International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 04(April),2026
Received: 18-02-2026 Accepted: 23-03-2026 Published: 02-04-2026

Implementing Of FPM Using Approximate Radix-8 And Radix-4 Booth encoding

P. Vijaya,
Dept. of Electronics and Communication Engineering,
Annamacharya Institute of Technology and Sciences, Kadapa, India.
vijayaaitsk@gmail.com

Mitta Pavithra,
Dept. of Electronics and Communication Engineering,
Annamacharya Institute of Technology and Sciences, Kadapa, India.
pavithrareddymitta@gmail.com

Nagamuthi Sashi Ram,
Dept. of Electronics and Communication Engineering,
Annamacharya Institute of Technology and Sciences, Kadapa, India.
sashiram191@gmail.com

Shaik Hameed,
Dept. of Electronics and Communication Engineering,
Annamacharya Institute of Technology and Sciences, Kadapa, India.
shaikhameed91820 @gmail.com

Thuggali Nagaraju,
Dept. of Electronics and Communication Engineering,
Annamacharya Institute of Technology and Sciences, Kadapa, India.
thuggalinag@gmail.com

***Abstract*— Modern digital signal processing, scientific computation, and artificial intelligence applications—where fast speed and low power consumption are critical design requirements—all heavily rely on floating-point multiplication. This work describes a hybrid Booth encoder-based floating-point multiplier that uses CMOS 0.18-µm technology and is both low-power and area-efficient. In order to minimize hardware complexity, the suggested architecture uses a hybrid encoding technique, processing the most significant bits using Exact Radix – 4 Booth Encoding scheme while the least important bits are processed using the approximate Radix – 8 Booth Encoding scheme. Microwind is used to implement the design at the transistor level for performance evaluation, while DSCH is used for modeling and functional validation. When examined alongside the traditional radix – 8, radix – 4, and contemporary hybrid architectures, simulation results show notable savings in gate count, power consumption, and operating current. The suggested multiplier is appropriate for low-power VLSI and floating-point processing applications since it achieves increased speed and energy efficiency while preserving a respectable level of computational accuracy.**

Keywords— Floating-Point Multiplier, DSCH, Microwind, Low-Power VLSI, CMOS 0.18-µm Technology, Booth Encoding, Hybrid Radix-4/Radix-8, Approximate Computing, and Partial Product Reduction.

I. INTRODUCTION

In contemporary digital systems, arithmetic computation is essential on account that majority of processing tasks count on effective mathematical and logical operations. The Arithmetic Logic Unit (ALU), a basic and indispensable processor component, is responsible to execute all the arithmetic operations. Operations like addition, multiplications, subtractions, division, cubing, squaring are amidst the statistical Operations. Furthermore, the ALU is segregated into two partitions, only in some processor designs: the Arithmetic Unit, responsible for mathematical processes, and the Logic Unit, responsible for the logical processes. Previously implemented as a distinct numeric co-processor in personal computers, the Floating-Point Unit is a specialized module that handles floating-point calculations. Because it allows for the scaling of numerical quantities, multiplication is one of the most commonly employed ALU operations. Numerical analysis, scientific computing, multimedia processing, and other fields frequently use floating-point multiplication [1]. Multimedia systems, digital signal processing, neural networks, image processing, and machine learning are just a few of the modern computing applications that depend on multipliers as fundamental building pieces [2]. They are essential for carrying out important tasks like correlation, filtering, noise reduction, and domain transformations. Multipliers facilitate complicated data processing activities and enhance overall system performance and accuracy in a variety of real-time and computationally demanding applications by facilitating quick and effective numerical computations [3].

