IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 30, 2021, accepted August 21, 2021, date of publication August 27, 2021, date of current version September 7, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3108443

A Novel Approximate Adder Design Using Error Reduced Carry Prediction and Constant Truncation

JUNGWON LEE, (Graduate Student Member, IEEE), HYOJU SEO, (Graduate Student Member, IEEE), HYELIN SEOK, (Student Member, IEEE), AND YONGTAE KIM, (Member, IEEE)
School of Computer Science and Engineering, Kyungpook National University, Buk-gu, Daegu 41566, South Korea

Corresponding author: Yongtae Kim (yongtae@knu.ac.kr)

This work was supported in part by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIT) under Grant NRF-2020R1A4A1019628, and in part by the Ministry of Education under Grant NRF-2019R1I1A3A01061266.

ABSTRACT This paper proposes a novel approximate adder that exploits an error-reduced carry prediction and constant truncation with error reduction schemes. The proposed adder design techniques significantly improve overall computation accuracy while providing excellent hardware efficiency. Particularly, the proposed carry prediction technique can reduce a prediction error rate by up to 75% compared to existing approximate adders considered in this paper. Furthermore, the error reduction technique also enhances the overall computation accuracy by decreasing the error distance (ED). Our experimental results show that the proposed adder improves the normalized mean ED (NMED) and mean relative ED (MRED) by up to 91.4% and 98.9%, respectively, compared to the other approximate adders. Importantly, an excellent design tradeoff allows the proposed adder to be the most competitive of the adders under consideration. Specifically, the proposed adder achieves up to 95.7%, 91.1%, and 93.2% reductions of the power-NMED, energy-NMED, and area-delay product (ADP)-NMED products, respectively, compared to the other adders. Our adder enhances the power-, energy-, and ADP-MRED products by up to 99.4% compared to the others. In particular, the figure of merit (FoM) considering both hardware and accuracy of the proposed adder is up to 93.05% smaller than that of the other approximate adders considered herein. Furthermore, we confirm that the approximation errors caused by the proposed adder have very little impact on output quality when adopted in practical applications, such as digital image processing and machine learning.

INDEX TERMS Approximate adder, approximate computing, carry prediction, constant truncation, error reduction.

I. INTRODUCTION

With the prevalence of battery-operated mobile and portable devices, power and energy consumption become the key constraint in system design because applications on these devices process a vast amount of computationally intensive information, such as multimedia (*i.e.*, image, video, and audio) processing, deep learning, data mining, and recognition, under a limited power and energy budget [1]–[6]. Many applications do not always require perfect computation accuracy [7]–[9]. For example, multimedia processing that involves human senses is error-tolerant. In other words, humans usually do not perceive the output quality degradation caused by computation errors on these applications, and a certain level of errors can be acceptable. The limitation of human perception offers an opportunity for a new computing paradigm, approximate computing, trading computation accuracy for power and energy [10]–[12]. Because adders are fundamental arithmetic components in computing systems, the design of efficient approximate adders is a practical way to enable approximate computing. Therefore, it has gained remarkable attention from researchers and a significant number of approximate adder designs have been presented in the technical literature [13]–[35]. We will review some existing approximate adders in Section II.

Approximate adders can be classified as block-based and full adder (FA)-based designs. Block-based approximate adders split an entire adder into smaller multiple sub-adders that perform partial additions concurrently [22]–[29]. The main idea of this approach is to cut a long carry propagation

The associate editor coordinating the review of this manuscript and approving it for publication was Cihun-Siyong Gong.

chain to achieve faster additions. However, it requires more area and power than FA-based approximate adders. FA-based adders use approximate 1-bit FAs to add some lower-order input bits approximately by replacing accurate FAs with approximate ones in the corresponding bit positions [13]–[21]. This improves the area and power performance at the expense of the computation accuracy degradation.

In this paper, we propose a new approximate adder design based on new approximate FA cells, enhanced carry prediction, and a constant truncation with error reduction. The proposed carry prediction scheme significantly reduces the prediction error rate by up to 75% compared to existing approximate adders considered here. Also, the truncation with error reduction logic enhances the overall computation accuracy while reducing energy and power consumption. When implemented in a 32-*nm* CMOS technology, the proposed adder is 1.49×, 1.90×, and 3.12× better area-, power-, and energy-efficient, respectively, than a traditional adder. Furthermore, compared to existing approximate adders, our adder improves overall computation accuracy by up to 98.9%. When jointly analyzing the adders in terms of hardware and accuracy, the proposed design is the most competitive among the adders considered.

In summary, this paper makes the following key contributions in designing approximate adders:

- We present a novel efficient approximate adder design that effectively trades off between hardware cost and computation accuracy through systematic analysis, and prove that our design outperforms the others by extensively comparing it with 12 approximate adders.
- We propose 1) a new carry prediction scheme that reduces the prediction error rate by up to 75% compared to the others, 2) approximate FA cells that improves accuracy, and 3) a constant truncation with an error reduction scheme that reduces hardware cost while offering good accuracy performance.

The remainder of this paper is organized as follows. Section II provides a brief review of existing approximate adders. In Section III, we present the proposed adder, which consists of our proposed approximate FAs, novel carry prediction, and constant truncation with error reduction. Illustrated examples of the adder operation and mathematical analysis of the carry prediction error rate and overall error rate are also provided. Then, Section IV explains the experimental results and systematic analysis of the proposed adder as well as extensive comparison with the 12 existing approximate adders. Also, a joint analysis of the adders in hardware and accuracy aspects is presented. In Section V, the application of the approximate adders to digital image processing and machine learning are presented. Finally, Section VI concludes the work.

II. RELATED WORKS

The lower-part OR adder (LOA) and error tolerant adder I (ETAI) are two representative approximate adders implemented using an approximate FA for the least significant bits (LSBs) of a multibit adder [13], [19], and many of their variants were presented so far [14]–[18], [20], [21].

The LOA consists of two parts: an accurate part and an inaccurate part [13]. The former part uses a traditional precise adder, such as the ripple carry adder (RCA) and carry-lookahead adder (CLA), to calculate the most significant bits (MSBs) with no computation error. Whereas, the latter part only uses an OR operation to approximately obtain LSB summations. Furthermore, the output of an AND operation for the MSB input pair of the inaccurate part is utilized as a carry input to the accurate part to improve overall computation accuracy. Design variants based on the LOA have been proposed to further optimize the LOA, such as LOA without the AND-based carry prediction (LOAWA), optimized lower-part constant OR adder (OLOCA), hardware optimized and error reduced approximate adder (HOERAA), hardware optimized adder having a near-normal error distribution (HOAANED), and hybrid error reduction lower-part OR adder (HERLOA) [14]–[18]. The LOAWA is identical to the LOA, except for the AND-based carry prediction [14]. In other words, the carry input to the accurate part is fixed to a constant "0," which degrades accuracy but improves the computation speed. The OLOCA is also similar to the LOA in that the OR operation is utilized for the inaccurate part approximation, but it outputs a constant "1" to a few LSBs regardless of the corresponding bit inputs [15]. This also degrades accuracy a bit while reducing hardware cost. In addition to the OLOCA, the HOERAA uses the OR operation for two MSB input pairs of the inaccurate part and sets the remaining LSB outputs to a constant "1" regardless of the inputs [16]. For the MSB output of the inaccurate part, it uses a 2-to-1 multiplexer to select "0" or an OR operation output of the corresponding input pairs. The multiplexer output is then used in an OR operation with the AND gate output of the second MSB input pair of the inaccurate part. Also, it includes an AND-based carry prediction for the accurate part, which also serves as the selection input of the multiplexer. The HOAANED is derived from the HOERAA by including one additional OR gate at the MSB of the inaccurate part [17]. This OR gate contributes to the improvement of an error metric, and thus, the HOAANED produces outputs with almost normal error distributions. To enhance overall computation accuracy, the HERLOA combines the basic LOA structure with the hybrid error reduction scheme [18]. Figure 1 shows the architecture of the inaccurate part of the HERLOA. Note that the accurate part is the same as the LOA (*i.e.*, precise adder). When the second MSB input pair of the inaccurate part is both "1," error reduction logic decreases the error distance (ED) by investigating the MSB input pair. The grayed gates in Figure 1 are the hybrid error reduction logic, while the others are the LOA logic. The error rate is reduced by replacing an OR gate at the MSB in the LOA with an XOR gate in the HERLOA.

The ETAI, like the LOA, divides an adder into two parts [19]. The inaccurate part of the ETAI utilizes its own modified XOR operation instead of the traditional

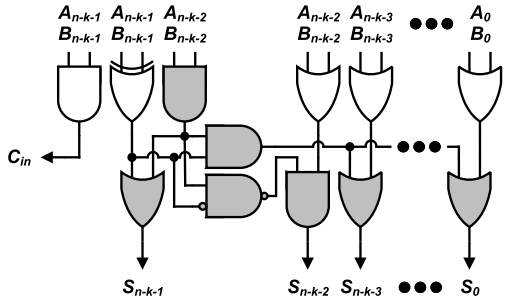


FIGURE 1. Architecture of inaccurate part of HERLOA [18].

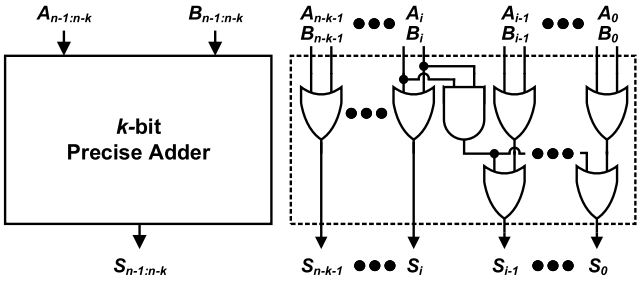


FIGURE 2. Block diagram of SETA [21].

