A High-Accuracy Hardware-Efficient Multiply–Accumulate (MAC) Unit Based on Dual-Mode Truncation Error Compensation for CNNs

SONG-NIEN TANG (Member, IEEE), AND YU-SHIN HAN
Department of Information and Computer Engineering, Chung Yuan Christian University, Taoyuan 320314, Taiwan

ABSTRACT This paper presents a multiply–accumulate (MAC) unit that enables a dual-mode truncation error compensation (TEC) scheme based on a fixed-width Booth multiplier (FWBM) for convolutional neural network (CNN) inference operations. The proposed tailored TEC schemes of Modes 1 and 2 can achieve high MAC accuracy for a general or rectified linear unit-based CNN model with general (Mode 1) or positive/zero (Mode 2) input patterns. By pre-calculating the pre-known CNN model coefficients, the proposed dual-mode TEC scheme can be realized using minimal partial product operations with high hardware efficiency using a software–hardware codesign approach. Further, a reconfigurable architecture of the resultant MAC unit is presented to realize the proposed dual-mode TEC scheme. By evaluating the accuracy for $9-N$ and $25-N$ MAC operations ($N$ denotes the number of times MAC is performed), a MAC operation using the proposed TEC scheme can achieve the highest accuracy for Modes 1 and 2, relative to contrast samples that directly employ the FWBM with a conventional TEC function. The hardware performances of $9-N$ and $25-N$ MAC units are also evaluated using the TSMC 40-nm standard cell library. Compared with the contrast TEC-enabled designs, the proposed MAC unit exhibits higher hardware efficiency in terms of area, delay, and power consumption and achieves a minimum reduction of more than 40% in both area-delay-error and power-delay-error products. Moreover, the resultant $9-N$ and $25-N$ MAC units are verified using a system-on-chip field-programmable gate array platform to test a CNN model for handwritten digit classification.

INDEX TERMS Multiply-accumulate, MAC, 2D convolution, convolutional neural network, CNN, accelerator, truncation error compensation, booth multiplier.

I. INTRODUCTION

A convolutional neural network (CNN) is a popular group of deep learning models that has shown considerable performance in many applications, such as image, signal processing, pattern recognition, and computer vision. With the development of edge artificial intelligence applications [1], many CNN inference functions are expected to be executed in client-side devices. Therefore, localized CNN processing schemes that can perform the real-time inference function in the edge end are in demand [1]–[3]. Generally, CNN operations can be conducted on a local platform using central processing units (CPUs) or graphical processing units (GPUs) [1], [2]. However, these approaches generally cause computational speed or power consumption issues. Thus, the CPU- or GPU-based platform may be unsuitable for the timing-critical or computation-intensive edge CNN operations [3]. Consequently, considerable research has been conducted on hardware (HW) acceleration schemes using application-specific integrated circuits or field-programmable gate array (FPGA) units, which can perform efficient client-side CNN inference computations [3]–[8]. Two-dimensional (2D) convolution and inner product computations are the main operations of the convolution and fully connected layers in a CNN [1] and are performed using a series of multiply–accumulate (MAC)
operations. Consequently, MAC units are essential components to construct the processing elements (PEs) for CNN acceleration [9]–[14]. By considering the fixed-point MAC operation, a suitable bit width can be determined based on the CNN classification accuracy [8], [15]. To avoid significant bit-width increases because of accumulation in a MAC unit, one approach is to enable the multiplication output with the same bit width as the input in an CNN accelerator [4], [13]. Booth multipliers [16] are frequently used as the MAC multiplication schemes for CNNs [9]–[11]. For MAC units using Booth multipliers, the aforementioned approach can be employed using a fixed-width Booth multiplier (FWBM), which allows an $L$-bit $\times L$-bit Booth multiplier to truncate the $L$ least significant bits (LSBs) and preserve the $L$ most significant bits (MSBs) at its output for a full numerical range. To reduce HW costs for the operations associated with the truncated $L$-bit LSBs of a full-width product, the concept of truncation error compensation (TEC) has been proposed for fixed-width multiplier designs [17]–[32]. TEC uses an estimated bias function to compensate for the truncation error associated with reduced HW costs.

In this study, we aim to design a FWBM-based MAC unit for CNN inference operations and propose the corresponding TEC schemes. The main features and contributions of the proposed design are listed:

- A TEC scheme of Mode 1 provides an ensemble high-accuracy TEC function to improve the overall MAC operation accuracy with general values input patterns. Because CNN model coefficients are generally preknown, the proposed Mode 1 TEC scheme can be realized with minimal HW overheads.

- A TEC scheme of Mode 2 provides a high-accuracy TEC function adaptable to positive or zero value input patterns for MAC operations in CNNs using the rectified linear unit (ReLU) activation function. Based on the preknown CNN coefficients together with the MSB processing of partial products, the proposed Mode 2 scheme can be realized using HW resources similar to those used for the Mode 1 scheme.

- The efficient HW architecture of the MAC unit realizes the aforementioned TEC scheme of Modes 1 and 2. The proposed design can perform dual-mode TEC with high HW efficiency using a reconfigurable configuration of the partial product array.

- The design extension based on the proposed Mode 2 TEC scheme supports the MAC unit design for general applications that are not targeted only for CNN operations in the edge end.

The remainder of the paper is organized as follows. Section II introduces the background and literature review. Section III outlines the proposed TEC schemes of Modes 1 and 2 and their contributions. Section IV presents the architecture of the proposed MAC unit supporting the dual-mode TEC scheme. Section V evaluates the accuracy and HW performance of the proposed design. Finally, Section VI highlights the conclusions of this study.

Figure 1. System configuration for CNN inference acceleration based on a software–hardware codesign scheme.

II. BACKGROUND

A. MAC ACCELERATION IN CNNs

A general CNN model mainly comprises three types of layers, namely, convolution, pooling, and fully connected layers. A convolution layer primarily performs 2D convolution for feature extraction. For a set of input feature map ($F_{\text{in}}$) and kernel map ($K$), an output feature map ($F_{\text{out}}$) can be obtained using the 2D convolution operation, as described in (1), where $F_{\text{in}}$, $K$, and $F_{\text{out}}$ are the forms of a 2D matrix. By splitting (1) into a one-dimensional form, (1) can be transformed into a MAC operation for the $F_{\text{in}}$ and $K$ operands. Fully connected layers comprise several neurons that are linked to the layers. At each neuron node, input and coefficient vectors are set to perform the inner product calculation, which also corresponds to a MAC operation. Therefore, MAC operations are dominant in the CNN model.

$$F_{\text{out}}(u, v) = K \otimes F_{\text{in}} = \sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} K(m, n) \times F_{\text{in}}(u - m, v - n) \tag{1}$$

A training process based on a cost function and backpropagation algorithm is necessarily performed to determine the parameters in the specified CNN model [1]. Generally, the CNN is trained on a high-end CPU/GPU-based platform in floating-point operations [10], [11]. After training, the kernel and weight coefficients are determined for CNN inference applications. When an inference operation is executed using HW acceleration at the edge end, a device that uses a software (SW)–HW codesign scheme [33]–[35] is often employed, which is developed by evaluating the work division. For example, computation-demanding 2D convolution is often performed using an HW accelerator, while other low-effort CNN operations and system controls are processed at the SW end [3]–[5], [33]. In addition to 2D convolution, several studies have further explored HW acceleration for inner product operations in the fully connected layer [8], [36]. A system configuration for CNN inference acceleration based on the SW–HW codesign scheme is shown in Fig. 1, which comprises a host CPU, external memory, and accelerator. At the HW end, numerous PEs are used to perform CNN acceleration in fixed-point operations. These PEs contain
TABLE 1. Mapping results for the Booth encoder and partial products.

| \(v_{i,j}^{(k)}\), \(w_{i,j}^{(k)}\), \(a_{i,j}^{(k)}\) | \(d_{j}^{(k)}\) | \(p_{i,j}^{(k)}\), \(p_{i-1,j}^{(k)}\), \(p_{i-2,j}^{(k)}\), \(p_{i-3,j}^{(k)}\), \(p_{i-4,j}^{(k)}\) | \(n_{i,j}^{(k)}\) |
|---|---|---|---|
| 0 0 0 | 2 | 0 (000) | 0 0 0 \(\ldots\) 0 0 0 0 |
| 0 1 1 | 1 (001) | 1 \(\ldots\) 1 \(\ldots\) 1 \(\ldots\) 0 |
| 1 0 0 | 2 (010) | 0 0 0 \(\ldots\) 0 0 0 0 |
| 1 0 0 | 2 (010) | 1 \(\ldots\) 1 \(\ldots\) 1 \(\ldots\) 0 |
| 0 1 1 | 2 (010) | 2 \(\ldots\) 2 \(\ldots\) 2 \(\ldots\) 2 |
| 1 0 0 | 2 (010) | 1 \(\ldots\) 1 \(\ldots\) 1 \(\ldots\) 1 |

MAC units are responsible for MAC operations, as expressed in (2), where \(W_k\) and \(A_k\) are the coefficients (for kernel maps or weights) and feature map pixels, respectively, in fixed-point values (e.g., \(L\) bits) and \(N\) is the number of times MAC is performed. In acceleration operation, the predetermined \(W_k\) values stored in the external memory can be initially sent to the coefficient buffer (Fig. 1). Subsequently, the \(A_k\) data are sent to the data buffer and operated with \(W_k\) using the MAC units. The associated data (including the MAC results) are transmitted through direct memory access (DMA).

\[
MAC = \sum_{k=1}^{N} W_k \times A_k
\]  

