Proceedings of the 6th International Conference on Trends in Material Science and Inventive Materials (ICTMIM-2026)
IEEE Xplore Part Number: CFP260J1-ART; ISBN: 979-8-3315-7006-4

Fast and Compact Approximate Recursive Multiplier using a Modified Carry Bypass 4:2 Compressor

[1*]Selvarasan R
Research Scholar
Department of ECE
Bharath Institute of Science and Technology
Bharath Institute of Higher Education and Research
Chennai-600073, India.
Assistant Professor
Department of ECE
Kuppam Engineering College
Kuppam-517425, India.
arasanece@gmail.com

[2]Sudhagar G
Associate Professor
Department of ECE
Bharath Institute of Science and Technology
Bharath Institute of Higher Education and Research
Chennai-60073, India.
sudhagar.ece@bharathuniv.ac.in

[3]Rasadurai K
Professor
Department of ECE
Kuppam Engineering College
Kuppam, Andhra Pradesh-517425
India.
krasadurai@gmail.com

2026 6th International Conference on Trends in Material Science and Inventive Materials (ICTMIM) | 979-8-3315-7006-4/26/$31.00 ©2026 IEEE | DOI: 10.1109/ICTMIM68190.2026.11507316

***Abstract*— Recursive multipliers are widely used in real-time image enhancement, biomedical and low-power edge AI-based applications, since recursive method will help to reuse small multiplier blocks and to construct a larger unit, and provide a significant reduction in logic size, critical path delay and power consumption, compared to the traditional tree and array multipliers. This recursive technique works by partitioning the operands into smaller segments and recombining the partial results, allowing faster computation and reduced logic activity. However, many approximate multipliers still suffer from heavy logic depth, high switching and inconsistent error behavior. In contrast, the recursive structure offers better scalability and predictable dataflow, but the existing recursive designs build three techniques of bitwise, recursive with PBO-5 and hybrid approximate method-based modules, which still face problems such as unbalanced error distribution, which increases uncertainty output for small operands with redundant carry propagation. In this work, the main challenge is to reduce the logic size and delay of the hybrid recursive method without modifying the internal structure; thus, the proposed architecture introduces a modified carry bypass 4:2 compressors and integrates it into the recursive framework to create a fast and compact approximate multiplier. The complete design is implemented in Verilog HDL, and the functional verification and error analysis are done in ModelSim, and this work is synthesized in Xilinx Vertex-5 XC5VLX50-2FF324 FPGA.**

Keywords — approximate multiplier, carry bypass 4:2 compressors, recursive multiplier, hybrid approximate technique.

I. INTRODUCTION

Modern digital systems increasingly rely on the high-speed performance and energy consumption of arithmetic operations, especially multiplications, which are at the core of real-time image processing, biomedical signal processing, embedded intelligence and neural network acceleration with edge computing applications [1], [2]. These systems will frequently operate on larger datasets at high throughput, with the support of efficient multiplications standing as a key determinant of overall system performance [1]. Although, the recursive multipliers have gained particular attention in this domain because they offer a structurally compact solution and break down the larger multiplication into smaller ones with the support of reusable blocks [3]. This partitioning greatly reduces the routing complexity, lowers power consumption, and enables better resource utilization than the fully parallel array-based architecture [4].

Furthermore, the error-tolerant applications, such as visual computing, pattern recognition, and approximate interference, which allow small computational errors without noticeable degradation in output quality, and they also make an approximate recursive multiplier an attractive candidate for low-power VLSI and FPGA design [1], [2], [5]. However, these promising benefits coexist with several challenges in conventional and approximate multiplier structure, and it motivates deeper investigation that can deliver a balanced tradeoff between speed, area, accuracy and hardware complexity [6], [7], [8].Although their application, many existing approximate multipliers suffer from fundamental drawbacks; for example, the conventional array multiplier, which is an accurate one, but it produces more latency to produce output and consumes high area due to long carry propagation and dense partial Product accumulation [4].