OR operation. Note that the ETAI also uses a precise adder for MSBs additions. Furthermore, the carry prediction for the accurate part is a key difference from the LOA. The lack of prediction reduces accuracy while improving the speed. The carry predicting ETA (CPETA) was presented to improve computation accuracy by including a carry prediction scheme to the accurate part [20]. The CPETA adopts the AND-based carry prediction, which is the same as the LOA. Figure 2 shows a block diagram of the simplified ETA (SETA), which optimizes the ETAI's modified XOR operation to reduce hardware costs without a significant accuracy loss [21]. The modified XOR operation of the ETAI checks the input pairs from the MSB to LSB direction of the inaccurate part to check that both bits of the corresponding input pair are "1" whereas the SETA only checks a specific input pair to test if both bits are "1." This reduces hardware costs compared to the ETAI without significant accuracy degradation.

Different from the LOA, ETAI, and their variants that split an adder into two parts, other approximate adder structures have been presented in the literature as well. The reconfigurable approximate CLA (RAP-CLA) comprises several small sized blocks (*i.e.*, windows) that are overlapped each other to predict the carry of each bit position [22]. In other words, each carry is speculated by a sub-block to reduce the critical path delay by cutting the long carry propagation path. The block-based carry speculative approximate adder (BCSA) includes a number of non-overlapped blocks, each of which consists of a sub-adder, a carry predict unit, a select unit, and a multiplexer [23]. Each block's carry is predicted by either the carry predict unit or the sub-adder and selected by the selection unit. Additionally, the BCSA with its own error recover unit (BCSA_{ERU}) can improve the accuracy of the original BCSA without increasing the delay when an error occurs in a certain condition.

III. PROPOSED APPROXIMATE ADDER

This section presents our proposed FA cell-based approximate adder, which exploits a novel carry prediction scheme and a constant truncation technique to reduce the ED and improve overall computation accuracy. We call our adder the error reduced carry prediction approximate adder (ERCPAA). We denote two n-bit input operands and one n-bit output of the adder as $A_{n-1:0}$, $B_{n-1:0}$, and $S_{n-1:0}$, respectively. Also, A_i, B_i, and S_i represent the $(i)^{th}$ LSBs of $A_{n-1:0}$, $B_{n-1:0}$, and $S_{n-1:0}$, respectively.

A. OVERALL ADDER ARCHITECTURE

Figure 3 shows the overall hardware architecture of the proposed approximate adder with n-bit inputs. An n-bit adder is divided into two parts: a k-bit accurate part and an $(n - k)$-bit inaccurate part, where $k < n$. The accurate part simply consists of a k-bit precise adder that produces an accurate output (*i.e.*, $S_{n-1:n-k}$) from k MSB inputs (*i.e.*, $A_{n-1:n-k}$ and $B_{n-k:n-k}$) and a carry input (*i.e.*, C_{in}). The inaccurate part uses some of the remaining LSB inputs to generate an approximate output and a carry input to the precise adder. Note that the sizes of the accurate part and inaccurate part do not have to be equal, and the precise adder can be implemented in any type of traditional adders, such as RCA and CLA. The inaccurate part is further divided into three parts: an array of the proposed approximate FA cells, a carry prediction logic, and a constant truncation with error reduction logic. The proposed FA cell (see blue-highlighted box) simplifies the conventional single-bit FA cell to produce an approximate summation and an approximate carry, and is placed in some higher-order bit positions of the inaccurate part. The carry prediction logic, which is highlighted in green, generates the carry input to the precise adder. While most FA-based approximate adders employ an AND operation with the MSB inputs of the inaccurate part to produce the carry input, our prediction logic leverages the two MSB inputs to improve carry prediction accuracy at the cost of two additional logic gates. The constant truncation with error reduction logic highlighted in red sets l LSB outputs (*i.e.*, $S_{l-1:0}$) to either a constant "0" or "1" to reduce hardware costs depending on input conditions. In other words, the l LSB inputs are not used to generate approximate summations. It also assigns the other output bits except for the MSB of the inaccurate part to a constant "0" to reduce the ED under certain input conditions. We will describe the condition to determine to fix the output bits to "0" or "1" with illustrative examples in Section III-D.

B. PROPOSED APPROXIMATE FULL ADDER

An FA is the key building block for carry propagate adders (*e.g.*, RCA). The traditional 1-bit FA adds two inputs, A_i and B_i, as well as a carry from the previous bit position C_{i-1} and

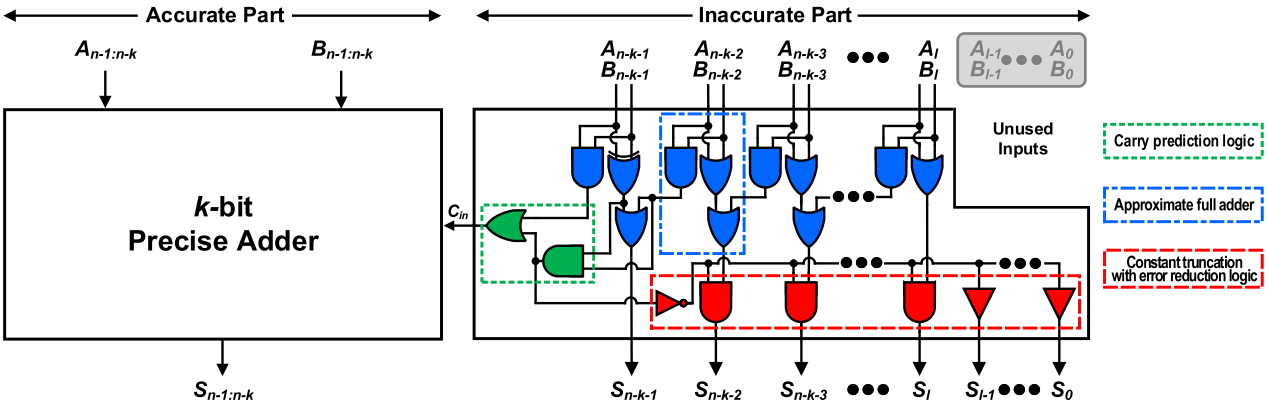


FIGURE 3. Overall hardware architecture of the proposed approximate adder, termed error reduced carry prediction approximate adder (ERCPAA).

produces a sum S_i and a carry output C_i using

$$S_i = A_i \oplus B_i \oplus C_{i-1} \tag{1}$$
$$C_i = A_iB_i + A_iC_{i-1} + B_iC_{i-1} \tag{2}$$

Although the FA requires two XOR gates to generate a sum, we replace the XOR gates with OR counterparts to do the same approximately to reduce hardware overhead in our approximate FA. In addition, the FA generates a carry output C_i from not only the two inputs, A_i and B_i, but also the carry from the previous bit position C_{i-1}. In other words, the carry of the previous bit position can be propagated to the next bit position through the current FA, resulting in a long critical path delay and degraded hardware performance in the carry propagate adders. To reduce the critical path delay and hardware overhead, we remove the dependency of the carry from the previous bit position to generate the carry output in our FA. Thus, the Boolean equations of our approximate FA are given by

$$S_{i,ERCPAA} = A_i + B_i + C_{i-1} \tag{3}$$
$$C_{i,ERCPAA} = A_iB_i \tag{4}$$

Consequently, the approximate part using the proposed FA cell does not form the carry propagation chain from the lower to the higher-order bit positions and thus the delay of the approximate part is consistent, although the size of the approximate part is larger (*i.e.*, k decreases under a given n). Note that the MSB position of the inaccurate part has a different configuration of the FA, which uses an XOR gate instead of the OR gate to generate the sum of the two input operands A_i and B_i. This improves the overall computation accuracy since the XOR-based FA gives a more accurate sum, and it also allows the carry prediction logic to produce a more accurate carry input to the precise adder than the OR-based FA. Table 1 depicts the truth table of the traditional and proposed FAs. The proposed FAs introduce errors if either of the operands is "1" and the carry of the previous bit position is "1." The OR-based FA causes an additional error at sum S_i when both operands are "1" and the carry of the previous bit position is "1".

TABLE 1. Truth table for traditional FA and proposed approximate FAs.

Inputs			Traditional FA Outputs		Proposed FA Outputs			
					OR-based		XOR-based	
A_i	B_i	C_{i-1}	S_i	C_i	S_i	C_i	S_i	C_i
0	0	0	0	0	0	0	0	0
0	0	1	1	0	1	0	1	0
0	1	0	1	0	1	0	1	0
0	1	1	0	1	1*	0*	1*	0*
1	0	0	1	0	1	0	1	0
1	0	1	0	1	1*	0*	1*	0*
1	1	0	0	1	1*	1	0	1
1	1	1	1	1	1	1	1	1

*Incorrect output.

C. PROPOSED CARRY PREDICTION TECHNIQUE

The accurate part can take a carry input generated from the inaccurate part to improve overall computation accuracy at the expense of a few logic gates [13], [15]–[18], [20]. The AND-based carry prediction scheme, which has an error of approximately 25%, is widely adopted since it is easily implemented by performing an AND operation with the inaccurate part's MSB inputs (*i.e.*, A_{n-k-1} AND B_{n-k-1}) to produce the carry input to the precise adder. In our proposed prediction, only two additional gates (*i.e.*, an AND gate and an OR gate in the green highlighted box in Figure 3) are utilized to produce the carry input with twice the prediction accuracy of the conventional AND-based one. Also, the inputs of the MSB and its previous bit position of the inaccurate part (*i.e.*, $A_{n-k-1:n-k-2}$ and $B_{n-k-1:n-k-2}$) are exploited to predict the carry input. Let P_i denotes the propagate signal of the $(i)^{th}$ bit position, and the carry from the previous bit position C_{i-1} is propagated to the carry output C_i if the propagate signal is "1," defined as

$$P_i = A_i \oplus B_i \tag{5}$$

$$C_i = C_{i-1} \quad if\ P_i = 1 \tag{6}$$

Since our carry prediction scheme leverages the inputs of two bit positions, a carry can be generated from either the $(n-k-1)^{th}$ or $(n-k-2)^{th}$ bit position. If a carry is produced in the $(n-k-1)^{th}$ bit position, the carry input C_{in} is simply C_{n-k-1}. On the other hand, if a carry is generated in the $(n-k-2)^{th}$ bit position, the carry C_{n-k-2} should be propagated through $(n-k-1)^{th}$ bit position to pass it to the accurate part. Therefore, the carry input C_{in} is derived by

$$C_{in} = C_{n-k-1} + P_{n-k-1}C_{n-k-2} \tag{7}$$

where C_i is defined in (4). According to Equation (7), one XOR, three AND, and one OR gates are required to generate the carry input C_{in}. C_{n-k-1} and C_{n-k-2} can be obtained from the proposed FAs in the corresponding bit positions and P_{n-k-1} can also be calculated using the XOR gate of the FA in the MSB position of the inaccurate part. It is worth noting that one of the reasons to replace the OR with an XOR in the FA at the MSB is to generate a P_{n-k-1} signal. Therefore, we only need two additional gates (see green box in Figure 3) to implement the proposed carry prediction logic.