B. FIXED-WIDTH BOOTH MULTIPLIER (FWBM)

Booth multipliers are popular because they reduce the number of operated partial products (PPs), thus yielding high HW efficiency [37], [38]. The Booth encoding method can be described as follows. In (2), the 2’s complement of \(A_k\) and \(W_k\) in \(L\) bits can be represented by (3). The superscript \((k)\) in (3) annotates the \(k\)th multiplication stage mapping to (2). In this study, the symbol \((k)\) has the same meaning for other formulas and is sometimes omitted in related descriptions for convenience. By applying the Booth encoding to \(W_k\) and \(A_k\) can be further expressed in \(d_j\) as (4), where \(Q = (1/2) \times L\) and the encoded \(d_j\) values associated with \((w_{j+1}, w_j, w_{j-1})\) are listed in Table 1. Each \(d_j\) term can also be represented using three bits of \((b_2, b_1, b_0)\) corresponding to \((negative, two, one)\) information. Thus, a 2\(L\)-bit full-width product (FWP) for \(W_k \times A_k\) can be obtained using (5). Based on (5), \(Q\) rows of PPs can be obtained, which are associated with each \(d_j\). Using binary arithmetic for the multiplication of \(d_j\) in (5), the PPs \((P_{i,j}\) and \(n_{i,j}\); \(i\) is from \(L\) to \(0\)) for each \(d_j\) can be derived in terms of \(a_i\), \(0\), or \(1\) as listed in Table 1, and \(n_j\) is an LSB plus term for each row.

\[
\begin{align*}
A_k &= -a_{2j-1}^{(k)} \cdot 2^{L-1} + \sum_{i=0}^{L-2} d_{j}^{(k)} \cdot 2^i \\
W_k &= -w_{2j-1}^{(k)} \cdot 2^{L-1} + \sum_{i=0}^{L-2} w_{j}^{(k)} \cdot 2^i \\
W_k &= \sum_{j=0}^{Q-1} d_{j}^{(k)} \cdot 2^{2j}, \quad d_{j}^{(k)} = -2w_{2j+1}^{(k)} + w_{2j}^{(k)} + w_{2j-1}^{(k)}
\end{align*}
\]  

\[
FWP^{(k)} = \left(\sum_{j=0}^{Q-1} d_{j}^{(k)} \cdot 2^{2j}\right) \times \left(-a_{2j-1}^{(k)} \cdot 2^{L-1} + \sum_{i=0}^{L-2} d_{j}^{(k)} \cdot 2^i\right)
\]  

Based on (5) and the terms listed in Table 1, Fig. 2 shows the structure of the partial product (PP) array for the \(L\)-bit \(W_k \times A_k\) FWBM. As shown in Fig. 2, the PP terms can be divided into two groups: main part (MP) and truncation part (TP). An \(L\)-bit FWBM preserves \(L\)-bit MSBs and truncates \(L\)-bit LSBs of FWP. Corresponding to the PP array structure (Fig. 2), MP is operated to obtain the required \(L\)-bit outcome, while TP is related to the computation for the truncated \(L\) LSBs of FWP in FWBM.

C. FWBM TEC SCHEMES (LITERATURE REVIEW)

A simple method for realizing FWBM is to calculate all PPs in MP and TP and then approximate the full-width result to \(L\) MSBs by rounding. Such an approach is called the post-truncation (PT) scheme, which can achieve high accuracy but leads to significant HW complexity. Another simple scheme is direct truncation (DT), which directly truncates the PP terms in TP; thus, only MP is calculated. The DT scheme reduces HW costs; however, its operation accuracy is very low. To address the PT and DT issues, numerous TEC-based methods have been proposed for FWBMs [21]–[32] or even fixed-width Baugh–Wooley array multipliers [17]–[20]. For a general FWBM design realizing the TEC scheme, Fig. 2 shows the general TEC operations. As shown in this figure, TP can be further divided into major (\(TP_{major}\)) and minor (\(TP_{minor}\)) sets. \(TP_{major}\) has a dominant influence on accuracy than \(TP_{minor}\) with respect to the carry term from TP to MP. A variable \(w\) (e.g., \(w = 1, 2, 3\) shown in Fig. 2) indicates the column range of \(TP_{major}\), which adjusts the effect on accuracy contributed by \(TP_{major}\). For the derivation, the \(w\)-th column \(TP_{major}\) can be mapped to the \(2^{-w}\) digit; thus, the FWP result in (5) can be rewritten as (6). In TEC operations (ref. Fig. 2), \(TP_{minor}\) in (6) is truncated and a bias term is estimated to compensate for the truncation error associated with \(TP_{minor}\). Accordingly, an FWBM result (i.e., an \(L\)-bit quantized FWP, \(FP_{q}\) from FWP) can be obtained, as expressed
in (7), where \( B^{(k)} \) indicates the estimated bias value and \( R(\cdot) \) denotes the rounding operation. When \( N \) FWBMs with TEC are directly used to perform MAC computation in (2), a quantized MAC result in \( (E_{\text{acc}} + L) \) bits is obtained, as shown in (8). \( E_{\text{acc}} \) is the number of extended bits owing to accumulation, which equals the roundup value of \( \log_2(N) \).

\[
FWP^{(k)} = MP^{(k)} + \left( TP^{(k)}_{\text{major}} + TP^{(k)}_{\text{minor}} \right) \cdot 2^L \tag{6}
\]

\[
FWP_q^{(k)} = MP^{(k)} + \sigma_k \cdot 2^L, \quad \sigma_k = R \left\{ TP^{(k)}_{\text{major}} + B^{(k)} \right\} \tag{7}
\]

\[
MAC_q = \sum_{k=1}^{N} \left( MP^{(k)} + \sigma_k \cdot 2^L \right) \tag{8}
\]

Numerous FWBM designs with TEC schemes have been reported [21]–[32]. Generally, these studies obtained the bias value (i.e., \( B^{(k)} \)) in (7) using computer simulation [21]–[25] and probability estimation [25]–[32] methods. For the simulation-based methods [21]–[25], in [21], the bias terms were derived using simulation findings, followed by the Karnaugh map simplification. Linear regression analysis and simulation were employed in [22] to generate the bias terms. Moreover, several studies ([23], [24]) have derived formulas for the TEC bias based on simulation results. In [24] and [25], the Booth-encoded results were utilized to determine the bias terms using a simulation to further improve accuracy. The simulation-based methods presented in [21]–[25] are practical but usually consume exhaustive simulation time to determine the TEC bias. Instead of exhaustive simulation, the work in [26] used the expected \( PP \) value to derive the bias terms. The authors of [25] also presented a probabilistic analysis, together with their simulation-based works. Moreover, the probability estimation methods [27]–[32] derived the closed forms of the bias function based on the expected value or conditional probability for \( PP \) terms. In [27], the expected values for two groups of \( TP_{\text{minor}} \) (with or without \( n_0, \ldots, n_{Q-1} \) terms) are individually estimated and combined to obtain the probabilistic estimation bias when \( w = 1 \). Moreover, a generalized probabilistic estimation bias (GPEB) method [28] further enhanced the work in [27] for the cases of \( w = 2 \) and 3. Based on the GPEB methods ([27], [28]), a simple 1- or 2-bit constant bias function was derived. The author of [29] presented a bias estimation form using conditional probability based on nonzero Booth encoder outputs for each row of \( TP_{\text{minor}} \). A more complex method based on [29] was presented in [30] using a conditional probability model for the \( TP_{\text{minor}} \) rows that slightly improved the accuracy but increased HW overhead. In [31], the authors combined two schemes based on conditional probability [29] and expected values [28] to develop an accuracy-area improved bias function using probability estimation and computer simulation (PACS). A Booth-encoded sign-digit-based conditional probability (BSCP) method was proposed in [32], which further introduced the sign of nonzero Booth encoder output to generate a relatively high-accuracy bias function; however, only the case of \( w = 1 \) was considered.

### III. PROPOSED DUAL-MODE TEC SCHEME

Several studies have particularly considered the design of the MAC units for CNN acceleration [9]–[14]; however, TEC functions were not considered. A convenient method is the direct use of conventional TEC-enabled FWBMs [21]–[32] to devise MAC units for CNNs. However, such an approach may not achieve optimized accuracy or HW efficiency owing to a lack of considerations for the features of CNN inference operations. Therefore, this study proposes a tailored dual-mode TEC scheme to improve the accuracy with minimal HW costs for the aimed MAC unit design.

#### A. MODE 1 TEC SCHEME (GENERAL PATTERN)

Mode 1 operation concerns MAC computation for general values in a Gaussian or uniform distribution, which is common in many CNN models.