Both very small and very big numerical values can be represented with excellent precision thanks to the floating-point format's extraordinarily wide dynamic range. The majority of computations in mini wave pattern-based and also for the connected node network-based multimedia applications rely on floating-point arithmetic, which makes effective floating-point computing units—particularly multipliers—essential. To improve overall system performance and energy efficiency, these units must be built to achieve shorter path delays, lower hardware use, and lower power consumption. In sophisticated real-time applications, an efficient multiplier design greatly increases processing speed. The Booth algorithm is a popular multiplication method that improves hardware resource usage in digital systems and minimizes computing time by lowering the amount of partial products produced during multiplication [4-5]. Approximate or inexact computing has drawn a lot of interest as a successful method for creating low-power, high-performance arithmetic devices. It is possible to effectively create floating-point multipliers with better efficiency by using a radix-8 approximate multiplication approach. By reducing the levels of partial products generated during multiplication, this method speeds up computation and requires less hardware. Because the Modified Booth methodology operates at higher radix level, The computational effort required by the Modified Booth methodology is reduced with a compromise in its design complexity. This compromise in the design complexity is because the Modified Booth methodology functions at a higher Radix Level. Radix-8 encoding lowers the number of partial products substantially and requires nothing but simple

International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 04(April),2026
Received: 18-02-2026 Accepted: 23-03-2026 Published: 02-04-2026

bit-shift operations as supplementary logic. It covers in achieving a proper balance between speed and critical circuits. It suitable choice for modern digital system implementations when examined together the other radix alternates methods.

II. Floating Point Multiplication

The IEEE benchmark, and widely utilized floating-point algorithm multiplication norm offers efficient numerical computation and logical encoding in many types of digital systems. Sign determination, exponent computation, mantissa multiplication, product generation, rounding, and normalization are among the steps necessary for the floating-point numbers multiplier. For validating accuracy, precision during computing, every above-mentioned process is indispensable. Over the course of time, experts have suggested a number of optimization strategies to elevate the floating-point multiplication's effectiveness and performance. These methods predominantly emphasize on cutting down the power consumption, minimizing hardware complexity, and scaling down the computational time. As an example, Kuang et al. engineered enhanced multiplier design techniques to refine resource utilization and elevate speed in floating-point arithmetic operations, making them more suitable for high-performance applications [6-7].

In DSP applications, the floating-point multiplication is frequently used for critical system performance and speed. One such example is the mantissa multiplication that implements a 24x24-bit integer multiplier. In contrast to the traditional designs, a Vedic design-based IEEE-754 standard single-precision FPM designed using Verilog and executed using the Spartan-6 FPGA depicts an increased speed and decreased hardware area [8]. Moreover, by scaling down the size and power consumption, the rough Radix 4 Booth Multipliers elevate the performance without compromising the feasible accuracy [9]. Owing to great computational efficiency, Dadda, Booth, and Wallace multipliers are habitually used for mantissa multiplication in dominant parts of floating-point multiplication designs [10-12]. When creating and decreasing partial products, these multipliers predominantly result in considerable delays and raised power consumption. This could affect the overall performance of the system in a drastic manner. The speed of the multiplier is largely dependent on the efficiency of the partial product reduction step. Additionally, the precision, delay and efficiency of the final multiplication result is predominantly dependent on the design of the final adder which is responsible to combine the final two product arrays [13].

Approximation approaches have been found in [14] dramatically reduce area, latency, and power consumption in floating-point multiplication without appreciably impairing multiplier performance. Furthermore, because of their modular design, low power consumption, built-in parallelism, and effective pipelining support, FPGA-based systems are seen to be ideal for high-speed computing architectures [15], power-efficient HFPM with an near hybrid radix-4 and radix-8 Booth encoder to maximize system use and speed. With just a slight loss in accuracy the hybrid architecture lowers computing complexity, size, and power consumption by decreasing the level of partial products, as measured by Error Rate (ER) and Normalized Error Distance (NED) [16].

A. EXISTING DESIGN

The current design uses a 25 × 25 dual radix-4/radix-8 Booth encoder-based multiplier for effective mantissa multiplication. In Phase I, the multiplicand processing unit receives the processed mantissas of 24 bits and converts the input mantissas (m A, m B) into 25-bit operands. In order to make the data applied to the Booth multiplier 25-bit numbers, zeros are prefixed to the most important places of the rectified mantissas. The data processing reduces computational stability inside the multiplier structure. This is possible by streamlining the encoding process, thereby ensuring steady handling of signed data during generation of partial product. The purpose to stable speed and economy power, the implemented circuit uses a dual Booth encoding system that joins radix-4 in Fig. 1 and radix-8 in Fig. 2 encoding approaches. Radix-4 encoding decreases levels of partial products for an N-bit multiplier to around N/2. This allows for implementation mostly by shifting operations, which facilitates quicker multiplication. By reducing switching activity, radix-8 encoding after reduces the levels of partial products to roughly N/3, increasing power efficiency. However, radix-8 encoding necessitates the creation of ±3Y as shown in table 2, which entails addition and shifting operations and truth table for the radix-4 as shown in the table 1.

