Int. J. Electron. Commun. (AEÜ) 172 (2023) 154887

Contents lists available at ScienceDirect

Int. J. Electron. Commun. (AEÜ)

journal homepage: www.elsevier.com/locate/aeue

Design space exploration for energy-efficient approximate Sobel filter

Alain Aoun [a,*], Mahmoud Masadeh [b], Sofiène Tahar [a]

[a] Electrical and Computer Engineering, Concordia University, Montréal, Canada
[b] Computer Engineering Department, Yarmouk University, Irbid, Jordan

ARTICLE INFO

Keywords:
Approximate computing
Approximate full adders
Approximate arithmetic
Sobel filter
Digital circuits
Energy efficiency
Edge detection

ABSTRACT

Approximate computing (AC) is an emerging computing paradigm for energy efficiency. AC is most suitable for error-tolerant applications, e.g., image processing. The Sobel filter is an edge detector which is used heavily in image processing. One of the basic blocks in the hardware implementation of the Sobel filter is the full adder (FA), which approximation can greatly reduce the energy consumption of the filter. In this paper, we propose three new Non-exact FAs (NeFAs) that are suitable for image processing. The proposed NeFAs along with existing approximate FAs are used to create a library of approximate FAs. We use this library to perform a design space exploration (DSE) of the approximate Sobel filter, which is an essential step when searching for an optimized implementation. Experiments have shown that the executed DSE was able to achieve a target reduction of up to 75% in area and power. We analyzed the generated designs objectively and subjectively. Using the subjective assessment, we defined two Pareto optimal criterion where we found that the implementations based on the proposed NeFA are in the Pareto optimal for high target reduction, i.e., most efficient designs. Based on the objective assessment, we found that the NeFA-based designs achieve outstanding quality and produce finer edges than the exact design in some cases.

1. Introduction

Approximate computing (AC), known as best-effort computing, is a nascent computing paradigm that sacrifices accuracy where imprecise results are acceptable. AC generates designs with a reduced area, power, delay, or energy. Various applications, e.g., image processing, machine learning, and digital signal processing, show intrinsic error tolerance due to various factors, namely, (*i*) redundant and noisy input data; (*ii*) lack of golden or single output; (*iii*) imperfect perception in the human sense; and (*iv*) implementation algorithms with self-healing and error attenuation patterns. Research in the field of AC has investigated the approximation of the input data, e.g., approximate load value [1], or the approximation of the arithmetic operations such as the work in [2]. Arithmetic operations, e.g., addition, subtraction, multiplication, division, multiply-and-accumulate (MAC), squaring, and square root, received great attention for approximation. In this work, we target the addition operation, where we propose three new approximate full adders (FAs) suitable for image processing applications.

Application of artificial intelligence, e.g., self-driving cars, has been in high demand recently where these applications have a great dependence on computer vision. Objects in an image can be detected by a computer using various techniques such as edge detection [3]. Edges of images represent a jump intensity between adjacent pixels. Edge detection preserves the fundamental structural properties of an image while reducing the amount of data and eliminating nonessential information. Existing techniques of edge detection belong to two main classes: (1) *Gradient*: such as the Sobel filter, which detects edges by locating the minimum and the maximum in the first derivative of the image [4]; and (2) *Laplacian*: such as Gaussian filter, which looks for zero crossings in the second derivative of the image [5].

Edges are points in the image with a sudden change in color. It indicates a transition between objects or a transition between an object and its background. Edge detection includes noise reduction, edge enhancement, and edge localization. For instance, the Sobel filter detects and emphasizes edges [4]. It calculates the gradient of image intensity at each pixel and finds the direction of the largest increase from light to dark and the rate of change in that direction. The result shows how suddenly the image changes at each pixel, and the probability that pixel defines an edge. The Sobel filter utilizes two 3×3 kernels/masks matrices, given as M_x and M_y as follows:

$$M_x = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad M_y = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

where the M_x kernel evaluates changes in the horizontal direction, while the M_y kernel evaluates changes in the vertical direction. By

* Corresponding author.
E-mail address: a_alain@ece.concordia.ca (A. Aoun).

https://doi.org/10.1016/j.aeue.2023.154887
Received 12 May 2023; Accepted 31 August 2023
Available online 24 September 2023
1434-8411/

convoluting the image with two kernels, it is possible to approximate the derivatives, especially when using small integer values, which makes this approach computationally efficient. This method is commonly applied in autonomous systems with a sensor camera for frame preprocessing, where the Sobel filter is a popular choice. By using the Sobel filter, it becomes possible to perform feature extraction in real-time, making it an ideal for collision avoidance [6].

of Circuits, Systems and Computers The Sobel filter implementation heavily relies on addition operations where FAs are the basic building blocks for most of its hardware. Therefore, due to their significant importance, we propose three Non-exact FAs (NeFAs) that are most suitable for image processing. The proposed NeFAs are competitive with other existing approximate FAs in terms of resource usage and quality. Therefore, with the aim of identifying the most suitable approximate implementation of the Sobel filter, we perform a design space exploration (DSE). During the DSE process, we look into designs achieving various savings in resource usage while maintaining an acceptable level of quality. The quality is assessed objectively and subjectively. The objective quality assessment relies on the multi-scale structural similarity (MS-SSIM) [7] factor. The subjective assessment is performed by visually inspecting the quality. The performed DSE allows us to identify the most suitable approximation settings, i.e., the proper approximate FA and the optimal approximation level.

In the remainder of this paper, we will discuss the related work in Section 2. Thereafter, we will present the proposed NeFAs in Section 3 followed by the proposed methodology for DSE of approximate Sobel filter in Section 4. In Section 5 we will present an analysis of the generated approximate designs in terms of resource usage and quality, i.e., objective and subjective assessment. We conclude this work in Section 6.

2. Related work

In this section, we highlight some of the literature work that is most related to the work proposed in this paper, namely, (*i*) approximation of the Sobel filters; and (*ii*) design of approximate full adders.

