Design and Analysis of Energy-Efficient Dynamic Range Approximate Logarithmic Multipliers for Machine Learning

Peipei Yin, Chenghua Wang, Haroon Waris, Weiqiang Liu, *Senior Member, IEEE,*
Yinhe Han, *Senior Member, IEEE,* and Fabrizio Lombardi, *Fellow, IEEE*

Abstract—Approximate computing provides an emerging approach to design high performance and low power arithmetic circuits. The logarithmic multiplier (LM) converts multiplication into addition and has inherent approximate characteristics. In this paper, dynamic range approximate LMs (DR-ALMs) for machine learning applications are proposed; they use Mitchell's approximation and a dynamic range operand truncation scheme. The worst case (absolute and relative) errors for the proposed DR-ALMs are analyzed. The accuracy and the hardware overhead of these designs are provided to select the best approximate scheme according to different metrics. The proposed DR-ALMs are compared with the conventional LM with exact operands and previous approximate multipliers; the results show that the power-delay product (PDP) of the best proposed DR-ALM (DR-ALM-6) are decreased by up to 54.07% with the mean relative error distance (MRED) decreasing by 21.30% compared with 16-bit conventional design. Case studies for three machine learning applications show the viability of the proposed DR-ALMs. Compared with the exact multiplier and its conventional counterpart, the back-propagation classifier with DR-ALMs with a truncation length larger than 4 has a similar classification result for the three datasets; the K-means clustering application with all DR-ALMs has a similar clustering result for four datasets; and the handwritten digit recognition application with DR-ALM-5 or DR-ALM-6 for LeNet-5 achieves similar or even slightly higher recognition rate.

Index Terms—Approximate computing, logarithmic multiplier, operand truncation, low power.

---　✦　---

1 INTRODUCTION

As Dennard scaling is coming to an end, computer technology is unlikely to further improve performance due to severe power limitations. Power consumption has become an significant challenge. A rapidly expanding computing paradigm is approximate computing [1] - [3]; approximate computing reduces power requirements at the cost of precision. Many computing applications which are inherently error-resilient (such as signal processing, machine learning, and data mining) do not always require an exact computation. Using approximate computing, results are still of a reasonable good quality; therefore, full accuracy is not always necessary, thus enabling approximate computing to be employed for this type of applications.

Arithmetic units consumes a substantial amount of energy. As basic operations for an arithmetic processor, addition and multiplication have been extensively studied for approximate computing and reduce power consumption [4]. Approximate adder designs mainly include speculative adders [5], segmented adders [6], carry select-skip adders [7], and transistor-level approximate adders [8]. An approx-

imate floating-point adder has also been studied in [9].

Multiplication is particularly attractive to approximation due to the high cost incurred in terms of power, area, and delay. Generally, a multiplier consists of three stages: partial product (PP) generation [10] - [14], PPs accumulation [15] - [23], and a final addition. So methodologies based on these three stages are also used for the approximate design of a multiplier. A so-called underdesigned multiplier (UDM) [10] based on inaccurate 2×2 underdesigned multiplier bloacks has been proposed by changing an entry of the Karnaugh-map. A generalized design of UDM has been further studied with carry-in prediction during the PP accumulation stage [11]. However, UDMs can only perform unsigned multiplication. Approximate Booth encoders have also been studied. In [12], approximate radix-8 Booth multipliers have been proposed by using a 2-bit approximate recoding adder to generate the ×3 multiplication. Two approximate Booth multipliers based on radix-4 modified Booth encoding (MBE) algorithms have been proposed in [14]. In [13], high-radix approximate Booth multipliers have been proposed based on a hybrid radix encoding. A truncation scheme when applied to a PP tree is often used to simplify accumulation. An inexact multiplier has been proposed by replacing the least significant columns of the PPs as a constant carry [15]. In [16], a correction constant according to both the reduction and the rounding errors is added to the truncated PP. A truncated multiplier with variable correction has been proposed in [17]. Recently some error compensation strategies have been proposed to further improve the accuracy of fixed-width Booth multipliers [15], [18]. A PP perforation method has been proposed for

- P. Yin, C. Wang, H. Waris, and W. Liu are with the College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, 211106, China.
  E-mail: {ppyin, chwang, haroonwaris, liuweiqiang}@nuaa. edu.cn.
- Y. Han is with the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190 China.
  E-mail: yinhes@ict.ac.cn.
- F. Lombardi is with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA, 40125.
  E-mail: lombardi @ece.neu.edu.

creating approximate hardware multipliers [19]; successive rows of PPs are omitted before accumulation. Compressors are widely used to accelerate the accumulation of PPs in the design of a high-speed multiplier. An inaccurate 4:2 counter has been used to design an approximate 4-bit Wallace multiplier which is then used as a block in larger size multipliers [20]. Two approximate 4:2 compressors have been proposed in [21] and used in a 8×8 Dadda multipliers. A fast approximate adder that cuts the carry propagation chain has been used in an approximate multiplier [22]. Four 8×8 multipliers have been designed based on approximate 4:2 compressors in [23].

The circuit structure can be potentially used for the operands. In [24], the operands are split into two parts: a multiplication part that includes a number of more significant bits and a non-multiplication part made of the remaining lower significant bits. In [25], a leading-1 detector (LOD) has been used to provide accuracy higher than by simply truncating the operands, although it incurs in considerable area and power consumption. A dynamic range unbiased multiplier (DRUM) [26] uses the significant segments of the operands (so the most significant k bits) to perform the multiplication. Multipliers using the logarithmic transformation (which has inherent approximate characteristics) convert multiplication into addition. In the logarithmic conversion, an inherent error is however present, thus, power consumption is often decreased at the cost of precision. The logarithmic multiplier (LM) was first proposed by Mitchell [27]; LMs can be classified into two categories, non-iterative [28] and iterative [29]. In [30], the adder units in the LM architecture are replaced by approximate adders to save power. In [31], the DRUM is used in the Mitchell's algorithm. In this paper, a new scheme to decrease the power dissipation of three units (logarithmic converter, adder, and antilogarithmic converter) in LM are proposed. Compared with the conventional LM, smaller size units are used, so resulting in lower circuit overheads and improvements in figures of merit. Compared with the designs in [31], the compensation including the compensation unit is added to significantly improve the accuracy. Also in the signed designs, the sign converter is performed approximately and efficiently.

This paper is an extended version of our previous work [32], which introduced techniques to design approximate LMs. The approximation is realized by truncating the operation, compensating the adder results and using the inherent approximate characteristics of the LM. Error and circuit-level metrics due to approximate schemes are also measured to evaluate the designs of DR-ALMs. Compared with [32], the main differences are summarized as follows:

- The worst case absolute and relative errors (WCE and WCRE) of the proposed DR-ALMs are formally analyzed.
- Circuit-level designs for the proposed DR-ALMs are further improved to evaluate the power, delay and area.
- Comprehensive comparisons are provided for 8 and 16-bit designs with previous approximate multipliers.
- The scalability of the proposed scheme is examined

in term of increased bit-width.
- Additional comparisons in case studies for three applications (*i.e.*, back-propagation classifier, K-means clustering and handwritten digit recognition) are provided.

The rest of the paper is organized as follows. Section 2 provides a brief review of LM arithmetic and the exact LM architecture. Section 3 presents the designs of the proposed DR-ALMs. Section 4 analyzes the error for the proposed DR-ALMs. Section 5 assesses the accuracy and circuit-level metrics of different approximate designs. The proposed DR-ALMs are also compared with previous approximate designs in this section. Section 6 shows the application of the proposed DR-ALMs to a back-propagation classifier, K-means clustering and handwritten digit recognition. Finally, Section 7 provides the conclusion.

2 CONVENTIONAL LOGARITHMIC MULTIPLIER

In this section, conventional LM is briefly reviewed. A LM converts a binary number format to a logarithmic number format; so, it performs multiplication using addition.

Mitchell's algorithm has been first proposed in [27]. The binary representations of two n-bit input numbers A and B are given by:

$$A = 2^{k_1}(1 + x_1) \tag{1}$$

$$B = 2^{k_2}(1 + x_2) \tag{2}$$

where k_1 and k_2 denote the most significant bit of operand A and B (with value of '1') , respectively. x_1 and x_2 in the range [0,1], denote the fractional parts [29]. The product of operands A and B is given in Eq. (3).

$$P = 2^{k_1+k_2}(1 + x_1)(1 + x_2) \tag{3}$$

The base-2 logarithm of the product is given by:

$$log_2 P = (k_1 + k_2) + log_2(1 + x_1) + log_2(1 + x_2) \tag{4}$$

In Mitchell's algorithm, if $0 \leq x < 1$, $log_2(1 + x)$ is approximately equal to x; so the logarithm of the product and the product are expressed as follows:

$$log_2 P = (k_1 + k_2) + x_1 + x_2 \tag{5}$$

$$P \approx P_{LM} = \begin{cases} 2^{k_1+k_2}(1 + x_1 + x_2), & x_1 + x_2 < 1 \\ 2^{k_1+k_2+1}(x_1 + x_2), & x_1 + x_2 \geq 1 \end{cases} \tag{6}$$

The sum of the characteristic numbers determines the most significant bit of the approximate product; the sum is then shifted left by $k_1 + k_2$ or by $k_1 + k_2 + 1$ positions, depending on $x_1 + x_2$.

The architecture of a non-iterative LM is illustrated in Fig. 1. It consists of LODs, logarithmic converters, adder and an antilogarithmic converter. The LODs find the most significant one bit; the logarithmic converters product the logarithmic number of the binary numbers. Then the logarithmic numbers are added as product. The antilogarithmic converter decodes the result to a binary number.
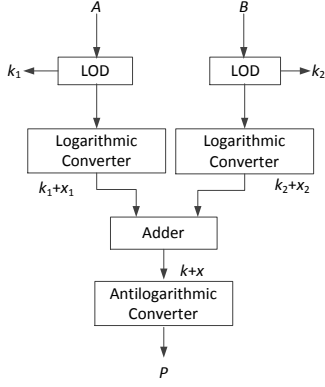
Fig. 1. Non-iterative LM architecture.

