Tuijin Jishu/Journal of Propulsion Technology
ISSN: 1001-4055
Vol. 45 No. 3 (2022)

Design and Implementation of Inexact Multiplier Using 4:2 Compressors with Adaptive Error Control

Talla Srinivasa Rao[1], Dr Ch Srinivasu[2], Dr K Babulu[3]

[1,3] Department of ECE, Jawaharlal Nehru Technological University Gurajada Vizianagaram (JNTUGV), Vizianagaram, Andhra Pradesh, India.

[2] Department of ECE, Raghu Engineering College, Visakhapatnam, Andhra Pradesh, India.

Abstract: **-** Multipliers are crucial in Digital Signal Processing (DSP) applications, but traditional designs have drawbacks like high power use, large size, and long processing times. Inexact multipliers, which allow for small inaccuracies, have become a promising solution as they save power and reduce delays while maintaining acceptable accuracy. However, modern applications require high accuracy, leading to the need for energy-efficient and compact inexact multipliers. In this article, we introduce a new design for an inexact multiplier that is both low-power and space-saving while achieving high accuracy. A key feature is an error compensation circuit designed to improve accuracy. We use approximation techniques at both the compressor and multiplier stages, significantly cutting down on size and power use. This balance of efficiency and precision meets the demands of current DSP applications. Our experiments show that our custom-designed multiplier performs exceptionally well compared to existing solutions. It reduces power use by 72%, cuts latency by 27%, and decreases size by 7%, all without sacrificing accuracy. This demonstrates the effectiveness and competitiveness of our innovative approach.

Keywords*: Inexact Multiplier, 4:2 compressor, Error Compensation Circuit.*

1. Introduction

Multipliers are essential in various fields like digital signal processing (DSP), computer vision, multimedia processing, image recognition, and artificial intelligence. These fields require many multiplication operations, leading to high power consumption, which is a major issue, especially for mobile devices where energy efficiency is crucial. To address this, researchers have explored ways to reduce the power usage of multiplier circuits, and one promising method is to use approximate multiplication. Approximate multiplication is suitable for applications that can tolerate some errors, especially those involving human sensory perception, such as visual or auditory tasks. Since humans have limited sensory ranges, perfect accuracy is not always necessary. In these cases, inexact multiplication can save power while still providing results that are good enough for human perception. Inexact multipliers trade some precision for benefits like smaller size, faster operation, and lower power consumption. There are two main types of inexact multipliers. The first type controls the timing of the multiplier, often by adjusting the voltage. Lowering the voltage increases the delay in the critical path, leading to errors and inexact results. The second type changes the functional behavior of the multipliers themselves. This involves redesigning traditional accurate multiplier circuits, such as the Wallace Tree and Dadda Tree multipliers. Many studies have focused on using inaccurate m:n compressors, where ‘m’ is the number of inputs and ‘n’ is the number of outputs. These compressors compress partial products, which is important because it typically consumes a lot of energy and causes delays. Previous inexact multiplier designs focused on fixed accuracy and power usage. However, there's a growing need for flexible accuracy and power usage, especially in evolving applications like artificial intelligence. Achieving this flexibility usually requires additional hardware. In our research, we introduce a new, highly accurate 4:2 compressor as a key component. Based on this, we develop a high-precision inexact

multiplier. Additionally, we present a technique for adjustable input truncation, allowing for real time adjustments to both accuracy and power usage. The key contributions of our work can be summarized as follows:

- We propose an advanced 4:2 compressor with a focus on high accuracy, which serves as a fundamental building block for our inexact multiplier.
- To enhance accuracy even further, we introduce a straight forward error compensation circuit.
- Our adjustable input truncation technique empowers users to fine tune accuracy and power utilization, making it particularly well suited for DSP applications.

Leveraging the 4:2 compressor, error compensation circuit, and adjustable input truncation technique, we unveil a high precision and adaptable inexact multiplier, offering a comprehensive solution for applications requiring flexibility in accuracy and power utilization. The experimental findings underscore the effectiveness of our adjustable inexact multiplier, which incorporates adjustable input truncation. When compared to the Wallace Tree Multiplier, our approach showcases significant advantages: It reduces delay by 27%, which is crucial for applications demanding real-time processing. The average power utilization is cut down by 40%, with some cases showing an impressive reduction of up to 72%. This is a significant advancement in energy efficiency. The error rate of our proposed multiplier is a mere 11.57%, and it outperforms other inexact multipliers by having the smallest mean error distance. Additionally, our adjustable inexact multiplier with adjustable truncation exhibits lowers area overhead, in contrast to alternative programmable multipliers, making it a compelling choice. The remainder of this paper is structured as follows: Section II offers a comprehensive review of relevant literature, providing a contextual backdrop for our research. Organized into four key areas, we explore the design, implementation, and evaluation of inexact computing components. Focusing on inexact multipliers, 4:2 compressors, adjustable inexact computing, and performance measures, this discussion establishes the current landscape, setting the stage for our contributions in advancing the field of inexact computing. Section III presents our innovative approach to inexact multiplication. Section IV conducts a comprehensive comparison across multiple dimensions, including accuracy, delay, area, and power. Section V describes conclusions based on our findings and contributions.