Since our carry prediction is achieve using the inputs of the two MSB positions, it is correct when a carry is generated from any of these two bit positions. However, the carry prediction would be incorrect when a carry is produced from any lower-order bit position beyond the $(n-k-2)^{th}$ bit position and this carry is propagated through the $(n-k-2)^{th}$ and $(n-k-1)^{th}$ bit positions. Assuming that the two operands A and B are bitwise independent, then the propagated signal and carry are also bitwise independent. We denote an event that a carry is generated from $(n-k-3)^{th}$ or any of its lower-order bit positions by E_{ca}:

$$\begin{aligned} E_{ca} = {} & C_{n-k-3} \\ & + P_{n-k-3}C_{n-k-4} \\ & + P_{n-k-3}P_{n-k-4}C_{n-k-5} \\ & + \cdots + \prod_{i=1}^{n-k-3} P_iC_0 \end{aligned} \tag{8}$$

where C_i and P_i are defined in (4) and (5), respectively, and the probability of this event is given by

$$\begin{aligned} \mathbf{P}(E_{ca}) = {} & \mathbf{P}(C_{n-k-3}) \\ & + \mathbf{P}(P_{n-k-3}C_{n-k-4}) \\ & + \mathbf{P}(P_{n-k-3}P_{n-k-4}C_{n-k-5}) \\ & + \cdots + \mathbf{P}\Big(\prod_{i=1}^{n-k-3} P_iC_0\Big) \\ = {} & \mathbf{P}(C_{n-k-3}) \\ & + \mathbf{P}(P_{n-k-3})\mathbf{P}(C_{n-k-4}) \\ & + \mathbf{P}(P_{n-k-3})\mathbf{P}(P_{n-k-4})\mathbf{P}(C_{n-k-5}) \\ & + \cdots + \mathbf{P}(P_{n-k-3})\cdots\mathbf{P}(P_1)\mathbf{P}(C_0) \end{aligned}$$

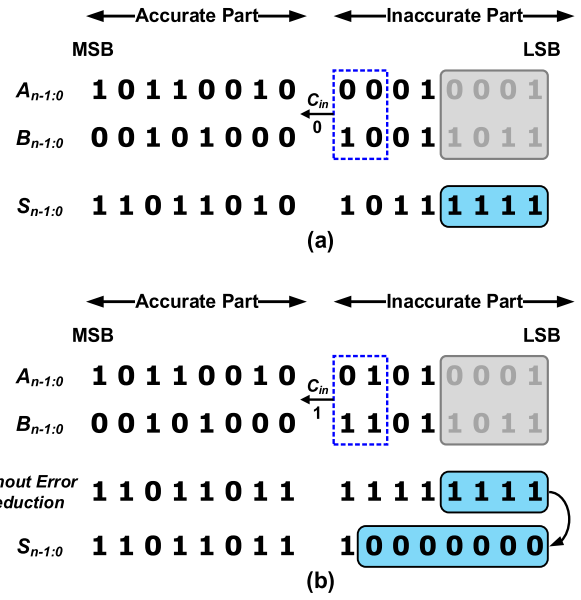


FIGURE 4. Operations of the proposed adder; (a) constant "1" truncation and (b) constant "0" truncation with error reduction.

$$\begin{aligned} & = \frac{1}{4} + \frac{1}{2}\cdot\frac{1}{4} + \frac{1}{2}\cdot\frac{1}{2}\cdot\frac{1}{4} + \cdots + \frac{1}{2^{n-k-3}}\cdot\frac{1}{4} \\ & = \frac{1}{2}\left(1 - \frac{1}{2^{n-k-2}}\right) \end{aligned} \tag{9}$$

where C_{n-k-3}, $P_{n-k-3}C_{n-k-4}$, $P_{n-k-3}P_{n-k-4}C_{n-k-5}$, $\cdots$, $\prod_{i=1}^{n-k-3} P_iC_0$ are mutually exclusive. Therefore, the error rate of the carry prediction of the proposed adder $\mathbf{ER_{CP}}$ is given by

$$\begin{aligned} \mathbf{ER_{CP}}(n,k) & = \mathbf{P}(P_{n-k-1}P_{n-k-2}E_{ca}) \\ & = \mathbf{P}(P_{n-k-1})\mathbf{P}(P_{n-k-2})\mathbf{P}(E_{ca}) \\ & = \frac{1}{2^3}\left(1 - \frac{1}{2^{n-k-2}}\right) \end{aligned} \tag{10}$$

Note that P_{n-k-1}, P_{n-k-2}, and E_{ca} are independent.

D. CONSTANT TRUNCATION WITH ERROR REDUCTION

The proposed adder outputs a constant to a few LSBs to reduce hardware overhead by sacrificing overall accuracy slightly since the lower-order outputs have relatively less impact on the accuracy than higher-order outputs [15]–[17], [33]–[35]. Figure 4 exhibits an example of constant truncation operations with error reduction using the adder design parameters $n = 16$, $k = 8$, and $l = 4$. As shown in Figure 4(a), our adder sets the l LSB outputs to "1" regardless of the inputs of the corresponding bit positions. When a carry is generated from $(n-k-2)^{th}$ bit position and then propagated through $(n-k-1)^{th}$ bit position, our adder performs error reduction. In short, the reduction is performed when $P_{n-k-1}C_{n-k-2} = 1$. Under this given input condition, the correct output of $(n-k-1)^{th}$ bit is "0," however, our FA produces "1" as the output at this bit position as shown

in Figure 4(b). This means that the approximate summation will be larger than the correct one in this case. Instead of forcing the l LSB outputs to "1," hence, the proposed adder sets all outputs of the inaccurate part except for its MSB position to "0," (*i.e.*, $S_{n-k-2:0} = 0$) making the approximation output closer to the correct addition. Under the given input shown in Figure 4(b), the ED, defined by $|S_{approximate} - S_{correct}|$ where $S_{approximate}$ and $S_{correct}$ are approximate and correct summations, respectively, decreases from 211 to 84. This reduction technique allows up to a $2^{n-k-1} - 1$ decrease in the ED.

E. ERROR RATE ANALYSIS

The proposed adder generates an output error when two input operands A_i and B_i of any bit position from $(n-k-2)^{th}$ to $(l)^{th}$ LSBs are both "1." In other words, if the inputs of at least one OR-based FA are both "1," an error occurs. According to Table 1, the OR-based FA produces an incorrect output at sum S_i when $A_i = 1$ and $B_i = 1$, whereas the XOR-based counterpart does not. This input condition generates a carry generation for the next bit position, which results in an output error at the sum of the next bit position. Furthermore, an error occurs when both the inputs of any bit position at the constant truncation part (*i.e.*, $(l-1)^{th}$ to $(0)^{th}$ bit position) are either "0" or "1" because the part fixes the output to "1." In other words, the constant truncation part output is always correct when either of the two inputs is "1." To simplify the error rate analysis, we first calculate a probability of the input condition to make the output of the adder correct, and then the error rate can be achieved by obtaining its complement. Since the proposed adder produces correct outputs when $A_i \neq 1$ and $B_i \neq 1$ where $l \leq i \leq n-k-2$ and $A_i \neq B_i$ where $0 \leq i \leq l-1$, we can define an event E_{co} that the adder generates always correct outputs as follows:

$$E_{co} = \prod_{i=l}^{n-k-2} \left(\overline{A_i B_i}\right) \cdot \prod_{i=0}^{l-1} \left(A_i \overline{B_i} + \overline{A_i} B_i\right) \tag{11}$$

and the probability of this event is given by

$$\begin{aligned}
\mathbf{P}(E_{co}) &= \mathbf{P}\Big(\prod_{i=l}^{n-k-2} \overline{A_i B_i}\Big)\mathbf{P}\Big(\prod_{i=0}^{l-1}(A_i\overline{B_i} + \overline{A_i}B_i)\Big) \\
&= \mathbf{P}(\overline{A_{n-k-2}B_{n-k-2}}) \cdots \mathbf{P}(\overline{A_l B_l}) \\
&\quad \times \mathbf{P}(A_{l-1}\overline{B_{l-1}} + \overline{A_{l-1}}B_{l-1}) \cdots \mathbf{P}(A_0\overline{B_0} + \overline{A_0}B_0) \\
&= \left(\frac{3}{4}\right)^{n-k-l-1}\left(\frac{1}{2}\right)^{l}
\end{aligned} \tag{12}$$

Note that we assumed that the two input operands A and B are bitwise independent. The error rate of the proposed adder is the probability of the complement of the event. Therefore, the error rate $\mathbf{ER}_{\mathbf{ERCPAA}}$ is given by

$$\begin{aligned}
\mathbf{ER}_{\mathbf{ERCPAA}}(n,k,l) &= 1 - \mathbf{P}(E_{co}) \\
&= 1 - \left(\frac{3}{4}\right)^{n-k-l-1}\left(\frac{1}{2}\right)^{l}
\end{aligned} \tag{13}$$

IV. EXPERIMENTAL RESULTS AND COMPARISON

In this section, we evaluate the performance of the proposed adder in terms of both hardware costs and computation accuracy through systematic analysis. Also, an extensive comparison with other existing approximate adders is presented to demonstrate the potential benefits of the proposed adder.

