



# Design Of An Efficient Single Precision Floating Point Unit

<sup>1</sup>Sushant S Chachadi, <sup>2</sup>Shreyas U Naik, <sup>3</sup>Raveendra Arjun Honnakore, <sup>4</sup>Atharv Satish Bharadwaj, <sup>5</sup>Sujatha K

<sup>1</sup>Student, <sup>2</sup>Student, <sup>3</sup>Student, <sup>4</sup>Student, <sup>5</sup>Associate Professor

<sup>1</sup>Department of Electronics and Communication,

<sup>1</sup>B.M.S College of Engineering, Bengaluru, India

**Abstract:** In this paper, we dive into the design of a Single Precision Floating Point Unit (FPU), a key player in the world of modern processors. FPUs are essential for handling complex numerical calculations with high precision and a broad range, making them indispensable in areas like scientific research, graphics rendering, and machine learning. Our design centers around two main components: the Brent-Kung adder and the radix-4 Booth multiplier. The Brent-Kung adder is our go-to for fast addition and subtraction. Thanks to its clever parallel-prefix structure, it keeps delays minimal even as the numbers get bigger. For multiplication, we turn to the radix-4 Booth multiplier. This powerhouse streamlines the multiplication process by cutting down the number of partial products and operations needed, handling both positive and negative numbers with ease. By integrating these components, our FPU can handle floating-point arithmetic with great efficiency and reliability. In scientific computing, this means more accurate simulations and data analyses. For graphics processing, it translates to better image rendering and smoother visual effects. And in machine learning, it supports robust training and execution of algorithms on massive datasets, ensuring dependable model performance.

**Index Terms** - Brent-Kung adder, Floating Point Unit, Radix4 Booth Multiplier, Single Precision, Verilog

## I. INTRODUCTION

In the realm of modern computing, the Floating-Point Unit (FPU) is a vital component, playing a pivotal role in executing arithmetic operations on floating-point numbers. FPUs are integral to processors, enabling them to handle a wide range of numerical calculations with high precision and efficiency. This capability is essential for applications that demand extensive computational power, such as scientific research, graphics rendering, and machine learning. The design and optimization of FPUs are crucial for enhancing the performance and efficiency of these applications. Floating-point and fixed-point are two fundamental methods for representing real numbers in digital systems, each with distinct advantages and trade-offs. Fixed-point representation uses a fixed number of bits to represent the integer and fractional parts of a number, with the position of the radix point (decimal point) remaining constant. This representation is straightforward and efficient for hardware implementation, making it suitable for applications where the range of values is known and limited, such as embedded systems and digital signal processing. However, fixed-point representation can suffer from limited dynamic range and precision, leading to potential inaccuracies in calculations involving very large or very small numbers. In contrast, floating-point representation employs a scientific notation-like format, where the number is divided into a significand (or mantissa) and an exponent. This format, defined by the IEEE 754 standard, allows for a wide dynamic range and greater precision, accommodating a vast spectrum of values with more significant digits. Consequently, floating-point representation is essential for applications requiring high precision and extensive numerical computations, such as scientific simulations, graphics processing, and

machine learning. However, floating-point arithmetic is generally more complex and resource-intensive than fixed-point, necessitating sophisticated hardware like the Floating-Point Unit (FPU) to perform efficient computations. The single-precision floating-point format (FP32 or float32) is a 32-bit numerical representation that uses a floating radix point to cover a wide dynamic range. Compared to fixed-point formats of the same bit width, it supports a broader range of values but with some loss of precision. For example, while a signed 32-bit integer can store values up to 2,147,483,647, a 32-bit IEEE 754 floating-point number can represent values as large as approximately  $3.4028235 \times 10^{38}$ . Additionally, any integer with up to seven decimal digits and any  $2^n$  for  $-149 \leq n \leq 127$  can be exactly represented. The IEEE 754 binary32 format is defined as follows:

- Sign bit: 1 bit
- Exponent width: 8 bits
- Significand precision: 24 bits (23 explicitly stored)