2.1. Approximate Sobel filter

In [8], the authors presented different approximations of the Sobel filter using truncation techniques. They compared the output when the data size is reduced by one bit to the exact output, and found a match. However, a significant loss of quality was observed when the data size was reduced by two bits. When the data size was reduced by three bits, the output of the filter was barely recognizable. In [9], the authors investigated workload-aware approximate computing applied to the Sobel filter. Their design of the Sobel filter featured 18 multipliers and 17 adders, each with different settings. They discovered that even with the same approximation configuration, different input workloads could produce varying output qualities.

A study conducted in [10] utilized a Sobel filter application to evaluate 10 designs of approximate full adders (FAs). The authors observed that an approximate adder logic with the appropriate approximate bits can be used for edge detection filtering, resulting in satisfactory error rates and image quality. In another approach proposed in [11], the authors introduced Learned Approximate Computing (LAC), which focused on optimizing application kernels instead of hardware approximations. The effectiveness of LAC was assessed through various applications, including the 3×3 Sobel filter for edge detection. Additionally, the authors of [12] extended a RISC-V processor by incorporating a variable bit-width memory unit alongside the existing variable bit-width arithmetic units. They evaluated the impact of both variable bit-width arithmetic and memory units on the output accuracy and energy consumption of the Sobel filter. In [13], the authors explored the usage of approximate parallel-prefix adders (PPA) in the Sobel operator. They indicated that the CRA-based filter is the most energy efficient. However, PPA filters are faster than CRA designs.

2.2. Approximate full adders

The hardware implementation of the Sobel filter relies heavily on the addition operation. Thus, one approach of investigating an approximate implementation of the Sobel filter is to approximate the addition operation. Thus, we dedicate this section to present related work that targeted the approximation of the adders. The authors of [14] designed a memristor-based approximate FA and subtractor with a logic minimization technique. The proposed design is verified by developing an 8-bit Carry Ripple Adder (CRA) to perform bitwise pixel addition of two grayscale images. Similarly, the designed 8-bit Ripple Borrow Subtractor (RBS) is verified on foreground detection. In [15], the authors proposed a novel design of approximate 4–2 compressors. They presented a modified architecture of the Dadda Multiplier [16] and effectively utilized the proposed compressor with a reduced error at the output. The proposed compressor consists of a FA and two 2-to-1 multiplexers. Similarly, the authors of [17] presented an efficient imprecise 4:2 and 5:2 compressors by changing the truth table of the exact compressors to gain simpler logic functions with fewer output errors. Then, efficient approximate multipliers are designed based on the presented inexact compressors.

The authors of [18] proposed three approximate FAs with acceptable accuracy, low power, and low delay, where they evaluated the effects of die-to-die (D2D) process variation on the threshold voltage of approximate FAs. They utilized the proposed approximate FAs in the ripple carry adder structure and image Sharpening algorithm. The authors of [19] enhanced the accuracy of approximate multipliers using a set of proposed approximate compressors which can compress any number of inputs to arbitrary numbers of output bits. They used the probability of being one in the input bits to determine the compression ratio. The authors of [20] proposed six novel 1-bit approximate FAs where they compared their performance against the related work, which showed that two proposed approximate FAs are competitive.

In [2], the authors approximated the mirror-based FA and called it AMA. They generated five AMAs by eliminating a set of transistors in each iteration of the approximation process. In the first iteration, they generated AMA1 which is the closest to the exact implementation. Furthermore, the last iteration generated AMA5 which is a buffer. Based on the investigation in [21], the AMA showed superiority. Subsequently, we only include the AMAs in the library of approximate FAs used in the DSE of this paper.

Previous studies have mainly focused on approximation techniques for arithmetic units. Thereafter, the overall performance of these units was evaluated by using the Sobel filter as an application. However, as the Sobel filter is critical in image processing, finding the optimal approximated version is crucial. Therefore, in this paper, we aim to perform a DSE to determine the most appropriate design settings of the approximate Sobel filter.

3. Proposed approximate full adders

The edge detection using the Sobel filter heavily relies on arithmetic addition. Subsequently, the hardware implementation of the Sobel filter will consist of a large number of FAs. Thus, developing an approximate FA that greatly reduces resource usage while achieving good quality is direly needed to optimize the implementation of the approximate Sobel filter. To this aim, we propose three Non-exact FAs (NeFA) that are most suitable for the approximation of the Sobel filter. Fig. 1 depicts their schematic diagrams, i.e., NeFA1, NeFA2 and NeFA3. The NeFA1 is a result of eliminating four logic gates from the exact FA, i.e., $1 \times$ XOR, $1 \times$ OR and $2 \times$ AND gates. In addition, since in a CRA the carry generation is in the critical path, the carry out is assigned as the second input. In another direction of designing the NeFA, we use NAND and NOR gates as they are basic gates with minimal resource usage, i.e., require four transistors. Moreover, the quality of the output is improved by adding an Inverter, i.e., two transistors, to the circuit.

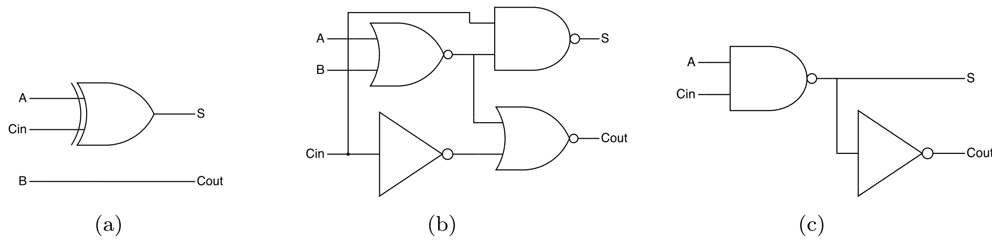

Fig. 1. Schematic of the proposed (a) NeFA1, (b) NeFA2, and (c) NeFA3.