Further, the basic operation of the recursive multiplier reduces the complexity, but it has difficulty in logic size and power consumption [3]. Additionally, the hybrid design that combines with a partial truncation and bitwise OR operation consumes less power but often misleads accuracy, and it generates unstable error characteristics, especially when small operands are multiplied [9]. Even the recent approximate compressors, such as OR-based, majority or simplified 4:2 compressor structures, show inconsistent behavior, and they produce non-zero output, and some output

produces unpredictable error accumulation [5], [6], [10]. These limitations create a gap where existing designs fail to simultaneously achieve fast computation, compact structure, stable error analysis, and hardware efficiency in recursive multipliers [1], [7], [8], [11].

In the context of recursive multipliers, the problem becomes more pronounced because the structure of identical partial products is repeatedly used in all the blocks; any inefficiency, redundant logic or error imbalance inside the compression path is magnified across the recursive stage [3]. When conventional compressors or adders are used within the recursive framework, they introduce long carry chains to produce the final output [12]. Similarly, the hybrid approximation multiplier will have used partial bits in OR logic; often it reduces a logic count but brings poor precision for higher-order bits and creates inconsistent carry cancellation [9]. Thereafter, most of the earlier recursive architecture that integrated approximate compressors still suffered from carry propagation overhead and produced non-uniform approximation behavior and also increased error rates from specific operand patterns [6], [7], [8], [10]. Therefore, the main problem addressed in this work is the lack of a recursive multiplier architecture that can provide fast operations, minimized logic depth, and predictable approximation behavior [1], [6].

To address these problems, this work proposes a fast and compact approximate recursive multiplier that integrates with modified carry bypass 4:2 compressors within the 8x4 recursive partitioning framework [5], [6], [10]. This novel structure significantly reduces the critical path delay and internal logic transitions and mitigates unnecessary carry propagation, and it's inherently better suited for recursive applications [1], [7]. This proposed architecture is implemented using Verilog HDL, and functional verification is performed with Modelism to verify timing correctness and error analysis of MED, NMED and the design synthesized on a Xilinx Virtex-5 xcvlx50-2ff324 FPGA to evaluate practical hardware performance in terms of area, delay, and resource utilization [3], [4]. In the remaining part of this paper, Section II, presents related work of existing methods, parallel partial product addition-based multipliers, and the hybrid recursive 8-bit multiplier employing two approximate 8x4 multipliers [3], [5], [10]. Section III describes the proposed carry bypass compressor design description and the hybrid multiplier design using the proposed compressor [6], [10]. The results and evaluation are described in section IV, and the conclusion of this work is given in section V.

II. Related work

Multiplication in digital systems is traditionally performed with the generation of a number of partial products, and addition helps to reduce the number of partial products, where 64 partial products are generated for an unsigned 8x8 multiplication [1], [4]. The traditional array multiplier sums all these partial products through a fixed arrangement of adders, but this method suffers from a long carry chain, high area, and high critical path [1], [4]. To overcome these disadvantages, the parallel partial product (PPA) structure was introduced [4]. Fig. 1 shows the reference multiplication of this PPA structure, which organizes partial products into six structures of adder stages, which are named Adder-1 to Adder-6. These adders are designed for the specific purpose of reducing multiplier rows of partial products in parallel while balancing the carry propagation delay [6]. Each adder handles the specific section of the partial product matrix. Adder-1 deals with the lowest weight, and Adder-2 and Adder-3 handle mid-weight columns. The same process will be copied in the next addition of Adder 4 to Adder 6, as shown in Fig. 1 [4]. Whereas, this PPA structure will not reduce the maximum logic size, and it will take more carry chain operation in the internal adder structure and also external adder interconnection [4]. Although PPA improves delay, the size of the adder tree and the large number of adders still contribute to hardware cost [4], [6]. To overcome this issue, the recursive multiplier breaks the 8x8 multiplication into smaller 8x4 or 4x4 blocks [3].

