ISSN: 2395-1680 (ONLINE)
DOI: 10.21917/ijme.2026.0383

DESIGN AND IMPLEMENTATION OF APPROXIMATE PARALLEL PREFIX MULTIPLIER (AXPPM)

Chintam Shravan, T. Srilatha, O. Anjali and V. Madhuri
Department of Electronics and Communication Engineering, Rajiv Gandhi University of Knowledge Technologies, India

Abstract

Approximate multipliers are often utilized in cases where small computation errors are tolerated in return for improved performance. This paper builds and evaluates an effective approximate multiplier that decreases power consumption, delays, and hardware complexity. The proposed design reduces the amount of logic used while maintaining a reasonable degree of correctness by changing some product production and accumulation phases. Performance is evaluated using parameters such as error metrics, power consumption, propagation delay and area usage. Experimental results show that, the BrentKung approximate multiplier gains up to 76% reduction in area, 57% reduction in power, and 62% reduction in delay, making it the most balanced design. The Ladner-Fischer design gives the highest improvement in power and PDP, with reductions of approximately 81%, while Kogge-Stone provides the highest area reduction of 80%. The designs are hence suitable for low-power VLSI systems and energy-efficient embedded applications.

Keywords:

Approximate computing, Approximate multipliers, Parallel prefix adders (PPA), AxPPM, AxPO's

1. INTRODUCTION

One of the most fundamental operations in mathematics in em bedded systems, digital signal processing (DSP), image processing, and machine learning is multiplication. Multiplier speed and power utilization have a major impact on total effectiveness in high-performance VLSI systems. High accuracy is guaranteed by standard exact multipliers, although they frequently result in higher area, delay, and energy consumption. However, a lot of modern error-resistant applications, like neural networks and multimedia processing, can withstand minor computational errors. Approxima tion computing approaches, especially approximation multipliers, have grown up as a result, trading computational exactness for benefits in power, time, and complexity of the hardware [6], [8].

High speed multiplier architectures may utilize parallel prefix adders (PPAs) to speed up the last carry propagation stage. The adders by Peter M.Kogge and Harold S.Stone (Kogge-Stone), Richard P.Brent and H.T.Kung (Brent-Kung), John Sklansky (Sklansky), Richard E.Ladner and Michael J.Fischer(Ladner-Fischer), Han Carlson (Han-Carlson) are some of the most essential prefix architectures. The fan-out, silicon area, wiring complexity, and logic depth of these systems will be changed. As an example, Brent-Kund architecture will have area savings with a small bit of more delay [2], whereas the Kogge-Stone architecture gives fast speed and low logic depth at the cost of larger area and circuit difficulty [1]. Ladner-Fischer delivers a scalable trade-off between connection difficulty and logic levels, whereas the Sklansky topology decreases logic levels but raises fan-out [3], [4]. Hybrid structures namely Han-Carlson merge the advantages of Kogge-Stone and Brent-Kung architectures to compromise performance and hardware excess [5].

The Brent–Kung, Kogge–Stone, Sklansky, Ladner–Fischer, and Han–Carlson prefix designs are utilized in this research to compare and evaluate approximation multipliers. To figure it the ideal tradeoff between hardware effectiveness and analytical accuracy, the suggested architectures are reviewed in terms of error metrics, power consumption, area usage, and propagation delay. For better computational performance, these prefix adders usually go into the very last addition step in multiplier design. Additional power usage and critical path time savings can be obtained by adding approximation methods into the prefix processing network, partial product lowering, or partial product generation. Approximate parallel prefix adders and multipliers have been proven in recent studies to supply large savings in energy with acceptable error characteristics, which makes them appropriate for low-power VLSI and FPGA implementations [7], [9], [10].

2. RELATED WORK

The growth of high-speed arithmetic units, particularly approximate multipliers applying parallel prefix structures such as Brent Kung (BK), Kogge–Stone (KS), Sklansky (SK), Ladner–Fischer (LF), and Han–Carlson (HC) architectures, has been greatly affected by recent developments in low-power VLSI design and computing algorithms [9]. Fast and efficient partial product accumulation can be achieved using standard accurate multipliers mixed with these prefix networks; but these networks have significant power usage, high hardware costs, higher interconnection complexity, and larger silicon area [1]. These disadvantages limit their application in energy-constrained systems involving machine learning accelerators, integrated DSP processors, and image processing modules. To try to overcome all of this, approximate multiplier designs based on BK, KS, SK, LF, and HC architectures implement controlled approximation in reducing stages, the final prefix addition network, or partial creation of products [2]. The prefix level in the least significant bit (LSB) region decreases in approximation BK and HC multipliers to reduce switching activity and logic levels, which reduces dynamic power consumption [4]. Conversely, approximation KS and SK multipliers lower wire overload and critical path time by easing carry propagation by eliminating unique prefix nodes in lower-order bits [3]. LF-based approximation multipliers are using the strategies of selective carry truncation and speculative carry generation to lower hardware complexity yet retain a suitable degree of computational accuracy [5].

Compared these approximation multiplier implementations to their perfect counterparts, experimental investigations reveal significant savings in silicon area, power consumption, and

Power–Delay Product (PDP) [6]. For the purpose to boost energy use while keeping appropriate error metrics such as Mean Absolute Error (MAE) and Normalized Mean Error Distance (NMED), hybrid multiplier architectures integrate accurate computation in the most significant bit (MSB) region with approximate computing in the LSB region [10]. All things looked at, approximate multipliers based on BK, KS, SK, LF, and HC prefix architectures give a promising approach for building fast, low-power arithmetic units suitable for error-resistant applications in embedded DSP systems, multimedia processing, and neural network inference.

3. METHODOLOGY

3.1 SYSTEM OVERVIEW

An approximate parallel prefix multiplier (AxPPM) has three main elements: final carry-propagate expansion, partial product elimination, and partial product output. The Least Significant Bit (LSB) space properly applied for approximation to raise energy usage level while the Most Significant Bit (MSB) area ensures exact calculation for total numerical accuracy. Partial Product Generation: For two N-bit operands *A* and *B*, the partial products are generated as

$$PP_{i,j} = A_i \cdot B_j \tag{1}$$

All partial products are computed in parallel using AND gates to reduce switching activity and logic complexity, approximation is applied in the LSB region by simplifying lower-weight partial products as:

$$PP_{i,j} \approx A_i \tag{2}$$

$$PP_{i,j} \approx B_i \tag{3}$$

This truncation-based simplification lowers dynamic power and transistor count. Partial Products reduction: The generated partial products are compacted using a tree-based reduction network. In the approximate region, simplified compressors are employed.

