

Implementation of High-Speed Area Efficient Multiplier Using 7:3 Compressors

G.Harika^{1,*}, Chennakeswari Pappula², Swathi Mattaparthi³, Vasantha Savitri Rameswarapu⁴ and B.S. Vinay Mulaparthi⁵

¹⁻⁵Department of ECE, BVCITS, Amalapuram, AP-533221, India.

*Corresponding Author: G.Harika. Email: harikabvts@gmail.com

Received: 07/02/2026; Revised: 28/02/2026; Accepted: 21/03/2026; Published: 31/03/2026.

DOI: <https://doi.org/10.69996/fmep.2026003>

Abstract: When it comes to processing digital signals, images, and communications, multiplication is a crucial arithmetic operation. How well these systems work is highly dependent on how fast and efficient the multiplier unit is. Array multipliers and Wallace tree multipliers are two examples of conventional multiplier designs that, as the number of bits rises, typically experience increased hardware complexity and substantial propagation delays. This research suggests a 7:3 compressor-based, space-efficient multiplier for partial product reduction to circumvent these restrictions. The proposed architecture generates partial products and reduces them using compressor circuits, which significantly decreases the number of reduction stages and improves computational speed. A Carry Lookahead Adder (CLA) is used in the final addition stage to further reduce carry propagation delay. The design is implemented using VHDL and analyzed through FPGA synthesis tools to evaluate its performance. Experimental results show that the proposed multiplier achieves improved speed, reduced area utilization, and lower power consumption compared to conventional multiplier architectures, making it suitable for high-performance digital and DSP applications.

Keywords: High-Speed Multiplier, 7:3 Compressor, Partial Product Reduction, Carry Look ahead Adder (CLA), Vedic Multiplier.

1.Introduction

Multipliers are widely used in modern computer applications like image processing, machine learning, and Digital Signal Processing (DSP). Due to the high number of arithmetic operations required by these applications, computationally costly tasks such as Fast Fourier Transform (FFT), Discrete Cosine Transform (DCT), and Convolutional Neural Networks (CNNs) rely heavily on multiplication [1-4]. Unfortunately, the efficiency and performance of the system are negatively impacted by the fact that accurate multipliers increase computational latency and use a lot of power and space. In response to this issue, researchers have developed approximate multipliers that are well-suited for error-tolerant applications due to their reduced power consumption, area, and delay while maintaining high accuracy [5-7].

Approximate computation approaches can greatly enhance system performance, as recently shown by arithmetic optimization [8]. Approximate multipliers reduce power dissipation and circuit complexity while maintaining adequate accuracy by easing the necessity for exact results. By minimizing the amount of logic gates and critical path time, many designs have been developed to enhance multiplication processes [9-11]. To construct approximation multipliers, truncation, bit-width reduction, and error-resilient arithmetic modifications are the main methods.

Appropriate multipliers are popular in areas where near-perfect accuracy is not critical, such as neuromorphic computing, real-time image processing, and digital signal processor accelerators, despite the fact that they cause tiny computational errors. To find a happy medium



between precision and efficiency, researchers have come up with a number of approximation solutions, such as fused add-multiply (FAM) units and approximate Booth encoding [12-14]. Various approximation strategies have been investigated in recent studies with the aim of improving performance in targeted applications by reducing error and increasing energy economy [15-17].

How well a multiplier works is a major factor in how fast a processor is. Consequently, there is a growing need for compact and efficient multipliers that can handle high speeds while consuming minimal power. The last several decades have seen the development and exploration of numerous novel multiplier designs. The multipliers' function is the foundation of processor development. A number of requirements, such as minimal power consumption and area occupied, should be satisfied by the developed multiplier. Over the past few years, we have seen the introduction and successful implementation of new, redesigned multiplier designs. The circuit's lock time is brief, and its factor range for multiplication and division is unregulated. To keep the circuit's total power consumption low, a short power approach has been used.

Optimized multiplication procedures with reduced logic complexity and efficient arithmetic encoding strategies are necessary for high-speed computing applications. This article compares and contrasts traditional multipliers with several approximation multiplier designs using performance criteria like power consumption, area occupied, and delay. We want to significantly improve computing efficiency while developing an efficient multiplier architecture that is suitable for error-resilient applications.

2. Literature Survey

2.1 Conventional Multiplier Architectures

Traditional multiplier designs such as Array Multipliers, Booth Multipliers, and Wallace Tree Multipliers have been widely used in digital signal processing and arithmetic logic units. These architectures mainly focus on generating partial products and reducing them using different adder structures.

