2021 IEEE 18th India Council International Conference (INDICON) | 978-1-6654-4175-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/INDICON52576.2021.9691492

Energy Efficient Approximate MACs

Thejaswini P*†, John Jose†, Sukumar Nandi†,
* Department of ECE, JSS Academy of Technical Education, Bangalore, India
† Department of CSE, Indian Institute of Technology Guwahati, Assam, India

Abstract—**Approximate computing or sacrificing computation efficiency for computational effort, has recently emerged as a promising design strategy. Several research studies have investigated approximate computation at both the software and hardware levels of abstraction over the last decade, with promising results. The multiply-add unit, the most basic and commonly used arithmetic block in digital systems has gotten a lot of attention for approximation at the hardware level of abstraction. We propose six Approximate MAC units (AMACU-1 to AMACU-6) for high-performance and energy-efficient approximate computing in this paper. The proposed AMACUs' main design goal is to reduce the number of partial product stages to a minimum error rate. In terms of error efficiency and overhead analysis, the proposed AMACUs along with the conventional MAC and existing approximate MAC units are compared. It is noticed that, there is a considerable reduction in area and power by 64.45AMACUs along with the existing approximate MAC units in an image smoothing application. Good PSNR and SSIM values were obtained.**

Index Terms—**Approximate computing, approximate MAC, approximate adder, QoR**

I. Introduction

In today's IoT era, lot of edge devices handle data sensing and processing before sending the data to the central servers. For edge devices associated with signal and image processing applications, the sensed data might have to go through many operations which involves filtering, smoothing, compressing and comparing. The edge devices can't afford power hungry circuits and need to carry out the above operations using low latency circuits as well [1]. To achieve power reduction, edge devices have to make compromises in accuracy. In smart devices, few applications has error endurance that compromise accuracy over time and power for greater efficiency. Since the Quality of Experience (QoE) at end user of an application may change altogether at runtime, it is ideal to plan quality-configurable frameworks that can leverage computation quality concurring to application prerequisite [2].

Multiplier and Accumulator (MAC) units are integral part of signal and image processing operations [3]. Speedup and efficiency of most of the complex arithmetic circuits are determined by the MAC unit that lies in its critical path. MAC units are considered to be one of the most energy-intensive digital component [4]. Traditional approach to build a MAC unit is to introduce multiplication and accumulation as individual functional blocks, then cascade them. It consists of partial product generation, partial product reduction, carry propagation and accumulation stages. In order to employ MAC units in edge devices, we have to re-design them from the energy-efficiency aspect and this has been a growing area of research interest [5]. Approximate computing done by trading accuracy over power and latency is presently one of the promising approach that can be in line with these objective [6]. A big challenge in approximate computing is to decide the level of approximation needed in various stages of data processing so as to meet QoE requirements.

To increase the energy efficiency of MAC unit, we must micromanage at carry propagation, accumulator level and partial product reduction stage. MAC unit approximations [7] are introduced in any of the three stages (1) truncated multipliers in the partial products generation stage [8], (2) approximate compressors in the partial products reduction stage [9] and (3) approximate adders in the carry-propagate addition stage [10]. Combining multiplication and accumulation operations into a single functional block can give speedup. The partial products reduction network using tree architectures can speed up the multiplication process [11]. Even though these approximation approaches are energy efficient, our in-depth analysis shows the scope of further optimization with reasonable compromise in accuracy. We make the following contributions in this paper,

- We propose a novel Approximate Full Adder (AFA), to reduce power consumption and area of multipliers. This AFA is used in partial product reduction stage.
- We propose five different variants of Approximate MAC (AMACU) architectures by varying the design parameters which helps the end users to handpick the best variant according to the application requirement.
- We experimentally prove that the proposed architectures has area and power reduction upto 35% with respect to the state of art techniques [12] [13].
- We consider an image smoothing application to validate our proposed techniques in terms of PSNR and SSIM with respect to state of art techniques [12] [13].

II. Related Work

Approximate circuits have been a well researched area due to trade off between accuracy, power and latency. The Low-

Power Approximate MAC Unit [14], uses partial product approximate compression, which includes approximate counters made up of OR gates. To further reduce energy, selected columns of partial product terms are neglected. Compensation factor is introduced to fine tune the approximation error depending on the application. This technique was validated on an image processing application that demonstrated a good quality-power trade-off with acceptable image quality degradation.