Exact full-adder equations:

$$\text{Sum} = A \oplus B \oplus C \tag{4}$$

$$\text{Carry} = AB + BC + AC \tag{5}$$

Approximate reduction logic:

$$\text{Sum} = A \oplus B \tag{6}$$

$$\text{Carry} \approx 0 \tag{7}$$

By suppressing carry propagation in lower columns, the reduction tree depth and switching power are significantly decreased. Prefix-Based Final Addition: After reduction, the remaining two rows are added using a Parallel Prefix Adder (PPA).

Propagate and generate signals are defined as

$$P_i = A_i \oplus B_i \tag{8}$$

$$G_i = A_i \cdot B_i \tag{9}$$

The associative prefix operator is voiced as

$$P = P_i \cdot P_{i+1} \tag{10}$$

$$G = G_{i+1} + (P_{i+1} \cdot G_i) \tag{11}$$

To improve energy efficiency, Approximate Prefix Operators (AxPOs) are in included in the LSB region:

$$P \approx P_{i+1} \tag{12}$$

$$G \approx G_{i+1} \tag{13}$$

This eliminates AND/OR logic in approximate prefix nodes, reducing logic depth and power consumption.

3.2 EXPERIMENTAL SETUP AND PERFORMANCE EVALUATION

The sixteen-bit (16x16) multipliers with a configurable approximation parameter *K* were executed. *K* denotes the number of approximation LSB columns, while the upper bits continue accurate. The evaluation metrics were assessed: Area (Gate count/LUT utilization), Power consumption, Critical Path Delay, Power-Delay Product. Designs for ASIC validation were generated using a 65-nm CMOS common cell library at a supply voltage of 1.25 V. To obtain actual switching activity for exact power assessment, post-synthesis simulations were performed. Implementations have been defined onto a Xilinx Virtex-7 device for FPGA-based verification. Analysis was done on LUT utilization, delay, and energy efficiency. 100,000 pseudorandom 16-bit input vectors served to compare to an exact multiplier in order to scientifically evaluate error patterns. Application-Based Case Studies: To prove practical applicability, AxPM designs were combined into: FIR filter accelerator, SSD-based video processing accelerator.

$$\text{SNR} = 10\log_{10}\left(\frac{\text{Signal}^2}{\text{Noise}^2}\right) \tag{14}$$

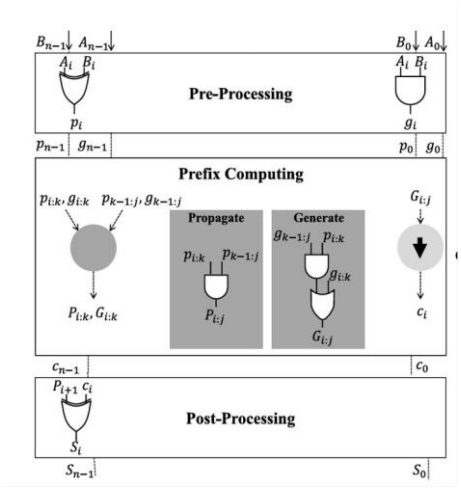
In the FIR case study, performance breakdown due to approximation was calculated using Signal-to-Noise Ratio (SNR). For the SSD accelerator, output quality was rated using: Normalized Cross-Correlation (NCC), Peak Signal-toNoise Ratio (PSNR).

3.3 EXACT PARALLEL PREFIX MULTIPLIERS (PPM)

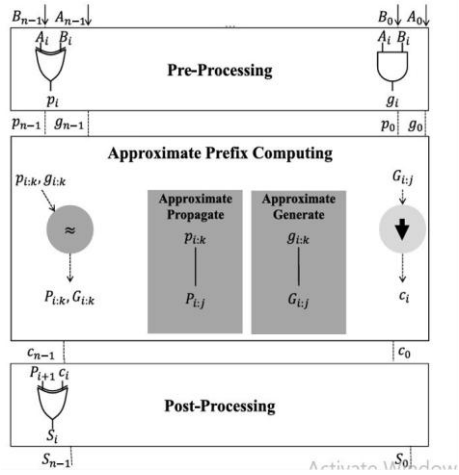
Absolute precision multiplication with accurate and free of errors carry propagation at each step of operation is the aim of exact parallel prefix multipliers. Partial product production, partial product reduction (often accomplished with carry-save adders), and a last carry-propagate adder (CPA) contains the three major steps of a typical design. The last CPA step's structure has a major effect on the multiplier's all over performance. In comparison to ripple-based systems, the critical path delay can be largely reduced through the usage of structured parallel prefix networks, which allows the carry computation to be finished in logarithmic time. Exact parallel prefix multipliers are commonly used in high-performance processors, floating-point units, cryptographic accelerators, and scientific computing systems where computation exactness is important because of their predictable operation and ensured numerical correctness. These generic blocks of adders are applied in exact and approximate multipliers shown in Fig.1.

3.3.1 Brent-Kung Multiplier:

The Brent-Kung multiplier, among the most famous architectures, utilizes the Brent-Kung adder in the carry-propagate stage. The logarithmic delay provided by this topology is 2 log2 W − 1. W-1 with limited fan-out and a comparably low number of prefix nodes.

