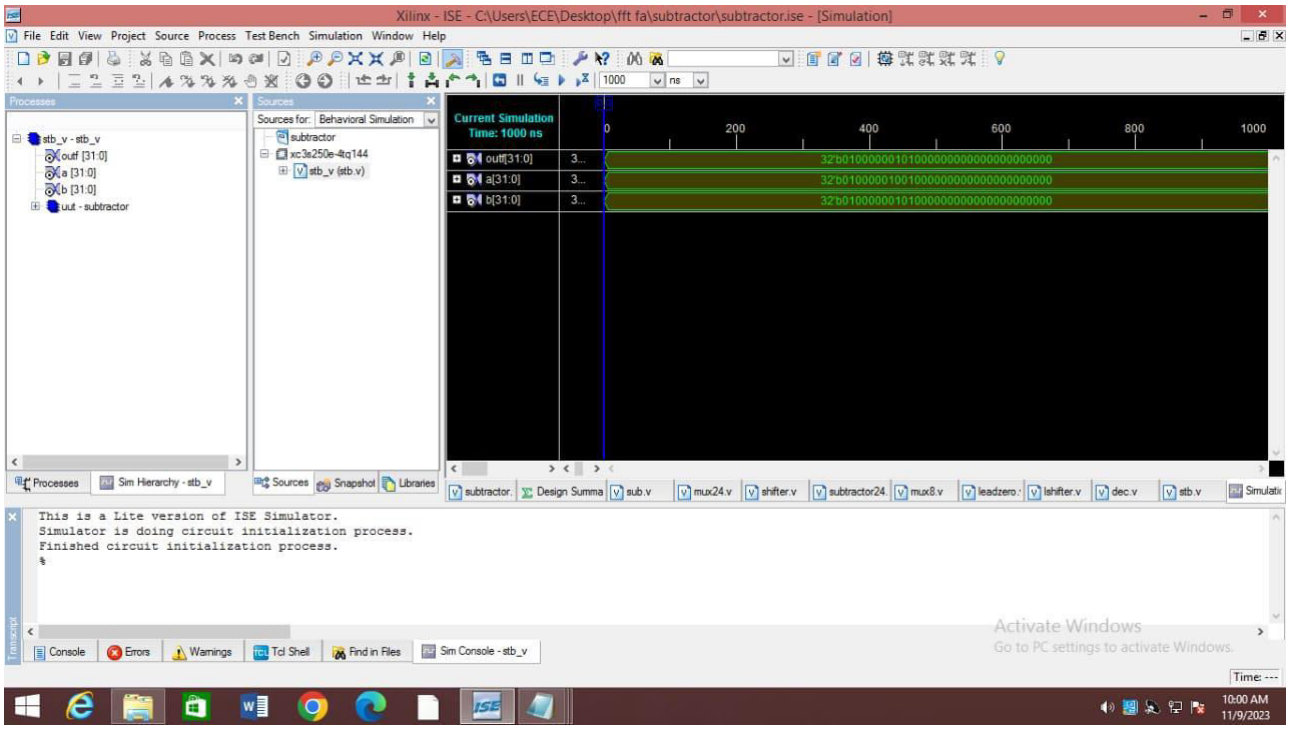
| DOI:10.15680/IJIRSET.2024.1302058 |

III. EXPERIMENTAL RESULTS

3.1 Floating Point Addition



3.2 Floating Point Subtraction

