WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Design and Verification of 64-Bit Floating-Point Processor using Chipscope

Dr. Girija S[1], Akshatha D[2], Harshitha L Venkat[3]

[1]Assistant Professor, Department of Electronics and Communication Engineering, Dr Ambedkar Institute Of Technology, India
Email: girija.ec@drait.edu.in
[2]Student, Department of Electronics and Communication Engineering, Dr Ambedkar Institute Of Technology, India
Email: akshathadiwalar@gmail.com
[2]Student, Department of Electronics and Communication Engineering , Dr Ambedkar Institute Of Technology, India
Email: harshithalvenkat2511@gmail.com

ABSTRACT:

Contemporary computers employ high-accuracy and wide-range floating-point processors, which play a vital role in scientific computing, artificial intelligence, and signal processing. This paper details the design and validation of a high-performance 64-bit floating-point processor implemented in Verilog, verified using ChipScope. The processor is of modular architecture, including separate modules for addition, subtraction, multiplication, and division, complying with the IEEE-754 standard to ensure compliance with established norms of floating-point arithmetic. It also integrates ChipScope for debugging, signal analysis, and observability of the processor's functionality. Together, simulation results and power consumption analysis have confirmed the design correctness and efficiency.

Keyword: ChipScope; Floating-point arithmetic; IEEE-754; Modular design; Verilog.

1. INTRODUCTION

Floating point units have become essential in any field where accuracy and flexibility are required. These processors permit the performance of arithmetic calculations with excellent precision and comply with the IEEE-754 standard, which regulates the conduct and structure of floating-point values. Almost universally accepted, this standard solves the problems associated with disparate systems and structures by becoming one of the universal principles of engineering calculation systems.

The focus of this work is the design and validation of a 64-bit floating point processor. Its architecture is modular and contains independent modules for the four basic arithmetic operations: plus, minus, times and divide. All modules are constructed to manage edge conditions, overflow and underflow, as well as special circumstances like Not-a-Number (NaN) and infinity. The application of ChipScope for debugging also improves the design, as it allows for any signal to be analyzed in real time, facilitating the design process. The work intends to provide a coherent approach to the workflow of the design, having a particular focus on methods used, results obtained, and problems faced during implementation.

2. RELATED WORKS

The design and optimization of floating-pointprocessors have been widely investigated. Many such works have delivered precious insights on performance, accuracy, and efficiency.The IEEE 754-2019 standard [1] provides the core basis for establishing floating-point arithmetic behavior across all devices and architectures in a well-defined and compatible manner. In [2], the design and simulation of a 32-bit floating-point ALU, following this standard, were described with a focus on modularity and robust handling of special cases. Power-efficient designs, such as the 64-bit floating-point ALU using a block enabling technique, were presented in [3], showing considerable power savings without performance degradation. Implementations of double-precision floating-point arithmetic on FPGAs, with pipeline architectures for throughput improvement, were discussed in [4]. Analogously, optimized non-restoring division algorithms designed for hardware implementation were developed in [5], reporting on the power efficiency and simplicity of this algorithm. Power-performance trade-offs in floating-point units for GPUs are analyzed in [6], which offers a rich source of insights in energy-efficient design for high-performance systems. Verification strategies for IEEE-754 compliance in floating-point units were presented in [8] to emphasize reliability in processor design. These advancements align with the foundational principles of computer architecture outlined in [7], emphasizing quantitative approaches to performance optimization. Together, these works offer a robust framework for developing efficient and standards-compliant floating-point processors.

3. Design

3.1 processor architecture

The 64-bit floating-point processor is built to be modular so as to open the way for scalability and debugging. The processor include four distinct modules of work that can be performed in parallel, each for specific arithmetic operation. These modules are:

- Add/subtractor module: performs Add / Sub operation by aligning their exponents and performing arithmetic on mantissa.
- Multiplication Module: Multiplies the floating-point numbers by using Booth's algorithm
- Division Module: Uses Non-restoring Division to compute accurate quotients
- Control Logic Unit— Controls the modules and enables access to correct operation mode selected via user input.

3.2 adder / subtractor module

The Adder/Subtractor module which is an important section of processor which does arithmetic operations fastly and properly IEEE-754 compliant. The exponents of the two input numbers is aligned in this module. That way the mantissas are matched with respect to scaling for an arithmetic operation. Then add/sub the aligned mantissas according to the operation we want. We normalize the result with a leading Zero detector so that IEEE-754 representation is fulfilled on the final output.