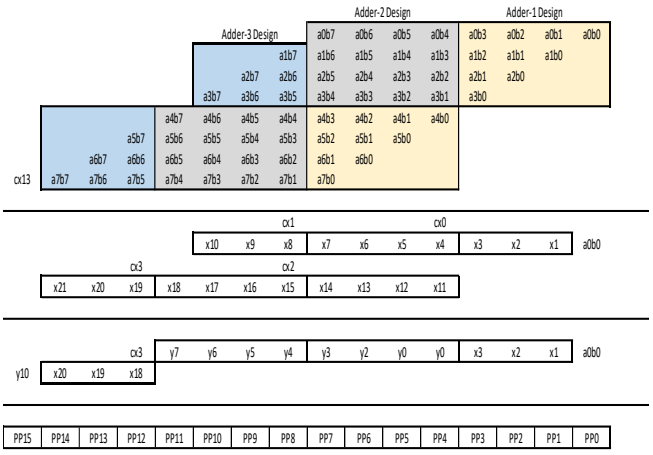

Fig.1. Conventional 8x8 Unsigned Multiplier using Parallel Partial product addition with different Adder design

Fig. 2 shows the two independent 8x4 multipliers each generate their own partial product columns. Since, this 8x4 multiplication operates completely in parallel, and it helps the recursive method to reduce computational depth and area [3], [10]. Further, this independent 8x4 structure is also designed with small-sized PPA-based adders in the method of a hybrid recursive approach, which eliminates the large PPA structure of conventional design and significantly reduces the number of required full adders and half adders [5],[10].

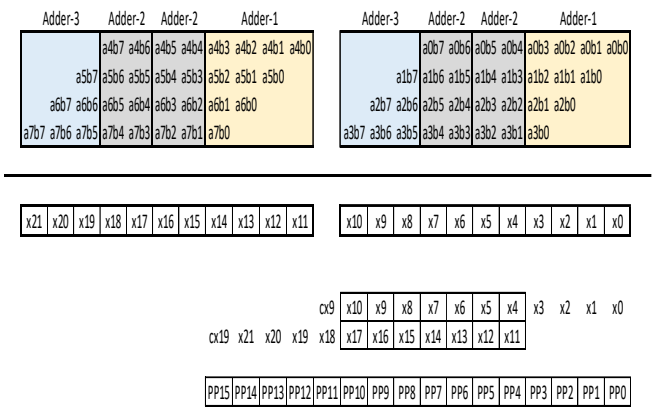

Fig. 2. Recursive method based 8x4 to 8x8 Unsigned Multiplier using different Adder design

Similarly, the mathematical procedure of this Fig. 2 recursive multiplication with two independent structures is configured as (1) [3].

$$A \times B = A_H \times B_H + A_H \times B_L + A_L \times B_H + A_L \times B_L \quad (1)$$
$$A \times B = A \times B_H + A \times B_L$$

However, even the hybrid recursive multiplier faces certain limitations; the major one is logic size contributed by using full adders, half adders and buffer registers in each stage of the architecture [3]. When two approximate 8x4 multiplications are merged, the resulting structure still contains a three-stage accumulation pipeline that depends on the ripple carry addition inside each stage [3], [10]. This carry chain limits the reduction in critical path delay; due to the number of full adders and half adders in the multiplications, it will reduce only two inputs or three inputs from the partial products. This creates multiple sequential stages, and its increasing power consumption and logic size [5]. Hence, a 4:2 compressor-based architectures are priority one to reduce four input bits plus a carry-in into two outputs of sum and carry in just one logical stage [2]. This 4:2 compressor reduces some delay compared to the traditional full adder and half adder design [6], [12]. At the same time, these 4:2 compressor operations will occupy more carry chain operations, and even approximate compressors also take more carry logic [6], [10]. The hybrid technique of existing 4:2 approximate two-compressor designs, with the generation of sum and carry operations, will be described in equations (2) [6], [7], [8].

$$Sum1 = a \oplus b \oplus c \oplus d$$
$$Carry1 = ab + ac + ad + bc + bd + cd$$
$$Sum2 = a + b + c + d$$
$$Carry2 = abc + acd + bcd \quad (2)$$