2. Related work

Inexact multipliers can be created using various methods, including adjusting the supply voltage, omitting specific partial product rows, simplifying the multiplier equations, and using inexact compressors to reduce the number of partial product rows. In references [1] and [2], voltage scaling was used to control the supply voltage to the logic gates, effectively reducing power consumption. However, lowering the supply voltage below the nominal level could lead to timing violations, resulting in inexact outcomes. When timing violations occurred in critical pathways, the inaccuracies could be significant. In references [3] and [4], inexact multipliers were designed by truncating partial product columns near the least significant bit (LSB) column. This truncation aimed to reduce the propagation length for carry operations. Since partial products near the LSB column had smaller weights, omitting them did not introduce substantial errors. These methods demonstrate different strategies for constructing inexact multipliers, each optimizing power efficiency and minimizing errors in its way. Zendegani et al. [5] introduced an inexact multiplier by modifying the precise multiplication equation. This modification involved using rounded values of the inputs to create an inexact version, strategically employing values nearest to 2^n for approximation to optimize hardware efficiency and reduce costs. Wallace et al. [6] presented a high-speed multiplier architecture known as the Wallace tree multiplier. This design used accurate 2:2 compressors (half-adders) and 3:2 compressors (full adders) to efficiently reduce the number of partial product rows. Additionally, as explained in [7], a Wallace Tree Multiplier can incorporate precise 4:2 compressors to create a more structured layout. The combined effect of these compressors consolidates the partial products, which are then summed by a carry propagating adder to produce the final product. Due to its widespread adoption and utility, many studies have focused on developing inexact compressors derived from the accurate 4:2 compressor. Momeni et al. [8] introduced two types of inexact 4:2 compressors. The first, called Momeni acc, has five inputs and three outputs, closely resembling a conventional 4:2 compressor. The second, Momeni fast, has four inputs and two outputs, similar to other inexact 4:2 compressors. Yang and colleagues [9] modified the traditional 4:2 compressor to

propose three inexact versions: ACCI1, ACCI2, and ACCI3, focusing mainly on the equations for generating the sum bit. Ha et al. [3] improved upon Yang's ACCI3 by creating a simple error recovery circuit for the adapted compressor. ACCI3 experienced errors when both X3 and X4 equaled 1. Ha's modification involved altering the Boolean function of the carry to ensure a consistent error value of 1, simplifying error recovery. Lin and colleagues [10] made a notable alteration by replacing XOR gates with MUX, significantly reducing the delay of the inexact 4:2 compressor. Like Yang's ACCI1, errors occurred when X1, X2, X3, and X4 were all set to 1. However, Lin's compressor had an error distance of two, compared to one. Edavoor and colleagues [11] introduced a dual-stage 4:2 compressor with an error distance designed to be either positive or negative. They partitioned the partial product reduction into multiple stages and cascaded the dual-stage 4:2 compressor, managing errors at various stages. Sabetzadeh et al. [12] presented a majority-based 4:2 compressor, generating the carry bit with a 3-input majority gate and setting the sum bit consistently to 1. Strollo [13] used a stacking circuit technique that involved counting the number of 1's in the inputs to design the inexact 4:2 compressor. This stacking circuit efficiently transformed four inputs into three inputs, and a full adder was used to produce the sum and carry bits. Xiao and collaborators [14] considered the nonuniform data distribution in Convolutional Neural Networks (CNNs) activations and weights, which often follow Gaussian like distributions. Their work focused on achieving a balanced trade-off between latency, power utilization, and accuracy. These various inexact 4:2 compressors collectively offered a means to rapidly reduce partial products while delivering lower timing delays, reduced cell area, and decreased power utilization compared to the traditional 4:2 compressor. Akbari et. al [15] introduces four dual-quality 4:2 compressors for use in configurable parallel multipliers, which enhance speed and power efficiency at the cost of accuracy in approximate mode, showing significant delay and power reductions in a 32-bit Dadda multiplier and demonstrating their effectiveness in image processing applications.

3. Proposed Inexact Multiplier

This section initiates by highlighting the distinctions between the conventional multiplication flow and our innovative approach. We provide a detailed description on architecture of our proposed inexact multiplier, then delve into the specifics of our high-accuracy 4:2 compressor and an accompanying error compensation circuit. Following that, we explore the concept of adjustable input truncation, a crucial component in the assembly of our adjustable multiplier.

3.1 Outlined Methodology

In Fig. 1(a), we depict the conventional multiplication flow, a process designed for the generation of precise results. This method begins with the production of precise partial products via 2 input AND gates, which are subsequently subjected to compression by accurate compressors. Ultimately, the compressed partial products are summated by accurate adders to yield the final result.

