Journal of Circuits, Systems, and Computers

World Scientific
www.worldscientific.com

Implementation of Energy Efficient Approximate Computing via Recursive Multiplier in Error Tolerant Applications*

Karthikkumar M †

Assistant Professor, Department of Electronics and Communication Engineering, Erode Sengunthar Engineering College, Thuduppathi, Tamil Nadu 638057 India ‡
karthikkumarM554@outlook.com

Kalaivani R

Professor, Department of Electronics and Communication Engineering, Erode Sengunthar Engineering College, Thuduppathi, Tamil Nadu 638057 India
kalaivani765@gmail.com

Received (Day Month Year)
Revised (Day Month Year)
Accepted (Day Month Year)
Published (Day Month Year)

Approximate computing (AC) in arithmetic logic has become a viable option in applications requiring error tolerance in energy-efficient architectures. AC relied on approximate arithmetic functions to reduce delay, area, and power consumption while sacrificing accuracy to reduce delay, power, and area. When multipliers are used in AC systems, they are responsible for performing calculations and consume a great deal of power. In this research, a novel approximate recursive multiplier (RMul-1, RMul-2), and approximate adder has been proposed to reduce complexity and power consumption. The proposed approximate recursive multiplier (RMul-1, RMult-2) are designed using NOR, AND, Full Adder, and Half Adder for low power design, and area efficiency. Moreover, the proposed power efficiency approximate adder is designed using AND, OR, and MUX for low delay. In image processing applications, the proposed approximate RMul-1 and RMul-2 are quite effective. The Xilinx ISE 13.2 tool has been used to simulate all proposed designs and current multipliers. In comparison to the accurate and current approximation designs, the proposed concept drastically decreases area, power, and delay. The proposed RMul-1 and RMult-2 have the potential to decrease power consumption by 2.5% and 3.75% in comparison to the existing MLAFAs and DSM. In addition, the proposed approximate adder produces comparatively low results when compared to a normal adder, with a delay of 0.765 ns, power of 0.101 W, and area of 2 LUTs. Furthermore, compared to existing approximation multipliers and adders, the proposed approach has demonstrated superior performance.

Keywords: Approximate computing; approximate adder; approximate recursive multiplier; power consumption.

* Implementation of Energy Efficient Approximate Computing via Recursive Multiplier in Error Tolerant Applications.
† karthikkumarM554@outlook.com.
‡ Assistant Professor, Department of Electronics and Communication Engineering, Erode Sengunthar Engineering College, Thuduppathi, Tamil Nadu 638057 India.

1. Introduction

Energy efficiency and on-chip area have emerged as the two main design limitations for hardware in recent years [1]. In a number of newly emerging applications across multiple domains, approximate computing can be extremely helpful in reducing power, area, and computational delay. Generally speaking, approximate computation is used in any applications that can handle results with limited errors [2]. With applications ranging from multimedia processing to machine learning, approximate computing is developing as a promising technology that achieves these aims at the expense of the output' arithmetic correctness [3]. Both logic gates and transistors can be used as implementation levels for approximations.

Approximate computing can improve power and performance in these circuits because of the high multiplier complexity [4]. Because it is essential to many computing applications, including DSP and machine learning algorithms, the design of approximate arithmetic units has drawn a lot of attention at the circuit level. These applications often involve a lot of addition (or accumulation) and multiplication in their arithmetic processing. Due to their fundamental nature as arithmetic processor operations, addition and multiplication have received a great deal of research attention in with the goal of reducing power consumption and obtaining high performance.

In order to reduce power consumption and delay, approximate adders have been thoroughly investigated in technical literature. Compared to addition, multiplication is a more difficult process. A common multiplier can have approximate designs for its operands, compressors, PP trees, and partial product (PP) production. Accuracy can be sacrificed for improved performance and reduced power consumption with an approximation multiplier. In fact, energy efficiency is now a top priority in the development of portable electronics, outweighing other factors [6-8]. Consequently, a rational trade-off between hardware efficiency and precision is generated in the design of approximation circuits, which is highly effective in minimizing circuit energy loss in image processing applications [9-14]. Several authors have used 4 x 4 approximation multipliers to construct various 8 × 8 multiplier options recursively [15-17]. One of the primary challenges faced by approximate designs is minimizing size and latency while avoiding substantial errors. In this research, a novel approximate recursive multiplier (RMul-1, RMul-2), and approximate adder has been proposed to reduce complexity and energy usage. The following are the main contributions of the proposed methods are

- The main aim of the study is to introduce an approximate multiplier and adder to enhance the effectiveness and energy efficiency.
- The proposed approximate recursive multiplier (RMul-1, RMult-2) are designed using NOR, AND, Half Adder, and Full Adder for low power design, and area efficiency.
- Moreover, the proposed power efficiency approximate adder is designed using AND, OR, and MUX for low delay.
- The proposed approximate RMul-1 and RMul-2 are very efficient in image processing applications.

- The efficiency of all proposed designs and exsiting multipliers has been estimated with respect to area, power, and delay through simulation using the Xilinx ISE 13.2 tool.

This research paper is structured as follows. Section 2 analyses the study based on approximate computing, approximate multiplication, and recursive multipliers. Section 3 provides the background study based on Half Adder (HA), and Full Adder (FA). Section 4 provides a detail description of the proposed approximate recursive multipliers 1, approximate recursive multipliers 2, and approximate adder. Section 5 provides a detail description on Image smoothening, and image sharpening in approximate recursive multiplier. Section 6 provides the conclusion.

2. Literature Survey

This section provides the different approximate computing, approximate multiplication, and recursive multipliers are considered in this paper. In recent years the performance of a device is still a research topic for many scholars. Some of those methods are briefly discussed in this section.

In 2020, Jothin, et al. [18] proposed the use of CEETAs (Compact Energy-efficient Error-Tolerant Adders) as an economical design metric for data-intensive applications. The suggested adder shows less Power-Delay Product (PDP), less Area-Delay Product (ADP), and low power consumption when compared to the traditional ETCSLA, SAET-CSLA, CSLA, HSETA, HSSSA, HPETA-II, and HPETA-I. Moreover, the proposed method provides the savings of 51.63%, 43.87%, 48.57%, 36.52%, 36.84%, 15.72%, and 18.18% for area compared to existing deigns.

