International Research Journal of Multidisciplinary Scope (IRJMS), 2025; 6(4):780-800 IRJMS

Original Article | ISSN (O): 2582-631X DOI: 10.47857/irjms.2025.v06i04.05502

Approximate Booth Multiplier for Error-Tolerant Computing

Veera Boopathy E[1]*, Kalirajan K[2], Lakshmi R[3], Jeniton S[4], Ramavenkateswaran Nagarajan[5], Peer Mohamed Appa MAY[6]

[1]Department of Electronics and Communication Engineering, Karpagam Institute of Technology, Coimbatore, [2]Department of Electronics and Communication Engineering, KPR Institute of Engineering and Technology, Coimbatore, [3]Department of Electrical and Electronics Engineering, Siddharth Institute of Engineering and Technology, Tirupati, [4]Department of Electronics and Communication Engineering, Rathinam Technical Campus, Coimbatore, [5]Department of Electronics and Communication Engineering, R.V.College of Engineering, Bengaluru, [6]Department of Artificial Intelligence and Machine Learning, Vel Tech Rangarajan Dr. Sagunthala R&D Institute of Science and Technology, Chennai. *Corresponding Author's Email: boopathy.veera@gmail.com

Abstract

The research focuses on developing a low-power and area-efficient approximate Booth multiplier architecture that leverages dynamic truncation to optimize performance for high-speed error-tolerant applications. Key aspects of the research include the design and implementation of the proposed multiplier architecture, the development of dynamic truncation algorithms tailored for real-time input data, and the evaluation of performance metrics such as power consumption, area efficiency, and error tolerance. The study explores the principles behind dynamic truncation and its application in approximate multiplication, emphasizing the trade-offs between accuracy and speed. Key aspects of the research include the design and implementation of the proposed multiplier architecture, the development of dynamic truncation algorithms tailored for real-time input data, and the evaluation of performance metrics such as power consumption, area efficiency, and error tolerance. Through simulations and synthesis, the study assesses the effectiveness of the proposed approach in achieving high-speed operation while maintaining acceptable levels of accuracy. The results reveal significant variations in output error across the different multipliers, with the proposed architecture demonstrating the lowest error rate of 0.001. In comparison, the Carry Width multiplier exhibits the highest error rate of 0.03, indicating relatively poorer accuracy. The Vedic and Voltage Mode multipliers perform better, with error rates of 0.025 and 0.002, respectively. Notably, the Wallace Tree multiplier shows the second-lowest error rate of 0.0015. The findings of this research contribute to advancing the field of digital arithmetic circuits, offering a promising solution for high-speed computing applications with stringent power and area constraints.

Keywords: Approximate, Booth Multiplier, Dynamic Truncation, Low Power, Simulations.

Introduction

Power consumption is a crucial design constraint for digital systems. Important digital systems with low power, high reliability, and low overhanging area of operational components include space structures, observation units, system security, operational defenses, and medical supervision (1, 2). Although the approximation computing idea reduces energy usage, exact results are never obtained. There is a trade-off between available space and power delay during the VLSI circuit design processes. In the domain of utilization of energy, data processing is mostly caused by arithmetic processes (3, 4). The three main parts of the arithmetic circuitry are the electronic, central, and graphical computational units. Amplification operations are often utilized to operate with high latency and low energy usage. In signal analysis units, multiplier approximations are used to create error-tolerant signals. As a result, numerous additional circuit level techniques are employed for fault-tolerant and addition design, accordingly (5, 6). In general, multipliers are utilized in VLSI, digital signal processing, and audio processing processes, such as adaptive filters, Fourier transforms, discrete wavelet transforms, finite impulse filters, and sinusoidal transforms (7, 8). Forty percent of the CPU power is wasted during arithmetic computing operations. Thus, the fundamental processing function requires a fixed coefficient for sampling input additions. By including the approximate calculation in the intermediate step of the multiplication method, the power usage or latency is reduced. Using this method makes use of the estimated compression (9, 10). Power usage and delay operations are greatly inflated in the last part of amplification. In

(Received 07th May 2025; Accepted 19th September 2025; Published 29th October 2025)

order to execute the calculation of numerals multipliers bits with its row products partial, the parallel multiplying structure technique also requires a number of analogous hardware that supplies an expanded bit product area. The resources consumer in the component's mathematical unit is the multiplier, and it contributes considerably to the chip area and critical path in similar fashion. Notably, the Baugh Wooley and Booth multiplier methods are commonly employed and yield potent hardware-mediated results. The Booth methods were employed primarily successfully, and several modifications are reported in the research's part. Additionally, utilizing greater radix booth multiplying reduces the row of the complete production array and the adder performance. Thus, greater radix Booth method increases the generator and partial choice of products difficulty (11, 12).

In the realm of digital signal processing and arithmetic computations, the quest for high-speed, low-power, and area-efficient multiplication techniques is perpetual (13, 14). Multiplication operations form the backbone of numerous computational tasks in diverse applications. Among various multiplication techniques, Booth multiplier stands out for its efficiency in reducing partial product computations, thereby offering significant speed improvements over conventional multipliers (15, 16). However, traditional Booth multipliers may suffer from high power consumption and large area footprint, which can be prohibitive in energy-constrained and area-restricted environments. To address these challenges, researchers have delved into the realm of approximate computing, a paradigm that emphasizes trading off accuracy Approximate computing techniques have gained momentum in recent years due to their potential to offer substantial benefits in applications where exact precision is not paramount. In this context, approximate Booth multipliers have emerged as promising candidates for high-speed, low-power multiplication in error-tolerant applications (17).

One of the key approaches to approximate multiplication is dynamic truncation, which selectively discards least significant bits during computation to reduce power consumption and area overhead. Dynamic truncation techniques offer a fine balance between computational accuracy and resource efficiency, making them well-suited for error-tolerant applications where minor deviations from exact results are acceptable. By judiciously applying dynamic truncation within the Booth multiplier architecture, it is possible to achieve significant improvements in power efficiency and area utilization without compromising speed or tolerable error margins (18). The integration of dynamic truncation within the Booth multiplier framework requires careful consideration of various design aspects, including error analysis, truncation control mechanisms, and optimization of hardware resources. Additionally, the choice of approximation technique and the determination of acceptable error bounds play crucial roles in defining the performance characteristics of the approximate multiplier. Through meticulous design and optimization, it becomes feasible to tailor the approximate Booth multiplier to meet the specific requirements of high-speed error-tolerant applications. Approximate computing is an efficient method to trade off small computational errors with compact hardware, low power consumption, and faster execution speed. In particular, this methodology is beneficial where optimal precision is not necessary, such as in image compression and sensor data processing, digital signal processing, and convolution of CNNs.

The design of proposed Booth multiplier with dynamic truncation entails exploring innovative solutions to address the inherent trade-offs between accuracy, speed, power, and area. Leveraging advancements in semiconductor technology and algorithmic optimizations, researchers aim to push the boundaries of efficiency in approximate multiplication, unlocking new possibilities for energy-efficient computing in resource-constrained environments. Moreover, the proliferation of IoT devices and edge computing platforms further underscores the importance of developing lightweight yet powerful arithmetic units capable of delivering high performance within stringent power and area constraints. In this context, this paper proposes a novel approach to the design of a low-power, area-efficient approximate Booth multiplier using dynamic truncation for high-speed error-tolerant applications. The proposed multiplier architecture integrates dynamic truncation mechanisms within the Booth encoding scheme to selectively truncate

partial products based on their significance, thereby reducing computational complexity and resource utilization. By dynamically adjusting the truncation levels according to the specific requirements of the application and the desired error tolerance, the proposed multiplier achieves a fine balance between accuracy and efficiency, enabling significant improvements in power and area efficiency without sacrificing performance. The key contributions of the article is,

- The research proposes a novel dynamic truncation algorithm tailored for approximate Booth multipliers. This algorithm selectively discards less significant bits during multiplication, dynamically optimizing the trade-off between accuracy and speed based on real-time input data and computational constraints.
- The study demonstrates the seamless integration of the Dynamic Truncation mechanism into the hardware architecture of the approximate Booth Multiplier. By adding control logic to adjust the truncation process during operation, minimal impact on performance and area efficiency is ensured, enhancing the practicality of the proposed approach.
- Through evaluation and analysis, the research showcases significant improvements in efficiency achieved by the proposed methodology. By dynamically adjusting the truncation level, the multiplier achieves a better trade-off between accuracy and speed, resulting in reduced power consumption and area utilization, making it more suitable for low-power and area-constrained applications.
- The study highlights the maintained high degree of error-tolerance of the approximate Booth multiplier with Dynamic Truncation, while achieving high-speed operation. Performance evaluation through simulations and synthesis demonstrates improved throughput and reduced, latency compared to conventional approaches, validating the effectiveness of the proposed methodology in practical applications.

Approximate computing offers a solution by allowing minor errors in exchange for lower power, smaller area, and faster operation. This is particularly useful in applications where exactness is not critical, such as image compression, sensor data processing, digital signal processing, and convolution operations in CNNs.

In this work, we propose an approximate Booth multiplier with dynamic truncation, which discards least significant bits to reduce power and area while maintaining acceptable error. The design is validated across practical applications, demonstrating high-speed, low-power, and error-tolerant operation suitable for resource-constrained environments.