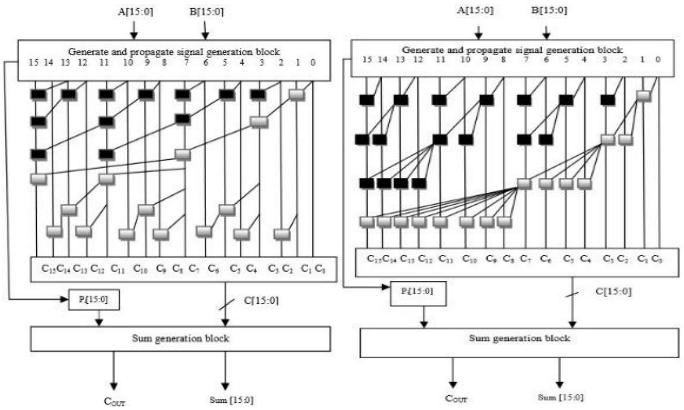


(a)

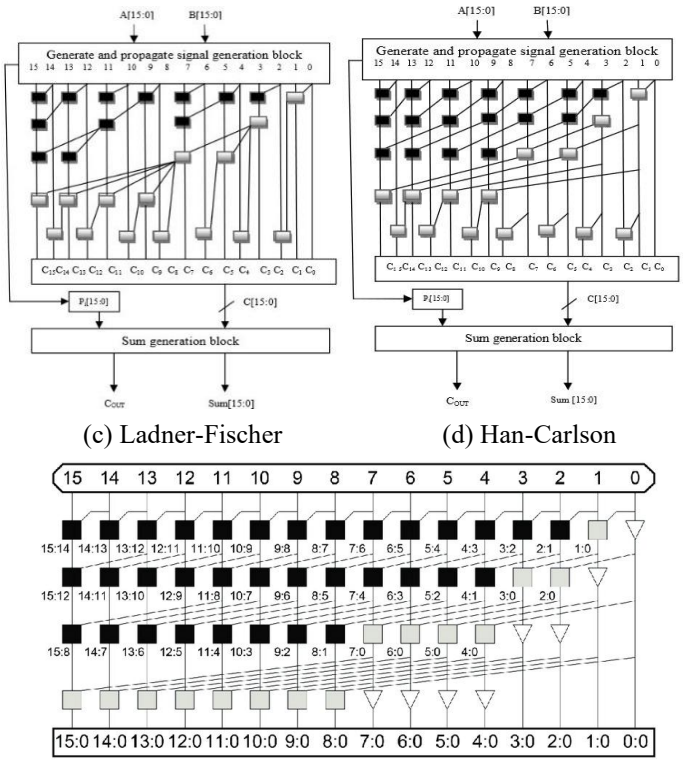


(b)

Fig.1. Pre-processing, Prefix computation, and post-processing steps of (a) Exact PPA and (b) AxPPA proposal



(a) Brent-Kung (b) Sklansky



(c) Ladner-Fischer (d) Han-Carlson

(e) Kogge-Stone

Fig.2. Architectures of Multipliers

It is particularly attractive for VLSI implementations because of the planned and uniform connecting pattern, which saves wiring effort and routing latency. As a result, the Brent-Kung multiplier delivers a fair trade-off between power and silicon size, making it ideal for mediumto high-performance arithmetic units that require good architecture adaptability. Different architectures of multipliers are shown in Fig.2.

3.3.2 Kogge-Stone Multiplier:

One of the least critical paths among prefix adders is made by this connection fully parallel carry algorithm. The Kogge-Stone multiplier is commonly used in performancecritical data paths because it may reach very high operating frequencies. Still, the huge interconnection network and high number of prefix nodes lead to higher area consumption and routing cost. In high-speed processors design when delay elimination is a key goal, it is still the best choice even because of this drawback. Its minimal logic depth of log2(W), the Kogge-Stone multiplier combines the Kogge-Stone adder in the last step of addition.

3.3.3 Ladner Fischer Multiplier:

It utilizes the benefits of the LadnerFischer adder structure. Compared to the Kogge-Stone designs, this topology gives logarithmic delay with little fan-out and more prefix nodes. Its flexible prefix organization allows designers to maintain the speed and complexity of the hardware effectively. In addition to this, the Ladner-Fischer multiplier offers an effectiveness trade-off between area and performance, making it suitable for application that has controlled wiring complexity while retaining high-speed functioning. Because of this, it is far faster than basic designs like ripple-carry adders. By combining multiple carry bits

in parallel, the prefix design rapidly carries out the carry propagation, leading in this logarithmic delay.

3.3.4 Sklansky Multiplier:

The divide-and-take prefix design, frequently referred to as the Sklansky adder, is implemented by the Sklansky multiplier. By rapidly merging signals in the initial stages, this structure reduces logic depth and gains quick carry computing. This results in high-speed operation and least logic levels, but it additionally raises fan-out in the first prefix stages, which could result in higher switching power and capacitive stress. Yet, for applications where fan-out boundaries can be effectively managed, the Sklansky-based multiplier has competitive delay performance when properly optimized.

3.3.5 Han-Carlson Multiplier:

In the final carry-propagate step, the Han-Carlson multiplier contains the hybrid Han-Carlson adder topology. When compared to completely parallel networks, this structure obtains logarithmic delay with few prefix nodes by merging elements of Brent-Kung and Kogge-Stone designs. They offer more flexibility for larger operand widths, balanced fan-out, and controlled wiring overhead. As a result, the Han-Carlson multiplier gives a helpful trade-off between area, speed, and routing complexity.

3.4 APPROXIMATE PARALLEL PREFIX MULTIPLIERS (AXPPM)

In order to reduce complexity, critical path time as well as power consumption, approximate parallel prefix multipliers apply calculated improvements to the partial product reduction and carrypropagation steps. Partial product generation, reducing with carrysave designs, and the last carry-propagate adder (CPA) compose a standard multiplier structure. In approximate designs, the MSB zone is kept as accurate to reduce the maximum error magnitude, while approximation usually takes place in the LSB partial product columns or within the chosen prefix nodes of the final CPA step. These multipliers are appropriate for error-tolerant applications because they develop large energy savings with bounded computational error by selectively easing carry accuracy in non-critical regions.

3.4.1 Brent-Kung Multiplier:

In the final carry-propagate stage, the approximate Brent–Kung multiplier (AxPM-BK) utilizes a reduced version of the Brent-Kung adder. Approximate Prefix Operators (AxPOs) can be utilized in place of individual prefix nodes in the LSB region in this structure, and truncated or modified lowering logic may be applied to lower partial product columns. To prevent worst-case errors, the MSB area preserves an exact Brent-Kung architecture. As a result, AxPM-BK gives an appropriate agreement between inaccuracy, delay, and area. Because of the Brent–Kung topology's significantly reduced fan-out and small wire complexity, this approximate image provides lowered logic depth in the LSB stages, decreased prefix node counts, and improved energy savings with a little loss in computation accuracy.