Table 1
Truth table for the proposed approximate full adders.

Inputs			NeFA1			NeFA2			NeFA3		
A	B	Cin	Cout	Sum	ED	Cout	Sum	ED	Cout	Sum	ED
0	0	0	0	0	0	0	1	−1	0	1	−1
0	0	1	0	1	0	0	0	1	0	1	0
0	1	0	1	0	−1	0	1	0	0	1	0
0	1	1	1	1	−1	1	1	−1	0	1	1
1	0	0	0	1	0	0	1	0	0	1	0
1	0	1	0	0	2	1	1	−1	1	0	0
1	1	0	1	1	−1	0	1	1	0	1	1
1	1	1	1	0	1	1	1	0	1	0	1

Table 2
Synthesis of the library of basic arithmetic units.

Design	Area (μm^2)	Power (μW)	Delay (fs)
NeFA1	3.60	0.858	40
NeFA2	5.04	0.606	30
NeFA3	2.52	0.210	20
AMA1 [2]	8.28	1.624	50
AMA2 [2]	3.96	0.466	40
AMA3 [2]	3.24	0.333	30
AMA4 [2]	3.60	0.459	30
AMA5 [2]	2.16	0.145	0
Exact	10.08	2.716	80

The truth table of the NeFAs is shown in Table 1 where the error distance (ED), which represents the difference between the exact and approximate values, is shown. A positive ED indicates that the exact FA gives a higher value than the approximate. However, the negative ED indicates that the approximate FA has a higher value than the exact FA. Error cancellation due to positive and negative errors proved efficiency in enhancing the quality of approximate designs [22]. The three proposed NeFA, i.e., NeFA1, NeFA2 and NeFA3, produce an average ED of 0, 0.125, and 0.25 and a maximum ED of 2, 1 and 1, respectively. The error rate (ER) is the count of erroneous outputs divided by the total number of outputs, i.e., 2^3 for a FA. From Table 1 we determine ERs of 62.5%, 62.5% and 50% for NeFA1, NeFA2, and NeFA3, respectively.

In addition to having an acceptable quality, the resource usage of an approximate FA is important. For this purpose, we performed the synthesis of the proposed NeFAs in addition to the five approximate mirror adders (AMA) proposed in [2]. The synthesis is performed using *Synopsys Design Vision* [23] and TSMC 65 nm CMOS technology [24]. The resource usage of the eight approximate FAs is shown in Table 2. Based on the synthesis results, we notice that the proposed NeFAs are highly competitive with the existing AMAs. The obtained area, power, and delay fall within the minimum and maximum values of the related designs with a great reduction compared to the exact design.

4. Design space exploration of approximate Sobel filter

Approximate computing is based on the principle of significance-driven approximation, making it important to identify the parts of a system that can be approximated and determine the appropriate approximation settings [25]. If the most significant bits (MSBs) of a hardware design are approximated, it can lead to poor quality. Our proposed approach aims to limit the negative impact of approximation on the MSBs while maximizing the benefits of using approximate hardware. We search for designs that result in minimal quality loss while achieving a maximum reduction in resource usages, such as area and power.

The Sobel filter includes various arithmetic operations, e.g., multiplication, addition, and square root. We introduce approximation into multiple operations to generate an approximate filter with a specified design metric, e.g., a design with 45% less power consumption. For instance, as shown in Fig. 2, we propose to design an approximate filter by approximating its basic operations with specific configurations, e.g., swapping 10 exact FAs with approximate ones in a 16-bit CRA. Accordingly, we generate suitable designs for a selected reduction in a given metric. In this work, we generate hardware solutions that shrink area and power usage where the target reduction ranges from 5% to 95% with a step of 5% for each metric. The reductions are achieved using the method proposed in [26] which solves for *position independent* replacement. The *position independent* replacement relies on measuring the resource usage saving, i.e., area and power, based on the quantity while neglecting the placement and routing. For example, if a circuit has eight exact FAs, when replacing only one exact FA with its approximated counterpart, the total area and power usage are projected to be the same regardless of the position of the FA that was replaced. The equations that solve for *position independent* replacement are:

$$M_T = \sum_{i=1}^{n} Q_i \times M_i \tag{1}$$

$$Q_T = \sum_{i=1}^{n} Q_i \tag{2}$$

where n: the types of basic cells, M_T: the total resource usage, Q_i: the quantity of a given unit, e.g., exact full-adder, Q_T: the total quantity, and M_i is the correspondent hardware usage, e.g., area or power usage, of a given unit. Eq. (1) estimates the total resource M_T, e.g., total area usage, by summing the product of the quantity and the resource usage of each basic component. Furthermore, Eq. (2) looks at the total quantity of basic components. For instance, if a circuit contains eight FAs, the quantity of approximate and exact FAs must be eight. We applied *position independent* replacement and generated various implementations of the approximate Sobel filter. For that, we replace the exact FAs, with AMA1-AMA5 [2]. Moreover, we utilize our newly proposed approximate FAs, i.e., NeFA1, NeFA2, and NeFA3.

As shown in Fig. 2, as a first step, the hardware usage of the basic cell (M_i) has to be identified. This step is required to solve Eq. (1). The results of the synthesis are summarized in Table 2. Then, we decide the Q_T required to implement the hardware of the Sobel towards solving Eq. (2). As depicted in Fig. 3, the Sobel circuit receives eight inputs, i.e., A, B, C, D, E, F, G and H, illustrating the bordering pixels of the targeted pixel. The Sobel circuit includes twelve 8-bit CRA units, two 8-bit 2's complement units, two 8-bit squaring (SQR) units, one 16-bit CRA unit and one 16-bit square root unit. In this paper, the square root unit is exact, and the 2's complement is performed using inverters and 8-bit CRA. Thus, the total number of 8-bit CRA units is 14. Also, the

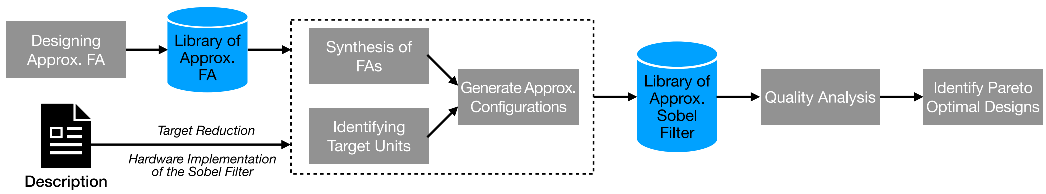

Fig. 2. Proposed methodology to generate approximate Sobel filter.

