International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 14, Issue 1, January 2025)

Review of VLSI Implementation of Approximate Booth Multiplier

[1]Mohd Shariq Zia , [2]Manish Gupta, [3]Dr Anshuj Jain
[1]Research Scholar, Dept. of Electronics and Communication Engineering, SCOPE College of Engineering, Bhopal, India
[2]Assistant Professor, Dept. of Electronics and Communication Engineering, SCOPE College of Engineering, Bhopal, India
[3]Associate Professor & HOD, Dept. of Electronics and Communication Engineering, SCOPE College of Engineering, Bhopal, India

***Abstract*— The rapid advancement of computing applications in recent years has created a growing demand for high-speed and energy-efficient arithmetic operations. Multipliers are among the most critical components of digital circuits, particularly in signal processing and machine learning applications, where they are often utilized for large-scale computations. Booth multipliers, known for their ability to perform signed multiplications efficiently, have been widely studied and implemented in various Very Large-Scale Integration (VLSI) designs. However, exact implementations of Booth multipliers can be computationally intensive, leading to higher power consumption and larger area requirements in hardware. Approximate computing has emerged as a promising approach to mitigate these challenges by trading off a degree of computational accuracy for improvements in power, area, and delay metrics. This review explores the state-of-the-art techniques for VLSI implementation of approximate Booth multipliers.**

Keywords*— *VLSI, Approximate, Booth Multiplier, FPGA.

I. INTRODUCTION

Multiplication is a fundamental operation in digital computation, forming the cornerstone of applications ranging from digital signal processing (DSP) and artificial intelligence (AI) to cryptographic systems and real-time data analytics. Among the various multiplier architectures developed over the decades, Booth multipliers stand out due to their ability to perform signed multiplications efficiently by reducing the number of partial products. This efficiency, achieved through a unique encoding algorithm, makes Booth multipliers a preferred choice in many high-performance VLSI designs. However, as the demand for resource-constrained computing grows, conventional Booth multipliers face significant challenges in balancing computational accuracy with power, area, and delay constraints.

The emergence of approximate computing offers a paradigm shift in addressing these challenges. Approximate computing, which deliberately introduces errors in computation to achieve substantial gains in power efficiency and silicon area reduction, is particularly suited for applications that can tolerate some degree of inaccuracy. Examples include multimedia processing, machine learning inference, and other domains where perceptual or statistical accuracy is often more critical than exact numerical precision. Within this context, the implementation of approximate Booth multipliers in VLSI design has garnered significant attention as a means to meet the dual goals of energy efficiency and performance optimization.

This review delves into the intricacies of designing approximate Booth multipliers and examines the state-of-the-art methodologies employed to achieve approximate functionality. A typical Booth multiplier operates by encoding input operands into a reduced number of partial products using Booth's algorithm, followed by their efficient accumulation. While this method inherently optimizes the multiplication process, further approximations in the encoding, partial product generation, or accumulation stages can yield substantial benefits in hardware performance. For instance, truncation of lower-order bits in partial products, simplification of Booth encoding logic, or error-tolerant designs have demonstrated varying degrees of success in reducing power consumption and chip area.

The introduction of approximate designs into Booth multipliers, however, is not without its challenges. Trade-offs between accuracy and hardware efficiency must be carefully considered to ensure that the approximate design aligns with the specific application requirements. Additionally, metrics such as error bounds, signal-to-noise ratio (SNR), and application-driven quality constraints must guide the design process. This review synthesizes insights from existing research to provide a comprehensive understanding of the principles, benefits, and limitations of approximate Booth multipliers.

Key topics covered include an overview of approximate computing, a detailed examination of traditional Booth multiplier designs, and a critical evaluation of approximate techniques implemented in Booth multipliers. Furthermore, the review investigates application-specific case studies, such as the use of approximate Booth multipliers in DSP and AI accelerators, highlighting the real-world impact and future research directions.

By providing a systematic analysis of the VLSI implementation of approximate Booth multipliers, this review aims to bridge the knowledge gap between theoretical advancements and practical design considerations. The insights presented herein are intended to serve as a foundation for researchers and engineers working on energy-efficient hardware design, enabling the development of innovative solutions that address the computational demands of next-generation applications.

