Received 22 October 2018; revised 7 April 2019; accepted 5 July 2019.
Date of publication 17 July 2019; date of current version 16 September 2021.

Digital Object Identifier 10.1109/TETC.2019.2929100

Design and Analysis of Majority Logic-Based Approximate Adders and Multipliers

WEIQIANG LIU, (Senior Member, IEEE), TINGTING ZHANG, EMMA MCLARNON, MAIRE O'NEILL, (Senior Member, IEEE), PAOLO MONTUSCHI, (Fellow, IEEE), AND FABRIZIO LOMBARDI, (Fellow, IEEE)

W. Liu and T. Zhang are with the College of Electronic and Information Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing, 211106, China
E. McLarnon and M. O'Neill are with the Institute of Electronics, Information and Communication Technologies, Queen's University Belfast, Belfast, BT3 9DT, United Kingdom
P. Montuschi is with the Department of Control and Computer Engineering, Politecnico di Torino, Turin, 10129, Italy
F. Lombardi is with the Department of Electrical and Computer Engineering, Northeastern University, Boston, MA 02115, USA
CORRESPONDING AUTHOR: W. LIU (liuweiqiang@nuaa.edu.cn)

ABSTRACT As a new paradigm for nanoscale technologies, approximate computing deals with error tolerance in the computational process to improve performance and reduce power consumption. Majority logic (ML) is applicable to many emerging nanotechnologies; its basic building block (the 3-input majority voter, MV) has been extensively used for digital circuit design. In this paper, designs of approximate adders and multipliers based on ML are proposed; the proposed multipliers utilize approximate compressors and a reduction circuitry with so-called complement bits. An influence factor is defined and analyzed to assess the importance of different complement bits depending on the size of the multiplier; a scheme for selection of the complement bits is also presented. The proposed designs are evaluated using hardware metrics (such delay and gate complexity) as well as error metrics. Compared with other ML-based designs found in the technical literature, the proposed designs are found to offer superior performance. Case studies of error-resilient applications are also presented to show the validity of the proposed designs.

INDEX TERMS Majority logic, approximate adder, approximate multiplier, complement bits, approximate compressor, image processing

I. INTRODUCTION

As one of the main obstacles to attain high performance, power dissipation is increasingly been investigated for IC design. Approximate computing is a promising technique to reduce power consumption and improve performance of circuits and systems by introducing computational errors for error-tolerant applications, such as multimedia signal processing, machine learning and pattern recognition [1], [2].

Approximate computer arithmetic circuits based on CMOS technology have been extensively studied. Designs of approximate adders, multipliers and dividers for both fixed-point and floating-point formats have been proposed [3], [4], [5], [6]. Error metrics such as the mean error distance (MED), the normalized MED (NMED) and the relative MED (RMED) [7] have been proposed to analyze the errors introduced in the operations of approximate arithmetic circuits.

As CMOS is approaching its technology limitations, emerging nanotechnologies have been proposed at the end of the so-called Moore's Law, such as Quantum-dot Cellular Automata (QCA) [8], [9], nanomagnetic logic (NML) [10], and spin-wave devices (SWD) [11]. All of these technologies rely on majority logic (ML) as digital design framework; this is different from conventional Boolean logic. The majority gate performs a multi-input logic operation (Figure 1); the logic expression of the 3-input majority gate (voter, MV) is given by

$$F = M(A, B, C) = AB + BC + AC. \quad (1)$$

It is expected that significant improvement in power consumption could be achieved by applying approximate computing also to emerging nanotechnologies. However, approximate designs of CMOS circuits cannot be directly applied due to the underlying different logic; few designs of ML based approximate circuits have been studied [12], [13], [14], [15]. [12] has proposed a 1-bit approximate full adder

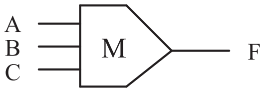

FIGURE 1. **Majority gate (3-input voter).**

(AFA), but no multi-bit designs suitable for practical applications have been outlined. Several ML-based AFAs have been proposed in [13]; 1-bit as well as multi-bit AFAs are considered. For an approximate multiplier (AM), [14] has proposed a 4:2 approximate compressor based on truth table manipulation for designing an approximate multiplier for image processing. [15] has proposed two 4:2 approximate compressors based on the 1-bit AFA of [12].

