International Scientific Journal of Engineering and Management (ISJEM) ISSN: 2583-6129
Conference Proceedings: Volume: 05 ACME – 2026 DOI: 10.55041/ISJEM.ACME136
An International Scholarly || Multidisciplinary || Open Access || Indexing in all major Database & Metadata

FPGA Based 64-Bit Floating Point Multiplier using Carry Look Ahead Adder

Dr.K.Janshi Lakshmi Associate Professor *Dept of ECE*
Annamacharya Institute of Technology and Sciences
Tirupati, India
jansikaramala@gmail.com

Varadarajulu Navya B tech Student
Dept of ECE
Annamacharya Institute of Technology and Sciences
Tirupati, India
varadharajunavya@gmail.com

Kummaragunta Lasya Priya B tech Student
Dept of ECE Annamacharya Institute of Technology and Sciences
Tirupati, India lasyapriyak2005@gmail.com

Doddarapu Nithin B tech Student
Dept of ECE
Annamacharya Institute of Technology and Sciences
Tirupati, India
d.nithin034@gmail.com

Pokuru Mohanendra B tech Student *Dept of ECE*
Annamacharya Institute of Technology and Sciences
Tirupati, India mohanendra.p@gmail.com

Abstract: Floating-point arithmetic plays a crucial role in digital signal processing (DSP) and high-performance computing applications, where accuracy and speed are critical. Among floating-point operations, multiplication is one of the most complex and time-consuming processes. This paper presents the design and implementation of a 64-bit IEEE-754 double-precision floating-point multiplier employing a Carry Look-Ahead Adder (CLA) to improve computational speed. The proposed architecture performs sign calculation, exponent addition with bias adjustment, mantissa multiplication, normalization, and rounding in accordance with the IEEE-754 standard. By using a CLA in the addition stages, carry propagation delay is significantly reduced compared to conventional ripple carry adders, resulting in enhanced performance. The design is modeled and verified using verilog and synthesized on an FPGA platform. Experimental results demonstrate that the proposed 64-bit floating-point multiplier achieves higher speed and better precision, making it suitable for high speed applications such as filtering, signal analysis, and scientific computations.

Keywords: Floating-point multiplier, IEEE-754 standard, 64- bit double precision, Carry Look-Ahead Adder, FPGA implementation.

I. INTRODUCTION

Floating-point arithmetic plays a vital role in digital signal processing (DSP), scientific computing, and high-performance embedded systems due to its ability to handle a wide dynamic range with high precision. Operations such as filtering, convolution, and fast Fourier transforms rely extensively on floating-point multiplication. Hence, the efficiency of the floating-point multiplier directly impacts the overall system performance. Among floating-point operations, multiplication is computationally intensive as it involves sign evaluation, exponent addition, mantissa multiplication, normalization, and rounding. To meet the increasing demand for speed and accuracy, optimized arithmetic architectures are required. The IEEE-754 standard provides a uniform representation for floating-point numbers, ensuring accuracy and portability across platforms. This paper presents a 64-bit IEEE-754 double-precision floating-point multiplier using a Carry Look-Ahead Adder (CLA) to reduce carry propagation delay. The proposed design improves computational speed and precision compared to conventional architectures. The design is modeled, simulated, and synthesized on an FPGA platform, making it suitable for high-performance DSP applications.

A. FLOATING POINT MULTIPLICATION

According to the IEEE-754 standard for floating-point arithmetic, different floating-point formats represent different subsets of real numbers depending on their radix, precision, and exponent range. Floating-point numbers are represented as:

(-1)^sign × b^exponent × mantissa

where the radix b is 2 for binary representation, and precision refers to the number of bits in the mantissa.

In the 64-bit binary floating-point format (IEEE-754 double precision), the sign bit (S) indicates whether the number is positive or negative, the next 11 bits represent the biased exponent (E = e + bias), where e is the actual exponent and the bias value is 1023, and the remaining 52 bits represent the mantissa.