A. EXPERIMENT SETUP AND EVALUATION

We designed our adder in Verilog HDL and synthesized it with the Synopsys 32-*nm* generic library (SAED32) using Synopsys Design Compiler to examine the hardware characteristics of the proposed approximate adder in terms of area, delay, power, and energy [36]–[38]. We implemented a 16-bit adder using an 8-bit RCA-based precise adder (*i.e.*, $n = 16$ and $k = 8$). Prior studies suggested that a size of 7 to 9 bits for the inaccurate part would be appropriate to obtain a good tradeoff between output quality and power and energy saving for practical applications, such as video and image processing, and a 16-bit adder was widely adopted in these applications [7], [9], [23], [32], [39]. Therefore, we chose the adder design parameters of $n = 16$ and $k = 8$. In addition to the hardware cost, we also analyzed the error characteristics of the proposed adder by developing a software-based simulator. To exhaustively test a 16-bit adder, 2^{32} distinct input pairs can be considered but it is extremely intensive to compute. Therefore, we use 10 million (*i.e.*, 10^7) input pairs, each of which was uniformly distributed random input, to the proposed adder to obtain the error characteristics measured by various error metrics, such as the overall error rate, carry prediction error rate, mean error distance (MED), normalized mean error distance (NMED), and mean relative error distance (MRED).

B. TRADEOFF ANALYSIS OF THE PROPOSED ADDER

In our proposed design, the area, power, and energy performance degrade as the design parameter l decreases because a smaller l requires more logic gates to implement the adder. However, as l decreases, the overall computation accuracy performance improves. The power-NMED product was introduced to assess approximate adders considering the power and accuracy performance together [40]. Since this metric does not consider the area aspect, we can consider a new joint metric, the area-NMED product, to analyze the area and accuracy performance collectively. Similarly, the power-MRED and area-MRED products can be employed to jointly analyze the costs and accuracy.

To seek the best tradeoff between the hardware cost and accuracy of our adder, we adjusted design parameter l and obtained the power-NMED/MRED products and area-NMED/MRED products. It is noteworthy that the delay is consistent, although l varies, and thus we exclude the delay for the tradeoff analysis. Figure 5 shows the tradeoff of the hardware costs and accuracy for the proposed adder with various values of l. We varied the design parameter l from 1 to 6 because our adder requires at least two FAs at the

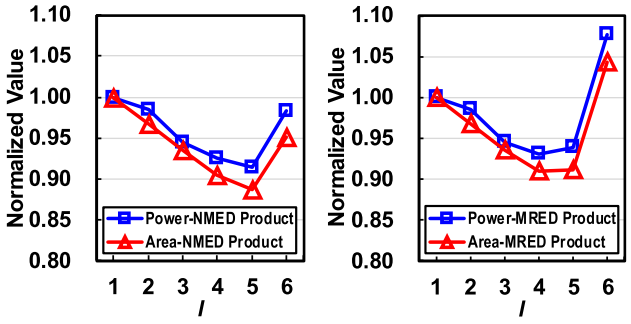


FIGURE 5. Tradeoff analysis between hardware costs and accuracy for the proposed 16-bit adder with various values of *l*, ranging from 1 to 6.

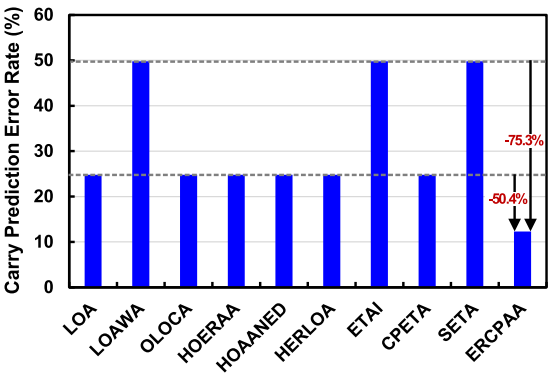


FIGURE 6. Comparison of carry prediction error rates of approximate adders.

TABLE 2. Accuracy performance of the proposed adder under various values of the design parameters.

Parameters (n, k)	*Error Rate* (%)	*MED*	*MRED*
(32, 28)	71.8796	1.72	1.38*e*-8
(32, 24)	98.2402	2.71*e*+1	1.93*e*-7
(32, 20)	99.8915	4.32*e*+2	3.52*e*-6
(32, 16)	99.9929	6.91*e*+3	4.48*e*-5
(32, 12)	99.9996	1.11*e*+5	5.87*e*-4
(32, 8)	100	1.77*e*+6	7.19*e*-3
(32, 4)	100	2.83*e*+7	7.96*e*-2

$(n-k-1)^{th}$ and $(n-k-2)^{th}$ bit positions to produce the carry input (*i.e.*, $l = 6$) and at least one constant truncation bit (*i.e.*, $l = 1$). As expected, the power and area become better and the NMED does worse as l increases. Specifically, the power dissipations at $l = 1$ and $l = 6$ are $40.6\mu W$ and $35.3\mu W$, respectively, and the area occupations are $150.4\mu m^2$ and $126.7\mu m^2$, respectively. On the contrary, the NMED degrades from 0.864×10^{-3} at $l = 1$ to 0.974×10^{-3} at $l = 6$. To effectively see the tradeoffs, the product values are normalized using the corresponding value of the adder with $l = 1$. Note that the lower the product value is, the better the tradeoff between the hardware costs and accuracy. According to the power-NMED and area-NMED products in Figure 5, the proposed adder has the best tradeoff performance at $l = 5$, which means a 5-bit constant truncation. In fact, from the power-MRED and area-MRED products' perspective, the best tradeoff of the adder is found at $l = 4$. While the power-/area-MRED products at $l = 4$ and $l = 5$ are almost the same, the value difference between the NMED counterparts at $l = 4$ and $l = 5$ are relatively larger than that of the MRED counterparts. Therefore, we use our 16-bit adder design with a parameter of $l = 5$ for comparison with other adders (*i.e.*, $n = 16$, $k = 8$, and $l = 5$).

C. ACCURACY OF THE PROPOSED ADDER WITH DIFFERENT PARAMETERS

To examine the accuracy performance of the proposed adder under different adder sizes and design parameters, we adjusted parameters k and l of the proposed 32-bit adder (*i.e.*, $n = 32$). Table 2 lists the error rate, MED, and MRED of the proposed adder at various values of the parameters. Here, we made the approximate part of the adder to have three non-constant bits according to the previous tradeoff analysis and thus parameter l was set to $n - k - 3$. As the parameter k increases, in other words, the size of the accurate part increases, at a given n, the accuracy performance gets better in terms of error rate, MED, and MRED as expected. The error rate drastically gets worse as k decreases and quickly reaches almost 100%. The MED and MRED values increase more than 15× and 11×, respectively, when the parameter k decreases by 4 at the given $k = 32$.

D. PERFORMANCE COMPARISON WITH EXISTING ADDERS

We also designed nine existing approximate adders that have similar architectures (LOA, LOAWA, OLOCA, HOERAA, HOAANED, HERLOA, ETAI, CPETA, and SETA) and an accurate adder (RCA) to compare them with our adder. To be fair, the adders were synthesized with the same 32-*nm* library using Synopsys Design Compiler, and 16-bit adders with an 8-bit RCA-based precise adder were implemented. For the OLOCA and SETA, design parameters of l and i were set to 6 and 7, respectively [15], [21]. Also, the error metrics were extracted under 10 million uniformly generated random input pairs. Furthermore, three more approximate adders that employ different architectures (RAP-CLA, BCSA, and BCSA_{ERU}) were designed using the identical design methodology and included in the comparison for completeness. The 16-bit adders with an RCA-based sub-adder and its block size of 4 were used for these adders [22], [23].

First, to demonstrate the superiority of our carry prediction technique, we compare the carry prediction error rate of the approximate adders, as shown in Figure 6. Note that the RAP-CLA, BCSA, and BCSA_{ERU} were excluded for this comparison because they do not have the carry to the precise adder due to a different architecture. The absence of the carry prediction in the LOAWA, ETAI, and SETA results in a nearly 50% carry prediction error. Note that the carry input to the precise adder is set to "0" in these adders. The LOA, OLOCA, HOERAA, HOAANED, HERLOA, and

TABLE 3. Performance summary of various adders.

Design	*Area* (μm^2)	*Delay* (ps)	*Power* (μW)	*PDP* (fJ)	*Error Rate* (%)	*MED*	*NMED* (1e-3)
RCA	190.4	1790	58.5	104.7	—	—	—
LOA	115.8	878	33.4	29.3	90.0	111.1	1.70
LOAWA	107.6	854	29.8	25.4	90.0	191.7	2.93
OLOCA	102.1	878	30.9	27.1	99.1	115.2	1.76
HOERAA	108.0	886	32.0	28.3	98.8	95.3	1.45
HOAANED	108.3	883	31.9	28.2	98.8	95.3	1.45
HERLOA	131.7	878	35.9	31.5	84.4	85.1	1.30
ETAI	126.9	854	32.5	27.8	90.0	179.2	2.73
CPETA	136.1	880	36.5	32.1	86.7	88.8	1.36
SETA	114.2	854	30.6	26.1	90.0	183.8	2.81
RAP-CLA	333.8	458	71.7	32.9	7.8	681.6	10.41
BCSA	242.6	665	64.8	43.1	21.1	479.7	7.35
$BCSA_{ERU}$	282.5	669	72.7	48.7	5.9	241.0	3.69
ERCPAA	128.2	942	35.6	33.6	98.2	58.9	0.90

CPETA include the AND-based carry speculation to the precise adder, which reduces the error rate to approximately 25%. The proposed prediction scheme is the most accurate among all adders and has an error rate of 12.305%, which is identical to the one calculated using Equation (10). Furthermore, our adder achieves error rate reductions of 75.3% and 50.4% on average compared with the adders without carry prediction and with the AND-based carry prediction, respectively.