In 2021, Pandey, et al [19] suggested a novel multiply two unsigned binary values using an approximate multiplier based on segmentation (SAM). By employing a smaller partial product matrix (R-PPM) of level 4 × 2n, the size of the partial products matrix (PPM) of order n × (2n — 1) is decreased. The basic design yields an output that is 89.1% more accurate than other state-of-the-art designs like TOSAM, LETAM, and DQ4:2C4 and uses 32.43% less on-chip area than the traditional Wallace tree multiplier.

In 2020, Van Toan, Lee, [20] suggested an approximation multiplier that may be effectively used on Field Programmable Gate Arrays (FPGAs) at varying accuracy levels by utilising recently designed approximate logic compressors. A peak signal-to-noise ratio (PSNR) of 46.81 dB, SSIM of 0.9989, and a dynamic power reduction of 36.7% have also been demonstrated with 8 x 8 multipliers. A 16 x 16 multiplier can achieve a PSNR of 80.25 dB, a SSIM of 1.0, and a power savings of up to 58.15% in terms of dynamic power savings.

In 2019, Van Toan, Lee, [21] suggested a roughly multiplier architecture that works well with Field Programmable Gate Arrays (FPGAs). With same accuracy, their approximation multipliers provide larger gains in energy-area products than the existing methods. Experiments demonstrate that their proposed multiplier can achieve a high PSNR of 45.34 dB and save up to 45.0% of power usage when compared to the exact IP multiplier.

In 2022, Zacharelos, et al. [22] suggested two novel approximate multipliers of 4x4 that were produced by carry manipulation. Various error-performance trade-offs are achieved by combining them to create 8x8 designs. When compared to an identical 8x8 multiplier, the design proposed in this study has the lowest dissipative power consumption, silicon area, and shortest latency, at 46% and 21%, respectively. In comparison to the least power-hungry recursive circuit published in the literature, it provides 81% higher accuracy while consuming 14% less power.

In 2021, Waris, et al. [23] suggested a novel two approximate 2x2 multipliers with a double-sided error distribution feature. Compared with the prior approximated 2x2 multiplier, the developed research, with bounded error behaviour, delivers a 52% reduction in area and demonstrates a 25% improvement in delay. Modern error-tolerant applications like convolutional neural networks make use of the recommended approximation multipliers. AxRM2, which yields 32.64% power savings and 1.10% greater accuracy.

In 2019, Darshini, et al. [24] suggested a novel concept is put forth that makes use of partial product partitions and recursive multiplication to create approximative multipliers. Experimental analysis depicts that the that the developed design achieves significant power and latency reductions, and performs more accurately than previous approximations.

In 2022, Di Meo, G., et al. [25] suggested a novel floating-point multiplier with low power approximation. They use a special approximation method based on static segmentation to mantissa multiplication because most of the power consumption is in the mantissa computation. When using single-precision floating-point format, the resultant NMED falls between 10−5-7 × 10−7, and the MRED falls between 3 × 10−3–1.7 × 10−4. Synthesis yields up to 82% and 85% power and 28 nm CMOS display area savings, respectively, over the accurate floating-point multiplier.

In 2022, Nunziata, I., et al. [26] suggested a recursive multiplier that utilises distinct 4x4 multiplier pieces as a basic foundation. By using carry truncation and error compensation, they were able to get three estimated 4x4 multipliers, each with a distinct error-precision trade-off. These fundamental building elements can be used to create 8x8 approximation multipliers. Performance advantages over the state-of-the-art are achieved by the suggested circuits, which are built using 14 nm FinFET technology.

In 2017, Mazahir, S., et al. [27] suggested a error probability analysis for approximation multipliers that recursively have approximate partial products. The outcomes demonstrate how useful the suggested methodology is as a tool for assessing, comparing, and forecasting the accuracy of different multipliers. The findings demonstrate that they obtain reliable error performance evaluation for most recursive multipliers.

Several related studies have been conducted in order to reduce the area, power, and delay in multiplication. This paper proposed a novel approximate recursive multiplier (RMul-1, RMul-2), and approximate adder has been proposed which number of logic gates to reduce power, increase speed.

3. Background Study

3.1. *Half Adder using NOR gate*

The half adder is the most fundamental adder circuit. A half adder is a type of combinational arithmetic circuit that generates SUM and CARRY when two numbers are added. He suggested a half adder that uses a nor gate in Figure 2. It consists of two inputs, and two outputs and the output are calculated using eq (1) and (2).

$$SUM = A \oplus B \quad (1)$$

$$CARRY = A * B \quad (2)$$

Table 1. Truth Table for HA.

INPUT		OUTPUT	
A	**B**	**SUM**	**CARRY**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

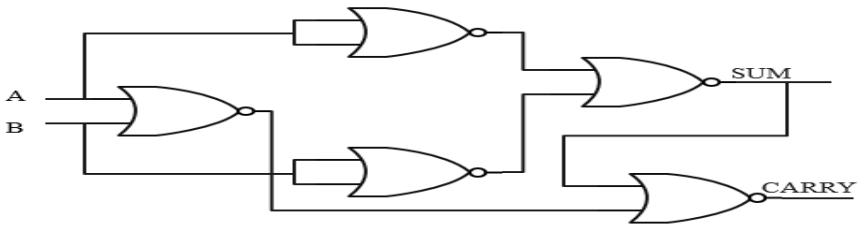


Fig. 1. Circuit diagram for half adder

From figure 1, the SUM output is produced by XORing the two inputs, A and B. The XOR gate adds binary numbers by performing addition operations only when one of its inputs is 1. Only then does a "1" show up in the SUM output of the gate. The CARRY output is the most significant bit (MSB) of the result. indicating the existence of a CARRY over following the addition of the two inputs. The outcome of ANDing inputs A and B is CARRY. Only when both inputs are 1 does the AND gate's CARRY output result in a "1".