3 DESIGN OF DR-ALMS

The proposed DR-ALM is shown in Fig. 2 and Algorithm 1. In a conventional LM, the most significant ones are first located in the operands based on the LODs. Thus, from the most significant one, the operands can be dynamically truncated with different length according to the accuracy requirement. As the product of a LM is always smaller than its exact counterpart, the last bit of the truncated operands is set to logic one to compensate negative errors. In Fig. 2, LODs and dynamic truncation complete this process. In the LOD, the operand is split into two parts with equal length. If the higher bit part is zero, then the most significant bit of k is set to 0, and the lower bit part is continued to be split. Otherwise the most significant bit of k is set to 1, and the higher bit part is continued to be split. The operand is split until it has only one bit. For the truncated fractional operands x_t and their exponents k_t, a smaller bit-width adder is used to generate the logarithmic number of the result after the logarithmic converter. A logic one is added to the addition product to compensate the result. The antilogarithmic converter decodes the result to binary number. Then the result is shifted according to the location of the two leading ones to generate the final approximate result.

Algorithm 1 The Proposed DR-ALM Algorithm

Input: A,B: N-bit operands t: truncated width
Output: AP: the approximate product
  1. LOD:
  $k_1 \leftarrow$ LOD(A), $k_2 \leftarrow$ LOD(B)
  $x_1 \leftarrow A<<(n-k_1-1)$, $x_2 \leftarrow B<<(n-k_2-1)$
  2. Dynamic Truncation:
  $x_{1t} \leftarrow \{x_1[n-2..n-t-1], 1'b1\}$
  $x_{2t} \leftarrow \{x_2[n-2..n-t-1], 1'b1\}$
  3. Logarithmic Converter:
  $op_1 \leftarrow \{k_1, x_{1t}\}$, $op_2 \leftarrow \{k_2, x_{2t}\}$
  4. Adder and Compensation:
  $L \leftarrow op_1 + op_2 + 1$
  5. Antilogarithmic Converter:
  $x_t \leftarrow \{1'b1, L[t-2..0]\}$
  $k \leftarrow L[t-1+log_2(n)..t-1]$
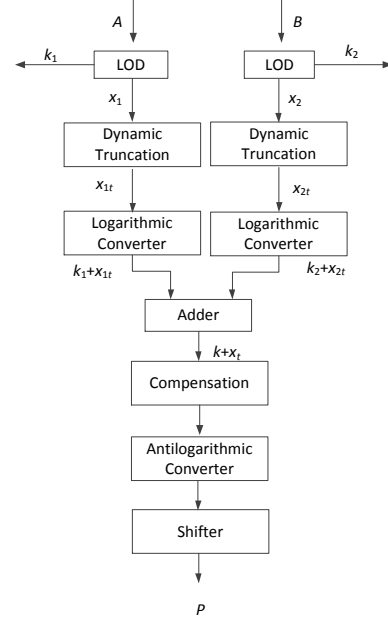 $AP \leftarrow x_t << (k-t+1)$

Fig. 2. The proposed DR-ALM.

The product terms after dynamic truncation and compensation of the fractional portions are as follows:

$$AP_0 = \begin{cases} 2^{k_1+k_2}(1+x_{1t}+x_{2t}+2^{-(t-1)}), \\ \qquad\qquad\qquad\qquad x_{1t}+x_{2t} < 1 \\ 2^{k_1+k_2+1}(x_{1t}+x_{2t}+2^{-(t-1)}), \\ \qquad\qquad\qquad\qquad x_{1t}+x_{2t} \geq 1 \end{cases} \quad (7)$$

where t denotes the truncation width for the operands.

Compared with the conventional LM, dynamic truncation and compensation unit are added to the proposed DR-ALM. Based on the LOD, the input A and B are truncated to x_{1t} and x_{2t} with t bit-width from the leading one, hence the output bit-width of the dynamic truncation unit is decreased from N to t. Then the binary number with k-bit exponent and $(t-1)$-bit mantissa is converted to a $(k+t-1)$-bit logarithmic number $((k+N-1)$-bit in the conventional LM). Next, the adder bit-width is decreased from $k+N-1$ to $k+t-1$. Due to the logic one of the least significant bit in x_{1t} and x_{2t} and the compensation after the adder, the least significant bit and the carry to the neighboring bit of the addition result are both set to logic one, such that the compensation unit complexity is not increased. Finally, the $(k+t)$-bit logarithmic number is converted to a binary number, in which the bit-width is $k+N$ in the antilogarithmic converter of the conventional LM. Based on the dynamic truncation and compensation units, the sizes of the logarithmic converter, adder and antilogarithmic converter are all decreased. Thus, a lower circuit complexity is obtained for the proposed DR-ALMs. The simulation results in Section 5.2 are also applicable.

To assess the most appropriate t, an error analysis of the logarithmic multiplier is carried out in Section 4. A step-by step example of the proposed non-iterative DR-ALM is shown as Example where $t = 6$.
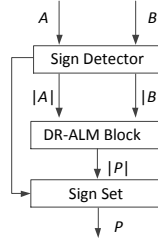
Example

Fig. 3. Signed operation unit.

Let the two binary numbers A and B be
$A = (0101011100001101)_2 = (22285)_{10}$
$B = (0010000101011010)_2 = (8538)_{10}$
The exact product is given by $P = (190269330)_{10}$
$k_1 = (1110)_2; k_2 = (1101)_2$
$x_1 = (0.01011100001101)_2; x_2 = (0.000010101101)_2$
$A_t = (101011)_2; B_t = (100001)_2$
$x_{1t} = (0.01011)_2; x_{2t} = (0.00001)_2$
$x_1 + x_2 = (0.01100111000001)_2 < 1$
$x_{1t} + x_{2t} = (0.01100)_2 < 1$
$P_{LM} = (1011001110000010000000000000)_2$
$RED_{LM} = (P - P_{LM})/P = 1.073\%$
The compensation is $(x_{1t} + x_{2t}) = (0.01101)_2 < 1$
$AP_0 = (1011010000000000000000000000)_2$
$RED_0 = (P - AP_0)/P = 0.801\%$

Thus, a smaller error is introduced when the proposed approximation is used for a DR-ALM.

The LM architectures can be extended for signed operation by using additional hardware to find the absolute values of the operands prior to the unsigned operation and then applying the correct sign to the unsigned result. The sign detector and sign set unit are both performed approximately, which use one's complement and an OR operation with the logic one in the least significant bits, instead of the negation when the inputs are negative. Fig. 3 shows the signed DR-ALM architecture block.

4 ERROR ANALYSIS

For approximate designs, several metrics have been proposed to measure the error; they include the error distance (ED), the mean error distance (MED), the worst case error (WCE), the relative error distance (RED), the worst case relative error (WCRE) [33], the mean RED (MRED) and the normalized MED (NMED) [34].

In the unsigned DR-ALM, the error is given as:

$$
\begin{aligned}
ED_0 &= |P - AP_0| \\
&= \begin{cases} |2^{k_1+k_2}x_1x_2 + 2^{k_1+k_2} \\ (x_1 - x_{1t} + x_2 - x_{2t} - 2^{-(t-1)})|, \\ \qquad\qquad\qquad x_{1t} + x_{2t} < 1 \\ |2^{k_1+k_2}(1-x_1)(1-x_2) + 2^{k_1+k_2+1} \\ (x_1 - x_{1t} + x_2 - x_{2t} - 2^{-(t-1)})|, \\ \qquad\qquad\qquad x_{1t} + x_{2t} \geq 1 \end{cases} \\
&= |ED_{0LM} + ED_{0T}| \qquad (8)
\end{aligned}
$$

where ED_{0LM} and ED_{0T} refer to the errors due to the logarithmic conversion and truncation, respectively.

The least significant bit of the truncation operands determines the sign of the terms $x_1 - x_{1t}$ and $x_2 - x_{2t}$. When

the least significant bit is 0, the ED_{0T} is negative and the $|ED_{0T}|$ has the largest value. The worst case error due to the dynamic truncation and compensation is as follows:

$$WCE_{T1} = |ED_{0Tmax}| = |-2^{2n-1} \times 3 \times 2^{-(t-1)}| \qquad (9)$$

So, the WCE_{T1} decreases rapidly with the ratio of $2^{-(t-1)}$ for a given input bit width n. When the least significant bit is 1, the ED_{0T} is positive and the largest $|ED_{0T}|$ is as follows:

$$
\begin{aligned}
WCE_{T2} &= |ED_{0Tmax}| \\
&= |2^{2n-1}(2^{-(t-2)} - 2^{-(n-1)} - 2^{-(t-1)})| \\
&= |2^{2n-1}(2^{-(t-1)} - 2^{-(n-1)})| \qquad (10)
\end{aligned}
$$

The WCE from the dynamic truncation and compensation is based on the bit width of the input and the truncation length.

The worst case error from the inherent approximate characteristics of the logarithmic multiplier is given by:

$$WCE_{LM} = |ED_{0LMmax}| \qquad (11)$$

When $x_1 = x_2 = x$ or $(1 - x_1) = (1 - x_2) = x$, $k_1 = k_2 = n - 1$ and $x_{1t} + x_{2t} = 1$, the ED_{0LM} has the largest value $2^{2(n-1)} \times x^2$, where n is the bit width of the operands and x is based on the truncation length t.

From Eqs. (9)-(11), when the WCE is mostly due to dynamic truncation and compensation, the WCE for the DR-ALM is given by:

$$WCE_1 = WCE_{T1} - ED_{0LM} \qquad (12)$$

In this case, the operands can be denoted as $[1..1]_{t-1}[0..0]_{n-t+1}$. When the WCE is from the inherent approximate characteristics of the logarithmic multiplier, the WCE for the DR-ALM is given by:

$$WCE_2 = WCE_{LM} + WCE_{T2} \qquad (13)$$

In this case, the operands can be denoted as $10[1..1]_{n-2}$. For example, if the input bit width is 16, $WCE_{LM} = 2^{30} \times (0.01..1)^2$, $WCE_{T1} = 2^{31} \times 3 \times 2^{-(t-1)}$ and $WCE_{T2} = 2^{31} \times 2^{-(t-1)}) - 2^{16}$. So when $t < 6$, $WCE = WCE_1$, and the WCE decrease rapidly. When $t > 6$, $WCE = WCE_2$, and the WCE decreases slowly.

