International Journal of Reconfigurable and Embedded Systems (IJRES)
Vol. 15, No. 2, July 2026, pp. 269~280
ISSN: 2089-4864, DOI: 10.11591/ijres.v15.i2.pp269-280

High-performance approximate MAC multiplier using majority logic compressors for CNNs

Selvarasan Radhakrishnan[1,2], Sudhagar Govindhaswamy[1], Rasadurai Kumaravel[2]
[1]Department of Electronics and Communication Engineering, Bharath Institute of Higher Education and Research, Chennai, India
[2]Department of Electronics and Communication Engineering, Kuppam Engineering College, Kuppam, India

Article Info

Article history:

Received Nov 28, 2025
Revised Mar 6, 2026
Accepted May 30, 2026

Keywords:

Approximate multiplier
Convolutional neural network
Dadda multiplier
Full adder
Multiple accumulate

ABSTRACT

This research presents an optimized multiple accumulate (MAC) unit multiplier design for efficient convolutional neural network (CNN) operations. This design mainly focuses on making the multiplier systems smaller by using approximate majority compressor methods instead of the usual and traditional approximate methods. The traditional approximate multiplier compressor techniques are leads to increases in logic size, critical path delay, and power consumption; however, the proposed research mitigates these problems and solves them with a novelty-based approach in the Dadda multiplier technique. The novelty of this approach is to reduce the number of stages in the multiplier design using 4:2, 5:2, and 7:2 compressors. This compressor is designed with an approximate method using majority logic; compared to this traditional method, the proposed majority approximate compressor method processed less error differences in multiplication output. The proposed approaches resulted in significant reductions in area, power, and delay relative to traditional multipliers. This research compared seven unique comparisons of MAC-based multiplier architecture, and it will have been developed in Verilog hardware description language (HDL) and synthesized on the Xilinx Vertex-5 FPGA, providing reductions of 58.4% in lookup table (LUT) and 76.2% in occupied slices, and proving less power consumption. This design is a highly suitable approach for real-time CNN and digital signal processing (DSP) applications.

Corresponding Author:

Selvarasan Radhakrishnan
Department of Electronics and Communication Engineering
Bharath Institute of Higher Education and Research
Chennai, India
Email: arasanece@gmail.com

1. INTRODUCTION

The convolutional neural network (CNN) will prioritize advancements in the latest artificial intelligence and machine learning techniques, based on numerous applications. These applications include object detection, image improvement, medical image diagnostics, and other similar applications [1], [2]. Here, CNN is particularly skilled in the process of extracting image features from high-dimensional data, and it can facilitate the production of exact decisions in real-time image applications [3]. In particular, these applications required hardware accelerators that were able to successfully manage the processing needs of CNN operations. Employing them in conjunction with edge computing environments was particularly effective [1], [2]. It is also possible to combine the multiple accumulate (MAC) operations with this technique to boost the performance of the CNN in terms of increased speed, and efficiency in hardware while

maintaining a balanced error accumulation in the MAC operations, as discussed by Park *et al.* [4], and also addressed by Edavoor *et al.* [5], who designed approximate multipliers for efficient neural network acceleration. The traditional method, which uses precise Dadda, Wallace tree, and Vedic multipliers, usually leads to more complicated logic and higher power consumption [6]–[8]. For instance, the Dadda and Wallace multipliers will require several stages for the reduction of partial products, and the Wallace tree, in particular, will also integrate with parallel prefix adders to provide very fast operations, as discussed by Momeni *et al.* [6], and further elaborated by Strollo *et al.* [7] and Jiang *et al.* [8], who analyzed the trade-offs between different multiplier architectures. The fact that these processes will take up several logical adders, half adders, and compressors typically results in an increase in the amount of area, delay, and power consumption. These details about compressor and multiplier structure were explained by Bala *et al.* [9], with additional discussion by Ghasemzadeh *et al.* [10] and Fathi *et al.* [11], who focused on high-speed and efficient compressor designs. It is necessary for the proposed approach to lower the number of logic gates to lessen the complexity of the logic size [12], [13]. The standard approach of a full adder will need up to five logic gates, and 4:2 compressors will utilize up to 10 logic gates, similarly, the proposed technique of an approximate majority adder will occupy only two logic gates, and using this approximate full adder [14] in 4:2, 5:2, and 7:2 compressors, it will reduce number of logic gates and reduces the complexity of multiplication. The goal of this study is to reduce multiplication complexity by using majority logic compressors. An additional aspect to consider is the multiplier design that incorporates MAC architecture.

The applications of digital signal processing (DSP) require careful design and optimization to resolve power saving issues while maintaining an acceptable error rate [3], [15], [16]. The evaluation of error metrics, which in this case includes error rate (ER%), mean error distance (MED), normalized mean error distance (NMED), and mean relative error distance (MRED), is done to demonstrate the architecture's resilience in situations where approximate computation is acceptable, as described by Zendegani *et al.* [15], and also discussed by Ansari *et al.* [3] and Mrazek *et al.* [16], who evaluated error metrics for various approximate multipliers. Detailed information on the precise and proposed majority logic compressors for 4:2, 5:2, and 7:2 configurations is provided in the remaining part of this study, which is section 1. Section 2 describe the overview of exact and approximate compressors, the remainder of this paper is organized as follows. Section 3 presents the methodology for the exact multiplier design using an approximate adder architecture, including the architectural framework and operational principles. Section 4 discusses the experimental results and performance evaluation, highlighting the effectiveness of the proposed design in terms of accuracy, power consumption, area utilization, and computational efficiency. Finally, section 5 concludes the paper by summarizing the key findings and outlining potential directions for future research.

2. OVERVIEW OF EXACT AND APPROXIMATE COMPRESSORS

Compressors improve the design of hardware by reducing the number of stages and partial products in the multiplication operations, as described by Zendegani *et al.* [15] and Esposito *et al.* [17]. In this compressor architecture, there are two categories of the design: exact and approximate methods, as discussed by Mrazek *et al.* [16]. Due to the fact that the exact compressor will take up more logic area, it is preferred for applications that require a higher level of accuracy, as explained by Momeni *et al.* [6]. However, in many applications, this higher level of accuracy is not required; instead, the main priority is minimizing logic size and power consumption. For such cases, approximate compressors are utilized, as reported by Zendegani *et al.* [15] and Esposito *et al.* [17]. A fundamentally wide-ranging adder-based compressor design is commonly used; the compressor logic size depends upon the underlying adder design. When the adder occupies more logic, the compressor also increases in logic and power consumption. To address this, we introduced a novel adder architecture in this design, which minimizes logic size and also introduces approximation in the output, as seen in the works of Edavoor *et al.* [5] and Strollo *et al.* [7]. Additionally, this approach minimizes the complexity of the hardware, as well as the amount of power consumed and delay experienced. There are three distinct designs for the proposed compressor method introduced: an approximate full adder-based compressor, an approximate full adder [18]–[20] with a multiplexer-based compressor, and a majority gate-based compressor [21]–[25] each of these designs features its own unique collection of architectural elements. The performance of each of these compressors was evaluated and compared to traditional exact compressors.