In comparison to radix-4 encoding, the additional adder adds carry propagation delay, which impacts total operating speed. Despite this drawback, compared to traditional multiplier architectures, the current design's hybrid usage of radix-4 booth encoding and radix-8 encoding offers increased computational efficiency and less hardware complexity. The current hybrid design uses 25-bit inputs to conduct the multiplier operation, which corresponds to the corrected mantissas mA and mB. Radix-4 and radix-8 Booth encoding techniques has combined to process these operands in order to increase multiplication performance while preserving a reasonable level of computational accuracy. Radix-8 Booth encoding is applied for the lower twelve least significant bits out of the 25 bits, while radix-4 Booth encoding is used for the thirteen most major bits. While radix-4 encoding in the higher significance region maintains numerical accuracy because errors in the most significant bits would have a substantial impact on the final multiplication result, radix-8 encoding in the lower significance region helps minimize partial product levels.

TABLE 1. RADIX-4 BOOTH ENCODER TRUTH TABLE[16]

a_k+1	a_k	a_k—1	P_K	PP_{JK}	$2P_k$	P_k
0	0	0	0	0	-	0
0	0	1	Y	bj	-	1
0	1	0	Y	bj	-	1
0	1	1	2Y	bj—1	1	0
1	0	0	-2Y	bj—1	1	0
1	0	1	-Y	—bj	0	1
1	1	0	-Y	—bj	0	1
1	1	1	0	0	0	0

2250-3676 ijesatj@gmail.com https://www.ijesat.com/

International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 04(April),2026
Received: 18-02-2026 Accepted: 23-03-2026 Published: 02-04-2026

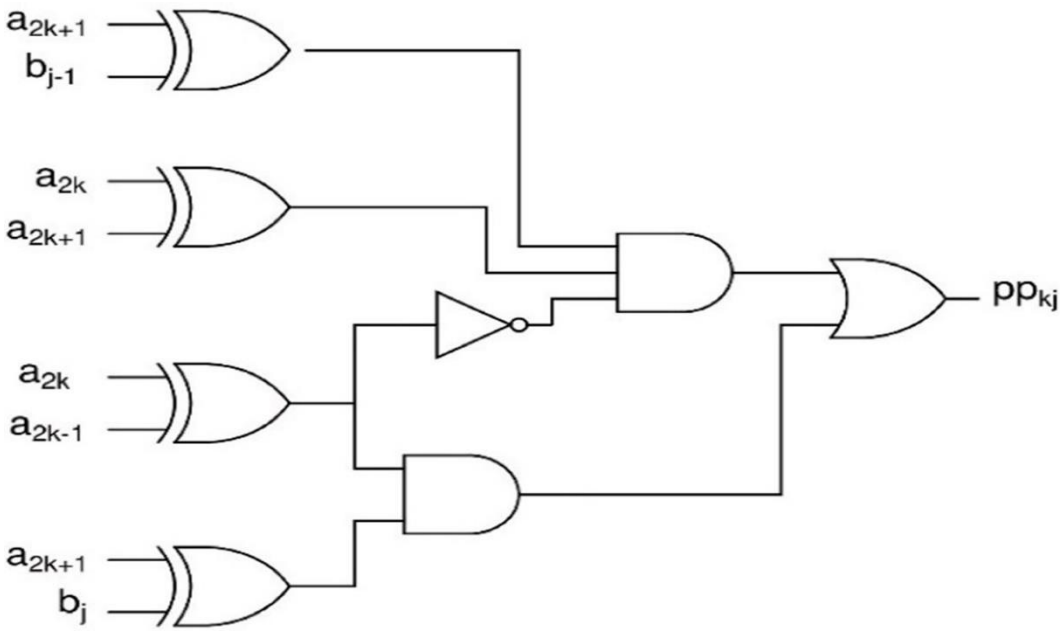

Fig. 1. Rdaix-4 Booth encoder for partial product generation [16].