1) DERIVATION FOR MODE 1 TEC SCHEME

In Fig. 2, a closed form for \( TP_{\text{major}} \) computation associated with \( w \) is derived in (9), wherein a floor operator. Apart from \( TP_{\text{major}} \), there are rows of \( TP_{\text{minor}} \) in the \( TP \) PP array. For convenience, we denote the top and bottom rows as the \( 0^{\text{th}} \) and \( y^{\text{th}} \) row ( ), respectively, for \( TP_{\text{minor}} \). The value of the \( j^{\text{th}} \) row of \( TP_{\text{minor}} \) in the \( k^{\text{th}} \) MAC multiplication stage can be calculated using (10). Based on the probabilistic analysis, the expected value of \( P_{i,j} \) (i is obtained from \( 0 \) to \( L-2-j-1-w \)) or \( n_j \) in (10) is \( 1/2 \) ([27]–[32]). However, for a specified \( d_j \), several values of \( P_{0,j} \) and \( n_j \) can be exactly determined to be 0 or 1, as observed in Table 1. Therefore, we use the hybrid of probabilistic or deterministic values of \( P_{i,j} \) and \( n_j \) to calculate the expected value of \( TP_{\text{minor},j} \) using (10). By adopting a scheme using hybrid values, the \( PP \)s \( (P_{0,j}, n_j) \) and \( E[TP_{\text{minor},j}] \) (the expected value of \( TP_{\text{minor},j} \)) according to \( d_j \) values are listed in Table 2.

\[
TP_{\text{major}} = \sum_{w=1}^{w} 2^{-w} \left( \sum_{j=0}^{\varphi-1} P_{L-2-j-w,j} + n_{w}^{(k)} \right), \quad \varphi = Q - 1 - \left( \lfloor w/2 \rfloor - 1/2 \right) \tag{9}
\]

\[
TP_{\text{minor},j} = 2^{-L+2j} \cdot n_{0,j}^{(k)} + 2^{-L+2j+1} \cdot n_{1,j}^{(k)} + \cdots + 2^{-1-w} \cdot n_{0,j}^{(k)} \tag{10}
\]

where \( 0 \leq j \leq Q - 1 - \left\lfloor w/2 \right\rfloor \).
TABLE 2. Partial products and based on hybrid values (blank: probabilistic values; gray: deterministic values).

| d_j | Hybrid Values for P_{j}^{(k)} and n_{j}^{(k)} | E(T_{\text{minor},j}) |
|-----|------------------------------------------|-------------------|
| 0   | All = 0                                  | 0                 |
| 1   | \( \frac{1}{2}; \frac{1}{2} \); \( \frac{1}{2} \); \( \frac{1}{2} \); \( n_{j}^{(k)} = 0 \) | (1/2) \( 2^{-2} - (1/2) \cdot 2^{-l+2} \) |
| -1  | \( \frac{1}{2}; \frac{1}{2} \); \( \frac{1}{2} \); \( n_{j}^{(k)} = 1 \) | (1/2) \( 2^{-2} + (1/2) \cdot 2^{-l+2} \) |
| 2   | \( \frac{1}{2}; \frac{1}{2} \); \( n_{j}^{(k)} = 0 \) | (1/2) \( 2^{-2} - 2^{-l+2} \) |
| -2  | \( \frac{1}{2}; \frac{1}{2} \); \( n_{j}^{(k)} = 1 \) | (1/2) \( 2^{-2} + 2^{-l+2} \) |

In (13), the original \( E_{\text{acc}} \) result in (8) can be revised as (14), where a global bias (i.e., \( B_{M1} \)) is introduced in the TEC operation using (12). In practice, the \( B_{M1} \) value in (14) requires only fractional precision for the \( w \) digit using (15), in which a \( F_{w} \) floor function is employed for truncating \( B_{M1} \) to \( X \) integer part and \( w \)-bit fractional part.

\[
E\left[T_{\text{minor},j}^{(k)}\right] = \begin{cases} 
0, & d_j = 0 \\
2^{-w-1} \pm (1/2) \cdot 2^{-L+j}, & d_j = \pm 1 \\
2^{-w-1} \mp 2^{-L+j}, & d_j = \pm 2 
\end{cases} \tag{11}
\]

\[
E\left[T_{\text{minor},j}^{(k)}\right] = \sum_{j=0}^{N} E\left[T_{\text{minor},j}^{(k)}\right] \tag{12}
\]

\[
MAC = \sum_{k=1}^{N} MP^{(k)} + \left( \sum_{k=1}^{N} TP_{\text{major}}^{(k)} + \sum_{k=1}^{N} TP_{\text{minor}}^{(k)} \right) \cdot 2^{L} \tag{13}
\]

\[
MAC_{q} = \sum_{k=1}^{N} MP^{(k)} + \sum_{k=1}^{N} E\left[T_{\text{minor},k}^{(k)}\right] + \left( \sum_{k=1}^{N} TP_{\text{major}}^{(k)} \right) \cdot 2^{L} \tag{14}
\]

\[
F_{w}(B_{M1}) = X + x_{1} \cdot 2^{-1} + x_{2} \cdot 2^{-2} + \cdots + x_{w} \cdot 2^{-w} \tag{15}
\]

2) OPERATIONS WITH MODE 1 TEC

Compared with (7) and (8), which are based on conventional TEC-enabled FWBM, \( B_{M1} \) in (14) is a global bias accumulating the TEC effect in (12) for each FWBM, thus improving the MAC accuracy by considering overall operations [39]. Moreover, \( TP_{\text{major}} \) of all \( N \) FWBMs is accumulated in (14); therefore, the rounding noise only existed in the final stage. Although \( B_{M1} \) in (14) can provide effective global biasing, the calculation for \( B_{M1} \) practically consumes HW costs. In Mode 1 TEC design, \( B_{M1} \) can be generated using minimal HW resources by exploiting the feature that \( W_{k} \) coefficients are generally known before CNN inference execution, thus allowing Booth-encoded result \( d_{j} \) to be preobtained. Accordingly, \( B_{M1} \) in (14) can be calculated in advance based on Table 2. Consequently, a \( PP \) array only necessarily operates \( MP \) and \( TP_{\text{major}} \) terms, thereby significantly reducing HW overheads for \( B_{M1} \) generation. In an SW–HW codesign approach (Fig. 1), precalculated \( B_{M1} \) can be stored and sent to MAC units for final accumulation. Fig. 3 illustrates the schematic of an \( N \) 8-bit MAC operation using the FWBM-based TEC scheme of Mode 1 with \( w = 1 \). As shown in Fig. 3, \( PPs \) for \( N \) sets of \( MP \) and \( TP_{\text{major}} \) are calculated (represented by the left \( N \) rectangles) and accumulated with a precalculated \( B_{M1} \) term. Furthermore, a \( PS_{M1} \) value can be produced by summing three-row MSB “1” of all \( N \) FWBMs in signed numbers. Similar to the \( B_{M1} \) process, this \( PS_{M1} \) term can also be summed at the final stage so that the required HW computation for the MSB-one addition in a \( PP \) array can be further reduced. Using the analogy in Fig. 3, Mode 1 TEC scheme applied to the same MAC operation with \( w = 2 \) and 3 are depicted in Figs. 4(a) and 4(b), respectively.

B. MODE 2 TEC SCHEME (POSITIVE/ZERO PATTERN)

The ReLU activation function rectifies its negative input to zero. Thus, MAC operations (shown in (2)) for an ReLU-based CNN model only have input patterns (i.e., \( A_{k} \)) of either positive values or considerable-amount zeros [4], [40]. Such a condition of zero-valued input patterns particularly occurs in image processing with sufficient black-color pixels
or CNN applications with no pooling functions [41]. The operation of Mode 2 considers such MAC execution in positive/zero values.

1) DERIVATION FOR MODE 2 TEC SCHEME

In Mode 2 condition, FWBM with TEC practically disables its bias function and outputs a zero product when its input pattern is zero to prevent unexpected accuracy degradation. This requirement makes the Mode 1 TEC scheme unsuitable for handling zero patterns in Mode 2 because $B_{M1}$ is a presumed global bias that assumes that all $N$ $E[TP_{minor}]$ values are employed in (14), which cannot be dynamically changed using $A_r$ by the MAC unit HW. Accordingly, Mode 2 TEC scheme is proposed to provide high-accuracy biasing for positive patterns, while disable biasing for zero patterns in each FWBM. Such a function can be directly offered using (12) in cooperation with an option of setting $E[TP_{minor}] = 0$; however, complex HW is required owing to the wide-range digits and arithmetic of signed numbers for calculating $E[TP_{minor, j}]$, as shown in Table 2. For this issue, the $E[TP_{minor, j}]$ value is further simplified for nonzero $d_j$ using an ensemble average that excludes the case of $d_j = 0$. The $E[TP_{minor, j}]$ value in Table 2 can be approximated, as shown in (16), based on the probability distribution for nonzero $d_j$ in Table 1. Consequently, the original $E[TP_{minor}]$ value in (12) can be approximated using (17), where $\delta_j$ indicates whether $d_j$ is a nonzero term (i.e., $\delta_j = 1$ for nonzero $d_j$), which can be mapped to the exclusive-OR operation of $(b_1, b_0)$ bits in Table 1. In (17), the TEC bias is only computed at the $2^{-w-1}$ digit for the $\delta_{0,y}$ term.

$$E\left[ TP_{minor, j}^{(k)} \right] \approx (2^{-w-1} - 2^{-L+2j-1}) \times (2^{/6}) + (2^{-w-1} + 2^{-L+2j-1}) \times (2^{/6}) + (2^{-w-1} - 2^{-L+j}) \times (1^{/6}) + (2^{-w-1} - 2^{-L+2j}) \times (1/6) = 2^{-w-1}$$

$$E\left[ TP_{minor}^{(k)} \right] \approx \sum_{j=0}^{y} (2^{-w-1} \cdot \delta_j^{(k)}) = (b_1 \oplus b_0)\delta_j^{(k)}$$

(16)

A more efficient method of bias generation is using the deterministic information instead of $\delta_j$ for the bottom row of $j = y$. As shown in Fig. 2 with $w = 1$, $P_{0,0-1}$ and $\eta_{d-1}$ can be employed to replace $\delta_{Q-1}$, where $\gamma = Q - 1$ at the $2^{-2}$ digit. Moreover, for $w = 2$, $\delta_{Q-2}$ ($\gamma = Q - 2$) at the $2^{-3}$ digit can be substituted by $P_{1, Q-2}$ together with a $C_{out}$ term that is carried from the $2^{-4}$ digit addition of $P_{0, Q-2}$ and $\eta_{Q-2}$. Moreover, we can use $P_{0, Q-2}$ and $\eta_{Q-2}$ to substitute $\delta_{Q-2}$ at the $2^{-3}$ digit for the case of $w = 3$. Consequently, the bias function in (17) can be modified to (18), (19), and (20) for FWBM with $w = 1, 2,$ and $3$, respectively, where an additional subscript “$M2$” emphasizes the updated $E[TP_{minor}]$ value for Mode 2. Based on the derivations in (18)–(20), a general bias function for FWBM can be obtained as (21) for odd or even $w$. Thus, the MAC calculation in (14) for Mode 1 is updated to (22) for Mode 2, where an accumulated bias (i.e., $B_{M2}$) in (21) is applied to the TEC function.