Table 3 summarizes the hardware costs and accuracy performance of various adders. The RCA is the slowest adder because of the long carry propagation chain from the LSB to MSB. The longest delay results in the largest energy (*i.e.*, power-delay product; PDP) consumption although the $BCSA_{ERU}$ dissipates the largest power. The RAP-CLA is the fastest thanks to the relatively shorter carry chain generated by the blocks but occupies the largest area because a significant number of blocks is required to predict the carry for each bit position. Although the RAP-CLA consumes the second largest power, its energy consumption is relatively small and similar to that of the LOA, ETAI, their variants, and the proposed adder ERCPAA. The BCSA and $BCSA_{ERU}$ have almost the same delay but $BCSA_{ERU}$ has slightly larger area, power, and energy consumption than the BCSA while it shows slightly better accuracy performance. The adders that fix some LSB outputs to "1" and have AND-based carry prediction (*i.e.*, OLOCA, HOERAA, and HOAANED) show similar hardware characteristics. Furthermore, the HOERAA and HOAANED are almost the same in both hardware and accuracy performance. The lack of a carry prediction to the accurate part allows the corresponding adders (*i.e.*, LOAWA, ETAI, and SETA) to be the fastest among the FA-based adders, however, it results in poor MED and NMED performance compared to other FA-based approximate adders. In terms of area and power, the proposed adder ERCPAA is comparable to the HERLOA. It has the longest delay

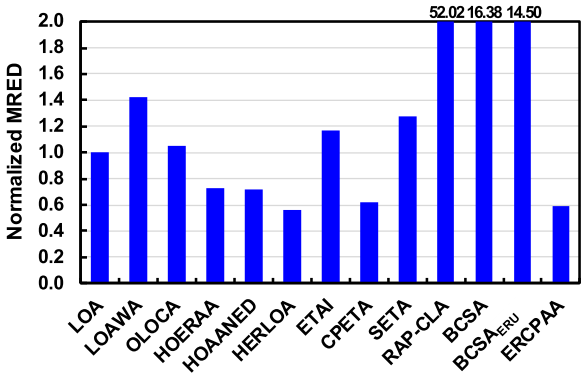


FIGURE 7. Comparison of mean relative error distances (MREDs) of approximate adders.

among the approximate adders due to the proposed carry prediction scheme and causes a relatively larger energy consumption, and it still has 3.12× higher energy efficiency than the RCA. The LOA, LOAWA, ETAI, and SETA have the same error rate of 90.0%, but the LOA has at least 61% better MED and NMED performance than others. The LOA variants that force a few LSB outputs to "1" (*i.e.*, OLOCA, HOERAA, and HOAANED) degrade the error rate compared to the LOA, which is up to 99% while maintaining a similar MED and NMED performance. The RAP-CLA, BCSA, and $BCSA_{ERU}$ show very good error rate performance less than 22% but relatively poor MED and NMED performance than the others. These adders can cause computation errors on the higher-order bit positions, whereas errors of other FA-based approximate adders concentrate on the lower-order bit positions (*i.e.*, approximate part). Although the $BCSA_{ERU}$ has the lowest error rate among the approximate adders, the proposed adder has the best MED and NMED performance. Specifically, the proposed adder shows 4.09× and 4.1× greater MED and NMED than the $BCSA_{ERU}$, respectively.

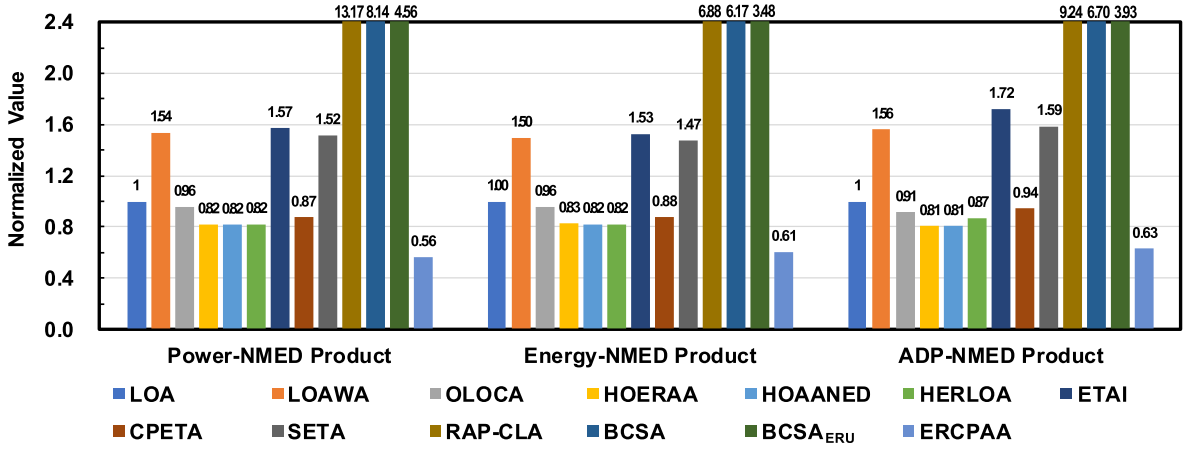


FIGURE 8. Normalized power-normalized mean error distance (NMED), energy-NMED, and area-delay product (ADP)-NMED products of approximate adders.

Figure 7 shows the MREDs of the approximate adders. To effectively make the comparison, the MRED values were normalized using the corresponding values of the LOA. The MREDs of the LOAWA, OLOCA, ETAI and SETA are slightly greater than that of the LOA. In particular, the MRED values of the RAP-CLA, BCSA, and BCSA$_{ERU}$ far exceed those of the others, and their values were inserted outside the bars. Specifically, the RAP-CLA exhibits the worst MRED performance, which is 52.02× greater than the LOA. The HOERAA and HOAANED have almost identical MRED values, which are 27.5% less than the LOA on average. In terms of MRED performance, the proposed adder is comparable to the HERLOA and CPETA. Specifically, the proposed adder reduces the MRED by 41.2% and 98.9% compared to the LOA and RAP-CLA, respectively.

E. JOINT ANALYSIS BETWEEN HARDWARE AND ACCURACY OF APPROXIMATE ADDERS

The error rate is an important metric to assess the accuracy of approximate adders. Unfortunately, its usefulness to evaluate the adder might be limited because it only considers the presence of an error but not the implication (*e.g.*, distance/magnitude) of the error on the additions [40]. Hence, we adopted ED based metrics, such as NMED and MRED, to better represent the accuracy of the adders rather than the error rate in the joint analysis. The power-NMED product is widely used to evaluate approximate adders in terms of power and accuracy jointly [40]. Similarly, the energy-NMED product was considered to analyze the energy aspect [18]. Unfortunately, neither of these two products do not includes the area or delay of approximate adders. The area-delay product (ADP) is a widely employed metric to evaluate hardware resources in terms of area and delay [15]. Therefore, we can consider a new joint metric, the ADP-NMED product, to analyze the tradeoff between area, delay, and accuracy.

Figure 8 exhibits the power-NMED, energy-NMED, and ADP-NMED products for 13 approximate adders. To compare these products effectively among the adders, they were normalized by the corresponding LOA values, and the values were inserted outside the bars. Undoubtedly, the proposed adder outperforms the other approximate adders in all these joint metrics. The RAP-CLA, BCSA, and BCSA$_{ERU}$ show very poor tradeoff performance and the three product values far exceed the other adders because they are a bit faster but consume a larger area, power, and energy than the other adders as shown in Table 3. Also, they exhibit relatively worse accuracy that deteriorates the tradeoff performance. Among the FA-based approximate adders, the ETAI has the worst tradeoff performance and the approximate adders that exclude the carry prediction (*i.e.*, LOAWA and SETA) have similar values with the ETAI. Although the lack of carry prediction allows these adders to be relatively efficient in terms of area, delay, and power, the poor accuracy degrades the overall tradeoff performance so that their three product values are at least 50% higher than those of the LOA. The OLOCA and CPETA have similar power-NMED and energy-NMED products, which are slightly better than the LOA. In addition, the HOERAA and HOAANED are comparable in all three products because of almost identical hardware architecture. The HERLOA is nearly the same power-NMED and energy-NMED products as the HOERAA and HOAANED. However, the larger area occupation stems from the hybrid error reduction scheme results in a higher ADP-NMED product. In summary, our adder has the best tradeoff performance among the compared approximate adders. Specifically, the power-NMED, energy-NMED, and ADP-NMED products of the proposed adder are 95.7%, 91.1%, and 93.2% lower than those of the RAP-CLA, respectively.

Similar to the joint metrics using NMED, we can take into account the metrics using MRED as well. Figure 9 shows the power-MRED, energy-MRED, and ADP-MRED products for the approximate adders. The values that were added outside the bars were normalized by the corresponding LOA values, and the three products using MRED exhibit

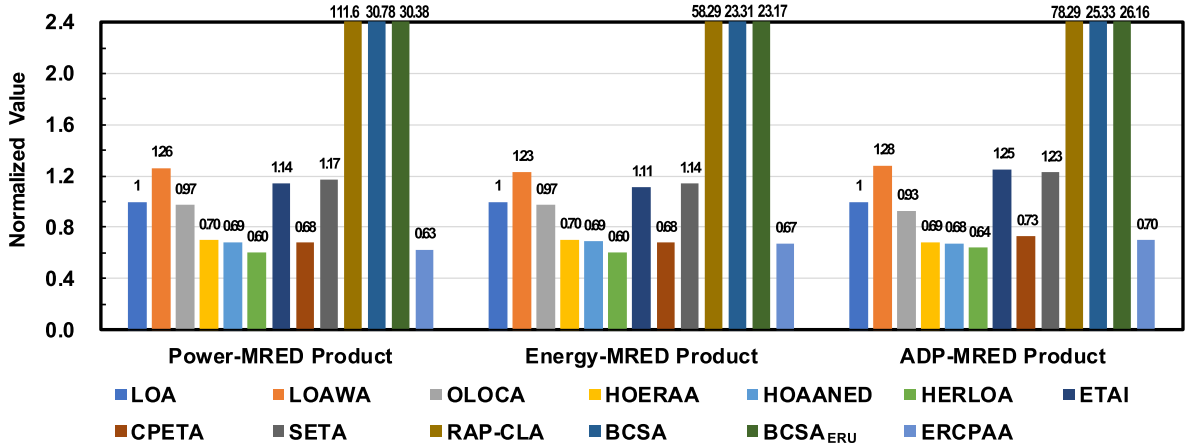


FIGURE 9. Normalized power-normalized mean relative error distance (MRED), energy-MRED, and area-delay product (ADP)-MRED products of approximate adders.

similar trends to those using NMED. The RAP-CLA, BCSA, and BCSA_{ERU} have significantly larger product values than the others. Specifically, the power-MRED, energy-MRED, and ADP-MRED products of these adders are at least 30×, 23×, and 25× greater than those of the LOA, respectively. Among the FA-based adders, the LOAWA shows the worst performance in the products using MRED, and the values of the ETAI and SETA are close to those of the LOAWA. The LOA and OLOCA have a similar tradeoff performance that the gap between the product values is less than 7%. The proposed adder demonstrates excellent tradeoff performance and is comparable to the HOERAA, HOAANED, HERLOA, and CPETA in all three products. Particularly, our adder achieves reductions in the power-MRED, energy-MRED, and ADP-MRED products, respectively, of 99.4%, 98.9%, and 99.1% of the RAP-CLA.