2.1. Exact compressor

Exact compressors were developed to carry out arithmetic operations with exact precision, ensuring that the sum and carry bits are calculated precisely and without any loss of bits. Utilization of these compressors is widespread in applications where accuracy is of the utmost importance, as highlighted by Momeni *et al.* [6] and Ghasemzadeh *et al.* [10]. Although they are extremely reliable, accurate compressors

might need a significant amount of hardware resources, which can result in more power consumption, increased logic size, and greater computational latency compared to their approximate equivalents, as discussed by Fathi *et al.* [11], Reddy *et al.* [12], and Zendegani *et al.* [15]. The exact compressor architectures employed in the multiplier reduction stage have been shown in Figure 1. These 4:2, 5:2, and 7:2 architectures are the most common types of exact compressor designs which form basic units in reducing the partial-products since they combine a plurality of inputs into fewer outputs without changing the arithmetic correctness. The following logical (1) are relevant to this design.

$$\begin{aligned} Sum &= x1 \oplus x2 \oplus x3 \oplus x4 \oplus cin \\ Carry &= \big((x1 \oplus x2 \oplus x3) \cdot x4\big) + \Big(cin \cdot \big((x1 \oplus x2 \oplus x3) \oplus x4\big)\Big) \\ Cout &= (x1 \cdot x2) + \big(x3 \cdot (x1 \oplus x2)\big) \end{aligned} \quad (1)$$

Based on the configuration of the 4:2 compressors, which can be seen in Figure 1(a), this architecture is developed with a 5-input and 3-output design, including carry-in and carry-out, as described by Ghasemzadeh *et al.* [10] and Reddy *et al.* [12]. The next architecture (Figure 1(b), the 5:2 compressors, has five input bits ($x1, x2, x3, x4, x5$) combined with two carry-in signals (cin_1 and cin_2). These 5:2 compressors are able to improve upon the capabilities of the 4:2 compressors and also reduce the number of partial products in multiplier design, as discussed by Fathi *et al.* [11] and Edavoor *et al.* [5]. In the case of the 5:2 compressors, the logical equations are presented in (2).

$$\begin{aligned} Sum &= x1 \oplus x2 \oplus x3 \oplus x4 \oplus x5 \oplus cin_1 \oplus cin_2 \\ Carry5 &= (x5 \cdot cin_2) + (cin_2 \cdot (x4 \oplus cin_1 \oplus (x1 \oplus x2 \oplus x3)) + \\ &\quad ((x4 \oplus cin_1 \oplus (x1 \oplus x2 \oplus x3)) \cdot x5) \\ Cout1 &= (x1 \cdot x2) + \big(x3 \cdot (x1 \oplus x2)\big) \\ Cout2 &= (x4 \cdot cin_1) + \big(cin_1 \cdot (x1 \oplus x2 \oplus x3)\big) + \big((x1 \oplus x2 \oplus x3) \cdot x4\big) \end{aligned} \quad (2)$$

Therefore, the 5:2 compressors are considered better for multiplication operations, although it requires more logic area due to the number of full adders used in its design, as analyzed by Edavoor *et al.* [5] and Fathi *et al.* [11]. Similarly, the 7:2 compressor method also uses five full adders in its design, as verified in Figure 1(c). This architecture has seven input bits ($x1, x2, x3, x4, x5, x6, x7$) and two carry-ins (cin1 and cin2), resulting in four outputs ($sum, carry, Cout_1, and\ Cout_2$), as described by Ghasemzadeh *et al.* [10] and Fathi *et al.* [11]. The details of this logic are briefly explained in the following logical (3).

$$\begin{aligned} Sum &= x1 \oplus x2 \oplus x3 \oplus x4 \oplus x5 \oplus x6 \oplus x7 \oplus cin \\ Carry &= \big((x1 \oplus x2 \oplus x3 \oplus x4 \oplus x5 \oplus x6 \oplus x7) \cdot cin_1\big) + (\text{cin\_2}((\text{x1} \oplus \text{x2} \oplus \\ &\quad \text{x3} \oplus \text{x4} \oplus \text{x5} \oplus \text{x6} \oplus \text{x7}) \oplus \text{cin\_1})) \\ Cout_1 &= (x1 \cdot x2) + \big(x3 \cdot (x1 \oplus x2)\big) \\ Cout_2 &= \big((x1 \oplus x2 \oplus x3) \cdot x4\big) + \Big(x5 \cdot \big((x1 \oplus x2 \oplus x3) \oplus x4\big)\Big) \\ Cout_3 &= \big((x1 \oplus x2 \oplus x3 \oplus x4 \oplus x5) \cdot x6\big) + \Big(x7 \cdot \big((x1 \oplus x2 \oplus x3 \oplus x4 \oplus x5) \oplus x6\big)\Big) \end{aligned} \quad (3)$$

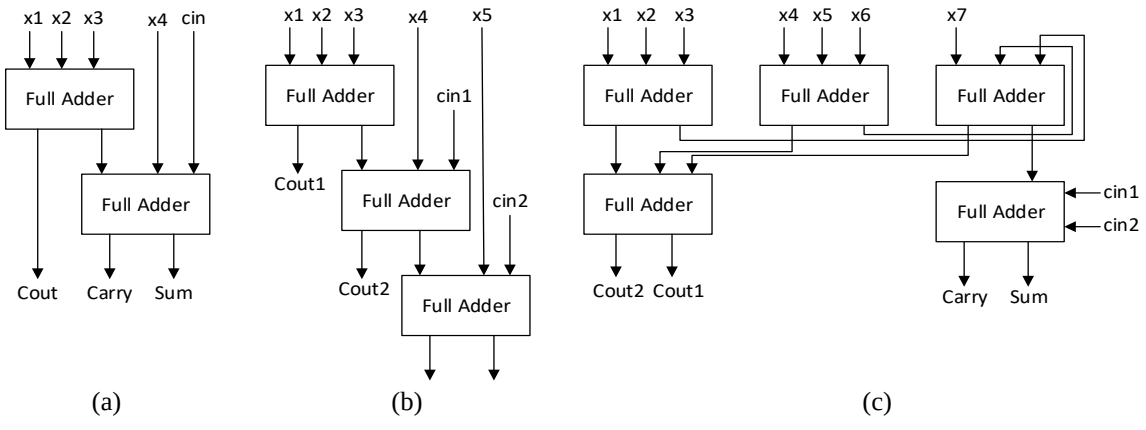


Figure 1. Architectures of compressor circuits: (a) 4:2 exact compressor, (b) 5:2 exact compressor, and (c) 7:2 exact compressor