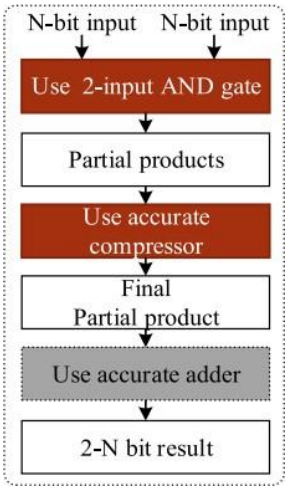

Fig. 1(a) Conventional multiplier flow

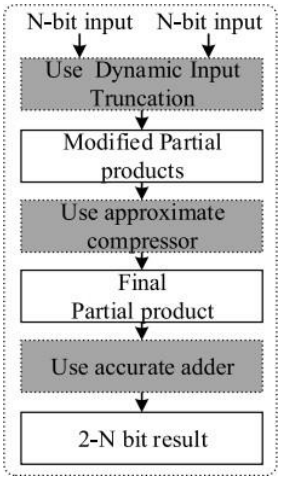

Fig. 1(b) Inexact multiplier flow

In contrast, Fig. 1(b) illustrates our proposed work flow for the inexact multipliers. The key distinctions between conventional multiplication and our proposed approach lie in the phases associated with generat- ing and compressing the partial products. These distinctions will be elaborated upon in Section 3.2. Within the phase of generating partial products, we employ adjustable input truncation, an essential concept to be introduced in Section 3.3, to produce modified partial products.

3.2 Proposed High-Accuracy Inexact 4:2 Compressor

In this research paper, we introduce a 4:2 inexact compressor with a focus on high accuracy and low power utilization, as depicted in Fig. 2. The design of this proposed 4:2 inexact compressor is elaborated upon as follows: We employ four inputs, labeled X_1 to X_4, to generate W_1 to W_4 using equations (1) to (4).

$$W_1 = X_1 \; AND \; X_2 \tag{1}$$

$$W_2 = X_1 \; OR \; X_2 \tag{2}$$

$$W_3 = X_3 \; AND \; X_4 \tag{3}$$

$$W_4 = X_3 \; OR \; X_4 \tag{4}$$

$$W_5 = W_1 \; OR \; W_3 \tag{5}$$

$$W_6 = W_2 \; AND \; X_4 \tag{6}$$

$$Carry = W_5 \; OR \; W_6 \tag{7}$$

Recognizing that an incorrectly computed carry bit introduces a higher error distance than the sum bit, we take measures to ensure the carry bit is consistently generated correctly. The equations for generating the carry bit are outlined in (5) to (7). The carry bit is designed to assume a value of 1 under three specific circumstances: one, when both X_1 and X_2 are set to 1; two, when both X_3 and X_4 are set to 1; and three, when either X_1 or X_2 is 1 and either X_3 or X_4 is 1. Equation (5) addresses the first two scenarios, Equation (6) addresses the third situation, and Equation (7) is responsible for producing the final carry bit.

X_3	X_3	X_2	X_1	carry	sum	diff.
0	0	0	0	0	0	0
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	1	0	0
0	1	0	0	0	1	0
0	1	0	1	1	0	0
0	1	1	0	1	0	0
0	1	1	1	1	1	0
1	0	0	0	0	1	0
1	0	0	1	1	0	0
1	0	1	0	1	0	0
1	0	1	1	1	1	0
1	1	0	0	1	0	0
1	1	0	1	1	1	0
1	1	1	0	1	1	0
1	1	1	1	1	1	-1

Table 1. Truth table of 4:2 Compressor

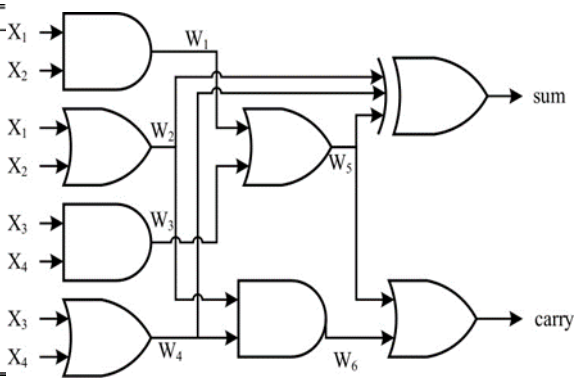

Fig. 2 Logic diagram of 4:2 compressor

The equation proposed for generating the sum bit is detailed in equation (8). In a precise 4:2 compressor, the sum bit is typically produced using four XOR gates within the two full adders. However, in our innovative compressor design, we generate the sum bit by inputting W_2 and W_4 into a 2 input XOR gate, making use of the signals also utilized for generating the carry bit. This shared utilization of signals serves to reduce both the circuit area and static power utilization. Nevertheless, we observed that the error distance tends to be significant when only W_2 and W_4 are fed into a 2 input XOR gate. This is primarily due to the fact that W_2 and W_4 are generated using OR gates, leading to errors in scenarios where both X_1 and X_2 or both X_3 and X_4 are set to 1. In such instances, the sum bit yields a result of 1 when it should be 0. To enhance accuracy, we introduce W_5, a signal used to identify these specific cases, into the XOR gate. For instance, if both X_1 and X_2 are 1, both W_2 and W_5 will assume a value of 1, resulting in the sum bit being '0 XOR W_4,' effectively producing W_4 as the sum bit. This allows us to narrow