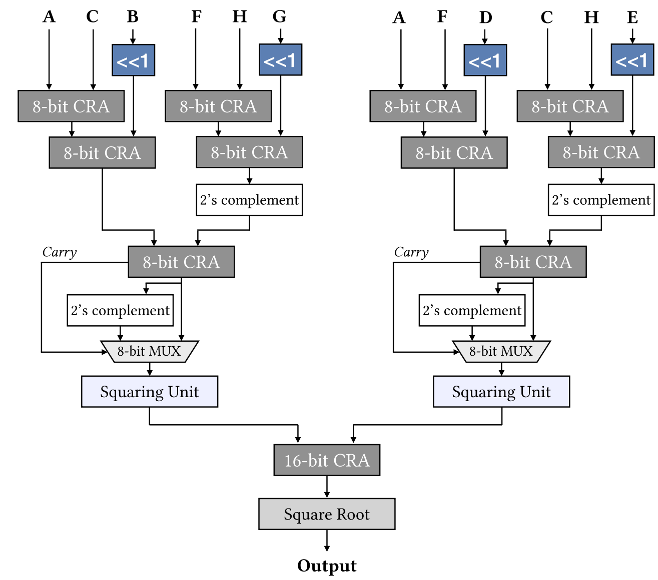

Fig. 3. Hardware implementation of the Sobel filter.

Table 3
Quantities of exact FAs in an 8-bit CRA for various target reduction.

Target Reduction	NeFA1	NeFA2	NeFA3	AMA1	AMA2	AMA3	AMA4	AMA5
5%	7	7	7	5	7	7	7	7
10%	6	6	6	3	6	6	6	6
15%	6	5	6	1	6	6	6	6
20%	5	4	5	−1	5	5	5	5
25%	4	4	5	−4	4	5	4	5
30%	4	3	4	−6	4	4	4	4
35%	3	2	4	−8	3	3	3	4
40%	3	1	3	−10	2	3	3	3
45%	2	0	3	−13	2	2	2	3
50%	1	0	2	−15	1	2	1	2
55%	1	−1	2	−17	0	1	1	2
60%	0	−2	1	−19	0	0	0	1
65%	−1	−3	1	−22	−1	0	−1	1
70%	−1	−4	0	−24	−2	−1	−1	0
75%	−2	−4	0	−26	−2	−1	−2	0
80%	−2	−5	−1	−28	−3	−2	−2	−1
85%	−3	−6	−2	−31	−4	−3	−3	−1
90%	−4	−7	−2	−33	−4	−3	−4	−2
95%	−4	−8	−3	−35	−5	−4	−4	−2

number of FAs is eight, sixteen and thirty-one in the 8-bit CRA, 16-bit CRA and SQR units, respectively. Thereafter, we define the quantities of FAs in the Sobel hardware as $Q_T = 190$.

Using the synthesis results of Table 2 and Q_T, we solve Eqs. (1) and (2) for various target reductions in order to identify the number of exact and approximate FAs in each unit that satisfies the required reduction. We solve the two equations to find the configuration that achieves the chosen reductions in terms of area and power, simultaneously. For example, for a 25% reduction goal, we seek to gain a reduction of at least 25% in both area and power. In this work, we target homogeneous approximate designs where only one type of approximate basic cell is considered. Thus, for Eqs. (1) and (2), we determine $n = 2$ (exact and approximate).

Table 3 shows the quantities of exact and approximate FAs in the 8-bit CRA for a provided target reduction when using a specific approximate FA. The given number indicates the number of exact FAs located in the most significant bits (MSBs) while the remaining are approximate FAs located in the least significant bits (LSBs). As shown in Table 3, solving Eqs. (1) and (2) resulted in a negative quantity of FAs, i.e., quantities highlighted in red, for some target reduction. This indicates that the target reduction cannot be achieved using this type of approximate FA. The trend of negative quantities is the same across the three units, i.e., 8-bit CRA, 16-bit CRA and SQR units. Thus, Table 3 is illustrative to a success or failure of a given configuration. From Table 3, we notice that all designs (AMA1-AMA5, NeFA1-NeFA3) can achieve a reduction of 5%, 10% and 15%. On the other hand, none of the designs can achieve a reduction of more than 75%. A reduction of 70% and 75% is achievable by two designs only. One design is based on AMA5 and the other design is utilized on NeFA3. All of the designs, except the AMA1-based design, can achieve a reduction from 20% to 50%. All of the designs, except the ones based on AMA1 and NeFA2, can achieve a reduction of 55% to 60%. A 65% reduction is achieved by three designs only, i.e., based on NeFA3, AMA3, and AMA5

5. Design analysis

In this section, we discuss the hardware resource usage and the output quality of the various approximate Sobel filter implementations that are generated by the DSE performed in this paper. We compare them with the exact design to determine their effectiveness. The generated implementations of the Sobel filter are modeled in VHDL and Matlab [27] for synthesis and quality analysis, respectively.

