International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 05 | May 2026 www.irjet.net p-ISSN: 2395-0072

Static Segmentation based Approximate MAC for High Performance

D. Anantha Lakshmi [1], P. Ramya Sree [2], G. Sravanthi [3], R. Likhil Vamsi [4], B. Karthik [5], Y. Sujatha [6]

[12345] *B. Tech Students,[6]Sr. Assistant Professor, Department of Electronics and Communication Technology, Sri Vasavi Engineering College, Tadepalligudem, Andhra Pradesh, India*

Abstract - *The demand for high-performance and energy-efficient digital systems is increasing, so improving arithmetic units has become important. The Multiply–Accumulate (MAC) unit is a key part of many digital systems, but traditional MAC designs consume more power and require more area because of complex multiplication and addition operations. In this paper, a modified MAC unit using a static segmentation method is proposed to improve hardware efficiency. The design Y=A×B+C uses segmented multiplication and a single carry propagate adder to make the circuit simpler and reduce complexity. Multiplexers are used to select partial products and control the flow of data. An error compensation method is also included to reduce calculation errors and maintain acceptable accuracy. The proposed design aims to balance accuracy and hardware usage. Simulation results show that this design reduces power and area while still providing good performance compared to existing MAC units. To see how well the proposed MAC unit works we look at error metrics like Relative Error Distance (RED), Mean Relative Error Distance (MRED) and Normalized Mean Error Distance (NMED). We did some calculations. Found that the Relative Error Distance is 0.045, 0.052 and 0.058 for different inputs. On average the Mean Relative Error Distance is 0.0516. The Normalized Mean Error Distance is 0, 1 and 0 which means the proposed method keeps the error low for most of the time. The simulation results show that the proposed approximate MAC uses 60% less power because it switches less does computations in parts and has a simpler way of handling carries. The hardware area is also smaller by 30% because it uses fewer transistors generates fewer partial products and has a more compact design. The proposed MAC unit is a choice when we need to balance performance and energy efficiency. We can use the proposed MAC unit in applications where energy efficiency and reduced hardware complexity is very important.*

***Key Words*: Approximate computing, static segmentation, error compensation, carry propagate adder.**

1. INTRODUCTION

Very Large Scale Integration (VLSI) technology plays a part in making modern digital systems like digital signal processing and image processing work. This technology is also used in machine learning accelerators. We have a lot of high-performance electronic devices now. So reducing power consumption is a problem, in Very Large Scale Integration circuit design while we still want these devices to work really fast. Very Large Scale Integration circuit design is very important here. Approximate Computing has emerged as an effective design paradigm to address this challenge by allowing small and acceptable computational errors in exchange for significant improvements in power efficiency, hardware complexity, and processing speed [1]. Several researchers have proposed approximate arithmetic circuits, including adders, multipliers, and compressors, to reduce delay, area, and power consumption in digital systems [2],[3],[4],[5],[6]. These techniques are particularly suitable for error-tolerant applications such as multimedia processing, signal processing, and machine learning. Among the fundamental arithmetic components, the Multiply–Accumulate (MAC) unit is one of the most critical building blocks in many computational systems, especially in DSP, image processing, and neural network accelerators. Since MAC units perform a large number of arithmetic operations, they significantly influence the overall system power consumption and performance. Therefore, designing efficient MAC architectures has become an important research focus in low-power VLSI design [7],[8],[9]. [10] The proposed design integrates a low-power, high-speed, error-configurable 8-bit approximate multiplier that significantly improves hardware efficiency compared to a conventional Wallace tree multiplier, achieving about 54.4% area reduction, 66% lower power consumption, and 51.5% reduction in delay. [11] Its extended idea to improve the overall system reliability and flexibility by adjusting the conditions of the operation.[12] The MAX-DNN extends the concept of dynamic energy & accuracy trade-offs specifically into DNN hardware with the implementation of multi-level arithmetic approximations. The DNN hardware allows for high efficiency of performance to sustain equivalent levels of accuracy whilst continuing to provide a means for scalability between dynamic energy & accuracy. Recently, approximate MAC architectures based on static segmentation have been proposed to reduce computational complexity and improve energy efficiency [13]. However, further improvements are

still required to optimize power consumption and maintain acceptable computational accuracy.