3.4.2 Kogge-Stone Multiplier:

Inspired on the Kogge-Stone adder, the approximation Kogge-Stone multiplier (AxPM-KS) includes reductions into the carry-propagate process. Approximation typically refers by truncating lower partial product columns, eliminating non-critical carry paths, and reducing the total number of black and gray prefix cells in the LSB region due to the Kogge–Stone design is marked by minimal logic depth and fully parallel carry computation. By significantly lowering switching activity and connection complexity, these modifications decrease dynamic power consumption. Since the specific Kogge–Stone design gives a time delay, log2(W) an approximation variation achieves significant energy savings while preserving rapid performance in the MSB zone.

3.4.3 Ladner Fischer Multiplier:

In the last one stage, a modified version of the Ladner-Fischer adder is executed by the approximate Ladner–Fischer multiplier (AxPM-LF). The Ladner–Fischer design's active and responsive prefix topology allows a new type of approximation applying the clarified structure of lower-level prefix nodes and the first decreased carry computing depth. Hardware difficulty is more decreased by carefully truncating LSB partial products. The Ladner-Fischer network's approximate versions usually achieve suitable delay elimination and regulated area savings yet keeps proper error properties because it automatically maintains fan-out and logic depth. This frequently gives increases to an acceptable Pareto trade-off between exactness and energy savings.

3.4.4 Sklansky Multiplier:

The Sklansky adder topology is utilized to generate the approximation Sklansky multiplier (AxPM-SK). This structure displays high fan-out in its initial phases but lowers logic depth from strong signal combining. Its approximate implementation supports the accuracy of the MSB region while reducing low LSB prefix nodes and almost truncating carry propagation in the lowest levels. AxPM-SK reduces switching activity and variable power consumption while may enhance delay performance by relaxing carry computation in high fan-out areas. Still, because of the intrinsic fan-out features of the topology, careful approximation management is required to prevent excessive error propagation.

3.4.5 Han-Carlson Multiplier:

Simple characteristics are introduced into the hybrid Han-Carlson adder design by the approximation Han-Carlson multiplier (AxPM-HC). Using a combination of the Brent-Kung and Kogge-Stone topology properties, the Han-Carlson topology allows adjustable wire complexity and logarithmic delay. The LSB prefix levels are reduced in its approximation form, but higher-order bits remain perfect computation. This hybrid approximation method provides balanced delay and scaling for large operand widths while decreasing the number of prefix nodes, logic gate consumption, and switching power. This results in AxPM-HC a successful trade-off between error resilience, hardware overhead, and speed.

4. IMPLEMENTATION DETAILS

4.1 DESIGN FLOW

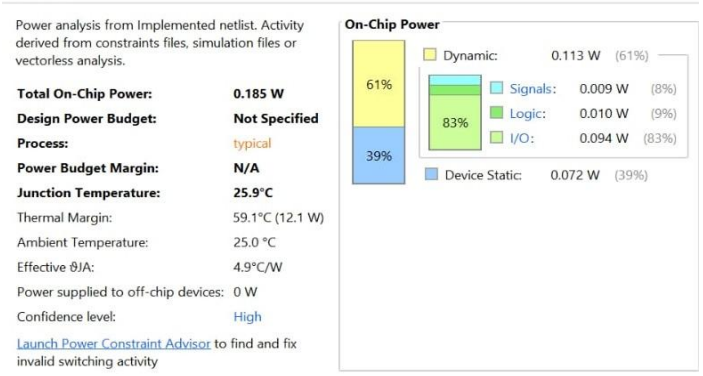
An organized RTL-to-hardware design technique is applied in the design of approximation parallel prefix multipliers depending on the Brent–Kung (BK), Kogge–Stone (KS), Ladner–Fischer (LF), Sklansky (SK), and Han–Carlson (HC) designs. HDL modeling, simulation-based functional confirmation, and hardware implementation with FPGA and ASIC synthesis tools

compose the overall layout flow. The most significant bit (MSB) region of the partial product elimination or final carry-propagate adder (CPA) in all approximate structures is maintained exact to limit the maximum error magnitude while preventing excess error propagation, while managed improvements take place in the LSB region.

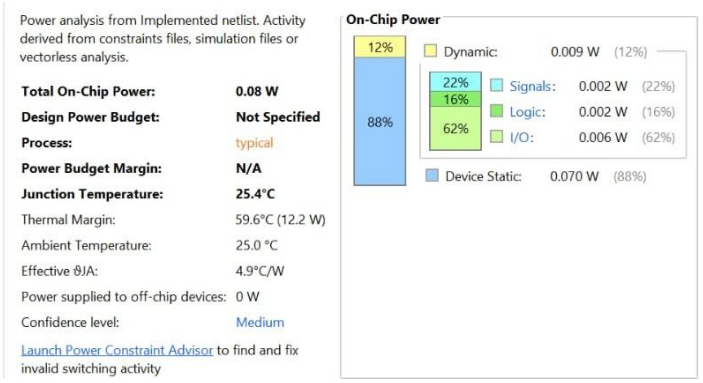
4.2 HDL MODELING (VERILOG/VHDL)

Verilog or VHDL can be used to conclude the suggested approximate multipliers at the Register Transfer Level (RTL). Based on the selected prefix topology, each structure is divided into three main blocks: partial product creation, partial product reduction (generally performed with carry-save adders), and a final carry-propagate adder. Changing approximation depths and operand lengths can be made available by parameterized coding techniques.

A modified version of the Brent-Kung adder structure is applied in the AxPM-BK architecture for implementing the last CPA level. Decreasing certain LSB prefix nodes or swapping simplified logic expressions for exact prefix operators gives approximation. Scalable RTL implementation with explained hardware overhead can be done practical by the Brent-Kung topology's natural limited fan-out and uniform connections patterns. Power and Area reports are provided in Fig.3 and Fig.4 respectively.