Array multipliers provide a simple and regular layout, which makes them suitable for hardware implementation. However, they suffer from large propagation delay because the addition of partial products occurs sequentially.

Although these architectures improve multiplication performance compared to basic designs, they still face limitations in speed, hardware utilization, and power consumption for large bit-width operations.

2.2 Compressor-Based Multiplier Designs

To overcome the limitations of conventional multipliers, researchers introduced compressor-based reduction techniques. Compressors are digital circuits that combine multiple input bits into fewer output bits, thereby reducing the number of reduction stages in multiplication.

Common compressors include 3:2, 4:2, and higher-order compressors, which are widely used in high-speed multipliers such as Wallace and Dadda multipliers. Compressor-based multiplier architectures therefore provide better speed, area efficiency, and power optimization, making them suitable for modern DSP, VLSI, and high-performance computing systems.

3. Proposed Multiplier Architecture

The proposed multiplier architecture focuses on achieving high speed, reduced area, and low power consumption by combining three important techniques:

- A. Vedic multiplication (Urdhva-Tiryakbhyam algorithm)

- B. 7:3 compressor based partial product reduction
- C. Efficient final addition stage

The multiplier is implemented for an 8×8 multiplication operation and designed using VHDL in Xilinx Vivado. The architecture reduces the delay by minimizing the number of addition stages required during partial product reduction.

1. Overall Architecture of Proposed Multiplier

The architecture of the proposed multiplier consists of three main stages.

1. Partial Product Generation
2. Compressor Based Partial Product Reduction
3. Final Carry Propagation Addition

The Vedic multiplication algorithm generates partial products using a vertical and crosswise multiplication technique, which allows parallel generation of partial products

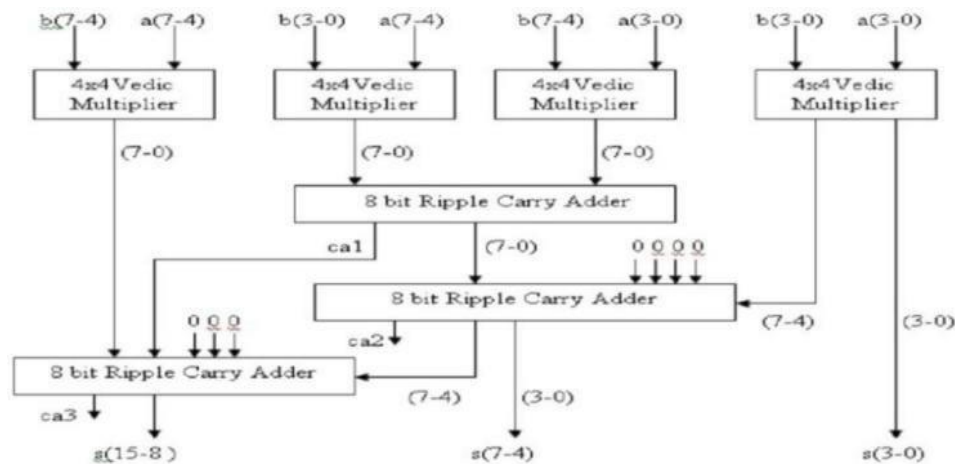


Figure 1: 8 bit Ripple Carry Adder

2. Partial Product Generation Using Vedic Algorithm

The Urdhva-Tiryakbhyam method from Vedic mathematics is used for generating partial products. This algorithm performs multiplication using vertical and crosswise operations, which allows multiple partial products to be generated simultaneously.

Example of Vedic Multiplication (4×4 Concept)

Let $A_3 A_2 A_1 A_0, B_3 B_2 B_1 B_0$

Step1: $A_0 \times B_0$

Step2: $(A_1 \times B_0) + (A_0 \times B_1)$

Step3: $(A_2 \times B_0) + (A_1 \times B_1) + (A_0 \times B_2)$

Step4: $(A_3 \times B_0) + (A_2 \times B_1) + (A_1 \times B_2) + (A_0 \times B_3)$

This approach produces partial products in parallel, improving the speed of multiplication.

3. Partial Product Reduction Using 7:3 Compressors

Before moving on to the last step of addition, the partially produced items must first be reduced to two rows. This is achieved using 7:3 compressors.