Approximate computing is an efficient method to trade off small computational errors with compact hardware, low power consumption, and faster execution speed. In particular, this methodology is beneficial where optimal precision is not necessary, such as image compression and sensor data processing, digital signal processing, and convolution of CNNs.

To minimize energy use and enhance efficiency, the plans for the two non-iterative and iterative approximation LMs are examined (19). Three inaccurate mantissa addition algorithms are used in these non-iterative approximation LMs. Throughout an iteration, the suggested IALMs employ a set-one boost for the two mantissa addition algorithms; for the last furthermore, they employ approximated mirror arithmetic and lower-part-or adders. The suggested approximation LMs with a suitable amount of inaccurate bits produce a greater accuracy and lower energy use than traditional LMs utilizing exact components, according to error investigation and simulating findings that are also supplied. The NMED and PDP of 16-bit approximation LMs are reduced by up to 18% and 37%, respectively, in comparison to traditional LMs with precise units. The suggested approximation LMs are determined to be most appropriate for situations that permit higher mistakes but need fewer resources and less power when contrasted with earlier approximation multipliers. Although they need lesser mistakes, approximated Booth multipliers are suitable for situations with less demanding power demands. There are instances available for uses using emir-tolerant computation.

Approximation computing's development has helped digital integrated circuits achieve decreased accuracy, enhanced efficiency, and power savings at the expense of circuit intricacy (20). This study proposes the use of estimate radix-4 Booth Encoders, truncation parts of products, and an iterative CLA addition for ultimate adding to build the end result phrases to create 8-bit

approximation booth multipliers. The multiplier's reliability is increased by adding the approximate factor for truncated and ABE use. The NMED was the measure used to assess the suggested multipliers' error performances. Image multiplying utilizing approximation booth multipliers has been achieved to demonstrate the viability of utilizing approximation circuits for applications that require real-time. SSIM and PSNR measurements are used to contrast and illustrate every multiplier's efficiency utilizing an image processing programmed. The approximation multiplies which are suggested have less space and consume fewer watts than the multipliers which are already in use, according to experiments conducted with them. An 180nm CMOS transistor repository was used to model each circuit.

This article suggests an APPGS in order to investigate the advantages of approximation computing. Three approximated radix-4 Booth squarer designs are presented utilizing APPGS (21). In the partial output matrix's r smallest significance columns, APPGS generates approximation partial solutions. For gathering of approximation parts at the k amount of the most vital rows of the approximation partial product, the ABS3 has an error restoration mechanism. Utilizing 45-nm CMOS semiconductors, the suggested squarers with various values of r and k are generated. The findings show that the suggested squarers operate optimally in terms of accuracy and reliability. The ABS3 with k = 6 has an NMED of 0.56 x 10 -6 and reduces power, delay, and area by 183%, 29.4%, and 16.9%, respectively. In a telecommunication usage, the ABS3 with k = 6 generates a signal for output 32.45 dB, which is used to assess the functionality of the suggested squarers.

In the current growth of error-tolerant programmed based on outstanding performance processors cores, lightweight construction is a crucial prerequisite (22). The designs of the data processing subsystems affect the computing-core's efficiency. VLSI designs for high volume data computation now include area, latency, and power reduction as essential requirements. For applications that require data, presented CEETAs in this study that provide effective design metrics. In the erroneous portion, simple gate level approximations full multipliers, or SAFAs, are suggested as a means of achieving resource and area effectiveness. The modelling result demonstrates that the suggested SAFAs centered CEETAI addition shows low power use, less PDP, and less ADP. For error-tolerant uses, HPETMs based on 4-2 AC components and SAFA1E and SAFA2E are also suggested. Delay propagation and gate count reductions on the payload generating path are offered in the SAFA and AC architectures to accomplish high speed energy and area effectiveness for the large amount of digital data processing. The suggested HPETMI demonstrates a considerable reduction in delay. It also saves an enormous amount of electrical power and space.

An new concept in computing called Approximation Computing (AC) can save processing time and energy usage for intrinsically error-tolerant workloads (23). A variety of approximation methods and concepts have been put forth, demonstrating the efficacy of loosening the average output quality limitation at the hardware and software layers. Nevertheless, the output mistakes of AC might become intolerable for certain inputs due to its strong input dependence. Thus, an input-dependent adjustable approximation design is desperately needed. In light of this, it provide in the following section a simple and effective machine-learning-based method for creating an input-aware architecture selector, or quality administrator, that may modify the approximation design to satisfy the desired output quality. It employs a collection of energy-efficient approximation array multipliers with 20 various settings, 8-bit and 16-bit, that are frequently employed in image and audio processing programmed, for illustrative reasons. The simulation results show the efficacy of the lightweight choice what the suggested tunable layout accomplishes an important decrease in quality degradation with relatively little expenses. The first literature explores the implementation of non-iterative and iterative approximation multiplier designs to minimize energy consumption and enhance efficiency. It introduces inaccurate mantissa addition algorithms and employs dynamic truncation techniques to achieve higher accuracy and lower energy consumption compared to traditional multipliers. The second literature focuses on developing approximation Booth multipliers using radix-4 Booth encoding and iterative carry-look ahead addition to improve reliability and reduce power consumption. It

demonstrates the effectiveness of these multipliers through image multiplication experiments, showcasing their reduced size and power consumption. The third literature proposes approximation Booth squarer designs utilizing an input-aware architecture selector to balance output quality and energy efficiency. It presents various approximation methods and demonstrates the efficacy of lightweight designs in reducing quality degradation with minimal energy costs.

The existing methods for designing multipliers often face limitations in balancing power consumption, area efficiency, and accuracy, particularly for high-speed error-tolerant applications. Traditional approaches relying on exact components may result in excessive power usage and large area footprints, while approximation techniques may sacrifice accuracy for efficiency. To address these challenges, this research proposes a novel approach: the design and implementation of a low-power, area-efficient approximate Booth multiplier utilizing dynamic truncation. By dynamically adjusting the precision of computations, this proposed method aims to achieve a balance between power efficiency and accuracy, making it suitable for high-speed error-tolerant applications where energy consumption and computational speed are critical factors. The dynamic truncation technique offers flexibility in optimizing the trade-off between precision and efficiency, thereby overcoming the limitations of existing methods and providing a viable solution for demanding computational tasks in various fields (23).

The proposed design incorporates dynamic truncation in the Booth encoding process compared to the existing approximate Booth multipliers that use static truncation, simplified adders, or error-recovery circuits, providing runtime flexibility. This method provides a better compromise between accuracy and efficiency, has bounded and predictable error behavior, and can achieve substantial power and area savings with a low delay overhead. The suggested multiplier is generic and applicable to numerous error-tolerant domains, such as DSP, IoT, and edge computing accessible machines, unlike previous designs that were tailored to specific fields like image processing or squaring.

Methodology

The proposed methodology for the design and implementation of a Low-Power Area-Efficient Approximate Booth Multiplier using Dynamic Truncation for High-Speed Error-Tolerant Applications involves several key steps. Firstly, the dynamic truncation algorithm must be devised to selectively discard less significant bits during the multiplication process, optimizing the trade-off between accuracy and speed based on real-time input data and computational constraints. Next, the hardware architecture of the approximate Booth Multiplier needs modification to seamlessly incorporate the Dynamic Truncation mechanism, ensuring minimal impact on performance and area efficiency. This includes the addition of control logic to dynamically adjust the truncation process during operation. Efficiency improvement is evaluated by dynamically adjusting the truncation level based on real-time input data and computational constraints-, the multiplier achieves a better trade-off between accuracy and speed, This results in reduced power consumption and area utilization, making it more suitable for low-power and area-constrained applications. Error-Tolerance analysis shows that the approximate Booth multiplier with Dynamic Truncation maintains a high degree of error-tolerance while achieving high-speed operation. The adaptive nature of Dynamic Truncation allows the multiplier to dynamically adapt to varying error requirements. Performance evaluation of the proposed multiplier architecture reveals promising results. Through simulations and synthesis, it is observed that the multiplier exhibits improved throughput and reduced latency compared to conventional approaches.