$$E\left[ TP_{minor, M2}^{(k)} \right]_{w=1} = \sum_{j=0}^{Q-2} 2^{-j} \cdot \delta_j^{(k)} + 2^{-2} \cdot (P_{0,0-1} + \eta_{Q-1})$$

(18)

$$E\left[ TP_{minor, M2}^{(k)} \right]_{w=2} = \sum_{j=0}^{Q-3} 2^{-j} \cdot \delta_j^{(k)} + 2^{-3} \cdot (P_{1, Q-2} + C_{out} + \eta_{Q-2})$$

(19)

$$E\left[ TP_{minor, M2}^{(k)} \right]_{w=3} = \sum_{j=0}^{Q-3} 2^{-j} \cdot \delta_j^{(k)} + 2^{-4} \cdot (P_{0, Q-2} + \eta_{Q-2})$$

(20)

$$E\left[ TP_{minor, M2}^{(k)} \right]_{even} = \sum_{j=0}^{y-1} 2^{-w-1} \cdot \delta_j^{(k)} + 2^{-w-1} \cdot (P_{0, 0-1} + \eta_{y})$$

$$E\left[ TP_{minor, M2}^{(k)} \right]_{odd} = \sum_{j=0}^{y-1} 2^{-w-1} \cdot \delta_j^{(k)} + 2^{-w-1} \cdot (P_{1, 0-1} + \eta_{y})$$

(21)

$$MAC_q = \sum_{k=1}^{N} MP^{(k)}$$

$$+ R \left\{ \sum_{k=1}^{N} TP_{major}^{(k)} + \sum_{k=1}^{N} E[TP_{minor, M2}]_{w=2}^{(k)} \right\} \cdot 2^L$$

(22)

2) OPERATIONS WITH MODE 2 TEC

The MSB process for Mode 2 can be further improved using positive/zero-only input patterns. Because the sign bit of positive or zero patterns is always zero, all MSB $PPs$ ($P_{L,0}$) in $L$-bit FWBM can be exactly determined using the sign of $d_j$ (i.e., “1” for positive $d_j$; “0” for negative $d_j$) and are denoted as $S_j$ (Table 1). Similar to the principle of $B_{M1}$ generation (Mode 1), these $S_j$ terms can be obtained in
In Fig. 5(a), the original adding operations for the signed presummation for MSB “1” (Fig. 3) can be upgraded using the predefined 8-bit MAC layer. Based on (18)–(20), (22), and $PS_M^2$ generation method, Figs. 5(a), 5(b), and 5(c) depict the schematic of the N 8-bit MAC operation for positive patterns (i.e., $A_k > 0$ or $A_k! = 0$ in Mode 2) using the FWBM-based TEC scheme of Mode 2 with $w = 1, 2, 3$, respectively. As shown in Fig. 5(a), the original adding operations for $S_{0\ldots3}$ terms are moved to presummation and an extra addition for $d_{0\ldots2}$, $P_{0,3}$, and $n_3$ is executed at the $2^{-2}$ digit. Comparing Fig. 5(a) with Fig. 3 for Modes 1 and 2, the required operation amounts are nearly the same and the numerical range shows a 1-bit shift, as expressed by the ($<1$) and ($>1$) symbols. Similar outcomes were also observed for the two cases of $w = 2$ and 3, as schematically shown in Figs. 4(a) and 5(b) and Figs. 4(b) and 5(c), respectively.

In addition to positive patterns, MAC operations in Mode 2 are also characterized using zero input data (i.e., $A_k = 0$). In this situation, FWBM is expected to generate a zero product for the zero-valued input. For TEC-enabled FWBM, a practical approach is to set its $MP$, $TP_{major}$, and bias terms to zero or multiplexing its product with zero. However, such an approach cannot be directly applied to the proposed design because the presumed MSB-one and $S_j$ of FWBM already affects the MAC result in our design, and all other portions (including residual $MP$, $TP_{major}$, and bias) are set to zero. This issue can be resolved by optionally adding of “1” to each row of $PPs$ according to the sign of $d_j$ in FWBM. As an exemplification, we use an 8-bit $W_k \times 0$ operation with $w = 1$, in which the Booth decoder result of $W_k$ is $(d_3, d_2, d_1, d_0)$ and the signs of $d_{0\ldots3}$ are $(-, -, +, +)$. Fig. 6(a) shows the generated $PP$ array based on Fig. 5(a) and Table 1 for the aforementioned example. As shown in Fig. 6(a), only an expected zero product can be obtained when “1” carried from the negative-$d_j$ row of $TP_{minor}$ is added to $MP$, $TP_{major}$, and presummation portions if $TP_{minor}$ is truncated. Using binary arithmetic, the $PP$ array in Fig. 6(a) can be modified to Fig. 6(b) by adding “1” at the digit next to the presumed MSBs and setting other $PP$s to zeros. For all possible $d_{0\ldots3}$ conditions, a general-modified $PP$ array using the same addition-of-1 scheme can be deduced using the analogy shown in Fig. 6(c), where variable $z_j$ is defined by $z_j = 1$ for $d_j < 0$ and $z_j = 0$ for $d_j \geq 0$.

**IV. PROPOSED MAC UNIT WITH DUAL-MODE TEC**

As mentioned in the previous section (III/B/2), the required operations for Modes 1 and 2 ($A_k! = 0$) are nearly the same. Moreover, the operations for Mode 2 ($A_k = 0$) can be performed in the original $PP$ array for Mode 2 ($A_k! = 0$) using the inserted $z_j$ and sufficient “0” (Fig. 6(c)). Therefore, this study aims to develop a MAC unit that employs a reconfigurable structure to enable multiple operations for the proposed dual-mode TEC scheme.

**A. RECONFIGURABLE ARCHITECTURE**

In CNN convolution layers, the dimension of kernel map $K$ in (1) is generally $k_j \times k_d$, where $k_d$ is an odd integer. Among them, $3 \times 3$ or $5 \times 5$ kernel maps are usually employed for many CNN applications. By considering the parameters in (1) and (2), the $N$ values in the MAC operation mapped to the $3 \times 3$ and $5 \times 5$ dimensions are 9 and 25, respectively. Taking the 9-$N$ 8-bit MAC operation for 2D convolution as an example, Fig. 7(a) presents the overall architecture of the corresponding MAC unit using the proposed dual-mode TEC scheme with $w = 1$. As shown in this figure, the preknown $W_k$ is first sent to the coefficient buffer and the precalculated $B_{M1} + PS_{M1}$ (for Mode 1) or $PS_{M2}$ (for Mode 2) is initially set for the calculation. $A_k$ data are prepared depending on Mode 1 or 2. For Mode 1, the 8-bit $A_k$ is the original $(a_7, a_6, \ldots, a_1, a_0)$ digits, while for Mode 2, the form $(a_0, a_5, a_5, a_0, 0)$ is assigned to the 1-bit shift operation (i.e., the “1≪” symbol), as shown in Figs. 5(a)–5(b). For
each iteration of MAC operations, three-term $A_k$ data are sequentially sent to the data buffer and processed by the P.P. generator (Fig. 7(a)), which generates the PP terms (including $pp_{j, i}, n_3, \delta_j$, and $z_j$) according to the Booth encoder result of $W_k$ and mode control. The generated PP terms are then sent to nine FWBMs to perform the 9-N MAC operation with selected TEC of Mode 1 or 2. The PP array for each FWBM is structured using a carry-save adder (CSA) tree, followed by a carry-propagation adder (CPA) summation. The final MAC result is obtained by accumulating the nine FWBM results and optionally adding $B_{M1}$, $PS_{M1}$, or $PS_{M2}$ based on Mode 1 or 2. Fig. 7(b) shows the actual operations performed in one of nine ($k^{th}$) FWBMs for Modes 1, 2 ($A_k' = 0$, and 2 ($A_k = 0$), which correspond to the operations shown in Fig. 3, 5(a), and 6(c), respectively. As shown in Fig. 7(b), multimode operations are required for the proposed TEC scheme as follows. In Mode 1, all $pp_{8-1,0-3}$ terms from the P.P. generator (Fig. 7(a)) are directly mapped to the PP terms associated with $P_{8-1,0-3}$, as arrayed in Fig. 3. In Mode 2 ($A_k' = 0$) condition, because the 1-bit-shifted $A_k'$ has been assigned (Fig. 7(a)), the $pp_{7-1,0-3}$ terms are alternatively mapped to $P_{6-0,0-3}$ at the corresponding digit in the PP array in Fig. 5(a). However, the $pp_{8,0-3}$ values should be adjusted in the P.P. generator to obtain the expected $P_{7,0-3}$ and $P_{8,0}$ terms in Fig. 5(a) using the mapped results in Table 1. Thus, we use a dashed line to frame $pp_{8,0-3}$ in Fig. 7(b) to highlight such a process. Moreover, the operation terms of the last column in the PP array are necessarily selectable ($pp_{7,5,3,0-2}$ or $\delta_{0-2}/n_3$) for Mode 1 or 2 ($A_k' = 0$) conditions, which are mapped to the contents in Figs. 3 and 5(a). In Mode 2 ($A_k = 0$) condition, the P.P. generator is set to assign all $pp_{j, i}, n_3, \delta_j$ terms with “0” (the region included in the red-line frame in Fig. 7(b)) and generate $z_{0-3}$ for optional addition; $z_{0-3}$ is the previously defined parameter for Mode 2 ($A_k = 0$), which equals 1 (for $d_j < 0$) or 0 (for $d_j \geq 0$). Therefore, the contents of the PP array associated with $z_{0-3}$ operations must be optionally adjusted in Mode 2 ($A_k = 0$) condition.