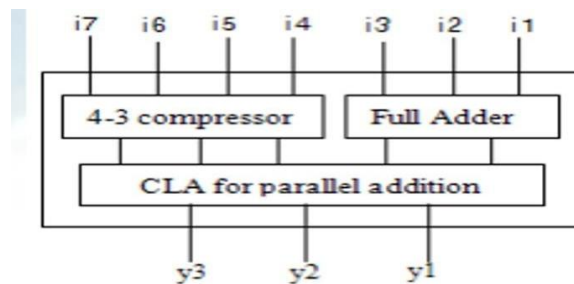


Figure 2: 7:3 Compressor

A 7:3 compressor accepts seven input bits and produces three output bits, thereby reducing the number of addition levels.

This approach produces partial products in parallel, improving the speed of multiplication.

4. Carry Look ahead Adder(CLA)

To circumvent the carry propagation delay in more traditional adders, including tripple carry adders, digital arithmetic circuits employ a Carry Lookahead Adder (CLA), a high-speed adder. As the number of bits increases, the total delay in ripple carry adders increases because each bit must wait for the carry from the previous step. The CLA eliminates this problem by calculating carry signals in advance using generate and propagate functions. This significantly improves the speed of addition operations in high-performance multipliers and processors.

In multiplier architectures, especially those using compressor-based reduction techniques, the final stage requires a fast adder to combine the reduced partial products. The CLA is commonly used in this stage because it minimizes propagation delay and improves overall system performance.

Generate and Propagate Logic

The CLA uses two key signals to determine carry values: Generate (G)

A carry is generated when both input bits are 1.

- $G_i = A_i \cdot B_i$ Propagate (P)

A carry is propagated when at least one input bit is 1.

- $P_i = A_i \oplus B_i$

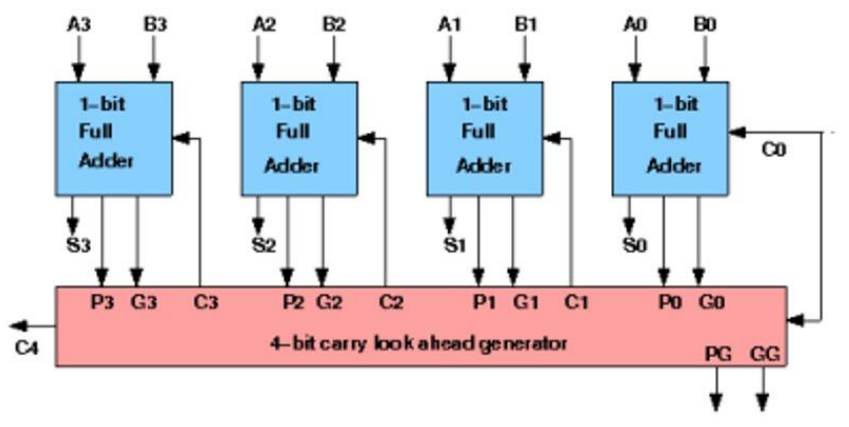


Figure 3: Adder

Using these signals, the carry output can be computed without waiting for previous carries.

Carry Calculation For a 4-bit CLA:

- $C1 = G0 + P0C0$
- $C2 = G1 + P1G0 + P1P0C0$
- $C3 = G2 + P2G1 + P2P1G0 + P2P1P0C0$
- $C4 = G3 + P3G2 + P3P2G1 + P3P2P1G0 + P3P2P1P0C0$

This parallel carry generation reduces the delay significantly compared to ripple carry adders.

5.Final Addition Stage

After compression, two rows of partial sums remain. These are combined using a carry propagation adder to produce the final product.

4. Experimental Results and Discussions

In order to assess its performance in terms of speed, area usage, and power efficiency, the suggested multiplier design was put into action and evaluated utilizing 7:3 compressors and a Carry Lookahead Adder (CLA). The design was tested for hardware efficiency and functional correctness using Xilinx Vivado and VHDL, respectively.

The experimental evaluation focuses on comparing the proposed multiplier with conventional multiplier architectures such as array multipliers and Wallace tree multipliers. The comparison is performed based on three important parameters: propagation delay, logic utilization, and computational efficiency.

RTL Analysis

RTL (Register Transfer Level) analysis is used to examine the internal structure and functionality of the proposed multiplier design before hardware implementation. It represents the logical behavior of the circuit in terms of registers, combinational logic, and data flow between different modules. The RTL view helps designers verify whether the multiplier architecture operates correctly according to the specified design. In the proposed 7:3 compressor based multiplier, the RTL schematic shows the interaction between the major modules such as partial product generation, compressor reduction network, and final CLA adder.