This provides a precision of about 6 to 9 significant decimal digits. A decimal number with up to 6 significant digits should remain unchanged when converted to and from IEEE 754 single-precision format. The sign bit indicates whether the number is positive or negative, while the 8-bit exponent ranges from 0 to 255 in a biased form, with 127 representing zero. The actual exponent values span from  $-126$  to  $+127$ , as  $-127$  and  $+128$  are reserved for special cases. The significand consists of 23 fraction bits stored in memory, along with an implicit leading bit, providing an effective 24-bit precision. Implementing an efficient Floating-Point Unit (FPU) requires designing high-performance arithmetic circuits to ensure accuracy and speed. In this report, we focus on two critical components: the Brent-Kung adder and the radix-4 Booth multiplier. These units are employed to perform addition, subtraction, and multiplication operations, which are fundamental to floating-point arithmetic. The Brent-Kung adder is an efficient parallel-prefix adder known for its logarithmic delay in relation to the operand size. It achieves this by dividing the addition process into smaller, manageable stages that can be executed in parallel. This structure significantly reduces the delay associated with the carry propagation, which is a common bottleneck in addition operations. By organizing the computation in a tree-like structure, the Brent-Kung adder ensures that the delay grows logarithmically with the number of bits, making it a highly efficient solution for fast addition and subtraction operations in an FPU.

Multiplication, another critical operation in floating-point arithmetic, is handled by the radix-4 Booth multiplier. The radix-4 Booth algorithm is an advanced technique that enhances multiplication efficiency by reducing the number of partial products generated during the multiplication process. It achieves this by encoding the multiplier in a way that allows multiple bits to be processed simultaneously, thereby minimizing the number of required operations. This approach not only speeds up the multiplication process but also reduces power consumption and circuit complexity. The radix-4 Booth multiplier is particularly adept at handling both positive and negative operands, making it a versatile and robust component for the FPU.

The integration of the Brent-Kung adder and the radix-4 Booth multiplier within the FPU allows for efficient and accurate floating-point computations. These components work in tandem to execute arithmetic operations swiftly and precisely, catering to the needs of high-performance computing tasks. The following sections of this paper will delve into the detailed design and implementation of these components, exploring how their integration enhances the overall performance of the Single Precision Floating Point Unit.

## II. LITERATURE SURVEY

N. A. S. Adela, A. N. B. Yousuf, and M. M. Eljhani presented a paper titled "Design and Implementation of Single Precision Floating-Point Arithmetic Logic Unit for RISC Processor on FPGA." This research focuses on designing and implementing a single-precision floating-point arithmetic logic unit (ALU) as part of a math coprocessor. Floating-point representation offers a significant advantage over fixed-point and integer formats by supporting a broader range of values. The ALU performs essential arithmetic operations, including addition, subtraction, multiplication, and division, using inputs in the IEEE-754 single-precision format (32-bit). High-performance arithmetic hardware is crucial due to the extensive use of microprocessors and signal processors. The ALU, a key component of microprocessors, directly influences processing speed. Modern CPUs integrate dedicated circuits for arithmetic calculations, and on-chip memory or cache enhances performance by minimizing data access latency. Floating-point operations are widely used across industries due to their broad dynamic range, straightforward rules, and high precision. The growing demand for high-speed floating-point units stems from their necessity in scientific computing and high-speed data signal processing. Additionally, floating-point arithmetic is increasingly utilized in commercial, financial, and web-based applications. This representation method, commonly used in scientific notation, employs a sliding window mechanism to adjust precision dynamically, accommodating values ranging from  $1 \times 10^{12}$  to  $1 \times$

10<sup>-12</sup>. Implementing floating-point arithmetic on FPGAs remains a complex challenge due to the intricate nature of the required algorithms.

[2] Samraj Daphni, Kasinadar Sundari Vijula Grace explored the performance of binary adders, which play a critical role in determining the speed and accuracy of processors and control systems. Traditionally, 32-bit carry adders such as Ripple Carry Adders (RCA), Carry Propagate Adders (CPA) and Carry Look-Ahead Adders (CLA) were widely used, differing in delay, area, and power consumption. The key factor influencing their performance is the propagation of the carry bit across bit positions, which directly impacts overall delay. Conventional 32-bit carry adders experience significant delays in higher-order bits due to their sequential nature, where each stage must wait for the previous carry result. To mitigate this issue, modern systems prefer Parallel Prefix Adders (PPA) for their ability to accelerate addition while reducing delay. PPAs strike a balance between area, speed, and power efficiency. While earlier research focused on smaller adders like 8-bit and 16-bit designs, this study examines 32-bit PPAs, including the Kogge-Stone Adder, Brent-Kung Adder, and Ladner-Fischer Adder. The primary advantage of PPAs over traditional carry adders is their lower delay, achieved through an optimized design based on the CLA. PPAs are also referred to as logarithmic delay adders since their delay scales with the logarithm of the bit count. The PPA addition process consists of three key stages: Pre-computation (generating propagate and generate signals), Prefix-computation (grouping carry signals), and post-computation (producing the final sum).