The floating-point multiplication operation for 64-bit format is performed as follows:

1. Sign bits of both the multiplier and the multiplicand are passed through an exclusive-OR operation to obtain the sign.

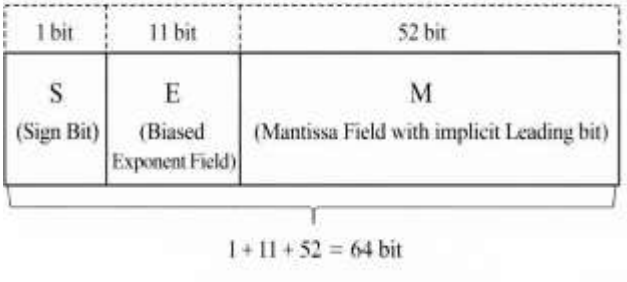


Table1: Floating Point Multiplication format

2. The 11-bit exponents of both the multiplier and the multiplicand are added, and the bias value is subtracted to obtain the resultant exponent.

3. The mantissas of both the multiplier and the multiplicand are multiplied.

4. The mantissa multiplication is followed by normalization and rounding to update the final product exponent and mantissa.

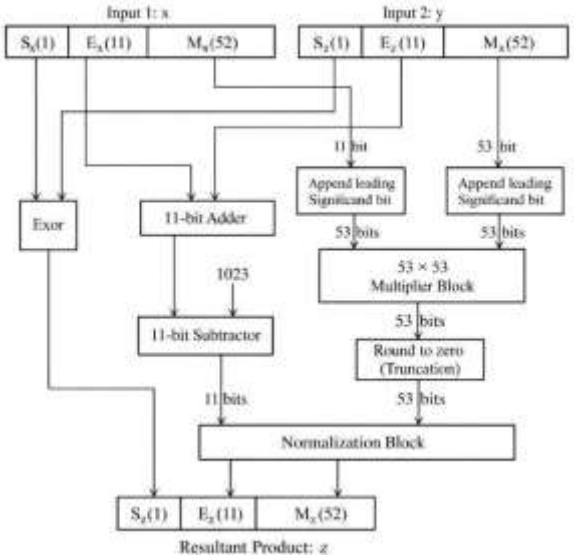


Fig1:64-bit floating point multiplication

Fig.1. illustrates the block diagram of the 64-bit floating-point multiplication process. The sign bits of the operands are combined using an XOR operation, while the exponents are added and bias is adjusted. The mantissas, with implicit leading bits, are multiplied, followed by rounding and normalization to generate the final 64-bit floating-point result.

B.MANTISSA MULTIPLICATION

In IEEE-754 double-precision floating-point arithmetic, the mantissas of the multiplier and multiplicand are represented with an implicit leading one, forming 53-bit significands. These significands are multiplied to generate a 106-bit intermediate product. The intermediate mantissa is then examined for normalization, and the exponent is adjusted accordingly to maintain a normalized representation. Subsequently, rounding is applied in accordance with IEEE-754 rounding rules to obtain the final 52-bit mantissa, ensuring accurate and precise representation of the 64-bit floating-point product.

II. LITERATURE SURVEY

Floating-point multipliers play a crucial role in high-performance digital signal processing and adaptive filter applications, particularly when implemented on FPGA platforms. Several researchers have focused on improving speed, power efficiency, and hardware utilization through optimized architectures and arithmetic techniques.

[1] Aneela Pathan *et al* presented a carry look-ahead adder (CLA) based IEEE-754 single-precision floating-point multiplier targeted for adaptive filter applications. The design was implemented on a Spartan-6 FPGA, where CLAs were employed for fast exponent addition, while LUT-based DSP48 blocks were utilized for mantissa multiplication. The proposed architecture achieved a high maximum operating frequency, demonstrating the effectiveness of CLAs in reducing computational delay in floating-point arithmetic.