As per these compressor equations (2), the compressor will occupy more logic size in carry operations, while using approximate methods, it will consume a number of AND & OR gates, and this compressor cell significantly adds the delay and power in recursive multipliers, and it has more bottleneck errors [6], [10]. Thus, the proposed technique of this work comes up with a solution using the carry bypass method; it directly forwards a selected input as the carry and produces the sum through XOR gates. By removing the internal carry computation entirely, the compressor achieves near-zero carry propagation delay, lower logic depth, fewer gates and also reduced dynamic power [6], [7], [8]. This makes the carry bypass compressor far more suitable for recursive multipliers, where a fast and lightweight compressor directly improves overall performance, timing and hardware efficiency in recursive multipliers, while maintaining acceptable approximation accuracy [1], [5], [6].

III. Proposed Hybrid Recursive Multiplier using Carry Bypass Method

In the proposed hybrid recursive multiplier, the carry bypass technique plays a key role in reducing the delay and hardware overhead that normally arise from serial carry propagation in the partial product reduction stage [3], [10]. In the conventional recursive 8x8 architecture, each 8x4 multiplier generates a column-wise sum, and these sums depend heavily on ripple-style carries; it will create growth with bit position, leading to a long critical path and additional buffer as the bit width increases [3], [4]. By contrast, our design bypasses these internal carries by directly forwarding the lowest bit, denoted as d, to the compressor carry output as shown in Fig. 3, while the sum is generated with an accurate one with the help of an XOR gate, and only the carry produces an approximation [6], [7], [8].

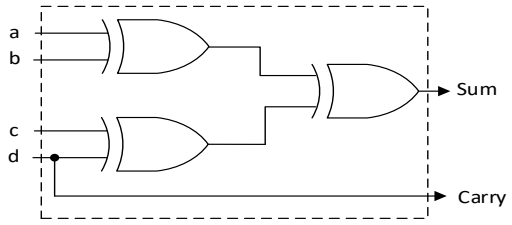

Fig.3. Proposed carry bypass 4:2 compressor design

Mathematically, the bypassed compressor 4:2 is expressed as (3), it effectively breaks the carry dependency and allows several it columns to operate in parallel [6], [7].

$$Sum = (a \oplus b) \oplus (c \oplus d)$$
$$Carry = d \quad (3)$$

This block becomes important in the recursive structure; it feeds the output directly into the next reduction stage, where intermediate sums are merged with the outputs of the two 8x4 multipliers [3], [10]. The detailed architecture of the proposed multiplier is shown in Fig. 4. As per the results, the next stage receives the values that are already independent of carry chains, and it enables faster accumulation with fewer adders and reduced gate depth [6], [8]. The main advantage of this approach is the substantial reduction in area, power and critical path delay, as also observed in carry disregard multipliers, where ignoring the carry significantly improves hardware efficiency [1], [5], [6].

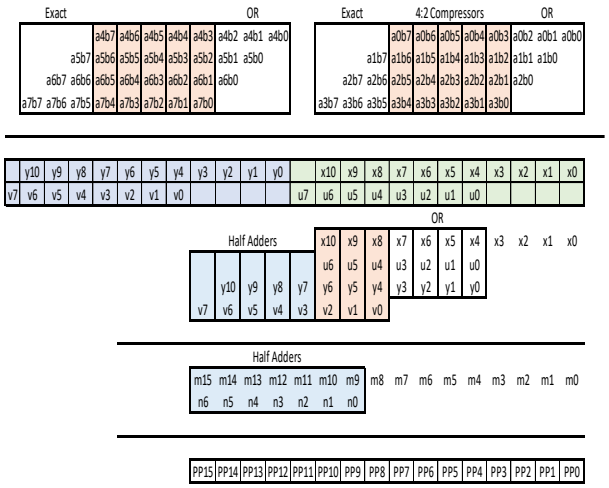

Fig. 4. Hybrid recursive approximate multiplier using carry bypass 4:2 compressor.