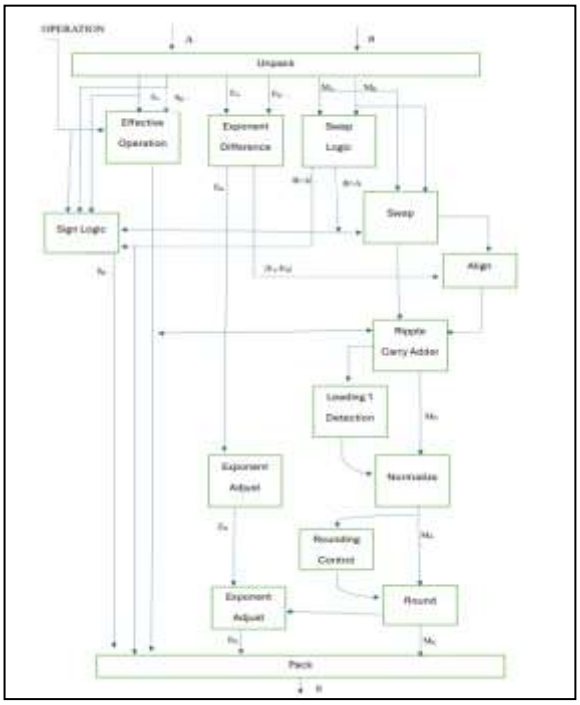

Fig.1. Adder/Subtractor block

Fig .1 depicts data path diagram of a floating-point addition/subtraction unit showcasing the step-by-step processing of unpacking, alignment, arithmetic operation, normalization, rounding, and repacking to produce an IEEE 754-compliant result.

3.3 . Multiplier Module

Multiplier module is used for computing the product of two floating-point numbers The module employ Booth's algorithm for generating partial products in a module. This are destructive in nature, so the partial products are further reduced in a Wallace tree to reduce the number of intermediate additions. Input exponents are added and then the result is nudged back into normalization for output correctness. Output correctness is obtained by detecting and handling overflow, underflow conditions.

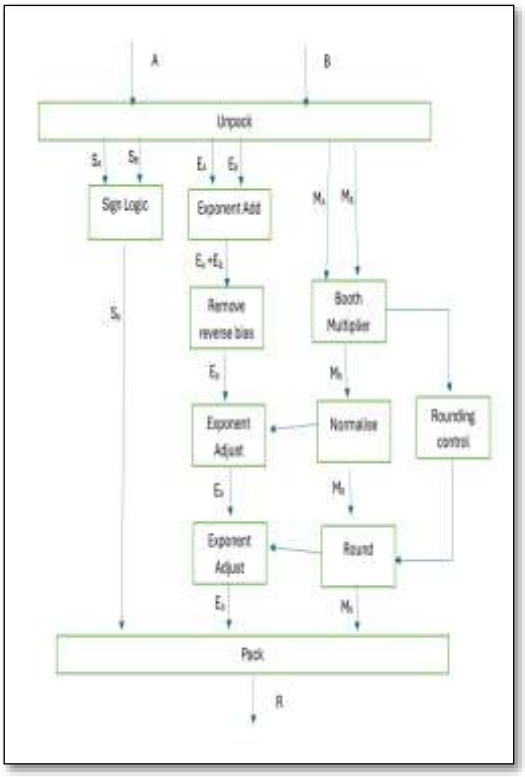

Fig.2. Multiplier block

Fig .2. illustrates the data path of a floating-point multiplication unit detailing the stages from operand unpacking, exponent addition, mantissa multiplication using Booth's algorithm, normalization, rounding, and final packing to produce an IEEE 754-compliant result.

3.4 division module

The division module uses the non-restoring division algorithm to divide two floating point numbers and within it is calculated as quotient. The algorithm does quotient by performing iterative subtraction and shifting operations in order to be exact. As for special cases (e.g. Divide by zero, infinity + real number and so on), IEEE-754 rules are followed The module also normalizes the compute result to fit the standard.