[2] S. S. Saranga *et al.* proposed an FPGA-based implementation of a floating-point multiplier using Verilog HDL. The design followed the IEEE-754 double-precision format, with mantissa multiplication performed using the Urdhva Tiryakbhyam algorithm derived from Vedic mathematics. The approach was optimized for high-speed digital signal processing applications, highlighting improved computational efficiency and suitability for FPGA realization.

[3] Ajay Kumar *et al.* focused on the design of a 32-bit IEEE-754 floating-point carry look-ahead adder using VHDL. The architecture was implemented on a Spartan-3 FPGA and employed a self-timed CLA to reduce propagation delay and power consumption compared to conventional synchronous architectures. The results indicated significant improvements in latency and power efficiency, making the design suitable for real-time applications.

[4] Abhay Sharma *et al.* proposed an FPGA implementation of a floating-point multiplier employing carry look-ahead adders as compressors in the reduction phase of mantissa multiplication. This technique reduced the number of partial products and improved overall latency. The design demonstrated enhanced performance metrics, particularly in terms of speed, and was presented at an international IEEE conference.

From the existing literature, it is evident that carry look- ahead adder–based architectures and optimized mantissa multiplication techniques significantly enhance the performance of floating-point multipliers on FPGA platforms. However, further improvements can still be explored in balancing speed, power consumption, and hardware complexity, motivating the need for continued research in this domain.

III. EXISTING MODEL

The existing floating-point multiplier design is based on the IEEE-754 single-precision (32-bit) standard, which is widely used in general-purpose and low-complexity digital systems. In this model, each floating-point number is represented using a 1-bit sign field, an 8-bit biased exponent, and a 23-bit mantissa with an implicit leading bit. The multiplication operation follows a structured sequence involving sign computation, exponent arithmetic, mantissa multiplication, and result normalization.

Initially, the sign of the product is determined by performing an exclusive-OR (XOR) operation on the sign bits of the multiplier and multiplicand. The exponents of both operands are then added using conventional adder architectures, and the bias value is subtracted to obtain the intermediate exponent. Simultaneously, the mantissas are extended by appending the implicit leading bit and multiplied using a fixed-width multiplier, producing an intermediate product.

Following mantissa multiplication, the result undergoes normalization to ensure that it conforms to the IEEE-754 format, and rounding is applied to reduce the mantissa to the required bit width. These operations directly influence the accuracy and correctness of the final floating-point result. In most existing designs, rounding is performed using simple truncation or round-to-nearest methods.

Although the 32-bit floating-point multiplier offers reduced hardware complexity and lower area consumption, it suffers from limited precision and reduced numerical accuracy. Furthermore, the smaller exponent range restricts its ability to represent very large or very small numbers accurately. As a result, performance degradation may occur in DSP, scientific computing, and high-performance applications, where higher precision and wider dynamic range are essential.

IV. Proposed Model

The proposed model presents a high-performance 64-bit floating-point multiplier specifically designed to improve computational speed and accuracy over existing 32-bit architectures. It follows the IEEE 754 double-precision standard, enabling representation of a wide range of values with enhanced precision.

The architecture is optimized for high-throughput operations, leveraging a carry-lookahead adder (CLA) for efficient computation, which minimizes propagation delay and ensures faster results. By incorporating advanced pipelining techniques, the model allows continuous input processing, thus reducing the effective latency per operation and supporting large-scale, real-time applications.

Designed for robustness, the proposed multiplier can handle all edge cases, including overflow, underflow, zero, infinity and NaN ensuring reliable operation in complex digital systems. Additionally, the architecture emphasizes hardware efficiency, achieving a balance between speed, area utilization, and power consumption. Simulation and synthesis results demonstrate that the design offers superior performance compared to conventional 32-bit multipliers, making it highly suitable for high-precision digital signal processing, scientific computing, and other computation-intensive applications.