5.1. Resource usage

The resource usage of the approximate Sobel filter designs generated by the DSE is estimated using *Synopsys Design Vision* [23] and TSMC 65 nm CMOS technology [24]. The synthesis results are summarized in Fig. 4. As shown in Table 4 we found that in average, the achieved reduction in area is almost equal to the target reduction, i.e., actual reduction ≈ targeted reduction. These measurements confirm the effectiveness of the proposed DSE. On another note, from Table 4 we can notice that the actual power reduction, i.e., static, dynamic and total power, exceeds the target reduction since all approximate FAs in the library offer greater reductions in power compared to the area. Furthermore, even though the *position independent* replacement does not consider the delay, the actual reduction in delay also exceeded the target reduction. On the other hand, the designs are generated by solving Eqs. (1) and (2) for area and power simultaneously while assuring that the target reductions in the two aspects are satisfied. Subsequently, for a given target reduction more approximate FAs are required to achieve the aimed area reduction than the quantity needed to achieve the desired power reduction.

Fig. 4(a) shows the obtained area for all generated designs. We can note that for a given target reduction, all designs achieve a similar area usage except when AMA5 is used. This is due to the simple structure

Table 4
Target reduction versus actual reduction in resource usage.

Target (%)	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75
Area (%)	7.5	11.2	13.6	16.2	19.1	22.5	28.0	31.6	41.7	45.9	46.2	55.2	44.1	100	100
Delay (%)	8.6	16.2	19.6	27.1	31.6	37.2	43.1	48.8	53.9	60.4	62.9	69.5	74.8	82.7	84.9
Static power (%)	9.2	22.3	27.5	33.6	39.9	47.8	56.9	65.0	71.1	78.7	79.9	88.8	92.8	99.9	99.9
Dynamic power (%)	9.9	23.7	29.1	35.0	41.5	49.6	58.6	66.7	72.6	80.1	81.4	89.7	92.6	100	100
Total power (%)	9.4	22.4	27.6	33.7	40.0	47.8	56.8	64.8	70.8	78.4	79.6	88.3	92.1	99.3	99.4

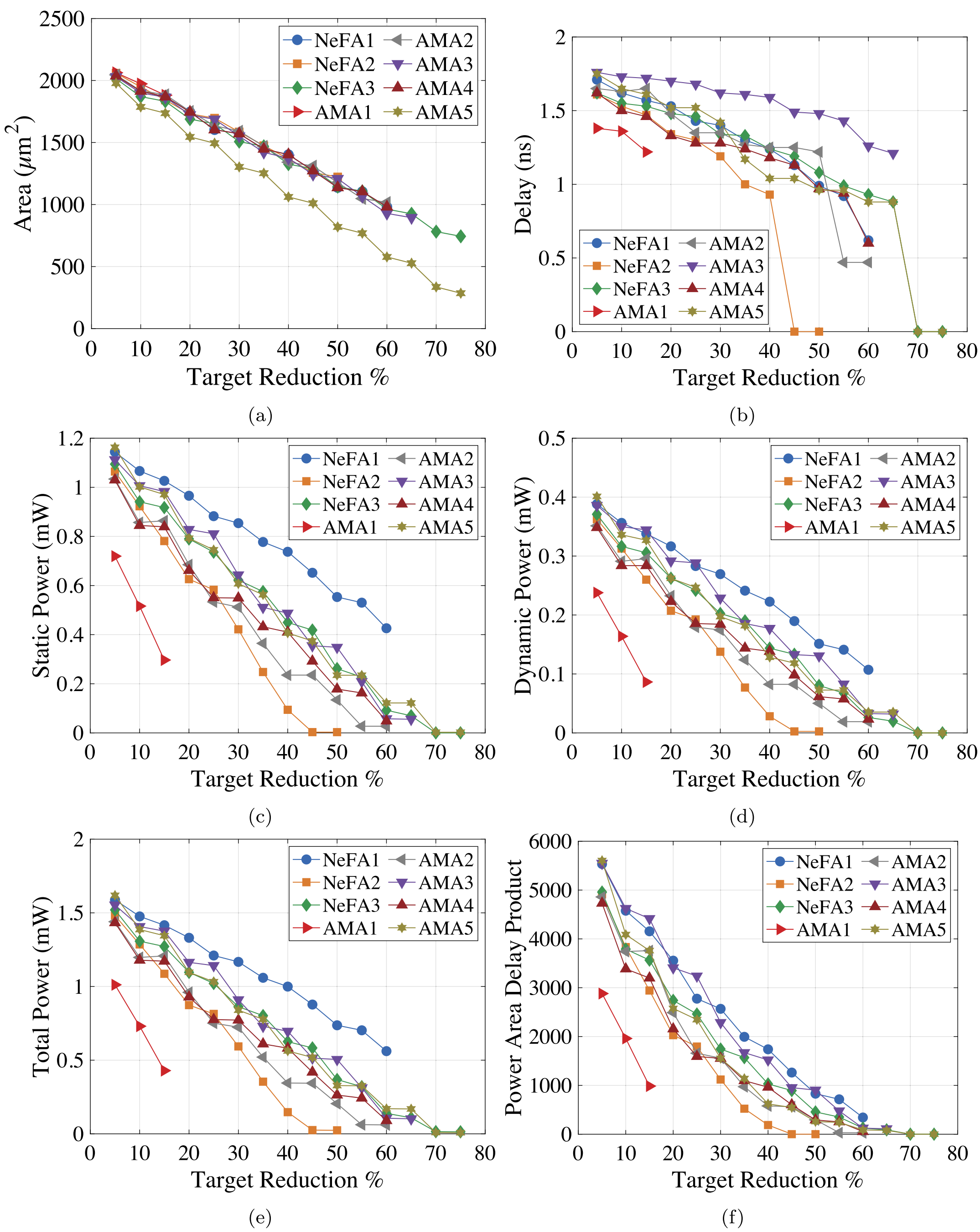

Fig. 4. (a) Area, (b) Delay, (c) Static power, (d) Dynamic power, (e) Total power, and (f) Power area delay product of the designs generated by the DSE.

of AMA5, i.e., buffer, which when cascaded does not increase the area usage linearly, i.e., the hardware can be simplified. The obtained delay of all proposed designs is shown in Fig. 4(b). For a target reduction ranging from 5% to 15% the implementation based on AMA1 has the lowest delay. Moreover, for a target reduction of 5% to 65% the implementation based on AMA3 results in the longest delay.