Hence in order to enhance, in this work a modified Multiplier–Accumulator (MAC) architecture based on static segmentation, implemented using Segmented Sub-Multiplier (SSM) helps make big multipliers work better. It breaks them into parts. This makes them use hardware and less power. They also become easier to scale up in VLSI chip designs. The Segmented Sub-Multiplier makes this all. The proposed design introduces architectural modifications to reduce power consumption while maintaining computational effectiveness, making it suitable for energy-efficient VLSI-based computing systems. The design is evaluated using standard error metrics such as Relative Error Distance (RED), Mean Relative Error Distance (MRED), and Normalized Mean Error Distance (NMED), showing that increasing approximation improves efficiency but also raises error levels. The proposed method achieves a balanced and narrower error distribution, significantly reducing overall error (NMED), even for large approximation levels and realistic datasets.

2. MAC UNIT DESIGN FOR PROPOSED WORK

The multiplier block does the multiplication of two input numbers first. It makes products and then combines them to get the final product. In the design the people use simple ways, like cutting off some partial products or using easier logic to use less power and be faster. The multiplier block still gets a good result. The result then goes to the adder. The adder adds this result to the sum that it already has. The adder does this times adding new products each time. The adder is made to be fast and not make many mistakes. This way the multiplier block and the adder use energy and still get pretty good results. The multiplier block and the adder are important because they need to balance using energy and getting good results.

1.1 Multiplier

It describes that the m × m multiplier for implementing an n ×n product (with n/2≤m<n) provided by the SSM and shows operand A partitioned into a lower area (LA) that contains its m least considerable bits (LSBs); and an upper region (HA) that contains its m most significant bits (MSBs). Same for operand B.

$$L_A = A[m-1:0]$$

$$H_A = A[n-1:n-m]$$

Where m=4, n=8

Let us consider the two inputs of multiplier as A_{ssm} and the other input as B_{ssm}. Here the two operands A and B are constructed using 4-bit 2x1 mux. Segmentation is applied to both the inputs and the output of the multiplier is approximated as

$$m \times m = A_{ssm} \times B_{ssm}$$

SSM requires only m × m multiplier for the inner product $A_{ssm} \times B_{ssm}$, which is 4x4 multiplier is used to reduce area efficiency. The output of multiplier is given as the input for the accumulator as shown in fig 1.

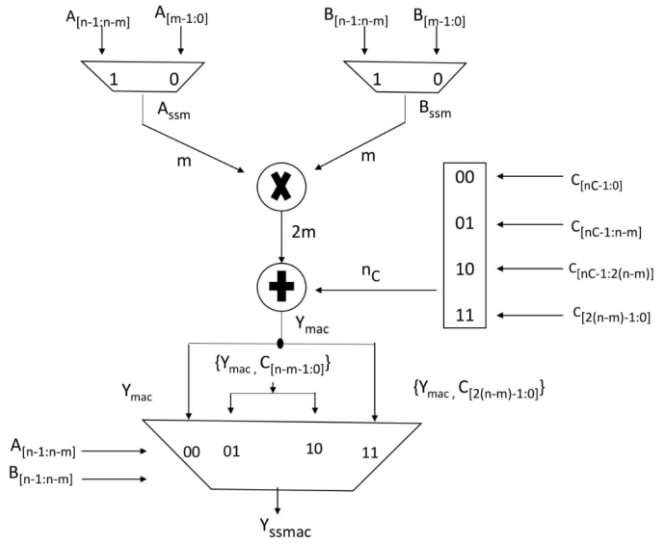


Fig 1: Implementation of static segmented multiplier-accumulator

1.2 Segmentation of addend C

The C is choosing between the portions C[nC-1:0], C [nC-1: n-m], and C[nC-1:2·(n-m)]. The multiplexer on the output rearranges the result including the bits C[n-m-1:0] and C[2·(n-m)-1:0] at the least significant positions in case of right-shift. We subdivide the input C into two portions as lower portion (L_C) and upper portion (H_C), which constitute the m_c bits, with $n_c/2 \le m_c < n_c$

$$L_C = C[m_C - 1:0]$$

$$H_C = C[n_C - 1:n_C - m_C]$$

The proposed work Static-Segmented MAC (SSMAC) can be organized with two parameters: m that controls the segmentation of A and B, and m_c which controls the segmentation of C. The proposed work shows the optimization of C, which means using the same 8-bit adder for designing but using different logic style which reduces the transistor count, as shown in the table-1. For the construction of adder C we used the Pass Transistor Logic (PTL) to design because it reduces the area, low power consumption, hardware complexity. It also Increase the speed of the performance.