[4] Shahzad Asif and Yinan Kong presented a study titled "Performance Analysis of Wallace and Radix-4 Booth-Wallace Multipliers." Multiplication is a fundamental arithmetic operation, and high-speed, area-efficient multipliers are essential in modern computing. Wallace tree-based multipliers offer an efficient approach for fast multiplication by minimizing the number of reduction stages. Traditionally, Booth encoding has been widely integrated into tree multipliers to enhance speed. However, as technology scales down, the effectiveness of Booth encoders diminishes. This study demonstrates that in deep submicron technology, incorporating Booth encoders into Wallace multipliers increases delay and power consumption rather than improving performance.

### III. PROBLEM STATEMENT AND PROPOSED SOLUTION

#### 3.1 Problem Statement

In today's computing landscape, the demand for high-performance computing systems capable of handling complex numerical computations is ever-growing. One critical aspect of such systems is the efficient processing of floating-point arithmetic operations. Floating-point arithmetic is fundamental in various fields such as scientific computing, engineering simulations, financial modelling, and graphics rendering. However, traditional arithmetic logic units (ALUs) may not meet the performance and power efficiency requirements for handling floating-point operations effectively. Therefore, the objective of this project is to design and implement a single precision floating-point unit optimized for both speed and power consumption. This specialized ALU will be capable of performing addition, subtraction, multiplication, division, and other logical operations on floating-point numbers with high efficiency.

#### 3.2 Proposed Solution

The solution involves creating a Single Precision Floating Point Unit that integrates advanced computational units to achieve high-speed and accurate arithmetic operations.

- **Floating Point Unit Design:** The FPU is designed to perform essential arithmetic operations while adhering to the IEEE 754 standard for single-precision floating-point numbers. This ensures a wide dynamic range and precision, necessary for various high-performance computing applications.
- **Brent-Kung Adder:** To handle addition and subtraction, the FPU employs the Brent-Kung adder, known for its efficient parallel-prefix structure. This adder reduces delay by minimizing the carry propagation time, which is crucial for rapid arithmetic operations.
- **Radix-4 Booth Multiplier:** For multiplication operations, the FPU uses the radix-4 Booth multiplier. This multiplier enhances performance by reducing the number of partial products and efficiently handling both positive and negative operands, streamlining the multiplication process.

