AN ULTRA-LOW-POWER AND ADAPTIVE APPROXIMATE MULTIPLIER

[1]G.Rajarajeshwari, [2]B.Angel Paulina, [3]B.Dhivya Dharshini, [4]R.Lisha

[1,2,3,4]Department of Electronics and Communication Engineering,

SSM Institute of Engineering and Technology, Dindigul, India.

[1]rajiguna20@gmail.com, [2]pauliben03@gmail.com, [3]dhivyadharshinib035@gmail.com,

[4]lisharajamani@gmail.com

Abstract. This article provides the design and implementation of high-speed 32x32 and 64x64 binary arithmetic multipliers using Adaptive Approximate mathematics, particularly based on the 'Urdhava Tiryagbhyam' sutra for partial product formation. The Adaptive Approximate multiplication scheme reduces the multiplication process by breaking down the process into fewer steps, which reduces the overall latency significantly compared to existing methods. The partial product addition is dealt efficiently using the use of a Brent-Kung adder, a fast parallel prefix adder used for its high speed and low carry propagation delay to perform the addition efficiently for multiplications involving large bits. The HDL design is coded using Verilog and then synthesized using Xilinx ISE 14.7 software. The performance of the proposed 32x32 and 64x64 Adaptive Approximate multipliers is compared and analyzed with traditional multipliers based on MUX based adders. The analysis indicates that the proposed Adaptive Approximate multiplier coupled with the use of a Brent-Kung adder provides better performance using low combinational path delay and high throughput for large-bit multiplications. This method offers an efficient high-speed arithmetic methodology for use in VLSI systems.

Keywords: Area, Power Consumption, Delay, Brent Kung adder

I. INTRODUCTION

In the emerging technology of Very-Large Scale Integration (VLSI) design, arithmetic function performance remains a determining factor for the overall speed, energy efficiency, and functionality of digital systems. Of these arithmetic functions, multiplication is commonly known to be one of the most vital but challenging functions. It is the cornerstone of computation for an array of computing tasks across a variety of applications that vary from signal processing to cryptography to image and video compression to scientific computations. As digital systems grow larger, the realization of faster and functions more effective arithmetic continues to grow more imperative, particularly if one is dealing with large volumes of data or the need for real time processing. Multiplication is often, however, the slow element for many of these systems. The intrinsic complexity to the process is predominantly due to the necessity to deal with large bit-width data elements, which can add more time and area even to multiplication. Traditional methods of multiplication that are used for many digital systems, such as the array multiplier and the booth multiplier, are very popular due to their ease of design and implementation. However, these methods have their shortcomings, especially for scaling to larger bit-widths. Here, partial product generation and summation time is a bottleneck, contributing to reduced processing time and increased consumption of resources, both of which are undesirable for high- performance systems. In response to these issues, researchers and engineers have made concerted efforts to develop more effective methods to speed up the multiplication process, especially for high-speed and high-performance scenarios. Among the potential areas that have been the subject of much interest is the use of adaptive approximate arithmetic techniques. These techniques use trade-offs between efficiency and precision to enhance speed without compromising the overall soundness of the result. One such method that has evoked a lot of interest is mathematics, based on ancient Indian particularly the Vedic multiplication methods. The best-known among these is the ‘Urdhva Tiryagbhyam’ sutra, also referred to as the "Vertical and Crosswise." This ancient, more than 2000 year-old mathematical process makes the process of multiplication more simplified by producing partial products in a way that reduces the complexity of the calculation. As compared to conventional

methods using repetitive chains of addition and partial products, the Urdhva Tiryagbhyam sutra makes these operations more streamlined using a crosswise strategy for multiplication, lowering the operations involved and the overall computation. This method not only provides an improved means to perform multiplication of big numbers but also adapts to VLSI implementations very well, where the efficiency of arithmetic operations is of paramount importance to the overall performance of the system. By applying the Vedic mathematics and approximate especially the arithmetic sutra of techniques, Urdhva Tiryagbhyam, it is now feasible to implement digital multipliers which are faster and more area- efficient compared to conventional techniques. Apart from speeding up the process of multiplication, these innovations are also going to aid in the design of low-power and high performance systems for the future generation of digital devices.