To enable the multimode operations using the aforementioned approach, Fig. 7(c) shows the resultant FWBM configuration, in which the CSA tree and CPA are implemented using full adders (FA) and half adders (HA). In Fig. 7(c), type-1 multiplexers (mux1) are used to select $pp_{7,5,3,0-2}$ or $\delta_{0-2}/n_3$ data for Mode 1 or 2 ($A_k = 0$) conditions. Type-2 multiplexers (mux2) are responsible for the optional addition of $z_{0-3}$, which can be merged with the existing PP array, allowing no extra $z_j$ addition HW costs because sufficient PPs are set by “0” in Mode 2 ($A_k = 0$) condition. In our design, PPs can be practically set to zero using control logic to set $(b_2, b_1, b_0)$ in Table 1 to (0, 0, 0) in the Booth encoder. Based on the 1-bit shift in the numerical range of Mode 1 or 2, the 9-bit FWBM output (including the $w$ digit) can be mapped to $o_{15-01}$ or $o_{14-06}$, as shown in Fig. 7(c).

In Fig. 7(b), multimode operations cause variations in the number of PPs ($N_{pp}$) of the last column (i.e., $w = 1$) of a PP array, which is 4 or 5 for Mode 1 or 2, respectively. Thus, the associated PP array portion (Fig. 7(c)) is structured to address the $N_{pp}$ difference of 4 and 5 using mux1. For $w = 2$ or 3, the FWBM configuration varies with the contents of multimode operations and $N_{pp}$ of the last $w$ column. Multimode operations and the PP array configuration for the $k^{th}$ FWBM in a 9-N 8-bit MAC unit with $w = 2$ and 3 are illustrated in Figs. 8 and 9, respectively, where the addition of $n_2$ or $n_3$ at the correct digit is also considered. As shown in Fig. 8, $N_{pp}$ of the last two columns (i.e., $w = 2$) (denoted as $(N_{pp,2} & N_{pp,1})$) are (4 & 5) and (5 & 4) for Modes 1 and 2, respectively. In Fig. 9, the corresponding $N_{pp}$ count of the last three columns (i.e., $w = 3$) for Modes 1 and 2 are (4 & 5 & 3) and (5 & 3 & 4), respectively. For the operations involving multiple $N_{pp}$, the proposed FWBM structure for $w = 2$ and 3 effectively use mux1 to enable data selection and
the biasing method based on the precalculation (e.g., $B_{M1}$ in Mode 1) is not feasible because both MAC input patterns (i.e., $W_k$ and $A_k$) are indeterminate. Moreover, the presummation method involving the zero MSB for positive/zero patterns (e.g., $PS_{M2}$ in Mode 2) is not applicable owing to the random values of $W_k$ and $A_k$ data. Consequently, we can use (21) and (22) (i.e., $B_{M2}$ generation) as a bias function and only reserve the of MSB “1” (i.e., $PS_{M1}$ in Mode 1) for a general-purpose MAC unit. Considering a 9-$N$ 8-bit MAC unit with $w = 1$ as an example, Fig. 10 shows the operation and $PP$ array for the $k$th FWBM. As shown in this figure, only 3-row MSB “1” of nine FWBMs are presumed to reduce the related HW costs of adding 1. In addition to the operation for $TP_{major}$, additional calculation for $\delta_{0-2}$, $P_{0,3}$, and $n_k$ at the $2^{-2}$ digit is required for TEC biasing. Compared with Figs. 7(b) and 7(c), the operation in Fig. 10 is only in one mode and the $PP$ array is a direct operation-mapped configuration with extra FAs for bias terms; however, the corresponding configuration can avoid extra multiplexing for resource sharing.

V. PERFORMANCE COMPARISONS AND EXPERIMENTS

A. ACCURACY PERFORMANCE COMPARISONS

Considering 2D convolution with $3 \times 3$ or $5 \times 5$ kernel maps as two MAC operation samples (i.e., 9 or 25 $N$), the accuracy performance based on the proposed dual-mode TEC scheme is addressed and evaluated for Modes 1 and 2, respectively, in the following subsections.

1) EVALUATION RELATED TO MODE 1

For accuracy performance, the signal-to-noise ratio (SNR) is the most important parameter. In the targeted MAC operation, SNR can be defined using (23), where $MAC$ is the original full-width MAC result obtained by (2) given $L$-bit ($W_k$, $A_k$) and $MAC_q$ is the quantized MAC outcome with TEC generated using (8), (14), or (22). Moreover, two other parameters, namely, mean error ($\bar{E}$) and mean absolute error ($\bar{|E|}$) (defined in (24)) are employed for accuracy evaluation. In (23) and (24), $E[.]$ indicates the calculation of an averaged value. $MAC_q$ obtained using the DT and PT methods (specified in (25)) is also considered for a complete comparison involving lowest (DT) and highest (PT) operation accuracy.

$$SNR(dB) = 10 \cdot \log_{10} \left( \frac{E[|MAC|^2]}{E[|MAC - MAC_q|^2]} \right)$$  \hspace{1cm} (23)$$

$$\bar{E} = E[MAC - MAC_q]/2^L$$ \hspace{1cm} (24)
The comparison results demonstrate that our Mode 1 TEC scheme outperforms all listed TEC-based methods in terms of minimizing the mean error. Moreover, a higher SNR can be achieved with larger $N$ using Mode 1 TEC scheme relative to other TEC works because the rounding noise at the final stage in our design generally has a stable magnitude, although the value of signal power increases with $N$. The leading performance of Mode 1 TEC scheme with respect to the mean absolute error is also exhibited in Table 6, which is consistent with the SNR results listed in Tables 3 and 4.

2) EVALUATION RELATED TO MODE 2

As discussed in Section III/B, when an FWBM has zero-valued input data in Mode 2, directly using Mode 1 TEC scheme potentially induces unwanted biasing in the FWBM. Table 7 presents SNR degradation when positive/zero patterns in Mode 2 are applied to the 9/25-N MAC operation using Mode 1 TEC scheme with $w = 1$. In Table 7, TEC-M1 (original) corresponds to the original results of Mode 1 TEC with $w = 1$ obtained in Tables 3 and 4 and TEC-M1 (Mode 2) indicates the condition that the original random input patterns of Mode 1 TEC design are replaced by the positive or zero data assumed in half (1/2) probability to meet the feature of Mode 2. As shown in Table 7, the decrease in SNR is exhibited in the TEC-M1 (Mode 2) condition and the associated SNR performances are inferior to many SNR performances of PACS or BSCP methods, as listed in Tables 3 and 4. Therefore, we alternatively used the proposed Mode 2 TEC scheme to reevaluate the accuracy performance based on the positive/zero input patterns in Mode 2. Tables 8 and 9 present the SNR values of the 9-N and 25-N MAC operations assigned by the positive/zero input data of (1/2) probability, respectively, for various TEC schemes. In Tables 8 and 9, the contrast GPEB, PACS, and BSCP schemes are assumed to possess an additional function that detects the zero input patterns and disables biasing for MAC operation, the $N$ sets of mean error are accordingly accumulated and produce significant overall impairments. As shown in Tables 3—5, the mean error issue usually results in SNR degradations with an increase in $N$ for most situations of TEC methods, in which DT is the worst case. Alternatively, the proposed Mode 1 TEC scheme outperforms all listed TEC-based methods in terms of minimizing the mean error. Moreover, a higher SNR can be achieved with larger $N$ using Mode 1 TEC scheme relative to other TEC works because the rounding noise at the final stage in our design generally has a stable magnitude, although the value of signal power increases with $N$. The leading performance of Mode 1 TEC scheme with respect to the mean absolute error is also exhibited in Table 6, which is consistent with the SNR results listed in Tables 3 and 4.

### Table 3. SNR values of the 9-N MAC operation for various schemes.

| Scheme     | $L=8$ | $L=10$ | $L=12$ | $L=14$ | $L=16$ |
|------------|-------|--------|--------|--------|--------|
| GPEB [28]  | 33.69 | 41.39  | 55.13  | 68.00  | 75.45  |
| PACS [31]  | 35.16 | 45.22  | 57.09  | 68.90  | 80.63  |
| BSCP [32]  | 35.72 | 46.72  | 58.73  | 70.16  | 81.86  |
| TEC-M1     | 37.92 | 49.49  | 61.11  | 72.55  | 83.98  |
| w=2 GPEB [28] | 34.79 | 48.22  | 57.81  | 70.86  | 83.51  |
| PACS [31]  | 36.45 | 48.30  | 60.15  | 72.10  | 83.97  |
| TEC-M1     | 43.27 | 54.53  | 66.64  | 77.84  | 89.37  |
| w=3 GPEB [28] | 36.89 | 49.01  | 60.08  | 72.61  | 84.88  |
| PACS [31]  | 37.14 | 49.08  | 61.11  | 73.04  | 85.01  |
| TEC-M1     | 46.91 | 57.46  | 69.43  | 81.15  | 92.74  |
| DT         | 14.07 | 23.65  | 34.08  | 44.70  | 55.62  |
| PT         | 47.16 | 58.72  | 71.06  | 83.02  | 95.08  |

To evaluate the accuracy performance of our design, we compared the SNR performance of 9-N and 25-N MAC operations, either using the proposed Mode 1 TEC scheme or directly using conventional TEC-enabled FWBMs. For comparison samples, we selected the FWBM designs whose TEC function is based on probability estimation methods ([25]–[32]) as the aimed group, which are the GPEB [28], PACS [31], and BSCP [32] methods. The GPEB scheme [28] (an extension of [26] and [27]) generated the TEC bias in constant-valued 1-bit or 2-bit digits. The PACS work [31] accumulated the number of non-zero $d_j$ terms at the $2^{w-1}$ digit as TEC biasing, which is an enhanced design of [29] and [30]. PACS-similar bias functions were also derived in [25] or [24] (a simulation-based work), but only the case of $w=1$ was included; and their accuracy is lower than or closer to that of PACS [31]. The BSCP scheme [32] derived a bias function ((19) in [32]) to improve the accuracy, considering the magnitude and sign of non-zero $d_j$ terms. However, [32] only included the case of $w=1$, and the accuracy decreased in case the modified bias function ((20) in [32]) was employed for practical bias generation. In this study, the evaluation data for the BSCP method (in Tables 3—9) were obtained using the high-accuracy bias formulation presented in [32] ((19) in [32]). Tables 3 and 4 present the SNR results for the 9-N and 25-N MAC operations using the aforementioned TEC schemes, respectively. As illustrated in Tables 3 and 4, the proposed Mode 1 TEC scheme (i.e., the TEC-M1 item) can achieve the best SNR performance except the PT method. The comparison results demonstrate that our Mode 1 TEC scheme can further improve the operation accuracy by the use of hybrid probabilistic and deterministic $P$ values with the design consideration of the overall MAC operation.