3.2. *Full Adder using NOR gate*

A full adder is an adder that produces two outputs after adding three input values. Two half adders are combined to form a full adder. The suggested full adder with nor gate is seen in Figure 3. A complete adder has two outputs, SUM and CARRY, and three inputs, A, B, and C and the output is calculated using (3) and (4).

$$SUM = A \oplus B \oplus C \quad (3)$$

$$CARR = C(A \oplus B) + AB \quad (4)$$

Table 2. Truth Table for FA.

INPUT			**OUTPUT**	
A	**B**	**C**	**SUM**	**CARRY**
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

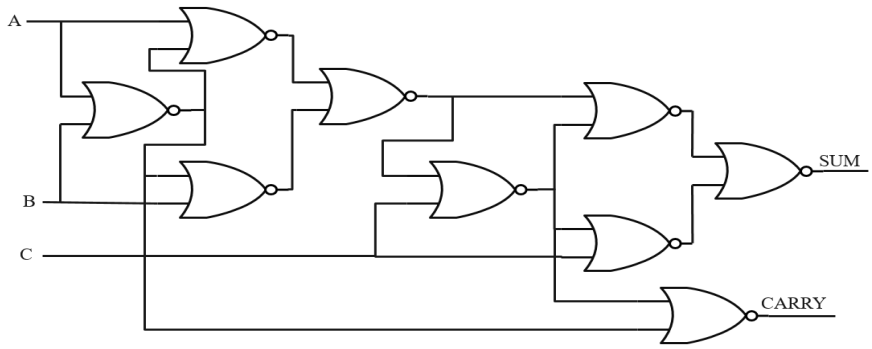


Fig. 2. Circuit diagram for FA

From figure 2, the SUM output is produced by XORing the two inputs, A and B. It applies the XOR operation on the result using CARRY. The CARRY output is the most significant bit (MSB) of the result. When inputs A and B are ANDed, the result is CARRY. Apply the XOR function on inputs A and B. Execute the OR function on the two outputs obtained from the preceding step. The whole adder utilising NOR gate truth table is displayed in Table 2.

4. Proposed Methodology

In this research, a novel approximate recursive multiplier (RMul-1, RMul-2), and approximate adder has been proposed to reduce complexity and power consumption. The proposed approximate recursive multiplier (RMul-1, RMult-2) are designed using NOR, AND, Half Adder, and Full Adder for low power design, and area efficiency. Moreover, the proposed power efficiency approximate adder is designed using AND, OR, and MUX for low delay. The proposed approximate RMul-1 and RMul-2 are very efficient in image processing applications.

4.1. *Proposed approximate recursive multiplier 1 (RMul-1)*

The proposed approximate 8 x 8 recursive multiplier which consist of two inputs namely X and Y X is the multiplier and Y is the multiplicand. In step 1 multiplier is divided into two parts namely A and C where A is the MSB where MSB is represented in green color and C is the LSB where LSB is represented in blur color. Multiplicand is divided into two parts namely B and D where B is the MSB where MSB is represented in green color and D is the LSB where LSB is represented in blur color. In step 2 multiplication and addition (half adder and full adder) operations were perform. In step 3 concatenation operation will perform from O [0] to O [7] and P [0] to P [7] will be the approximate multiplication value for the 8 x 8 multiplier. The circuit diagram for proposed approximation RMul-1 is shown in figure 3.

4.2. *Proposed recursive multiplier 2 (RMul-2)*

The proposed a novel 8 x 8 recursive multiplier-2 which consist of two inputs namely X and Y X is the multiplier and Y is the multiplicand. In step 1 multiplier is divided into two parts namely X1, and X2 where X1 is the first half of X and X2 is the second half of X. And Y is divided into two parts namely Y1, and Y1 where Y1 is the first half of Y and Y2 is the second half of Y. In step 2 X1 is divided into two parts namely A and C. X2 is divided into two parts namely E and G. Y1 is divided into two parts namely B and D. And Y2 is divided into two parts namely F and H. In step 3 multiplication and addition (half adder and full adder) operation were perform. In step 4 the product of A and B produce a 4bit output namely T [0], T [1], T [2], and T [3] where A and B were represented in purple color. The circuit diagram for proposed approximation RMul-2 is depicted in Figure 4.