The RED for the proposed unsigned DR-ALM is given by :

$$
\begin{aligned}
RED_0 &= ED_0/P \\
&= \begin{cases} |\frac{x_1x_2}{1+x_1+x_2+x_1x_2} + \frac{x_1-x_{1t}+x_2-x_{2t}-2^{-(t-1)}}{1+x_1+x_2+x_1x_2}|, \\ \qquad\qquad\qquad x_{1t} + x_{2t} < 1 \\ |\frac{(1-x_1)(1-x_2)}{1+x_1+x_2+x_1x_2} + \frac{2(x_1-x_{1t}+x_2-x_{2t}-2^{-(t-1)})}{1+x_1+x_2+x_1x_2}|, \\ \qquad\qquad\qquad x_{1t} + x_{2t} \geq 1 \end{cases} \\
&= |RED_{0LM} + RED_{0T}| \qquad (14)
\end{aligned}
$$

where RED_{0LM} and RED_{0T} refer to the relative errors due to the logarithmic conversion and the truncation, respectively. RED is due to dynamic truncation, compensation and the inherent approximate characteristics of the logarithmic multiplier.

Same as for ED, the RED_{0T} is negative when the least significant bit of the truncation operands is 0; then the worst case relative error incurred from the dynamic truncation and compensation is:

$$WCRE_{T1} = |RED_{0Tmax}| = |3 \times 2^{-(t-1)}| \qquad (15)$$

When the least significant bit is 1, the RED_{0T} is positive and the largest value of the $|RED_{0T}|$ is:

$$
\begin{aligned}
WCRE_{T2} &= |2^{-(t-2)} - 2^{-(n-1)} - 2^{-(t-1)}| \\
&= |2^{-(t-1)} - 2^{-(n-1)}| \quad (16)
\end{aligned}
$$

The largest RED incurs from the inherent approximate characteristics of the logarithmic multiplier so:

$$
WCRE_{LM} = \frac{x^2}{(1+x)^2} \quad (17)
$$

when $x_1 = x_2 = x$ or $(1-x_1) = (1-x_2) = x$.

From Eqs. (15)-(17), when the RED is due to the dynamic truncation and compensation, the WCRE for the DR-ALM is given by:

$$
WCRE_1 = WCRE_{T1} - RED_{0LM} \quad (18)
$$

In this case, $x_1 = x_2 = 0$. So $RED_{0LM} = 0$, $WCRE = WCRE_1 = WCRE_{T1}$. When the WCRE is due to the inherent approximate characteristics of the logarithmic multiplier, then the WCRE for the DR-ALM is given by:

$$
WCRE_2 = WCRE_{LM} + WCRE_{T2} \quad (19)
$$

In this case, the operands can be denoted as $10[1..1]_{n-2}$.

From the above analysis, both WCE and WCRE have two possible values (WCE1, WCE2, and WCRE1, WCRE2) in an iteration based on the least significant bit of the truncated operation. The final WCE and WCRE are the largest among the two possible values. Fig. 4 shows the plots of WCE for the 16-bit and 32-bit proposed DR-ALMs, respectively; in Fig. 4 (a), when $t < 6$, $WCE = WCE_1$, the WCE decreases rapidly. When $t > 6$, $WCE = WCE_2$, and the WCE decreases slowly. Thus, the proposed DR-ALM has an appropriate truncated width t in an iteration which is further validated in Section 5. Fig. 4 (b) shows a similar trend in the 32-bit DR-ALM. Fig. 5 shows the plots of WCRE for the 16-bit and 32-bit proposed DR-ALMs, respectively. Both 16-bit and 32-bit DR-ALMs have now a similar t that allows to trade off accuracy for a lower hardware.

5 SIMULATION RESULTS

In this section, the accuracy and hardware requirements are assessed for the proposed DR-ALMs along with their conventional LM (LM). The proposed unsigned DR-ALMs are compared with ALM-SOA(M=m) [30], Mitch-w [31], DRUM(K=k) [26], UDM [10], mul8x8-xxx [35], mul16x16-xxx [35], and BAM-l [36]. The signed DR-ALMs are compared with Mitch-w-C1 [31], R4ABM($p=i$) [37] and R4ARBM($p=i$) [38]. The acronyms used in this manuscript are shown in Table 1.

5.1 Accuracy

Assume that the input data is uniformly distributed in the full range of the 8-, 16-, and 32-bit fixed-point format for 8-, 16-, and 32-bit multipliers. The functions of the proposed DR-ALMs are analyzed and simulated using C, their MREDs, NMEDs, WCEs and WCREs are reported in Table 2.

From Table 2, the proposed 8-, 16- and 32-bit approximate multipliers with the same truncated bit width (L) of input have a similar error trend. In the proposed 8-, 16-

TABLE 1
Acronyms of Approximate Multiplier Designs

Acronym	Description
LM	Conventional non-iterative logarithmic multi-plier
DR-ALM-L_i	Dynamic range approximate non-iterative logarithmic multiplier with L_i-bit operand
DR-IALM-L_i-L_j	Dynamic range approximate iterative logari-thmic multiplier with L_i-bit operand in Ite-ration 1 and L_j-bit operand in Iteration 2
ALM-SOA (M=m) [30]	The best approximate non-iterative logarit-hmic multiplier with SOA from [30]
DRUM (K=k) [26]	Dynamic range unbiased multiplier with k bits in the operate from the leading one
UDM [10]	The underdesigned multiplier which leve-raged a modified inaccurate 2x2 building block
Mitch-w Mitch-w-C1 [31]	Efficient Mitchell's approximate unsigned and signed Log multiplier with w-bit operand
mul8x8-xx mul16x16-xxx [35]	8-, 16-bit approximate multipliers from the EvoApproxLib [35]
BAM-l [36]	Two broken-array multipliers with the vertical broken length = l
R4ABM($p=i$) [37]	Radix-4 approximate Booth multipliers with an approximate factor of i
R4ARBM($p=i$) [38]	Radix-4 approximate redundant Booth multip-liers with an approximate factor of i

and 32-bit DR-ALMs, the error due to truncation is evident; with an decrease in the truncation length (L), the MRED and NMED both slightly decrease and from $L = 6$ (7 in 8-bit DR-ALM) they rapidly increase; the WCE and WCRE both decrease slightly and from $L = 6$ (7 in 8-bit DR-ALM) they rapidly decrease. For $L = 5$, the MRED and NMED are almost the same as with no truncation, so accuracy is best at $L = 6$ (7 in 8-bit DR-ALM). The WCE and WCRE have the same trend. They all slightly decrease until $L = 6$. These results are consistent with the error analysis in Section 4. In the following simulation, 8- and 16-bit DR-ALMs are mainly analysed.

5.2 Circuit-Level Assessment

All designs are implemented and simulated by the Synopsys Design Compiler using the Nangate 45nm Open Cell Library. In the simulation of each circuit, a supply voltage of 1.25V and room temperature are assumed. The power consumption is simulated by using the Synopsys Power Complier. When synthesizing, the maximum area and delay are set to $0\mu m^2$ and 0.2ns, respectively. The simulation setting for the time scale is $1ns/100ps$ in the Design Compiler script. The value of the power-delay product (PDP) is calculated as a comprehensive metric.

Table 3 reports the delay, area and power consumption of the proposed unsigned DR-ALMs. The first design is the exact 16-bit fixed-point multiplier; the second is the conventional LM; the 3rd to the 10th designs are the proposed DR-ALMs using a different truncation length of the operand bits.

Consider Table 3, a decrease in the truncation length of the operand for an approximate LM leads to a significant reduction in power consumption and PDP. The PDP of DR-ALMs are all lower than the exact multiplier and the conventional LM. The delay is also reduced. As the MRED
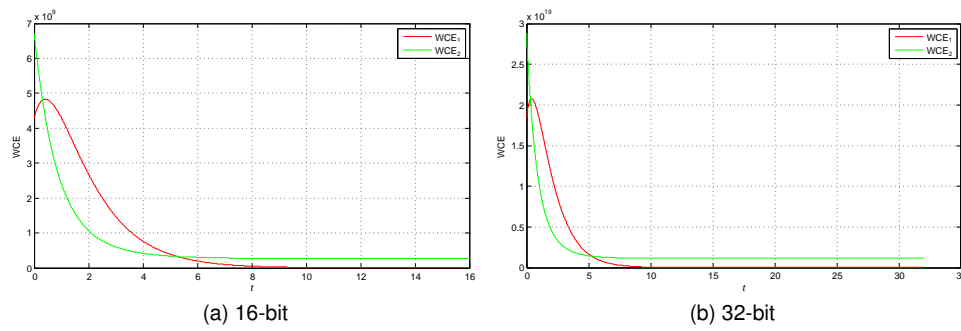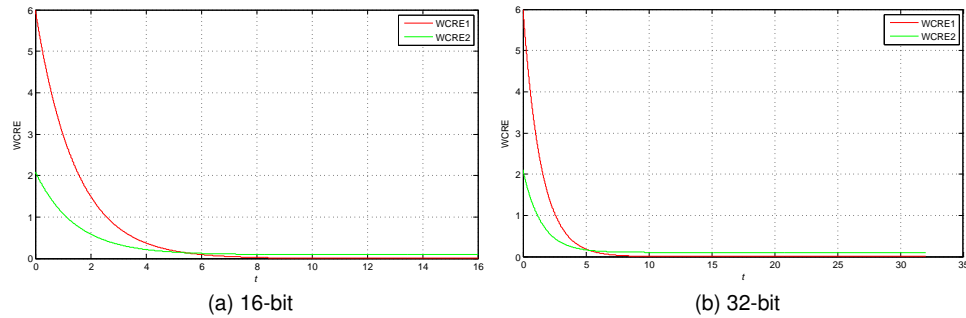
(a) 16-bit (b) 32-bit

Fig. 4. The trend of WCE for DR-ALM.

(a) 16-bit (b) 32-bit

Fig. 5. The trend of WCRE for DR-ALM.