2.2. Approximate compressor

There is a hardware-efficient alternative to precise compressors known as approximate compressors. These compressors were developed primarily for applications that can tolerate modest computational errors, as discussed by Reddy *et al.* [12] and Jooq *et al.* [13]. On the other hand, these systems sacrifice accuracy in exchange for considerable reductions in hardware complexity, power consumption, and latency, as shown by Edavoor *et al.* [5] and Strollo *et al.* [7]. As a result of the architectural design that incorporates approximate full adders, approximate compressors provide outputs that are approximate equivalent, as presented by Edavoor *et al.* [5] and Jooq *et al.* [13]. The (4) contains the logical equations of an approximate complete adder, which are responsible for the creation of the sum and carry variables.

$$Sum = x1 \oplus x2 \oplus cin$$
$$Carry = x2 \cdot cin \quad (4)$$

The design architecture of the approximate full adder [14], as shown in Figure 2 uses only two logic gates (XOR and AND gates), as proposed by Reddy *et al.* [12] and Jooq *et al.* [13]. Compared to the exact method, the traditional architecture uses five logic gates; this reduction contributes to lower power consumption, reduced area, and faster computation times, as demonstrated by Edavoor *et al.* [5] and Strollo *et al.* [7]. Using the design of an approximate full adder [14], The approximate-adder-based compressor architectures used in the multiplier design are shown in Figure 3. The figure shows the summary of approximate addition principles used to realise compressors of different input-output configurations, 4:2, 5:2 and 7:2 compressors. the proposed method integrated the architecture of 4:2, 5:2, and 7:2 compressors and demonstrated their performance, as shown in Figure 3(a) 4:2 compressors, Figure 3(b) 5:2 compressors, and Figure 3(c) 7:2 compressors, following the strategies discussed by Edavoor *et al.* [5] and Jooq *et al.* [13].

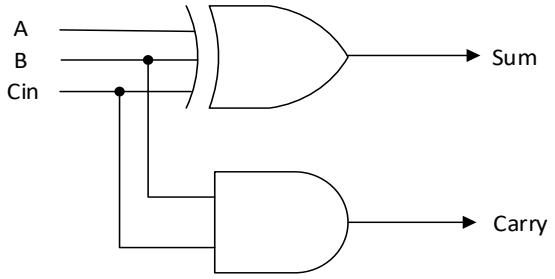
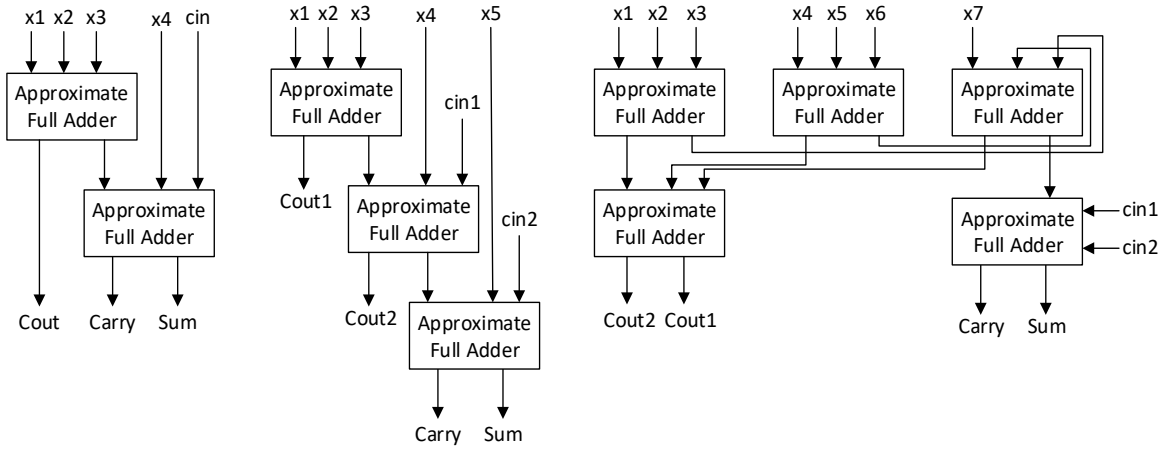


Figure 2. Approximate full adder design



(a) (b) (c)

Figure 3. Approximate-adder-based architectures: (a) 4:2 approximate compressor, (b) 5:2 approximate compressor, and (c) 7:2 approximate compressor

The core of approximate full adder based 4:2 compressor logical (5) are as:

$$Sum = x1 \oplus x2 \oplus x3 \oplus x4 \oplus cin$$
$$Carry = x4.cin$$
$$Cout = x2.x \quad (5)$$

for 5:2 compressors using approximate full adder [14], the following is the logical equation that should be used (6).

$$Sum = x1 \oplus x2 \oplus x3 \oplus x4 \oplus x5 \oplus cin_1 \oplus cin_2$$
$$Carry = x5.cin_2$$
$$Cout1 = x2.x3$$
$$Cout2 = x4.cin_1 \quad (6)$$

With regard to 7:2 compressors using approximate full adder, the following is the logical (7).

$$Sum = x1 \oplus x2 \oplus x3 \oplus x4 \oplus x5 \oplus x6 \oplus x7 \oplus cin1 \oplus cin2$$
$$Carry = cin_1.cin_2$$
$$Cout1 = (x2.x3) \oplus (x5.x6) \oplus ((x4 \oplus x5 \oplus x6).x7))$$
$$Cout2 = (x5.x6).((x4 \oplus x5 \oplus x6).x7)) \quad (7)$$

The 4:2, 5:2, and 7:2 methods of approximate compressor design, provide an efficient partial product reduction in multipliers.