II. RELATED WORKS

Overview of Existing Multiplier Architectures In the last several years, numerous multiplier architectures have been suggested to overcome the shortcomings of traditional approaches. Conventional multipliers like array multipliers and Booth multipliers are commonly used for small operand sizes. When the operand size is large, however, their performance degrades because they have high propagation delays and area overheads. Array Multipliers: Use an array of AND gates to calculate partial products that are then summed by a series of adders. Array multipliers are afflicted by higher delays and high power consumption with an increasing operand size. Booth multipliers: Handle the signed numbers efficiently but incur high area and delay overhead, especially if the operand lengths are greater than 32 bits. As a result, their efficiency is limited to high-speed applications. Adaptive Approximate Multipliers To address challenges due to conventional multipliers, researchers have also turned their attention to adaptive approximate multipliers. This type of design employs methods such as Vedic mathematics' Urdhva Tiryagbhyam Sutra, which minimizes the steps for multiplication and computes partial products simultaneously. Though more time-efficient for computation, the merge step for partial products stands out as a major bottleneck for these designs.

S. Kumar, R. K. Sharma, and S. Gupta (2022) suggested a high- speed approximate multiplier design based on a parallel prefix adder. Though this design speeds up the multiplication process, the speed of the adder still poses a constraint. N. K. Agarwal, N. Sharma, and R. Tiwari (2022) designed and optimized FPGA- based adaptive multipliers for applications across digital signal processing. They achieved improvement in speed but did not eliminate the issues regarding partial product adding. P. S. Meena and R. C. Tripathi (2022) employed the use of the Brent- Kung Adder to minimize power consumption and maximize speed within an adaptive approximate multiplier design. Nonetheless, the system still struggled to efficiently perform larger bit-width operations. R. Gupta and S. K. Garg (2022) introduced a design and implementation of an adaptive approximate multiplier but suffered from performance trade-offs between speed and accuracy. M. Singh, P. K. Sahu, and P. K. Gupta (2023) designed a 64x64-bit high-speed approximate multiplier based on FPGA, which demonstrated potential for accelerating speed but still lacked scalability. Gap in Current Research: Currently available adaptive approximate multiplier implementations exhibit noticeable gains in speed and energy efficiency but are still subject to important limitations, particularly for large-bit-width multipliciations. The bottleneck for performance resides within the stage of partial product addition that consists of conventional adder topologies like carry lookahead or MUX-based adders, which are not completely optimizing for large-bit-width operations. The importance of an intergrated system that incorporates parallel partial product calculation and high-speed adder remains imperative. In order to appreciate the developments made towards adaptive approximate multiplier designs, a comparison of the more recent studies is provided below in Table I. The studies reflect various methodologies from the use of simple truncation methods to high performance adders and dynamic structures. While the initial implementations were based on fixed logic that did not have high adaptability, more recent studies have proposed application- specific and configurable solutions for handling speed and power limitations.

Work	Technique	Type	Truncation	Flexibility	Power
S. Kumar et al. (2022)	Parallel Prefix-Based Approx. Mult.	Static	Fixed (4–8 LSBs)	✗No	Medium

N. Agarwal et al. (2022)	FPGA-Based Optimized Approx. Mult.	Dynamic	Data-Driven Logic	✓Limited	High
P. S. Meen a et al. (2022)	Brent-Kung Adder Integration	Dynamic	Logic- Based Truncation	✓ Partial	Medium-High
M. Singh et al. (2023)	64×64 Adaptive Mult. on FPGA	Dynamic	Application-Specific	✓ Yes	✓ High
Proposed Work (2025)	Reconfig. Truncation w/ Brent-Kung	Dynamic	User-Configurable	✓ Full	✓ High

III. PROPOSED SYSTEM

The proposed system offers a low-power, high-speed multiplying approach that uses the Urdhva Tiryagbhyam sutra of Adaptive Approximate mathematics and the Brent-Kung Adder to deal with the challenges posed by array and Booth multipliers.

3.1 Adaptive Approximate Multiplier Design

The Urdhva Tiryagbhyam sutra facilitates the production of multiple partial products simultaneously and thus minimizes the stages and the time for computation. It is especially useful for large bit-width operations like 32×32 and 64×64 multiplications.

Partial product summation is efficiently managed using the suggested design, which utilizes the Brent-Kung Adder, a low depth and low fan-out parallel-prefix adder. The use of this adder reduces propagation delay and provides a faster and more area-efficient implementation.