The three stage operation of this Fig. 4 architecture of the proposed hybrid recursive approximate multiplier, where carry bypass 4:2 compressors are integrated into selected bit columns, which using 12 Colum, and 12 half adders will reduce hardware complexity while preserving accuracy in higher weight positions [6], [10]. However, the limitation of direct carry removal in compressor is increases the potential rise in numerical error reduction [6], [8]. Similarly, higher significance columns of existing hybrid multiplier design, which uses Compressor 1 and Compressor 2 from equation

(2) [6], [7]. This work mitigates this by combining the OR-gated-based sum with a deliberate carry bypass compressor only in the selected region of the recursive tree, ensuring that important upper bits retain accuracy, while the lower and middle level bits have benefit from aggressive approximation [5], [10]. Finally, the novelty of this technique is integrating with a lightweight carry bypass compressor, which is used in an 8x4 sub-recursive multiplier, and finally merging stages simultaneously benefit from reduced carry pressure [3], [10]. It achieves faster and more compact hardware without the high error usually associated with carry disregard methods [1], [5], [6].

IV. Result and Comparisons

Approximate multiplier is intentionally designed to trade with an arithmetic accuracy for improvement in hardware efficiency, such as reduced area, lower power consumption, shorter delay and improved energy performance product [1], [4], [5]. From using this benefits, the approximate multipliers, which highly suitable for error resilient applications, the deviation from the ideal arithmetic results, which introduced numerical errors that must be carefully quantified [1], [2]. This error analysis is an essential component of the design, validation and compression of the approximate multiplier. It allows measuring; how the approximate outputs deviate from exact outputs and its helps determine whether the design is usable for different application domains such as image processing, filtering, and the neural network inference [1], [2]. As highlighted in recent literature, the error analysis metrics, such as error rate, mean error and relative error distance, form the backbone of the quantitative evaluation framework, which is used to ensure that approximate architectures maintain acceptable levels of computational correctness [2], [7], [8]. In this proposed hybrid recursive multiplier using the carry bypass mechanism, with the internal carry chain, are deliberately removed in selected columns to reduce delay and area [6], [7], [10]. Although this bypassing achieves significant hardware advantages, it can introduce numerical error during partial product accumulation [6], [8].

The formal error analysis of this design is essential for determining if the reduction in hardware cost is worth the resulting deviation in numerical results [1], [2]. The analysis of the error assessment framework is widely adopted in recent approximate research; the error analysis of MED (Mean Error Distance) is presented in equation (4) [2], [8].

$$MED = \frac{1}{N}\sum_{i=1}^{N} | ExactOutput_i - ApproxOutput_i | \quad (4)$$

Moreover, rendering the scale-independent statistic method to analyze dynamic range in NMED standardized output is especially beneficial to evaluating the design of different output ranges. The equation of NMED, which is present in (5),

$$NMED = \frac{MED}{Maximumoutput - Minimumoutput} \quad (5)$$

Further analysis, the recursive multiplication design functional verification done in ModelSim, Fig. 5 shows the results and where the approximate output closely follows the exact multiplication result. This design was done with Verilog HDL and synthesized using Xilinx-Vertex-5 FPGA, and the proposed design was compared with the existing exact method, the recursive 8x8 – 8x4 multiplier, the hybrid 8x8 multiplier using PBO-5 architecture, and the recursive 8x8 – 8x4 multiplier with two different compressor methods [3], [5], [10].