2.3. Approximate with mux-based compressor

The approximate with mux-based compressors are a further enhancement of approximate compressor designs, incorporating multiplexers to improve resource efficiency and computation speed, as discussed by Edavoor *et al.* [5] and Jooq *et al.* [13]. As presented by Edavoor *et al.* [5], by integrating multiplexers, these compressors dynamically handle the selection of intermediate results, further reducing logic complexity and delay while maintaining an approximate output. These compressors preserve the approximate full adder at their core but add multiplexer logic to handle specific input configurations, allowing for a more efficient reduction of partial products, as demonstrated in both Edavoor *et al.* [5] and Jooq *et al.* [13]. The multiplexer-based design ensures that only critical inputs contribute to the computation at each stage, minimizing unnecessary logic operations and enhancing the overall efficiency of the compressor, as shown in the hardware evaluations by Edavoor *et al.* [5]. These designs are implemented across 4:2, 5:2, and 7:2 compressor architectures, each optimized for specific input configurations. This follows the same procedure as the approximate full adder [14] compressor technique, but the boolean equation for the internal architecture of this compressor is further reduced to produce the final sum output, effectively reducing delay and hardware complexity. The logical (8) should be used for 4:2 compressors using an approximate mux-based full adder, as shown by Edavoor *et al.* [5].

$$Sum = \left(\overline{x4} \oplus (x1 \oplus x2 \oplus x3)\right) + \left(x4.(x2.x3)\right)$$
$$Carry = \left(\overline{x4}.(x2.x3)\right) + x4 \quad (8)$$

Similarly, in the 5:2 compressors, intermediate results are routed through a multiplexer, ensuring efficient computation of sum and carry outputs (Cout1 and Cout2) with reduced logic depth, as detailed by Edavoor *et al.* [5] and Jooq *et al.* [13]. The (9) will be used for 5:2 compressors following the methodology discussed in these works.

$$Sum = (\overline{cin} \oplus (x1 \oplus x2 \oplus x3) \oplus x4 \oplus x5)) + (cin.(x4.x5))$$
$$Carry = \left(\overline{cin}.(x4.x5)\right) + \left(cin.(x2.x3)\right) \quad (9)$$

The more complex design of 7:2 architectures, which use multiple approximate full adders to compute intermediate sums (sx1 and sx2), and multiplexers to dynamically manage the flow of these results, produces the final sum and carry outputs, as described by Edavoor *et al.* [5] and Jooq *et al.* [13]. This process is detailed in the logical (10). The architecture design of the approximate mux-based compressor, shown in Figure 4, provides a scalable solution that ensures minimal hardware complexity, making it ideal for large-scale input configurations, as discussed by Jooq *et al.* [13]. The proposed 4:2 approximate compressor

employs one approximate full adder and two multiplexers to generate the Sum and Carry outputs (Figure 4(a)). The proposed 5:2 approximate compressor utilizes three approximate full adders and two multiplexers (Figure 4(b)). The proposed 7:2 approximate compressor consists of four approximate full adders and two multiplexers (Figure 4(c)). The architecture achieves higher compression capability while maintaining low power consumption and high computational efficiency.

Across all three architectures, the use of multiplexers reduces the critical path delay and power consumption, while the approximate logic maintains acceptable accuracy levels for error-tolerant applications, as demonstrated by both Edavoor *et al.* [5] and Jooq *et al.* [13].

$$Sum = \left(\overline{cin} \oplus \left(\left((x1 \oplus x2 \oplus x3) \oplus x4 \oplus x5\right) \oplus x6 \oplus x7\right)\right) + \left(cin.\left(x6.x7\right)\right)$$
$$Carry = \left(\overline{cin}.\left(x6.x7\right)\right) + \left(cin.\left(x4.x5\right)\right) \quad (10)$$

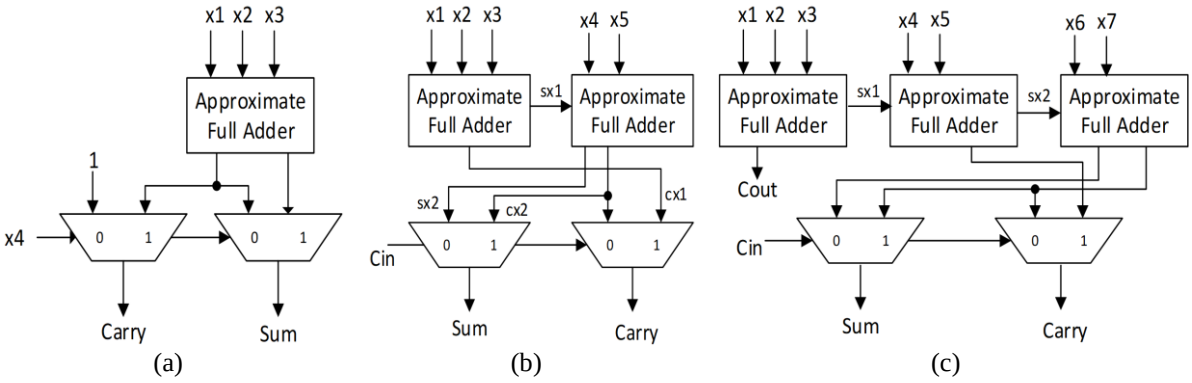


Figure 4. Proposed multiplexer-based approximate compressor architectures: (a) 4:2 approximate mux based compressor, (b) 5:2 approximate mux based compressor, and (c) 7:2 approximate mux based compressor

2.4. Majority gate-based compressor

The proposed design of majority gate-based compressors presents an effective and optimal alternative to previous compressor designs. This design utilizes a very small number of 3-input majority logic gates in the compressor design, which reduces the total logic size and power consumption, as shown by Reddy *et al.* [12] and Jooq *et al.* [13]. The expression of majority gate logic with multi-input expression is (11).

$$Majority(A,B,C) = (A.B) + (B.C) + (C.A) \quad (11)$$

The operations of 4:2 compressor required two majority logic gate is expressed in (12).