Low-Power Configurable Adder for Approximate Applications [15] proposed a carry-maskable adder with configurable accuracy at runtime. This scheme flexibly choose the duration of the carry propagation to meet the quality requirements by dynamically changing the length of the carry propagation. Additional logic blocks is not required in this technique. This technique achieves accuracy configuration, decreased power consumption and critical path delay compared to state of art technique.

Energy-Efficient unsigned Approximate MAC Unit architecture [16], includes approximation in both the multiplication and accumulation stages. A total of four different approximate MAC architectures are proposed. The least-significant bits of the product are approximated in the multiplication stage depending on the leading-1's in the least-significant portion of the multiplicand. Error detection and compensation bits are generated for precision control. This technique has considerate amount of reduction in the area-power product.

An energy efficient MAC [12] uses the existing Approximate Tree Compressor (ATC) to trade off accuracy. ATC is used in the partial product reduction stage. In the carry propagation stage, carry propagate adder is used to add sum and carry outputs.The accuracy compensation vectors are used for error vector recovery. This technique was validated on an image processing application that demonstrated a good quality-power trade-off with high recognition rate.

A low-power approximate multiply-accumulate (MAC) [13] proposed Carry-Maskable Adder (CMA). This technique makes use of an approximate tree compressor and CMA to achieve configurability for accuracy scaling. This approximate tree compressor is utilized to reduce the number of rows in partial product reduction stage. Experimental results demonstrate reduced power consumption and circuit area, without impacting accuracy significantly when compared to the conventional MAC unit.

When compared to a traditional MAC unit constructed using the Baugh-Wooley multiplier, an efficient MAC [17] design for 2's complement numbers with reduced latency, area, and power consumption is proposed. The partial product rows are generated using modified Booth multiplier. In order to increase performance even further, the partial product generation employs the Twin Precision Concept. Carry save adders are used to reduce delay in the partial product reduction stage. Speed of the multipliers are improved significantly by carefully determining the number of channel stages and location where the pipeline registers will be introduced. This technique achieves good delay-power product over conventional MAC.

III. Motivation

Some soft real time systems include components that consists of resilient portions. During processing of data, presence of small errors in these components will not affect the Quality of Result (QoR). Any variation in the least significant bits of data may not create substantial effect on QoR [18]- [20]. Running extra iterations with reduced precision of intermediate computations can still provide the same QoR in many iterative refinement algorithms [21]- [24]. By exploiting this error margin, approximate circuits can improve performance and energy efficiency.

In image processing domain, Peak Signal to Noise Ration (PSNR) value greater than 30dB is generally acceptable. Many existing approximate circuits used in image processing applications have PSNR of 40.06dB [12], 35.66dB [13] and structural similarity index measure (SSIM) of 0.90 [12], 0.37 [13] at the expense of higher area and energy footprint. Hence, a cost effective and energy efficient approximate MAC unit architectures are proposed with minor variations in the design parameters that helps end users to choose the approximate MAC according to the requirement. An image smoothing application is considered for validating the proposed architectures in terms of PSNR and SSIM.

IV. Proposed Architecture

Most widely used MAC is a combination of Wallace tree multiplier and Carry propagate adder. A multiplier usually has three phases: *Partial Product Generation* (PPG), *Partial Product Accumulation* (PPA), and a *Carry Propagate Addition* (CPA). In PPA stage, full adder is used to carry out addition of the three partial product grouping and half adders are used to carry out addition of the two partial product grouping. In the final CPA stage, carry propagate adder is used to add the contents of the accumulator with the obtained accumulated partial products. The product will be stored in the accumulator itself. The operation of MAC is represented as,

$$ACC = (MD * MR) + ACC \tag{1}$$

In order to improve the speed and performance of the MAC unit, we need to reduce the number of partial products that is used in the multiplication block. We also need to address the long paths for the carry propagation involved in addition of large operands. In the proposed work, both the multiplication and accumulation operations are carried out within the same functional block. Approximation is introduced both at the multiplication and at the accumulation stages. We propose an unsigned Approximate MAC Unit (AMACU) and its six minor variants (AMACU-1, AMACU-2, AMACU-3, AMACU-4, AMACU-5 and AMACU-6).