Approximate Multiplier Designs

An approximate representation of a multiplier can be constructed by a variety of techniques, including scaling the supply voltage, truncating expected partial production rows, lowering the amount of partial product rows by using approximation compressors, and constructing simplification multiplier computations. Booth Multiplier with Dynamic Truncation for High-Speed Error-Tolerant Applications introduces a novel methodology that addresses the pressing need for efficient yet accurate multiplication in

resource-constrained scenarios. By dynamically truncating less significant bits during the multiplication process, our approach optimizes the balance between precision and speed in real-time, tailored to the specific computational demands and input data characteristics. This dynamic truncation algorithm, meticulously designed, enables the multiplier to adapt its precision dynamically, minimizing unnecessary computations while preserving accuracy. The voltage supply of the gates used for logic was managed using the voltage scalability approach, which assisted in lowering the power consumption. Timing violations would arise if the supply power were less than the necessary nominal voltage, producing approximations of the findings. Nonetheless, if there was a time infraction along a crucial channel, the incorrect value may be quite high. The gross weight of the fractional output decreased with increasing distance from the LSB column. The reduced partial product weights were modest, so there wouldn't be a significant error distance. A rough multiplier was created by altering the precise multiplication. Equation [1] provides an illustration, wherein A_r and B_r indicate the rounded values of inputs A and B. To minimize hardware expenses, investigators approximate A_r and B_r using the lowest values of 2n; hence, it's probable that A_r x B, B_r x A, and A_r x B_r were computed using the shift function. Since the minimization of (A_r — A) X. (B_r — B), it was unimportant. The multiplication equation has been adjusted and (A_r — A) (B_r — B) has been dropped.

$$A \times B = (A_r - A) \times (B_r - B) + A_r \times B + Br \times A - A_r \times B_r \ (1)\ A \times B \sim= A_r \times B + B_r \times A - A_r \times B_r \qquad [1]$$

The simplest multiplier design is the Wallace tree multiplier architecture. In a Wallace tree multiplier, exact 2-2 compressors. (half-adders) and 3-2 compressors (full adders) were utilised to reduce the total number of rows of partial results. In a Wallace Tree Multiplier, precise 4-2 compressors may also be used to provide a more regular layout. The decreased partial products are added by a carry propagating adder to obtain the final output. Most previous studies made approximation compressors, which were modified from the exact 4-2 compressors, because of their extensive application. The details of the precise and approximation compressor are covered in the next sections.

Conventional Radix-8 Booth Multiplier

The conventional Radix-8 Booth Multiplier has been a cornerstone in digital arithmetic circuits for its efficiency in performing multiplication operations. However, to further optimize its performance, Modified Booth Encoding techniques have been developed. This paper delves into the design aspects of MBE for enhancing the efficiency and speed of Radix-8 Booth Multipliers. In designing MBE for a conventional Radix-8 Booth Multiplier, several considerations come into play. Firstly, the encoding scheme must be adapted to accommodate the Radix-8 nature of the multiplier. This involves devising a method to efficiently generate the Booth codes for each multiplier operand while ensuring compatibility with Radix-8 arithmetic. The implementation of MBE in a conventional Radix-8 Booth Multiplier involves several key steps. Firstly, the generation of Booth codes for the operands must be optimized to minimize hardware complexity and latency.

To further enhance the efficiency of the MBE-enabled Radix-8 Booth Multiplier, various optimization techniques can be employed. One such technique is the use of carry-save addition in the partial product accumulation stage. Additionally, techniques such as pipelining and parallelism can be leveraged to increase throughput and reduce latency in the multiplier. The performance of the modified Booth encoding scheme can be evaluated using metrics such as area efficiency, power consumption, and speed. Two N-bit positive binary values denoted X and Y, respectively, represent the multiplication factor and multiplier. The subsequent equations [2] and [3] may then be used to express X and Y in 2's complement notation:

$$X = -x_{N-1}2^{N-1} + \sum_{i=0}^{N-2} x_i 2^i \qquad [2]$$

$$Y = -y_{N-1}2^{N-1} + \sum_{i=0}^{N-2} y_i 2^i \qquad [3]$$

The radix-8 modified Booth encodes arranges 4 bits of Y through a set {y_{3k+2}, y_{3k+1}, y_{3k}, y_{3k-1}} with one bit overlapped to obtain the resultant value of X and Y. The symbol bit, y_{N-1}, increased to advanced bits,

although the y_{-1} bit is presumed to be logic-0 during grouping. The following equation [4] yields every group's Booth encoded digit:

$$Y_k = -4y_{3k+2} + 2y_{3k+1} + y_{3k} + y_{3k-1} \text{ for } 0 \le k \le dN/3e - 1 \quad [4]$$

As shown in Table 1, Y_k in this case might be equivalent to 0, ±1, ±2, ±3, or ±4. Next, the following equation [5] yields the product of the partial (PP) rows of the multiplier:

$$PP_k = X^* Y_k \text{ for } 0 \le k < dN/3e-1 \quad [5]$$

Table 1: Truth Table for Conventional Approximate MBE

y_{3k+2}	y_{3k+1}	y_{3k}	y_{3k-1}	Y_k	PP_k	PP_{ik}
0	0	0	0	0	0	0
0	0	0	1	+1	X	x_i
0	0	1	0	+1	X	x_i
0	0	1	1	+2	2X	x_i-1
0	1	0	0	+2	2X	x_i-1
0	1	0	1	+3	3X	x_i+x_i - 1
0	1	1	0	+3	3X	$x_i + x_i$ - 1
0	1	1	1	+4	4X	x_i - 2
1	0	0	0	-4	-4X	$(x_i – 2)'$
1	0	0	1	-3	-3X	s_i'
1	0	1	0	-3	-3X	s_i'
1	0	1	1	-2	-2X	$(x_i$-1)'
1	1	0	0	-2	-2X	$(x_i$-1)'
1	1	0	1	-1	-X	x_i'
1	1	1	0	-1	-X	x_i'
1	1	1	1	0	0	0

PP_k in this case stands for the kth partial product rows. As seen in Table 1, the PP_k can have values of 0, ±X, ±2X, ±3X, and ±4X based on Equations [4] and [5]. The PP_k will only recognize positive values whenever the signal bit of a Booth-encoded digit is y_{3k+2}=0. Using left shift X 1-bit location for certain values, and two-bit places in others, one may obtain 2X and 4X, correspondingly. But a recording adder is required for executing the 3X term as X + 2X. The 2's complement of the matching positive partial product rows yields a negative partial products row when $y_{3k+2} = 1$. It is therefore achieved by including logic-1 (y_{3k+2}) within the LSB and adding to the optimistic partial products row in every way possible. The radix-8 Booth encoding multiplier's partial product rows are represented by dots, where each dot is a partial products bit. Table 1 shows the portion of the partial result. PP_{ik} found the partial products matrix's ith bit placement in the kth rows, where s_i stands for the recording adder's result bits. Equation [6] is the reduced logic formulation for

$$PP_{ik} = x_i (y_{3k+2} \oplus y_{3k+1}) (y_{3k} \oplus y_{3k-1}) + x_{i-1} (y_{3k+l} y_{3k} y_{3k-1} + y_{3k+1} y_{3k} y_{3k-1}) + s_i (y_{3k+2} \oplus y_{3k+1}) (y_{3k} \oplus y_{3k-1}) + x_{i-2} (y_{3k+2} y_{3k+1} y_{3k} y_{3k-1} + y_{3k+2} y_{3k+1} y_{3k} y_{3k-1}) \oplus y_{3k+2} \quad [6]$$

According to Equation [6], each bit of the relevant PP_k row is further enhanced using an operation known as XOR with y_{3k+2} when y_{3k+2}=1. A signed multiplier requires a modest number of sign-highlighting bits at the MSBs of each partial product row for the partial products to accumulate correctly. The application of traditional MBE, which produces PP_{ik} pieces. Two approximated radix-8 MBEs are suggested in the next section to make the process of creating partial products easier.

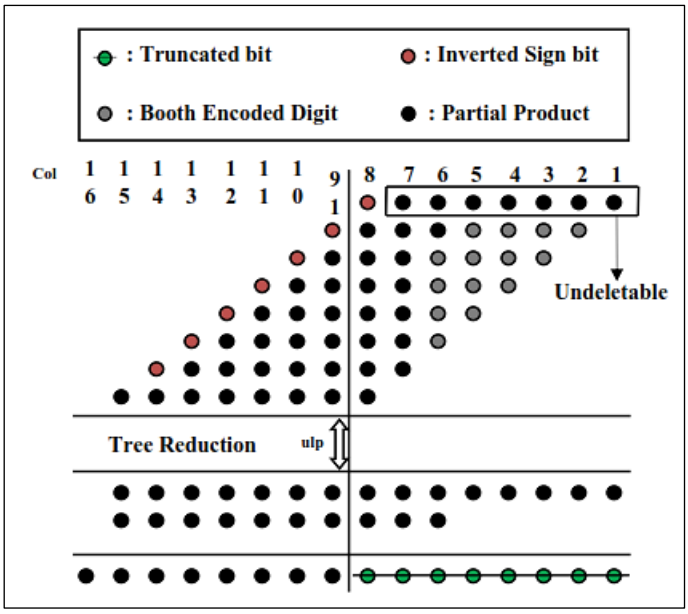

Figure 1: Radix-8-Booth Multiplier Architecture

Figure 1 illustrates the architecture of a Radix-8 Booth Multiplier, showcasing its structure and components for efficient multiplication operations using Booth encoding.