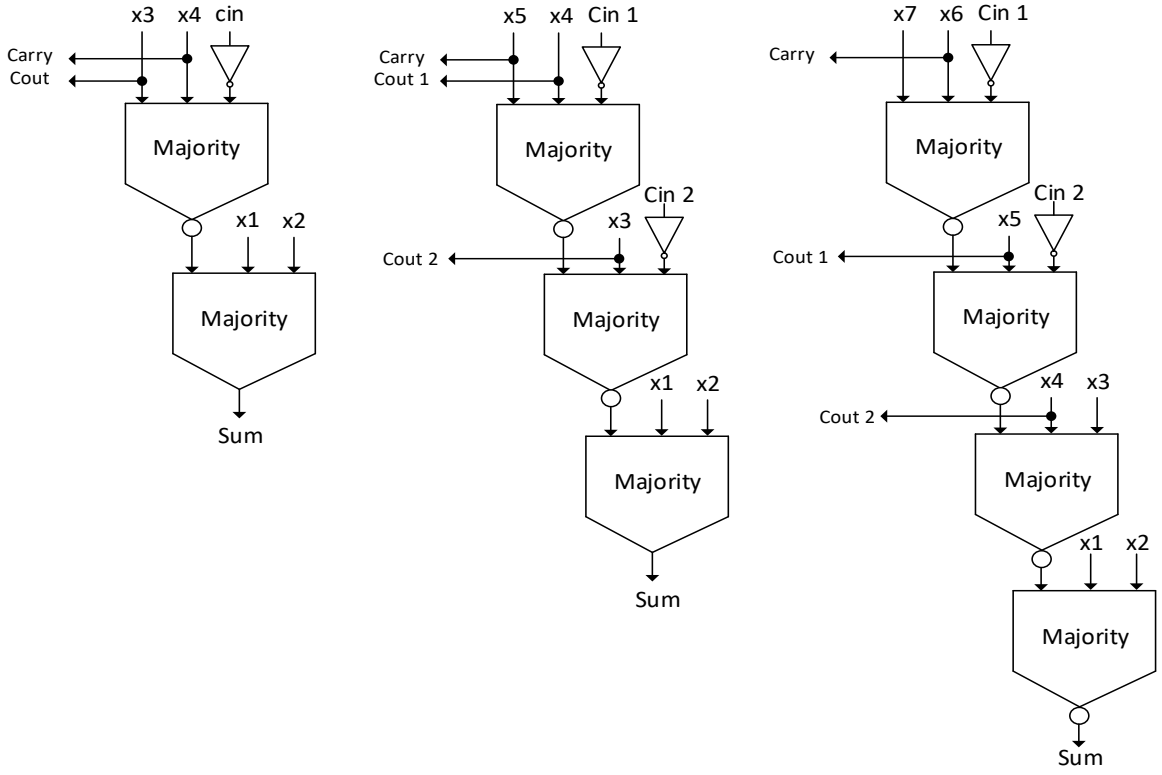
$$Sum = \overline{\left((x3.x4) + \left(x4.\overline{cin}\right) + \left(x3.\overline{cin}\right)\right)}.(x1 + x2) + (x1.x2)$$
$$Carry = x4$$
$$Cout = x3 \quad (12)$$

The 5:2 compressor utilizes three layer of majority logic gates, and the expression of 5 input and 2 output based compressor using majority logic is (13).

$$Sum = (\overline{\left(x4\left(x5 + \overline{cin1}\right) + \left(x5.\overline{cin1}\right)\right)}.(x3 + \overline{cin2}) + (x3.\overline{cin2}).(x2 + x1) + (x2.x1)$$
$$Carry = x5$$
$$Cout1 = x4$$
$$Cout2 = x3 \quad (13)$$

Furthermore, the logical equation of the 7:2 compressor design that makes use of the majority logic approach and has the capacity to accommodate seven primary inputs and two outputs is demonstrated in (14), as

discussed by Reddy *et al.* [12] and Jooq *et al.* [13]. In this particular instance, it makes use of four layers of majority gates. The proposed majority-logic based compressor architectures for partial product reduction in the approximate multiplier design are shown in Figure 5. These compressor structures are the basic building blocks of the reduction stage which use majority-gate logic to achieve the circuit implementation and lower hardware complexity, power consumption and computation efficiency. The structure of the proposed 4:2 majority-logic compressor is illustrated in Figure 5(a). The input bits go through a small majority-gate network where the sum and carry outputs are calculated with minimum logic elements. The proposed 5:2 majority-logic compressor can be seen as an expansion of the majority-gate concept by one more input bit and is depicted in Figure 5(b). By using the cascaded majority-gates structure, we get the effective realization of output bits with the same simplicity of implementation and short critical path. Figure 5(c) illustrates the 7:2 majority-logic compressors, which can operate with increased input bits number, providing further partial products compression.



(a) (b) (c)

Figure 5. Proposed majority-logic-based compressors (a) 4:2 majority-logic compressor, (b) 5:2 majority-logic compressor, and (c) 7:2 majority-logic compressor

3. EXACT MULTIPLIER DESIGN USING APPROXIMATE ADDER ARCHITECTURE

This section presents the methodological framework adopted for the exact multiplier design using an approximate adder architecture and proposed Approximate multiplier design. The exact method of multiplier design, which is commonly used in precision-critical applications such as image and signal processing, tends to occupy a significantly larger logic area, as reported by Momeni *et al*. [6]. Traditional multipliers such as Wallace tree, Dadda, Vedic, and array multipliers also use this exact computation sequence. A common issue with exact multiplication is increased logic size and power consumption. For that reason, recent research has focused on various approximation-based multiplication techniques [1], [4], [15], [16]. Recent studies, including those by [4], [15], [1], [16] have demonstrated the effectiveness of approximation techniques as feasible replacements to reduce logic area, power consumption, and delay. To address these challenges, the proposed technique in this study introduces an approximate full adder [14] technique, it's inspired by the work of Strollo *et al.* [7], which produces the same sum output as a conventional full adder but simplifies the carry generation. Figure 6 shows the traditional design of Dadda 8×8 multiplier its highlights these

improvements. While this design taken from the conventional Dadda multiplier architecture, its produces 64 partial products and typically requires four reduction stages, the proposed design minimizes the number of stages by employing a combination of half adders and approximate full adders, effectively reducing internal complexity.