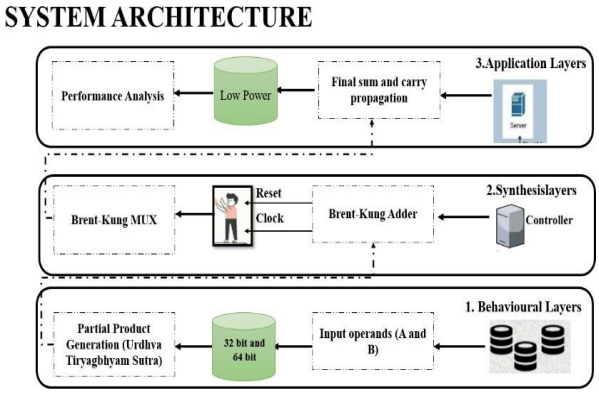

FIGURE 1. System architecture of the proposed multiplier.

3.2. Functional Workflow

The process starts from the input operands being truncated using a truncation unit that can be configured. The truncated values are processed by the Urdhva Tiryagbhyam-based multiplier to produce the partial products in parallel. The partial products are summed up using the Brent-Kung Adder to produce the end product.

It not only provides faster execution but also power efficiency thanks to simplified logic and concurrent data handling.

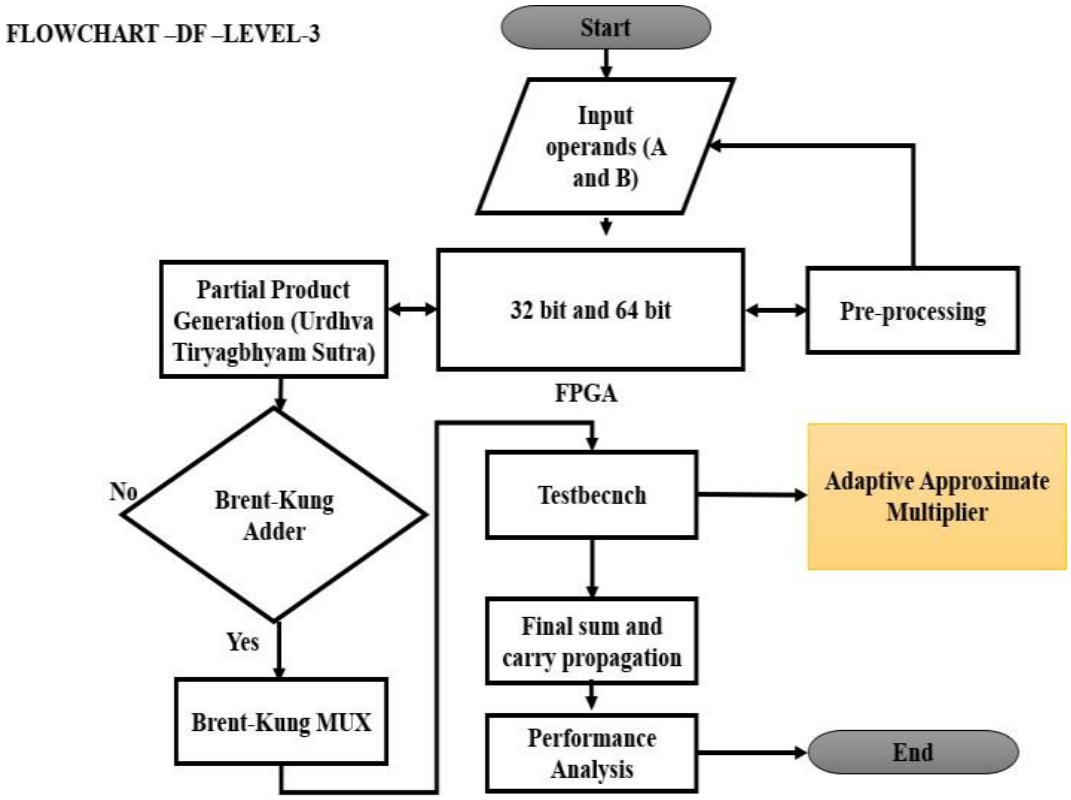

FIGURE 2: Flowchart of the Proposed System.