Figs. 4(c), 4(d) & 4(e) shows the static, dynamic and total power consumption of the generated designs, respectively. From these figures, we can notice that the applied *position independent* replacement will result in an even reduction in static and dynamic power. The approximate Sobel filters based on AMA1 have minimal power for a reduction of 5%, 10%, and 15%. However, for other target reductions, the DSE is unable

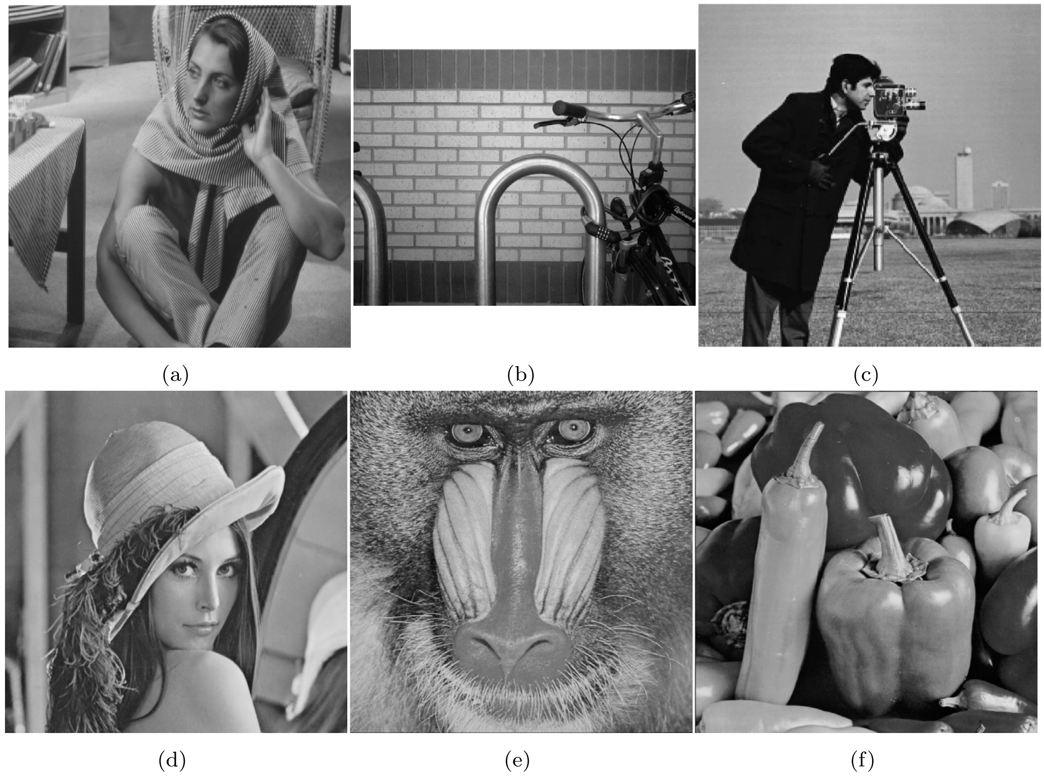

Fig. 5. Six benchmarks images: (a) barb, (b) bikesgray, (c) cameraman, (d) lena, (e) mandrill, and (f) peppers.

to generate a design utilizing AMA1. For a reduction of 20% to 50%, the designs based on NeFA2 have minimal power. The Power-Area-Delay-Product (PADP) of the generated designs is shown in Fig. 4(f). The designs based on AMA1 and the proposed NeFA2 have minimal PADP for a target reduction ranging from 5% to 15% and 20% to 50%, respectively.

5.2. Quality analysis

The quality of an approximate design is equally important to resource usage. We dedicate this section to analyzing the quality of the 92 designs generated by the DSE. For this purpose, we use six well-known benchmark images as shown in Fig. 5, namely, "barb", "bikesgray", "cameraman", "lena", "mandrill", and "peppers". The quality assessment of the resulting edge detection is performed objectively and subjectively, i.e., visually. The multiscale structural similarity (MS-SSIM) [7] index, which evaluates the similarity of the input image in comparison to the reference image, is used to analyze the results objectively. The MS-SSIM value ranges from 0 to 1 where a greater value for MS-SSIM indicates a better output quality.

Objective Quality Assessment: When assessing the quality objectively, we noticed that the trend in the obtained MS-SSIM is independent of the input images. However, the quality depends on the utilized design. Thus, in the objective quality assessment we limit the interpretation to the average MS-SSIM or the overall trend. Fig. 6 shows the average MS-SSIM of the various benchmark images when using the various designs. In general, we can notice that for a higher target reduction, the quality decreases as more approximation is introduced into the design. However, in some cases, a higher target reduction, i.e., a higher level of approximation, resulted in better quality. For instance, the approximate Sobel filter design based on NeFA3 with a target reduction of 55% resulted in better quality than the design based on the same full adder, i.e., NeFA3, and a target reduction of 50%. This is due to the nature of the application, i.e., edge detection, where different algorithms may produce different results. Thus by introducing approximation, we

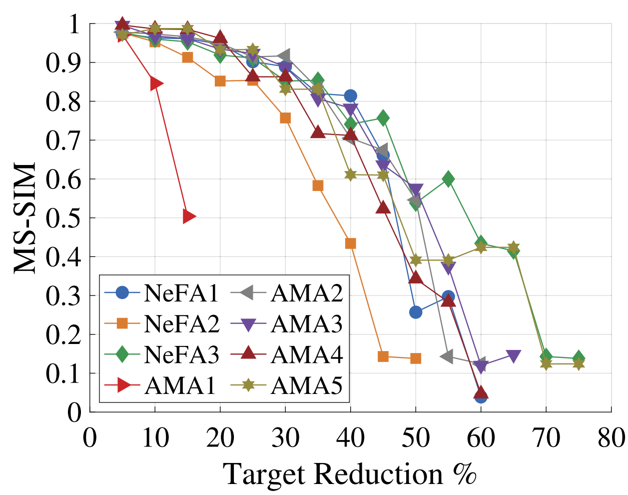

Fig. 6. Target reduction versus average output quality for the 92 designs generated by the DSE.

introduce a variation to the Sobel filter. Moreover, from Fig. 6 we can notice that a single approximate FA will not always lead to the highest quality.