In this paper, both ML-based AFAs (MLAFAs), and ML-based AMs (MLAMs) are designed. For the MLAFA, multi-bit MLAFAs are designed by combining 1-bit MLAFAs. Moreover, a 2-bit MLAFA with a higher accuracy is designed based on logic reduction. For the MLAM, a 2×2 design with complement bits is proposed. Furthermore, complement bit selection is analyzed as function of the size of a multiplier; a so-called influence factor is introduced to assess the importance of different complement bits. Few ML-based approximate compressors (MLACs) are designed by MLAFAs or K-Map simplification; then they are employed in the reduction circuitry. Error analysis and a hardware evaluation are presented to validate the proposed designs. Case studies with the proposed approximate adders and multipliers for image processing are also provided as part of this assessment.

This paper has been extended significantly from its previous conference version [13]. The main differences are summarized as follows:

1) A new 2-bit MLAFA is proposed based on truth table reduction; it can be used for multi-bit approximate adder design;
2) A 2×2 MLAM is proposed and complement bits are introduced;
3) A novel analysis for selecting complement bits is presented;
4) MLACs based on K-Map simplification and 1-bit MLAFAs are proposed;
5) Exact as well as approximate pipelined reduction circuits for 4×4 and 8×8 MLAMs are proposed;
6) Case studies are provided for image processing as application using the proposed MLAFAs and MLAMs.

The paper is organized as follows: Section II reviews related works and error metrics. Designs of ML based approximate full adders are presented in Section III (together with evaluation and application). Section IV presents the design of approximate multipliers by introducing complement bits and approximate compression which utilizes approximate compressors and approximate adders. The application of the proposed approximate multipliers to image processing and comparison with previous designs are also presented in Section IV. Section V concludes this paper.

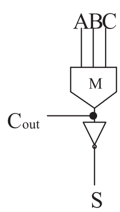

FIGURE 2. **The schematic diagram of MLAFA1 [12].**

TABLE 1. **Truth table of 1-bit MLAFAs.**

Inputs			EFA		MLAFA1[12]		MLAFA2	
A	B	C	C_{out}	S	C_{out}	S	C_{out}	S
0	0	0	0	0	0	1	0	0
0	0	1	0	1	0	1	1	0
0	1	0	0	1	0	1	0	1
0	1	1	1	0	1	0	1	0
1	0	0	0	1	0	1	0	1
1	0	1	1	0	1	0	1	0
1	1	0	1	0	1	0	0	1
1	1	1	1	1	1	0	1	1

II. RELATED WORKS

A. ML-BASED APPROXIMATE DESIGNS

1) ML-BASED APPROXIMATE FULL ADDER

A 1-bit MLAFA (MLAFA1) has been proposed in [12] (Figure 2). The inputs are given by A, B, C while S and C_{out} are the outputs. MLAFA1 generates the output S as the complement of C_{out}; it introduces 2 errors (among the 8 input combinations) when computing the output S (Table 1), but saving two majority gates and one inverter. The circled entries in the truth table denote the instances in which the outputs of MLAFA differ from the exact full adder (EFA). The equations for the carry out and the sum are as follows:

$$C_{out} = M(A, B, C) \tag{2}$$

$$S = \overline{C_{out}}. \tag{3}$$

2) ML-BASED APPROXIMATE 4:2 COMPRESSOR

As part of a multiplier, a compressor plays an important role. Let the inputs be P_5, P_4, P_3, P_2, P_1 and the outputs be *Sum*, C_{out}, *Carry*; the implementation of a 4:2 compressor consists of two serially connected 1-bit full adders [16].