TABLE 2. RADIX-8 BOOTH ENCODER TRUTH TABLE [16].

ak+2	*ak+1*	*ak*	*ak-1*	P_k	PP_{JK}	PP_{JK}
0	0	0	0	0	0	0
0	0	0	1	Y	*bj*	*bj*
0	0	1	0	Y	*bj*	*bj*
0	0	1	1	2Y	*bj*—1	*bj*—1
0	1	0	0	2Y	*bj*—1	*bj*—1
0	1	0	1	3Y	*si*	*bj*—2
0	1	1	0	3Y	*si*	*bj*—2
0	1	1	1	4Y	*bj*—2	*bj*—2
1	0	0	0	-4Y	*bj*—2	*bj*—2
1	0	0	1	-3Y	*si*	*bj*—2
1	0	1	0	-3Y	*si*	*bj*—2
1	0	1	1	-2Y	*bj*—1	*bj*—1
1	1	0	0	-2Y	*bj*—1	*bj*—1
1	1	0	1	-Y	*bj*	*bj*
1	1	1	0	-Y	*bj*	*bj*
1	1	1	1	0	0	0

III. PROPOSED DESIGN

The partial products of the 25 × 25 multiplication are organized into 49 columns in the suggested hybrid multiplier design. The proposed near radix-8 Booth encoding Fig. 4 generates the least significant twenty-four columns, while the exact radix-4 Booth encoding Fig. 3 yields the most significant twenty-five columns and its truth table shown in table 3. To ensure that any little inaccuracies produced do not significantly affect the final floating-point result, the approximation is purposefully applied exclusively to the lower significance bits. The design benefits from less complexity in the lower part of the multiplier while maintaining computational accuracy by maintaining the accuracy of the higher significance section. The proposed design the generate of partial products more efficient by approaching shift-based approximations for complex radix-8 calculations. Generating phrases like ±3Y in conventional radix-8 encoding, as shown in Table 4, mandates additional adder circuits, which enhance hardware usage and effects carry propagation delays. The multiplier operates more rapidly and uses minimal power due to scaled down switching activity and shortened critical routes. The proposed approach reduces the amount of combinational circuitry needed by swapping these processes with simpler shifting logic

In difference to the existing architecture, the proposed multiplier reduces the gate count by minimizing logic and arithmetic components. Simplified control logic and the deletion of additional adders guide to a more concise approach with higher area efficiency. Additional advantages of this hardware complexity reduction include optimized timing performance and minimal power dissipation. In the light of all factors, the proposed hybrid design accurately evens off accuracy and efficiency, making it well matched for floating-point and signal processing applications at places where high speed and low hardware cost are key design goals.

International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 04(April),2026
Received: 18-02-2026 Accepted: 23-03-2026 Published: 02-04-2026