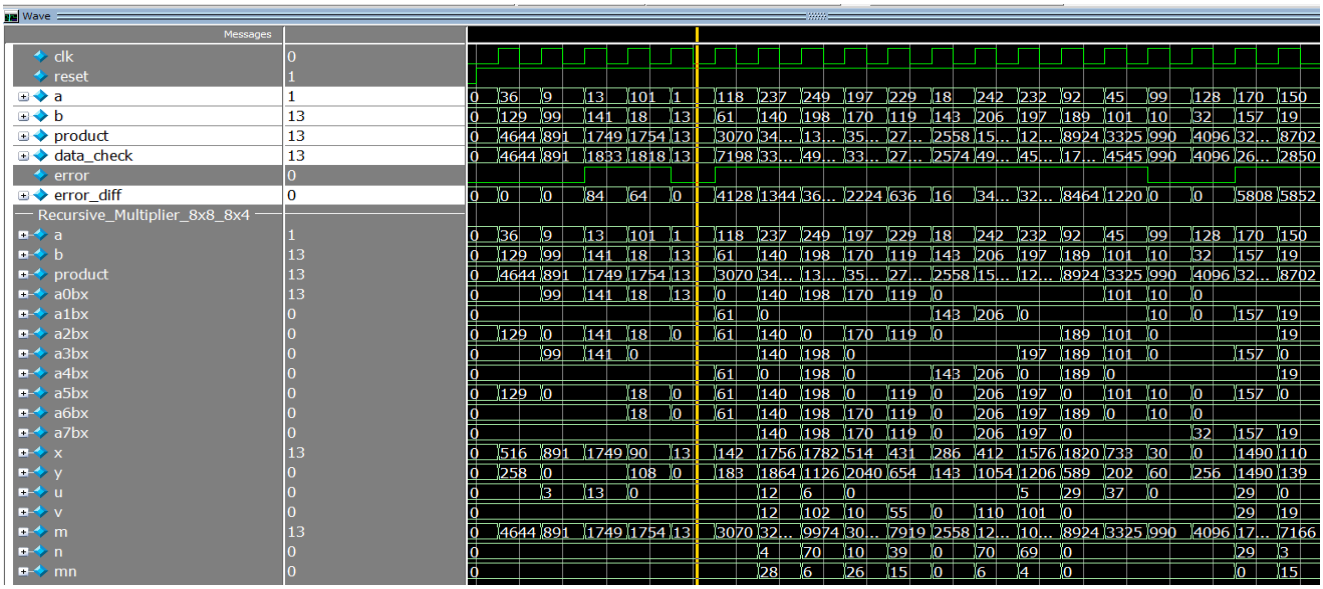

Fig. 5. Functional verification in ModelSim, Multiplication results present with accurate and approximate values.

Fig. 6 highlights that the proposed design summary has 78 LUTs and 33 registers, which represent a reduction of nearly 66% and 54% compared to the exact method [4]. Further, Fig. 7 illustrates the RTL schematic of the proposed work. The analysis of the complete design given in Table I show that the proposed method achieves the lowest hardware usage among all the compressor designs, and comparative analysis chart has shown in Fig. 8 [6], [5], [10], [11].

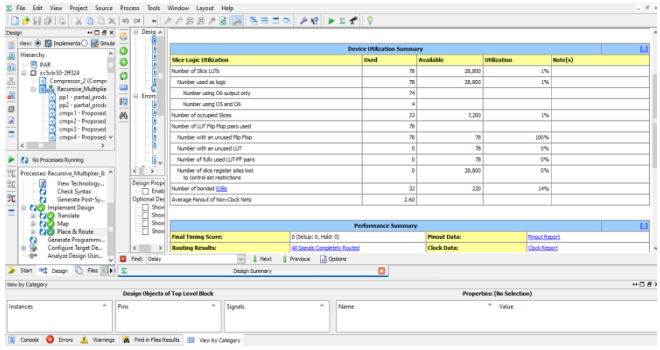

Fig. 6. Design summary analysis using Xilinx Vertex-5 FPGA

The proposed multiplier's error analysis is shown in Fig. 9. Table II compares the MED and NMED of three approximate compressor-based multipliers highlights its superior accuracy characteristics [2], [7], [8]. The proposed multiplier reduces MED by more than 39% and NMED by over 38% compared to the compressor 1 and compressor 2 designs, demonstrating that the carry bypass approach occupied only limited and controlled numerical error [6], [7], [8]. These combined results indicate that the proposed hybrid recursive design provides an excellent balance between computation accuracy and hardware efficiency, making it suitable for real-time and energy-constrained signal processing applications [1], [5], [2].