down the consideration to just X_3 and X_4 in such cases. However, when all four inputs are set to 1, the sum bit indeed turns out as 1, resulting in an error distance of 1.

$$Sum = W_2\ XOR\ W_4\ XOR\ W_5 \quad (8)$$

In Table 1, we present the truth table for our proposed 4:2 compressor, highlighting a key characteristic: the potential for errors arises when all four inputs are set to 1. This eventuality is quite rare, as it only occurs when all four inputs align in this manner. To delve into the probabilities, we first note that the chance of a single input in the partial product being set to 1 stands at 1/4. This is because we consider the likelihood of a bit in both the multiplicand and multiplier simultaneously being 1, which is reproposed by or 1/4. Consequently, the probability of all four inputs coinciding as 1 is 1/16. In other words, the chances of this rare event transpiring are quite slim. Furthermore, even if such an error does occur, the discrepancy between the accurate output and the result produced by our compressor is merely 1. This difference is negligible in practical terms, especially when compared to the overall performance of the compressor.

$$Error = W_1\ AND\ W_3 \quad (9)$$

To facilitate error detection, a straightforward solution involves incorporating an additional AND gate within our system. This AND gate serve the purpose of ascertaining whether both W_1 and W_3 are set to 1. This design choice aligns with our existing configuration, where W_1 employs an AND gate to identify whether both X1 and X2 are 1, and similarly, W3 utilizes an AND gate to detect whether both X_3 and X_4 are 1. The error detection circuit (EDC) is succinctly reproposed in Equation (9). With this in place, integrating an error compensation circuit into our proposed 4:2 compressor becomes a straightforward task – it essentially entails the addition of an extra AND gate. This extra component not only simplifies the error detection process but also enhances the overall reliability and functionality of our compressor design. In order to achieve an adaptable inexact multiplier with the flexibility to adjust its behavior at run time, we introduce a adjustable input truncation technique. This method is illustrated in Fig. 3 and utilizes two 2-input AND gates to generate a partial product, as described by Equation (10), where 'A' represents the multiplicand and 'B' signifies the multiplier. The critical component in this process is the 'Trunc' signal, which serves to determine whether the partial product, denoted as PPD, should undergo truncation.

$$PPD_{ij} = Trunc\ AND\ B_i\ AND\ A_j \quad (10)$$

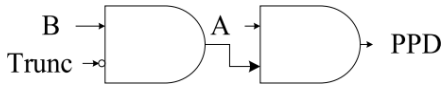

Fig. 3 Modified partial product

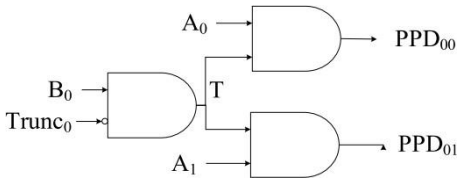

Fig. 4. Illustration of gate sharing

When the Truncating signal is set to 1, the partial product is effectively truncated to 0. In essence, the Truncating signals play a pivotal role in conserving power by strategically truncating the partial products to zero. In simpler terms, we can envision the Truncating signals as functioning to disable the hardware units within the corresponding columns, thus providing a method to economize power and tailor the multiplier's performance according to specific run time requirements.

3.3 Adjustable Input Truncation

In order to achieve an adaptable inexact multiplier with the flexibility to adjust its behavior at run time, we introduce a adjustable input truncation technique. This method is illustrated in Fig. 3 and utilizes two 2-input AND gates to generate a partial product, as described by Equation (8), where 'A' represents the multiplicand and 'B' signifies the multiplier. The critical component in this process is the Truncating signal, which serves to determine whether the partial product, denoted as PPD, should undergo truncation. When the Truncating signal is set to 1, the partial product is effectively truncated to 0. In essence, the Truncating signals play a pivotal role in conserving

power by strategically truncating the partial products to zero. In simpler terms, we can envision the Truncating signals as functioning to disable the hardware units within the corresponding columns, thus providing a method to economize power and tailor the multiplier's performance according to specific run time requirements. To optimize the hardware costs in an 8x8 multiplier, we've devised a strategy that involves sharing gates with an additional AND gate. In this approach, each bit of the multiplier corresponds to 8 bits of the multiplicand, allowing us to reduce the number of gates required. For instance, when computing PPD_0, it is equivalent to Trunc0 multiplied by B_0 and A_0, while PPD_{01} corresponds to $Trunc_0$ multiplied by B_0 and A_1. To achieve this, we pre-compute the product of $Trunc_0$ and B_0 to generate a mask, and then utilize three 2-input AND gates to complete the necessary multiplications. This approach is depicted in Fig. 4. The control of the Truncating signals in our proposed inexact multiplier will be discussed in greater detail in the following subsection, addressing how we manage and apply these signals effectively.