Tables 5 and 6 list the mean error and mean absolute error values for the 9-N and 25-N MAC operations using various TEC schemes, respectively. Various TEC methods generally lead to different mean error effects on FWBM. However, in $N$ MAC operation, the $N$ sets of mean error are accordingly accumulated and produce significant overall impairments. As shown in Tables 3—5, the mean error issue usually results in SNR degradations with an increase in $N$ for most situations of TEC methods, in which DT is the worst case. Alternatively, the proposed Mode 1 TEC scheme outperforms all listed TEC-based methods in terms of minimizing the mean error. Moreover, a higher SNR can be achieved with larger $N$ using Mode 1 TEC scheme relative to other TEC works because the rounding noise at the final stage in our design generally has a stable magnitude, although the value of signal power increases with $N$. The leading performance of Mode 1 TEC scheme with respect to the mean absolute error is also exhibited in Table 6, which is consistent with the SNR results listed in Tables 3 and 4.

### Table 4. SNR values of the 25-N MAC operation for various schemes.

| Scheme     | $L=8$ | $L=10$ | $L=12$ | $L=14$ | $L=16$ |
|------------|-------|--------|--------|--------|--------|
| GPEB [28]  | 33.69 | 38.42  | 52.79  | 67.11  | 71.86  |
| PACS [31]  | 35.83 | 44.66  | 56.04  | 67.90  | 79.67  |
| BSCP [32]  | 35.71 | 46.31  | 58.45  | 70.06  | 81.88  |
| TEC-M1     | 39.49 | 49.98  | 61.43  | 72.71  | 84.23  |
| w=2 GPEB [28] | 38.45 | 48.01  | 59.89  | 71.29  | 83.46  |
| PACS [31]  | 35.87 | 48.01  | 59.89  | 71.29  | 83.46  |
| TEC-M1     | 45.02 | 56.51  | 67.57  | 78.72  | 90.16  |
| w=3 GPEB [28] | 37.75 | 48.83  | 59.14  | 71.89  | 84.71  |
| PACS [31]  | 37.03 | 48.90  | 61.09  | 72.76  | 84.85  |
| TEC-M1     | 48.62 | 60.31  | 72.15  | 83.44  | 95.05  |
| DT         | 8.97  | 19.11  | 29.74  | 40.26  | 51.20  |
| PT         | 51.12 | 63.38  | 75.43  | 87.41  | 99.50  |

\[
\begin{align*}
\text{MAC}_{q}\left|_{DT} &= \sum_{k=1}^{N} MP^{(k)} \\
\text{MAC}_{q}\left|_{PT} &= \sum_{k=1}^{N} MP^{(k)} \\
&+ R \left\{ \sum_{k=1}^{N} (TP_{major}^{(k)} + TP_{minor}^{(k)}) \right\} . 2^L
\end{align*}
\]
TABLE 5. Mean error values of the 9-N and 25-N MAC operations for various schemes.

| Scheme   | GEPB[28] | PACS[31] | BSCP[32] | TEC-M1   | GEPB[28] | PACS[31] | TEC-M1   | GEPB[28] | PACS[31] | TEC-M1   | DT       | PT       |
|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| L=8      | -0.0385  | 0.3624   | 0.0763   | -0.0032  | -0.8253  | 0.1630   | -0.0183  | -0.4547  | 0.0810   | -0.0089  | 13.7216  | -0.0016  |
| L=10     | 1.6671   | -0.4965  | -0.0297  | -0.0310  | 0.2051   | 0.2516   | -0.0141  | 0.1429   | 0.1157   | -0.0076  | 16.8309  | 0.0007   |
| L=12     | -1.0853  | 0.6203   | 0.0206   | -0.0240  | 0.8466   | 0.2589   | -0.0134  | 0.4577   | 0.1257   | -0.0081  | 20.2863  | 0.0006   |
| L=14     | 0.5626   | 0.5431   | -0.0109  | -0.0170  | -0.5653  | 0.2807   | -0.0091  | -0.2830  | 0.1403   | -0.0052  | 25.5902  | -0.0004  |
| L=16     | -2.1636  | -0.5714  | 0.0044   | -0.0139  | 0.2815   | 0.2746   | -0.0068  | 0.1413   | 0.1391   | -0.0037  | 27.0003  | -0.0003  |

TABLE 6. Mean absolute error values of the 9-N and 25-N MAC operations for various schemes.

| Scheme   | GEPB[28] | PACS[31] | BSCP[32] | TEC-M1   | GEPB[28] | PACS[31] | TEC-M1   | GEPB[28] | PACS[31] | TEC-M1   | DT       | PT       |
|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| L=8      | 1.2122   | 1.0082   | 0.9094   | 0.7203   | 1.0445   | 0.8679   | 0.3935   | 0.8215   | 0.7692   | 0.2571   | 13.7216  | 0.25     |
| L=10     | 1.8254   | 1.1648   | 0.9841   | 0.7256   | 0.8417   | 0.8088   | 0.4005   | 0.7530   | 0.7543   | 0.2880   | 16.8309  | 0.25     |
| L=12     | 1.4640   | 1.1763   | 0.9694   | 0.7498   | 1.1594   | 0.8674   | 0.4209   | 0.8593   | 0.7807   | 0.2967   | 20.2863  | 0.25     |
| L=14     | 1.2992   | 1.1962   | 1.0278   | 0.7960   | 0.9883   | 0.8721   | 0.4485   | 0.8053   | 0.7765   | 0.3036   | 23.5902  | 0.25     |
| L=16     | 2.3053   | 1.2577   | 1.0738   | 0.8474   | 0.8747   | 0.8409   | 0.4577   | 0.7687   | 0.7608   | 0.3151   | 27.0003  | 0.25     |

B. HARDWARE PERFORMANCE COMPARISONS

A comparison of HW performances of the 9-N and 25-N MAC units, including area, power consumption, and critical-path delay for various TEC schemes, is presented in Tables 10 and 11, respectively, in which HW parameters were generated by logic synthesis using the Synopsys Design Compiler tool with the TSMC 40-nm typical standard cell library and the TEC-Dual items represent the proposed work using reconfigurable structures. In our evaluation, data and coefficient buffers are excluded (ref. Fig. 7(a)), and we directly redesigned MAC units for the contrast TEC schemes based on their presented works and re-performed the logic synthesis to obtain the HW performances’ data for those.
TABLE 10. Comparison of HW performance (area, delay, and power consumption) of a 9-N MAC unit for various schemes.

| Scheme     | \( L=8 \) | Area (\( \mu m^2 \)) | Delay (ns) | Power (mW) | Area (\( \mu m^2 \)) | Delay (ns) | Power (mW) | Area (\( \mu m^2 \)) | Delay (ns) | Power (mW) | Area (\( \mu m^2 \)) | Delay (ns) | Power (mW) |
|------------|-----------|----------------------|------------|-----------|----------------------|------------|-----------|----------------------|------------|-----------|----------------------|------------|-----------|
| PT         | 7075      | 2.75                 | 3.204      | 10131     | 3.20                 | 4.919      | 14236     | 3.79                 | 6.882      | 18669     | 4.19                 | 8.865      | 23954     | 4.72                 | 12.282     |
| GPEB [28]  | 3710      | 2.33                 | 1.673      | 5378      | 2.73                 | 2.525      | 7433      | 3.08                 | 3.538      | 9740      | 3.48                 | 4.778      | 12380     | 3.85                 | 6.294      |
| PACS [31]  | 3879      | 2.47                 | 1.739      | 5549      | 2.82                 | 2.618      | 7710      | 3.14                 | 3.749      | 10082     | 3.67                 | 4.989      | 12786     | 4.07                 | 6.455      |
| BSCP [32]  | 3922      | 2.42                 | 1.751      | 5667      | 2.81                 | 2.634      | 7871      | 3.11                 | 3.767      | 10272     | 3.69                 | 5.033      | 12923     | 4.04                 | 6.519      |
| TEC-Dual   | 3756      | 2.38                 | 1.712      | 5450      | 2.77                 | 2.609      | 7519      | 3.11                 | 3.718      | 9850      | 3.60                 | 4.937      | 12464     | 3.91                 | 6.414      |

TABLE 11. Comparison of HW performance (area, delay, and power consumption) of a 25-N MAC unit for various schemes.