II. LITERATURE SURVEY

M. H. Haider et al.,[1] Multiplier is presented for the sensor-end computing of the linear neural network, and it decreases the inference energy consumption of the linear neural network by 93.2% compared to the unmodified exact multiplier, while also reducing the space required for the calculation by 77.32%. With some tweaks to the re-encoding signal configurations, suggested technique can be utilised to reduce energy usage during inference for most Booth multipliers. it is further show that the suggested re-encoding technique, when used in conjunction with the presented multipliers, significantly improves resource utilisation compared to state-of-the-art designs while having negligible effects on the inference accuracy of neural networks.

Zhang, H., et al.,[2] The suggested technique may be used with posit multipliers in both the linear and logarithmic domains. it is implemented and study the 8/16/32-bit variants of the suggested approximation posit multipliers. The suggested approximation posit multiplier may use 16% less power than the standard posits multiplier design when used with the 16-bit posit format, which is widely used in deep learning computing. A 15% reduction in power consumption is possible with the suggested 16-bit approximate logarithm multiplier over the current state-of-the-art posit approximate logarithm multiplier.

B. K. Mohanty et al.[3] Two hybrid encoding schemes, R4R64 and R4R256, are constructed by combining radix-4 with the suggested two-term radix-64 and three-term radix-256 encoding schemes, respectively, to illustrate the utility of the selection rule. Using the suggested hybrid encoding, individual multiplier designs may be created. In spite of the fact that the output accuracy of the two suggested approximation multipliers is similar, the hardware efficiency of the multiplier design based on the presented hybrid R4R64 encoding is much higher. The presented multiplier design utilises a hybrid R4R64 encoding, which results in nearly 15% less area-delay product (ADP) and marginally less power-delay-product (PDP) than the existing approximate multiplier based on hybrid R4R256 encoding for word-length 12, but more accurately calculates output.

G. Park et al.[4] The suggested approximation Booth multiplier reduces the amount of processing energy needed while maintaining the same number of approximate bits as prior designs since the internal mistakes are naturally balanced to have zero mean. The presented approximate Booth multiplier achieves 28% and 34% energy reduction compared to the exact Booth multiplier, respectively, with negligible accuracy loss, as shown by simulation results on FIR filtering and image classification applications.

Q. Cheng et al., [5] In this research, it is designing a low-power, high-performance sparse CNN accelerator by making optimal use of available MAC units. To lessen the workload placed on the encoder and the quantity of partial products (PPs), the accelerator makes use of a radix-4 Booth multiplier for pre-encoding weights. The following are three

characteristics of the suggested accelerator. To begin, it is using the radix-4 Booth algorithm and offline weight pre-processing to cut down on the total amount of bits required for PPs. Second, to save space, it is combining eight encoders from applicable multipliers into a single pre-encoding module. After encoding non-zero weights offline, it is next create an activation selector module to selectively apply multiple-add operations to the activations corresponding to the non-zero weights.

K. Chen et al., [6] R8AS1 and R8AS2 have two approximate partial product generators designed to streamline the Radix-8 Booth square encoder. To cut down on extra space and power requirements, R8AS3 employs approximation compressors with adjustment during the partial product compression stage. The results of a power, area, and delay synthesis using 28 nm CMOS technology are shown. The suggested 16-bit designs lower the PDP by 37%, and in general the PDP is reduced by up to 51%, compared to designs in the technical literature with the same precision. The suggested approximation squarers are then used to a communication application, a square-law detector, where they obtain an SNR of about 30 dB. High performance in classification is also achieved by using the three suggested approximation squarers to the k-means clustering technique for machine learning.

F. Zhu, et al.,[7] The number of partial products generated by multiplication is greatly reduced when using high radix Booth encodings. However, high radix Booth encodings can't be used without added latency and power owing to the production of hard multiples. To avoid the generation on hard multiples, this short presenteds a radix-256 Booth encoding, which is close to perfect. Partial product pairings are generated using a partial encoding strategy and are acquired with ease by conventional shifting and complementing procedures. As a result, it is may substitute the precise encoding values with the sum of the relevant partial products. For the purpose of measuring performance, a 16 16-bit multiplier using the suggested approximate radix-256 Booth encoding has been built.