TABLE 2
Errors of DR-ALMs with Different Width L

LMs	L	8-bit				16-bit				32-bit			
		MRED (%)	NMED (10^{-2})	WCE (10^3)	WCRE (%)	MRED (%)	NMED (10^{-2})	WCE (10^6)	WCRE (%)	MRED (%)	NMED (10^{-2})	WCE (10^{16})	WCRE (%)
LM		3.76	0.933	4.096	11.111	3.85	0.926	268.435	11.111	3.88	5.201	7.201	11.111
DR-ALM-12	12	-	-	-	-	3.81	0.917	268.861	11.129	3.85	5.16	7.192	11.125
DR-ALM-10	10	-	-	-	-	3.72	0.895	270.43	11.194	3.76	5.045	7.232	11.178
DR-ALM-8	8	-	-	-	-	3.43	0.823	276.726	11.455	3.47	4.663	7.28	11.438
DR-ALM-7	7	2.74	0.685	3.969	11.111	3.17	0.762	285.114	11.802	3.22	4.316	7.447	11.785
DR-ALM-6	6	2.91	0.699	4.225	11.581	3.03	0.73	301.892	12.496	3.09	4.077	7.997	12.470
DR-ALM-5	5	4.32	0.963	5.888	18.75	3.81	0.923	385.876	18.750	3.93	5.011	9.229	18.680
DR-ALM-4	4	8.87	1.929	11.263	37.494	7.67	1.821	738.198	37.50	8.11	10.215	19.364	37.419
DR-ALM-3	3	20.22	4.287	20.479	74.994	18.41	4.133	1342.18	75.000	20.40	25.688	35.946	74.897

TABLE 3
Circuit Assessment of 16-bit unsigned Multipliers

Multipliers	Power(μW)	Area(μm^2)	Delay(ns)	PDP(fJ)
Exact	641.3	1377.0	1.51	968.4
LM	168.1	781.2	1.00	168.1
DR-ALM-12	158.6	777.0	0.89	141.2
DR-ALM-10	131.2	684.9	0.89	116.8
DR-ALM-8	133.3	679.4	0.84	112.0
DR-ALM-7	111.6	610.7	0.81	90.4
DR-ALM-6	96.5	541.0	0.80	77.2
DR-ALM-5	77.5	452.7	0.80	62.0
DR-ALM-4	61.5	374.3	0.78	48.0
DR-ALM-3	52.3	339.4	0.66	34.5

TABLE 4
Circuit Assessment In LM and DR-ALM-6

		Power(μW)	Area(μm^2)	Delay(ns)
LM	LOD	10.75	72.62	0.24
	Log	54.70	218.92	0.26
	Adder	46.96	164.92	0.36
	Antilog	75.08	271.05	0.30
DR-ALM-6	LOD	10.75	72.62	0.24
	Log	39.68	166.52	0.24
	Adder	15.01	56.13	0.31
	Antilog	23.98	101.61	0.22

of the DR-ALM-6 is the smallest of all DR-ALMs, this design shows the best tradeoff between error and power consumption, achieving up to 54.1% and 92.0% improvements in PDP compared with the conventional LM and the exact multiplier, respectively.

Table 4 shows the hardware consumption of the every units for 16-bit LM and DR-ALM-6. From Table 4, except

the LOD unit, the circuit overhead for the other three units decreases. In the logarithmic converter, the power consumption, area and delay are saved up to 27.5%, 23.9% and 7.7%, respectively. In the adder, the power consumption, area and delay are saved up to 68.0%, 66.0% and 13.9%, respectively, while in the antilogarithmic converter, the power consumption, area and delay are saved up to 68.1%, 62.5% and 26.7%, respectively. With the decrease of bit-width in operands,

these parts incur in lower power, less delay and smaller area than with normal bit-width operands.

5.3 Comparison

In this section, 8-bit and 16-bit approximate multipliers are compared. All previous designs are selected as having similar metrics as the considered approximate multipliers for fair comparison. For the 8-bit unsigned designs, DR-ALM-3, DR-ALM-4, DR-ALM-5, DR-ALM-6 and DR-ALM-7 are selected for comparison with previous approximate multipliers including ALM-SOA(M=3), ALM-SOA(M=4), ALM-SOA(M=6), ALM-SOA(M=7) [30], Mitch-4, Mitch-5, Mitch-6, Mitch-7 [31], DRUM(K=4), DRUM(K=5), DRUM(K=6) [26], mul8x8-444, mul8x8-188, mul8x8-198 [35], and BAM-8, BAM-10 [36]. For the 8-bit signed designs, signed DR-ALM-3, DR-ALM-4, DR-ALM-5, DR-ALM-6 and DR-ALM-7 are compared with Mitch-4-C1, Mitch-5-C1, Mitch-6-C1, Mitch-7-C1 [31], R4ABM2(p=6), R4ABM2(p=8) [37], and R4ARBM2(p=6), R4ARBM2(p=8) [38].

For the 16-bit unsigned designs, DR-ALM-3, DR-ALM-4, DR-ALM-5, DR-ALM-6 and DR-ALM-7 are selected for comparison with previous approximate multipliers including ALM-SOA(M=11), ALM-SOA(M=13), ALM-SOA(M=15) [30], Mitch-6, Mitch-8 [31], DRUM(K=4), DRUM(K=5), DRUM(K=6) [26], UDM [10], mul16x16-023, mul16x16-024, mul16x16-043 [35], and BAM-18, BAM-22 [36]. For the 16-bit signed designs, signed DR-ALM-3, DR-ALM-4, DR-ALM-5, DR-ALM-6 and DR-ALM-7 are compared with Mitch-6-C1, Mitch-8-C1 [31], R4ABM2(p=14), R4ABM2(p=18) [37], and R4ARBM2(p=14), R4ARBM2(p=18) [38].

The power consumption, area, delay, PDP, NMED, M-RED, WCE and WCRE of 8-bit multipliers are reported in Table 5. Compared with the unsigned exact multiplier and the LM counterpart, DR-ALM-3 improves the PDP up to 84.40% and 75.63%, respectively. ALM-SOA(M=3) [30] has a slightly lower PDP but larger MRED than DR-ALM-6. ALM-SOA(M=4) [30] has a marginally higher PDP but smaller M-RED than DR-ALM-5. ALM-SOA(M=7) [30] has the smallest power, area, delay and PDP, but significantly large values of NMED, MRED and WCE. Mitch-4, Mitch-5, Mitch-6, Mitch-7 [31] have a higher PDP and a larger MRED than DR-ALM-4, DR-ALM-5, DR-ALM-6, DR-ALM-7, respectively. All DRUMs have a high PDP. DRUM(K=4) [26] has a slightly lower NMED, MRED, WCE and WCRE, but a significantly higher PDP than DR-ALM-4. DRUM(K=5) [26] has a similar NMED, MRED, WCE and WCRE, but significantly higher PDP than DR-ALM-6. DRUM(K=6) [26] has the smallest MRED and WCRE, but a high PDP, which is larger than the exact multiplier. Mul8x8-444 has the smallest NMED and WCE. Mul8x8-188 has a marginally higher MRED, but a large PDP, compared with DR-ALM-5. Mul8x8-198 [35] has a similar PDP and MRED, but significantly larger WCRE than DR-ALM-4. BAM-10 has a similar PDP but six times as large as DR-ALM-5. When the truncation bits decrease to 8 for BAM, the PDP is increased to the value of DR-ALM-7, but the MRED is still high. The comparison of PDPs and MREDs for the unsigned 8-bit multipliers is shown in Fig.6. The DR-ALMs are shown with an asterisk in different colors, respectively. DR-ALM-5, DR-ALM-6, ALM-SOA(M=3), and ALM-SOA(M=4) [30] have good performance. Both ALM-
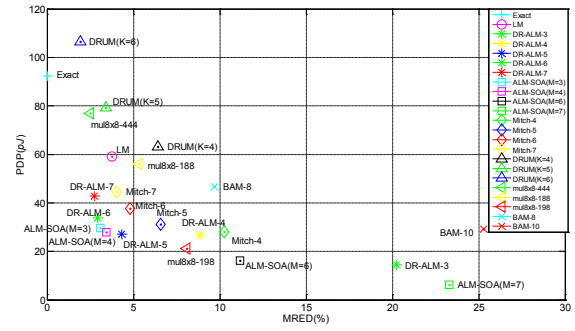

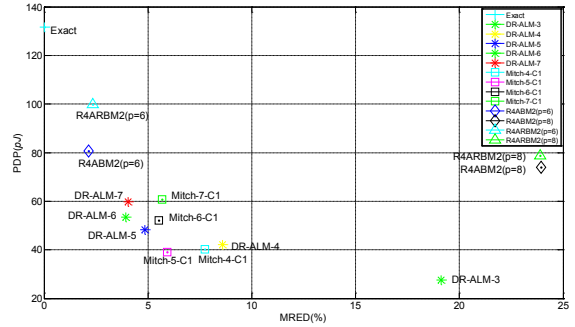Fig. 6. MREDs and PDPs of unsigned 8-bit multipliers.

Fig. 7. MREDs and PDPs of signed 8-bit multipliers.

SOA(M=3) [30] and ALM-SOA(M=4) [30] have a slightly higher PDP but a lower NMED than DR-ALM-5.

The comparison of PDPs and MREDs for the signed 8-bit multipliers is shown in Fig.7. The approximate Booth multipliers (R4ABM2 [37] and R4ARBM2 [38]) have a wider range of accuracy with the approximate factor p, but higher PDPs than the DR-ALMs. Mitch-5-C1, Mitch-6-C1, Mitch-7-C1 [31] have a similar MRED, whose value is between DR-ALM-4 and DR-ALM-5.