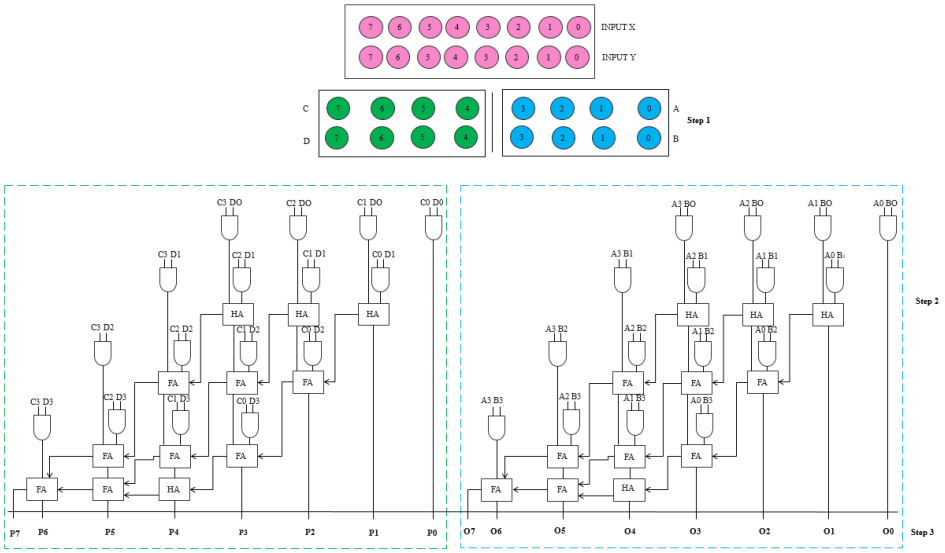


Fig. 3. Circuit for proposed recursive multiplier 1

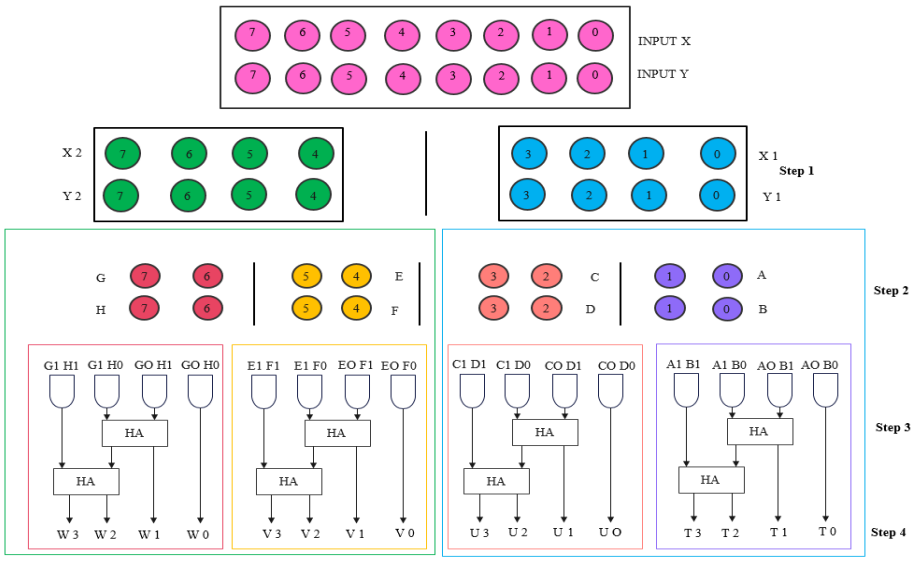


Fig. 4. Circuit for proposed recursive multiplier 2

The product of C and D produce a 4bit output namely U [0], U [1], U [2], and U [3] where C and D were represented in orange color. The product of E and F produce a 4bit output namely V [0], V [1], V [2], and V [3] where E and F were represented in yellow color. The product of G and H produce a 4bit output namely W [0], W [1], W [2], and W [3] where G and H were represented in red color. In step 4 concatenation of T [0] to T [3], U [0] to U [3], V [0] to V [3], and W [0] to W [3] will be the approximate multiplication value of the given 8 x 8 multiplier.

4.2.1. *Concatenation operation*

Concatenation is a process of complaining two are more values to form a single value. Concatenation is expressed as operands within {,} brackets with commas separating the operands. The size of the operands must be declared; unsigned operands are not allowed because the size of each operand must know for computing of the size of the result. Operation can be scalar net or registers, vector nest or register, bit select, part select or sized constant. As shown in the figure () the proposed recursive multiplier 1 which has two eight-bit output. The two eight-bit output is concatenated as follows. Initially the first eight-bit output are taken they are O [0] to O [7] and the second eight-bit output are taken they are P [0] to P [7]. The concatenation operation will first calculate the number of bits that is 16 followed by the value from O [0] to O [7] and P [0] to P [7]. Example for 8 x 8 multiplication output will be:16′b O [0], O [1], O [2], O [3], O [4], O [5], O [6], O [7], P [0], P [1], P [2], P [3], P [4], P [5], P [6], and P [7]. Where 16′b – represent the total number of bits. And O [0] to O [7] and P [0] to P [7] – represents the output values.

4.3. *Proposed approximate adder*

Initially the adder divides the input values into two parts namely approximate part and acute part. In which the MSB part of the number is accurate part and the MSB part are approximate part. Figure 5 shows the circuit diagram for proposed approximate adder. We

presented a new approximate adder in this work, which is divided into two parts: an approximate component and an accurate part. As seen in the illustration, the accurate portion is highlighted in yellow, while the approximate part is indicated in pink. The exact part, given an adder of size N bits and an approximation part of M bits, would be (N-M) bits. Concatenating the M sum bits from the approximation portion with the (N-M+1) sum bits from the accurate section results in the (N+1) adder's sum bits.

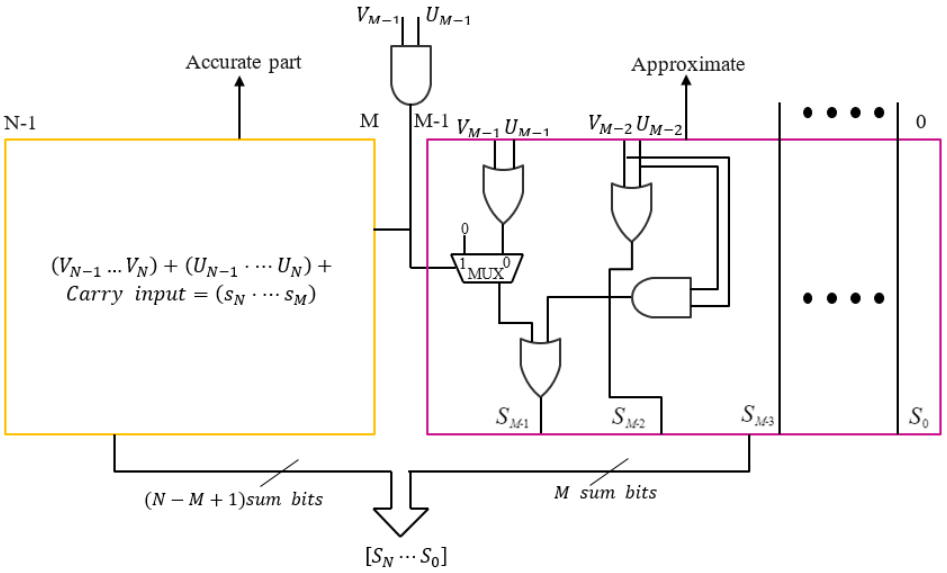


Fig. 5. Circuit diagram for proposed approximate adder

The adder output circuit diagram is shown as S in Fig. 5, and the adder inputs are represented by V and U. The least and most important bits of the adder inputs and outputs are indicated by the subscripts 0 and N, respectively. Except for the two most important sum bits, all other remaining sum bits in the approximation region are fixed at 1. The second-most important sum bit for the error component is obtained by ORing the matched input bit pair. The pair of bits AND represents the most significant input of the incorrect component. It acts as the choice input of a MUX built to handle the greatest sum bit of the erroneous section.