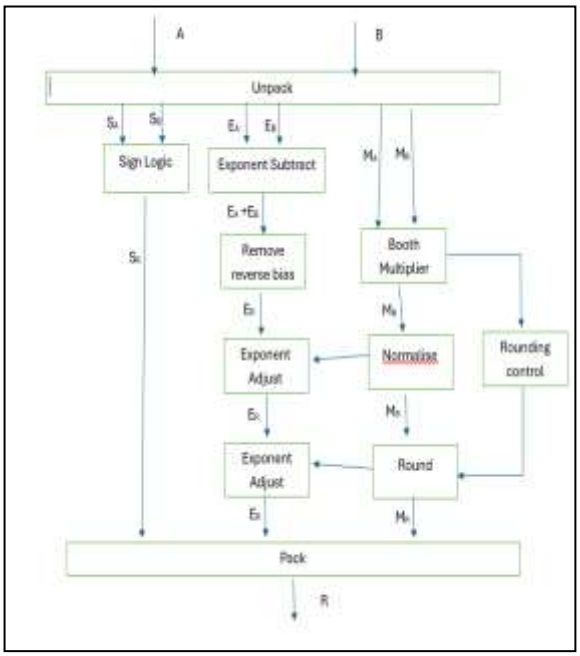

Fig.3. Divider block

Fig. 3 represents the data path of a floating-point division unit ,outlining the sequence of operations including unpacking operands, subtracting exponents, dividing mantissas using a Booth Multiplier, normalization, rounding, exponent adjustment, and packing to generate an IEEE 754-compliant division result.

3.5 top-level integration

Multiplexer-based control logic for the top-level integration that integrates all modules. This design overcomes to, where we can dynamically choose the operation by the user. It integrates that modules can communicate each other in no problem, and real-time monitoring of ChipScope signals on internal signals for debugging.

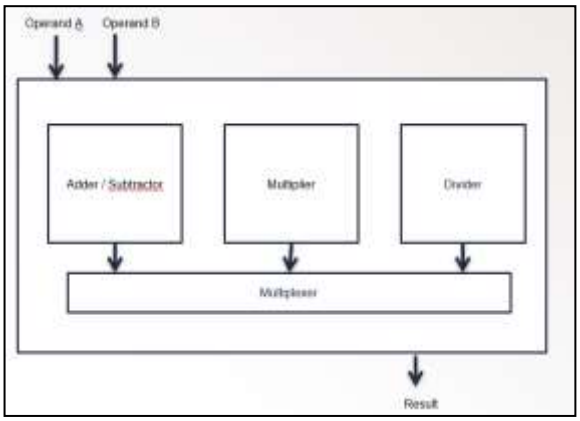

Fig.4. Top Module

Fig .4 illustrates **a** floating-point arithmetic unit (FPU), which takes two operands and performs one of the operations—addition/subtraction, multiplication, or division—based on control logic, with the final result selected using a multiplexer.

3.6 Ieee-754 compliance

The processor adheres to IEEE-754 standards by representing numbers with a sign bit, exponent, and mantissa. The design includes mechanisms to handle special cases such as subnormal numbers, NaN, and infinity. Rounding modes and precision settings are implemented to ensure consistency and accuracy.

3.7 Verification with chipscope

ChipScope is used extensively for verification and debugging. The tool enables real-time monitoring of internal signals, allowing designers to identify and resolve issues such as signal glitches, overflow, and underflow. The integration of ChipScope ensures that the processor's outputs match the expected results for a wide range of test cases.

4. Results

4.1 simulation results

Each module was rigorously tested using a comprehensive set of test cases to validate its functionality and compliance with IEEE-754 standards. The simulation results demonstrated the accuracy and reliability of the processor. Example test cases include:

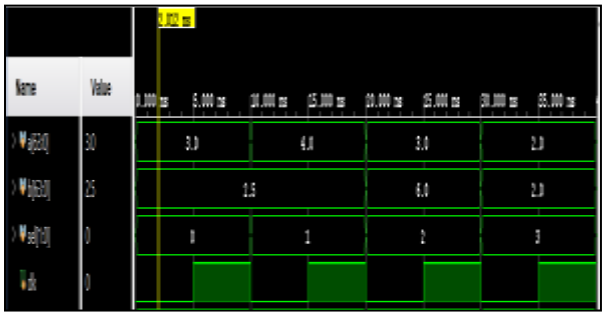

Fig.5. Simulation of Top Module

Fig .5 waveform represents the simulation of a floating-point arithmetic unit (FPU). The waveform shows changes in operands and selection at different clock cycles, verifying the correct operation of the FPU based on the sel value.