The power consumption, area, delay, PDP, NMED, M-RED, WCE and WCRE of 16-bit multipliers are also provided in Table 6. Compared with the unsigned exact multiplier and the LM counterpart, DR-ALM-3 improves the PDP up to 96.44% and 79.48%, respectively. Among all unsigned approximate LMs, ALM-SOA(M=15) [30] has the smallest power consumption, area, delay and PDP, but a significantly larger NMED, MRED and WCE than the DR-ALMs. For ALM-SOAs, when M is decreased, the error drops rapidly. ALM-SOA(M=11) [30] has a lower PDP but larger MRED than DR-ALM-6. ALM-SOA(M=13) [30] has a lower MRED but higher PDP than DR-ALM-4. ALM-SOA(M=15) [30] has a higher MRED but smaller PDP than DR-ALM-3. DR-ALM-6 has the smallest NMED and MRED among all approximate LMs. DR-ALM-7 has the smallest WCE and WCRE except the LM. Mitch-6 [31] has a similar PDP but larger MRED than DR-ALM-6. And Mitch-8 [31] has a similar PDP but larger MRED than DR-ALM-7. Among other approximate unsigned multipliers, DRUM(K=6) [26] is more accurate than all DR-ALMs but at a high PDP. All DRUMs have a higher PDP than DR-ALMs. UDM has the largest PDP among all multipliers, but a lower area than DRUM(K=6). Mul16x16-023 and mul16x16-024 [35] have a similar MRED

TABLE 5
Performance Comparison of 8-bit Multipliers

Unsigned Multipliers	Power (μW)	Area (μm^2)	Delay (ns)	PDP (fJ)	NMED (10^{-2})	MRED (%)	WCE (10^3)	WCRE (%)
Exact	115.4	350.9	0.80	92.3	0	0	0	0
LM	70.1	331.4	0.68	59.1	0.93	3.76	4.91	11.11
DR-ALM-3	27.6	178.0	0.52	14.4	4.29	20.22	20.48	74.99
DR-ALM-4	45.4	257.8	0.59	26.8	1.93	8.87	11.26	37.49
DR-ALM-5	43.2	234.9	0.63	27.2	0.96	4.32	5.89	18.75
DR-ALM-6	51.1	262.3	0.66	33.8	0.70	2.91	4.23	11.58
DR-ALM-7	63.0	309.1	0.68	42.8	0.69	2.74	3.97	11.11
ALM-SOA(M=3) [30]	45.0	224.8	0.66	29.7	0.78	3.08	4.16	12.19
ALM-SOA(M=4) [30]	46.4	239.1	0.60	27.8	0.85	3.43	5.37	13.96
ALM-SOA(M=6) [30]	31.8	185.4	0.51	16.2	2.61	11.16	15.10	24.61
ALM-SOA(M=7) [30]	16.1	108.8	**0.39**	6.3	5.46	23.28	28.42	99.22
Mitch-4 [31]	45.4	269.7	0.62	28.1	2.78	10.26	10.08	23.53
Mitch-5 [31]	48.9	278.0	0.64	31.2	1.78	6.57	6.83	17.25
Mitch-6 [31]	54.7	291.8	0.69	37.7	1.27	4.79	5.26	13.83
Mitch-7 [31]	65.4	326.9	0.68	44.5	1.03	4.02	4.48	12.03
DRUM(K=4) [26]	70.2	368.7	0.90	63.2	1.43	6.41	7.43	26.56
DRUM(K=5) [26]	86.2	474.3	0.92	79.3	0.73	3.39	3.90	12.89
DRUM(K=6) [26]	108.6	571.1	0.98	106.4	0.39	1.91	2.00	6.35
mul8x8-444 [35]	62.0	319.2	1.24	76.9	0.13	2.49	0.54	200.00
mul8x8-188 [35]	44.8	245.3	1.25	56.0	0.36	5.32	1.30	433.33
mul8x8-198 [35]	40.0	217.3	0.53	21.2	0.89	8.07	2.88	312.50
BAM-8 [36]	70.8	433.0	0.66	46.7	0.69	9.70	1.79	100.00
BAM-10 [36]	50.6	349.3	0.58	29.3	2.56	25.27	6.66	100.00
Signed Multipliers	**Power** (μW)	**Area** (μm^2)	**Delay** (ns)	**PDP** (fJ)	**NMED** (10^{-2})	**MRED** (%)	**WCE** (10^3)	**WCRE** (%)
Exact booth	188.0	788.4	0.70	131.6	0	0	0	0
signed DR-ALM-3	45.3	230.1	0.61	27.6	2.07	19.11	5.12	75.00
signed DR-ALM-4	62.9	295.5	0.67	42.1	0.92	8.59	2.82	75.00
signed DR-ALM-5	67.0	320.5	0.72	48.2	0.48	4.84	1.47	75.00
signed DR-ALM-6	74.0	332.5	0.74	53.5	0.38	3.94	1.15	75.00
signed DR-ALM-7	74.8	333.3	0.80	59.8	0.39	4.04	1.09	75.00
Mitch-4-C1 [31]	59.2	296.9	0.68	40.3	0.87	7.74	2.56	100.00
Mitch-5-C1 [31]	58.2	308.6	0.67	39.0	0.61	5.92	1.79	100.00
Mitch-6-C1 [31]	60.0	307.3	0.70	42.0	0.53	5.53	1.41	100.00
Mitch-7-C1 [31]	76.9	354.8	0.79	60.8	0.51	5.68	1.22	100.00
R4ABM2(p=6) [37]	139.1	575.0	0.58	80.7	0.10	2.13	0.17	1.06×10^4
R4ABM2(p=8) [37]	127.4	538.6	0.58	73.9	0.41	23.93	0.94	6.82×10^4
R4ARBM2(p=6) [38]	153.7	642.1	0.65	99.9	0.11	2.33	0.16	7.8×10^3
R4ARBM2(p=8) [38]	131.4	563.4	0.60	78.8	0.49	23.90	0.65	1.567×10^6

but a significantly higher PDP than DR-ALM-5 and DR-ALM-4, respectively. Mul16x16-043 [35] has a similar PDP as DR-ALM-6, but the largest MRED among all multipliers. 16-bit BAMs have high PDPs. BAM-18 [36] has a high PDP but the smallest MRED. BAM-22 [36] has a slightly larger MRED but significantly higher PDP than the DR-ALM-5.

Fig. 8 shows the comparison for both MRED and PDP with the 16-bit unsigned proposed DR-ALMs and the previous approximate multipliers with a low PDP and high accuracy. BAM-18 [36] has the smallest MRED but the highest PDP than other multipliers, while ALM-SOA(M=15) [30] has the largest MRED but the lowest PDP. DRUM(K=5) [26] has a similar MRED but higher PDP than DR-ALM-7. DR-ALM-7 has a smaller MRED with similar PDP than Mitch-8 [31]. ALM-SOA(M=11) [30] has a slightly larger MRED and lower PDP than DR-ALM-6. Mitch-6 [31] has a similar PDP but larger MRED than ALM-SOA(M=11) [30] and DR-ALM-6. DR-ALM-5 has a slightly larger MRED but lower PDP than ALM-SOA(M=11) [30]. DR-ALM-4 and DR-ALM-3 have lower PDPs but larger MREDs than other multipliers except ALM-SOA(M=15) [30]. With the decrease of the truncated bit width of the input, the PDP of DR-ALMs decreases rapidly until $L = 6$. When $L \geq 6$, the PDP decreases slightly.

The comparison of PDPs and MREDs for signed 16-bit multipliers is shown in Fig.9. Approximate Booth multi-

pliers have in general a higher PDP compared with DR-ALMs. When the approximate factor p is increased over 14, the accuracy decrease rapidly. Mitch-6-C1 [31] has a similar MRED and PDP as DR-ALM-5. Mitch-8-C1 [31] has a higher PDP and larger MRED than DR-ALM-5.

The simulation results show that the proposed DR-ALMs has excellent performance compared with previous multipliers based on operand approximation, artificial intelligence methods and approximate partial generation. Compared with ALM-SOAs, when the bit-width is increased, the proposed DR-ALMs have better performance. In applications that allow small power consumption but large errors, the proposed DR-ALM is the best choice.

5.4 Evaluation of DR-ALM Scaling

In this subsection, the scalability of the proposed scheme is examined in term of increased bit-width of the proposed approximate multipliers (DR-ALM-5 and DR-ALM-6).

Table 7 shows the comparison of energy and error of the exact and proposed multipliers for scaled multiplier sizes of 8, 16, 32 bit. From Table 7, DR-ALM-5 with a scaled bit width decreases the PDP up to 73.53%, 93.60% and 99.00% compared to 8-, 16-, 32-bit exact multipliers, respectively. DR-ALM-6 with a scaled bit width decreases the PDP up to 63.44%, 92.03% and 98.84% compared to 8-, 16-, 32-bit exact multipliers, respectively. Compared with the conventional

TABLE 6
Performance Comparison of 16-bit Multipliers

Unsigned Multipliers	Power (μW)	Area (μm^2)	Delay (ns)	PDP (fJ)	NMED (10^{-2})	MRED (%)	WCE (10^6)	WCRE (%)
Exact	641.3	1377	1.51	968.4	0	0	0	0
LM	168.1	781.2	1.00	168.1	0.926	3.85	268.44	11.111
DR-ALM-3	52.3	339.4	0.66	34.5	4.133	18.41	1342.18	75.000
DR-ALM-4	61.5	374.3	0.78	48.0	1.821	7.67	738.20	37.500
DR-ALM-5	77.5	452.7	0.80	62.0	0.923	3.81	385.88	18.750
DR-ALM-6	96.5	541.0	0.80	77.2	**0.730**	**3.03**	301.89	12.496
DR-ALM-7	111.6	610.7	0.81	90.4	0.762	3.17	**285.11**	**11.802**
ALM-SOA(M=11) [30]	89.0	515.2	0.81	72.1	0.806	3.33	318.65	12.794
ALM-SOA(M=13) [30]	69.4	427.5	0.75	52.1	1.319	5.36	520.03	17.836
ALM-SOA(M=15) [30]	**32.1**	**231.4**	**0.52**	**16.7**	5.460	22.43	1878.98	49.706
Mitch-6 [31]	90.0	580.4	0.79	71.1	1.442	5.90	370.05	14.699
Mitch-8 [31]	105.4	642.7	0.87	91.7	1.055	4.36	293.57	12.026
DRUM(K=4) [26]	86.4	592.1	1.08	93.3	1.410	5.89	519.96	26.563
DRUM(K=5) [26]	102.7	721.9	1.13	116.1	0.705	2.94	264.11	12.891
DRUM(K=6) [26]	124.7	858.0	1.19	148.4	0.353	1.47	133.04	6.348
UDM [10]	705.3	825.7	1.99	1420.4	1.300	3.33	401.23	13.490
mul16x16-023 [35]	562.7	1052.3	0.83	467.0	0.014	3.76	2.14	3.311×10^8
mul16x16-024 [35]	467.7	864	0.82	383.5	0.024	8.49	4.227	3.179×10^8
mul16x16-043 [35]	122.3	326	**0.62**	75.8	0.710	199.53	85.15	6.724×10^9
BAM-18 [36]	240.8	1320.4	0.98	236.0	0.022	0.63	3.801	100
BAM-22 [36]	140.5	893.0	0.84	118.0	0.268	4.75	46.01	100
Signed Multipliers	**Power (μW)**	**Area (μm^2)**	**Delay (ns)**	**PDP (fJ)**	**NMED (10^{-2})**	**MRED (%)**	**WCE (10^6)**	**WCRE (%)**
Exact booth	634.0	2647	1.02	646.7	0	0	0	0
signed DR-ALM-3	99.5	461.8	0.85	84.6	2.220	14.76	234.88	75.00
signed DR-ALM-4	117.1	540.5	0.95	111.2	1.243	7.63	167.77	50.00
signed DR-ALM-5	128.2	624.8	0.86	110.3	0.527	3.57	83.86	50.00
signed DR-ALM-6	149.0	656.0	1.00	149.0	0.272	2.44	45.07	50.00
signed DR-ALM-7	176.9	758.9	1.03	182.2	0.256	2.51	43.23	100.00
Mitch-6-C1 [31]	116.2	623.5	0.95	110.4	0.41	3.46	61.84	100.00
Mitch-8-C1 [31]	136.3	708.9	0.98	133.6	0.46	3.75	48.49	100.00
R4ABM2(p=14) [37]	479.7	2004	0.92	441.3	0.62×10^{-3}	3.30	0.13	2.52×10^6
R4ABM2(p=18) [37]	437.4	1788	0.89	389.3	1.12×10^{-2}	56.71	1.95	9.92×10^7
R4ARBM2(p=14) [38]	580.1	2294.2	0.89	526.3	7.60×10^{-4}	2.61	0.10	1.279×10^6
R4ARBM2(p=18) [38]	466.9	1864.1	0.82	382.9	1.04×10^{-2}	44.08	1.31	2.585×10^7