5. Results and Discussion

In this section, the proposed approximation RMul-1, RMul-2 and approximate adder is compared with existing designs and simulated in Xilinx Ise 14.2 tool in FPGA hardware (Artix7) based on 28nm technology. In order to calculated the overall area occupied by the design is determined by look up table count from device utilization report. The timing analyses is performed by using combinational path delay.

5.1. *Performance evaluation*

The efficiency of the developed design is evaluated in terms of area, power, delay.

Area: in order to calculate the overall are occupied by the design is by calculated by number of LUTS used. Xilinx Ise 14.2 is chosen as product version. Xc7a 350t-3ffg1156 is chosen as the target device.

Power: In order to calculate the overall power needed for this design is denoted by look up table. The overall power includes logical power, signal power, Ios power, leakage power. And the overall thermal properties include Junction temperature (c), Maximum Ambient (C), and Effective TJA (C/W).

Delay: The overall delay includes gate delay and net delay.

5.2. *Comparative analysis*

The efficiency of the proposed design was assessed in comparison to that of existing designs to demonstrate its accuracy and efficiency. The power, area, and delay are used to evaluate the effectiveness of the proposed approach. The proposed multiplier is compared with the current techniques, including MLAFAs [16], and DSM [17].

Table 3. Comparison of proposed with normal multiplier.

TYPE	AREA	POWER	DELAY
Normal multiplier	109	0.150 W	8.196 ns
Approximate RM 1	99	0.104 W	8.056 ns
Approximate RM 2	104	0.103 W	8.052 ns

The efficiency comparison between the proposed modules and the existing model is shown in Table 3. Compared with the existing technique the proposed recursive multiplier attain less are, power, and delay. The proposed RMul-2 design performs well compared to proposed RMul-1 design in terms of area, power and delay.

Table 3. Comparison table for approximate adder, and proposed approximate adder.

TYPE	AREA	POWER	DELAY
Normal adder	2	0.103 W	0.893 ns
Proposed Approximate adder	2	0.101 W	0.765 ns

Table 4 illustrate the performance comparison of the proposed approximate adder with normal adder where the proposed approximate adder attain less are, power, and delay. The proposed approximated adder yields the area of 2 LUTs, power of 0.101W and delay of 0.765 ns respectively.

Table 3. Comparison table for Proposed multiplier with existing meultipliers.

TYPE	AREA	POWER	DELAY
Proposed approximate RMul-1	99	0.104 W	8.056 ns
Proposed approximate RMul-2	104	0.103 W	8.052 ns
MLAFAs [16]	110	0.106 W	9.065 ns
DSM [17]	120	0.109 W	9.761 ns

Table 5 compares the performance of the suggested modules with the current model MLAFAs [16], DSM [17] based on delay, area, and power. In comparison to the other designs, as Table 5 illustrates, proposed approximate RMul-1 and proposed approximate RMul-1 produce better image qualities in terms of area, power, and delay; DSM [17] in particular require greater area. Furthermore, DSM's power ratings surpass those of MLAFAs, suggesting a strong resemblance between the roughly and precisely processed images.

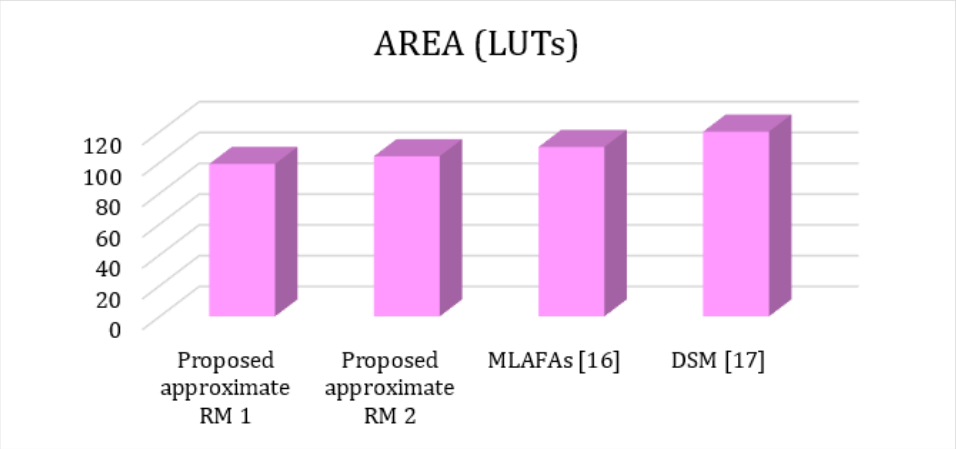


Fig. 6. Area required for the proposed and existing technique.

As shown in the figure 6, The area needed for the approximate recursive multiplier 1, recursive multiplier 2, MLAFAs [16], and DSM [17] are 99, 104, 110, and 120. Where our proposed approximate recursive multiplier 1, and approximate recursive multiplier 2. Require less area.