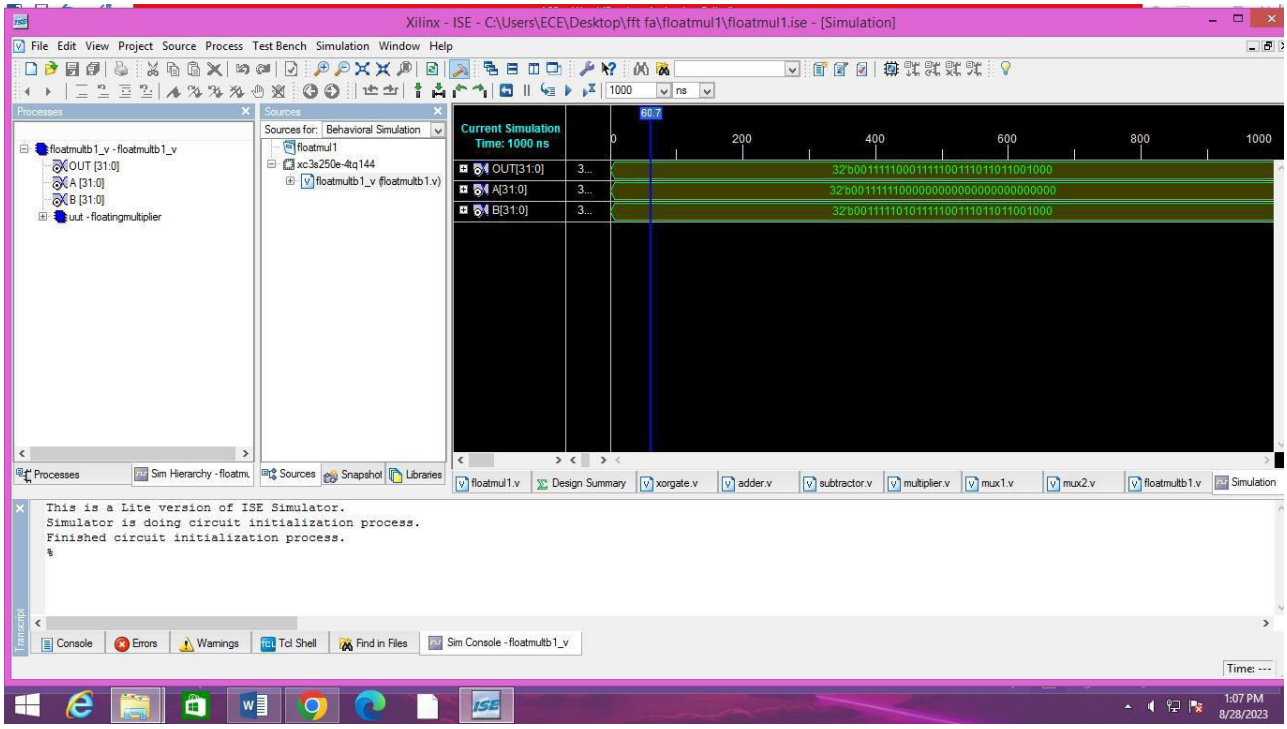


|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

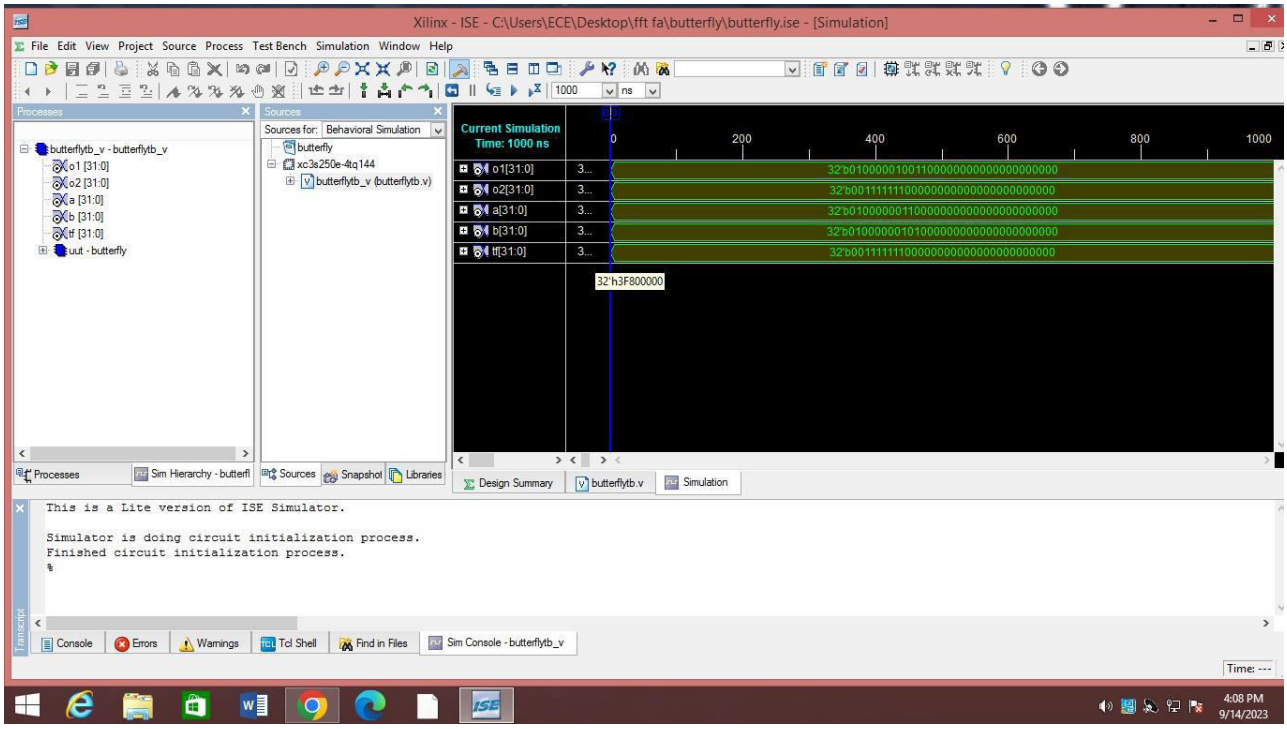
|| Volume 13, Issue 2, February 2024 ||