| Scheme     | \( L=8 \) | Area (\( \mu m^2 \)) | Delay (ns) | Power (mW) | Area (\( \mu m^2 \)) | Delay (ns) | Power (mW) | Area (\( \mu m^2 \)) | Delay (ns) | Power (mW) | Area (\( \mu m^2 \)) | Delay (ns) | Power (mW) |
|------------|-----------|----------------------|------------|-----------|----------------------|------------|-----------|----------------------|------------|-----------|----------------------|------------|-----------|
| PT         | 3943      | 2.56                 | 1.848      | 5597      | 2.78                 | 2.698      | 7763      | 3.21                 | 3.839      | 10046     | 3.63                 | 5.089      | 12768     | 3.92                 | 6.629      |
| GPEB [28]  | 4077      | 2.64                 | 1.894      | 5817      | 2.87                 | 2.825      | 7997      | 3.30                 | 3.968      | 10362     | 3.72                 | 5.318      | 13063     | 4.20                 | 6.779      |
| PACS [31]  | 3946      | 2.56                 | 1.872      | 5663      | 2.85                 | 2.802      | 7817      | 3.24                 | 3.941      | 10148     | 3.69                 | 5.322      | 12826     | 4.05                 | 6.757      |
| BSCP [32]  | 4016      | 2.63                 | 1.918      | 5795      | 2.82                 | 2.847      | 7914      | 3.29                 | 3.994      | 10212     | 3.70                 | 5.264      | 13029     | 4.05                 | 6.927      |
| PACS [31]  | 3410      | 2.71                 | 1.958      | 5945      | 2.93                 | 2.890      | 8169      | 3.41                 | 4.157      | 10576     | 3.81                 | 5.521      | 13321     | 4.27                 | 7.088      |
| TEC-Dual   | 4094      | 2.65                 | 1.961      | 5851      | 2.91                 | 2.865      | 8011      | 3.36                 | 4.135      | 10422     | 3.77                 | 5.508      | 13125     | 4.14                 | 7.063      |
| DT         | 3377      | 1.93                 | 1.439      | 4966      | 2.31                 | 2.194      | 6937      | 2.72                 | 3.138      | 9158      | 3.04                 | 4.264      | 11737     | 3.42                 | 5.647      |

works. To ensure a fair comparison, we added multiplexers and control logic to enable bias-disable functions in the contrast designs wherein FWBM could produce zero-valued outputs when zero input patterns were detected. Thus, the HW behavior of all designs in Tables 10 and 11 can be synchronized to the accuracy results shown in Tables 8 and 9. Moreover, according to [24], [32], and [44], the bias function for [32] in Tables 10 and 11 was structured using a general sorting circuit based on [24] to avoid the addition of negative digit values. The comparison results show that the non-TEC-based PT and DT methods require the highest and lowest HW costs, respectively, which are rational to their accuracy performances. The main HW feature of the proposed design can perform dual-mode TEC operations using the minimal CSA elements of a PP array in cooperation with additional accumulation resources (i.e., addition for \( B_{M1}, PS_{M1}/PS_{M2} \), and cross-FWBM TP\(_{major} \)). Moreover, our MAC unit can exclude the individual output-rounding stage in each FWBM and merge the final rounding operation in \( B_{M1} \) accumulation. However, to enable reconfigurable operations, extra data-selection multiplexers (for \( z_j \) and \( \delta_j \)) and control logic are required in our design. For TEC-enabled schemes in Tables 10 and 11, the proposed MAC unit has a smaller area, delay, and power consumption than most PACS [31] and BSCP [32] conditions because one more column of CSA addition or extra sorting circuit is required in each FWBM for the PACS or BSCP designs, respectively. Comparatively, the GPEB [28] scheme achieved better HW performance owing to its use of a simple constant 1- or 2-bit bias; however, the accuracy results of GPEB (Tables 3 and 4) are lower than those of the proposed scheme. Two design metrics, i.e., area-delay-error product (\( ADEP \)) and power-delay-error product (\( PDEP \)), defined in (26), can be employed to evaluate the overall design efficiency [32], where \( MSE \) is the mean square error (i.e., the \( E\left[ (MAC - MAC_q)^2 \right] \) term in (23)). Small \( ADEP \) or \( PDEP \) values exhibit good design efficiency considering both HW and accuracy performances. For the TEC-enabled designs (i.e., GPEB, PACS, and BSCP), Fig. 11 shows the decrease in the percentage values of \( ADEP \) and \( PDEP \) results of the 9-N MAC unit using the proposed dual-mode TEC scheme relative to other TEC methods for various \( L\)-bit operands. The same evaluation for the 25-N MAC unit is also shown in Fig. 12. In Figs. 11 and 12, the \( z_j \) (G), \( \delta_j \) (P), and \( \delta_j \) (B) items (shown in dark colors) represent the \( ADEP \) reductions achieved using our design associated with the GPEB(G), PACS(P), and BSCP(B) methods.
respectively. Similarly, the $Pr_{(G)}$, $Pr_{(P)}$, and $Pr_{(B)}$ items (shown in light colors) correspond to the PDEP reductions.

In this evaluation (Figs. 11 and 12), we use MSE values obtained in Mode 1 operations as the general cases. Alternatively, the proposed 9-N and 25-N MAC units demonstrate the least ADEP and PDEP values, likely achieving a minimum reduction of more than 40% for ADEP and PDEP in all cases of $L$ and $w$.

$$\begin{align*}
\text{ADEP} &= \text{Area} \times \text{Delay} \times \text{MSE} \\
\text{PDEP} &= \text{Power Consumption} \times \text{Delay} \times \text{MSE}
\end{align*}$$

C. DESIGN IMPLEMENTATION AND EXPERIMENTS

To verify the proposed design, we implemented 9-N and 25-N MAC units for $3 \times 3$ and $5 \times 5$ 2D convolution acceleration, respectively, on an SW–HW codesign platform using the Xilinx Zynq-7000 system-on-chip (SoC) FPGA device. An experiment was performed to demonstrate the handwritten digit classification using a simplified LeNet-5 [45] CNN model. The experimental CNN comprises two convolution and maximum pooling layers as the model structure summarized in Table 12. For verification, two scenarios were set for mode selection. In Scenario 1, the input feature map was the original handwritten digit pattern, which only had positive/zero pixel values, and an ReLU function was employed after performing 2D convolution. Accordingly, the 9-N and 25-N MAC units were configured for Mode 2 operations in this scenario. In Scenario 2, we deliberately added noise to the input data and used the “tanh” function instead of ReLU to generate nonzero patterns in the convolution layer. Therefore, Scenario 2 allowed the 9-N and 25-N MAC units to operate in Mode 1 condition. The division of HW and SW responsibilities in our experiment was as follows. The HW side was responsible for 2D convolution, while the SW side performed residual operations, including the summation of the convolution results, addition of an offset, activation function, maximum pooling, FC execution, and system control. The development flow of our experiment includes the training and generation of CNN models, HW design of the MAC units, and SW–HW codesign and experiment on FPGA, as described in Fig. 13. In a typical CNN inference execution, the low operation precision (e.g., 8-bit width or even fewer) is sufficient to preserve the classification accuracy [12], [15]. However, to achieve diverse precision required by different applications, a sufficiently high bit width (e.g., 16 bits) is considered in several CNN accelerator designs [3]–[5]. In our design, the 9-N and 25-N MAC units were operated using 16-bit operands with $w = 1$, which can achieve a high recognition rate for handwritten digit classification.

A Zynq-7000 SoC-FPGA device was integrated on an ARM CPU at the HW side, including FPGA logics and block RAMs. The ARM CPU communicates with the HW side through the AXI bus. Fig. 14 shows the schematic of the setup of our design implementation. The MAC unit and RAMs were implemented at the HW side for storing the coefficients ($W_k$) and data ($A_k$) to perform the 2D convolution operation.
Moreover, the SW commands were executed using the ARM CPU and external memory (i.e., DDR). When the 2D convolution of each layer was actuated, the coefficients and precalculated TEC-related terms (i.e., $PS_{M1}/B_{M1}$ for Mode 1 and $PS_{M2}$ for Mode 2) stored in external DDR were accessed and moved to a register-based coefficient RAM at the HW side using the DMA controller. Subsequently, the layer-inputted feature maps were fetched from the DDR to the HW block RAM via a DMA transmission. The MAC unit then accessed the $W_k$ and $A_k$ values for MAC operations and then sent the calculated result to DDR through DMA for follow-up SW processing. Through two separate rounds of HW acceleration for the two layers of 2D convolution, the entire CNN operations (Table 12) were completed to obtain the final inference results based on the SW–HW codesign approach. As shown in Fig. 13, the FPGA-based inference outcomes were further compared with the results generated using the fixed-point CNN model for verification. Table 13 lists the main HW resource usage on a Xilinx/Zynq-7000 (XC7Z020-CLG484) device for the FPGA design of the proposed 9-N and 25-N MAC units (including block RAMs for storing data of feature maps), and the items include lookup-table slices (LUT), flip-flop slices (FF), and block RAMs (BRAM). Relative to the prior FPGA designs for CNN acceleration (e.g., [7], [8]), our work did not utilize existing FPGA DSP modules for the multiplier implementation because the TEC-based FWBMs in our MAC units were manually designed by structural modeling with FA and HA cells. The HW performance of giga operations per second (GOPs) and the inference (classification) result of the recognition rate for handwritten patterns are also listed in Table 13. The GOPs performances of our design were obtained using a 50-MHz clock rate with values converted from the GMACs multiplied by 2 because each MAC execution contains multiplication and addition operations [7]. In our evaluation (i.e., the experiments based on Table 11), the CNN execution time using our FPGA-based design can be 5.7 times faster than the computation time using only the SW-based ARM CPU. Although the CNN acceleration and GOPs performances achieved in our experiment are not high, the proposed FPGA-based design focuses on MAC unit verification, which was developed with no emphasis on achieving high GOPs or significant acceleration effects using many parallel PEs or data reuse memory schemes, as presented in other FPGA-related studies [7], [8].