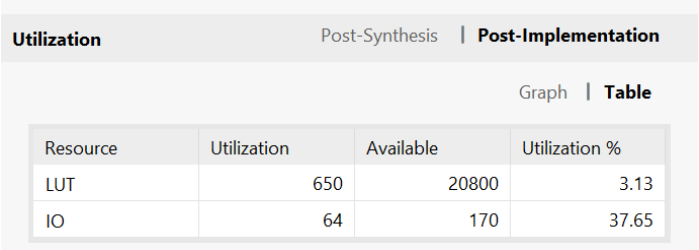
(a) Exact BK Multiplier



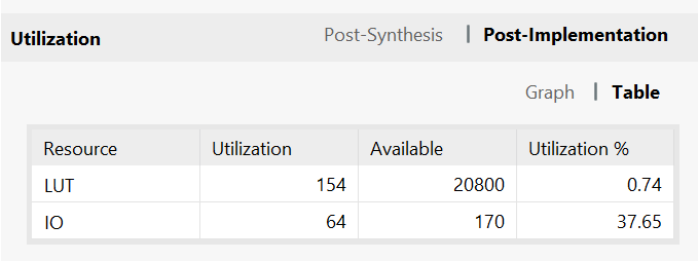
(b) AxPM-BK Multiplier

Fig.3. Power Comparison of BK Multipliers

A lowered form of the Kogge-Stone adder is requested in the closing step of the AxPM-KS multiplier. Approximation typically applies by removing or reducing certain black and gray prefix cells in the LSB region due to its fully parallel carry estimation and small logic depth. This provides exact computation in the essential MSB channel while minimizing switching activity and interconnect complexity. Power and Area reports are provided in Fig.5 and Fig.6 respectively.

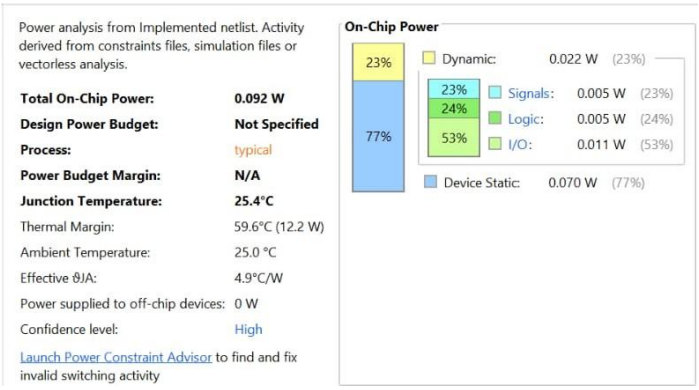


(a) Exact BK Multiplier

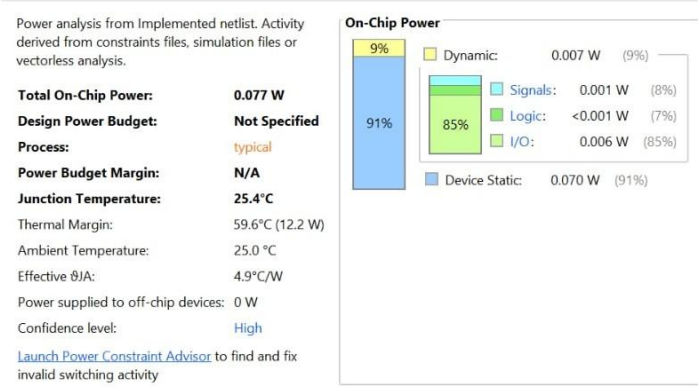


(b) AxPM-BK Multiplier

Fig.5. Area Comparison of BK Multipliers



(a) Exact KS Multiplier



(b) AxPM-KS Multiplier

Fig.6. Power Comparison of KS Multipliers

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	820	20800	3.94
IO	64	170	37.65

(a) Exact KS Multiplier

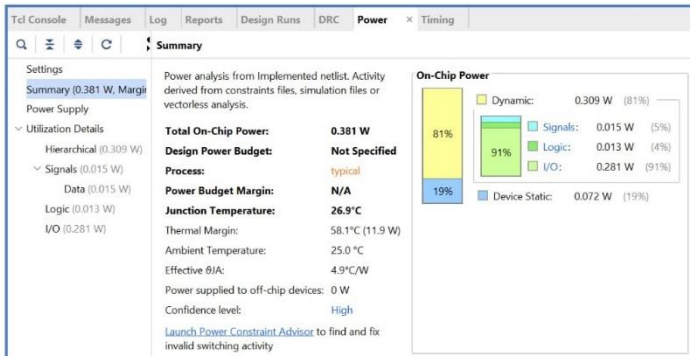
Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

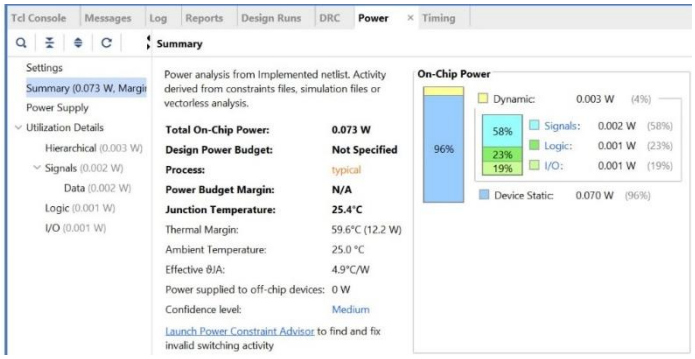
Resource	Utilization	Available	Utilization %
LUT	163	20800	0.78
IO	57	170	33.53

(b) AxPM-KS Multiplier

Fig.7. Area Comparison of KS Multipliers



(a) Exact LF Multiplier



(b) AxPM-LF Multiplier

Fig.8. Power Comparison of LF Multipliers

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	293	20800	1.41
IO	64	170	37.65

(a) Exact LF Multiplier

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

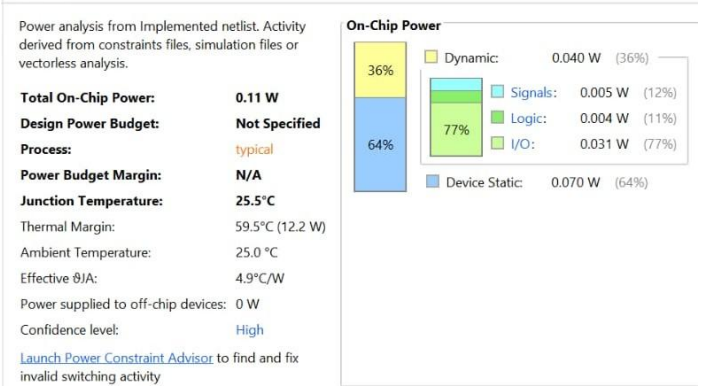
Resource	Utilization	Available	Utilization %
LUT	283	20800	1.36
IO	64	170	37.65