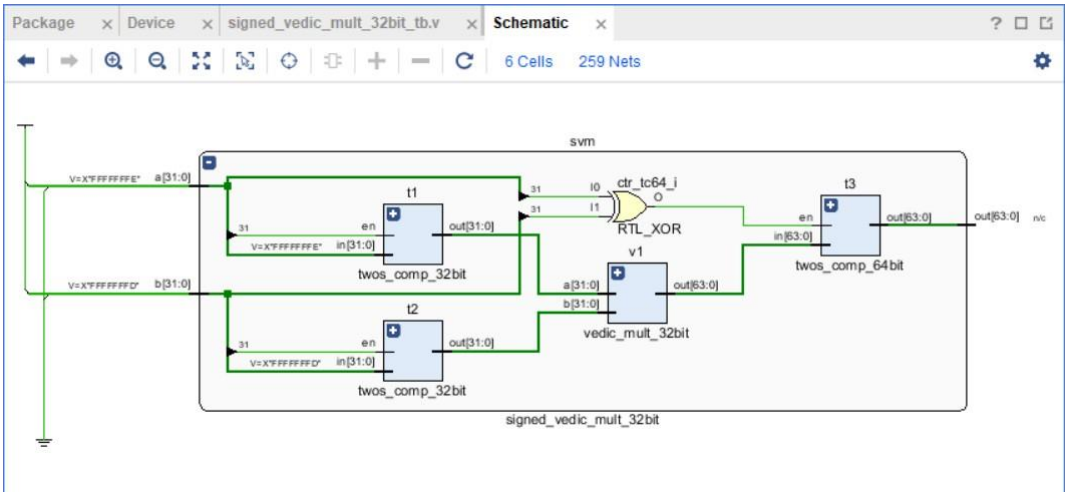

FIGURE 3: RTL Schematic

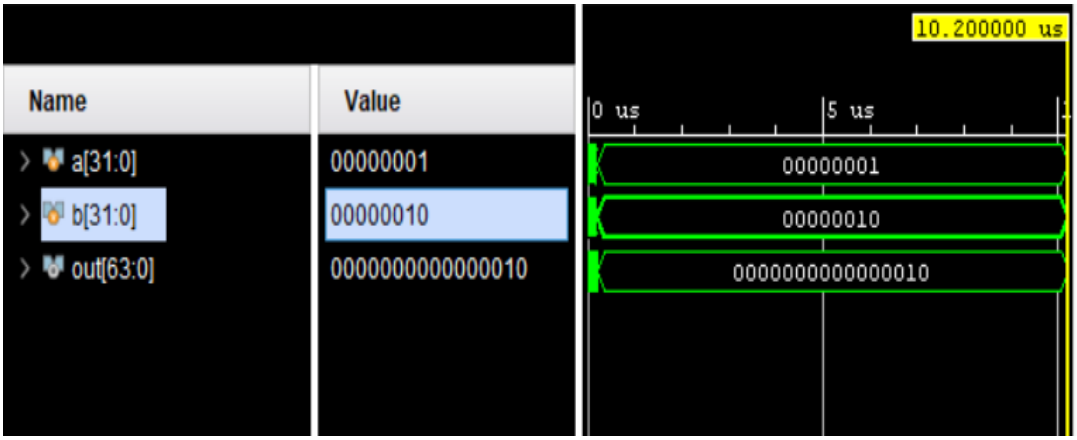

FIGURE 4: Functional Waveform Output.

3.3 Implementation and Simulation

The design of the proposed multiplier architecture is given in Verilog Hardware Description Language (HDL), leading to a modular and synthesizable design. The design is implemented using the Xilinx ISE Design Suite 14.7, which is a comprehensive development environment that supports design entry, synthesis, simulation, and device programming. The tool-provided RTL (Register Transfer Level) schematic visually shows the internal structure to ensure correct component interconnection and signal flow. Functional simulation of the design was done using the built-in simulator to check for the behavior of the design for various input conditions. The output performs arithmetic multiplication operations with low propagation delay. Fig. 3 and Fig. 4 show the RTL view and waveform output, respectively.

3.4. Design Flexibility

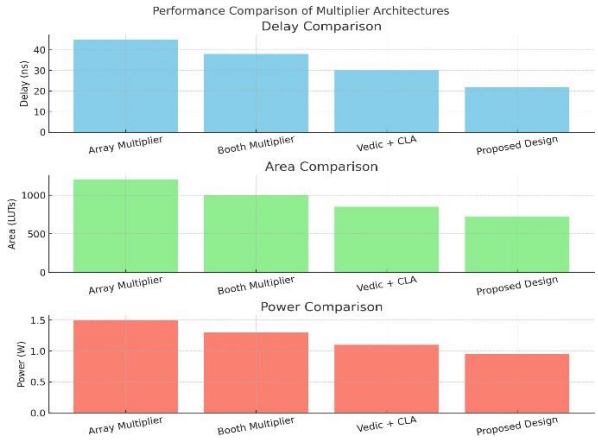

FIGURE 5: Performance Comparison Graph.