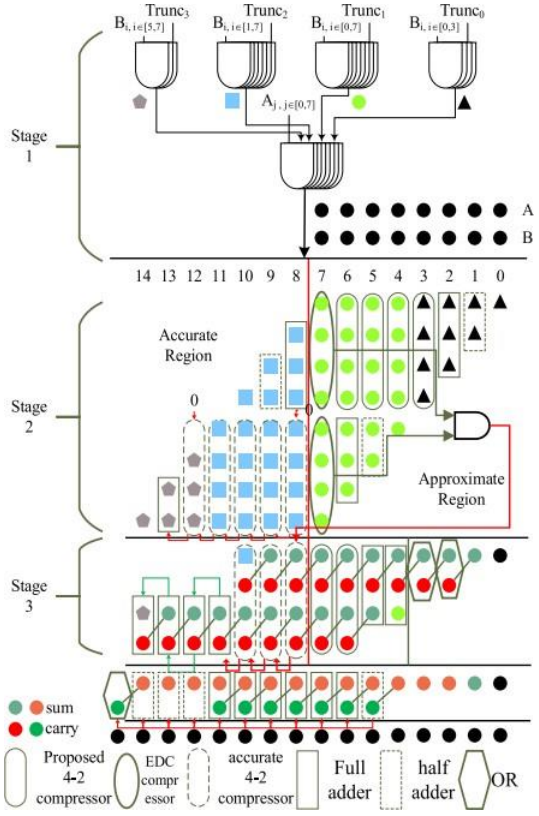

Fig. 5. Proposed inexact multiplier

3.4 Inexact Multiplier

In Fig. 5, we present an inexact multiplier employing the innovative technique. Although the multiplier's input width is initially designed for 8 bits, our approach can easily be extended to accommodate larger multipliers. This proposed inexact multiplier is structured into three distinct stages. In the first stage, each partial product is generated using two 2-input AND gates, as previously demonstrated in Fig. 3. To further economize on hardware resources, we incorporate the gate sharing technique shown in Fig. 4. The accuracy of these generated partial products is flexible and can be tailored to meet specific requirements, determined by the Truncating signal. In our innovative inexact multiplier, we've designed a 4-bit signal. Each bit of this signal controls multiple columns of partial products, a concept we refer to as the "3-4-4-4 partition". Specifically, each bit from the Most Significant Bit (MSB) to the Least Significant Bit (LSB) governs a range of columns, corresponding to khaki, sky blue, green, and black areas in Stage 2 of Fig. 4. For instance, if the Truncating (3:0) signal is set to 0101 in binary (or 2 in decimal), it signifies that columns 14^{th} to 12^{th} and 7^{th} to 4th are accurate, while columns 11^{th} to 8^{th} and 3^{rd} to 0^{th} are truncated. This approach optimizes control efficiency and minimizes hardware costs. The flexibility of the

Truncating signal control lies in how we partition the columns, allowing users to adapt the proposed multiplier to their specific needs. Through various experiments, we explored different partitioning options. Our findings revealed that both the ”3-4-4-4 partition” and the ”3-3-3-3-3 partition” strike a balance between power savings, accuracy, and area overhead. Finer partitions provide greater control over power savings and accuracy adjustment but come at the cost of increased area overhead. As a result, the ”3-4-4-4 partition” was consistently employed in our experiments. In Section V. In the second stage of the multiplier, we divide the generated partial products into two regions. Columns 14^{th} to 8^{th} are designated as the accurate region, while columns 7^{th} to 0^{th} form the inexact region. The decision to split these regions, typically half and half, is made based on a straightforward and intuitive approach. For instance, a 30-70 split would lead to significant accuracy loss due to excessive computations by the inexact multiplier, while a 70-30 split would have minimal impact on power reduction. Since the partial products in the accurate region carry greater weight and importance, we employ accurate 4:2 compressors to compress them. In the inexact region, we utilize our proposed inexact 4:2 compressors in conjunction with an error compensation circuit. In the third stage, we employ OR gates in columns 3^{rd} to 0^{th} to generate results while disregarding carry propagation, as these columns are closer to the Least Significant Bit (LSB) and their errors have a lesser impact on the final results. To detect errors in the second stage, we utilize the Error Detection Circuit (EDC), which consists of a single AND gate. This helps us determine whether a compensation bit should be generated. The remaining columns are processed using a combination of our proposed inexact 4:2 compressors, accurate 4:2 compressors, full adders, and half adders. Upon completion of the third stage, we obtain the final two rows of partial products, which are summed up using accurate adders to produce the ultimate results.