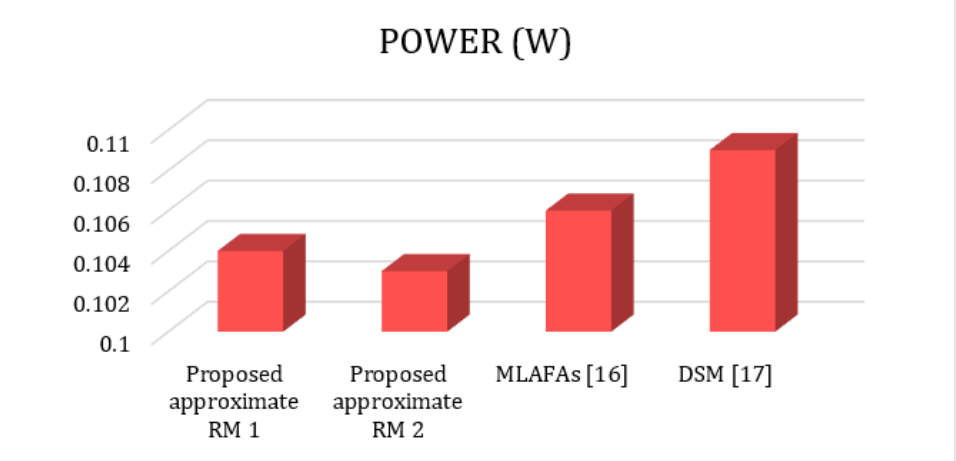
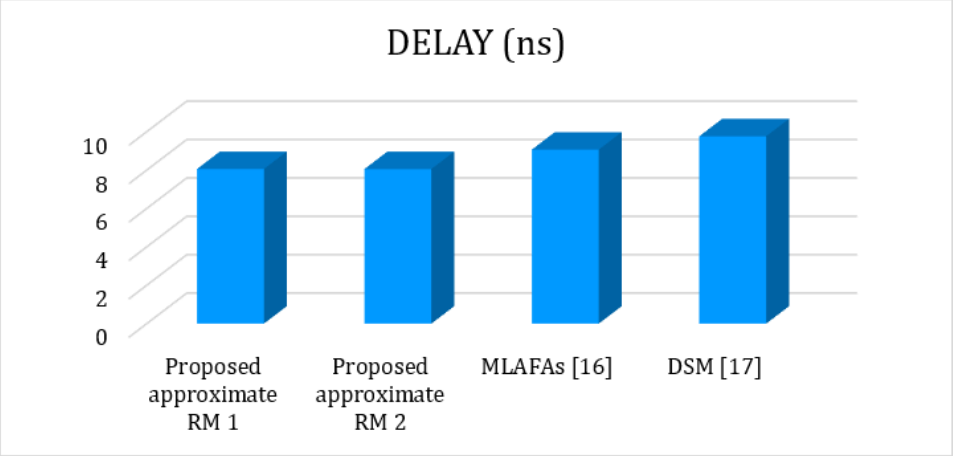


Fig. 7. Power required for the proposed and existing technique

Figure 7 shows power comparison of approximate recursive multiplier 1, approximate recursive multiplier 2, MLAFAs [16], and DSM [17] are 0.104 W, 0.103 W, 0.106 W, and 0.109 W. Where approximate recursive multiplier 2 require less power.



Figu. 8. Delay for the proposed and existing technique

Figure 8 shows delay comparison of approximate recursive multiplier 1, approximate recursive multiplier 2, MLAFAs [16], and DSM [17] are 8.056 ns, 8.052 ns, 9.065 ns, and 9.761 ns. Where approximate recursive multiplier 2 attains less delay and DSM [17] attain more delay.

5.3. *Proposed multiplier based on Image processing*

Image processing is one application where error resilience is most commonly believed, and many studies evaluate the proposed circuits in this setting. Two image processing applications images smoothing and image sharpening are examined in this work. The applications offer a deeper comprehension of the suggested designs' area of applicability.
High spatial frequency noise in images is efficiently removed using picture smoothing, which is the outcome of low pass filtering in image processing. With the help of a moving kernel, the low-pass filter analyses each pixel independently and adjusts it in reaction to neighboring pixels. Depending on the kernel size, processing each pixel necessitates many multiplications. The weighted average of the adjacent pixels really determines the value of the modified pixel. Furthermore, since minute features are invisible to the human eye, image smoothing is an error-tolerant application.
Utilising the studied multipliers, image processing has been done with the goal of blurring a test image. Figure displays the obtained images. Exact multipliers have also been used in the same processing to give a useful comparison for every design. The peak signal to noise ratio (PSNR) and the structural similarity index (SSIM) offer a numerical representation of each multiplier's efficacy in image smoothing.
The top section of the table contains recursive designs, whereas the bottom section has non-recursive designs. The suggested circuits, recursive multiplier 1 and recursive multiplier 2, exhibit competitive behaviour and achieve the most power reduction, as demonstrated in Figure (). They also share the best results with the designs MLAFAs [16] and DSM [17].

5.4. *Image Sharpening*

This section examines an algorithm for sharpening images as a use case for the suggested multiplier. For comparison, more designs for the Literature are offered. Metrics connected

to circuits and image quality are taken into account when evaluating the designs. Accuracy, area, power, and delay product are all jointly analysed. The sharpening algorithm executes the following tasks, assuming I is the original image and S is the processed image:

$$S(x,y) = 2I(x,y) - \frac{1}{273}\sum_{i=-2}^{2}\sum_{j=-2}^{2} G(i+6, j+6)(x-i, y-j) \quad (5)$$

Where G is the matrix given as:

$$\begin{bmatrix} 3 & 8 & 6 & 1 & 2 & 9 & 7 & 5 \\ 6 & 4 & 2 & 7 & 5 & 7 & 4 & 1 \\ 7 & 5 & 3 & 6 & 9 & 4 & 2 & 1 \\ 2 & 6 & 1 & 9 & 5 & 3 & 7 & 4 \\ 5 & 7 & 2 & 6 & 3 & 8 & 4 & 1 \\ 1 & 6 & 4 & 7 & 2 & 9 & 3 & 2 \\ 4 & 7 & 2 & 10 & 5 & 3 & 1 & 8 \\ 9 & 5 & 3 & 7 & 2 & 8 & 1 & 4 \end{bmatrix} \quad (6)$$

In an image, blocks of 8 X 8 pixels are used to run this algorithm. The only approximations in maths are addition and multiplication; division and subtraction are precise processes.

The suggested images by various multipliers are displayed in Figure 9. The simulation results for signal noise ratio (SNR) in relation to the precisely suggested images. SSIM measures how similar two images are, whereas SNR is described as

$$SNR = 10\log_{10}(A/MSE) \quad (7)$$

Where A is the Amplitude of a signal and MSE is given by:

$$MSE = \frac{1}{N}\sum_{T=0}^{N-1} error^2(t) \quad (8)$$

where N is the quantity of input and error is described as:

$$error(t) = a_t - b_t \quad (9)$$

As shown in the figure 9 the input image is compared with MLAFAs [16], DSM [17], approximate recursive multiplier 1, and approximate recursive multiplier 2. Out of which approximate recursive multiplier 1 multiplier is more accurate and need less area, low power, and less delay when compared with other technique. Whereas approximate recursive multiplier 2 require more area, more power, more delay when compared with approximate recursive multiplier 1.