4.2 power analysis

The processor's power consumption was analyzed for each module under typical operating conditions. The results are as follows:

- Adder/Subtractor: 0.176 W
- Multiplier: 0.137 W
- Divider: 0.088 W

These results highlight the energy efficiency of the modular design, ensuring suitability for applications requiring low-power consumption.

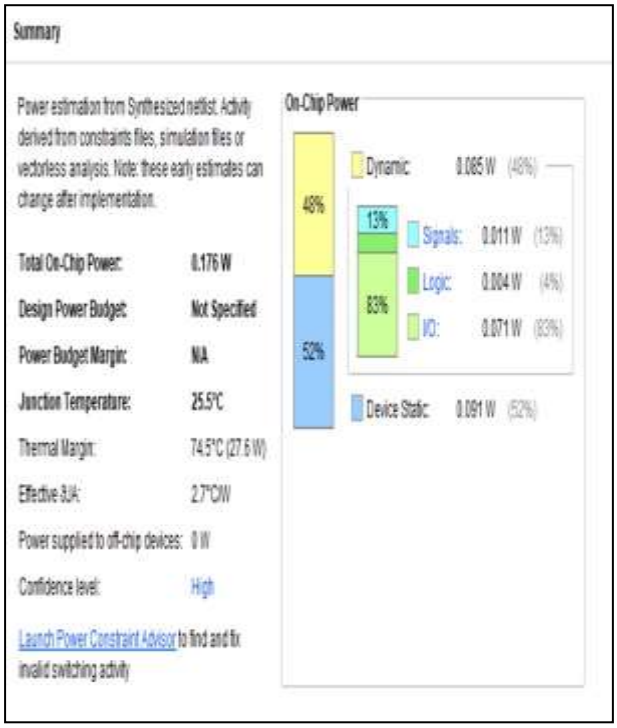

Fig.6. Adder/Subtractor power consumption summary

Fig .6 power estimation report for an FPGA design for adder/subtractor block with a total on-chip power consumption of 0.176 W, comprising **48**% dynamic power (mainly I/O at 83%) and 52% static (device) **power**, under a junction temperature of 25.5°C and a high confidence level in the estimation.

Fig.7. Multiplier power consumption summary

Fig .7 power estimation report for an FPGA design for multiplier block shows a reduced total on-chip power of 0.137 W, with 95% static power (0.131 W) and only 5% dynamic power (0.006 W)**.** The dynamic power is dominated by I/O (74%). The confidence level **is** Medium, and the process is marked as typical**.**

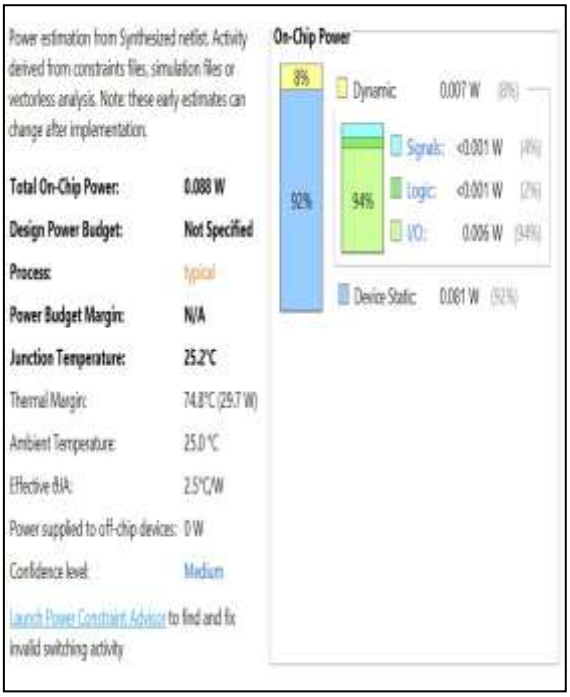

Fig. 8.Divider power consumption summary

Fig .8 power estimation report for an FPGA design for divider block .The total on-chip power reduced from 0.137 W to 0.088 W due to a significant drop in static power. I/O dynamic power increased to 94%, indicating higher external interfacing activity. Overall, the design became more power-efficient and thermally stable.