4. Experimental Frame Work and Results

In this section, we begin by presenting the experimental setup employed for assessing the performance of the inexact multipliers. Subsequently, we conduct a comparative analysis of these multipliers across various evaluation metrics. Our evaluation encompasses critical path delay, cell area, power utilization, and the power-delay product (PDP).

4.1 Experimental Frame Work

In the experimentation phase, the RTL (Register Transfer Level) codes of the inexact multipliers were implemented using Verilog Hardware Description Language (HDL). NCSim was utilized to simulate these inexact multipliers and generate waveforms, which allowed us to record the switching activities of logic gates. For synthesis, we employed Design Compiler to convert the RTL codes into gate-level netlists, utilizing the standard UMC 0.18micro meter CMOS cell library. The power utilization estimation, based on the generated waveforms, was carried out using Prime Time PX. It’s worth noting that all the inexact multipliers underwent synthesis and optimization using identical settings. Our comparison encompasses the proposed inexact multipliers, the accurate Wallace Tree multiplier [6], and previous inexact multipliers [3], [8], [9], [13], [16]. From the work of Momeni et al. [8], we selected two inexact multiplier designs:” Momeni fast,” known for its higher speed but lower accuracy due to the use of all inexact 4:2 compressors, and” Momeni acc,” which achieves higher accuracy by utilizing exact 4:2 compressors in the Most Significant Bit (MSB) columns and inexact 4:2 compressors in the Least Significant Bit (LSB) columns. Yang et al. [9] introduced three different inexact 4:2 compressors to construct various Dadda multipliers, known as “Yang high”, “Yang medium” and “Yang low.” These designations directly reflect the accuracy levels, with Yang high being the most accurate. Ha et al. [3] proposed an inexact multiplier featuring error correction circuits, referred to as “Ha”. We also considered the “Yang adj” inexact multiplier design [16], known for its adjustable accuracy. Additionally, Strollo et al. [13] proposed two inexact multipliers, one exclusively using inexact 4:2 compressors and the other, “Strollo acc” incorporating a mix of inexact and accurate compressors for enhanced accuracy. Our proposed inexact multiplier, as illustrated in Fig. 5, incorporates high-accuracy 4:2 compressors, a straightforward error compensation circuit, and adjustable input truncation. With a 4-bit width Trunc signal, there are sixteen possible configurations, denoted as “proposed 0000” to “proposed 1111”. However, for our comparison, we focus on three configurations, “proposed 0000”, “proposed 0001”, and “proposed 0011. These are selected for their enhanced accuracy, stemming from the absence of truncation in the eight leftmost partial product columns.

4.2 Accuracy Comparison

Table 2 offers a comparison of the accuracy metrics of various inexact multipliers obtained from different studies [3], [8], [9], [13], [16]. To ensure the reliability of the results, we used the same experimental data as provided in each respective work. In the table, we've highlighted the best results in blue, the second best in green, and the third best in yellow. In terms of error rate (ER), our proposed multiplier, "Proposed 0000" exhibits the second-lowest error rate at 11.57, following closely behind the 9.29 ER of "Strollo acc". The relatively low error rate of our proposed inexact multiplier can be attributed to the use of highly accurate 4:2 inexact compressors in the Least Significant Bit (LSB) columns, accurate 4:2 compressors in the Most Significant Bit (MSB) columns, and the incorporation of an error compensation circuit to minimize errors.

Table 2. Comparative analysis of various accuracy measures

Design	ER (%)	MED	RED	MRED	WED	NMED
Momeni_fast [8]	99.3	3518	278284	4.246	8640	5.4*10^-2
Momeni_acc [8]	85.77	51.43	2918	0.044	200	7.9*10^-4
Yang_high [9]	81.29	15.33	382.14	0.005	561	2.3*10^-4
Yang_medium [9]	83.47	25.73	503.82	0.007	561	3.9*10^-4
Yang_low [9]	83.74	35.09	556.46	0.008	577	5.4*10^-4
Ha [3]	82.91	28.05	514.25	0.008	385	4.3*10^-4
Yang_adj [16]	36.16	164.47	555.23	0.008	573	5.2*10^-3
Strollo_acc [13]	9.29	8.46	50.122	0.0007	536	1.3*10^-4
Proposed_0000	11.57	3.957	33.59	0.0005	452	6.1*10^-5
Proposed_0001	81.29	15.39	381	0.0058	481	2.3*10^-4