Finally, to evaluate the approximate adders in terms of the various hardware costs together with the accuracy performance (*i.e.*, NMED), we define the following product as a figure of merit (FoM) for the approximate adders.

$$\textbf{FoM} = \text{Energy} \times \text{Delay} \times \text{Area} \times \text{NMED} \tag{14}$$

In (14), the better energy efficiency, higher speed, and smaller area with good accuracy performance in the error distance for the approximate adders result in a smaller value for this FoM. Figure 10 exhibits the FoM of the approximate adders, and the values were normalized by the corresponding values of the LOA. Note that the lower the FoM value is, the better the approximate adder performance. The LOA, HERLOA, and CPETA have similar FoM values, and so do the LOAWA, ETAI, and SETA. Unfortunately, the FoMs of the RAP-CLA, BCSA, and BCSA_{ERU} are much greater than those of the other FA-based approximate adders and the numbers outside the bars indicate their FoM values. Specifically, the normalized FoM of these adders reaches greater than 10 because the poor NMED performance severely deteriorates the FoM, even though they are relatively faster than the others. The proposed adder ERCPAA

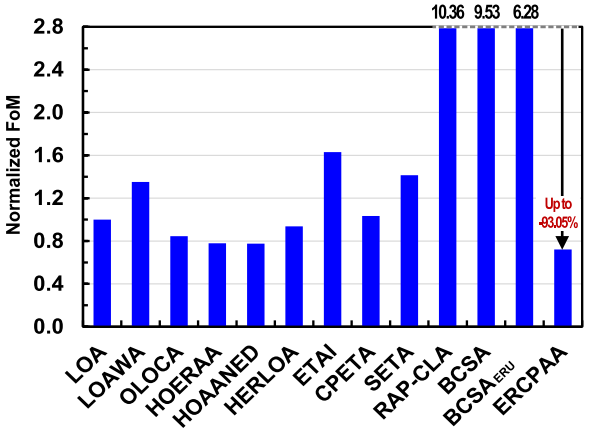


FIGURE 10. Normalized figure of merit (FoM) of approximate adders; energy-delay-area-normalized mean error distance (NMED) product.

is similar but shows better FoM performance than the OLOCA, HOERAA, and HOAANED. Obviously, the excellent design tradeoff between hardware costs and accuracy (*i.e.*, NMED) allows our design to be the most competitive adder among the approximate adders considered here. Particularly, the FoM of our adder is 93.05% smaller than that of the RAP-CLA.

V. APPLICATIONS OF APPROXIMATE ADDERS

Approximate adders can be utilized in many error-tolerant applications. To examine the effectiveness of the proposed adder in practical applications, we adopted our adder and existing approximate adders in a couple of applications and compared their performance.

A. DIGITAL IMAGE PROCESSING

First, the approximate adders were applied to digital image processing. Particularly, we considered Gaussian smoothing filtering, which is achieved by a 2-D convolution of an image and Gaussian kernel and used the peak signal-to-noise ratio (PSNR) to measure the output image quality. The

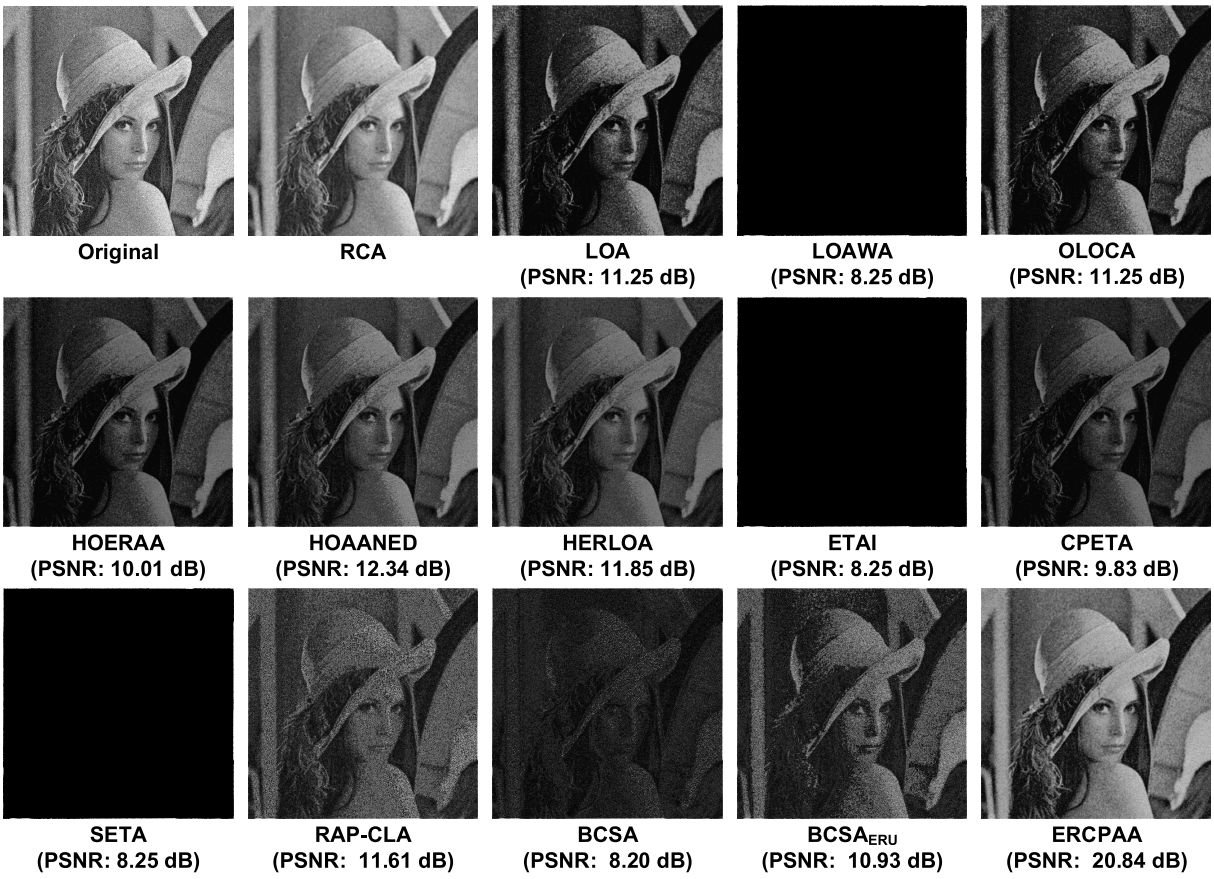


FIGURE 11. Original input image and output images with PSNRs of Gaussian smoothing filtering using an accurate adder, the existing approximate adders, and the proposed adder.

TABLE 4. PSNRs of various images by Gaussian smoothing filtering using the approximate adders.

Design	*cameraman*	*peppers*	*baboon*	*F-16*	*couple*	*fishing boat*	*clock*	*airplane*
LOA	11.46	12.16	11.30	12.60	10.48	10.83	12.09	12.41
LOAWA	7.88	9.00	7.47	5.41	8.33	7.70	5.17	4.88
OLOCA	11.48	12.17	11.30	12.60	10.48	10.83	12.09	12.41
HOERAA	10.08	10.86	9.71	8.73	9.65	9.54	8.57	8.35
HOAANED	13.14	13.11	11.86	10.99	12.08	11.86	10.63	10.57
HERLOA	12.40	12.60	11.21	9.93	11.70	11.44	9.52	9.46
ETAI	7.90	9.00	7.47	5.41	8.33	7.70	5.17	4.88
CPETA	9.97	10.64	9.37	8.46	9.48	9.30	8.07	7.98
SETA	7.88	9.00	7.47	5.41	8.33	7.70	5.17	4.88
RAP-CLA	10.97	12.39	10.22	8.20	11.87	11.31	7.28	7.43
BCSA	7.57	8.99	7.13	5.05	8.20	7.65	4.43	4.38
BCSA$_{ERU}$	11.88	11.93	9.93	9.29	10.36	7.65	8.39	8.72
ERCPAA	23.10	21.27	20.19	19.47	20.93	21.57	18.34	19.13

following 5×5 Gaussian kernel G is used for filtering [39].

$$G = \frac{1}{2^8}\begin{bmatrix} 1 & 3 & 6 & 3 & 1 \\ 3 & 15 & 25 & 15 & 3 \\ 6 & 25 & 41 & 25 & 6 \\ 3 & 15 & 25 & 15 & 3 \\ 1 & 3 & 6 & 3 & 1 \end{bmatrix} \tag{15}$$

For the Gaussian smoothing operation, the addition was performed using an accurate adder as well as the proposed and existing approximate adders, whereas multiplication and division were performed accurately. Additionally, since Gaussian smoothing filtering is useful to reduce image noise, we added zero-mean, Gaussian white noise with a variance of 0.01 to the original *lena* image, which is a grayscale image