### VI. CONCLUSION

In this paper, we presented a MAC unit that could support dual-mode TEC using FWBMs for CNN inference operations. The proposed dual-mode TEC scheme was derived using hybrid probabilistic/deterministic digit values to provide tailored and high-accuracy TEC functions for two operation modes. For the general CNN model in which the MAC units handle random input patterns, the TEC scheme of Mode 1 was proposed to optimize the overall MAC accuracy using a global bias value. For the MAC unit with positive/zero input patterns in an ReLU-based CNN model, the TEC scheme of Mode 2 was proposed to achieve high accuracy based on a combination of the input-adaptive bias term for each FWBM. The proposed TEC method could yield high accuracy of the MAC unit, while preserving the HW efficiency. By exploiting the feature that the $W_k$ coefficients were preknown, the TEC-related terms (i.e., $PS_{M1}, B_{M1}$, or $PS_{M2}$) could be precalculated and most multimode operations in an FWBM could be performed using existing PP terms. Thus, the resultant MAC unit only required minimal CSA resources to enable the proposed TEC operations using a reconfigurable structure with high HW efficiency. Compared with the 9-N or 25-N MAC operations that directly employed state-of-the-art FWBMs with TEC, the 9-N or 25-N MAC operations using the proposed TEC scheme outperformed contrast samples in terms of operation accuracy in cases of both Modes 1 and 2. Competitive HW performances, including area, critical-path delay, and power consumption, of the proposed 9-N
and 25-N MAC units were also shown and compared with those of the TEC-enabled contrast designs. Moreover, the proposed design could yield a significant overall design efficiency in terms of \textit{ADEP} and \textit{PDEP} values. Furthermore, the resultant 9-N and 25-N MAC units were verified by the SW–HW codesign approach using the Xilinx Zynq-7000 SoC-FPGA device. The FPGA-based verification demonstrated the Lenet-5 CNN model for handwritten digit classification. Although the MAC unit based on the proposed design could preserve the accuracy and reduce the HW costs, additional design contents for multiple modes were required and the contribution may be confined to CNN applications. However, an extended design based on Mode 2 TEC scheme can be applied to the design of the general-purpose MAC unit. Moreover, a systematic design method for determining the TEC contents and HW configurations can be developed in our future work.

\textbf{ACKNOWLEDGMENT}

Song-Nien Tang thanks Mr. J.-H Yeh for the supports in HW pre-coding and verification.

\textbf{REFERENCES}

[1] Z. Zhou, X. Chen, E. Li, L. Zeng, K. Luo, and J. Zhang, “Edge intelligence: Paving the last mile of artificial intelligence with edge computing,” \textit{Proc. IEEE}, vol. 107, no. 8, pp. 1738–1762, Aug. 2019.
[2] X. Wang, Y. Han, C. Wang, Q. Zhao, X. Chen, and M. Chen, “In-edge AI: Intelligentizing mobile edge computing, caching and communication by federated learning,” \textit{IEEE Netw.}, vol. 33, no. 5, pp. 156–165, Sep. 2019.
[3] L. Du, Y. Du, Y. Li, J. Su, Y.-C. Kuan, C.-C. Liu, and M.-C.-F. Chang, “A reconfigurable streaming deep convolutional neural network accelerator for Internet of Things,” \textit{IEEE Trans. Circuits Syst. I, Reg. Papers}, vol. 65, no. 1, pp. 198–208, Jan. 2018.
[4] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, “Eyeris: An energy-efficient reconfigurable accelerator for deep convolutional neural networks,” \textit{IEEE J. Solid-State Circuits}, vol. 52, no. 1, pp. 127–138, Jan. 2017.
[5] J. Jo, S. Cha, D. Rho, and I.-C. Park, “DSIP: A scalable inference accelerator for convolutional neural networks,” \textit{IEEE J. Solid-State Circuits}, vol. 53, no. 2, pp. 605–618, Feb. 2018.
[6] X. Chen and Z. Yu, “A flexible and energy-efficient convolutional neural network acceleration with dedicated ISA and accelerator,” \textit{IEEE Trans. Very Large Scale Integr. (VLSI) Syst.}, vol. 26, no. 7, pp. 1408–1412, Jul. 2018.
[7] S. Moini, B. Alizadeh, M. Emad, and R. Ebrahimpour, “A resource-limited hardware accelerator for convolutional neural networks in embedded vision applications,” \textit{IEEE Trans. Circuits Syst. II, Exp. Briefs}, vol. 64, no. 10, pp. 1217–1221, Oct. 2017.
[8] L. Gong, C. Wang, X. Li, H. Chen, and X. Zhou, “MALOC: A fully pipelined FPGA accelerator for convolutional neural networks with all layers mapped on chip,” \textit{IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.}, vol. 37, no. 11, pp. 2601–2612, Nov. 2018.
[9] H. Zhang, H. Jae Lee, and S.-B. Ko, “Efficient fixed/floating-point merged mixed-precision multiply-accumulate unit for deep learning processors,” in \textit{Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)}, May 2018, pp. 1–5.
[10] H. Zhang, D. Chen, and S.-B. Ko, “New flexible multiple-precision multiply-accumulate unit for deep neural network training and inference,” \textit{IEEE Trans. Comput.}, vol. 69, no. 1, pp. 26–38, Jan. 2020.
[11] H. Zhang, J. He, and S.-B. Ko, “Efficient post multiply-accumulate unit generator for deep learning applications,” in \textit{Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)}, May 2019, pp. 1–5.
[12] V. Camus, L. Mei, C. Enz, and M. Verhelst, “Review and benchmarking of precision-scalable multiply-accumulate unit architectures for embedded neural-network processing,” \textit{IEEE J. Emerg. Sel. Topics Circuits Syst.}, vol. 9, no. 4, pp. 697–711, Dec. 2019.
S.-N. Tang, Y.-S. Han: High-Accuracy Hardware-Efficient Multiply–Accumulate (MAC) Unit

[35] G. Zhong, A. Dubey, C. Tan, and T. Mitra, “Synergy: An HW/SW framework for high throughput CNNs on embedded heterogeneous SoC,” ACM Trans. Embedded Comput. Syst., vol. 18, no. 2, pp. 1–23, Apr. 2019.

[36] D. Manatunga, H. Kim, and S. Mukhopadhyay, “SP-CNN: A scalable and programmable CNN-based accelerator,” IEEE Micro, vol. 35, no. 5, pp. 42–50, Sept./Oct. 2015.

[37] V. G. Oklobdzija, D. Villeger, and S. S. Liu, “A method for speed optimized partial product reduction and generation of fast parallel multipliers using an algorithmic approach,” IEEE Trans. Comput., vol. 45, no. 3, pp. 294–306, Mar. 1996.

[38] Y. He and C. Chang, “A new redundant binary booth encoding for fast 2n-bit multiplier design,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 56, no. 6, pp. 1192–1201, Jun. 2009.

[39] D. De Caro, N. Petra, A. G. M. Strollo, F. Tessitore, and E. Napoli, “Fixed-width multipliers and multipliers-accumulators with min-max approximation error,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 60, no. 9, pp. 2375–2388, Sep. 2013.

[40] S. Han, J. Pool, J. Tran, and W. Dally, “Learning both weights and connections for efficient neural network,” in Proc. Adv. Neural Inf. Process. Syst., vol. 28, 2015, pp. 1135–1143.

[41] C. Dong, C. C. Loy, K. He, and X. Tang, “Image super-resolution using deep convolutional networks,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 38, no. 2, pp. 295–307, Feb. 2016.

[42] W. Choi, K. Choi, and J. Park, “Low cost convolutional neural network accelerator based on bi-directional filtering and bit-width reduction,” IEEE Access, vol. 6, pp. 14734–14746, 2018.

[43] P. Gysel, M. Motamedi, and S. Ghiasi, “Hardware-oriented approximation of convolutional neural networks,” 2016, arXiv:1604.03168. [Online]. Available: http://arxiv.org/abs/1604.03168

[44] R. R. Osorio and G. Rodriguez, “Truncated SIMD multiplier architecture for approximate computing in low-power programmable processors,” IEEE Access, vol. 7, pp. 56353–56366, 2019.

[45] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” Proc. IEEE, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

SONG-NIEN TANG (Member, IEEE) was born in Kaosiung, Taiwan, in 1972. He received the B.S. degree in electrical engineering from Fu Jen Catholic University, Taipei, Taiwan, in 1996, the M.S. degree in communication engineering from National Chiao Tung University, Hsinchu, Taiwan, in 1998, and the Ph.D. degree in electrical engineering from National Tsing Hua University, Hsinchu, in 2011. From 1999 to 2001, he was with SiS Inc., Taiwan, working in ADSL system and IC designs. From 2001 to 2005, he was with Realtek Inc., Taiwan, developing systems and ICs for WLAN IEEE 802.11a/g/n applications. He joined the Industrial Technology Research Institute (ITRI), Hsinchu, to develop medical image systems and ICs. From 2014 to 2015, he was with Realtek Inc., as a Project Manager of 40G/25G ethernet system and IC architecture designs. Since 2015, he has been with the Faculty of the Information and Computer Engineering, Chung Yuan Christian University. His research interests include IC designs for biomedical and communication systems, hardware accelerator for deep learning applications, SoC developments, and software/hardware co-design.

YU-SHIN HAN was born in Nantou, Taiwan, in 1997. She received the B.S. degree in information and computer engineering from Chung Yuan Christian University, Taoyuan, Taiwan, in 2020. She is currently pursuing the M.S. degree with the Artificial Intelligence Graduate Program, National Chiao Tung University, Hsinchu, Taiwan. Her research interests include software design, computer algorithm, and artificial intelligence system developments.