Using a truth table, [14] has proposed an approximate 4:2 compressor (MLAC1) (Figure 3(a)). In MLAC1, the *Carry* output has the same logic with the input P_5 in 24 out of 32 cases, and similarly, C_{out} has the same value as P_4 in 24 out of 32 cases. So *Sum* output is modified to reduce the error. The equations for the outputs are as follows:

$$C_{out} = P_4 \tag{4}$$

$$Carry = P_5 \tag{5}$$

$$Sum = M(P_2, P_3, \overline{M}(P_4, P_5, \overline{P_1})). \tag{6}$$

IEEE TRANSACTIONS ON
EMERGING TOPICS
IN COMPUTING

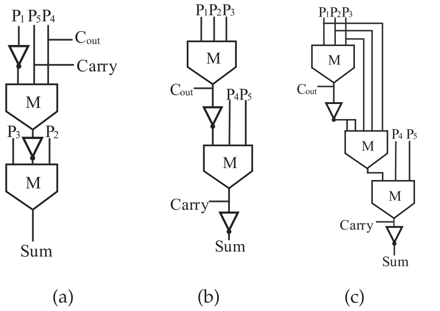

FIGURE 3. Schematic diagrams of 4:2 MLACs: (a) MLAC1 [14], (b) MLAC2 [15] (two 1-bit MLAFAs), and (c) MLAC3 [15] (one 1-bit MLAFA and one 1-bit EFA).

[15] has proposed two MLACs (Figures 3(b) and 3(c)). Figure 3(b) employs two 1-bit MLAFA1s [12] to substitute the two 1-bit EFAs, so resulting in 12 errors out of 32 cases; to improve the accuracy, the 1-bit MLAFA1 replaces one of the two EFAs in MLAC3. MLAC3 requires a 5-input majority gate; the proposed designs are all based on 3-input majority gates, therefore MLAC3 is not considered for comparison in the paper.

3) ERROR METRICS

As approximate computing introduces errors, metrics are required to evaluate the accuracy of approximate circuits. In this paper, we evaluate approximate designs by the Normalized Mean Error Distance, and the Maximum Absolute Error (MAE). The NMED is the normalizing Mean Error Distance. The Mean Error Distance is defined as the average of the Error Distance (ED) which is the absolute difference between the approximate and the accurate results across all possible inputs. MAE is defined as the maximum absolute error. The definitions of ED, MED, NMED, and MAE are as follows:

$$ED = |(ExR - ApR)| \tag{7}$$

$$MED = \frac{\sum ED}{N} \tag{8}$$

$$NMED = \frac{MED}{MAX} \tag{9}$$

$$MAE = max\{ED\}, \tag{10}$$

where ExR, ApR, N and MAX denote the accurate result, the approximate result, the counts of all possible inputs and the maximum value of the result, respectively.

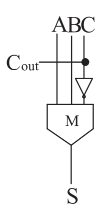

FIGURE 4. The schematic diagram of proposed MLAFA2.

TABLE 2. Comparison of 1-bit MLAFAs.

Types of 1-bit Adders	MV	INV	D	D_{carry}	NMED	MAE
EFA	3	2	2	1	0	0
MLAFA1[12]	1	1	1	1	0.083	1
MLAFA2	1	1	1	0	0.083	1

III. ML-BASED APPROXIMATE FULL ADDER

In this section, a new 1-bit MLAFA (MLAFA2) is proposed; it is also compared with the 1-bit EFA and the previous 1-bit MLAFA1 of [12]. Moreover, 2-bit MLAFAs are proposed by utilizing two methods: the first method merges the proposed and the previous 1-bit MLAFAs; the second method is based on a truth table reduction process for the 2-bit design. Multi-bit MLAFAs are also designed by cascading the proposed designs. Both designs and corresponding errors are evaluated and assessed. A case study for image processing is also provided.

A. PROPOSED 1-BIT MLAFA

A new 1-bit MLAFA, namely MLAFA2 is proposed (Figure 4). Consider Table 1, C_{out} is nearly the same as C except two cases out of the 8 input cases. Therefore, in Eq. (11), C can be approximately made equal to C_{out}.

$$C_{out} = C. \tag{11}$$

The approximate output C_{out} can be substituted into the exact expression of S to obtain the approximate S as follows:

$$S = M(\overline{C_{out}}, M(A, B, \overline{C}), C) = M(A, B, \overline{C}). \tag{12}$$