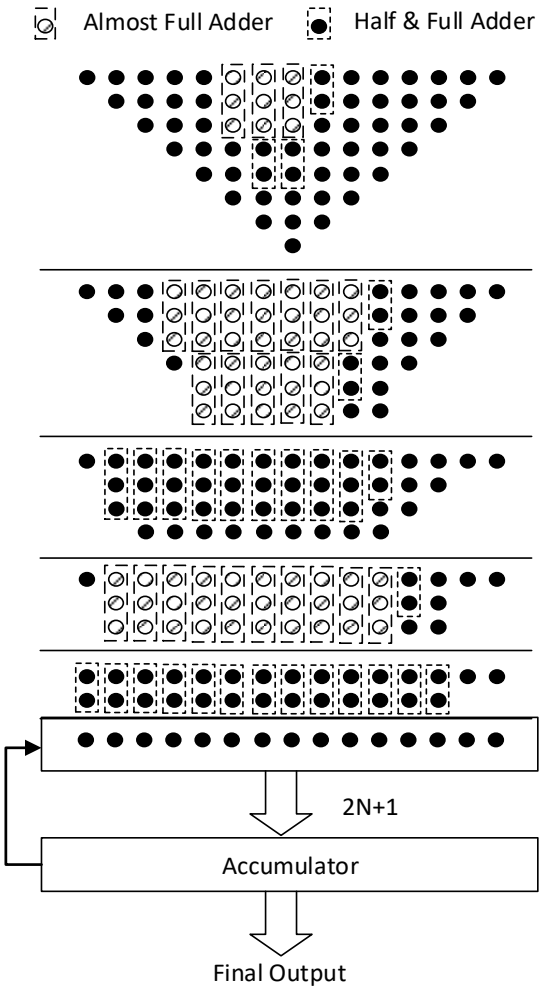


Figure 6. 8×8 Dadda multiplication using approximate full adder

3.1. Proposed approximate multiplier design using 4:2, 5:2, and 7:2 compressors architecture

The proposed approach of this research using approximate multiplier architecture is specifically designed to reduce the number of stages in the multiplication process. To achieve this, 4:2, 7:2, and 5:2 majority logic compressors are additionally integrated into the multiplier design architecture, enabling a high level of compression of partial products in fewer steps, as discussed by [10], [11]. This maximum input reduction enhances both speed and efficiency in three levels of multiplier design that combine exact, approximate, and transduce techniques. Compared to the traditional design of multipliers using only approximate 4:2 compressor architecture, the proposed design significantly reduces the logic size and computational complexity through this innovative approach, as described by Edavoor *et al.* [5] and further validated in Reddy *et al.* [12]. Notably, as shown in Figure 7, the multiplier requires only two stages in the 8×8 multiplication architecture. Specifically, in the proposed approximate multiplier, stage 1 is exclusively handled by 7:2 and 5:2 compressors, which reduce the number of partial products; as a result, the number of intermediate signals passed to stage 2 is minimal. Stage 2 further compresses the data using exact 4:2 compressors, full adders, and half adders, maintaining accuracy where it is most critical. This combination of optimized techniques ensures a compact design with low latency and energy efficiency.

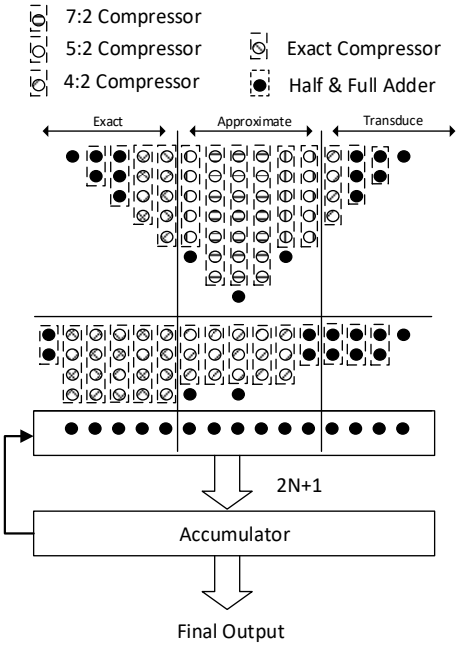


Figure 7. Proposed higher bit compressor based 8×8 Dadda multiplier

4. RESULTS

The performance and error analysis of the proposed approximate multiplier design, implemented in Verilog hardware description language (HDL), was synthesized using Xilinx Vertex-5 FPGA and compared for utilization results such as slice registers, slice lookup tables (LUTs), occupied slices, bonded IOB, delay, and power. The comparative performance results of the MAC-based 8×8 approximate Dadda multiplier design employing approximate compressors are presented in Table 1. In addition, error metrics—including analysis with the verification method in simulation using modelsim—were employed; the parameters analyzed include ER%, MED, NMED, and MRED, as supported by the work of [3], [8].

In a vital role of optimizing digital arithmetic circuits by reducing intermediate results during computations, various compressor designs of the proposed methodology—such as exact, approximate, those incorporating multiplexers, and majority gates—offer a tradeoff between utilization and performance. Table 2 shows the critical performance metrics across different compressor designs to evaluate their effectiveness. These results show that approximate compressors, particularly multiplexer- and majority logic based architectures, achieve notable reductions in LUT usage, occupied slices, and delay, as demonstrated by Edavoor *et al.* [5] and Reddy *et al.* [12].

Error analysis estimating the error analysis and its reliability using approximate multipliers, this work quantifies the errors introduced by approximate methods, which is essential to ensure the design meets the accuracy requirements of CNN applications, as demonstrated by Kim *et al.* [1] and Ansari *et al.* [3]. Key error metrics—ER, MED, NMED, and MRED—provide a structured framework to assess and compare the effectiveness of various approximate computing architectures, as discussed by Ansari *et al.* [3] and Jiang *et al.* [8]. These error metrics are chosen for the approximation strategy, which trades off between efficiency and precision, and directly affect the selection of approximate values in specific cases, especially in error resilient applications such as image processing and IoT systems, as reported by Kim *et al.* [1] and Hammad *et al.* [2]. This work is able to minimize errors while achieving gains in power, area, and speed. For instance, a higher ER% implies frequent computational deviation and may be unsuitable for precision-critical tasks, while a lower ER% indicates greater reliability and stability, even for approximate designs, as analyzed by Edavoor *et al.* [5] and Jiang *et al.* [8]. The analysis of the error assessment framework is widely adopted in recent approximate research, including the works of Hammad *et al.* [2], Ansari *et al.* [3], and Jiang *et al.* [8]. The frequency error analysis of ER% is (15), with the number of incorrect versus the total number of outputs.

Table 3 analyzes the evaluation of error metrics across multiple approximate multiplier design in which highlights the inherent tradeoff between accuracy and approximate values. In this architecture significantly improve the error behavior and observed through the integration of advanced techniques, including majority logic compressor logic. The conventional logic design exhibits a higher level of error at 6.1%, indicating a higher range of exact computations.