TABLE 7
Comparison of energy and error for exact and proposed unsigned multipliers at scaled multiplier bit-width

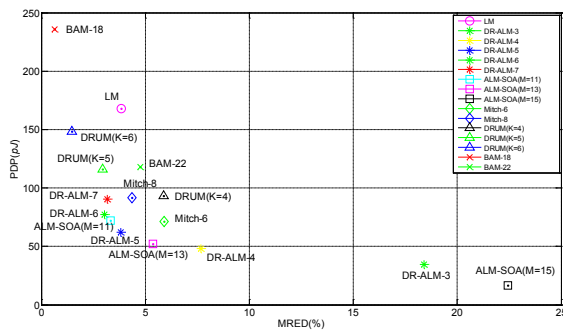| Multipliers	8-bit			16-bit			32-bit		
	PDP (fJ)	PDP Improvement (vs Exact)	MRED (%)	PDP (fJ)	PDP Improvement (vs Exact)	MRED (%)	PDP (fJ)	PDP Improvement (vs Exact)	MRED (%)
Exact	92.3	-	-	968.4	-	-	10737.3	-	-
LM	47.6	48.40%	3.76	168.1	82.64%	3.85	448.1	95.83%	3.88
DR-ALM-5	27.2	70.53%	4.32	62.0	93.60%	3.81	106.9	99.00%	3.93
DR-ALM-6	33.8	63.44%	2.91	77.2	92.03%	3.03	124.5	98.84%	3.09

Fig. 8. MREDs and PDPs of unsigned 16-bit multipliers with low PDPs and high accuracys.
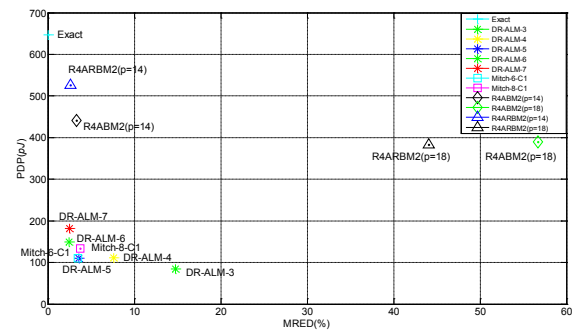

Fig. 9. MREDs and PDPs of signed 16-bit multipliers.

LM, the 8-bit DR-ALM-5 saves up to 42.86% of PDP with an increase of 14.89% in MRED. Compared with the 16-bit conventional logarithmic counterparts, the 16-bit DR-ALM-5 saves up to 63.12% of PDP with a similar MRED. For 32-bit multipliers, DR-ALM-5 saves up to 76.14% of PDP with an increase of 1.29% in MRED compared with the conventional logarithmic counterpart. For DR-ALM-6, the 8-bit design saves up to 28.99% of PDP with a decrease of 22.61% in MRED, compared with the conventional LM.

The 16-bit DR-ALM-6 saves up to 54.07% of PDP with a decrease of 21.30% in MRED compared with LM. For 32-bit multipliers, DR-ALM-6 saves up to 72.22% of PDP with a decrease of 20.36% in MRED compared with the conventional logarithmic counterpart.

The scaling behavior confirms that the proposed approximate multipliers consume less PDP and have a similar or even smaller MRED compared with the conventional counterparts; the advantage is more pronounced when the number of bits is increased.

6 APPLICATIONS

In this section, the proposed design schemes (DR-ALM-3, DR-ALM-4, DR-ALM-5, DR-ALM-6 and DR-ALM-7), the previous state-of-art designs (ALM-SOA(M=3), ALM-SOA(M=4) and Mitch-6 for an 8-bit multiplier , ALM-SOA(M=11) and Mitch-6 for a 16-bit multiplier) are applied to three applications, namely back-propagation (BP) classifier, K-means clustering and handwritten digit recognition; also, several University of California Irvine (UCI) benchmark datasets [39] are selected to test the applications using the proposed DR-ALMs. The approximate designs are modeled in C and simulated by MATLAB.

6.1 BP Classifier

To evaluate the proposed DR-ALMs, a classifier designed by a feedforward neural network, multilayer perceptron (MLP) which is a broadly used artificial neural network, is simulated. The neural network [40], [41] with acyclic connections from the input layer to the output layer is often utilized due to its simplicity compared to a recurrent neural network. In each layer, a neuron has multiple input connections (weights) from some neurons in the previous layer. The output of the network is determined by feed-forward from the input layer to the output layer with all weights. To train the network, each weight is updated to reduce the error in the output layer, usually with a given desired output. To update the weight, the back-propagation (BP) algorithm is used in this network.

The datasets used in this subsection include Iris, Wine and Breast. The two selection criteria are application diversity and having less than 100 attributes. For each dataset, the number of inputs and outputs are selected based on the typical number of attributes and classes in the examples of the UCI repository. The data characteristics are summarized in Table 8. The 8-bit conventional logarithmic multiplier, the proposed DR-ALMs and the previous multipliers (ALM-SOA(M=3), ALM-SOA(M=4) [30] and Mitch-6 [31]) are used to replace the exact multiplier in inference for multiplying the weights and the neurons. The accuracy is defined as the percentage of matches between the assigned class and the predicted output class.

Table 9 summarizes the classification accuracy using exact and approximate logarithmic multipliers in the network after the testing process. From Table 9, even the design that uses the exact multiplier cannot always classify all data correctly. Consider the Iris application first; DR-ALM-4, DR-ALM-5, DR-ALM-6, DR-ALM-7 and Mitch-6 designs have an accuracy of 96.97%. The accuracy of LM and ALM-SOA(M=3) is smaller than 90%. In the Wine application,

TABLE 8
Data Characteristics

Application	Description	#Input, #Output	η, #Hidden
Iris	Plants classification	4,3	0.2,6
Wine	Wine origin based on chemicals	13,3	0.2,4
Breast	Breast cancer diagnostic	30,2	0.1,14

TABLE 9
Classification Accuracy Using Exact, Conventional, and Proposed DR-ALM Design in the BP Network

Datasets	Multipliers	Accuracy
Iris	Exact	**96.97%**
	LM	85.82%
	DR-ALM-3	90.91%
	DR-ALM-4	**96.97%**
	DR-ALM-5	**96.97%**
	DR-ALM-6	**96.97%**
	DR-ALM-7	**96.97%**
	ALM-SOA(M=3) [30]	83.76%
	ALM-SOA(M=4) [30]	91.88%
	Mitch-6 [31]	**96.97%**
Wine	Exact	**97.10%**
	LM	86.35%
	DR-ALM-3	92.75%
	DR-ALM-4	94.20%
	DR-ALM-5	94.20%
	DR-ALM-6	94.20%
	DR-ALM-7	94.20%
	ALM-SOA(M=3) [30]	93.59%
	ALM-SOA(M=4) [30]	93.59%
	Mitch-6 [31]	94.20%
Breast	Exact	97.66%
	LM	92.39%
	DR-ALM-3	95.84%
	DR-ALM-4	96.75%
	DR-ALM-5	97.66%
	DR-ALM-6	**98.04%**
	DR-ALM-7	**98.04%**
	ALM-SOA(M=3) [30]	89.48%
	ALM-SOA(M=4) [30]	88.95%
	Mitch-6 [31]	96.46%

the accuracy of the exact multiplier is the largest. DR-ALM-4, DR-ALM-5, DR-ALM-6, DR-ALM-7 and Mitch-6 are the second best. In the Breast application, the accuracy of DR-ALM-6 and DR-ALM-7 are higher than the exact scheme. For the three datasets, DR-ALM-6 and DR-ALM-7 have a similar accuracy but a lower PDP than the exact design

6.2 K-Means Clustering

K-means clustering is a method for cluster analysis in data mining. It partitions n observations into K clusters with the nearest mean [42]. The proposed unsigned 16-bit LM, DR-ALMs, ALM-SOA(M=11) [30] and Mitch-6 [31] replacing the exact multiplier are applied to calculate the squared deviation between points belonging to different clusters. The F-measure value [43] is used as the metric to evaluate the clustering results. It considers both the precision and the recall of the test. So, the F-measure score can be interpreted as a weighted average of the precision and recall. The F-measure value has a range of 0 to 1; a higher value of F-measure value means that the design has a better performance. Each F-measure value is the average of 50 experiments for each data set.