We propose two methods to select the Pareto optimal configurations. The first selection method is based on the average MS-SSIM while in the latter the selection is based on the number of times a design achieves the best MS-SSIM in each of the six images, i.e., a voter-based selection. Table 5 shows the Pareto optimal design based on the two selection methods. For the first criterion, we notice the approximate filters based on AMA5 achieved the best quality in four cases, i.e., four different target reductions. However, for the second criterion, AMA2-based designs outperform AMA5-based designs. Furthermore, we notice that for a high target reduction, i.e., target reduction ≥40%, the designs

(a) (b) (c)

Fig. 7. Edge detection of "barb" using (a) Exact, (b) AMA5 (Target = 60%), and (c) NeFA3 (Target = 60%), Implementation of the Sobel filters.

Table 5
Pareto optimal designs.

Target reduction	1st Criterion	2nd Criterion
5%	AMA4	AMA4
10%	AMA5	AMA2
15%	AMA5	AMA2
20%	AMA2	AMA2
25%	AMA5	AMA5
30%	AMA2	AMA2
35%	AMA5	AMA5
40%	**NeFA1**	**NeFA1**
45%	**NeFA3**	**NeFA3**
50%	AMA3	AMA3
55%	**NeFA3**	**NeFA3**
60%	**NeFA3**	**NeFA3**
65%	**NeFA3**	**NeFA3**
70%	**NeFA3**	**NeFA3**
75%	**NeFA3**	**NeFA3**

based on our proposed approximate FAs, i.e., NeFAs, are the Pareto optimal except for a target reduction of 50%. Finally, we note that the best performing AMA-based design was at most four times in the Pareto optimal list. However, the proposed NeFA3 was six times on the Pareto list regardless of the selection criterion.

Subjective Quality Assessment: The previous analysis was based on the obtained MS-SSIM of the obtained designs. However, approximate computing has reemerged since the output quality of some applications is evaluated by human senses, e.g., the visual interpretation of images. From Fig. 7, we notice that the approximate Sobel filter based on NeFA3 and a target reduction of 60% outperforms the design based on AMA5 and the same target reduction. Comparing the results in Figs. 7(b) and 7(c) we notice that NeFA3 produced less noise around the head of "barb". Similarly, we can notice that the details of the face of "barb" is better detected using NeFA3 compared to AMA5. Another noticeable difference is the *grey* color of the background when using NeFA3 instead of the *black* background. This trend in the quality of edge detection when using the implementations based on AMA5 and NeFA3 with a target reduction of 60% is noticed among all the six benchmark images. However, the MS-SIM of the two designs with a target reduction of 60% showed equivalent performance in Fig. 6. Thus, a subjective quality assessment is inevitable when analyzing approximate Sobel filter designs.

Another example to illustrate the importance of subjective quality assessment is the edge detection results shown in Figs. 8 and 9. We can notice that the edge detection using the implementation of the approximate Sobel filter based on NeFA2 with a target reduction of 40% is analogous to the edge detection by the exact implementation. However, the average MS-SIM of this approximate design is only 0.44, i.e., poor quality. Moreover, from Figs. 8(a) and 9(a) we can notice that the approximate design may produce an edge detection with less noise, e.g., less noise on the ground. These observations are noticed among the three other benchmark images. Finally, from Fig. 10 we can notice that the usage of an approximate FA that is closest to the exact FA, e.g., AMA1, may not result in the best quality. For example, the implementation based on AMA1 produced noise around the bike lock, while the "more" approximate FA, i.e., NeFA2, produced finer edges. Based on the aforementioned observations and based on the resource usage, we identify the approximate Sobel filter implementation based on NeFA2 and a target reduction of 40% to be a "*near golden*" implementation of an approximate Sobel filter.

6. Conclusion

Approximate computing (AC) is a relatively new computing paradigm. This approach aims to reduce the area, power, delay, or energy required by sacrificing a certain level of accuracy where imprecise results are deemed acceptable. AC is used in various applications such as image processing and machine learning. In this paper, we targeted the approximation of the Sobel filter which is a popular image-processing application. The hardware implementation of the Sobel filter relies heavily on full adders (FAs). Therefore, due to their significant importance, in this paper we proposed three new approximate FAs, called NeFAs, that are most suitable for image processing and competitive with existing approximate FAs. Subsequently, we generated a library of approximate FAs that consists of the proposed NeFAs and five well-known approximate FAs, i.e., AMA1-AM5 [2].

Design space exploration (DSE) is a crucial step in finding energy-efficient solutions for computationally demanding applications. In this paper, the performed DSE searches for implementations of approximate Sobel filter that satisfy a target reduction of at least 5% to 95% with a step of 5% in the area and power. The reduction is achieved by replacing the exact FA with an approximate counterpart from the library. The implementations of the approximate Sobel filter use a single type of approximate FA at a time, i.e., homogeneous approximation. The least significant bits (LSBs) are approximated while keeping the calculation of the most significant bits (MSBs) exact. The designs generated by the DSE are analyzed objectively and subjectively. In the objective quality assessment, we identified two Pareto optimal criteria. However, regardless of the selection criterion we deduced that the implementations based on the proposed NeFAs are in the Pareto optimal for high target reduction. Moreover, in the subjective quality assessment, we noticed that the implementations based on the proposed NeFAs achieved an outstanding quality where in some cases the NeFA-based implementation produced finer edges compared to the exact design, i.e., less noise. In a future work, we aim to enrich the library of approximate FAs and use more than one type at a time, i.e., heterogeneous approximation. Moreover, we aim to extend the library of benchmark images in order to offer a stronger quality analysis.