(b) AxPM-LF Multiplier

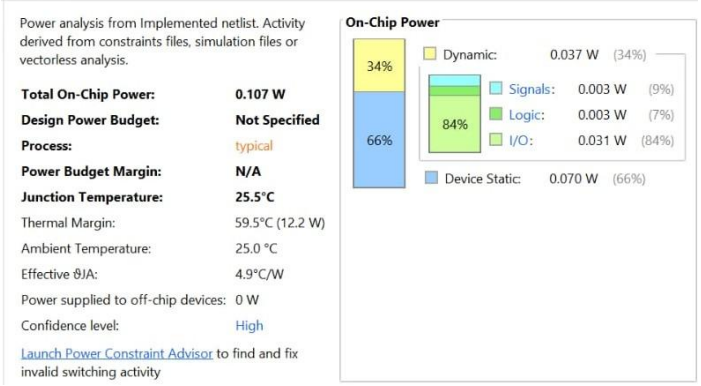
Fig.9. Area Comparison of LF Multipliers

The Ladner-Fischer adder topology is the source of the carrypropagate step in the AxPM-LF design. Its balancing prefix structure implies the introduction of approximation at lower prefix levels without materially changing global timing properties. As a result, logic depth and area are slightly lowered but delay performance stays uniform. Power and Area reports are provided in Fig.7 and Fig.8 respectively.

A reworked Sklansky adder architecture provides the basis for the AxPM-SK architecture. In order to minimize capacitive loading and dynamic power, approximation is carefully applied to LSB prefix nodes because the Sklansky topology shows noticeable fanout in the initial phases. To minimize worst-case error, careful RTL partitioning ensures that higher-order bits stay exact. Power and Area reports are provided in Fig.10 and Fig.11 respectively.



(a) Exact SK Multiplier



(b) AxPM-SK Multiplier

Fig.10. Power Comparison of SK Multiplier

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	274	20800	1.32
IO	64	170	37.65

(a) Exact SK Multiplier

Utilization Post-Synthesis | **Post-Implementation**

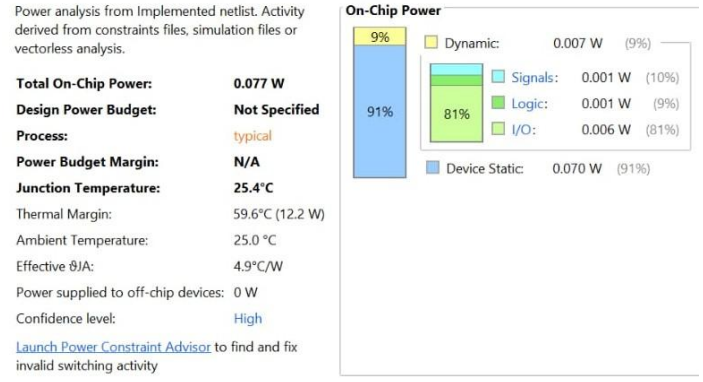
Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	181	20800	0.87
IO	59	170	34.71

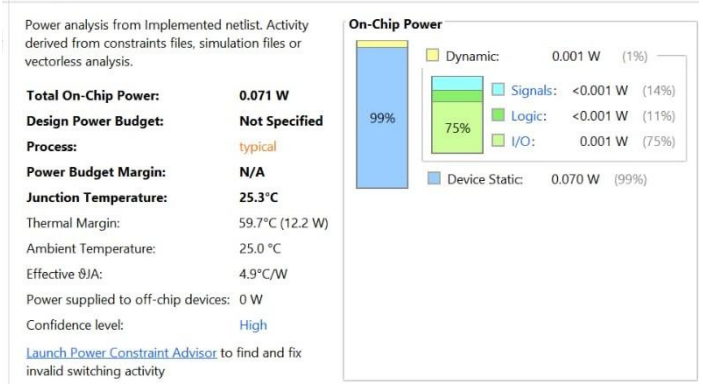
(b) AxPM-SK Multiplier

Fig.11. Area Comparison of SK Multipliers

In the final, a certain hybrid Han-Carlson adder circuit is worked by the AxPM-HC multiplier. While the Kogge–Stone-like the highest levels of the prefix network continue to remain as accurate, the Brent–Kung-like beginning layers generally include approximation. Expandable hardware implementation along with even timing performance are made practicable by this hybrid method. Power and Area reports are provided in Fig.12 and Fig.13 respectively.



(a) Exact HC Multiplier



(b) AxPM-HC Multiplier

Fig.12. Power Comparison of HC Multipliers

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	311	20800	1.50
IO	64	170	37.65

(a) Exact HC Multiplier

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	276	20800	1.33
IO	64	170	37.65

(b) AxPM-HC Multiplier

Fig.13. Area Comparison of HC Multipliers

5. EVALUATION METRICS

Area, power consumption, propagation delay, and Power–Delay Product (PDP) are essential hardware metrics used for evaluating the performance of the provided approximate BK, KS, LF, SK, and HC multipliers. In order to ensure fair comparison, all architectures have been studied under the same synthesis and operating parameters, which are provided in Table.1, Table.2 and Table.3.

5.1 AREA

For FPGA implementation, area is shown in terms of LUT count; for ASIC execution, area is given in terms of gate count. Because of its deep and totally parallel prefix layout, the KS-based multiplier normally takes up the most area. Specifically, the Kogge-Stone approximate multiplier gains the highest area reduction of nearly 80%, Brent-Kung multiplier with 76% reduction. The Sklansky architecture displays a moderate improvement of around 34%, while the Ladner-Fischer multiplier gives a minimal reduction of 3%. Han-Carlson multiplier achieves an increase in area of approximately 29%, indicating a trade-off between accuracy and hardware efficiency.

5.2 POWER CONSUMPTION

Dynamic and static elements are also included in power analysis. All five designs have lower dynamic power as the result of approximation in the LSB area, which decreases switching activity and logic complication. Among the calculated designs, the Ladner-Fischer approximate multiplier gives the highest power reduction of approximately 81%, followed by the Brent-Kung multiplier with 57% reduction. The Kogge-Stone multiplier shows a moderate improvement of 16%, while the HanCarlson multiplier gives a smaller reduction of nearly 7%. The Sklansky multiplier gives minimal improvement, with only 2–3% reduction in power consumption.