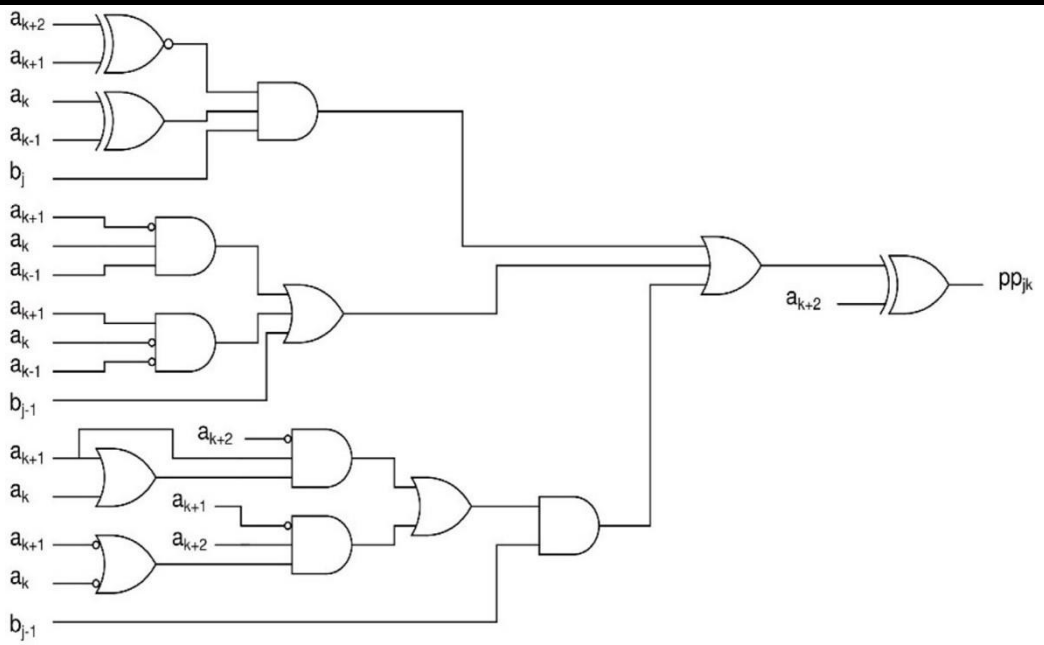

Fig. 2. Radix-8 Booth encoder for partial product generation [16]

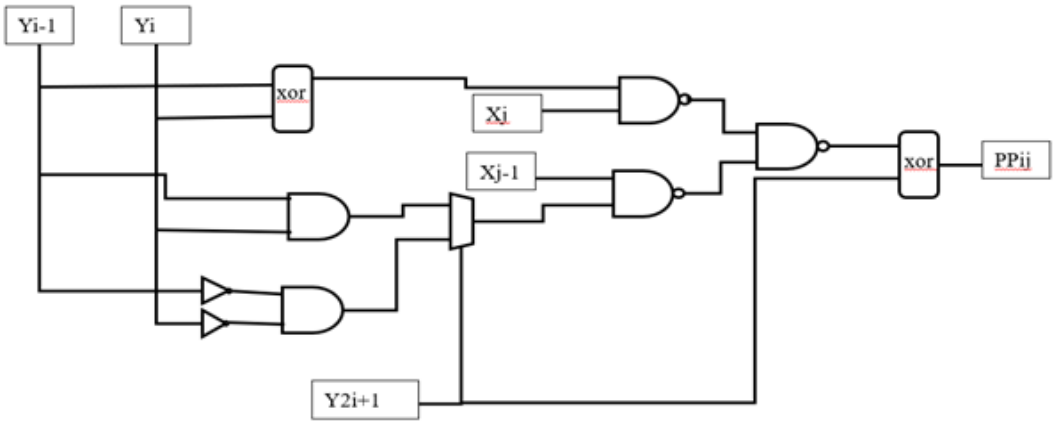

Fig. 3. Partial Product generator for Radix 4

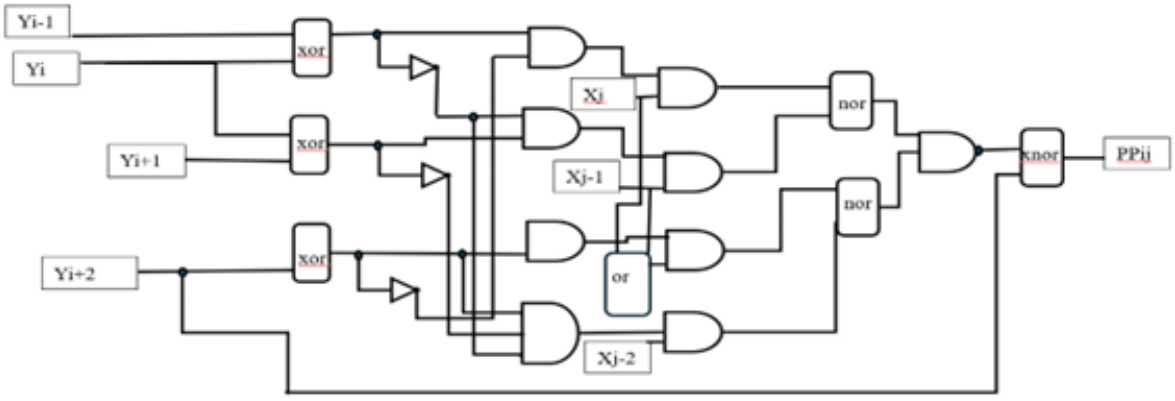

Fig. 4. Approximate Radix-8 Partial Product generator

International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 04(April),2026
Received: 18-02-2026 Accepted: 23-03-2026 Published: 02-04-2026