Table 1. Performance analysis of MAC based 8×8 approximate Dadda multiplier design using approximate compressors

Design	LUT	FF	Delay (ns)	Power (W)
Conventional design [17]	238	198	15.463	0.613
4:2 approximate FA	124	51	4.648	0.616
4:2 approximate FA-mux	135	62	4.648	0.617
4:2 approximate FA-majority compressor	122	50	4.648	0.612
Proposed compressor using approximate full adder	111	50	2.525	0.614
Proposed compressor using approximate with mux	101	47	2.525	0.612
Proposed compressor using majority compressor	99	51	2.525	0.606

Table 2. Comparisons and analysis of only compressors

Design	LUT	Slices	IOBs	Delay (ns)	Fan-out
Exact 4:2	6	4	8	5.779	1.73
Exact 5:2	9	5	11	6.782	1.75
Exact 7:2	15	6	13	6.857	1.83
Approximate 4:2	4	3	8	4.386	1.44
Approximate 5:2	3	2	11	5.180	1.17
Approximate 7:2	4	3	13	5.180	1.12
Approximate with mux 4:2	3	2	6	5.377	1.57
Approximate with mux 5:2	3	1	8	6.560	1.40
Approximate with mux 7:2	4	2	10	7.743	1.50
Majority 4:2	1	1	8	4.307	1.33
Majority 5:2	2	1	11	4.992	1.33
Majority 7:2	2	1	13	5.678	1.27

Table 3. Error analysis of approximate multiplier

Design	ER%	MED	NMED	MRED
Conventional approximate full adder	6.1	1.1	1.4×10^{-3}	2.3×10^{-3}
4:2 compressor using approximate full adder	5.4	1.0	1.2×10^{-3}	2.0×10^{-3}
4:2 compressor using approximate full adder with mux	4.8	0.9	1.1×10^{-3}	1.8×10^{-3}
4:2 compressor using majority gate	3.2	0.8	0.08×10^{-3}	1.5×10^{-3}
Proposed compressor using approximate full adder	3.0	0.8	1.7×10^{-3}	1.2×10^{-3}
Proposed compressor using approximate full adder with mux	2.6	0.7	1.6×10^{-3}	1.1×10^{-3}
Proposed compressor using majority gate	1.9	0.6	0.4×10^{-3}	0.8×10^{-3}

5. CONCLUSION

This research introduces an innovative approximation-based MAC multiplier architecture that significantly enhances computational efficiency for convolution operations in CNN applications. This design leverages advanced 4:2, 5:2, and 7:2 compressors utilizing approximate compressors, approximate with mux based compressor and majority compressors, achieving marked improvement over conventional accurate Dadda multipliers partial products reductions. The proposed methods demonstrate a notable reduction in logic size, area, and power consumption while maintaining computational accuracy. This work implementation on the Xilinx Vertex-5 FPGA proves the performance evaluation of LUT reductions up to 58.4%, an occupied slice of 76.2%, and a latency of 60.5%. Moreover, this research includes the detailed error analysis using metrics such as ER, MER, NMED, and MRED and proves the lowest error values across all metrics using majority logic-based compressors, which highlights their suitability for approximating computing environments. Further research may extend this work to higher order multipliers (16×16 and 32×32), explore dynamic approximation techniques, and additionally integrate with CNN applications to further achieve the benefits of approximation.

FUNDING INFORMATION

Authors state no funding involved.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration.

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Selvarasan Radhakrishnan	✓	✓	✓	✓	✓	✓		✓	✓	✓			✓	✓
Sudhagar Govindhaswamy		✓	✓	✓	✓	✓				✓				
Rasadurai Kumaravel		✓	✓	✓	✓		✓	✓	✓	✓			✓	

C : **C**onceptualization
M : **M**ethodology
So : **So**ftware
Va : **Va**lidation
Fo : **Fo**rmal analysis
I : **I**nvestigation
R : **R**esources
D : **D**ata Curation
O : Writing - **O**riginal Draft
E : Writing - Review & **E**diting
Vi : **Vi**sualization
Su : **Su**pervision
P : **P**roject administration
Fu : **Fu**nding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

Data availability is not applicable to this paper as no new data were created or analyzed in this study.

REFERENCES

[1] M. S. Kim, A. A. Del Barrio, H. J. Kim, and N. Bagherzadeh, "The Effects of Approximate Multiplication on Convolutional Neural Networks," *IEEE Transactions on Emerging Topics in Computing*, vol. 10, no. 2, pp. 904–916, Apr. 2022, doi: 10.1109/TETC.2021.3050989.

[2] I. Hammad, L. Li, K. El-Sankary, and W. M. Snelgrove, "CNN Inference Using a Preprocessing Precision Controller and Approximate Multipliers with Various Precisions," *IEEE Access*, vol. 9, pp. 7220–7232, 2021, doi: 10.1109/ACCESS.2021.3049299.

[3] M. S. Ansari, V. Mrazek, B. F. Cockburn, L. Sekanina, Z. Vasicek, and J. Han, "Improving the Accuracy and Hardware Efficiency of Neural Networks Using Approximate Multipliers," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 2, pp. 317–328, Feb. 2020, doi: 10.1109/TVLSI.2019.2940943.

[4] G. Park, J. Kung, and Y. Lee, "Design and Analysis of Approximate Compressors for Balanced Error Accumulation in MAC Operator," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 68, no. 7, pp. 2950–2961, Jul. 2021, doi: 10.1109/TCSI.2021.3073177.

[5] P. J. Edavoor, S. Raveendran, and A. D. Rahulkar, "Approximate multiplier design using novel dual-stage 4:2 compressors," *IEEE Access*, vol. 8, pp. 48337–48351, 2020, doi: 10.1109/ACCESS.2020.2978773.

[6] A. Momeni, J. Han, P. Montuschi, and F. Lombardi, "Design and analysis of approximate compressors for multiplication," *IEEE Transactions on Computers*, vol. 64, no. 4, pp. 984–994, Apr. 2014, doi: 10.1109/TC.2014.2308214.

[7] 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*, vol. 67, no. 9, pp. 3021–3034, Sep. 2020, doi: 10.1109/TCSI.2020.2988353.

[8] H. Jiang, C. Liu, F. Lombardi, and J. Han, "Low-Power Approximate Unsigned Multipliers with Configurable Error Recovery," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 66, no. 1, pp. 189–202, Jan. 2019, doi: 10.1109/TCSI.2018.2856245.

[9] S. T. Bala, D. Shangavi, and P. Sangeetha, "Area and power efficient approximate wallace tree multiplier using 4:2 compressors," in *Proceedings of IEEE International Conference on Intelligent Computing and Communication for Smart World, I2C2SW 2018*, IEEE, Dec. 2018, pp. 287–290, doi: 10.1109/I2C2SW45816.2018.8997160.

[10] M. Ghasemzadeh, A. Akbari, A. Soltani, and K. Hadidi, "A new ultra high speed 7-2 compressor with a new structure," in *Proceedings of the 22nd International Conference Mixed Design of Integrated Circuits and Systems, MIXDES 2015*, IEEE, Jun. 2015, pp. 262–265, doi: 10.1109/MIXDES.2015.7208523.

[11] A. Fathi, B. Mashoufi, and S. Azizian, "Very Fast, High-Performance 5-2 and 7-2 Compressors in CMOS Process for Rapid Parallel Accumulations," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 28, no. 6, pp. 1403–1412, Jun. 2020, doi: 10.1109/TVLSI.2020.2983458.