5.3 PROPAGATION DELAY

Timing analysis following synthesized is used to determine the delay. The Brent-Kung approximate multiplier gives the highest delay reduction of approximately 62%, making it the fastest architecture. The Kogge-Stone multiplier provides a moderate improvement of 23%, while the Sklansky and Ladner-Fischer multipliers show minimal delay reductions of nearly 3–4% and 2–3% respectvely. The Han-Carlson multiplier gives an increase in delay of 7%, indicating a trade-off between performance and design complexity.

5.4 THE POWER-DELAY PRODUCT (PDP)

It evaluates the effectively energy is utilized overall. The Ladner-Fischer approximate multiplier gives the highest PDP reduction of nearly 81%, indicating energy efficiency. The Brent-Kung multiplier also gives substantial improvement with 57% redution, while the Kogge-Stone multiplier provides a moderate reduction of approximately 35%. The Sklansky multiplier gives minimal improvement of 3%, and the Han-Carlson multiplier gives a negligible reduction of 1%.

5.5 AREA REDUCTION

Because partial product logic and least significant bit (LSB) prefix nodes are reduced; approximate architectures show large area savings. There are more logic gates and connections when carry accuracy is relaxed in less important areas. Because of their small prefix node count, BK and LF layouts usually exhibit the largest relative area reductions among the architectures, but KS displays a relatively smaller but still notable decrease because of its inherently dense topology.

5.6 POWER SAVINGS

Because of decreased switching activity and optimized carry propagation in the LSB region, all approximation multipliers obtain an apparent dynamic power savings. Reduced switching in lower prefix levels is an important benefit for architectures with large connection numbers, like KS. When the approximation can be restricted to the initial prefix steps, SK and HC show effective changing reduction, whereas BK and LF continuously save power because of regulated fan-out.

5.7 DELAY IMPROVEMENT

Because of lower phases that have shorter carry channels and less logic depth, propagation delay gives somewhat in approximation systems. While HC and LF make challenging time improvements, KS remains the lower absolute delay. Balancing delay enhancement has been shown by BK, and efficient fan-out tuning during synthesis is maintained for SK efficiency.

Table.1. Performance Comparison of Exact Multipliers

Multiplier Type	Area (LUTs)	Power (W)	Delay (ns)	PDP (nJ)
Brent Kung	650	0.185	35.682	2.511
Kogge Stone	820	0.092	22.937	2.110
Sklansky	274	0.110	17.550	1.931
Ladner Fischer	293	0.381	19.149	7.295
Han Carlson	213	0.077	17.648	1.360

Table.2. Performance Comparison of Approximate Multipliers

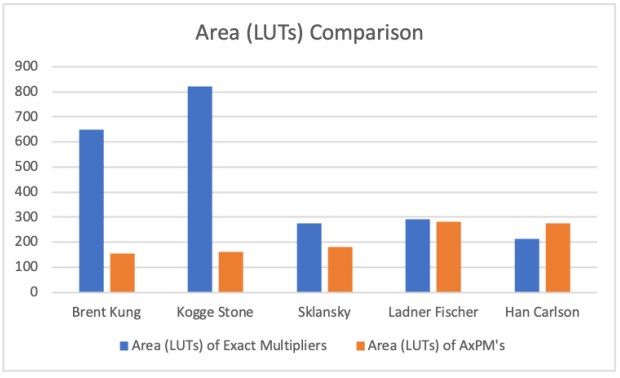
Multiplier Type	Area (LUTs)	Power (W)	Delay (ns)	PDP (nJ)
Brent Kung	155	0.080	13.575	1.086
Kogge Stone	163	0.077	17.669	1.381
Sklansky	181	0.107	16.891	1.867
Ladner Fischer	283	0.073	18.647	1.361
Han Carlson	276	0.071	18.972	1.350

Table.3. Improvement Percentage Comparison Table

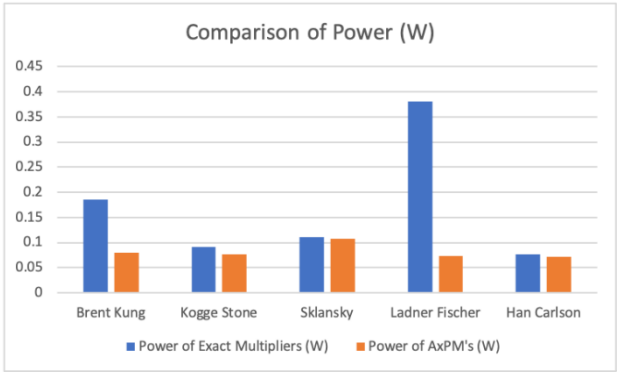
Multiplier Type	Area (%)	Power (%)	Delay (%)	PDP (%)
Brent Kung	76.15 ↓	56.76 ↓	61.94 ↓	56.75 ↓
Kogge Stone	80.12 ↓	16.30 ↓	22.97 ↓	34.55 ↓
Sklansky	33.94 ↓	2.73 ↓	3.75 ↓	3.31 ↓
Ladner Fischer	3.41 ↓	80.84 ↓	2.62 ↓	81.36 ↓
Han Carlson	29.58 ↑	7.79 ↓	7.50 ↑	0.74 ↓

6. RESULTS AND COMPARISON

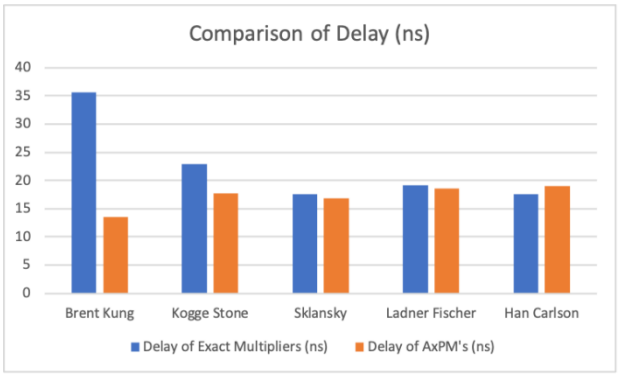
This section provides an identical synthesis and process conditions, the pro- posed approximate parallel prefix multipliers (BK, KS, LF, SK, and HC) have been evaluated to their correct alternates.