TABLE 10
F-Measure Results and Hardware Requirements of Clustering by
K-Means (NS: No. of Samples, NC: No. of Clusters)

Datasets	NS	NC	Multipliers	F-measure
Glass	214	9	Exact	0.5002
			LM	0.4495
			DR-ALM-3	0.5043
			DR-ALM-4	0.5317
			DR-ALM-5	0.5146
			DR-ALM-6	0.5317
			DR-ALM-7	0.5317
			ALM-SOA(M=11) [30]	0.4956
			Mitch-6 [31]	0.5146
Hayes-roth	132	4	Exact	0.4358
			LM	0.5201
			DR-ALM-3	0.5201
			DR-ALM-4	0.5201
			DR-ALM-5	0.5201
			DR-ALM-6	0.5201
			DR-ALM-7	0.5201
			ALM-SOA(M=11) [30]	0.5201
			Mitch-6 [31]	0.5201
Balance-scale	625	9	Exact	0.4750
			LM	0.4830
			DR-ALM-3	0.4806
			DR-ALM-4	0.4851
			DR-ALM-5	0.4820
			DR-ALM-6	0.4801
			DR-ALM-7	0.4801
			ALM-SOA(M=11) [30]	0.4870
			Mitch-6 [31]	0.4820
Customers	440	9	Exact	0.5740
			LM	0.4386
			DR-ALM-3	0.4932
			DR-ALM-4	0.4745
			DR-ALM-5	0.4641
			DR-ALM-6	0.4581
			DR-ALM-7	0.4474
			ALM-SOA(M=11) [30]	0.4413
			Mitch-6 [31]	0.4476

TABLE 11
Recognition Rate (%)

Multipliers	F6	F6/C5	F6/C5/C3	F6/C5/C3/C1
LM	98.39	98.36	98.39	98.42
DR-ALM-3	98.33	98.15	98.15	97.72
DR-ALM-4	98.40	98.35	98.34	98.36
DR-ALM-5	**98.44**	98.35	98.40	**98.45**
DR-ALM-6	98.42	98.37	**98.44**	98.40
DR-ALM-7	98.41	98.35	98.39	98.42
ALM-SOA(M=3) [30]	98.41	98.37	98.38	98.39
ALM-SOA(M=4) [30]	98.41	98.35	98.42	98.42
Mitch-6 [31]	98.39	98.36	98.41	98.36
signed DR-ALM-5	98.38	98.28	98.39	98.20
signed DR-ALM-6	98.41	98.35	98.39	98.35
Mitch-6-C1 [31]	98.41	**98.39**	98.30	97.62

The F-measure results are listed in Table 10. For the datasets of Glass, Hayes-roth and Balance-scale, the proposed DR-ALMs produce better results than the exact multiplier. For the datasets of Glass, the proposed DR-ALM-4, DR-ALM-6 and DR-ALM-7 have the best F-measure results. For the datasets of Hayes-roth, all approximate designs have the same F-measure result. For the datasets of Balance-scale, ALM-SOA(M=11) [30] is the best design. For the datasets of customers, the propose DR-ALMs generate better results than LM, although the results are worse than the exact multiplier. For all datasets, the clustering results are similar in general to those processed with the exact multiplier. Note that the K-means algorithm is sensitive to initial centroids. Hence, approximate multipliers can even achieve better accuracy than the accurate algorithm because the acceptable error introduced by approximate computing avoids overfitting the initial centroids.

6.3 Handwritten Digit Recognition

The handwritten digit recognition utilizes a deep neural network. A LeNet-5 [44] is implemented and analysed. It consists of two convolutional layers, each of which is followed by a max pooling layer, two fully-connected layers, and a softmax layer. The activation function is tanh. The training set and the test set include 60,000 and 10,000 images from the MNIST database [45], respectively. Double-precision floating-point operations are used for training. Note that only the inference phase is experienced to evaluate the approximate circuits. In addition, only multiplication operations in convolutional or fully-connected layers are performed using approximate 8-bit fixed-point multipliers. To compared the unsigned approximate multipliers fairly, the sign converter is computed exactly for LM, DR-ALMs, ALM-SOA(M=3), ALM-SOA(M=4) [30], and Mitch-6 [31]. And to valid the proposed signed design, the signed DR-ALM-5, DR-ALM-6 and Mitch-6-C1 [31] are also performed. The recognition rate is used as a metric to evaluate accuracy.

Table 11 shows the recognition rate of the handwritten digit recognition application. F6 and C5 are the fully-connected layers for Layer 6 and 5, respectively. C3 and C1 are the convolutional layers for Layer 3 and 1, respectively. When the multiplication operations in the F6 layer are performed approximately, the recognition rate is given as in the 2nd row of Table 11. The other rows show the corresponding recognition rates with the approximate multipliers for the layers. Note that when recognition is based on the exact floating-point multiplier, the recognition rate is 98.38%. As shown in Table 11, when the floating-point multiplier is replaced by DR-ALM-5 and DR-ALM-6, the recognition rates are similar or ever slightly higher (98.45%). For DR-ALM-3 and Mitch-6-C1, the recognition rate is lower than 98% when all convolutional and fully-connected layers are performed approximately. In general for the proposed signed designs (signed DR-ALM-5 and DR-ALM-6), the approximate sign converter does not affect the recognition. The recognition rate for the signed approximate multipliers remains high.

7 CONCLUSION

In this paper, unsigned and signed DR-ALM designs have been proposed; the worst case absolute and relative errors (WCE and WCRE) of the proposed DR-ALMs are formally analyzed. The accuracy and designs of DR-ALMs have been assessed by simulation. Simulation results have shown that the approximate LM design using 5, and 6-bit truncated operands (DR-ALM-5, DR-ALM-6) have efficient performance with moderate values of MRED and PDP compared with the conventional counterparts and the previous approximate multipliers. The PDP-MREDs of the 8 bit DR-ALM-5 and DR-ALM-6 decrease up to 47.1% and 55.7%, respectively, compared with conventional designs. For 16 bit designs, the PDP-MREDs decrease up to 63.5% and 63.7% for DR-ALM-5 and DR-ALM-6 of PDP-MREDs. All proposed designs have also been applied to machine learning applications to confirm the validity of the proposed

DR-ALM designs. In the BP classifier, DR-ALM-5, DR-ALM-6 and DR-ALM-7 have an accuracy like the exact multiplier. For K-means clustering, the F-measure values for all DR-ALMs are like or even higher than the exact and the conventional counterparts. In the handwritten digit recognition application, the recognition rates for unsigned and signed DR-ALM-5 and DR-ALM-6 remain high.

ACKNOWLEDGMENTS

REFERENCES

[1] W. Liu, F. Lombardi, and M. Shulte, "A retrospective and prospective view of approximate computing," *Proc. of IEEE*, vol. 108, no. 3, pp. 394-399, 2020.

[2] G. Jaberipur , B. Parhami , and D. Abedi, "Adapting computer arithmetic structures to sustainable supercomputing in low-power, majority-logic nanotechnologies," *IEEE Trans. Sustainable Computing*, vol. 3, no. 4, pp. 262-273, 2018.

[3] S. Venkataramani, S.T. Chakradhar, K. Roy, and A.Raghunathan, "Computing approximately, and efficiently," in *Proc. Design, Automation & Test in Europe Conference & Exhibition*, 2015, pp. 748-751.

[4] H. Jiang, C. Liu, L. Liu, F. Lombardi, and J. Han, "A review, classification, and comparative evaluation of approximate arithmetic circuits," in *ACM J. Emerg. Technol. Comput. Syst.*, vol. 13, no. 4, Aug. 2017, Art. no. 60.

[5] D. Esposito, G. Castellano, D. D. Caro, E. Napoli, N. Petra, A. G. M. Strollo, "Approximate adder with output correction for error tolerant applications and gaussian distributed inputs," in *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, 2016, pp. 1970-1973.

[6] A. B. Kahng, and S. Kang, "Accuracy-configurable adder for approximate arithmetic designs," in *Proc. of DAC Design Automation Conference*, 2012, pp. 820-825.

[7] Y. Kim, Y. Zhang, and P. Li, "Energy efficient approximate arithmetic for error resilient neuromorphic computing," *IEEE Trans. Very Large Scale Integr. (VLSI) Systems*, vol. 23, no. 11, pp. 2733-2737, 2015.

[8] Z. Yang, A. Jain, J. Liang, J. Han, and F. Lombardi, "Approximate XOR/XNOR-based adders for inexact computing," in *Proc. of IEEE International Conference on Nanotechnology*, 2013, pp. 690-693.

[9] W. Liu, L. Chen, C. Wang, M. ONeill, and F. Lombardi, "Design and analysis of inexact floating-point adders," *IEEE Trans. Comput.*, vol. 65, no. 1, pp. 308-314, Jan. 2016.

[10] P. Kulkarni, P. Gupta, and M. Ercegovac, "Trading accuracy for power with an underdesigned multiplier architecture," in *Proc. 24th Int. Conf. Very Large Scale Integr. (VLSI) Design*, 2011, pp. 346-351.

[11] K. Bhardwaj, P. Mane and Jörg Henkel, "Power-and area-efficient approximate wallace tree multiplier for error-resilient systems," in *Proc. 15th IEEE Int. Symp. Quality Electronic Design*, 2014, pp. 263-269.

[12] H. Jiang, J. Han, F. Qiao, and F. Lombardi, "Approximate radix-8 Booth multipliers for low-power and high performance operation," *IEEE Trans. Comput.*, vol. 65, pp. 2638-2644, Aug. 2016.

[13] V. Leon, G. Zervakis, D. Soudris, and K. Pekmestzi, "Approximate hybrid high radix encoding for energy-efficient inexact multipliers," *IEEE Trans. Very Large Scale Integr. (VLSI) Systems*, vol. 26, no. 3, pp. 421-430, 2018.

[14] W. Liu, L. Qian, C. Wang, H. Jiang, J. Han, and F. Lombardi, "Design of approximate radix-4 booth multipliers for error-tolerant computing," *IEEE Trans. Comput.*, vol. 66, pp.1435-1441, Aug. 2017.

[15] K. J. Cho, K. C. Lee, J. G. Chung, and K. K. Parhi, "Design of low error fixed-width modified Booth multiplier," *IEEE Trans. Very Large Scale Integr. (VLSI) Systems*, vol. 12, no. 5, pp. 522-531, 2004.

[16] M. J. Schulte, and E. E. Swartzlander Jr., "Truncated multiplication with correction constant," in *Proc. Workshop Very Large Scale Integr. (VLSI) Signal Process. VI*, 1993, pp. 388-396.

[17] E. J. King and E. E. Swartzlander Jr., "Data dependent truncated scheme for parallel multiplication," in *Proc. 31st Asilomar Conf. Signals, Circuits Syst.*, 1998, pp. 1178-1182.

[18] Y. H. Chen and T. Y. Chang, "A high-accuracy adaptive conditional-probability estimator for fixed-width Booth multipliers," *IEEE Trans. Circuits Syst. I: Reg. Papers*, vol. 59, no. 3, pp. 594-603, 2012.