Fig.9. Top module power consumption summary

Fig .9 power estimation report for an FPGA design for top module.

4.3 Debugging with chipscope

ChipScope played a crucial role in debugging and validation. The tool provided insights into signal propagation and timing issues, enabling the identification and resolution of potential problems. The real-time monitoring capabilities of ChipScope ensured that the processor met the required performance criteria.

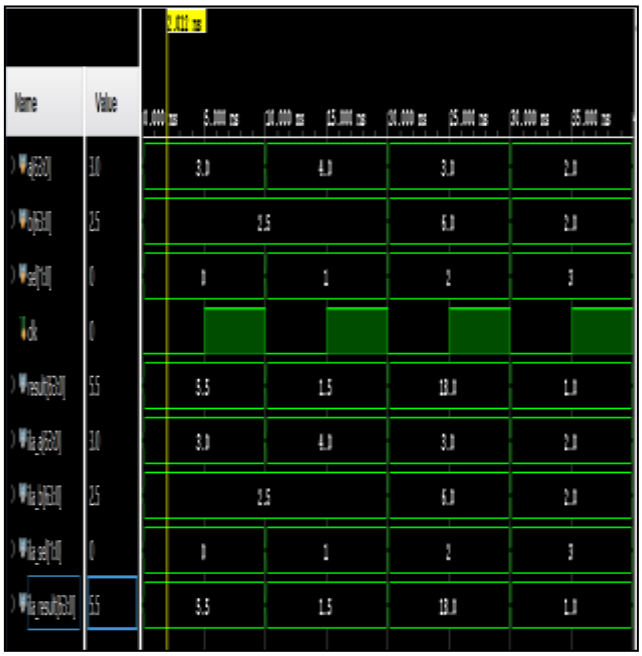

Fig.10. Signal monitoring using ILA

Fig.10 illustrates signal monitoring using an Integrated Logic Analyzer (ILA) in an FPGA development environment . This confirms the arithmetic MUX is functioning correctly across all sel values.

Type of circuit	**Power consumption by designed circuits**
Adder/Subtractor	**0.176W**
Multiplier	**0.137W**
Divider	**0.088W**

Table.1. Power consumption of Designed circuits

The power consumption study showed that the Adder/Subtractor is the most power-hungry circuit with 0.176W, while the Multiplier with 0.137W came next, and the Divider with 0.088W showed the lowest. Hence, the Divider has the lowest power consumption and is the energy-efficient design for the assessed arithmetic units.

5. Conclusion

In the paper, this work detailed design and verification for a 64-bit floating-point processor. By implementing a modular architecture with strict compliance to IEEE-754 standards, it ensured precise and reliable output. ChipScope integration helped ease debugging and validation processes, ensuring the processor to be used for high-performance applications. Future research will be to optimize the power efficiency of this design and then look into more applied applications in advanced domains like AI and machine learning.

References

[1] IEEE Standard for Floating-Point Arithmetic, IEEE 754-2019.

[2] Savaliya, Y., Rudani, J., "Design and Simulation of 32-Bit Floating-Point ALU," IRJET, 2020.

[3] Sekhar, K. C., "Power Optimized 64-Bit Floating-Point ALU," 2020.

[4] Paschalakis, P., "Double Precision Floating-Point Arithmetic on FPGAs," IEEE, 2019.

[5] Dineen, T., "Efficient Non-Restoring Division Algorithms for Hardware Implementation," Journal of Digital Systems, 2018.

[6] Patil, D., et al., "Power-Performance Analysis of Floating-Point Units in GPUs," IEEE Transactions on Computers, 2020.

[7] Hennessy, J. L., Patterson, D. A., "Computer Architecture: A Quantitative Approach," Morgan Kaufmann, 2021.

[8] Smith, J., "Verification Strategies for IEEE-754 Compliant Floating-Point Units," VLSI Design Journal, 2020.

[9] V. Patil ,"A Review On Comparative Analysis Of 16-Bit And 32-Bit RISC Processors," vol. 12, no. 6, pp. 1-8, June 2024.

[10] M. Zhang, "Design of Generic Floating Point Pipeline Based Arithmetic Operation for DSP Processor," vol. 44, no. 2, pp. 123-130, Feb. 2023.