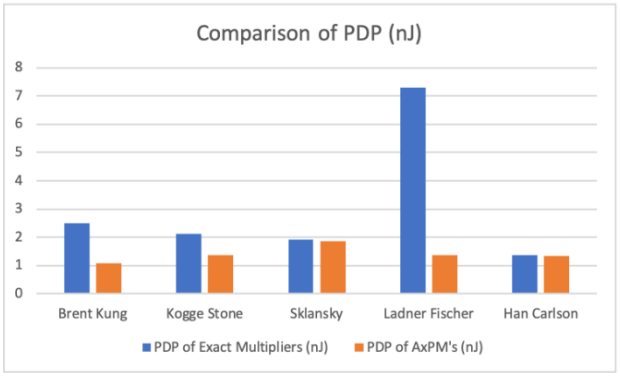
(a) Area



(b) Power



(c) Delay



(d) PDP

Fig.14. Performance Comparison of Multipliers

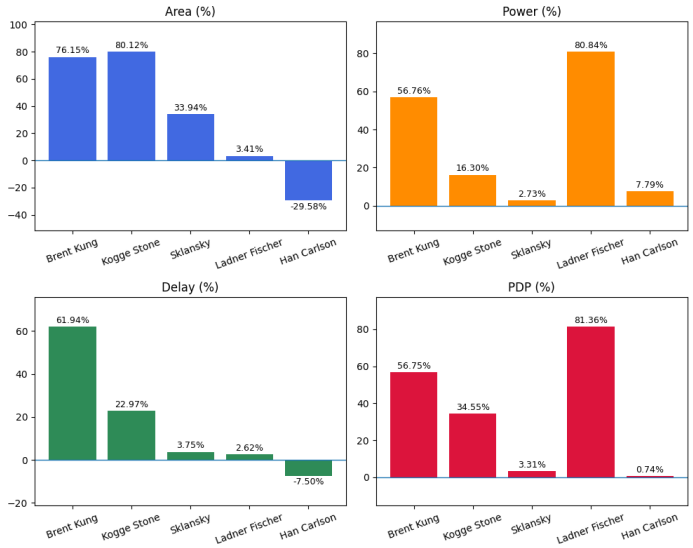


Fig.15. Improvement Metrics of Multipliers(a) Area (b) Power (c) Delay (d) PDP

Propagation delay, power consumption, and area are the main evaluating points, which are provided in Fig.14 and Fig.15. In terms of greatest Area, Delay, and Power-Delay Product (PDP), the Approximate Brent Kung multiplier works best throughout, 76% reduction in area, 57% reduction in power, 62% reduction in delay, 57% reduction in PDP. These results suggest that approximate multipliers surpass exact versions in terms of hardware performance and energy consumption, which are shown in Fig.15.

7. CONCLUSION

The construction and assessment of approximate parallel prefix multipliers based on the Brent–Kung (BK), Kogge–Stone (KS), Ladner–Fischer (LF), Sklansky (SK), and Han–Carlson (HC) architectures were described in this study. To limit computational error, a controlled approximation was implemented in the prefix network's least significant bit (LSB) zone while maintaining exact calculation in the most significant bit (MSB) region. In analysis to their accurate counterparts, post-synthesis studies displayed average development in propagation delay and constant reductions in area and power consumption over all approximate designs. Because of its low logic depth, the KS-based multiplier outperformed the other designs in terms of propagation latency, making it appropriate for high-speed applications. Favourable area efficiency and balanced hardware utilization were offered by the BK and LF architectures. There was less rationale in the SK structure. The approximation multipliers proved an effective trade-off between computational accuracy and energy savings from the viewpoint of energy efficiency. In order to give a main impact and area savings while maintaining tight error metrics, structures like HC and LF typically operate near to the Pareto-optimal area. Application-specific specifications, such as acceptable error tolerance, speed constraints, and hardware limitations, reply to which architecture is best.

Future research might be focused on dynamically changing approximation approaches that change accuracy levels in response to the changes in workload needs. In order to improve energy performance-accuracy trade-offs in approximate arithmetic construction, more research has to be into the hybrid reduction trees, technology-node scaling effects, and integration into application specific accelerators such as digital signal processing and machine learning systems may be best.

REFERENCES

[1] P.M. Kogge and H.S. Stone, "A Parallel Algorithm for the Efficient Solution of a General Class of Recurrence Equations", *IEEE Transactions on Computers*, Vol. 22, No. 8, pp. 786-793, 1973.

[2] R.P. Brent and H.T. Kung, "A Regular Layout for Parallel Adders", *IEEE Transactions on Computers*, Vol. 31, No. 3, pp. 260-264, 1982.

[3] J. Sklansky, "Conditional-Sum Addition Logic", *IRE Transactions on Electronic Computers*, Vol. 9, No. 2, pp. 226-231, 1960.

[4] R.E. Ladner and M.J. Fischer, "Parallel Prefix Computation", *Journal of the ACM*, Vol. 27, No. 4, pp. 831-838, 1980.

[5] R. Zimmermann, "Binary Adder Architectures for Cell-based VLSI and their Synthesis", *IEEE Transactions on Computers*, Vol. 45, No. 3, pp. 294-303, 1996.

[6] V. Gupta, D. Mohapatra, A. Raghunathan and K. Roy, "Low-Power Digital Signal Processing using Approximate Adders", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, Vol. 32, No. 1, pp. 124-137, 2013.

[7] M. Shafique, W. Ahmad, R. Hafiz and J. Henkel, "A Low Latency Generic Accuracy Configurable Adder", *Proceedings of International Conference on Design Automation*, pp. 1-6, 2015.

[8] J. Miao, K. He, A. Gerstlauer and M. Orshansky, "Modeling and Synthesis of Quality-Energy Optimal Approximate Adders", *Proceedings of International Conference on Computer-Aided Design*, pp. 728-735, 2012.

[9] S. Mittal, "A Survey of Techniques for Approximate Computing", *ACM Computing Surveys*, Vol. 48, No. 4, pp. 1-33, 2016.

[10] M. Ahmad and M. Shafique, "Accuracy Configurable Approximate Multipliers for Energy-Efficient Computing", *Proceedings of International Conference on Design, Automation and Test in Europe*, pp. 129-134, 2018.