Zhang, T., et al.,[8] Approximate PP generators are developed to generate either single- or double-sided mistakes based on these techniques. Next, approximation multipliers are built utilising the error characteristics to further minimise the PPs. To offset the precision drop and the hardware burden incurred by the PP decrease, supplementary solutions are devised under the direction of a study of error consequences. Full adders are then used to compress the decreased PPs. In order to meet the needs of a wide range of applications, four distinct approximation multipliers are given.

A. S. Roy et al.,[9] In order to provide low-power design solutions for error-tolerant applications, approximate computing has been popular in recent years. There is a justifiable need to build reconfigurable approximation circuits with variable power consumption proportionate to computing accuracy due to the fact that the accuracy requirements of an application might alter dynamically at run-time. In this work, it is offer a unique approximation booth multiplier circuit with programmable precision. Partial error repair is achieved by adding sign bits to a damaged array multiplier, as implemented in this architecture.

Z. Aizaz et al.[10] There are several error-tolerant applications that might benefit from the increased performance that approximate computing could provide. In many of these contexts, multipliers serve an essential function. In this condensed work, it is present a truncation-based Booth multiplier equipped with a compensation circuit made up of selected alterations to the k-map in order to avoid the carry that results from the shortened component. Careful mapping allows for simultaneous hardware reduction and output error reduction. Truncated and Approximate Carry based Booth Multipliers (TACBM) are suggested with a variety of designs dependent on the truncation factor w in the pursuit of power and accuracy trade-off. TACBM beats state-of-the-art multipliers in precision and Area-Power efficiency. When compared to the identical Booth multiplier, TACBM(w=10) results in a 23% decrease in Area-Power product and an MRED of 0.02%. Multilayer perceptron (MLP) neural network and picture blending are used to assess the multipliers, with MLP achieving a very high level of accuracy (95.63%).

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347 - 6435 (Online) Volume 14, Issue 1, January 2025)

III. CHALLENGES

The design and implementation of approximate Booth multipliers for VLSI systems present several challenges that need to be carefully addressed to ensure the desired balance between performance and accuracy. These challenges arise due to the inherent trade-offs involved in approximate computing and the stringent requirements of modern computational systems. Below are the key challenges:

1. Accuracy vs. Efficiency Trade-off

- **Error Propagation:** Introducing approximations in Booth multipliers can lead to errors that propagate through subsequent computations, potentially affecting the overall system performance. Determining acceptable error bounds is crucial, especially in applications with strict accuracy requirements, such as scientific computations or cryptography.
- **Application-Specific Requirements:** Different applications have varying tolerance levels for errors. Designing a multiplier that strikes the right balance between accuracy and efficiency for a specific use case requires extensive profiling and optimization.

2. Optimization of Hardware Metrics

- **Power, Area, and Delay:** While approximate designs aim to minimize power consumption and chip area, achieving these goals without excessively compromising delay is a significant challenge. Ensuring that the design meets timing constraints in high-speed applications adds complexity to the optimization process.
- **Interdependency of Metrics:** Optimizing one metric, such as power, often impacts others, such as area or delay. Designers must adopt holistic approaches to ensure overall efficiency.

3. Design Complexity

- **Approximating Booth Algorithm Components:** The Booth encoding, partial product generation, and accumulation stages are interdependent. Introducing approximations in one stage without disrupting the functionality of the others requires intricate design and testing.
- **Circuit-Level Implementation:** Translating high-level approximate computing principles into practical circuit designs involves significant complexity. For example, truncation or simplification of logic may require additional components to maintain functionality, potentially offsetting the intended benefits.

4. Testing and Validation

- **Error Characterization:** Accurately characterizing and quantifying errors in approximate Booth multipliers is challenging. Traditional testing approaches may not account for the probabilistic nature of approximate designs.
- **Simulation Overheads:** Exhaustive simulations are needed to evaluate the performance of approximate multipliers across various input scenarios. These simulations can be computationally expensive and time-consuming.
- **Reliability Testing:** Ensuring reliability in the presence of process variations, noise, and environmental factors is particularly challenging for approximate designs.