The RTL analysis confirms that the proposed architecture correctly implements the multiplication process while maintaining efficient data flow and reduced logic depth. This verification ensures that the design is suitable for further synthesis and hardware implementation in FPGA or VLSI systems”

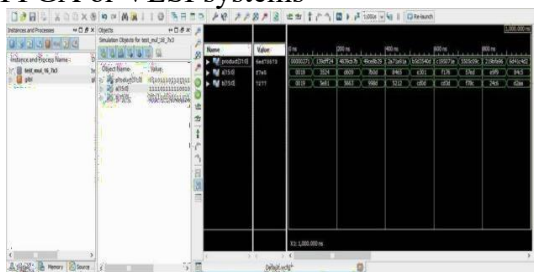


Figure 4: Simulation result of the proposed 16-bit multiplier using 7-3 compressor

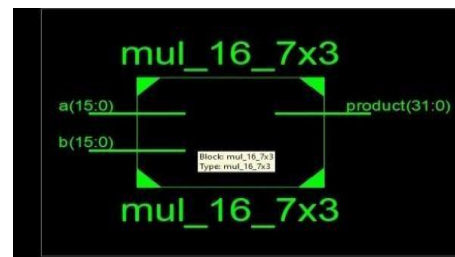


Figure 5: Block Diagram of 16-bit multiplier using 7-3 compressor

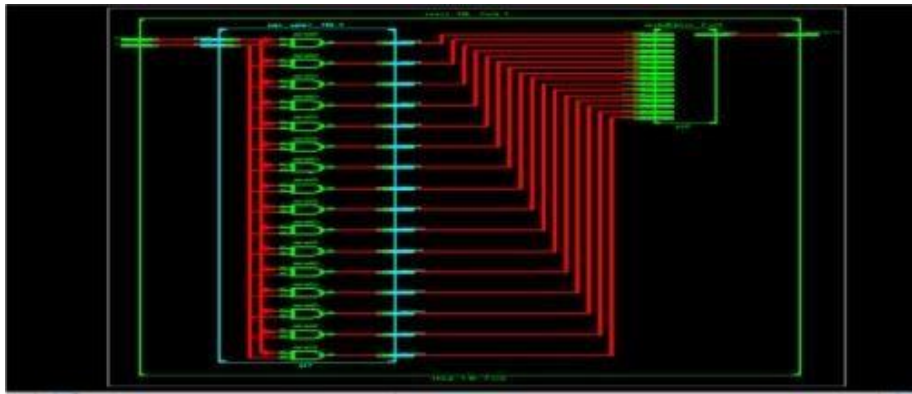


Figure 6: Internal View RTL schematic of the 16-bit multiplier using 7-3 compressor

The figure above shows how a digital multiplier circuit works. It uses parallel partial product generation and accumulation logic to perform multiplication. Several logic gates on the left side of the figure create partial products by doing bitwise AND operations on the bits of the multiplicand and the multiplier. There are horizontal signal lines that represent these partial products. The signals then travel through an organized network of red links, which align the partial products based on their positional weight in binary multiplication. The vertical lines in the middle indicate how the intermediate signals are prepared for the next stage. On the right side of the diagram, there is a series of connected lines that illustrate the addition or reduction stage. This is when the adders combine the partial products to produce the final multiplication result. This design allows the multiplier to perform operations quickly by generating all of the partial products simultaneously and then adding them up in a clear manner. This speeds up and improves the performance of digital arithmetic circuits used in processors and signal processing systems.



Figure 7: Internal Block Diagram of 16-bit multiplier using 7-3 compressor

The figure above shows the internal block structure of a 16-bit multiplier module, labeled mul_16_7x3:1. This module performs multiplication through stages of generating partial products and adding them. The design is split into two main parts: PP_gen_16 and addition_7x3.

The PP_gen_16 block creates the partial products from the input operands of the multiplier. These partial products represent the intermediate results of multiplication, generated by combining bits of the multiplicand and multiplier. The diagram displays these generated partial products as multiple parallel lines in the center.

Next, these intermediate values move to the addition_{7x3} block. This block sums the partial products using a network of structured adders. Signals labeled P1 and P2 show how data flows between the partial product generation and addition stages. The adder block combines the multiple partial products into the final multiplication result. Overall, this design makes multiplication more efficient by breaking the process into generating partial products and structured addition. This approach allows for faster and more organized hardware implementation of digital arithmetic operations.