Fig. 8. Edge detection of (a) "camerman", (b) "lena", and (c) "peppers" using the exact implementation of the Sobel filters.

Fig. 9. Edge detection of (a) "camerman", (b) "lena", and (c) "peppers" using NeFA2 (Target = 40%) Implementation of the Sobel Filters.

Fig. 10. Edge detection of "bikesgray" using (a) Exact, (b) AMA1 (Target = 15%), and (c) NeFA2 (Target = 15%), Implementation of the Sobel filters.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data that has been used is confidential.

Acknowledgment

A preliminary version of this work has been published in ICM 2022 [28].

References

[1] Aoun A, Masadeh M, Tahar S. A machine learning based load value approximator guided by the tightened value locality. In: Great lakes symposium on VLSI. ACM; 2023, p. 679–84.
[2] Gupta V, Mohapatra D, Raghunathan A, Roy K. Low-power digital signal processing using approximate adders. IEEE Trans Comput-Aided Des Integr Circuits Syst 2012;32(1):124–37.
[3] Ferrari V, Fevrier L, Jurie F, Schmid C. Groups of adjacent contour segments for object detection. IEEE Trans Pattern Anal Mach Intell 2007;30(1):36–51.
[4] Sobel I, Feldman G. A 3 × 3 isotropic gradient operator for image processing. In: Pattern classification and scene analysis. 1973, p. 271–2.
[5] Basu M. Gaussian-based edge-detection methods-a survey. IEEE Trans on Systems, Man, and Cybernetics, Part C 2002;32(3):252–60.
[6] Kamçı S, Aksu D, Aydin MA. Lane detection for prototype autonomous vehicle. 2019, arXiv preprint arXiv:1912.05220.
[7] Wang Z, Simoncelli E, Bovik A. Multiscale structural similarity for image quality assessment. In: Asilomar conference on signals, systems & computers, Vol. 2. 2003, p. 1398–402.
[8] Rodrigues G, Lima Kastensmidt F, Bosio A. Survey on approximate computing and its intrinsic fault tolerance. Electronics 2020;9(4):557.

[9] Ma D, Thapa R, Wang X, Jiao X, Hao C. Workload-aware approximate computing configuration. In: Design, automation & test in europe. 2021, p. 920–5.

[10] Chung Y, Kim Y. Comparison of approximate computing with sobel edge detection. IEIE Trans on Smart Processing and Computing 2021;10(4):355–61.

[11] Gupta V, Li T, Gupta P. LAC: Learned approximate computing. In: Design, automation & test in europe. IEEE; 2022, p. 1169–72.

[12] Ndour G, Jost TT, Molnos A, Durand Y, Tisserand A. Evaluation of variable bit-width units in a RISC-V processor for approximate computing. In: International conference on computing frontiers. 2019, p. 344–9.

[13] Soares LB, da Rosa MMA, Diniz CM, da Costa EAC, Bampi S. Exploring Power-Performance-Quality Tradeoff of Approximate Adders for Energy Efficient Sobel Filtering. In: Latin American symposium on circuits & systems. 2018, p. 1–4.

[14] Muthulakshmi S, Dash CS, Prabaharan S. Memristor augmented approximate adders and subtractors for image processing applications: An approach. AEU-Int J Electron Commun 2018;91:91–102.

[15] Reddy KM, Vasantha M, Kumar YN, Dwivedi D. Design and analysis of multiplier using approximate 4-2 compressor. AEU-Int J Electron Commun 2019;107:89–97.

[16] Dadda L. Some schemes for parallel multipliers. Alta Freq 1965;34:349–56.

[17] Ahmadinejad M, Moaiyeri MH, Sabetzadeh F. Energy and area efficient imprecise compressors for approximate multiplication at nanoscale. AEU-Int J Electron Commun 2019;110:1–11.

[18] Mirzaei M, Mohammadi S. Low-power and variation-aware approximate arithmetic units for image processing applications. AEU-Int J Electron Commun 2021;138:1–13.

[19] Shirzadeh S, Forouzandeh B. High accurate multipliers using new set of approximate compressors. AEU-Int J Electron Commun 2021;138:1–8.

[20] Parameshwara M. Approximate full adders for energy efficient image processing applications. Journal of Circuits, Systems and Computers 2021;30(13):1–15.

[21] Masadeh M, Hasan O, Tahar S. Comparative study of approximate multipliers. In: ACM great lakes symposium on VLSI. 2018, p. 415–8.

[22] Mazahir S, Hasan O, Shafique M. Self-compensating accelerators for efficient approximate computing. Microelectron J 2019;88:9–17.

[23] Synopsys. Design compiler graphical. 2022, URL https://www.synopsys.com/implementation-and-signoff/rtl-synthesis-test/design-compiler-graphical.html. Last Accessed May 12, 2022.

[24] CMC Microsystems. TSMC 65nm GP CMOS process technology. 2023, URL https://www.cmc.ca/tsmc-65-nm-gp-cmos/. Last Accessed May 12, 2023.

[25] Masadeh M, Hasan O, Tahar S. Input-conscious approximate multiply-accumulate (MAC) unit for energy-efficiency. IEEE Access 2019;7:147129–42.

[26] Aoun A. On the improving of approximate computing quality assurance (Master's thesis), Concordia University; 2021, URL https://spectrum.library.concordia.ca/id/eprint/988392/. Unpublished.

[27] Mathworks. What is MATLAB?. 2022, URL https://www.mathworks.com/discovery/what-is-matlab.html. Last accessed May 12, 2022.

[28] Aoun A, Masadeh M, Tahar S. On the Design of Approximate Sobel Filter. In: International conference on microelectronics (ICM). IEEE; 2022, p. 102–6.