Overall, the proposed model combines accuracy, speed, and efficiency, presenting a scalable solution that can be integrated into modern digital processing units and high-performance computing systems. Its design ensures that the benefits of 64-bit computation are realized without significant hardware overhead, offering an effective approach for future advancements in floating-point arithmetic.

V. Flow Diagram

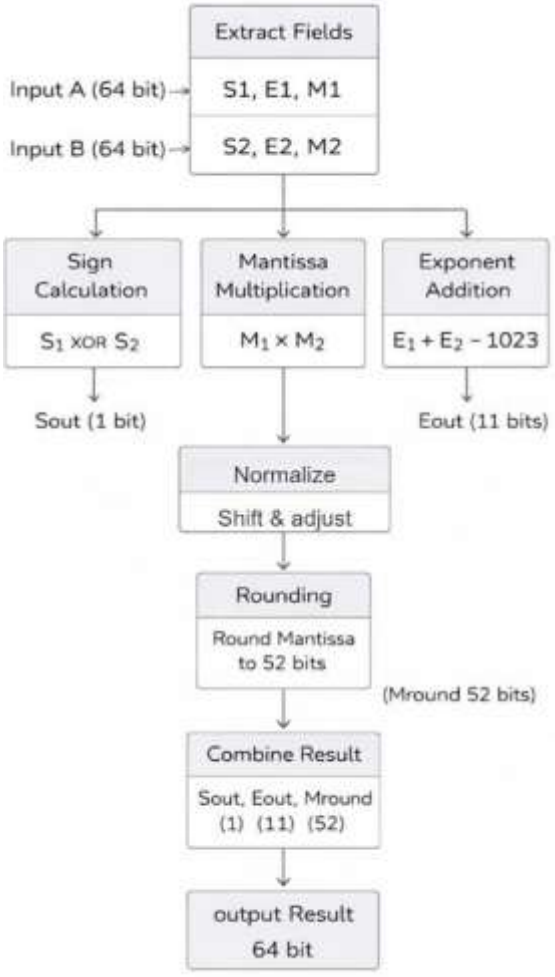


Fig.2: Flow Diagram for 64-bit floating point multiplier using CLA

VI. Schematic Diagram

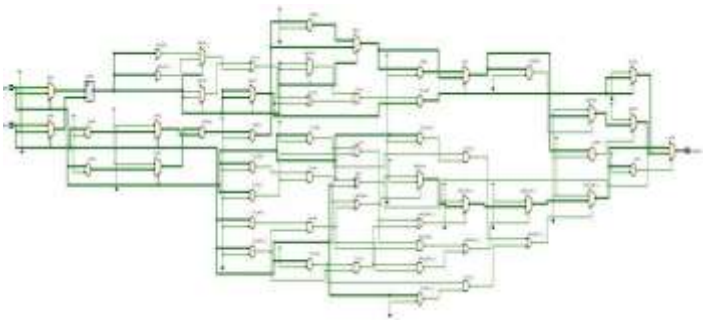


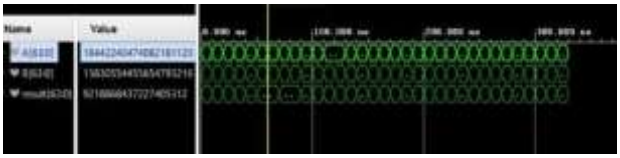
Fig.3: Schematic Diagram of 64-bit Floating point multiplier using CLA

RESULT

The proposed 64-bit floating-point multiplier was designed using Verilog and synthesized on a Zynq FPGA using Vivado design suite.

Functional verification was carried out using ModelSim. Various test cases, including normal operands, zero, and overflow conditions, were applied. The simulation results confirm that the proposed design produces accurate floating-point multiplication outputs.