AMACU-1: The architecture of the proposed AMACU-1 is shown in Fig 1. Usually in PPA stage, full adder is used for adding the generated partial products. A full adder can be expressed as,

$$S = A \oplus B \oplus C \tag{2}$$

$$Cout = A.Cin + B.Cin + A.B \tag{3}$$

where A, B, and Cin are the binary inputs and Sum (S), Cout are the binary outputs. In our proposed work approximation is carried out at the full adder level. First, variants of approximate full adder (AFA) is designed where Approximate Sum (AS) and Approximate Carry as (AC) are expressed as

$$AS = A + B + C \tag{4}$$

$$AC = \text{any one of the inputs (A/B/C)} \tag{5}$$

In AFA, as we have considered Cout as A instead of Equation 3, AC is correct for six out of eight combinations and therefore the probability of AC to be correct is $6/8$=0.75. Similarly, Approximate Sum (AS) is logical OR operation of the inputs instead of logical XOR as in equation 2, S is correct for five out of eight combinations and therefore the probability of AS to be correct is $5/8$=0.625.

Fig 1 represents operations of an approximate MAC unit (8bx8b+16b) with 64(= 8x8) partial products. A logical AND gate produces a 1-bit product from a 1-bit multiplicand and a 1-bit multiplier represented as a single dot in Stage 2. A row of black circles are used to represent Partial Products (PPn, where n=0,...,7) in the second level. The AFA decreases the PPs in the third stage while simultaneously adding the previously accrued value. A fused MAC device is proposed instead of naively combining a multiplier and an adder. It takes advantage of the PP reduction mechanism and incorporates the AFA accumulation. The upper and lower halves of the cumulative value (ACCH and ACCL) are represented by yellow and brown circles, respectively.

In stage 3 we form three groups of three rows out of eight rows of PPn (n=0,...,7). Each group of three rows are inputs to AFA by which three pairs of X and Y are generated. A square and a diamond represent X and Y, respectively. They are named (X1, Y1), (X2, Y2), (X3, Y3) and represented as,

$$\begin{aligned} X1 &= PP0 + PP1 + PP2 \\ X2 &= PP3 + PP4 + PP5 \\ X3 &= PP6 + PP7 + ACCH \\ Y1 = PP0, \quad Y2 &= PP3, \quad Y3 = PP6 \end{aligned} \tag{6}$$

In the third stage, the three rows, (X1, X2, X3) are operated by AFAs by which X4 & Y4 are generated. Carry generated Y1, Y2 and Y3 are logically ORed to obtain Y5.

$$\begin{aligned} X4 = X1 + X2 + X3, \quad Y4 &= X2 \\ Y5 &= Y1 + Y2 + Y3 \end{aligned} \tag{7}$$

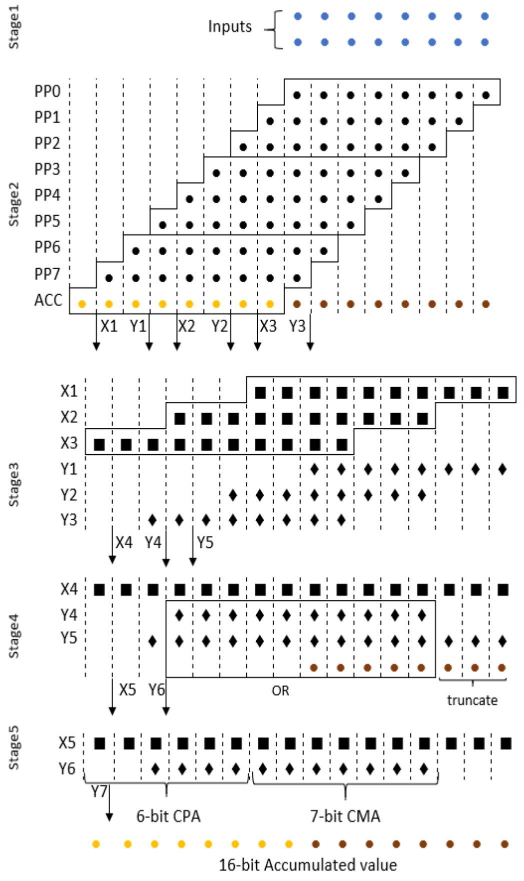


Fig. 1: AMACU-1

In the fourth stage, we perform logical OR operation on Y4, Y5 and ACCL after truncating least significant 3 bits of Y5 and ACCL to generate Y6. The least significant bits are truncated as their contribution is less to the accuracy.