| DOI:10.15680/IJIRSET.2024.1302058 |

3.3 Floating Point Multiplication



3.4 Butterfly unit : 2 point

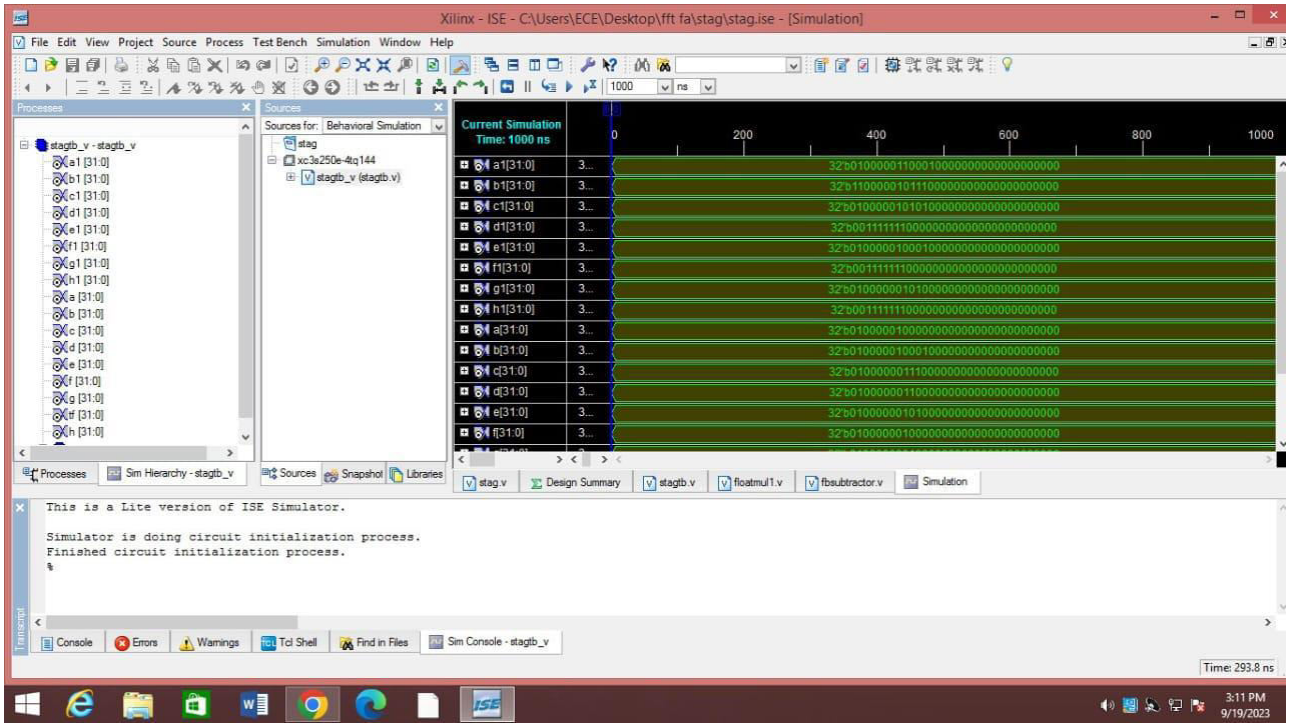