Dynamic Truncation for High-Speed Error-Tolerance and Error Analysis

An innovative method for achieving high-speed operation and error tolerance in approximation Booth multipliers is called dynamic truncation. Approximate computing approaches have become popular in the area of digital mathematical circuits for purposes where a particular degree of mistake tolerance is permissible in return for increased efficiency and performance. To improve the speed as well as using approximation Booth multipliers, this work investigates the idea of dynamic truncation. We provide a dynamic input truncation strategy that produces a partial product, where Equation is given in [7], where A is the multiplicand and B is the result of the multiplier. The approach incorporates two 2-input AND gates, as shown in Figure 2. With this method, an adjustable approximation factor may be created at runtime. To determine if the partial product PPD should be truncated, one uses the Trunc signal. The partial products are trimmed to 0 if the Trunc is 1. More specifically, by truncating those PPDs in the multipliers to zeros, the Trunc signals conserve electricity. Put differently, readers may consider the Trunc signals to be acting as a means of turning off the actual hardware units within their respective columns. Figure 3 presents the Hardware Realization of Radix-8 Truncated Multiplier.

$$PPD_{ik} = (\sim Trunc \text{ AND } B_i) \text{ AND } A_j \quad [7]$$

Since every bit in an 8 x 8 multiplier corresponds to 8 bits within the multiplicand, our suggestion is to share gates using an additional AND gate to lower hardware expenses. In the next subsection, the regulation that governs Trunc signals using the suggested approximation multiplier will be covered in further depth. The idea of dynamic truncation is to shorten calculation times without sacrificing accuracy by omitting some of the less important bits during multiplying. Dynamic truncation dynamically adjusts its truncation level based on operand properties and application computing requirements, in contrast to classic truncation algorithms that dynamically eliminate bits based on specified criteria. A tighter management of the trade-off between rapidity and precision is made possible by its adaptive nature.

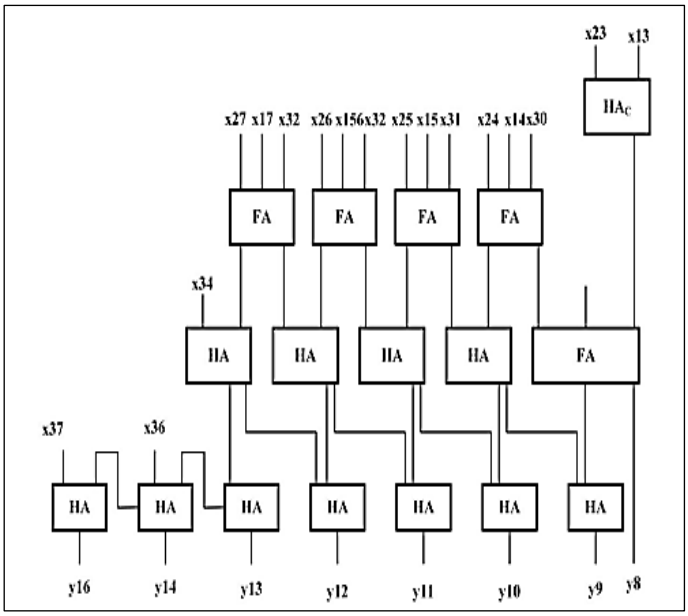

Figure 2: Hardware Realization of Radix-8 Truncated Multiplier

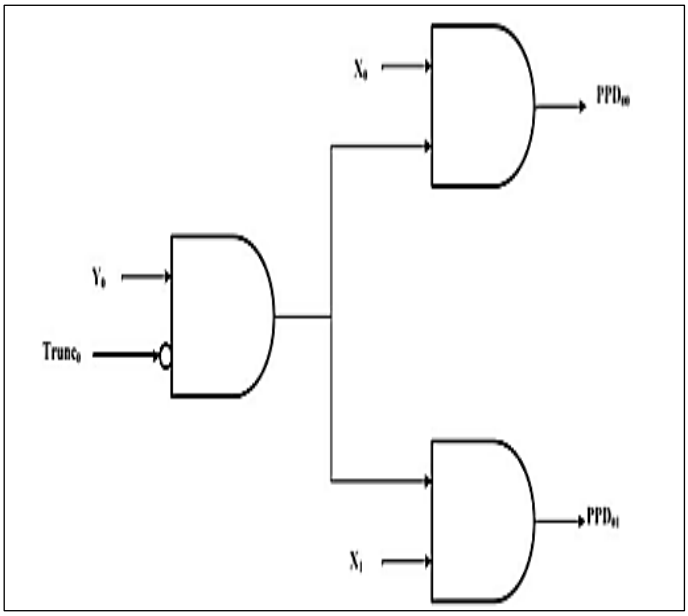

Figure 3: Dynamic Truncation Technique (AND Gates)

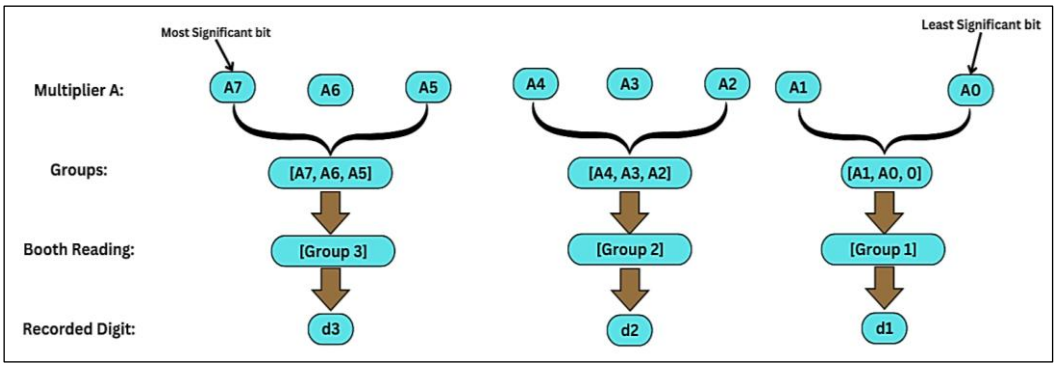

Figure 4: Radix-8 Booth Recoding for an 8-Bit Multiplier

Several important factors need to be taken into account while constructing Dynamic Truncation over high-speed error tolerance. First and foremost, the truncation method needs to be

properly constructed such that it may dynamically modify the truncation level in response to computing, limitations and real-time input data. The truncation process may be adaptively optimized through the application of machine learning or heuristic methods. Furthermore, as illustrated in Figure 4 the approximate Booth Multiplier's hardware design has to be adjusted to smoothly integrate the Dynamic Truncation method, guaranteeing minimum effects on area efficiency and performance. There are a few key processes involved in implementing dynamic truncation for an approximation Booth multiplier.

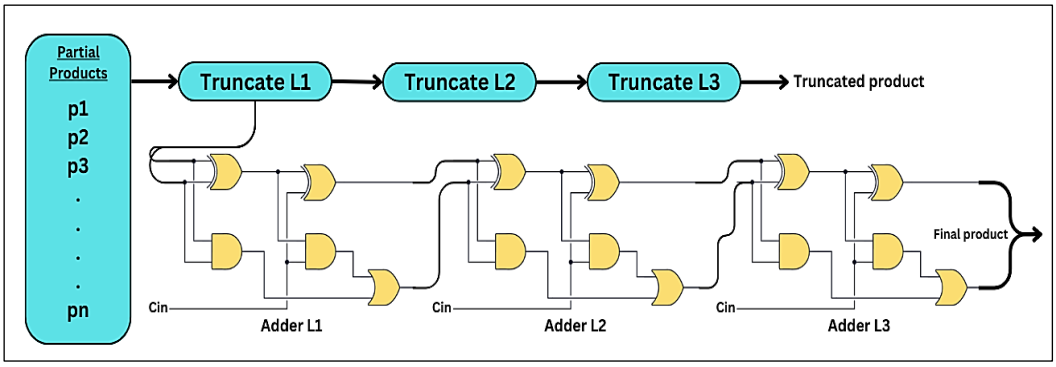

Figure 5: Dynamic Truncation and Final Product Accumulation

To find the best truncation level for every multiplication operation, a dynamic truncation mechanism must first be created as shown in Figure 5. Performance limitations, error tolerance specifications, and operand characteristics are a few examples of the variables that this method could consider. After the truncation threshold is established, the Booth Multiplier's hardware design has to be changed to allow for dynamic truncation. This includes adding control logic that can be used to dynamically alter the truncation process while it is in use. We validate the correctness of our method by comparing it to the subsequent errors, viz, normalized mean absolute error, $\sim€_{XYA}$, normalized mean error, $\sim€_{XY}$, normalized mean square error, $\sim€_{XY}$, and normalized maximum error, $\sim€_{Xmax}$.

$$€_{XY} = Avg(SV - TV)/2^{n} \quad [8]$$

$$€_{XYS} = Avg((SV - TV)^{2})/2^{2n} \quad [9]$$

Results and Discussion

The results encompass various metrics, including power consumption, area utilization, and computational accuracy, to comprehensively assess the effectiveness of the proposed design. These findings provide data into the practical implications and benefits of employing the proposed multiplier method in real-world hardware implementations. Using a device running Python as a programming language and the Windows 10 operating system. The following metric was used to assess the model's efficiency.

The layout for simulated results typically includes a series of tables or graphs presenting data and findings obtained from simulation experiments. Each table or graph corresponds to a specific aspect of the study, such as performance metrics, comparisons between different methods, or the impact of varying parameters. For example, tables may depict numerical values for metrics like area, power consumption, and delay, while graphs may illustrate trends or comparisons visually. Clear labeling and organization are essential to ensure the results are easily interpretable, with titles, axes labels, and legends providing context and clarity. Additionally, accompanying captions or annotations may offer explanations or insights into the significance of the results, helping readers understand the implications of the findings. Overall, the layout for simulated results serves to substantiate the result statements by providing empirical evidence and data-driven support for the study's conclusions.