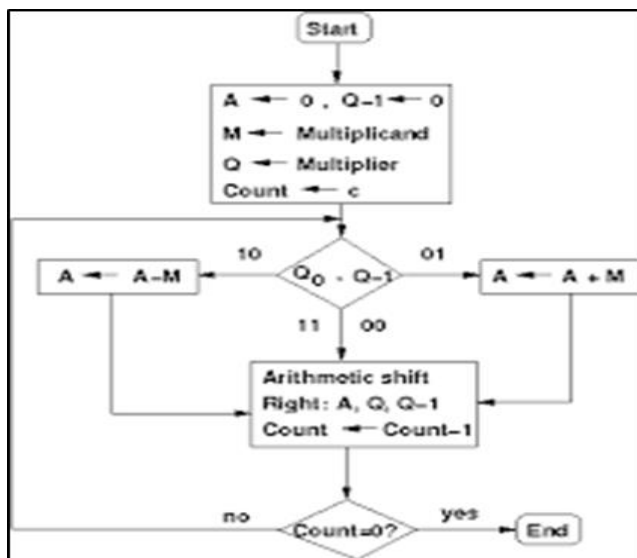


Figure 3.1 Block Diagram of Brent Kung Adder

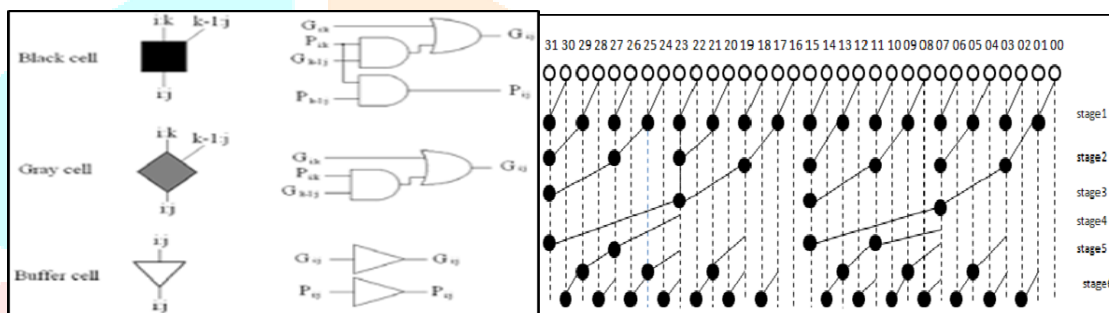


Figure 3.2 Flowchart of Radix4 Booth Multiplier

**IV. METHODOLOGY AND IMPLEMENTATION**

The implementation of the Single Precision Floating Point Unit (FPU) involved a structured approach, including the development, integration, and testing of individual components to ensure adherence to the IEEE 754 standard for floating-point arithmetic.

1. **Development of Individual Modules:** Brent-Kung Adder: The process began with writing a Verilog module for the 32-bit Brent-Kung adder, utilizing a parallel-prefix structure to achieve efficient addition with minimal delay. Radix-4 Booth Multiplier: Concurrently, a Verilog module for the radix-4 Booth multiplier was developed.
2. **Testing Individual Modules:** Both the Brent-Kung adder and the radix-4 Booth multiplier modules were individually tested using Vivado, an FPGA design suite. Testbenches were written to validate the functionality and performance of each module, ensuring they met the required specifications. **Integration into FPU Top Module:** Following the validation of the individual modules, integration into an FPU top module was performed. In this module, the Brent-Kung adder and the radix-4 Booth multiplier were instantiated to handle floating-point arithmetic operations.
3. **Preprocessing Inputs:** Before performing any operations, preprocessing of the inputs according to the IEEE 754 format was necessary. This involved extracting the sign bit, exponent, and mantissa from the 32-bit floating-point inputs. This preprocessing step was crucial for the FPU to correctly interpret and manipulate the floating-point numbers.
4. **Operation Execution:** Based on the operation code, the FPU determined which arithmetic operation to perform. Using the preprocessed inputs, the FPU executed the specified operation—whether addition, subtraction, or multiplication—by utilizing the instantiated Brent-Kung adder and radix-4 Booth multiplier.

- 5. Synthesis and Comparison: The synthesis of the Arithmetic Logic Unit (ALU) with and without the use of the Brent-Kung adder and radix-4 Booth multiplier was conducted using the Cadence Genus tool with 45nm technology. This step allowed for a detailed comparison of the performance, area, and power consumption of the ALU designs, highlighting the benefits of incorporating advanced computational units.
- 6. Final Testing and Verification: The final design of the entire FPU was tested using Vivado software. Comprehensive testbenches were written to simulate various scenarios and validate the FPU's performance and accuracy. These testbenches ensured that the integrated FPU operated correctly and met the design specifications under different conditions.

This structured methodology ensured that each component of the FPU was thoroughly tested before integration, and the final design adhered to the IEEE 754 standard. The approach facilitated the development of a high-performance Single Precision Floating Point Unit capable of efficiently handling complex numerical computations while demonstrating the advantages of using advanced arithmetic units.