|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

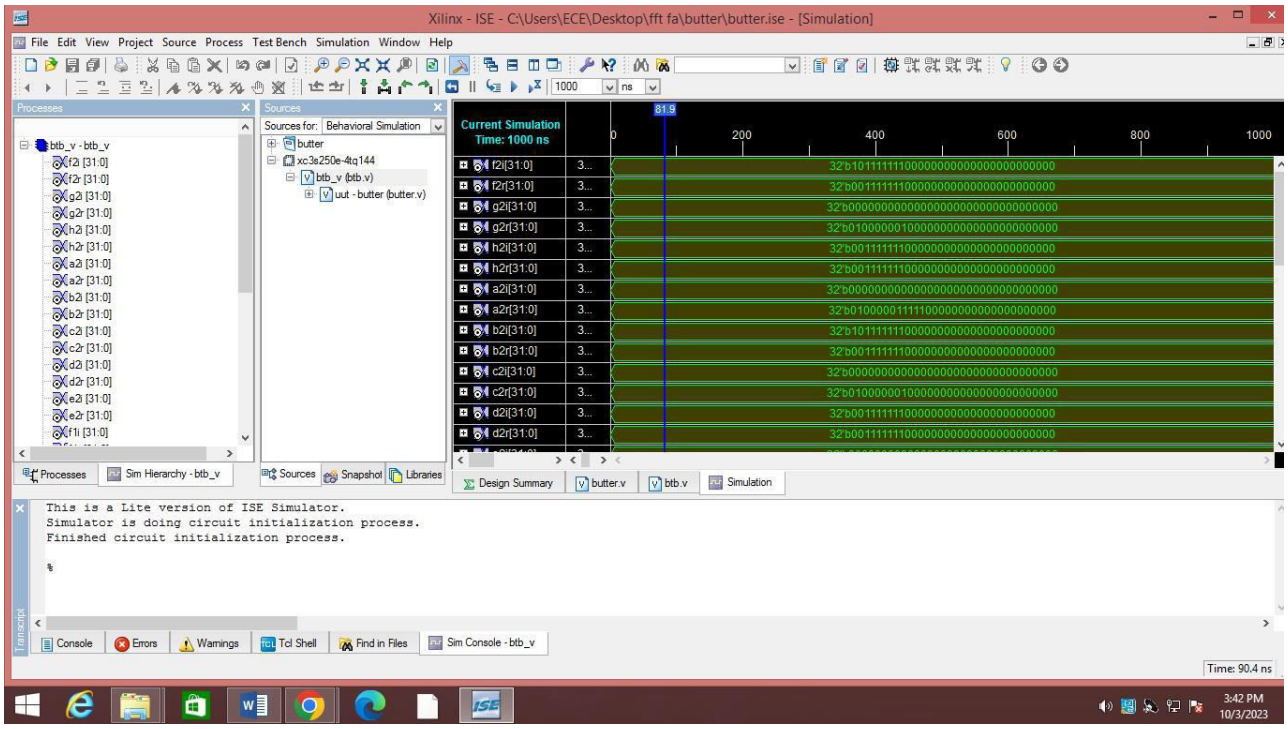
|| Volume 13, Issue 2, February 2024 ||

| DOI:10.15680/IJIRSET.2024.1302058 |

3.5 Stage I