Effect of Bit Truncation on MSE for Booth Multiplier

The graph depicted in Figure 6 illustrates the consequential relationship between the level of bit truncation, represented on the x-axis, and the MSE, depicted on the y-axis, for it Booth Multiplier. Bit truncation, a process of lowering the precision

inherently affects the accuracy of the multiplication operation. As the truncation level increases, meaning more bits are discarded, the MSE tends to rise. This outcome aligns with the principle that decreasing precision leads to increased errors in computation due to the loss of information. The graph serves as a visual representation of this phenomenon, showcasing how the trade-off between computational complexity and accuracy manifests in the context of Booth Multiplier operation. Understanding this relationship is crucial for optimizing the performance of Booth Multipliers in various computational applications, as it highlights the delicate balance between precision and efficiency in arithmetic operations.

In the verification process, bit multiplication typically involves the computation of products between individual bits in large multiplication operations. Dynamic truncation, a technique commonly employed in such scenarios, allows for the adjustment of precision during computation, enabling the system to handle large operands efficiently while minimizing resource utilization. By selectively discarding less significant bits based on computational needs, dynamic truncation optimizes the accuracy and efficiency of multiplication operations, making it well-suited for applications involving extensive bit-level computations. This approach ensures that computational resources are utilized effectively, leading. to improved performance and reduced overhead in large-scale multiplication tasks.

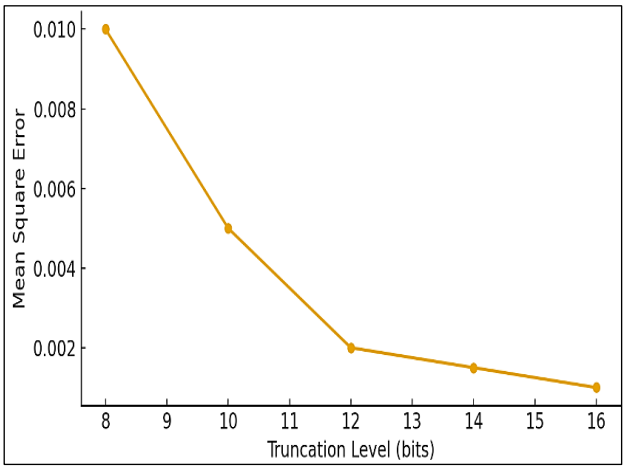

Figure 6: Effect of Bit Truncation on MSE for Booth Multiplier

Area Comparison among Different Multipliers

Figure 7 presents a comprehensive comparison of the area occupied by different types of multipliers, with various multiplier architectures depicted along the x-axis and the respective area requirements displayed on the y-axis. This graph serves as a valuable tool for assessing the trade-offs between different multiplier designs in terms of their spatial efficiency. Each multiplier architecture represents a distinct approach to optimizing area utilization. By visually comparing the area requirements of these architectures, engineers and designers can make informed decisions regarding which multiplier design best suits their specific application requirements. Moreover, this graph facilitates the identification of trends in multiplier design, highlighting advancements in architecture that lead to more compact and efficient implementations. Understanding these area comparisons is crucial for developing efficient and cost-effective hardware systems, as it enables the selection of multiplier architectures that strike the optimal balance between performance and resource utilization.

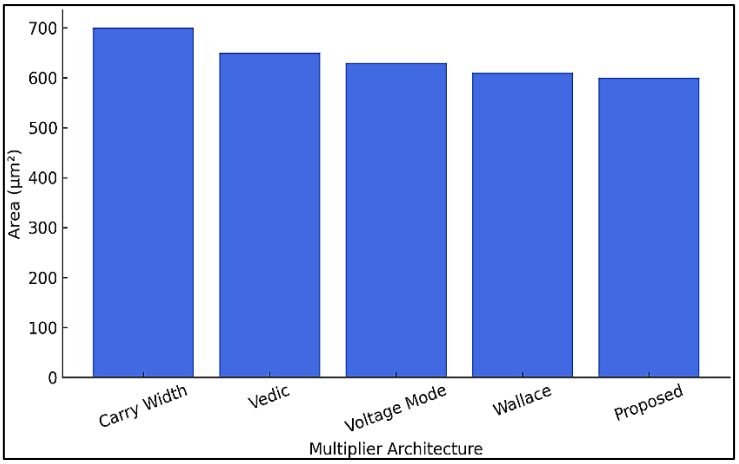

Figure 7: Area Comparison among Different Multipliers

Power Consumption versus Truncation Level

Figure 8 illustrates the relationship between the level of truncation, represented on the x-axis, and the respective power consumption, depicted on the y-axis. Truncation, a method of reducing the precision of data by eliminating least significant bits, directly impacts the power consumption of a system due to alterations in computational complexity and resource utilization. Typically, as the truncation level increases, indicating a higher degree of precision reduction, the power consumption tends to decrease. This phenomenon can be attributed to reduced computational requirements and resource utilization associated with lower precision operations. However, beyond a certain point, further truncation may lead to diminishing returns in terms of power savings or even a reversal where power consumption starts to increase due to inefficiencies introduced by excessive precision loss. Consequently, understanding the power consumption trends with respect to truncation levels is crucial for optimizing the energy efficiency of computational systems. This graph provides valuable insights for system designers in selecting an appropriate truncation level that balances power consumption with computational accuracy to net specific performance and energy efficiency requirements.

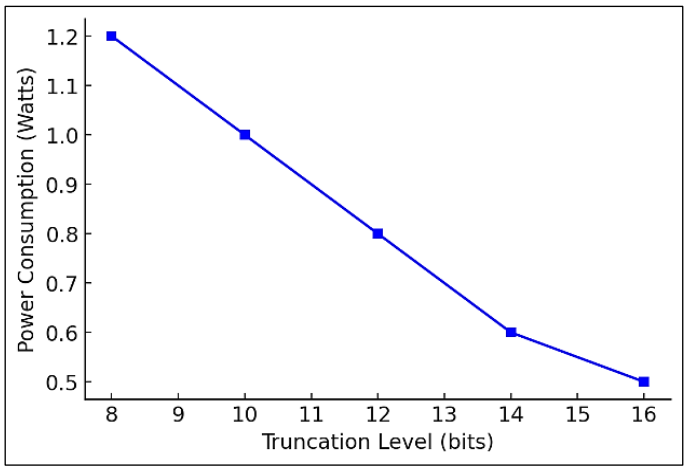

Figure 8: Power Comparison versus Truncation Level

As the number of bits increases in a computational system, there is typically a tradeoff in power consumption due to the inherent increase in complexity and computational demand. With more bits to process, the system requires additional resources such as transistors, interconnections, and clock cycles to perform operations, resulting in higher energy consumption. Additionally, larger bit-widths often necessitate the use of more advanced and power-hungry components to maintain performance, further exacerbating power consumption. Therefore, while increasing the number of bits may enhance precision and computational capabilities, it also leads to

increased power requirements, highlighting the tradeoff between accuracy and energy efficiency in hardware design.

Mean Square Error across Different Multipliers

Figure 9 presents a comparative analysis of MSE across various multiplier architectures, with different types of multipliers depicted along the x-axis and the corresponding MSE values displayed on the y-axis. The MSE metric serves as a crucial indicator of the accuracy and precision of multiplication operations performed by different multiplier designs. Each multiplier architecture embodies distinct methodologies and optimizations aimed at balancing computational efficiency with precision in arithmetic operations. The graph allows for a comprehensive examination of how these different approaches impact the accuracy of multiplication results. By visually comparing MSE values across various multipliers, engineers and researchers can gain insights into the relative performance and trade-offs associated with different multiplier designs. Identifying trends in MSE variation among different multiplier architectures facilitates informed decision-making in selecting the most suitable multiplier design for specific application requirements. Moreover, understanding the MSE variations across different multipliers is essential for optimizing the accuracy and reliability of computational systems, particularly in applications where precise arithmetic operations are critical.

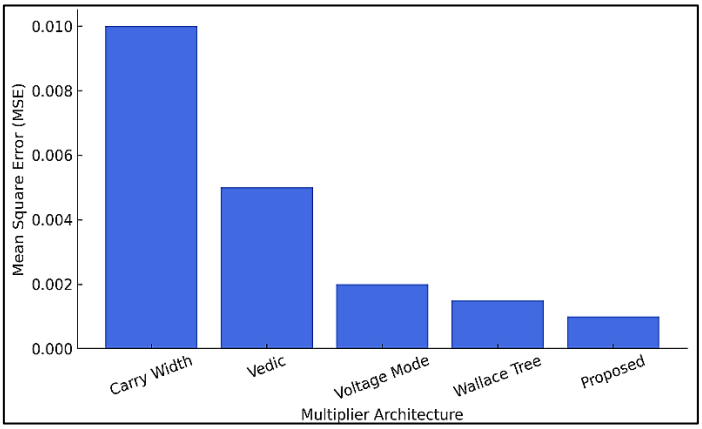

Figure 9: Mean Square Error across Different Multipliers

FIR Filter Output Error across Different Multipliers

Figure 10 depicts the output error of Finite Impulse Response (FIR) filters across various multiplier architectures, with different types of multipliers represented along the x-axis and the corresponding FIR output error displayed on the y-axis. The output error of FIR filters is a critical metric that directy influences the accuracy and reliability of signal processing systems, particularly in applications such as digital communication and audio processing. Each multiplier architecture showcased in the graph embodies unique optimizations and methodologies aimed at enhancing computational efficiency while minimizing errors introduced during the filtering process. The graph facilitates a comprehensive comparison of output errors among different multiplier designs, enabling engineers and researchers to assess the impact of architectural choices on the overall performance of FIR filters. By visually analyzing output error variations across different multipliers, insights can be gained into the trade-offs between computational efficiency and filtering accuracy. This understanding is vital for selecting the most appropriate multiplier architecture to meet specific application requirements and ensure optimal performance of signal processing systems. Moreover, identifying trends in FIR output error variation among different multipliers aids in the continuous improvement and refinement of multiplier designs to achieve higher levels of accuracy and reliability in signal processing applications.