[12] K. M. Reddy, M. H. Vasantha, Y. B. N. Kumar, and D. Dwivedi, "Design and analysis of multiplier using approximate 4-2 compressor," *AEU - International Journal of Electronics and Communications*, vol. 107, pp. 89–97, Jul. 2019, doi: 10.1016/j.aeue.2019.05.021.

[13] M. K. Q. Jooq, M. Ahmadinejad, and M. H. Moaiyeri, "Ultraefficient imprecise multipliers based on innovative 4:2 approximate compressors," *International Journal of Circuit Theory and Applications*, vol. 49, no. 1, pp. 169–184, Jan. 2021, doi: 10.1002/cta.2876.

[14] K. C. Pathak, A. D. Darji, and J. N. Sarvaiya, "Low power Dadda multiplier using approximate almost full adder and Majority logic based adder compressors," in *2022 IEEE Region 10 Symposium, TENSYMP 2022*, IEEE, Jul. 2022, pp. 1–6, doi: 10.1109/TENSYMP54529.2022.9864428.

[15] R. Zendegani, M. Kamal, M. Bahadori, A. Afzali-Kusha, and M. Pedram, "RoBA Multiplier: A Rounding-Based Approximate Multiplier for High-Speed yet Energy-Efficient Digital Signal Processing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 25, no. 2, pp. 393–401, Feb. 2016, doi: 10.1109/TVLSI.2016.2587696.

[16] V. Mrazek, Z. Vasicek, L. Sekanina, H. Jiang, and J. Han, "Scalable Construction of Approximate Multipliers with Formally

Guaranteed Worst Case Error," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 26, no. 11, pp. 2572–2576, Nov. 2018, doi: 10.1109/TVLSI.2018.2856362.
[17] D. Esposito, A. G. M. Strollo, E. Napoli, D. De Caro, and N. Petra, "Approximate multipliers based on new approximate compressors," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 65, no. 12, pp. 4169–4182, Dec. 2018, doi: 10.1109/TCSI.2018.2839266.
[18] T. Kong and S. Li, "Design and Analysis of Approximate 4-2 Compressors for High-Accuracy Multipliers," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 29, no. 10, pp. 1771–1781, Oct. 2021, doi: 10.1109/TVLSI.2021.3104145.
[19] S. F. P. and A. Mishra, "High speed and high performance approximate multipliers for error resilient applications," *Integration*, vol. 108, p. 102678, May 2026, doi: 10.1016/j.vlsi.2026.102678.
[20] G. Di Meo, G. Saggese, A. G. M. Strollo, and D. De Caro, "Design of Generalized Enhanced Static Segment Multiplier with Minimum Mean Square Error for Uniform and Nonuniform Input Distributions," *Electronics*, vol. 12, no. 2, p. 446, Jan. 2023, doi: 10.3390/electronics12020446.
[21] V. S. Chowdam, M. V. Naresh, and G. G. Kumar, "Energy-Efficient 8-Bit Approximate Multipliers Design and Analysis for Error-Resilient Applications," *Circuits, Systems, and Signal Processing*, vol. 45, no. 4, pp. 2948–2967, Apr. 2025, doi: 10.1007/s00034-025-03350-z.
[22] S. Ullah and A. Kumar, *Approximate Arithmetic Circuit Architectures for FPGA-based Systems*. Cham: Springer International Publishing, 2023, doi: 10.1007/978-3-031-21294-9.
[23] V. Tammineni, S. K. Beura, M. V. H. B. Murthy, S. Majumdar, and P. Saha, "Optimized recursive approximate multipliers for edge detection and image smoothing applications," *Microsystem Technologies*, vol. 31, no. 10, pp. 2771–2782, Oct. 2025, doi: 10.1007/s00542-024-05810-z.
[24] A. Sadeghi, R. Rasheedi, I. Partin-Vaisband, and D. Pal, "Energy Efficient Compact Approximate Multiplier for Error-Resilient Applications," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 71, no. 12, pp. 4989–4993, Dec. 2024, doi: 10.1109/TCSII.2024.3437235.
[25] I. Rizos, G. Papatheodorou, and A. Efthymiou, "Designing Approximate Reduced Complexity Wallace Multipliers," *Electronics (Switzerland)*, vol. 14, no. 2, p. 333, Jan. 2025, doi: 10.3390/electronics14020333.

BIOGRAPHIES OF AUTHORS

Mr. Selvarasan Radhakrishnan received his B.E. degree in Electronics and Communication Engineering (ECE) from Anna University, Chennai, and his M.E. degree in Communication Systems from Anna University in 2010. He is currently pursuing his Ph.D. degree at Bharath Institute of Higher Education and Research, Chennai, India. He is currently working as an Assistant Professor in the Department of Electronics and Communication Engineering at Kuppam Engineering, Andhra Pradesh, India. His research interests include VLSI design, deep learning, and image processing. He is a life member of ISTE. He can be contacted at email: arasanece@gmail.com.

Dr. Sudhagar Govindhaswamy is currently working as an Associate Professor in the Department of Electronics and Communication Engineering at Bharath Institute of Higher Education and Research, Chennai, India. He received his B.E. degree in Electronics and Communication Engineering (ECE) from Madras University in 2002, and obtained his M.E. degree in Applied Electronics from Sathyabama University, Chennai, in 2006. He completed his Ph.D. in Electronics and Communication Engineering (ECE) at Anna University. He has 21 years of teaching experience in various engineering colleges. His research interests included VLSI design and testing, digital signal processing, and image processing. He can be contacted at email: sudhagar.ece@bharathuniv.ac.in.

Dr. Rasadurai Kumaravel has been serving as a Professor in the Department of Electronics and Communication Engineering at Kuppam Engineering College, Kuppam, Andhra Pradesh, India, since 2017. He holds a Ph.D. in Wireless Broadband Communication Engineering from Anna University, Chennai; an M.E. in Embedded System Technologies from SA Engineering College; and a B.E. in Electronics and Communication Engineering (ECE) from Bharathidasan Engineering College. With academic experience spanning over 20 years, he has served as faculty in various engineering colleges across South India since 2006. He has successfully secured Rs. 51.86 lakhs in R&D grants from reputed funding agencies such as AICTE, MHRD, and DST. He has also served as a College SPOC for MOOC platforms such as NPTEL, SWAYAM, IIRS-ISRO, Spoken Tutorial, Code Chapter–IIT Madras, and the Quantum Innovation Centre. He has authored over 45 research papers, published three patents, participated in more than 32 national and international conferences, and co-authored three technical books in wireless communication and emerging network technologies. He can be contacted at email: krasadurai@gmail.com.