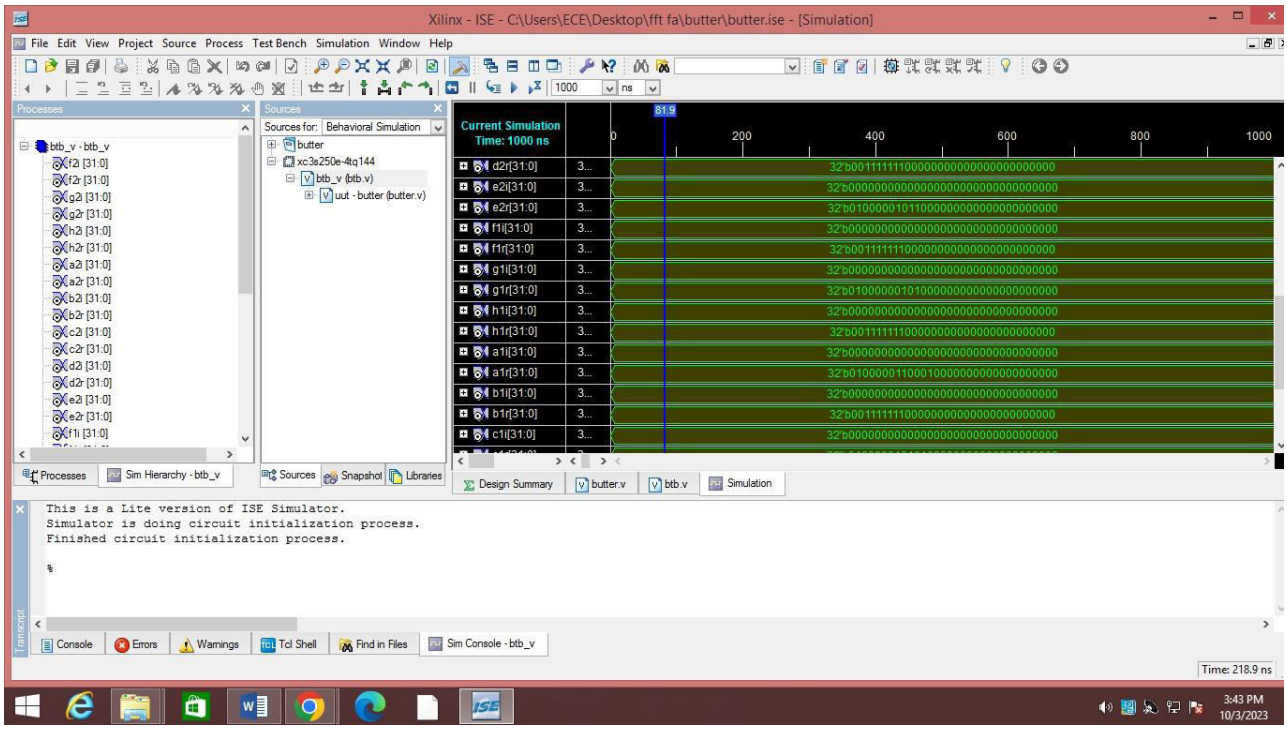
3.6 Stage II



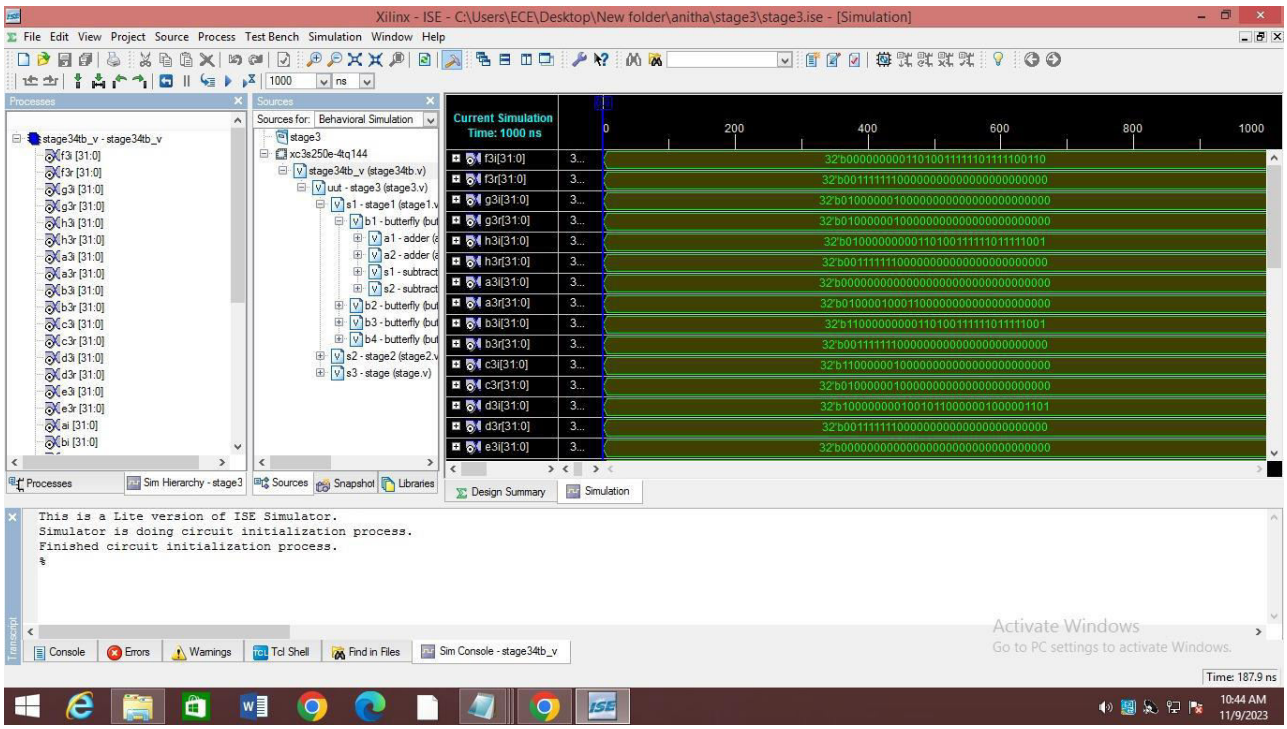
|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 13, Issue 2, February 2024 ||