5. Conclusion

In this paper, the authors detailed the process of creating a 7:3 compressor-based multiplier that is both fast and space economical. By cutting down on propagation latency and the amount of partial product reduction stages, the suggested architecture boosts multiplication performance. The use of compressor-based reduction techniques enables multiple bits to be processed simultaneously, which significantly improves the computational speed of the multiplier

Acknowledgment: Not Applicable.

Funding Statement: The author(s) received no specific funding for this study.

Conflicts of Interest: The authors declare no conflicts of interest to report regarding the present study

References

- [1] W.Liu,C.Gu,M.O'Neill,G.Qu,P.Montuschiand F.Lombardi,“Security in Approximate Computing and Approximate Computingfor Security: Challenges and Opportunities,”*Proceedings of the IEEE*, 2020.
- [2] W. Liu, F. Lombardiand M. Schulte,“Approximate Computing: From Circuits to Applications,”*Proceedings of the IEEE*, 2020.
- [3] A. Momeni, J. Han, P. Montuschi and F. Lombardi,“Design and Analysis of Approximate Compressors for Multiplication,”*IEEE Trans. on Computers (TC)*, 2015.
- [4] S.Venkatachalam and S.Ko,“Design of Power and Area Efficient Approximate Multipliers,”*IEEE Trans on Very Large-Scale Integration Systems (TVLSI)*, 2017.
- [5] M. Ha and S. Lee,“Multipliers With Approximate4–2 Compressors and Error Recovery Modules,”*IEEE Embedded Systems Letters(ESL)*, 2018.
- [6] O.Akbari, M.Kamal, A.Afzali-Kusha and M.Pedram, “Dual-quality 4:2 compressors for utilizing in dynamic accuracy configurable multipliers,”*IEEE Trans. on Very Large-Scale Integration Systems (TVLSI)*, 2017.
- [7] F.Sabetzadeh, M.H.Moayeri and M.Ahmadinejad,“A Majority-Based Imprecise Multiplier for Ultra-Efficient Approximate Image Multiplication,”*IEEE Trans on Circuits and Systems I (TCAS I)*, 2019
- [8] K.L.S. Sweeand Lo Hai Hiung, ”Performance Comparison Review of 32-Bit Multiplier Designs,” *4th International Conference on Intelligent and Advanced Systems (ICIAS)*, vol. 2, pp 836-841, 2012.
- [9] H.D. Tiwari, "Multiplier design based on ancient Indian Vedic Mathematics," *Int. SoC Design Conf.*, vol. 2. IEEE Proc.,pp. II-65 - II-68, 2008.
- [10] L. Sriraman, T.N. Prabakar, “Design and Implementation of Two Variable Multiplier Using KCM and Vedic Mathematics,” *1st Int. Conf. on Recent Advances in Information Technology*, Dhanbad, India, pp. 782-787, 2012.
- [11] Mehak Sharma, Vandana Khanna, Anu Tonk and Shaifali Ruhil, “Implementation and Comparative Analysis of Various Low Power FSM Synthesis Techniques,” *JEPE*, vol. 2, no. 1, pp.

54–66, 2025

- [12] S. R. Huddar, S. R. Rupanagudi, Kalpana and S. Mohan, “Novel High Speed Vedic Mathematics Multiplier Using Compressors,” *International Multi-Conference on Automation, Computing, Communication, Control and Compressed Sensing*, pp. 465-469, 2013.
- [13] D. Radhakrishnan and A. P. Preethy, "Low power CMOS pass logic 4-2 compressor for highspeed multiplication," *Circuits and Systems, Proc. 43rd IEEE Midwest Symp.*, vol. 3, pp. 1296- 1298, 2000.
- [14] C.Suneetha and P.Sree Lakshmi, “Energy Harvesting Based Metamaterial Microwave Antenna for Multi-BandOperation,” *Journal of Microwaves and Antenna Applications*, vol. 1, no. 2, 2025.
- [15] K.Prasad and K.K. Parhi, “Low power 4-2 and 5-2 compressors”, *35th Asilomar Conference on Signals, Systems and Computers*, vol.1, pp. 129 – 133, 2001.
- [16] A. Dandapat, S. Ghosal, P. Sarkar and D. Mukhopadhyaya, “A 1.2-ns 16×16-bit Binary Multiplier using High Speed Compressors”, *International Journal of Electrical, Computer and Systems Engineering*, vol. 4, No. 3, 2010, pp. 234 – 239
- [17] B.Ramkumar, Harish M Kittur and P.Mahesh Kannan, “ASIC implementation of Modified Faster Carry Save Adder,” *European Journal of Scientific Research*, vol.42, pp.53-58, 2010.