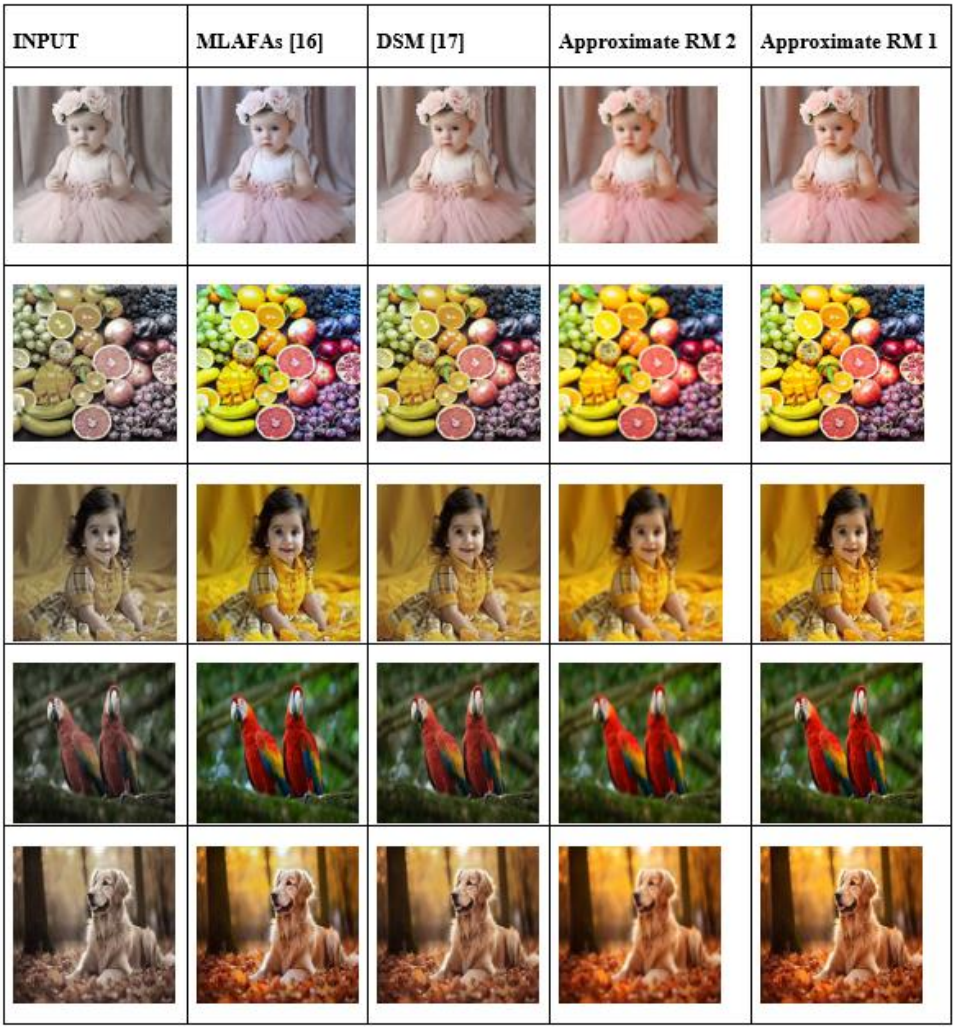


Fig. 9. Comparison of proposed with existing technique.

6. Conclusion

In this research, a novel approximate recursive multiplier (RMul-1, RMul-2), and approximate adder has been proposed to reduce complexity and power consumption. The proposed approximate recursive multiplier (RMul-1, RMult-2) are designed using NOR, AND, Half Adder, and Full Adder for low power design, and area efficiency. Moreover, the proposed power efficiency approximate adder is designed using AND, OR, and MUX for low delay. In image processing applications, the suggested approximate RMul-1 and RMul-2 are quite effective. The Xilinx ISE 13.2 tool has been used to simulate all proposed designs and current multipliers. In comparison to the accurate and current approximation designs, the proposed concept significantly decreases area, power, and delay. The proposed RMul-1 and RMult-2 have the potential to decrease power consumption by 2.5% and

3.75% in comparison to the existing MLAFAs and DSM. Moreover, the proposed approximate adder yields the delay of 0.765 ns, power of 0.101W and Area of 2 LUTs which is relatively low compare to normal adder. Meanwhile, the proposed adder improves delay by nearly 1.4% compared to RMul-1, RMul-2 and has the lowest power consumption. In addition, the proposed design has shown better results than other approximate multipliers and adder.

Acknowledgments

The author would like to express his heartfelt gratitude to the supervisor for his guidance and unwavering support during this research for his guidance and support.