5. Scalability and Portability

- **Scaling to Higher Bit-Widths:** Approximate techniques that work well for smaller bit-widths may not scale effectively for larger multipliers due to increased complexity and higher error magnitudes.
- **Technology Compatibility:** Designing approximate Booth multipliers that are compatible with different technology nodes (e.g., CMOS, FinFET) requires careful adaptation to account for process-specific characteristics.

9. Application-Specific Challenges

- **DSP Applications:** In signal processing, approximate errors may result in noticeable artifacts, such as noise or distortion, especially in audio and video processing.
- **AI Accelerators:** While AI workloads are inherently error-tolerant, certain operations, such as matrix multiplications, demand higher precision to avoid cascading errors that degrade model accuracy.

IV. CONCLUSION

VLSI implementation of approximate Booth multipliers represents a promising avenue for achieving energy-efficient, high-performance designs in modern computing systems. By leveraging approximation techniques such as bit truncation, simplified encoding, and error-tolerant architectures, these multipliers address the growing demand for resource-constrained applications like AI, IoT, and multimedia processing. However, challenges such as accuracy-efficiency trade-offs, design complexity, scalability, and system-level integration highlight the need for meticulous optimization and innovative solutions. Future research must focus on developing standardized evaluation frameworks, robust error characterization methods, and adaptive designs to ensure their widespread adoption across diverse applications. Despite the challenges, approximate Booth multipliers hold significant potential to revolutionize the landscape of power-aware and performance-driven VLSI systems, paving the way for advancements in both traditional and emerging technologies.

REFERENCES

1. M. H. Haider and S. B. Ko, "Booth encoding based energy efficient multipliers for deep learning systems," in IEEE Transactions on Circuits and Systems II: Express Briefs, doi: 10.1109/TCSII.2022.3233923.
2. H. Zhang and S. -B. Ko, "Efficient Approximate Posit Multipliers for Deep Learning Computation," in IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 13, no. 1, pp. 201-211, March 2023, doi: 10.1109/JETCAS.2022.3231642.
3. B. K. Mohanty, "Efficient Approximate Multiplier Design Based on Hybrid Higher Radix Booth Encoding," in IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 13, no. 1, pp. 165-174, March 2023, doi: 10.1109/JETCAS.2022.3229831.
4. G. Park, J. Kung and Y. Lee, "Simplified Compressor and Encoder Designs for Low-Cost Approximate Radix-4 Booth Multiplier," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 70, no. 3, pp. 1154-1158, March 2023, doi: 10.1109/TCSII.2022.3217696.
5. Q. Cheng et al., "A Low-Power Sparse Convolutional Neural Network Accelerator with Pre-Encoding Radix-4 Booth Multiplier," in IEEE Transactions on Circuits and Systems II: Express Briefs, doi: 10.1109/TCSII.2022.3231361.
6. K. Chen, C. Xu, H. Waris, W. Liu, P. Montuschi and F. Lombardi, "Exact and Approximate Squarers for Error-Tolerant Applications," in IEEE Transactions on Computers, doi: 10.1109/TC.2022.3228592.
7. F. Zhu, S. Zhen, X. Yi, H. Pei, B. Hou and Y. He, "Design of Approximate Radix-256 Booth Encoding for Error-Tolerant Computing," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 69, no. 4, pp. 2286-2290, April 2022, doi: 10.1109/TCSII.2022.3148122.
8. T. Zhang et al., "Design of Majority Logic-Based Approximate Booth Multipliers for Error-Tolerant Applications," in IEEE Transactions on Nanotechnology, vol. 21, pp. 81-89, 2022, doi: 10.1109/TNANO.2022.3145362.
9. A. S. Roy, H. Agrawal and A. S. Dhar, "ACBAM-Accuracy-Configurable Sign Inclusive Broken Array Booth Multiplier Design," in IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 4, pp. 2072-2078, 1 Oct.-Dec. 2022, doi: 10.1109/TETC.2021.3107509.
10. Z. Aizaz and K. Khare, "Area and Power Efficient Truncated Booth Multipliers Using Approximate Carry-Based Error Compensation," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 69, no. 2, pp. 579-583, Feb. 2022, doi: 10.1109/TCSII.2021.3094910.