TABLE 3. RADIX-4 BOOTH ENCODER TRUTH TABLE

(Y_i+1)	(Y_i)	(Y_i-1)	Ppi
0	0	0	0
0	0	1	+1
0	1	0	+1
0	1	1	+2
1	0	0	−2
1	0	1	−1
1	1	0	−1
1	1	1	0

TABLE 4. APPROXIMATE RADIX-8 BOOTH ENCODER TRUTH TABLE

yk+2	*Yk+1*	*yk*	*Yk-1*	Partial Products
0	0	0	0	0
0	0	0	1	Y
0	0	1	0	Y
0	0	1	1	2Y
0	1	0	0	2Y
0	1	0	1	4Y
0	1	1	0	4Y
0	1	1	1	4Y
1	0	0	0	-4Y
1	0	0	1	-4Y
1	0	1	0	-4Y
1	0	1	1	-2Y
1	1	0	0	-2Y
1	1	0	1	-Y
1	1	1	0	-Y
1	1	1	1	-Y

IV. RESULTS AND DISSCUSION

For the purpose of assessing the performance of established and suggested hybrid floating point multiplier designs, the logic verification method is used. Towards this goal, the software used is the Digital Schematic (DSCH). For the purpose of physical design and CMOS – level analysis, the software Microwind is used with 0.18 µm technology. The simulation trails monitored the parameters namely running current, power consumption, hardware complexity, and overall circuit efficiency. DSCH simulations trials validated the error free functional behavior of the multiplier architecture, simultaneously Microwind architectures provided capability for accurate transistor-level parameter evaluation.

It can be defined that the radix – 4 and radix – 8 Booth encoding steps acted as designed. This statement is given as the partial products were produced accordingly which is confirmed through the Functional verification in DSCH software. The timing waveforms validated precise multiplication results for diverse input combinations. Subsequent to logic verification, the circuits were fabricated at the transistor level in Microwind utilizing the CMOS 0.18-µm technology, facilitation for a comprehensive assessment of power characteristics and space utilization within the scope of practical switching scenarios.

TABLE 5. COMPARISION TABLE FOR GATE COUNT

DESIGN ARCHITECTURE	GATE COUNT
CONVENTIONAL RADIX-4	2570
CONVENTIONAL RADIX-8	3190
EXISTING HYBRID DESIGN	2290
PROPOSED DESIGN	2140

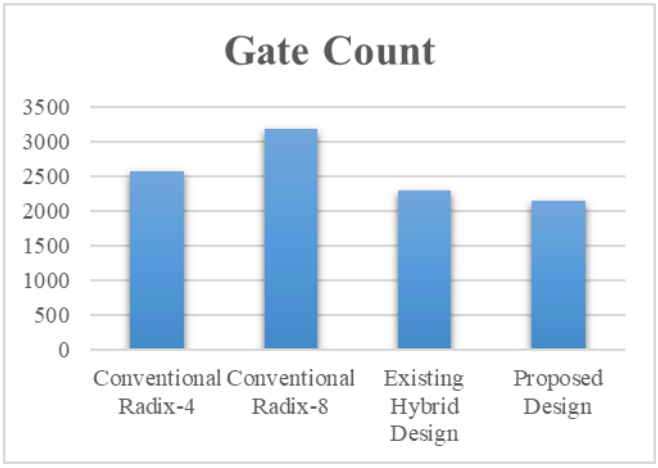

Fig. 5. Gate Count

Table 5 illustrates a analysis of hardware complexity from the perspective of gate count, and Fig. 5 portrays the equivalent graphical depiction. Consequent to the higher encoding complexity and additional arithmetic logic needed for partial product generation, the traditional radix-8 architecture has the maximum gate count, based on experimental findings. Despite the fact that radix-4 architecture reduces complexity to some extent, a noteworthy number of logic elements are continuing to be needed. By incorporating radix-4 and radix-8 encoding, the contemporary hybrid approach cuts the number of gates to 2290, enhancing hardware efficiency. By employing straightforward shift-based logic for multi-step operations, the suggested architecture intensively reduces hardware consumption and reaches the lowest gate count of 2140 gates. This decline exhibit that against the contemporary architectures, the suggested architecture facilitates superior area efficiency and mitigated circuit complexity.