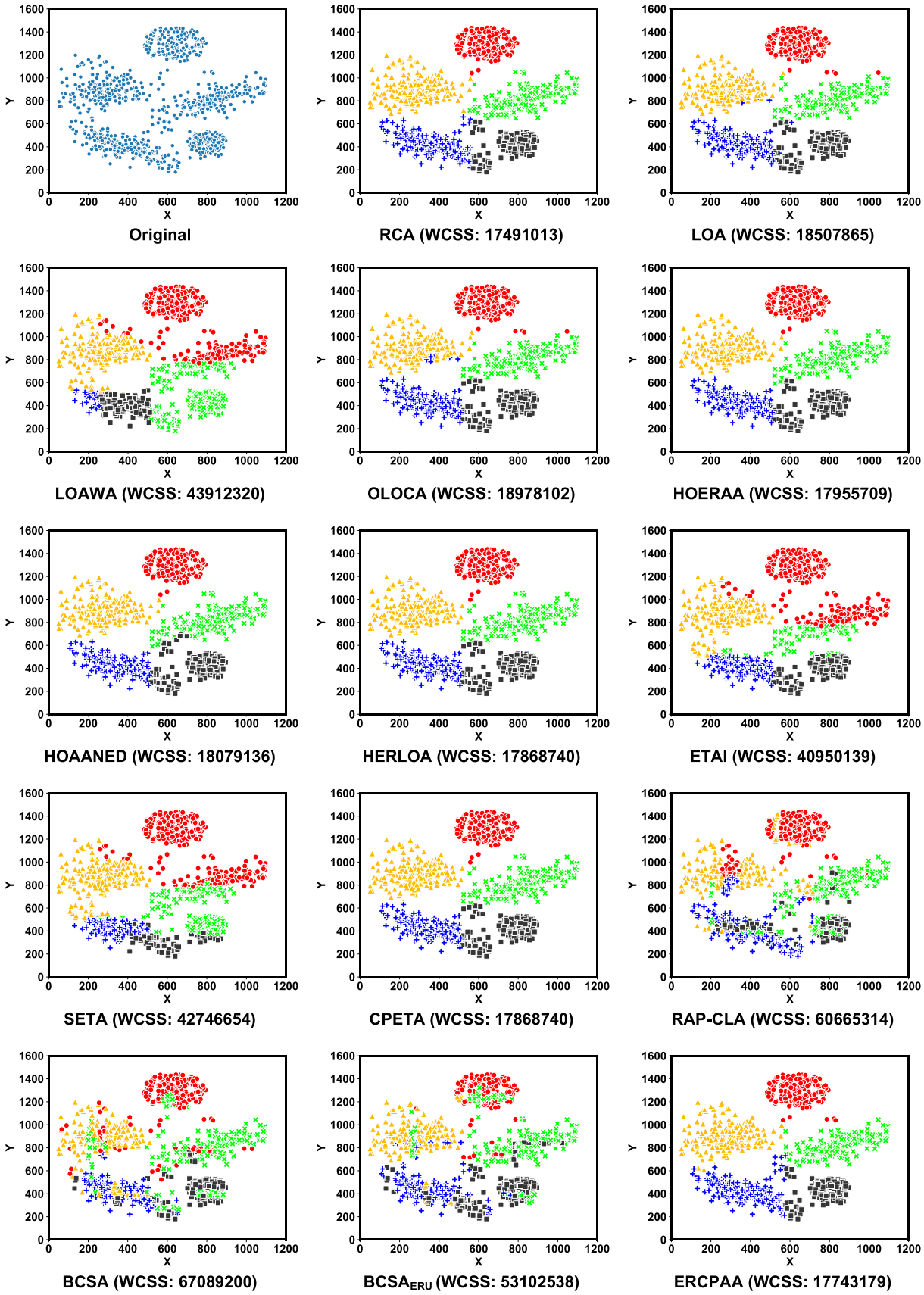


FIGURE 12. Original data and clustered data with WCSS by *k*-means clustering using various adders.

with a size of 512 × 512, and then performed filtering [41]. We employed an accurate adder (RCA), the proposed adder, and 12 existing approximate adders in the filtering. The PSNR values were calculated against the images obtained by applying Gaussian filtering to the original input image using the accurate adder. First, the approximate adders with design parameters of $n = 16$ and $k = 8$ were applied to the filtering and we found out that all approximate adders, except for the RAP-CLA, BCSA, and BCSA_{ERU} whose block sizes were set to 4, produce visually very similar output images, although our adder generates the best image quality with the highest PSNR. Therefore, to make the output images more visually distinguishable, we reduced the size of the accurate part to 3 and the block size of the approximate adders by half.

Figure 11 shows the original noisy image and output images of Gaussian smoothing filtering using various adders. The BCSA shows the worst PSNR value of 8.20dB among the images. The PSNR value of 8.25dB is identical to the output images processed by the LOAWA, ETAI, and SETA. Similarly, the LOA and OLOCA generate the same output image quality. The PSNRs of images with the HOERAA, CPETA, and BCSA_{ERU} range from 9.83dB to 10.93dB. In other words, the image quality processed by these adders is between those processed by the LOA/OLOCA and LOAWA/ETAI/SETA. The HOAANED, HERLOA, and RAP-CLA yield slightly better output images than the LOA/OLOCA. The proposed adder produces the best image quality distinctly seen in human vision with a PSNR value of 20.84dB, which means that the filtered image is the closest to the one generated by the accurate adder. This confirms that the approximation errors of the proposed adder have a negligible impact on the processing quality and thus, it is suitable for digital image processing applications. To further examine the approximate adders in the application, we performed the Gaussian smooth filtering for eight more well-known benchmark images (*cameraman, peppers, baboon, F-16, couple, fishing boat, clock,* and *airplane*) obtained from [42]. Note that the same white noise was added to these images. The PSNRs of the filtered output images generated by the approximate adders are listed in Table 4. All images exhibit a similar PSNR trend with the *lena* image. Evidently, our ERCPAA achieves the best PSNR value for all benchmark images among the approximate adders in the Gaussian smoothing filtering application.

B. MACHINE LEARNING

In addition to the filtering application, we also took machine learning into consideration to explore the efficacy of the proposed adder. Specifically, we examined the performance of the approximate adders in k-means clustering, which is an unsupervised machine learning algorithm and extensively utilized in data mining [43]. Basically, the algorithm groups a set of unlabeled data points into k different clusters that each data point belongs to only one cluster. When clustering, it minimizes the sum of distances between the data points to the centroids of the corresponding clusters, which is defined by the within cluster sum of squares (WCSSs). Therefore, it iteratively calculates the distances where the subtraction operation is mainly used in this algorithm. We applied the approximate adders to the operation [28]. Note that the subtraction can be done by 2's complement addition. We obtained an unlabeled dataset comprising 1000 data points from [44] and set the number of clusters k to 5.

Figure 12 demonstrates the visualized 2-D original dataset and clustered dataset using the accurate adder, the existing approximate adders, and the proposed adder. The WCSS values were extracted to evaluate the quality of the clustering results using the difference adders [28]. The value closer to the one clustered by the accurate adder indicates a better clustering result. The LOAWA, ETAI, and SETA show a similar clustering result, and so do the LOA and OLOCA. The LOA/OLOCA produce much better clustering quality than the LOAWA/ETAI/SETA because the latter does not include any carry prediction logic to the precise adder and this degrades computation accuracy. The proposed approximate adder exhibits the best clustering result closest to the one using the accurate adder. The HOERAA, HOAANED, HERLOA, and CPETA yield slightly worse results than the proposed adder. Unfortunately, the RAP-CLA, BCSA, and BCSA_{ERU} show poor clustering performance and do not allow the dataset to be partitioned properly. Specifically, the WCSS values of these adders are up to 384% and 378% larger than those of the accurate and proposed adders, respectively. In summary, the proposed adder has the best performance in terms of WCSS in k-means clustering as well.

VI. CONCLUSION

In this paper, we presented a new approximate adder that combines error-reduced carry prediction and constant truncation with error reduction schemes. The proposed carry prediction scheme achieves an error rate reduction of up to 75% compared to the existing approximate adder, and the proposed error reduction technique improves the overall computation accuracy by decreasing the error distance. We systematically analyzed our design and sought the best tradeoff between hardware costs and accuracy by adjusting the adder design parameter. When implemented in the 32-*nm* CMOS technology, the proposed design has 1.90× and 3.12× greater power- and energy efficiency, respectively, than the RCA, with NMED and MRED improvements of up to 91.4% and 98.9%, respectively, compared to the existing approximate adders. Importantly, our design achieves 95.7%, 91.1%, and 93.2% reductions in the power-NMED, energy-NMED, and ADP-NMED products, respectively, compared to the RAP-CLA due to an excellent design tradeoff. Our adder also reduces the power-, energy-, and ADP-MRED products by up to 99.4% compared to the others. Particularly, in terms of the FoM considering hardware resources (*i.e.*, energy, delay, and area) and the accuracy performance

(*i.e.*, NMED), the proposed adder is up to 93.05% better than the RAP-CLA. The proposed adder has been adopted in a digital image processing application and proves that the proposed adder rarely affects the output image quality that is the closest to the one with the accurate adder. Additionally, we have demonstrated the performance of our adder in a machine learning application and the result has shown that the proposed adder outperforms the other approximate adders. Therefore, the proposed adder is well applicable to energy-efficient and error-tolerant applications, such as machine learning, neuromorphic computing, and digital signal processing.

ACKNOWLEDGMENT

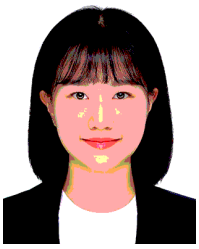
(Jungwon Lee and Hyoju Seo contributed equally to this work.)