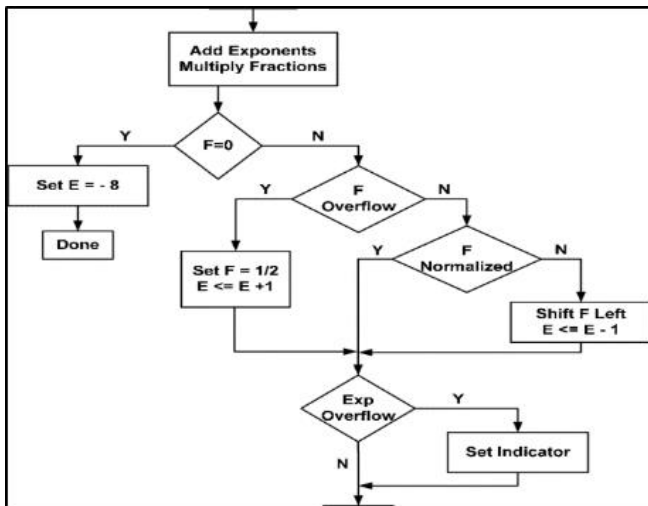


Figure 4.1 Preprocessing for Multiplication/Division

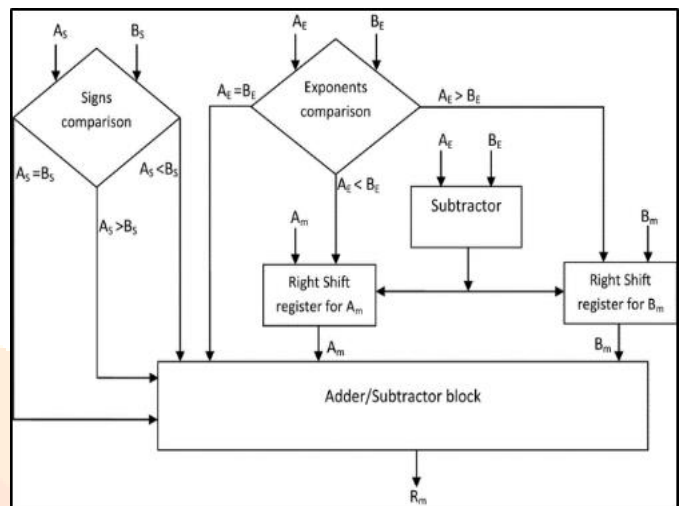


Figure 4.2 Preprocessing for Addition/Subtraction

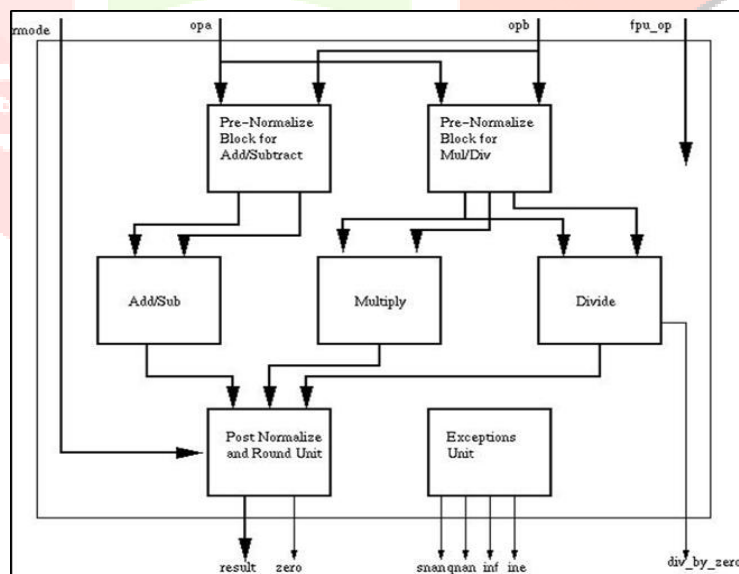


Figure 4.3 Overall Block Diagram of FPU

### 4.1 Cadence Design Tool

Cadence Genus Synthesis Solution is a comprehensive tool used for RTL synthesis and design implementation. It is widely recognized for its capabilities in transforming high-level Verilog or VHDL code into optimized gate-level netlists, specifically targeting ASIC (Application-Specific Integrated Circuit) designs.

Features and Capabilities:

1. High-Level Synthesis (HLS): Cadence Genus facilitates high-level synthesis by converting behavioral descriptions into optimized RTL code. This allows designers to focus on algorithmic design while the tool handles detailed implementation.
2. Optimized RTL Synthesis: The Genus tool excels in generating highly optimized gate-level netlists. It employs advanced algorithms to minimize critical path delays, reduce power consumption, and optimize area usage.
3. Power Analysis and Optimization: Power efficiency is crucial in modern digital designs. Cadence Genus includes robust power analysis tools that help in identifying power hotspots and applying optimizations to reduce overall power consumption.
4. Technology Mapping: The tool supports technology mapping to various process nodes, including the 45nm technology used in this project. This feature ensures that the synthesized design is well-suited for the target fabrication technology.
5. Constraint Handling: Designers can specify various design constraints, such as timing, area, and power budgets. Genus adheres to these constraints during the synthesis process, ensuring that the final design meets all specified requirements.
6. Incremental Synthesis: Cadence Genus supports incremental synthesis, which allows designers to make small changes to the design without having to re-synthesize the entire project. This feature significantly reduces iteration time and accelerates the design cycle.
7. The workflow of using Cadence Genus for RTL synthesis involves several key steps. Initially, the Verilog or VHDL code is imported into the Genus environment. Design constraints, such as timing, area, and power budgets, are then applied. The synthesis process transforms the high-level RTL code into a gate-level netlist. Various optimization techniques are employed to meet performance and design goals. Finally, the optimized gate-level netlist is exported for subsequent design stages, such as place-and-route. This structured approach ensures efficient and accurate digital design synthesis.