As mentioned in Section 3.4, our approach sacrifices minimal accuracy for efficiency by allowing columns 3rd to 0th in the partial products, which generate results with OR gates. This feature may account for the narrow 2.3 margin when com- pared to "Strollo acc". Concerning metrics like Mean Error Distance (MED), Normalized Mean Error Distance (NMED), (Relative Error Distance) RED and Mean Relative Error Distance (MRED), "Proposed 0000" stands out among the other designs. In the case of Worst Error Distance (WED), "Proposed 0000" exhibits the smallest value, except for "Momeni acc" and "Ha". However, the noteworthy point is that "Proposed 0000" outperforms "Momeni acc" and "Ha" in MED and MRED, indicating that the worst error distance in our proposed multiplier occurs less frequently. Consequently, the MED and MRED values are lower than those of "Ha" and "Momeni acc". In Fig. 5, we delve deeper into the comparison of error distance distributions for our proposed multiplier, "Proposed 0000" as well as "Momeni acc", "Ha," and "Strollo acc". On the x-axis, we represent the error distance, while the y-axis displays the number of occurrences. Notably, the errors generated by the proposed multiplier tend to have smaller error distances and occur less frequently when contrasted with "Momeni acc" and 'Ha." This reaffirms our earlier point that, despite having a larger WED than "Ha" and "Momeni acc" the MED and MRED of our proposed multiplier surpass those of "Ha", "Momeni acc" and "Strollo acc". Furthermore, the majority of errors in our proposed multiplier exhibit smaller Error Distances

(ED) compared to those in "Strollo acc". This clarifies why, even though the Error Rate (ER) of the proposed multiplier is slightly higher than that of "Strollo acc" the MED in our proposed multiplier is smaller than "Strollo acc". As for "Proposed 0001" it performs quite comparably to "Yang high" in terms of overall accuracy metrics. Consequently, there is little need for a direct comparison with "Yang medium" and "Yang low." Its demonstrates better overall results than "Ha" and "Momeni acc". "Proposed 0011," on the other hand, truncates more than half of the bits, resulting in lower accuracy. However, despite its modest accuracy, "Proposed 0011" significantly reduces power utilization, as detailed in Table 3.

4.3 Latency, Area and Power Comparison

Table 3 provides an overview of latency, area, and power assessments for both accurate and inexact multipliers. The accurate multiplier is constructed using an 8x8 Wallace Tree Multiplier with precise compressors. The proposed multipliers are evaluated alongside multipliers with similar results as shown in Table 2. Our proposed multiplier exhibits a substantial reduction in latency, slashing it by 27%, and an area overhead reduction of 7% compared to the accurate multiplier.

Table 3. Latency, Area and Power assessments

Design	Latency	Area	Power	PDP
Exact Design	100%	100%	100%	100%
Yang_adj [12]	81%	90%	84%	68%
Yang_high [15]	76%	86%	86%	66%
Strollo_acc [7]	70%	90%	92%	64%
Proposed_0000	73%	93%	82%	**60%**
Proposed_0001	73%	93%	69%	**51%**
Proposed_0011	73%	93%	28%	**21%**

Additionally, it demonstrates notable power savings. "Proposed 0011" achieves the highest power reduc- tion ratio, followed by "Proposed 0001", "Proposed 0000", "Yang adj", "Yang high," and "Strollo acc". The reduction in power utilization is particularly striking for "Proposed 0011" reaching up to 72%, which signifies a remarkable degree of power conservation. "Proposed 0001" and "Proposed 0000" also achieve substantial power reductions, with 31% and 18%, respectively. Overall, the proposed multiplier averages a 40% reduction in power utilization. Users can select different configurations of our design to optimize for accuracy or energy efficiency, depending on their specific requirements. The Power Delay Product (PDP) is a comprehensive metric that takes into account both power dissipation and propagation delay. It provides a holistic view of system performance, balancing energy consumption and timing delay. "Proposed 0011" stands out with the lowest PDP, indicating superior balance between power utilization and timing delay. Achieving low delay and energy consumption while maintaining good accuracy is a challenging trade-off. It's important to note that multipliers with lower accuracy tend to have lower energy consumption. Considering this trade-off, "Proposed 0000" demonstrates commendable accuracy compared to other designs and still manages to achieve an 8 percent and 4 percent reduction in PDP compared to "Yang adj" and "Strollo acc", "Proposed 0001" offers similar accuracy to "Yang high" while surpassing it by up to 15 percent in PDP reduction.

4.4 Area Overhead Comparison

Table 4 presents a comparison of the multiplier area between a programmable truncation approach proposed in [4] and our proposed multiplier utilizing adjustable input truncation technique. The area overhead of reconfigurability elements is outlined concerning our proposed inexact multiplier without reconfigurability, which has an initial area of 4587.

Table 4. Comparative Analysis of Area overheads

Original Area of Approximate Multiplier	Area Overhead for Truncation in [4]	Area Overhead for the Proposed Truncation
4587	487	206 (-42%)

In specific terms, when integrating reconfigurability components like Truncating and gate sharing circuit into our proposed inexact multiplier, an additional 206 units of area are required. Conversely, if the programmable truncation circuit proposed in [12] is implemented on our proposed multiplier, the additional area overhead increases to 487 units. Notably, the area overhead attributed to reconfigurability elements in our proposed multiplier is considerably reduced by 42 percent while maintaining the same level of accuracy. In [4], one partial product column is controlled at a time, whereas our proposed inexact multiplier allows one Truncating bit to control multiple partial product columns through adjustable input truncation. This highlights that adjustable input truncation significantly mitigates the extra hardware cost associated with implementing an adjustable design.