REFERENCES

[1] L. Jiao and J. Zhao, "A survey on the new generation of deep learning in image processing," *IEEE Access*, vol. 7, pp. 172231–172263, 2019.
[2] C. Lammie, A. Olsen, T. Carrick, and M. R. Azghadi, "Low-power and high-speed deep FPGA inference engines for weed classification at the edge," *IEEE Access*, vol. 7, pp. 51171–51184, 2019.
[3] Y. Kim, Y. Zhang, and P. Li, "A reconfigurable digital neuromorphic processor with memristive synaptic crossbar for cognitive computing," *ACM J. Emerg. Technol. Comput. Syst.*, vol. 11, no. 4, pp. 38:1–38:25, Apr. 2015.
[4] B. Liu, Z. Wang, W. Zhu, Y. Sun, Z. Shen, L. Huang, Y. Li, Y. Gong, and W. Ge, "An ultra-low power always-on keyword spotting accelerator using quantized convolutional neural network and voltage-domain analog switching network-based approximate computing," *IEEE Access*, vol. 7, pp. 186456–186469, 2019.
[5] I. Khan, S. Choi, and Y.-W. Kwon, "Earthquake detection in a static and dynamic environment using supervised machine learning and a novel feature extraction method," *Sensors*, vol. 20, no. 3, p. 800, Feb. 2020.
[6] Q. Wang, P. Li, and Y. Kim, "A parallel digital VLSI architecture for integrated support vector machine training and classification," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 8, pp. 1471–1484, Aug. 2015.
[7] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 32, no. 1, pp. 124–137, Jan. 2013.
[8] Y. S. Yang and Y. Kim, "Approximate digital leaky Integrate-and-fire neurons for energy efficient spiking neural networks," *IEIE Trans. Smart Process. Comput.*, vol. 9, no. 3, pp. 252–259, Jun. 2020.
[9] A. Raha, H. Jayakumar, and V. Raghunathan, "Input-based dynamic reconfiguration of approximate arithmetic units for video encoding," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 24, no. 3, pp. 846–857, Mar. 2016.
[10] T. Moreau, A. Sampson, and L. Ceze, "Approximate computing: Making mobile systems more efficient," *IEEE Pervasive Comput.*, vol. 14, no. 2, pp. 9–13, Apr. 2015.
[11] S. Mittal, "A survey of techniques for approximate computing," *ACM Comput. Surv.*, vol. 48, no. 4, pp. 1–33, May 2016.
[12] Q. Xu, M. Todd, and S. K. Nam, "Approximate computing: A survey," *IEEE Design Test*, vol. 33, no. 1, pp. 8–22, Feb. 2016.
[13] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 57, no. 4, pp. 850–862, Apr. 2010.
[14] P. Albicocco, G. C. Cardarilli, A. Nannarelli, M. Petricca, and M. Re, "Imprecise arithmetic for low power image processing," in *Proc. Conf. Rec. 46th Asilomar Conf. Signals, Syst. Comput. (ASILOMAR)*, Nov. 2012, pp. 983–987.
[15] A. Dalloo, A. Najafi, and A. Garcia-Ortiz, "Systematic design of an approximate adder: The optimized lower part constant-OR adder," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 8, pp. 1595–1599, Aug. 2018.
[16] P. Balasubramanian and D. L. Maskell, "Hardware optimized and error reduced approximate adder," *Electronics*, vol. 8, no. 11, p. 1212, Oct. 2019.
[17] P. Balasubramanian, R. Nayar, D. L. Maskell, and N. E. Mastorakis, "An approximate adder with a near-normal error distribution: Design, error analysis and practical application," *IEEE Access*, vol. 9, pp. 4518–4530, 2021.
[18] H. Seo, Y. S. Yang, and Y. Kim, "Design and analysis of an approximate adder with hybrid error reduction," *Electronics*, vol. 9, no. 3, p. 471, Mar. 2020.
[19] N. Zhu, W. L. Goh, W. Zhang, K. S. Yeo, and Z. H. Kong, "Design of low-power high-speed truncation-error-tolerant adder and its application in digital signal processing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 18, no. 8, pp. 1225–1229, Aug. 2010.
[20] Y. Kim, "An accuracy enhanced error tolerant adder with carry prediction for approximate computing," *IEIE Trans. Smart Process. Comput.*, vol. 8, no. 4, pp. 324–330, Aug. 2019.
[21] J. Lee, H. Seo, Y. Kim, and Y. Kim, "Approximate adder design with simplified lower-part approximation," *IEICE Electron. Exp.*, vol. 17, no. 15, pp. 1–3, Aug. 2020.
[22] O. Akbari, M. Kamal, A. Afzali-Kusha, and M. Pedram, "RAP-CLA: A reconfigurable approximate carry look-ahead adder," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 65, no. 8, pp. 1089–1093, Aug. 2018.
[23] F. Ebrahimi-Azandaryani, O. Akbari, M. Kamal, A. Afzali-Kusha, and M. Pedram, "Block-based carry speculative approximate adder for energy-efficient applications," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 67, no. 1, pp. 137–141, Jan. 2020.
[24] Y. Kim, Y. Zhang, and P. Li, "An energy efficient approximate adder with carry skip for error resilient neuromorphic VLSI systems," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, Nov. 2013, pp. 130–137.
[25] Y. Kim, Y. Zhang, and P. Li, "Energy efficient approximate arithmetic for error resilient neuromorphic computing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 23, no. 11, pp. 2733–2737, Nov. 2015.
[26] A. B. Kahng and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," in *Proc. 49th Annu. Design Autom. Conf. (DAC)*, 2012, pp. 820–825.
[27] M. Shafique, W. Ahmad, R. Hafiz, and J. Henkel, "A low latency generic accuracy configurable adder," in *Proc. 52nd Annu. Design Autom. Conf.*, Jun. 2015, pp. 81:1–81:6.
[28] J. Hu, Z. Li, M. Yang, Z. Huang, and W. Qian, "A high-accuracy approximate adder with correct sign calculation," *Integration*, vol. 65, pp. 370–388, Mar. 2019.
[29] V. Camus, M. Cacciotti, J. Schlachter, and C. Enz, "Design of approximate circuits by fabrication of false timing paths: The carry cut-back adder," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 8, no. 4, pp. 746–757, Dec. 2018.
[30] M. Pashaeifar, M. Kamal, A. Afzali-Kusha, and M. Pedram, "Approximate reverse carry propagate adder for energy-efficient DSP applications," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 26, no. 11, pp. 2530–2541, Nov. 2018.
[31] N.-C. Huang, S.-Y. Chen, and K.-C. Wu, "Sensor-based approximate adder design for accelerating error-tolerant and deep-learning applications," in *Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE)*, Mar. 2019, pp. 692–697.
[32] L. Soares, M. da Rosa, C. Machado, E. da Costa, and S. Bampi, "Design methodology to explore hybrid approximate adders for energy-efficient image and video processing accelerators," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 6, pp. 2137–2150, Jun. 2019.
[33] H. Seo and Y. Kim, "A new approximate adder with duplicate-constant scheme for energy efficient applications," in *Proc. IEEE Int. Conf. Consum. Electron. Asia (ICCE-Asia)*, Nov. 2020, pp. 1–2.
[34] F. Frustaci, S. Perri, P. Corsonello, and M. Alioto, "Energy-quality scalable adders based on nonzeroing bit truncation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 27, no. 4, pp. 964–968, Apr. 2019.
[35] H. Seo, Y. S. Yang, and Y. Kim, "An energy-efficient imprecise adder with a lower-part constant approximation," in *Proc. Int. SoC Design Conf. (ISOCC)*, Oct. 2020, pp. 143–144.
[36] H. Bhatnagar, *Advanced ASIC Chip Synthesis: Using Synopsys Design Compiler Physical Compiler and Prime Time*. Norwell, MA, USA: Kluwer, 2002.

[37] R. Goldman, K. Bartleson, T. Wood, K. Kranen, V. Melikyan, and E. Babayan, "32/28 nm educational design kit: Capabilities, deployment and future," in *Proc. IEEE Asia Pacific Conf. Postgraduate Res. Microelectron. Electron. (PrimeAsia)*, Dec. 2013, pp. 284–288.
[38] (Jan. 2012). *Synopsys Digital Standard Cell Library SAED_EDK32/28_CORE Databook Revision 1.0.0*. Accessed: Jul. 27, 2021. [Online]. Available: https://www.synopsys.com/community/university-program/teaching-resources.html
[39] B. Garg and S. K. Patel, "Reconfigurable carry look-ahead adder trading accuracy for energy efficiency," *J. Signal Process. Syst.*, vol. 93, no. 1, pp. 99–111, Jan. 2021.
[40] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," *IEEE Trans. Comput.*, vol. 62, no. 9, pp. 1760–1771, Sep. 2013.
[41] M. Masadeh, O. Hasan, and S. Tahar, "Input-conscious approximate multiply-accumulate (MAC) unit for energy-efficiency," *IEEE Access*, vol. 7, pp. 147129–147142, 2019.
[42] *The USC-SIPI Image Database*. Accessed: Jul. 27, 2021. [Online]. Available: http://sipi.usc.edu/database/database.php
[43] K. P. Sinaga and M.-S. Yang, "Unsupervised K-means clustering algorithm," *IEEE Access*, vol. 8, pp. 80716–80727, 2020.
[44] *Clustering Benchmark*. Accessed: Jul. 27, 2021. [Online]. Available: http://github.com/deric/clustering-benchmark

JUNGWON LEE (Graduate Student Member, IEEE) is currently pursuing the integrated B.S. and M.S. degrees with the School of Computer Science and Engineering, Kyungpook National University, Daegu, Republic of Korea. Her research interests include deep learning and approximate arithmetic.



HYOJU SEO (Graduate Student Member, IEEE) received the B.S. degree from the School of Computer Science and Engineering, Kyungpook National University, Daegu, Republic of Korea, in 2020, where she is currently pursuing the M.S. degree. Her research interests include approximate computing, neuromorphic computing, deep learning accelerator, and image processing.

HYELIN SEOK (Student Member, IEEE) is currently pursuing the integrated B.S. and M.S. degrees with the School of Computer Science and Engineering, Kyungpook National University, Daegu, Republic of Korea. Her research interests include approximate arithmetic and new computing systems.

YONGTAE KIM (Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from Korea University, Seoul, Republic of Korea, in 2007 and 2009, respectively, and the Ph.D. degree from the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, USA, in 2013. From 2013 to 2018, he was a Software Engineer with Intel Corporation, Santa Clara, CA, USA. Since 2018, he has been with the School of Computer Science and Engineering, Kyungpook National University, Daegu, South Korea, where he is currently an Assistant Professor. His research interests include energy efficient integrated circuits and systems, particularly, neuromorphic computing and approximate computing, and new memory devices and architectures.