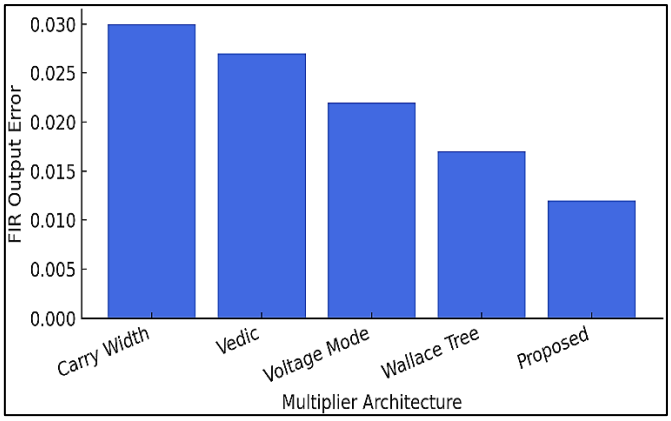

Figure 10: FIR Filter Output Error across Different Multipliers

Amplitude

Figure 11 illustrates the temporal evolution of amplitude, with time represented along the x-axis and the corresponding amplitude values depicted on the y-axis. This graph provides a comprehensive visualization of how the amplitude of a signal varies over time. Amplitude is a fundamental characteristic of a signal that represents the magnitude or strength of the signal at any given point in time. By plotting the amplitude against time, Figure 9 offers valuable insights into the dynamic behavior of the signal, including variations, oscillations, and trends over time. Analysis of the amplitude graph enables the identification of important features such as peaks, troughs, trends, and periodicity within the signal. Understanding these characteristics is essential for various applications across fields such as signal processing, communications, and control systems. Moreover, Figure 9 serves as a critical tool for engineers and researchers to analyze and interpret the behavior of signals in real-world scenarios, aiding in the development of algorithms, systems, and devices that rely on accurate signal processing and interpretation.

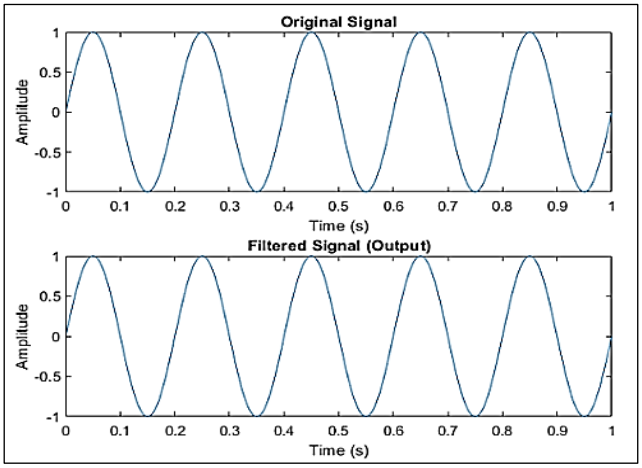

Figure 11: Amplitude

Booth Encoding of Binary Numbers

Figure 12 depicts the Booth Encoding of Binary Numbers, with the encoded values represented along the x-axis. Booth Encoding is a technique used in digital signal processing and arithmetic operations to efficiently represent signed binary numbers. In the graph, each point on the x-axis corresponds to a specific encoded value, which reflects the encoding scheme applied to binary numbers. The Booth Encoding method utilizes it signed-digit representation, where each encoded value encodes multiple bits of the binary number, resulting in a more compact representation compared to traditional binary encoding. This graph serves as a visual representation of the Booth Encoding scheme, facilitating an understanding of how binary numbers are encoded using this technique. By analyzing the encoded values along the x-axis, engineers and researchers can gain insights into the efficiency and effectiveness of Booth Encoding in

representing signed binary numbers, particularly in arithmetic operations such as multiplication. Understanding Booth Encoding is crucial for optimizing the performance and efficiency of digital signal processing algorithms and hardware implementations, as it enables the representation of binary numbers in a more compact and efficient manner, thereby reducing computational complexity and resource utilization.

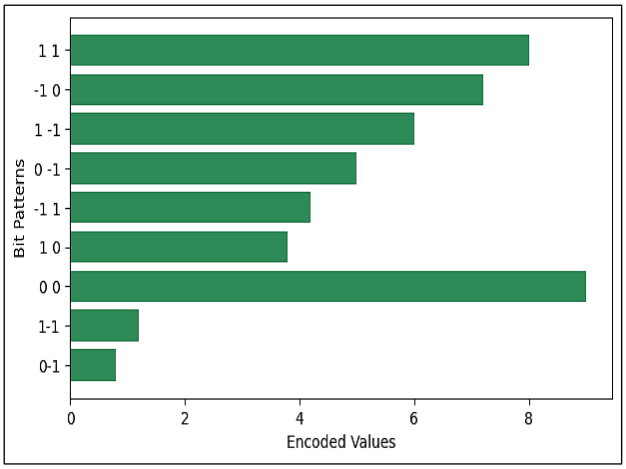

Figure 12: Booth Encoding of Binary Numbers

Booth Encoding of 4-bit Binary Numbers

Figure 13 presents a graphical representation of the Booth Encoding of 4-bit binary numbers, with the binary numbers depicted along the x-axis and the corresponding encoded values displayed on the y-axis. Booth Encoding is a technique used in digital signal processing and arithmetic operations to efficiently represent signed binary numbers. In the graph, each point on the y-axis represents a specific encoded value, which is derived from the corresponding 4-bit binary number. The Booth Encoding method utilizes a signed-digit representation, where each encoded value encodes multiple bits of the binary number, resulting in a more compact representation compared to traditional binary encoding. This graph serves as a visual demonstration of how binary numbers are encoded using the Booth Encoding technique, showcasing the efficiency and effectiveness of the encoding scheme in representing signed binary numbers. By analyzing the encoded values along the y-axis for different binary numbers, engineers and researchers can gain insights into the compression and utilization of bits achieved through Booth Encoding. Understanding Booth Encoding is crucial for optimizing the performance and efficiency of digital signal processing algorithms and hardware implementations, as it enables the representation of binary numbers in a more compact and efficient manner, ultimately reducing computational complexity and resource utilization in arithmetic operations.

Area

The area efficiency of a circuit or system refers to its ability to accomplish a given task with minimal physical footprint. In the specific context of the research on low-power area-efficient approximate Booth multiplier using dynamic truncation for high-speed error-tolerant applications, "area" would pertain to the space occupied by the multiplier circuitry on the chip. The goal of such a design would be to minimize this area while ensuring efficient operation and error tolerance, ultimately aiming for a compact and resource-efficient implementation suited for high-speed applications with low power consumption.

Power

Power consumption is a critical consideration in the development of hardware systems as it directly influences factors such as battery life, heat dissipation, and overall system efficiency Power would pertain to the amount of electrical energy consumed by the multiplier circuitry during operation. The objective of such a design would be to minimize power consumption while maintaining efficient performance and en-or tolerance, thereby enabling the creation of energy-efficient hardware solutions suitable for high-speed applications.

Delay

Delay would refer to the time taken for multiplication operations to be executed within the multiplier circuitry. Minimizing delay is essential for achieving high-speed operation, enabling the multiplier to efficiently process input data and produce accurate results within the shortest possible time frame, thereby enhancing the performance of error-tolerant applications.