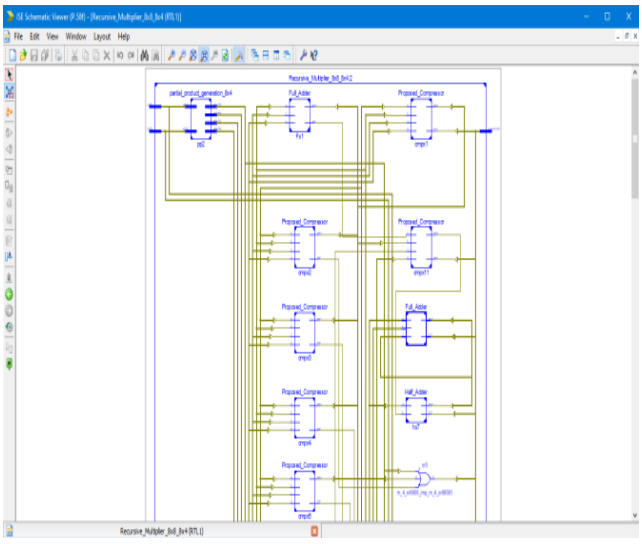

Fig. 7. RTL Schematic analysis of proposed hybrid recursive multiplier

TABLE I . COMPARISONS ANALYSIS OF RECURSIVE 8X8-8X4 MULTIPLIER DESIGN, SYNTHESIZE USING XILINX VERTEX-5 XC5VLX50-2FF324 FPGA

	LUT	FF	Delay(ns)
Exact	228	73	19.9
Recursive (bit-wise)	228	71	19.9
Recursive Hybrid PBO-5	186	76	15.9
Recursive compressor-1	118	40	11.7
Recursive compressor-2	118	42	11.6
Proposed Recursive Carry bypass compressor (This work)	78	33	11.6

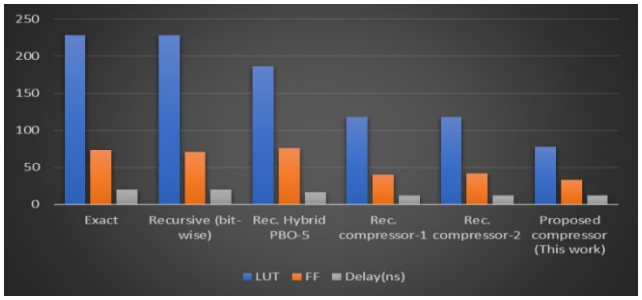

Fig. 8. Comparative analysis of LUT, Slice registers and delay performance using Recursive Multiplier

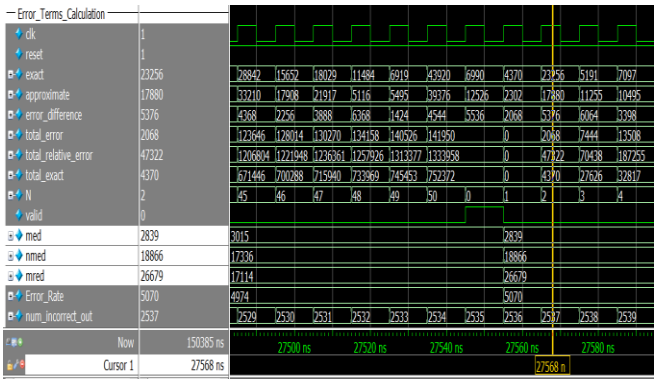

Fig. 9. Error analysis of hybrid recursive multiplication in ModelSim

TABLE II. ERROR ANALYSIS OF HYBRID RECURSIVE MULTIPLIER USING APPROXIMATE COMPRESSORS

	MED	NMED
Recursive compressor-1	5416	34976
Recursive compressor-2	4642	29980
Proposed Recursive Carry bypass compressor (This work)	2839	18866