## V. RESULTS

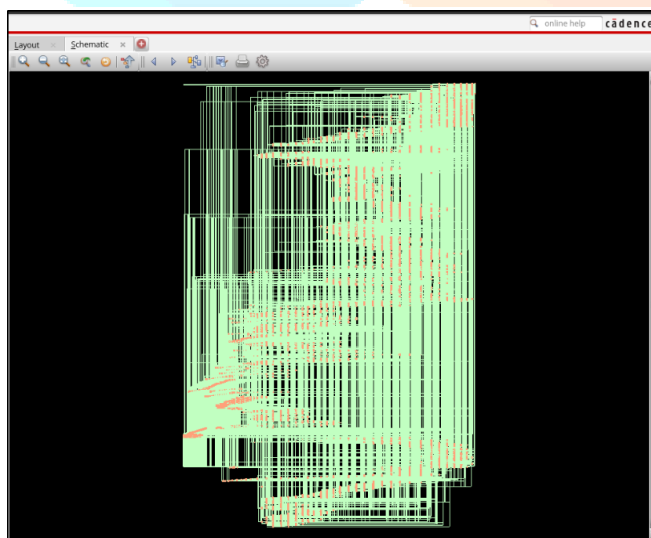


Figure 5.1 Synthesized Design of Floating-Point Unit

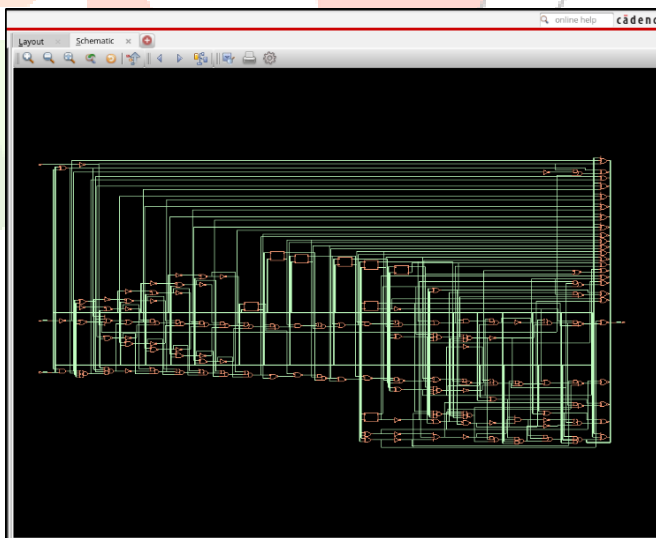


Figure 5.2 Synthesized Design of Brent-Kung Adder

Generic ALU	5401 cells	Proposed FPU	2246 cells
Ripple Carry Adder	140 cells	Brent-Kung Adder	72 cells
Generic Multiplier	1517 cells	Radix4 Booth Multiplier	1471 cells

Generic ALU	37mW	Proposed FPU	0.386mW
Ripple Carry Adder	41.93uW	Brent-Kung Adder	34.33uW
Generic Multiplier	84.5uW	Radix4 Booth Multiplier	98.45uW