The final approximation output of the SSMAC can be written as

$$Y_{SSMAC} \cong A_{ssm} \times B_{ssm} + C_{ssm}$$

Table -1: Performance Comparison

Logic style	Transistors for One full-adder	8-bit Adder
Gate logic	28	224
PTL	10	80
CMOS	20	160

3. RESULTS

Evaluation of proposed MAC unit over various error metrics:

We evaluate the area of the proposed approximate MAC unit in terms of error using Relative Error Distance (RED) and Mean Relative Error Distance (MRED).

a. Relative Error Distance (RED):

The formula for calculating the relative error distance for each test case is:

$$\text{RED}=\left|\frac{Y_{approx}-Y_{exact}}{Y_{exact}}\right|$$

Where Y_{exact} is the exact output while Y_{approx} is the approximate output

Computation Method:

By varying the input parameters of A, B, and C, we have obtained the exact output by using the method: Y=A×B+C
The output from the proposed approximate MAC unit is used to obtain the approximate output. The RED is then computed.

Example Calculations

$$\text{RED1} = \left|\frac{21-22}{22}\right| = 0.045$$

$$\text{RED2} = \left|\frac{18-19}{19}\right| = 0.052$$

$$\text{RED3} = \left|\frac{16-17}{17}\right| = 0.058$$

b. Mean Relative Error Distance (MRED):

The mean relative error distance (MRED) is computed by taking the average of all RED values:

$$\text{MRED} = \left(\frac{\text{RED1} + \text{RED2} + \text{RED3} + \ldots + \text{REDN}}{N}\right)$$

Final Calculation

$$\text{MRED} = \left(\frac{0.045 + 0.052 + 0.058}{3}\right)$$

$$\text{MRED} = 0.0516$$

Table -2: Error analysis of proposed approximate MAC

A	B	C	Exact Output	Approx Output	RED
5	4	2	22	21	0.045
3	6	1	19	18	0.052
2	7	3	17	16	0.058

c. Normalized Relative Error computation:

Normalized Relative Error (NRED) is a performance evaluation metric used to measure the accuracy of predictive models. It is derived from the Relative Error (RED), which quantifies the deviation between actual and predicted values relative to the actual value. The relative error for the i^{th}observation is defined as:

$$RED_i = \left|\frac{A_i - P_i}{A_i}\right|$$

where A_i and P_i denote the actual and predicted values, respectively. To ensure comparability across different observations, the RED values are normalized using the minimum and maximum values of RED. The normalized relative error is defined as:

$$NRED_i = \sqrt{\frac{RED_i - RED_{min}}{RED_{max} - RED_{min}}}$$

Table-3: Actual and Predicted values

Sl. No	A_i	P_i
1	100	90
2	200	150
3	300	330

Using the data in Table 3, the relative error values are calculated as follows:

$$RED_1 = \left|\frac{100-90}{100}\right| = 0.10$$

$$RED_2 = \left|\frac{200-150}{200}\right| = 0.25$$

$$RED_3 = \left|\frac{300-350}{300}\right| = 0.10$$

Thus, the minimum and maximum RED values are:

$$\text{RED}_{min} = 0.10, \text{RED}_{max} = 0.25$$

Table-4: Relative Error (RED)

Sl. No	A_i	P_i	RED_i
1	100	90	0.10
2	200	150	0.25
3	300	330	0.10

Using the NRED formula,

$$NRED_1 = \frac{0.10-0.10}{0.25-0.10}\frac{0.10}{0.15} = 0$$

$$NRED_2 = \frac{0.25-0.10}{0.25-0.10} = \frac{0.15}{0.15} = 1$$

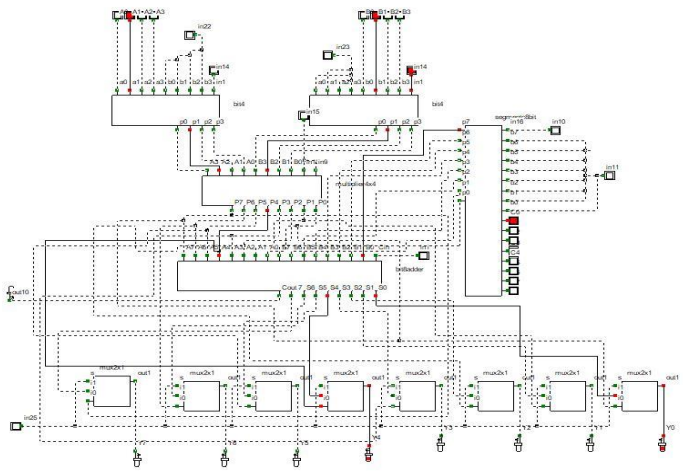
$$NRED_3 = \frac{0.10-0.10}{0.25-0.10} = \frac{0}{0.15} = 0$$