V. CONCLUSION

This work explored a complete design analysis for improving the efficiency of approximate multiplication by systematically examining multiple architectures, which are compared with the traditional design of an 8x8 array multiplier with a recursive 8x4 partition-based design, and evaluating hybrid approximation using partial bitwise OR logic, and also integrating different approximate compressor structures into the recursive framework. Across these stages,

the key challenge remained the high logic depth, unbalanced error characteristics and unnecessary carry propagation found in many existing approximate multiplier designs. To address this issue, the final architecture introduced in this work employs modified carry bypass 4:2 compressors that replace full adders and existing compressors. Further, this design is significantly improving speed and logic compactness within the recursive path and making the recursive technique more effective by stabilizing error behavior and reducing switching activity. The proposed approach demonstrates a clear improvement in structural efficiency while maintaining acceptable approximation characteristics and this multiplier is suitable for energy-aware and real-time signal processing systems. Future enhancement of this work may include extending the method of larger operand sizes, integrating dynamic accuracy tuning, combining carry bypass logic with adaptive truncation and validating with ASIC technologies to further reduce power consumption and improve timing performance in advanced VLSI applications.

REFERENCES

[1] A. Kumari and R. P. Palathinkal, “Design and Analysis of Energy Efficient Approximate Multipliers for Image Processing and Deep Neural Network,” IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 72, no. 2, pp. 854–867, Feb. 2025.

[2] S. Lee, J. Kim, and Y. Kim, “Accuracy Performance Analysis of Quantized DNN Models using Approximate 4-2 Compressor Based Multipliers,” 2025 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), 2025.

[3] D. Karthikeya, P. R. Teja Sree, and G. P. Mishra, “8-Bit Approximate Multiplier using Approximate Full Adder,” 2025 Devices for Integrated Circuit (DevIC), Kalyani, India, 2025.

[4] S. T. Petla, T. Nichenametla, V. S. Chowdam, P. Polturu, and P. Ponkala, “Comparative Analysis of Energy-Efficient Approximate Multiplier and Wallace Tree Multiplier for Error-Tolerant Applications,” 2025 3rd International Conference on Device Intelligence, Computing and Communication Technologies (DICCT), 2025.

[5] L. S. Koneru, Y. Pallavi, and P. Kodali, “A Novel Design of Power-Efficient and Accurate Approximate Multiplier using Approximate Compressors and Approximate Adders,” 2025 International Conference on Computational Robotics, Testing and Engineering Evaluation (ICCRTEE), 2025.

[6] A. G. M. Strollo, E. Napoli, D. De Caro, N. Petra, and G. Di Meo, “Comparison and Extension of Approximate 4-2 Compressors for Low-Power Approximate Multipliers,” IEEE Transactions on Circuits and Systems I: Regular Papers, 2020.

[7] J. Jeon and Y. Kim, “High-Precision, Low-Delay Approximate Multiplier Using 4-2 Compressor and Error Recovery,” 2025 International Technical Conference on Circuits/Systems, Computers, and Communications (ITC-CSCC), 2025.

[8] J. Jeon and Y. Kim, “Improving Error Metrics in Approximate Multipliers with an Error Recovery Unit,” 2025 International Conference on Electronics, Information, and Communication (ICEIC), 2025.

[9] S. K. Velagaleti and S. Malraju, “Design and Implementation of LUT Based Approximate Multiplier for Energy-Efficient Computing,” 2025 3rd International Conference on Device Intelligence, Computing and Communication Technologies (DICCT), 2025.

[10] L. S. Koneru, S. Ramagiri, and P. Kodali, “Design and Implementation of Approximate Multiplier Using Approximate 4-2 Compressors and Approximate Half Adders,” 2025 1st International Conference on Secure IoT, Assured and Trusted Computing (SATC), 2025.

[11] B. Nelmin N., R. S. Shaji, H. S., S. Shettygari, V. J. K., and V. Vijayan, “Design of Approximate Multiplier using Highly Compressed 5:2 Counter,” 2025 6th International Conference on Mobile Computing and Sustainable Informatics (ICMCSI), 2025.

[12] P. Balasubramanian and N. E. Mastorakis, “High Speed Gate Level Synchronous Full Adder Designs,” WSEAS Transactions on Circuits and Systems, vol. 8, no. 2, pp. 290–300, Feb. 2009.