References

1. W. Liu, F. Lombardi, and M. Shulte, A retrospective and prospective view of approximate computing point of view, *Proceedings of the IEEE,* **108**(3) (2020) 394-399. DOI: 10.1109/JPROC.2020.2975695
2. M. Gansen, J. Lou, F. Freye, T. Gemmeke, F. Merchant, A. Zeyer, M. Zeineldeen, R. Schlüter, and X. Fan, Discrete steps towards approximate computing, In *2022 23rd International Symposium on Quality Electronic Design (ISQED),* (2022) 1-6. IEEE. DOI: 10.1109/ISQED54688.2022.9806215
3. Y. Zhang, and D. El-Damak, A reconfigurable passive switched-capacitor multiply-and-accumulate unit for approximate computing, In *2020 IEEE 63rd International Midwest Symposium on Circuits and Systems (MWSCAS),* (2020) 921-924. IEEE. DOI: 10.1109/MWSCAS48704.2020.9184543
4. W. Liu, C. Gu, M. O'Neill, G. Qu, P. Montuschi, and F. Lombardi, Security in approximate computing and approximate computing for security: Challenges and opportunities, *Proceedings of the IEEE,* **108** (12) (2020) 2214-2231. DOI: 10.1109/JPROC.2020.3030121
5. S. Amanollahi, M. Kamal, A. Afzali-Kusha, and M. Pedram, Circuit-level techniques for logic and memory blocks in approximate computing systemsx, Proceedings of the IEEE, **108**(12), (2020) 2150-2177. DOI: 10.1109/JPROC.2020.3020792
6. P. Dahiya, and P. Jain, Efficient MDCT recursive structure for VLSI implementation, *Circuits Syst. Signal Process.* **39** (2020) 1372-1386. https://doi.org/10.1007/s00034-019-01195-x
7. R.L. Chung, C.W. Chen, C.A. Chen, P.A.R. Abu, and S.L. Chen, VLSI implementation of a Cost-Efficient Loeffler DCT algorithm with recursive CORDIC for DCT-based encoder, *Electron.* **10**(7) (2021) 862. https://doi.org/10.3390/electronics10070862
8. S.S. Raja, M.M. Mahipal, M.C. Anusha, and M.T. Venkataramana, Low power VLSI Architecture of Adaptive Recursive Karatsuba Multiplier with SquareRoot-Carry-Select-Adder using Reversible logic.
9. A. Naik, D. Deka, and D. Pal, ASIC implementation of high-speed adaptive recursive karatsuba multiplier with square-root-carry-select-adder, In *2020 IEEE 11th Latin American Symposium on Circuits & Systems (LASCAS),* (2020) 1-4. IEEE. DOI: 10.1109/LASCAS45839.2020.9068973
10. E.J. Rao, and P. Samundiswary, High-Speed and Low-Power Recursive Rounding Based Approximate Multipliers for Error-Resilience Applications. 2023. https://doi.org/10.1007/s11277-024-11283-0
11. S. Soni, A. Chauhan, Efficient VLSI Architecture for FIR Filter using Multiple Vedic Multiplier and Kogge Stone Adder: A Review.

12. S. Gorgin, H.Z Nejad, and J.A. Lee, A Practical Energy/Power Reduction Approach for Parallel Decimal Multiplier, *IEEE Access,* **10** (2022) 11372-11381. DOI: 10.1109/ACCESS.2022.3145001
13. P.Y. Chen, and Y.T. Huang, The recursive multiplexer-based multipliers, Int. J. Electron. Lett. **8** (4) (2020) 389-400. https://doi.org/10.1080/21681724.2019.1625964
14. G. Dharshini, S. Priyadharshini, K. Purnasnegha, T.D. Paul, and A. Kalaiselvi, Implementation of a recursive segmented multiplier using Static Segmentation, In *2023 2nd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA)*, (2023) 1-6. IEEE. DOI: 10.1109/ICAECA56562.2023.10199448
15. E. Zacharelos, I. Nunziata, G. Saggese, A.G. Strollo, and E. Napoli, Approximate squaring circuits exploiting recursive architectures, *Integration,* **91** (2023) 35-42. https://doi.org/10.1016/j.vlsi.2023.02.007
16. W. Liu, T. Zhang, E. McLarnon, M. O'Neill, P. Montuschi, and F. Lombardi, Design and analysis of majority logic-based approximate adders and multipliers, *IEEE Transactions on Emerging Topics in Computing,* **9**(3) (2019) 1609-1624, DOI: 10.1109/TETC.2019.2929100
17. 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,* 9 (2021) 7220-7232. DOI: 10.1109/ACCESS.2021.3049299
18. R. Jothin, M.P. Mohamed, and C. Vasanthanayaki, High performance compact energy efficient error tolerant adders and multipliers for 16-bit image processing applications, *Microprocess. Microsyst. **78*** (2020)103237. https://doi.org/10.1016/j.micpro.2020.103237
19. D. Pandey, S. Singh, V. Mishra, S. Satapathy, and D.S. Banerjee, Sam: A segmentation based approximate multiplier for error tolerant applications, In *2021 IEEE International Symposium on Circuits and Systems (ISCAS)*, (2021) 1-5. IEEE. DOI: 10.1109/ISCAS51556.2021.9401266
20. N. Van Toan, and J.G. Lee, FPGA-based multi-level approximate multipliers for high-performance error-resilient applications, *IEEE Access,* **8** (2020) 25481-25497. DOI: 10.1109/ACCESS.2020.2970968
21. N. Van Toan, and J.G. Lee, Energy-area-efficient approximate multipliers for error-tolerant applications on FPGAs, In *2019 32nd IEEE International System-On-Chip Conference (SOCC)*, (2019) 336-341. IEEE. DOI: 10.1109/SOCC46988.2019.1570548202
22. E. Zacharelos, I. Nunziata, G. Saggese, A.G. Strollo, and E. Napoli, Approximate recursive multipliers using low power building blocks, *IEEE Transactions on Emerging Topics in Computing,* **10**(3) (2022) 1315-1330. DOI: 10.1109/TETC.2022.3186240
23. H. Waris, C. Wang, C. Xu, and W. Liu, AxRMs: Approximate recursive multipliers using high-performance building blocks, *IEEE Trans. Emerging Top. Comput.* **10**(2) (2021) 1229-1235. DOI: 10.1109/TETC.2021.3096515
24. V.P. Darshini, V.N. Lakshmi, P.S. Rani, P.S. Lavanya, and S.M. Murthy, A Design of Recursive Based Approximate Multipliers. (2019).
25. G. Di Meo, G. Saggese, A.G. Strollo, D. De Caro, and N. Petra, Approximate Floating-Point Multiplier based on Static Segmentation, *Electron,* **11**(19) (2022) 3005. https://doi.org/10.3390/electronics11193005
26. I. Nunziata, E. Zacharelos, G. Saggese, A.M.G. Strollo, and E. Napoli, Approximate recursive multipliers using carry truncation and error compensation, In *2022 17th Conference on Ph. D Research in Microelectronics and Electronics (PRIME)*, (2022) 137-140, IEEE. DOI: 10.1109/PRIME55000.2022.9816787
27. S. Mazahir, O. Hasan, R. Hafiz and M. Shafique, Probabilistic error analysis of approximate recursive multipliers, *IEEE Trans. Comput.* **66**(11) (2017) 1982-1990. DOI: 10.1109/TC.2017.2709542