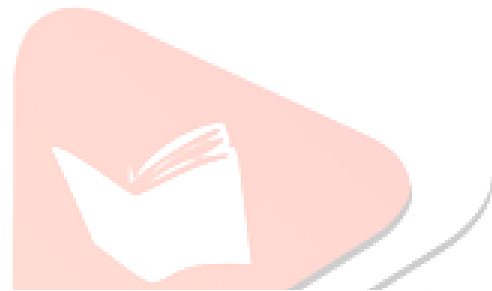
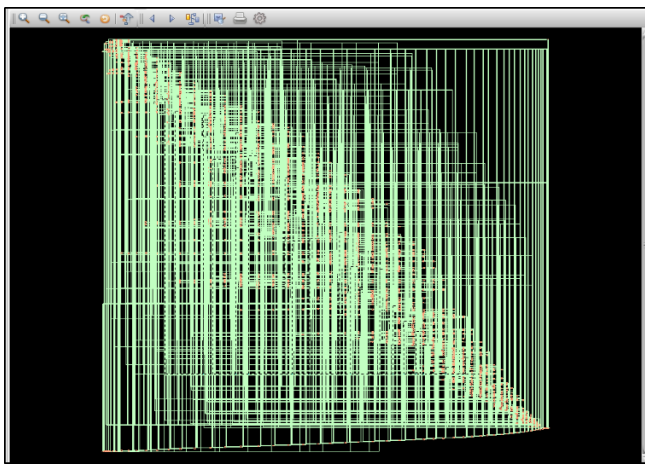


Table 5.1 Comparison of Power Consumption

Table 5.2 Comparison of Area Utilization

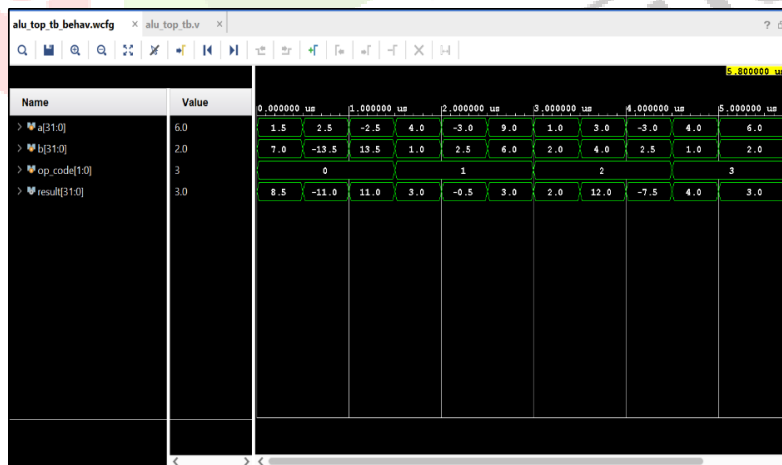


Figure 5.4 Functional Simulation Results

## VI. APPLICATIONS

The Single Precision Floating Point Unit (FPU) designed in this project has significant applications across various fields that demand high-performance computing and precise numerical computations. The integration of the Brent-Kung adder and the radix-4 Booth multiplier enhances its efficiency and performance.

- **Scientific Computing:** In scientific computing, accuracy and speed are paramount. The FPU handles simulations, numerical analysis, and complex calculations in fields like physics, chemistry, and

engineering, ensuring precise and swift computations for tasks such as climate modeling and molecular dynamics.

- **Graphics Processing:** Graphics processing relies heavily on FPUs for real-time rendering of high-quality images and animations. Applications include CAD, video games, virtual reality, and animation. The FPU's high-speed arithmetic operations facilitate complex calculations for shading, lighting, and transformations.
- **Machine Learning and AI:** Machine learning and AI require extensive numerical computations. The FPU's rapid and precise floating-point operations support tasks such as image recognition, natural language processing, and predictive analytics, improving the performance and reliability of machine learning models.
- **Financial Modelling:** In finance, precise numerical calculations are critical for risk assessment, option pricing, and quantitative analysis. The FPU supports financial modelling tasks like Monte Carlo simulations and statistical analysis, aiding financial analysts and traders in making informed decisions.
- **Embedded Systems:** Embedded systems in automotive, aerospace, and telecommunications require efficient floating-point arithmetic for signal processing, control systems, and real-time data analysis. The FPU ensures accurate and fast computations, enhancing the reliability and performance of embedded systems.

## VII. CONCLUSION AND FUTURE TRENDS

### 7.1 Conclusion

The design and implementation of the Single Precision Floating Point Unit (FPU) in this project successfully integrates a Brent-Kung adder and a radix-4 Booth multiplier to enhance arithmetic operation efficiency. This design adheres to the IEEE 754 standard, ensuring accurate and reliable floating-point computations. The methodology involved developing, testing, and integrating individual modules, followed by comprehensive verification using Vivado and synthesis using Cadence Genus with 45nm technology.