$$Y6 = Y4 + Y5 + ACCL \tag{8}$$

In the final fifth stage, (X4, Y6) are summed up using 6-bit CPA and 7-bit CMA. CPA is used to sum up higher 6-bits of X4 and Y6; CMA [15] is used to sum the rest 7-bits. In the fifth stage,Carry-Maskable Adder (CMA) [15] is

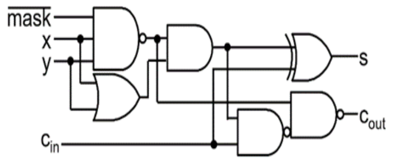


Fig. 2: Carry-Maskable Adder [15]

used to configure the accuracy of the final product. Figure 2 shows a 1-bit CMA. When mask=0 its carry-out is always zero, and thus, it is not propagated. Its sum works as $s = p+q$ otherwise, it works as the conventional full adder. We also assume that cin is zero. Since the most significant bits are needed to ensure accuracy, we sum them up with the use of conventional CPA.

AMACU-2: The architecture of the proposed AMACU-2 is similar to AMACU-1 with few minor variations to achieve a higher accuracy than AMACU-1. The variations are applied in calculations of carry Y1, Y2 and Y3 in stage 2, which are obtained as,

$$\begin{aligned} Y1 = PP1, \qquad Y2 = PP4 \\ Y3 = (PP6 * PP7) + (PP7 * ACCH) \\ +(ACCH * PP6) \end{aligned} \tag{9}$$

Calculation of X4, Y4 and Y5 in stage 3 is same. In the fourth stage, X5 and X6 is computed from X4, X5, Y4 and ACCL as shown in equation 10 and in the final fifth stage, product is computed using CPA on X5 and X6.

$$X5 = X4 + Y4, \qquad X6 = Y5 + ACCL \tag{10}$$

AMACU-3: The architecture of the proposed AMACU-3 makes use of variation of AF1 referred as AF2. In AF2, approximate sum is same as AF1 and approximate carry is the logical AND of all the three inputs. This variant achieved good accuracy with lesser overhead compared to AMACU-2. AMACU-3 consists of 5 stages out of which stage 1 is similar to AMACU-1. In stage 2 instead of AF1, AF2 is used to generate sum and carry pairs. X1, X2 and X3 are obtained as in AMACU-1. Y1, Y2, Y3 are obtained as,

$$\begin{aligned} Y1 = PP0 * PP1 * PP2 \\ Y2 = PP3 * PP4 * PP5 \\ X3 = PP6 * PP7 * ACCH \end{aligned} \tag{11}$$

In the third stage, three rows, (X1, X2, X3) are operated by AF2s by which X4 & Y4 are generated. Carry generated from previous stage Y1, Y2 and Y3 along with ACCL are logically ORed to obtain Y5.

$$\begin{aligned} X4 = X1 + X2 + X3 \\ Y4 = X1 * X2 * X3 \\ Y5 = Y1 + Y2 + Y3 + ACCL \end{aligned} \tag{12}$$

In the fourth stage, Y6 is computed from Y4 and Y5 by logical OR operation, truncating the least three significant bits of Y5, Y6= Y4 + Y5 (ignore least 3 significant bits of Y5) In the final fifth stage, product is computed using CPA on X4 and Y6.

AMACU-4: The architecture of the proposed AMACU-4 consists of 5 stages out of which stage 1 and stage 2 remains same as that of AMACU-1. In stage 3, the carry generated from stage 2 i.e. Y3 is right shifted by one place and similar operation is carried out as in AMACU-2. Generation of X5 and X6 in the fourth stage remains same as that of AMACU-2. In the final fifth stage, product is computed using CPA on X5 and X6. This variant also was able to achieve good accuracy with less overhead compared to AMACU-2&3.

AMACU-5: The architecture of the proposed AMACU-5 is similar to AMACU-4. In stage 3, the carry generated from stage 2 i.e. Y1, Y2 and Y3 are right shifted by one place and similar operation is carried out as in AMACU-4. Stage4 and stage5 remain same as that of AMACU-4. This variant was designed to have less overhead with a slight decrease in accuracy.

AMACU-6: The architecture of the proposed AMACU-6 also consists of 5stages.