5. Conclusion

In conclusion, this paper introduces an innovative high precision approximate 4:2 compressor, which serves as a key component for constructing an adjustable approximate multiplier. Our proposed approximate multiplier dynamically adjusts accuracy by truncating partial products, and it incorporates a straightforward error compensation circuit to minimize error. Compared to the Wallace tree multiplier, our adjustable approximate multiplier demonstrates substantial improvements. It achieves a 27% reduction in delay and a remarkable 72% reduction in power consumption and a noteworthy 7% decrease in area requirements. These efficiency gains are significant, making our solution highly competitive. Furthermore, when compared to other existing approximate multipliers, our proposed design stands out with the lowest mean error distance and the lowest average power consumption. This highlights the superior performance and efficiency of our innovative approach in achieving high accuracy while reducing computational delay and power consumption.

Refrences

[1] Bert Moons and Marian Verhelst. Dvas: Dynamic voltage accuracy scaling for increased energy-efficiency in approximate computing. In 2015 IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED), pages 237–242. IEEE, 2015.

[2] Debabrata Mohapatra, Vinay K Chippa, Anand Raghunathan, and Kaushik Roy. Design of voltage-scalable meta-functions for approximate computing. In 2011 Design, Automation & Test in Europe, pages 1–6. IEEE, 2011.

[3] Khaing Yin Kyaw, Wang Ling Goh, and Kiat Seng Yeo. Low power high speed multiplier for error-tolerant application. In 2010 IEEE international conference of electron devices and solid-state circuits (EDSSC), pages 1–4. IEEE, 2010.

[4] Manuel de la Guia Solaz, Wei Han, and Richard Conway. A flexible low power dsp with a programmable truncated multiplier. IEEE Transactions on Circuits and Systems I: Regular Papers, 59(11):2555–2568, 2012.

[5] Reza Zendegani, Mehdi Kamal, Milad Bahadori, Ali Afzali-Kusha, and Massoud 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, 25(2):393–401, 2016.

[6] Christopher S Wallace. A suggestion for a fast multiplier. IEEE Transactions on electronic Computers, (1):14–17, 1964.

[7] A Weinberger. 4-2 carry-save adder module. IBM technical disclosure bulletin, 23(8):3811–3814, 1981.
[8] Amir Momeni, Jie Han, Paolo Montuschi, and Fabrizio Lombardi. Design and analysis of approximate compressors for multiplication. IEEE Transactions on Computers, 64(4):984–994, 2014.
[9] Zhixi Yang, Jie Han, and Fabrizio Lombardi. Approximate compressors for error-resilient multiplier design. In 2015 IEEE international symposium on defect and fault tolerance in VLSI and nanotechnology systems (DFTS), pages 183–186. IEEE, 2015.
[10] Chia Hao Lin and Chao Lin. High accuracy approximate multiplier with error correction. In 2013 IEEE 31st international conference on computer design (ICCD), pages 33–38. IEEE, 2013.
[11] Pranose J Edavoor, Sithara Raveendran, and Amol D Rahulkar. Approximate multiplier design using novel dual-stage 4: 2 compressors. IEEE Access, 8:48337–48351, 2020.
[12] Farnaz Sabetzadeh, Mohammad Hossein Moaiyeri, and Mohammad Ahmadinejad. A majority based imprecise multiplier for ultra efficient approximate image multiplication. IEEE Transactions on Circuits and Systems I: Regular Papers, 66(11):4200–4208, 2019.
[13] Antonio Giuseppe Maria Strollo, Ettore Napoli, Davide De Caro, Nicola Petra, and Gennaro Di Meo. Comparison and extension of approximate 4-2 compressors for low power approximate multipliers. IEEE Trans- actions on Circuits and Systems I: Regular Papers, 67(9):3021–3034, 2020.
[14] Hang Xiao, Haobo Xu, Xiaoming Chen, Yujie Wang, and Yinhe Han. Fast and high accuracy approximate mac unit design for CNN computing. IEEE Embedded Systems Letters, 14(3):155–158, 2021.
[15] Omid Akbari, Mehdi Kamal, Ali Afzali-Kusha, and Massoud Pedram. Dual-quality 4: 2 compressors for utilizing in dynamic accuracy configurable multipliers. IEEE Transactions on Very Large-Scale Integration (VLSI) Systems, 25(4):1352–1361, 2017.
[16] Tongxin Yang, Tomoaki Ukezono, and Toshinori Sato. A low power high speed accuracy controllable approximate multiplier design. In 2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC), pages 605–610. IEEE, 2018.