International Journal of Engineering Science and Advanced Technology (IJESAT)
Vol 26 Issue 04(April),2026
Received: 18-02-2026 Accepted: 23-03-2026 Published: 02-04-2026

TABLE 6. COMPARISION TABLE FOR POWER AND CURRENT

Design Architecture	Power Consumption (mW)	Current (mA)
Conventional Radix-4	0.305	0.93
Conventional Radix-8	0.379	1.15
Existing Hybrid Design	0.272	0.83
Proposed Design	0.248	0.76

Operating current and power consumption from Microwind trials are outlined in Table 6, whereas the relative performance is graphically displayed in Fig. 6. The typical radix-8 multiplier has a 0.379 mW as upper limit of power consumption at an operating current of 1.15 mA caused by additional combinational circuitry and switching activity. The radix-4 architecture functions marginally better with a current of 0.93 mA and a power consumption of 0.305 mW. The radix-4 architecture indicates better efficiency than the radix-8 implementation, showing a moderate decrease in energy consumption. By decreasing the number of partial products generation and logic transitions, the current hybrid design further lowers power consumption to 0.272 mW. The suggested multiplier shows notable energy savings with the lowest power usage of 0.248 mW and operating current of 0.76 mA. The reduction in power and current is mainly attributed to simplified partial product generation and elimination of additional adder circuits in the approximate radix-8 encoding stage.

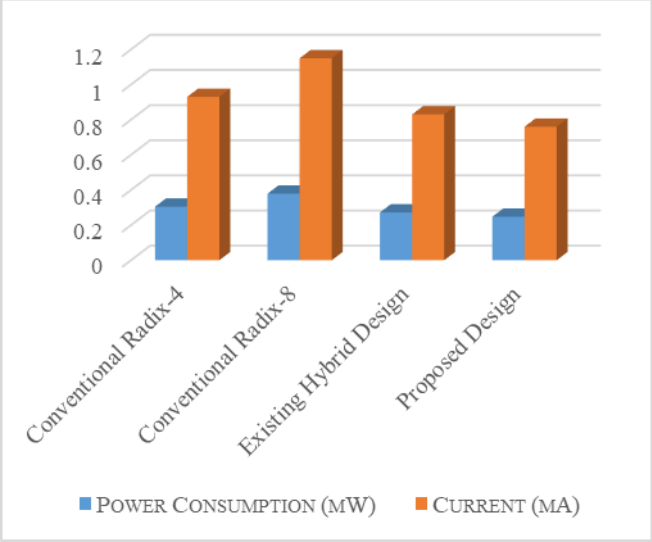

Fig. 6. Comparison of Power Consumption and current

The results clearly show that, while having minimal impact on calculation correctness, limiting approximate to the lowest-significant partial product column size significantly improves hardware efficiency. While approximate radix-8 encoding decreases switching activity in the lower significance region, exact radix-4 encoding maintains numerical precision in the most significant region. This well-balanced hybrid technique allows for increased performance without sacrificing dependability. Overall, the suggested multiplier architecture outperforms traditional radix-4, radix-8, and current hybrid designs, according to DSCH and Microwind simulation findings utilizing CMOS 0.18-µm technology. The outlined circuit highlights superior hardware efficiency, fewer gates, lower power consumption, and lower operating current. As result of these modifications, the architecture is well designed for low-power floating-point units, DSP systems, and high-speed VLSI applications, where area and energy optimization are essential design requirements.