Stage 1 is same as AMACU-1. In stage 2 X1, X2 and X3 are calculated using logical OR operation on the grouped rows and the four approximate carries Y1, Y2, Y3 and Y4 are generated using logical AND operation as,

$$\begin{aligned} Y1 = PP1 * PP2 \\ Y2 = PP3 * PP4 \\ Y3 = PP5 * PP6 \\ Y4 = PP7 * ACCH \end{aligned} \tag{13}$$

In the third stage, X4, Y5 and Y6 are computed as,

$$\begin{aligned} X4 = X1 + X2 + X3, \qquad Y5 = X2 \\ Y6 = Y1 + Y2 + Y3 + Y4 + ACCL + PP0 \end{aligned} \tag{14}$$

In the fourth stage, Y7 is computed from Y5 and Y6 by logical OR operation, truncating the least three significant bits of Y6,

$$Y7 = Y5 + Y6 \text{ (ignore least 3 significant bits of Y6)} \tag{15}$$

In the final fifth stage, product is computed using CPA on X4 and Y7. This variant was able to achieve higher accuracy than AMACU-5 with the same overhead maintained.

V. Experimental Results

Exact multipliers and state-of-art approximate multipliers [12], [13] are compared with the proposed Approximate MAC units (AMACU-1 to AMACU-6) in terms of error performance and overhead analysis.

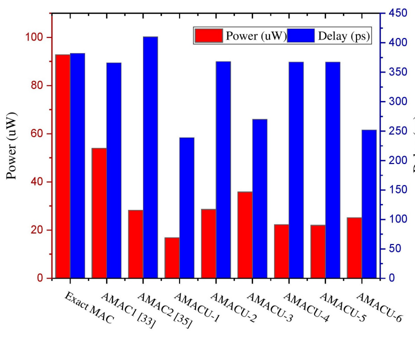
A. *Error performance*

For an unsigned 8*8 inputs with various accumulator values, Table I reports Mean Error Distance (MED), Mean Relative Error Distance (MRED) and Normalized Mean Error Distance (NMED) values. Error metrics were computed using 20,000 sets of randomly generated inputs. To measure an error distance value, the obtained output value was compared to an output value calculated using exact MAC operations.

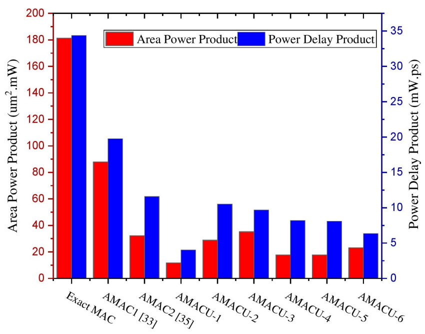
The MRED and NMED [25], are among the metrics proposed to assess the error characteristics of approximate arithmetic circuits.

The Error Distance (ED) is the difference between an Accurate Sum (AcS) and its Approximate Sum (AxS).

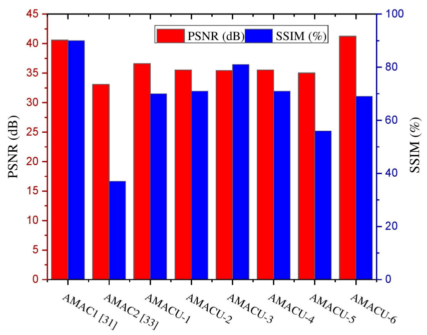
The Relative ED (RED) is defined as accurate sum over error distance obtained. The MRED is the average of REDs.



(a) Power Delay analysis



(b) PDP-APP Analysis



(c) PSNR SSIM Analysis

Fig. 3: Performance analysis of various architectures using different quality metrics

TABLE I: Error Analysis of Various Architectures.

Design	MED	MRED	NMED
AMAC1 [12]	247.13	0.01	1.10
AMAC2 [13]	2055.95	0.11	9.14
AMACU-1	1088.44	0.05	4.84
AMACU-2	1405.15	0.11	6.25
AMACU-3	1651.12	0.07	7.34
AMACU-4	1398.16	0.08	6.21
AMACU-5	1453.13	0.08	6.46
AMACU-6	1434.66	0.08	6.38

A metric for comparing multipliers of different sizes is the NMED.

$$\begin{aligned} ED &= |AxS - AcS| \\ RED = ED/AcS &= |AxS - AcS|/AcS \\ NMED &= MED/Mmax \end{aligned} \tag{16}$$