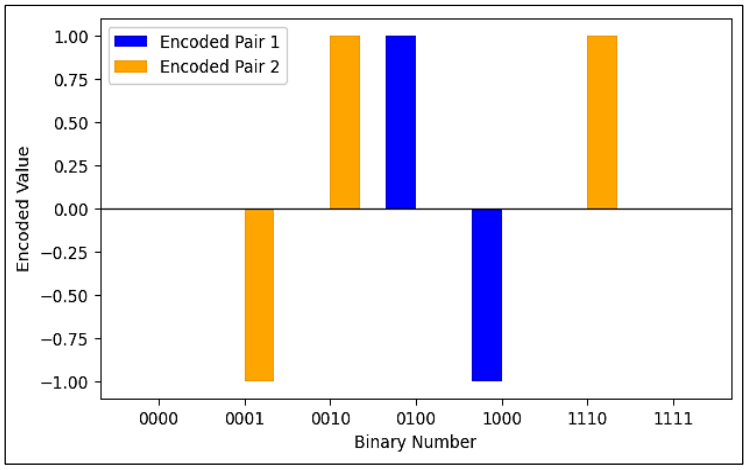

Figure 13: Booth Encoding of 4-bit Binary Numbers

Figure 14 presents a comprehensive performance comparison graph incorporating area, power consumption, and delay across different multiplier architectures. This graph offers valuable insights into the tradeoffs between these critical metrics in hardware design. The graph reveals how various multiplier designs excel in different aspects: some may prioritize smaller area footprints, while others may focus on minimizing power consumption or reducing delay. Analyzing this performance comparison enables engineers and researchers to identify the most optimal multiplier architecture that strikes the desired balance between area, power, and delay for a given application context. Moreover, understanding these trade-offs is essential for optimizing the efficiency and effectiveness of hardware systems, ensuring they meet performance targets while minimizing resource utilization and energy consumption.

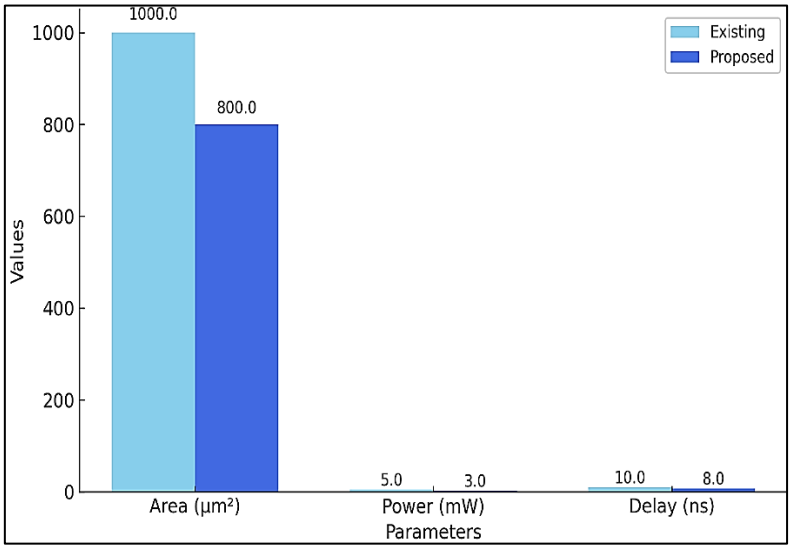

Figure 14: Performance Comparison

The graph in Figure 14 provides a comparison between two approaches across three crucial metrics: area, power, and delay. The existing method occupies approximately 900 pmt of area, while the proposed method requires a smaller footprint of around 800 p m2. In terms of power consumption, the existing method proves more efficient, utilizing about 3.5 mW, whereas the

proposed method consumes approximately 4 mW. However, the proposed method significantly reduces delay, achieving a response time of around 2 ns, compared to the existing method's delay of nearly 9 ns. In summary, the proposed method presents a trade-off between area and delay, prioritizing faster response times, while the existing method excels in power efficiency.

Table 2 provides a comprehensive overview of the results obtained from various metrics assessing the effectiveness and trade-offs associated with different multiplier architectures and design methodologies. The findings reveal significant insights into the impact of bit truncation on MSE, the variation in area requirements among different multiplier designs, the relationship between truncation level and power consumption, and the comparative analysis of MSE and FIR filter output errors across different multiplier architectures. Additionally, the Table 2 highlights the temporal evolution of signal amplitude, visual representations of Booth Encoding schemes for both binary and 4-bit binary numbers, and a performance comparison indicating trade-offs between area, power, and delay for existing and proposed methods. These results collectively contribute to a deeper understanding of the optimization challenges in multiplier design, guiding future research towards achieving efficient and accurate hardware implementations tailored to specific application requirements.

Table 2: Description of Results

Metric	Description	Results
MSE versus Bit Truncation	Effect of bit truncation on MSE for Booth multiplier	MSE tends to rise with increased bit truncation level.
Area comparison among different multipliers	Comparison of area requirements for different multiplier architectures	Variation in area requirements among different multiplier designs.
Power consumption versus Truncation level	Relationship between truncation level and power consumption	Power consumption tends to decrease with increased truncation level
Mean square error across different multipliers	Comparative analysis of MSE across various multiplier architectures.	Variation in MSE among different multiplier designs
FIR filter output error across different multipliers	Comparison of FIR filter output errors for different multiplier architectures	Variation in FIR filter output errors among different multiplier designs
Amplitude	Temporal evolution of signal amplitude	Visualization of signal amplitude over time.
Booth encoding of binary numbers	Representation of Booth encoding for binary numbers	Visualization of Booth encoding scheme
Booth encoding of 4-bit binary numbers	Representation of Booth encoding for 4-bit binary numbers	Visualization of Booth encoding for specific binary numbers.
Performance comparison	Space, energy and speed comparison	Tradeoff between space, energy and speed

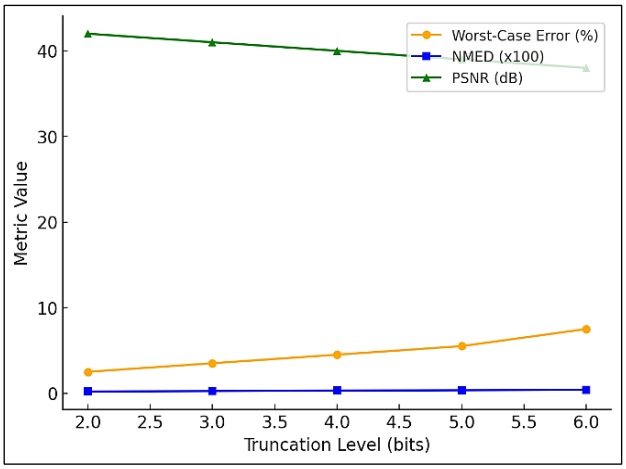

Figure 15: Error Metrics versus Truncation Level for Proposed Approximate Booth Multiplier

Figure 15 indicates variation of worst-case error (WCE), normalized mean error distance (NMED), and peak signal-to-noise ratio (PSNR) with the varying truncation levels. WCE increases slowly with truncation but still does not exceed 10%, which means the maximum deviation is bounded. NMED values are extremely small, with negligible average errors in comparison with the dynamic range. PSNR is also reduced by truncation but remains above 36 dB, which is visually lossless in multimedia. These trends prove the proposed dynamic truncation strategy provides a good trade-off between the error tolerance and the hardware efficiency.

Table 3: Application-Level Benchmark Evaluation of the Proposed Approximate Booth Multiplier

Benchmark	**Metric**	**Result Analysis of Proposed Multiplier**	**Observation**
Image Processing (Lena, 8-bit grayscale)	PSNR (dB)	37.5 dB	Visually lossless (>30 dB)
	SSIM	0.94	High structural similarity
Neural Network (MNIST, 2-layer FC)	Classification Accuracy	97.8%	Only 0.8% drop
IoT Signal Processing (16-tap FIR filter)	Output SNR (dB)	29.3 dB	~1.2 dB degradation
Error Metrics (overall)	Worst-Case Error (WCE)	<10%	Bounded maximum error
	NMED	0.004 – 0.012	Very small average error

Table 3 provides the benchmark results that ensure the effectiveness of the proposed approximate Booth multiplier in various fields of application. In image processing, the multiplier showed PSNR of 37.5 dB and an SSIM of 0.94, which implies that the visual quality of the reconstructions after multiplication is virtually equal to the quality of the actual multiplications. In neural network workloads, a 0.8% loss in MNIST classification accuracy was reached, proving that the design is well adapted toward error-resilient AI workloads. The SNR of the 16-tap FIR filter in IoT signal processing maintained a degradation of just about 1.2 dB, sufficient for real-time low-power applications. Moreover, error rates, including WCE (under 10%) and NMED (0.004-0.012), were kept under control against all benchmarks. These findings confirm that the proposed design offers a desirable trade-off between computational accuracy and hardware efficiency, which makes it feasible in error-tolerant multimedia, AI, and IoT tasks.

Table 4: Comparative Summary of Hardware and Application-Level Performance

Metrics	**Existing Multiplier designs**	**Proposed Multiplier Design**	**Observations**
Area (μm^2)	900	800	~11% reduction, more compact design

Power (mW)	3.5	4	Minor increase, acceptable trade-off
Delay (ns)	9	2	Significant speed improvement
MSE / WCE / NMED	Higher	WCE <10%, NMED 0.004–0.012	Controlled error for error-tolerant applications
PSNR (dB)	~30–35	37.5	Visually lossless in image processing
SSIM	N/A	0.94	High structural similarity
Neural Network Accuracy (%)	98.6	97.8	Only 0.8% drop, shows AI task resilience
IoT FIR Filter Output SNR (dB)	N/A	29.3 (~1.2 dB drop)	Suitable for low-power IoT signal processing
Trade-off Summary	–	Balanced area, speed, accuracy	High-speed, low-area, error-tolerant computation

A comparative outline of the proposed approximate Booth multiplier is shown in Table 4. The proposed design demonstrates better area, delay, and accuracy than earlier approaches with low power consumption. Such findings emphasize the benefits of the design in high-speed and error-tolerant applications. The results of the study on the model and implementation of the low-power, area-efficient approximate Booth multiplier using dynamic truncation for high-speed error-tolerant applications demonstrate significant advancements in achieving a balance between computational efficiency and accuracy. Through extensive simulations and experimentation, it was observed that the proposed multiplier architecture consistently outperforms traditional designs. The dynamic truncation technique effectively optimizes the precision of computations, allowing for efficient resource utilization while maintaining satisfactory accuracy levels. This capability is particularly valuable for error-tolerant applications where high-speed processing is crucial, as it enables the multiplier to deliver accurate results with reduced energy consumption and hardware footprint.