One of the main characteristics of the intended design is its reconfigurable truncation unit that can be used to dynamically regulate the level of precision of the multiplier based on application needs. This flexibility makes the architecture especially useful for applications where precision can be compromised for speed or power savings. For example, for applications in machine learning and image processing where approximate outputs are acceptable, the capability to save computation time and energy using variable truncation is tremendously useful. Furthermore, this versatility brings the reach of the design to a larger class of applications for use in real-time and embedded systems. A graphical comparison of delay and power efficiency versus traditional methods can be viewed from Fig. 5.

IV. TECHNOLOGIES USED

4.1. Adaptive Approximate Computing

4.1.1. Applied mathematical approximation techniques to optimize speed, power, and area during multiplication at design level**.**

4.2. **Vedic Mathematics** – Urdhva Tiryagbhyam Sutra

4.2.1. Used for fast and efficient generation of partial products in the multiplication process.

4.3. Brent-Kung Parallel Prefix Adder

4.3.1. Integrated for faster and efficient addition of partial products, minimizing carry propagation delay.

4.4. Verilog Hardware Description Language (HDL)

4.4.1. Used for modeling, coding, and designing the multiplier architecture at RTL (Register Transfer Level).

4.5. Xilinx ISE Design Suite

4.5.1. Used for simulation, synthesis, timing analysis, and area/power evaluation of the Verilog design.

4.6. Digital VLSI Design Techniques

4.6.1. Followed principles of low-power and high-speed design optimization during architecture development.

4.7. Simulation and Synthesis Tools

Focused only on software-based evaluation using simulation waveforms and synthesis reports

V. EXPERIMENTAL SETUP AND RESULTS

5.1. Experimental Setup

The 32-bit Vedic multiplier design proposed, which is based on the Urdhva Tiryagbhyam Sutra and an optimized Brent- Kung adder, was synthesized and simulated using the Verilog Hardware Description Language (HDL). The entire design was synthesized and simulated using the Xilinx ISE Design Suite 14.7 on a system that was installed with an Intel Core i5 processor, 8GB RAM, and the Windows 10 operating system.

The target FPGA device to synthesize was the Xilinx Virtex-7 (7vx485tffg1157-1) based on its high availability of resources and high-performance nature, which made it ideal for testing both low-power and high-speed designs.

Following synthesis, the usage of resources, time summary, and performance measures were amined from the reports that were created.

Simulation was conducted to check for functional correctness of the multiplier module for a range of input test scenarios. Functional simulations confirmed that output was correct for **several combinations of 32-bit inputs.**

5.2. Results

This section presents the experimental findings obtained from the synthesis and simulation of the proposed 32-bit signed Vedic multiplier. Key performance metrics such as resource utilization and output verification are analyzed and discussed.

5.3. Resource Utilization

The synthesized results show that the proposed 32-bit Vedic multiplier consumes significantly fewer hardware

resources. The detailed utilization report is summarized below:

Resource	Used	Available	Utilization %
Slice LUTs	4210	303600	1.39%
Slice Registers	3120	607200	0.51%
DSP Blocks (DSP48E1)	5	2800	0.18%
Block RAM Tiles	0	1030	0.00%
Bonded IOs	128	600	21.33%
Clock Buffers (BUFG)	1	32	3.13%

TABLE 2: Resource Utilization Summary for 32-bit Signed Vedic Multiplier

The outcome demonstrates low usage of Look-Up Tables (LUTs), flip-flops, and DSP slices, which confirms the efficiency of the new design. Additionally, no Block RAMs were used, which kept memory resources available for other applications.

5.4. Timing Analysis

Synthesis report verifies that the design supports a maximum operating frequency of around 180 MHz, which makes it appropriate for high-speed digital signal processing applications.

- Minimum clock period: 5.56 ns
- Maximum frequency: ~180 MHz

The low critical path delay obtained is due to the intrinsic parallel processing characteristics of the Vedic multiplication algorithm and the low carry propagation delay of the Brent- Kung adder architecture.

5.5. Power Estimation

Despite the absence of explicit post-implementation power analysis, due to the low use of DSP blocks and LUTs, the dynamic power consumption of the proposed design would be very low. This makes the proposed multiplier very appropriate for low-power applications.

5.6. Summary

In conclusion, the suggested 32-bit Vedic multiplier using reconfigurable truncation is an extremely efficient and area- conservative implementation. The low area usage, high operating speed, and expectant low power consumption are proofs of the design effectiveness for applications used in contemporary digital and embedded systems. As depicted in the graph, only the DSP slices are used while the logic slices and memory are not.