B. Power, Area and Delay Overhead Analysis

On the basis of area and power, the proposed MAC units are evaluated. The proposed unsigned 8×8 Approximate MAC configurations (AMACU-1 TO AMACU-6) are compared with conventional MAC unit consisting of the Wallace tree multiplier-CPA and existing approximate MAC units AMAC1 [12] , AMAC2 [13]. Designs are implemented in Verilog HDL. The Cadence incisive simulator is used for functional verification. The designs are synthesised in 90nm technology using Cadence Genus. The performance analysis of proposed architectures along with state-of-art techniques using power, area and delay metrics are shown in Fig 3. It is evident from Fig 3(a) that our proposed architecture AMACU-1 encounters less delay with minimal power consumption. In order to find the average amount of energy required to perform the computation and circuit area-power usage, Power Delay Product (PDP) & Area Power Product (APP) analysis is carried out and results are plotted in Fig 3(b). It is evident from the graphs that there is a significant decrease in the PDP and APP compared to the state-of-art techniques.

VI. Case Study: Image Smoothing

In modern smart world there are lot of edge devices operating on processing images. Cost effective energy efficient approximate MACs units is an integral part of such systems. Hence we consider an image soothing application to evaluate the performance of our proposed architectures. An image smoothing application that uses unsigned values is considered. Image smoothing systems use the MAC operation to create a smoothed output image. Using the next convolution kernel (G) [12], the image smoothing performs a Gaussian smoothing on the input image.

To generate pixels in the output image, the filter is slid over the input image pixel by pixel. The sum of products obtained by multiplying the filter weights by the current subset of input image pixels yields the corresponding pixel in the output image.

$$SI_m(i,j) = \frac{1}{273}\sum_{k=-2}^{2}\sum_{l=-2}^{2} G(k+2,l+2) * I_n(i+k,j+l) \tag{17}$$

Where, $SI_m(i,j) and I_n(i,j)$ are each pixel of the input and the smoothed images.

In the convolutions, only MAC operations are approximated. This application is written in the Verilog programming language. The inputs are 512x512 gray scale bitmaps of the popular Lena image with 8-bit pixels. Fig 4 shows the image smoothing results for these designs and we can observe that the proposed designs provides output images. We also study the Peak Signal to Noise Ratio (PSNR) and Structural

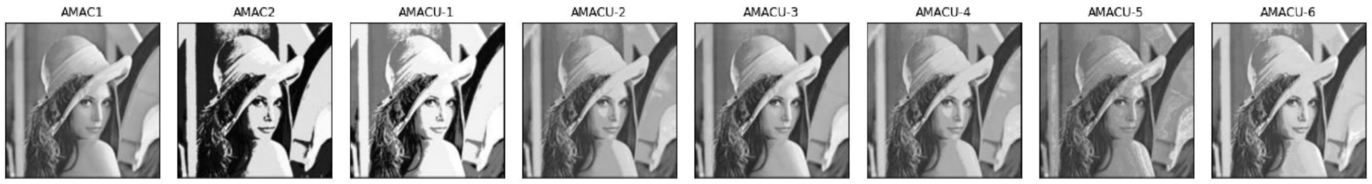


Fig. 4: Original and Smoothened images

Similarity Index (SSIM) values to determine the accuracy of the MAC operations in this application and the results are plotted in Fig 3(c).

VII. Conclusion

This paper proposed six variations of approximate MAC units AMACU-1 to AMACU-6. The AMAC design introduces approximation in both the multiplication stage and the accumulation stage of the MAC unit. Simulation results show that proposed multipliers exhibit a sensible reduction in NMED compared to AMAC2 [13] and up to 32% on an average and 47% in particular with respect to proposed AMACU-1. Exact multipliers and approximate multipliers proposed in [12], [13] are compared with the proposed Approximate MAC units- AMACU-1 to AMACU-6 in terms of error performance and overhead analysis. The APP and PDP of the proposed architectures are reduced by 87.65% and 77.3% with respect to Exact MAC, by 74.54% and 60.5% with respect to AMAC1 [12] and 30.4% and 32.6% with respect to AMAC2 [13].

The proposed and state-of-art designs were then used in an image-smoothing application which used Gaussian kernel. The proposed designs performed well in smoothing the image with good PSNR and SSIM values. Overall, it can be concluded that approximating the design will trade-off APP and PDP for accuracy. Ultimately it depends on what the application demands. From analysis, we conclude that if the application demands low APP then AMACU-1 is the best choice whereas if the requirement is low NMRED then also AMACU-1 is preferred. But when the optimization parameter is high PSNR we recommend to choose AMACU-2 or AMACU-4 and for cases with high SSIM it is recommended to use AMACU-3.