V. CONCLUSION

The design and implementation of a mix radix-4/radix 8 Booth encoder-based floating-point multiplier intended for low-power VLSI applications were presented in this study. The proposed design improves hardware efficiency while keeping acceptable computing accuracy by combining accurate encoding in the most important region with approximation encoding in the least important region. Reductions in gate count, power consumption, and operating current when compared to conventional multiplier designs were proven by simulation and layout analysis carried out using DSCH and Microwind tools. The results of this study showed how regulated approximation can efficiently maximize performance without causing significant errors. The proposed multiplier can be used in embedded processors, signal processing systems, and AI hardware accelerators where small hardware design and energy efficiency are essential.

REFERENCES

[1] M.K. Jaiswal, R.C. Cheung, Area-efficient architectures for double precision multiplier on FPGA, with run-time-reconfigurable dual single precision support, Microelectron. J. 44 (5) (2013) 421–430.

[2] Jiménez, A.; Sauceda, Á.; Muñoz, A.; Duarte, J.; Mireles, J., Jr. FPGA-Based Hardware Implementation of Homodyne Demodulation for Optical Fiber Sensors. *Photonics* 2023, *10*, 258.

[3] Tavakkoli, E.; Shokri, S.; Aminian, M. Comparison and design of energy-efficient approximate multiplier schemes for image processing by CNTFET. *Int. J. Electron.* 2023, *111*, 813–834.

[4] P. Yin, C. Wang, W. Liu and F. Lombardi, "Design and Performance Evaluation of Approximate Floating-Point Multipliers," *2016 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, Pittsburgh, PA, USA, 2016, pp. 296-301, doi: 10.1109/ISVLSI.2016.15.

[5] A.D. Booth, A signed binary multiplication technique, Q. J. Mech. Appl. Math. 4 (2) (1951) 236–240.

[6] IEEE standard for floating-point arithmetic, IEEE Standard 754-2008 (2008) 1–70.

[7] S.R. Kuang, J.P. Wang, H.Y. Hong, Variable-latency floating-point multipliers for low-power applications, IEEE Trans. Very Large Scale Integrat. (VLSI) Syst. 18 (10) (2010) 1493–1497.

[8] K. Paldurai and K. Hariharan, "FPGA implementation of delay optimized single precision floating point multiplier," *2015 International Conference on Advanced Computing and Communication Systems*, Coimbatore, India, 2015, pp. 1-5, doi: 10.1109/ICACCS.2015.7324094.

[9] S. Venkatachalam, E. Adams, H. J. Lee and S. -B. Ko, "Design and Analysis of Area and Power Efficient Approximate Booth Multipliers," in *IEEE Transactions on Computers*, vol. 68, no. 11, pp. 1697-1703, 1 Nov. 2019, doi: 10.1109/TC.2019.2926275.

[10] M. H. Haider, H. Zhang and S. -B. Ko, "Decoder Reduction Approximation Scheme for Booth Multipliers," in *IEEE Transactions*

on Computers, vol. 73, no. 3, pp. 735-746, March 2024, doi: 10.1109/TC.2023.3343093

[11] K. C. Pathak *et al*., "An Efficient Dadda Multiplier using Approximate Adder," *2020 IEEE REGION 10 CONFERENCE (TENCON)*, Osaka, Japan, 2020, pp. 176-181, doi: 10.1109/TENCON50793.2020.9293737.

[12] N. Sureka, R. Porselvi and K. Kumuthapriya, "An efficient high speed Wallace tree multiplier," *2013 International Conference on Information Communication and Embedded Systems (ICICES)*, Chennai, India, 2013, pp. 1023-1026, doi: 10.1109/ICICES.2013.6508192.

[13] N.V.V.K. Boppana, et al., Low-cost and high-performance 8× 8 booth multiplier, Circuits. Syst. Signal. Process. 38 (2019) 4357–4368.

[14] hardware. 2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC), 2014, pp. 1–6.

[15] S. Taslimi, R. Faraji, A. Aghasi, H.R. Naji, Adaptive edge detection technique implemented on FPGA, Iranian J. Sci. Technol. Trans. Electric. Eng. 44 (2020) 1571–1582.

[16] Edavoor, Pranose J., Aswini K. Samantaray, and Amol D. Rahulkar. "Design of floating point multiplier using approximate hybrid Radix-4/Radix-8 booth encoder for image analysis." *e-Prime-Advances in Electrical Engineering, Electronics and Energy* 8 (2024): 100546.