The FPU demonstrated significant improvements in performance and accuracy, making it suitable for a wide range of applications, including scientific computing, graphics processing, machine learning, financial modeling, and embedded systems. Each of these fields benefits from the FPU's ability to handle complex numerical computations efficiently.

This project underscores the importance of advanced arithmetic units in modern computing systems. By leveraging the strengths of the Brent-Kung adder and the radix-4 Booth multiplier, the FPU achieves a balance between speed and precision, addressing the computational demands of various high-performance applications. The successful implementation and testing of the FPU highlight its potential as a vital component in future digital design projects.

### 7.2 Future Trends

The future of floating-point unit (FPU) design is set to evolve significantly with several emerging trends:

1. **Semiconductor Technology Scaling:** Process nodes shrinking below 45nm will reduce power consumption, increase speed, and enhance transistor density. This will allow for more complex arithmetic units to be integrated within FPUs, boosting overall performance.
2. **AI and Machine Learning Integration:** FPUs will increasingly handle specialized AI workloads, supporting tensor operations and mixed-precision arithmetic. This adaptation will meet the rising demand for AI acceleration across various industries, such as autonomous vehicles and healthcare.
3. **FPGA and Customizable Hardware Accelerators:** Future FPUs will likely incorporate configurable logic blocks for dynamic adaptation to specific computational tasks. This will optimize performance and power efficiency, leveraging the growing trend of FPGA integration within computing systems.
4. **Quantum Computing:** While not replacing classical FPUs, quantum computing will complement them in solving complex problems currently infeasible with classical computing alone. FPUs will play a crucial role in interfacing classical and quantum systems, ensuring seamless integration and data exchange.
5. **Enhanced Error Correction and Fault Tolerance:** Advancements in error correction and fault tolerance techniques will make FPUs more reliable. This will be particularly important for safety-critical applications such as aerospace, automotive, and medical devices.
6. **Higher Performance and Power Efficiency:** Future FPUs will continue to improve in performance and power efficiency, addressing the growing computational demands of modern applications. Innovations in architecture and design will contribute to these improvements, ensuring FPUs remain at the forefront of high-performance computing.

7. Expanding Application Domains: The advancements in FPU design will expand their impact across various fields, from scientific research to everyday consumer electronics.

As FPUs become more capable, they will enable new applications and technologies, driving further innovation.

In summary, the future trends in FPU design highlight continuous technological advancements, integration with emerging computing paradigms, and enhanced reliability, all of which will drive innovation and expand the influence of FPUs across diverse industries.

## REFERENCES

- [1] N. A. S. Adela, A. N. B. Yousuf and M. M. Eljhani, "Design and Implementation of Single Precision Floating-point Arithmetic Logic Unit for RISC Processor on FPGA," 2023 IEEE 3rd International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA), Benghazi, Libya, 2023, pp. 130-134, doi: 10.1109/MI-STA57575.2023.10169623.
- [2] Samraj Daphni\*Kasinadar Sundari Vijula Grace. "Design and Analysis of 32-bit Parallel Prefix Adders for Low Power VLSI Applications." Advances in Science, Technology and Engineering Systems Journal Vol. 4, No. 2, 102-106 (2019)
- [3] Dr. S V Viraktimath, Khushi S A, Vaishnavi P, W Wiranchi. "Analysis of Parallel Prefix Adders." International Journal for Research in Applied Science & Engineering Technology (IJRASET) ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538 Volume 11 Issue XII Dec 2023
- [4] Asif, Shahzad & Kong, Yinan. (2015). "Performance Analysis of Wallace and Radix-4 Booth-Wallace Multipliers." June 2015 Conference: Electronic System Level Synthesis Conference (ESLsyn)
- [5] Athihrii, M. Stephen and S. Kumar, "Design and implementation of 32-bit ALU using verilog," 02 June 2016.
- [6] D. Malik and R. S. Rathore, "32-bit Arithmetic Logical Unit (ALU) using VHDL," vol. 1, no. 1, 26 NOV 2013.
- [7] R. Cherian, N. Thomas and Y. Shyju, "Implementation of Binary to Floating Point Converter using HDL," no. 461-64, 2013.
- [8] R. Payal, "Simulation and Synthesis Model for the Addition of Single Precision Floating Point Numbers Using Verilog," vol. 02, no. 09, Sep 2013.