VIII. Acknowledgement

We acknowledge ISEA (Information Security Education and Awareness) Project Phase II for the support and funding.

References

[1] Chippa VK, et al. ”Analysis and characterization of inherent application resilience for approximate computing.” Proceedings of the 50th Annual DAC, pp. 1-9, 2013.
[2] Ye R, et al. ”On reconfiguration-oriented approximate adder design and its application.” ICCAD, IEEE, pp. 48-54, 2013.
[3] Cilardo A, et al. ”High speed speculative multipliers based on speculative carry-save tree.” IEEE Transactions on Circuits and Systems, pp. 3426-3435, 2014.
[4] Horowitz M, ”1.1 Computing’s energy problem (and what we can do about it),” ISSCC, IEEE, pp. 10-14, 2014.
[5] King EJ and Swartzlander EE. ”Data-dependent truncation scheme for parallel multipliers.” Conference Record of the 31st ACSSC, Vol. 2. IEEE, pp. 1178-1182, 1997.
[6] Han J and Orshansky M. ”Approximate computing: An emerging paradigm for energy-efficient design.” 18th ETS, IEEE, pp. 1-6, 2013.
[7] Weste NH, and Harris D. CMOS VLSI design: a circuits and systems perspective, Pearson Education India, 2015.
[8] De Caro D, et al. ”Fixed-width multipliers and multipliers-accumulators with min-max approximation error.” IEEE Transactions on Circuits and Systems, pp. 2375-2388, 2013.
[9] Guo Y, et al. ”Low-cost approximate multiplier design using probability-driven inexact compressors.” APCCAS, IEEE, pp. 291-294, 2018.
[10] Shafique M, et al. ”A low latency generic accuracy configurable adder.” 52nd DAC, IEEE, pp. 1-6, 2015.
[11] Abdelgawad A, and Bayoumi M. ”High speed and area-efficient Multiply Accumulate (MAC) unit for digital signal processing applications.” ISCAS, IEEE, pp. 3199-3202, 2007.
[12] Yang T,et al. ”A low-power approximate multiply-add unit.” ISDCS, IEEE, pp. 1-4, 2019.
[13] Yang T et al. ”A Low-Power and Small-Area MAC Unit for Accuracy-Scalable Approximate Computing.” Fukuoka University review of technological sciences, pp. 1-8, 2020.
[14] Esposito D et al.”Low-power approximate MAC unit.” 13th Conference on PRIME, IEEE, pp. 81-84, 2017.
[15] Yang T et al. ”A low-power configurable adder for approximate applications.” 19th ISQED, IEEE, pp. 347-352, 2018.
[16] Adams E. ”Energy-efficient approximate MAC unit.” ISCAS, IEEE, pp. 1-4, 2019.
[17] Tamilselvan S and Arun A. ”An efficient MAC design for image processing application.” Indian Journal of Science and Technology, 2018.
[18] Cho K, et al. ”eDRAM-based tiered-reliability memory with applications to low-power frame buffers.” ISLPED, IEEE, pp. 333-338, 2014.
[19] Ganapathy S, et al. ”Mitigating the impact of faults in unreliable memories for error-resilient applications.” 52nd DAC, IEEE, pp. 1-6, 2015.
[20] Gupta V, et al. ”IMPACT: IMPrecise adders for low-power approximate computing.” ISLPED, IEEE, pp. 409-414, 2011.
[21] Tian Y, et al. ”ApproxMA: Approximate memory access for dynamic precision scaling.” Proceedings of the 25th edition on Great Lakes Symposium on VLSI, pp. 337-342, 2015.
[22] Raha A, et al. ”Quality configurable reduce-and-rank for energy efficient approximate computing.” DATE, IEEE, pp. 665-670, 2015.
[23] Chippa VK, et al. ”Scalable effort hardware design.” IEEE Transactions on Very Large Scale Integration Systems, pp. 2004-2016. 2014.
[24] Roldao-Lopes A, et al. ”More flops or more precision? Accuracy parameterizable linear equation solvers for model predictive control.” 17th IEEE Symposium on FCCM, pp. 209–216, 2009.
[25] Liu C et al. ”A low-power, high-performance approximate multiplier with configurable partial error recovery.” DATE, IEEE, pp. 1-4, 2014.