| DOI:10.15680/IJIRSET.2024.1302058 |



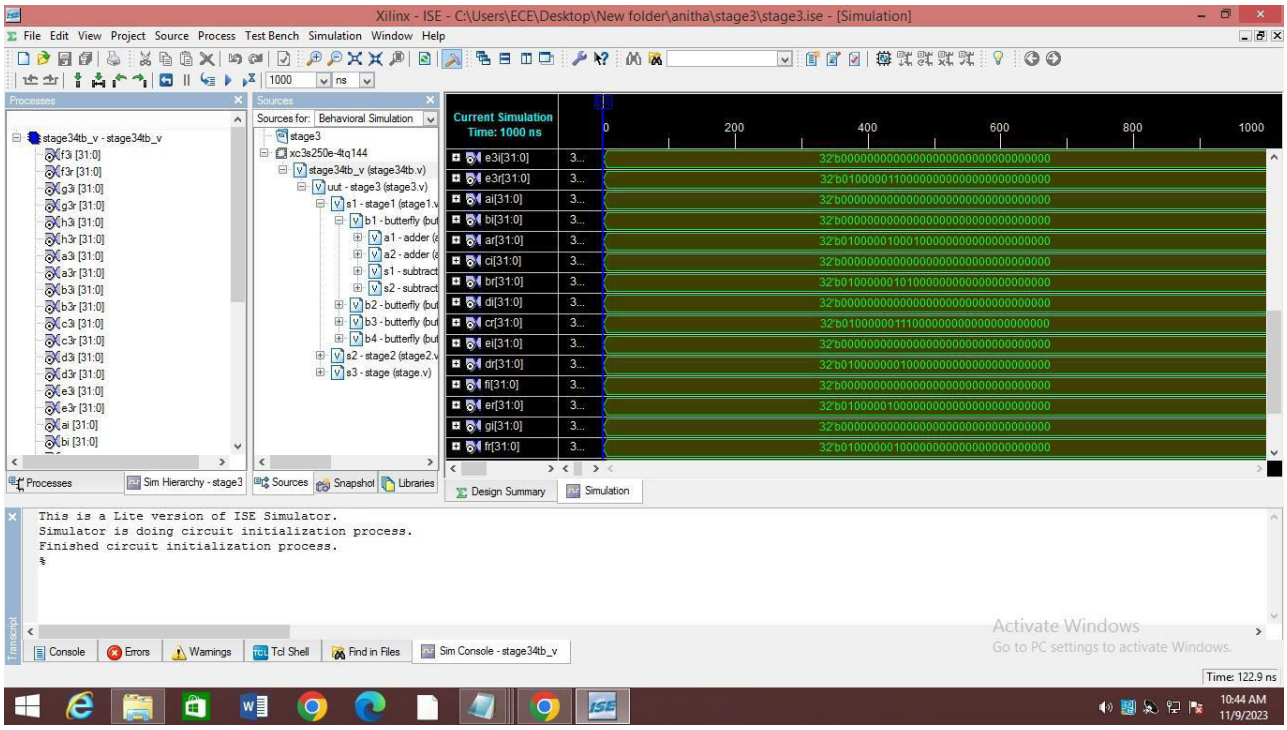
3.7 Stage III



|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 13, Issue 2, February 2024 ||