[19] G. Zervakis, K. Tsoumanis, S. Xydis, D. Soudris, and K. Pekmestzi, "Design-efficient approximate multiplication circuits through partial product perforation," *IEEE Trans. Very Large Scale Integr. (VLSI) Systems*, vol. 24, no. 10, pp. 3105-3117, 2016.

[20] C. H. Lin and I. C. Lin, "High accuracy approximate multiplier with error correction," in *Proc. International Conference on Computer Design*, 2013, pp. 33-38.

[21] A. Momeni, J. Han, P. Montuschi, and F. Lombardi, "Design and analysis of approximate compressors for multiplication," *IEEE Trans. Comput.*, vol. 64, no. 4, pp. 984-994, 2015.

[22] C. Liu, J. Han, and F. Lombardi, "A low-power, high-performance approximate multiplier with configurable partial error recovery," in *Proc. Design, Autom.,Test Eur. (DATE)*, 2014, pp. 1-4.

[23] N. Maheshwari, Z. Yang, J. Han, and F. Lombardi, "A design approach for compressor based approximate multipliers," in *Proc. Int. Conf. Very Large Scale Integr. (VLSI) Design and Embedded Systems*, 2015, pp. 209-214.

[24] K. Y. Kyaw, L. G. Wang, and K. S. Yeo,"Low-power high-speed multiplier for error tolerant application," in *Proc. IEEE International Conference of Electron Devices and Solid-State Circuits (EDSSC)*, 2010, pp. 1-4.

[25] S. Narayanamoorthy, H. A. Moghaddam, Z. Liu, T. Park, and N.S. Kim, "Energy-efficient approximate multiplication for digital signal processing and classification applications," *IEEE Trans. VLSI Systems*, vol. 23, no. 6, pp. 1180-1184. 2015.

[26] H. Soheil, B. R. Iris, and R. Sherief, "DRUM: A dynamic range unbiased multiplier for approximate applications," in *Proc. IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2015, pp. 418-425.

[27] J. N. Mitchell, "Computer multiplication and division using binary logarithms," *IRE Transactions on Electronic Computers*, vol. EC-11, no. 4, pp. 512-517, 1962.

[28] J. Low and C. Jong, "Unified Mitchell-based approximation for efficient logarithmic conversion circuit," *IEEE Trans. Computers*, vol. 64, no. 6, pp. 1783-1797, 2015.

[29] S. Ahmed, S. Kadam, and M. Srinivas, "An iterative logarithmic multiplier with improved precision," in *Proc. IEEE 23nd Symposium on Computer Arithmetic*, 2016, pp. 104-111.

[30] W. Liu, J. Xu, D. Wang, C. Wang, P. Montuschi, and F. Lombardi, "Design and evaluation of approximate logarithmic multipliers for low power error-tolerant applications," *IEEE Trans. Circuits and Systems I: Regular Papers*, vol. 65, no. 9, pp. 2856-2868, 2018.

[31] M. S. Kim, A. A. Del Barrio, L. Tavares Oliveira, R. Hermida, N. Bagherzadeh, "Efficient Mitchell's approximate log multipliers for convolutional neural networks," *IEEE Trans. Computers*, vol. 68, no. 5, pp. 660-675, 2019.

[32] P. Yin, C. Wang, W. Liu, and F. Lombardi, "Design of Dynamic Range Approximate Logarithmic Multipliers," in *Proc. Great Lakes Symposium on VLSI*, 2018, pp. 423-426.

[33] V. Mrazek, Z. Vasicek, L. Sekanina, H. Jiang, and J. Han, "Scalable construction of approximate multipliers with formally guaranteed worst case error," *IEEE Trans. VLSI Systems*, vol. 26, no. 11, pp. 2572-2576, 2018.

[34] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," *IEEE Trans. Computers*, vol. 63, no. 9, pp. 1760-1771, 2013.

[35] V. Mrazek, R. Hrbacek, Z. Vasicek, and L. Sekanina, "EvoApprox-Sb:Library of approximate adders and multipliers for circuit design and benchmarking of approximation methods," in *Proc. Design, Autom. Test Eur. (DATE)*, Mar. 2017, pp. 258-261.

[36] 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 and SystemsI: regular papers*, vol. 57, no. 4, , pp. 850-862, April 2010.

[37] W. Liu, L. Qian, C. Wang, H. Jiang, J. Han, F. Lombardi, "Design of approximate radix-4 booth multipliers for error-tolerant computing", *IEEE Trans. Computers*, vol. 66, no. 8, pp. 1435-1441, 2017.

[38] W. Liu, T. Cao, P. Yin, C. Wang, E. E. Swartzlander, F. Lombardi, "Design and analysis of approximate redundant binary multipliers", *IEEE Trans. Computers*, vol. 68, no. 6, pp. 804-819, 2019.

[39] K. Bache and M. Lichman, *UCI machine learning repository*, http://archive.ics.uci.edu/ml, 2013.

[40] J. Kung, D. Kim, and S. Mukhopadhyay, "A power-aware digital feed-forward neural network platform with backpropagation driven approximate synapses, in *Proc. IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED)*, 2015, pp. 85-90.

[41] Z. Du, A. Lingamneni, Y. Chen, K. Palem, O. Temam, and C. Wu, "Leveraging the error resilience of neural networksfor designing highly energy efficient accelerators," *IEEE Trans. Computer-aided Design of Integrated Circuits and Systems*, vol. 34, no. 8, pp. 1223-1235, 2015.

[42] E. Forgy, "Cluster analysis of multivariate data: Efficiency vs. interpretability of classifications," *Biometrics*, vol. 21, no. 3, pp. 768-769, 1965.

[43] L. Rushi, D. Snehlata, and M. Latesh, "Class imbalance problem in data mining: review," *Int. J. of Computer Science and Network*, vol. 2, no. 1, pp. 83-87, 2013.

[44] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, Vol. 86, No. 11, pp. 2278-2324, 1998.

[45] Y. LeCun, C. Cortes, and C. J.C. Burges, "The MNIST database of handwritten digits," http://yann.lecun.com/exdb/mnist/.

Weiqiang Liu (S'10-M'12-SM'15) received the B.Sc. degree in Information Engineering from Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China and the Ph.D. degree in Electronic Engineering from the Queen's University Belfast (QUB), Belfast, UK, in 2006 and 2012, respectively. In Dec. 2013, he joined the College of Electronic and Information Engineering, NUAA, where he is currently an Associate Professor. He has published one research book by Artech House and over 100 leading journal and conference papers. He serves as an Guest Editor of Proceedings of the IEEE, an Associate Editor of IEEE Transactions on Computers, IEEE Transactions on Circuits and Systems I: Regular Papers and IEEE Transactions on Emerging Topics in Computing, and a Steering Committee Member of IEEE Transactions on MultiScale Computing Systems (TMSCS). He has been a program committee member for several conferences including ARITH, DATE, ISCAS, ASP-DAC, ASAP, ISVLSI, GLSVLSI, NANOARCH, AisaHOST, SiPS, ICCD, AICAS and ICONIP. His research interests include computer arithmetic,approximate computing, emerging technologies in computing systems and hardware security.

Peipei Yin received the B.Sc. and M.Sc. degrees from Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China, in 2010 and 2013, respectively. Since Sep. 2015, she has been pursuing the Ph.D. degree in the College of Electronic and Information Engineering, NUAA, Nanjing, China. Her research interests include computer arithmetic, fault tolerance systems, and low power technologies in approximate computing.

Chenghua Wang received the B.Sc. and M.Sc. degrees from Southeast University, Nanjing, China, in 1984 and 1987, respectively. In 1987, he joined the College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, where he became a full Professor in 2001. He has published 6 books and over 100 technical papers in journals and conference proceedings. He is the recipient of more than ten teaching and research awards at the provincial and ministerial level. His current research interests include testing of integrated circuits, and circuits & systems for communications.

Haroon Waris received B.Sc. and M.Sc. degrees in electronics engineering from the Center for Advance Studies in Engineering (CASE), Islamabad, Pakistan, in 2009 and 2012, respectively. From May. 2011 to Aug. 2017, he worked as Embedded Design Engineer at Centre of Excellence in Science and Applied Technologies (CESAT), Islamabad, Pakistan. Since Sep. 2017, he has been pursuing the Ph.D. degree in the College of Electronic and Information Engineering, NUAA, Nanjing, China. His research interests include high-level synthesis for approximate computing, approximate image processing on FPGAs and approximate accelerators for convolutional neural networks (CNNs).

Yinhe Han (SM'06) received the B.Eng. degree from Nanjing University of Aeronautics and Astronautics, China, in 2001, and Ph.D. degree from the Institute of Computing Technology (ICT), Chinese Academy of Sciences (CAS), in 2006. He is currently a Professor at the State Key Lab. of Computer Architecture, ICT, CAS. His research interests include Computer Architecture, Chip design for Intelligent Robot. He has published more than 110 papers in these areas, including ISCA, HPCA, DAC, TC and other top conferences and journals. Prof. Han was program chair of ATS'2014, finance/publicity chair of HPCA'2013/17, program co-chair of WRTLT 2009, and served on the Technical Program Committees of several IEEE and ACM conferences, including DAC'17, PACT'14, HPCA'13 etc. He was the Chair of Young Computer Scientists & Engineers Forum (YOCSEF), China Computer Federation(CCF) (2016-2017), and is the Secretary General of CCF Technical Committee on Fault-Tolerant Computing (2016-2019). He is a senior member of IEEE.

Fabrizio Lombardi (M'81-SM'02-F'09) graduated in 1977 from the University of Essex (UK) with a B.Sc. (Hons.) in Electronic Engineering. In 1977 he joined the Microwave Research Unit at University College London, where he received the Master in Microwaves and Modern Optics (1978), the Diploma in Microwave Engineering (1978) and the Ph.D. from the University of London (1982). He is currently the holder of the International Test Conference (ITC) Endowed Chair Professorship at Northeastern University, Boston. In the past Dr. Lombardi was the Editor-In-Chief of the IEEE Transactions on Computers and the inaugural Editor-in-Chief of the IEEE Transactions on Emerging Topics in Computing. Currently, he is the Editor-in-Chief of the IEEE Transactions on Nanotechnology. In 2019, he will serve as the Vice President for Publications of the IEEE Computer Society. His research interests are bio-inspired and nano manufacturing/computing, VLSI design, testing, and fault/defect tolerance of digital systems.