Table-5: Normalized Relative Error (NRED)

Sl. No	RED_i	$NRED_i$
1	0.10	0
2	0.25	1
3	0.10	0

Thus, the normalized relative error values are $NRED_1 = 0,\ NRED_2 = 1,\ NRED_3 = 0$.

Proposed Static Segmentation Circuit with Conditional Output Generation



4. CONCLUSION:

In this paper, we have designed an approximate MAC unit Y=A×B+C using static segmentation to increase hardware efficiency. Segmentation is applied to all input operands, which makes the design efficient by decreasing its complexity. In addition, a carry-propagate adder is employed to minimize delays and power requirements. The design outperforms traditional MAC units in terms of both speed and area reduction. The proposed MAC unit is designed and simulated using DSCH3.9 tool based on 45nm technology. This implementation helps to reduce power usage by 60%. It also reduces the area by 30% compared to the MAC designs. Configurable parameters make the design flexible, allowing a balance between accuracy and efficiency depending on the application needs. Even if the approximate design is inherently inaccurate, the errors committed are tolerable within many applications. The use of error correction improves the accuracy of the overall system. This design has proven to be highly efficient in terms of power and hardware utilization.

REFERENCES

[1] Han, Jie, and Michael Orshansky. "Approximate computing: An emerging paradigm for energy-efficient design." 2013 18th IEEE European test symposium (ETS). IEEE, 2013.

[2] Raha, Arnab, Hrishikesh Jayakumar, and Vijay Raghunathan. "Input-based dynamic reconfiguration of approximate arithmetic units for video encoding." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 24.3 (2015): 846-857.

[3] Pashaeifar, Masoud, et al. "Approximate reverse carry propagate adder for energy-efficient DSP applications." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 26.11 (2018): 2530-2541.

[4] Ye, Rong, et al. "On reconfiguration-oriented approximate adder design and its application." 2013

IEEE/ACM international conference on computer-aided design (ICCAD). IEEE, 2013.

[5] Venkatachalam, Suganthi, and Seok-Bum Ko. "Design of power and area efficient approximate multipliers." IEEE Transactions on Very Large Scale Integration (VLSI) Systems 25.5 (2017): 1782-1786.

[6] Park, Gunho, Jaeha Kung, and Youngjoo Lee. "Design and analysis of approximate compressors for balanced error accumulation in MAC operator." IEEE Transactions on Circuits and Systems I: Regular Papers 68.7 (2021): 2950-2961.

[7] Mukherjee[1], Sumant, and Saurabh Mitra. "Energy Efficient Approximate MAC Unit for High Speed DSP Application."

[8] Esposito, Darjn, Antonio GM Strollo, and Massimo Alioto. "Low-power approximate MAC unit." 2017 13th Conference on Ph. D. Research in Microelectronics and Electronics (PRIME). IEEE, 2017.

[9] Palanisamy, Yogeswari, et al. "Power-efficient MAC unit for image processing using Dadda multiplier and approximate adder." Australian Journal of Electrical and Electronics Engineering (2025): 1-13.

[10] Delavari, Arvin, et al. "A reconfigurable approximate computing RISC-V platform for fault-tolerant applications." 2024 27th Euromicro Conference on Digital System Design (DSD). IEEE, 2024.

[11] Alioto, Massimo. "Energy-quality scalable adaptive VLSI circuits and systems beyond approximate computing." Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017. IEEE, 2017.

[12] Leon, Vasileios, et al. "Max-dnn: Multi-level arithmetic approximation for energy-efficient DNN hardware accelerators." 2022 IEEE 13th Latin America Symposium on Circuits and System (LASCAS). IEEE, 2022.

[13] Di Meo, Gennaro, et al. "Approximate MAC unit using static segmentation." IEEE Transactions on Emerging Topics in Computing 12.4 (2023): 968-979.