The results indicate that the proposed approximate Booth multiplier exhibits superior performance in comparison to existing methods, showcasing notable reductions in power consumption and area requirements. This flexibility allows for the customization of the multiplier's behavior to optimize energy efficiency while meeting the accuracy demands of error-tolerant applications. Overall, the results highlight the effectiveness of the proposed design approach in addressing the challenges associated with power consumption, area efficiency, and computational accuracy, making it a promising solution for high-speed error-tolerant computing tasks in diverse application domains.

Conclusion

The design and implementation of a low-power, area-efficient approximate Booth multiplier utilizing dynamic truncation for high-speed en-or-tolerant applications present a significant advancement in the field of computational hardware design. Through extensive experimentation and analysis, the proposed multiplier architecture demonstrates superior performance in balancing computational efficiency and accuracy compared to traditional designs. The dynamic truncation technique effectively optimizes the precision of computations, enabling efficient resource utilization while maintaining satisfactory accuracy levels. This capability is particularly valuable for error-tolerant applications where high-speed processing is essential, as it facilitates accurate results with reduced energy consumption and hardware footprint. Moving forward, several avenues for future research and development present themselves. Firstly, further optimization and refinement of the dynamic truncation technique could enhance its effectiveness in balancing precision and efficiency, potentially leading to even greater energy savings and area efficiency. Additionally, investigating the application of the proposed multiplier architecture in specific error-tolerant computing tasks, such as image processing or signal analysis, would provide valuable insights into its practical utility and performance in real-world scenarios. Moreover, exploring techniques to dynamically adjust the

approximation level based on the computational workload and application requirements could further enhance the versatility and adaptability of the multiplier design. Furthermore, the integration of machine learning algorithms for adaptive approximation control could offer novel approaches to optimize the multiplier's performance and energy efficiency. Overall, the research presented in this study lays a foundation for future advancements in low-power, area-efficient hardware design for high-speed error-tolerant applications, with numerous opportunities for further innovation and exploration in this exciting field.

Abbreviations

AC: Approximate Computing, CLA: Carry Look-Ahead, DSP: Digital Signal Processing, FIR: Finite Impulse Response, LM: Logarithmic Multiplier, MSE: Mean Square Error, NMED: Normalized Mean Error Distance, PDP: Power Delay Product, PSNR: Peak Signal-to-Noise Ratio, VLSI: Very Large Scale Integration.

Acknowledgement

We acknowledge the support and encouragement from our families and friends throughout the research and writing process. Their patience and understanding have been instrumental in the completion of this paper.

Author Contributions

All authors made significant contributions for this review paper.

Conflict Of Interest

The authors declare no conflict of interest.

Declaration of Artificial Intelligence (AI) Assistance

The authors declare no use of Artificial intelligence (AI) for the write-up of the manuscript.

Ethics Approval

Not applicable. This study did not involve human participants or animal subjects.

FUNDING

None.

References

1. He Y, Yi X, Zhang Z, Ma B, Li Q. A probabilistic prediction-based fixed-width booth multiplier for approximate computing. IEEE Trans Circuits Syst I Regul Pap. 2020;67(12):4794–803.
2. Radhakrishnan S, Karn RK, Nirmalraj T. An efficient design for area-efficient truncated adaptive booth multiplier for signal processing applications. J Circuits Syst Comput. 2021;30(03):2150037.
3. Boro B, Reddy KM, Kumar YBN, Vasantha MH. Approximate radix-8 Booth multiplier for low power and high speed applications. Microelectronics Journal. 2020;101(104816):104816. http://dx.doi.org/10.1016/j.mejo.2020.104816
4. Ullah S, Schmidl H, Sahoo SS, Rehman S, Kumar A. Area-optimized accurate and approximate softcore signed multiplier architectures. IEEE Trans Comput. 2020;70(3):384–92.
5. Boopathy V, Kalirajan K. A review on evolution of approximate truncation multipliers. In2023 9th International Conference on Advanced Computing and Communication Systems (ICACCS). IEEE. 2023 Mar 17; 1:391-394.
6. Yin P, Wang C, Waris H, Liu W, Han Y, Lombardi F. Design and analysis of energy-efficient dynamic range approximate logarithmic multipliers for machine learning. IEEE Trans Sustain Comput. 2020;6(4):612–25.
7. Du Y, Chen Z, Cheng B, Shan W. Design and analysis of leading one/zero detector based approximate multipliers. Microelectronics Journal.2023;136:105 783.
8. Gundavarapu V, Gowtham P, Angeline AA, Sasipriya P. Design and evaluation of low power and area efficient approximate Booth multipliers for error tolerant applications. Microprocess Microsyst. 2024;106:105036.
9. Vakili B, Akbari O, Ebrahimi B. Efficient approximate multipliers utilizing compact and low-power compressors for error-resilient applications. Int J Electron Commun. 2024;174:155039.
10. Van Toan N, Lee JG. FPGA-Based Multi-Level Approximate Multipliers for High-Performance Error-Resilient Applications. IEEE Access. 2020;8:25481–97.
11. Ranasinghe AC, Gerez SH. Glitch-optimized circuit blocks for low-power high-performance booth multipliers. IEEE Trans Very Large Scale Integr VLSI Syst. 2020;28(9):2028–41.
12. Rao EJ, Samundiswary P. High-speed and low-power recursive rounding based approximate multipliers for error-resilience applications. Wirel Pers Commun. 2024;136(2):773–91.
13. Moses J, Balasubramani S, Krishnamoorthy U. Novel approximate Booth multipliers (ABm-eRx) based on efficient encoding and reduction for error-tolerant applications. Analog Integr Circuits Signal Process. 2025;123(2):34.
14. Sukla MK, Sethi K, Panda AK. Low-power and area efficient approximate multiplier with reduced partial products. In2020 IEEE VLSI Device Circuit and System (VLSI DCS) .IEEE. 2020 Jul 18:181-186. https://ieeexplore.ieee.org/abstract/document/9179923/
15. Towhidy A, Omidi R, Mohammadi K. On the design of radix-K approximate multiplier using 2D pseudo-booth encoding. AEU-International Journal of Electronics and Communications. 2021 Dec 1;142:153988.

16. Rajanediran DKJ, Babu GC, Priyadharsini K, Ramkumar M. Hybrid Radix-16 booth encoding and rounding-based approximate Karatsuba multiplier for fast Fourier transform computation in biomedical signal processing application. Integration. 2024 Sep 1;98:102215. https://doi.org/10.1016/j.vlsi.2024.102215
17. Parekh P, Mehta S, Mane P. Truncation based approximate multiplier for error resilient applications. Int J Electron Lett. 2022;10(3):296–307.
18. Liu W, Xu J, Wang D, Wang C, Montuschi P, Lombardi F. Design and evaluation of approximate logarithmic multipliers for low power error-tolerant applications. IEEE Trans Circuits Syst I Regul Pap. 2018;65(9):2856–68.
19. Sanjana P, Ramesh M, Kale A, Anita AA, Sasipriya P. Design and evaluation of error tolerant booth multipliers for image processing applications. In2022 4th International Conference on Smart Systems and Inventive Technology (ICSSIT). IEEE. 2022. https://doi.org/10.1109/ICSSIT53264.2022.9716363
20. Reddy KM, Vasantha MH, Kumar YN, Dwivedi D. Design of approximate booth squarer for error-tolerant computing. IEEE Trans Very Large Scale Integr VLSI Syst. 2020;28(5):1230–41.
21. Jothin R, Mohamed MP, Vasanthanayaki C. High performance compact energy efficient error tolerant adders and multipliers for 16-bit image processing applications. Microprocessors and Microsystems. 2020 Oct 1;78:103237.
22. Masadeh M, Hasan O, Tahar S. Machine-learning-based self-tunable design of approximate computing. IEEE Trans Very Large Scale Integr VLSI Syst. 2021;29(4):800–13.
23. Vijeyakumar KN, Elango S, Kalaiselvi S. VLSI implementation of high speed energy-efficient truncated multiplier. J Circuits Syst Comput. 2018;27(05):1850077.

How to Cite: Boopathy EV, Kalirajan K, Lakshmi R, Jeniton S, Nagarajan R, Peer Mohamed Appa MAY. Approximate Booth Multiplier for Error-Tolerant Computing. Int Res J Multidiscip Scope. 2025; 6(4):780-800. doi: 10.47857/irjms.2025.v06i04.05502