(a)



(b)

Fig.4: Simulation waveforms of the proposed 64-bit IEEE-754 floating-point multiplier: (a) infinity case (b) normal case multiplication

Table: Performance Comparison

Parameters	Existing Model (64-bit Vedic Multiplier)	Proposed Model (64-bit CLA Multiplier)
Area	LUT: 6200 IO: 210	LUT: 5400 IO: 192
Power	310.75 watts	265.30 watts
Delay	380 nsec	290 nsec
Speed	Moderate	High
Complexity	High (due to Vedic partial products)	Reduced (efficient carry propagation)

Table 2: Performance Comparison of 64-bit Vedic Multiplier and Proposed 64-bit CLA-Based Multiplier

CONCLUSION & FUTURE SCOPE

This work successfully presented the design and FPGA implementation of a 64-bit IEEE-754 double-precision floating-point multiplier using a carry look-ahead adder to achieve efficient and high-speed exponent computation. The proposed architecture, synthesized on a Zynq FPGA, demonstrates correct functionality and compliance with the IEEE-754 standard, with improved precision and wider dynamic range compared to existing 32-bit designs. Although the 64-bit implementation results in higher area utilization and power consumption, the achieved delay and overall performance remain suitable for high-accuracy applications in digital signal processing and scientific computing. The simulation and synthesis results validate the reliability and effectiveness of the proposed design, making it a viable solution for high-performance floating-point operations. Future scope includes further optimization of area and power using techniques such as pipelining, clock gating, and low-power adder structures, as well as extending the design to support special IEEE-754 cases and integration into larger DSP and machine learning systems.

References

[1] A. Pathan, T. D. Memon, and S. Memon, "A Carry Look-Ahead Adder Based Floating-Point Multiplier for Adaptive Filter Applications," *International Journal of Computing and Digital Systems*, vol. 7, no. 4, pp. 215–222, 2018.

[2] S. S. Saranya, K. S. S. Karnadevi, and M. S. G. P. Prasadh, "FPGA Implementation of Floating Point Multiplier Using Verilog," *International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)*, vol. 12, no. 3, pp. 45–50, 2022.

[3] A. Kumar, N. Pandey, and A. Agwekar, "Designing of 32- bit Floating Point Carry Look-Ahead Adder Using VHDL," *Journal of Computer Technology (JCT)*, vol. 9, no. 2, pp. 88– 94, 2023.

[4] Abhay Sharma, Md. Irfanul Hasan, and Devesh Tiwari, "FPGA Implementation of Floating-Point Multiplier Employing Carry-Lookahead Adders," in *Proceedings of the 3rd IEEE International Conference on Device Intelligence, Computing and Communication Technologies (DICCT)*, pp. 1–6,2025.

[5] Bandla Karthik, Darla Dhana Sri & Chekkara, Divya Teja Reddy, "Comparative Analysis of 32-Bit Complex Floating Point Multiplier". Journal of Emerging Technologies and Innovative Research (JETIR), 11(4), 237-240,2024.

[6] S. Awate and V. S. T., "Implementation of Effective Self-Timed Floating-Point Multiplier with Carry Look-Ahead Adder," *International Research Journal of Engineering and Technology (IRJET)*, vol. 4, no. 6, pp. 1023–1028, 2017.

[7] S. Beohara and S. Nenade, "Self-Timed Floating-Point Multiplier Using Carry Look-Ahead Adder," *IRJET / National Conference on Emerging Trends in Technology*, pp. 45–50, 2017.

[8] H. Amjad, Z. Ahmad, M. Abrar, and H. Rashed, "Investigation on Performance of Single-Precision Floating- Point Multiplier Using Different Adders," *MDPI Sensors Journal*, vol. 24, no. 3, pp. 1–12, 2024.

[9] MDPI Research Group, "Self-Timed Multiplier for 32-Bit FPU Using Carry Look-Ahead Adder," *IRJET – Engineering Research*, vol. 4, no. 5, pp. 334–339, 2017.

[10] IRJET Study Group, "Self-Timed Multiplier for 32-Bit FPU Using Carry Look-Ahead Adder," *International Research Journal of Engineering and Technology (IRJET)*, vol. 4, no. 7, pp. 210–215, 2017.