| DOI:10.15680/IJIRSET.2024.1302058 |



3.8 Synthesis Output

Device utilization summary:
Selected Device: 5vlx30ff324-3
Slice Logic Utilization:
Number of Slice LUTs: 17132 out of 19200 89%
Number used as Logic: 17132 out of 19200 89%
Slice Logic Distribution:
Number of Bit Slices used: 17132
Number with an unused Flip Flop 17132 out of 17132 100%
Number with an unused LUT: 0 out of 17132 0%
Number of fully used Bit Slices: 0 out of 17132 0%
IO Utilization:
Number of IOs: 1024
Number of bonded IOBs: 1024 out of 220 465% (*)
Specific Feature Utilization:
Number of DSP48Es: 28 out of 32 87%
Maximum combinational path delay: 55.659ns

3.9 Device Utilization Summary

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	13939	960	1451%
Number of 4 input LUTs	25143	1920	1309%
Number of bonded IOBs	1024	66	1551%
Number of MULT18X18SIOs	2	4	50%

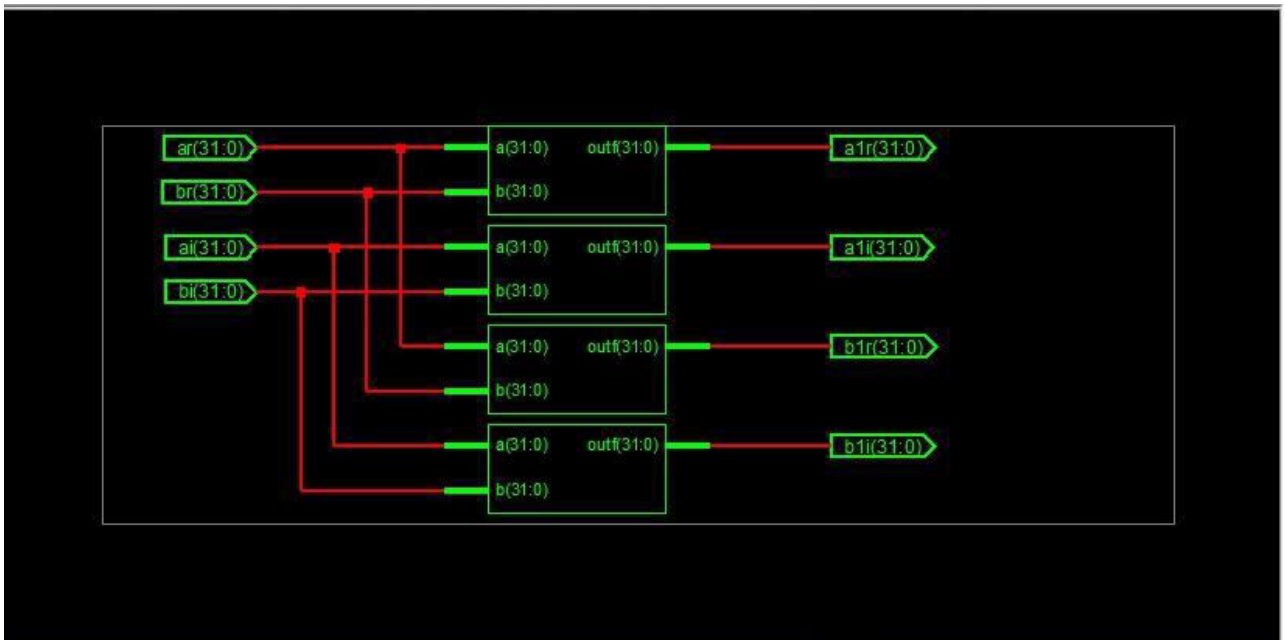
|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 13, Issue 2, February 2024 ||