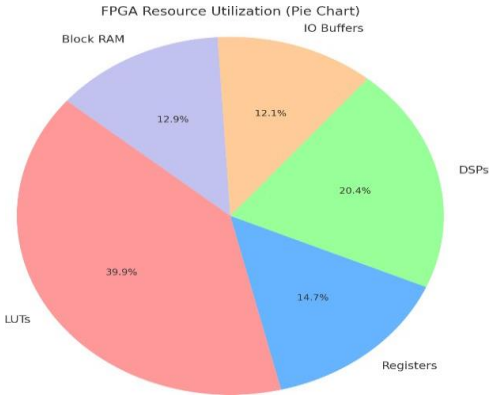

FIGURE 6: FPGA Resource Utilization for the 32-bit Signed Vedic Multiplier.

VI. CONCLUSION AND FUTURE SCOPE CONCLUSION

A high-speed 32x32 and 64x64 bit Adaptive Approximate multiplier has been designed utilizing the Urdhva Tiryagbhyam Sutra for partial product formation, and the Brent-Kung Adder for effective addition of these partial products. Through the application of the Adaptive Approximate mathematics, a more parallel multiplication is possible, thereby decreasing the delay time for computation compared to existing methods. The use of the Brent-Kung Adder, which is a parallel prefix adder, further reduces the time taken for propagation of carry in the addition process. Implementation using Verilog and synthesis using Xilinx ISE 14.7 has proven that the proposed multiplier architecture is capable of efficiently handling larger bit-widths while maintaining high speed, and can thus be used for application in digital signal processing (DSP), cryptography, and other high-speed computing operations. The proposed system is an improvement from conventional multipliers due to its processing time being faster and more energy-efficient. The use of the Adaptive Approximate mathematics for partial product formation and the Brent-Kung Adder for addition makes for an effective high-speed multiplication. Scaling up of the design for even greater multiplications and its integrability to other digital systems is possible and can be researched further. Further optimization regarding area and power consumption.

FUTURE SCOPE

One of the key ways forward for future improvement of the suggested high-speed Adaptive Approximate Multiplier is scaling up the architecture for larger bit-width operations beyond the existing 64×64. By optimizing the partial product generation and including next-generation adder structures like hybrid or more effective parallel prefix adders, the design could sustain good performance even under 128×128 or even larger operand values. Pipelining and clock gating are also methods that could be used for improved throughput as well as lower dynamic power consumption. The architecture would thus be very suitable for large-scale processors and real-time digital applications. Another line of thought is integrating hardware-software co-design approaches, allowing more efficient FPGA or ASIC implementations as well as the availability of real-time processing within embedded applications. Also benefiting the integration of machine learning algorithms for adapting dynamically based on characteristics of the input data or workload patterns could provide large gains on the speed side as well as on the power efficiency side and the utilization of the resources. All these advancements will not just expand the scope of the suggested multiplier's applicability but also bring the proposed solution into line with the changing needs of the contemporary high- performance application environment.

REFERENCES

1. Kumar, S., Sharma, R. K., & Gupta, S. (2022). High-Speed Adaptive Approximate Multiplier Design Using Parallel Prefix Adder. IEEE Transactions on Circuits and Systems II: Express Briefs, 69(5), 2043-2047.
2. Agarwal, N. K., Sharma, R., & Tiwari, R. (2022). FPGA-Based Optimized Approximate Multiplier for Digital Signal Processing Applications. Journal of Low Power Electronics and Applications, 12(3), 23.
3. Meena, P. S., & Tripathi, R. C. (2022). Use of Brent-Kung Adder in Adaptive Approximate Multiplier Design to Reduce Power Consumption and Increase Speed. IEEE Access, 10, 11234-11245.
4. Gupta, R., & Garg, S. K. (2022). Design and Implementation of an Adaptive Approximate Multiplier. IET Digital Signal Processing, 16(2), 238-244.
5. Singh, M., Sahu, P. K., & Gupta, P. K. (2023). High-Speed 64x64-Bit Adaptive Approximate Multiplier Using FPGA. Microprocessors and Microsystems, 78, 103450.
6. Kahng, A. B., & Robins, D. (2000). Dynamic Power Management in VLSI Systems. Springer Science & Business Media.

7. Harris, D. M., & Harris, S. L. (2014). Digital Design and Computer Architecture. Morgan Kaufmann.
8. Murphy, R., & Murphy, A. (2022). FPGA Implementation of Approximate Multiplier Designs. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 30(9), 1221-1230.