| DOI:10.15680/IJIRSET.2024.1302058 |

3.10 RTL Schematics

Stage 1

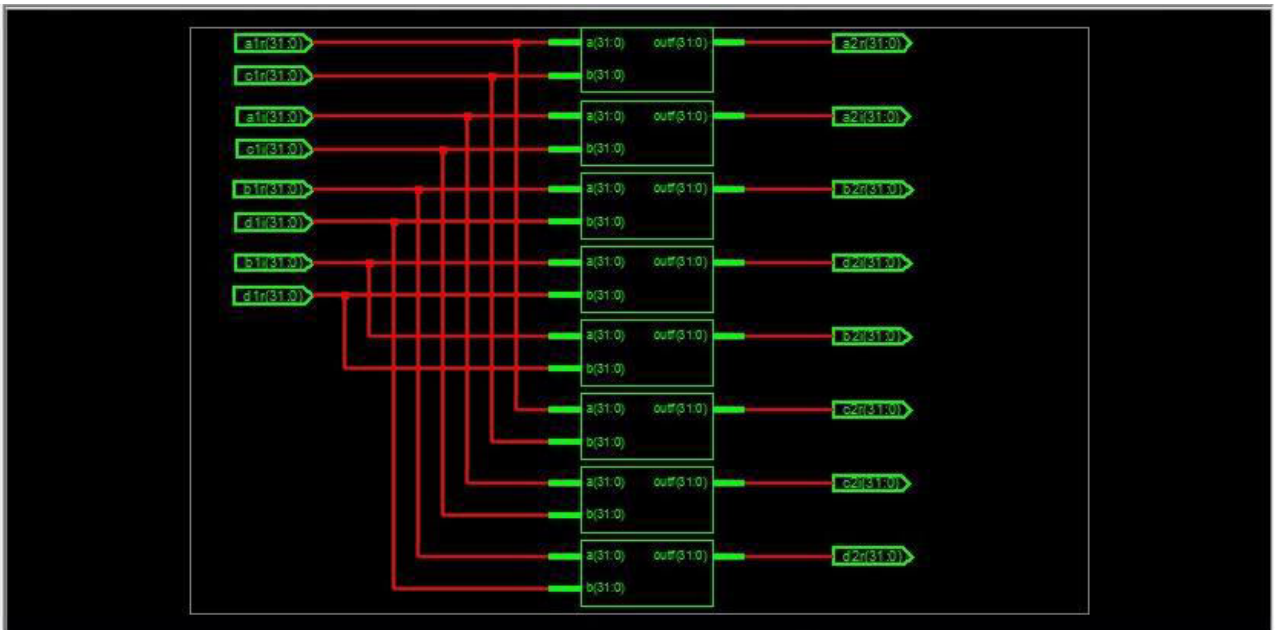


|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 13, Issue 2, February 2024 ||

| DOI:10.15680/IJIRSET.2024.1302058 |

Stage II



Stage III

|e-ISSN: 2319-8753, p-ISSN: 2347-6710| www.ijirset.com | Impact Factor: 8.423| A Monthly Peer Reviewed & Referred Journal |

|| Volume 13, Issue 2, February 2024 ||

| DOI:10.15680/IJIRSET.2024.1302058 |

IV. CONCLUSION

This project successfully implemented the Radix-2 8-point Decimation in Time Fast Fourier Transform (DIT FFT) on an FPGA using Xilinx ISE software. It involved floating-point addition, subtraction, multiplication, and the execution of all three stages of the DIT FFT algorithm.This achievement demonstrates the potential of FPGA technology for real-time signal processing and complex mathematical computations. It underscores the importance of FPGA in solving computational challenges and opens the door to innovative solutions in various fields**.**

REFERENCES

[1] Oppenheim, A. V., and Schafer, R. W. (2010). Discrete-Time Signal Processing. Prentice Hall
[2] Xilinx Inc. (2022). Xilinx ISE Design Suite: Getting Started Guide. [Online Document] Available at: [Insert URL]
[3] Proakis, J. G., and Manolakis, D. G. (2006). Digital Signal Processing: Principles, Algorithms, and Applications. Pearson Education.
[4] Brace, A. (1998). FFTs and the CORDIC algorithm. Newnes.
[5] Papoulis, A., and Pillai, S. U. (2002). Probability, Random Variables and Stochastic Processes. Tata McGraw-Hill Education.
[6] Xilinx Inc. (2022). Xilinx Application Notes. [Online Resource]

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi cross ref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

9940 572 462 6381 907 438 ijirset@gmail.com

www.ijirset.com

Scan to save the contact details