The MED and NMED of MLAFA2 are given by

$$MED_{MLAFA2} = \frac{1}{8}(0 + 1 + 0 + 0 + 0 + 0 + 1 + 0) = 0.25 \tag{13}$$

$$NMED_{MLAFA2} = \frac{MED_{MLAFA2}}{3} = 0.083. \tag{14}$$

A comparison in terms of number of majority gates (MV), number of inverters (INV), NMED, MAE, delay (D) and delay of carry (D_{carry}) between EFA, MLAFA1 [12] and the proposed MLAFA2 is reported in Table 2. When considering ML-based nanotechnologies, delay (as assessed in this paper) is normalized by the number of majority gates only (so, the delay for the inverters is not included because it is often very small compared to the ML gate) [17]. Compared with EFA, MLAFA2 saves two majority gates, one inverter and one delay. MLAFA2 decreases the delay of carry to 0, compared with MLAFA1 [12], which can reduce the length of the critical path for large scale designs. Although the proposed MLAFA2 incurs a large error for Cout (which could be propagated to the higher bits), the combination of MLAFA1 [12] and MLAFA2 introduce fewer errors than only cascading MLAFA1 [12]. This is verified next.

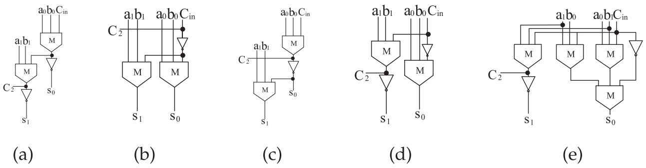

FIGURE 5. Schematic diagrams of proposed 2-bit MLAFAs: (a) MLAFA11, (b) MLAFA22, (c) MLAFA12, (d) MLAFA21, and (e) MLAFA33.

B. PROPOSED 2-BIT MLAFAS

In this section, 2-bit MLAFAs are proposed by using two methods. The first designs merge the proposed MLAFA2 and MLAFA1 (hence the hybrid nature); the second method designs the 2-bit MLAFA using a truth table reduction process. The inputs to the 2-bit adder are given by $A = a_1a_0$, $B = b_1b_0$, C_{in}, while $S = s_1s_0$, and C_2 are the outputs.

1) HYBRID 2-BIT MLAFA BASED ON MLAFA2

By cascading two 1-bit MLAFAs (MLAFA1 and MLAFA2), four different combinations are considered for the 2-bit MLAFAs; they are shown in Figures 5(a), 5(b), 5(c), and 5(d). MLAFA1 cascaded with MLAFA1 results in the 2-bit MLAFA11 design. Similarly, MLAFA2 cascaded with MLAFA2 results in the MLAFA22 design. MLAFA12 consists of MLAFA1 and MLAFA2, in which MLAFA1 is used to compute the least significant bits (LSBs); in MLAFA21, MLAFA2 is used to compute the LSBs.

2) 2-BIT MLAFA FROM TRUTH TABLE REDUCTION

For two operands A and B, there are four possible combinations. Under an assumed Gaussian distribution for image processing, $A = 00$ or $B = 00$ and $A = 11$ or $B = 11$ are not considered to ensure a low complexity by using a truth table. The reduced truth table is shown in Table 3. The exact expressions for the outputs in these eight cases are given in Eqs. (15), (16), and (17); the schematic diagram is illustrated in Figure 5(e), this design is hereafter referred to as MLAFA33.

$$C_2 = M(A_1, B_1, C_{in}) \tag{15}$$

$$\begin{aligned} s_0 &= M(M(\overline{A_0}, B_0, C_{in}), M(A_0, \overline{B_0}, C_{in}), \overline{C_{in}}) \\ &= M(M(A_1, B_0, C_{in}), M(A_0, B_1, C_{in}), \overline{C_{in}}) \end{aligned} \tag{16}$$

$$s_1 = \overline{C_2}. \tag{17}$$

TABLE 3. Reduced truth table of 2-bit MLAFA.

Inputs					MLAFA33		
A		B		C_{in}	C_2	S	
a_1	a_0	b_1	b_0			s_1	s_0
0	1	1	0	0	0	1	1
0	1	1	0	1	1	0	0
0	1	0	1	0	0	1	0
0	1	0	1	1	0	1	1
1	0	1	0	0	1	0	0
1	0	1	0	1	1	0	1
1	0	0	1	0	0	1	1
1	0	0	1	1	1	0	0

TABLE 4. Comparison of 2-bit MLAFAs.

Types of 2-bit Adders	MV	INV	D	MAE	NMED
MLAFA11	2	2	2	3	0.107
MLAFA22	2	1	1	3	0.107
MLAFA12	2	1	2	2	0.089
MLAFA21	2	2	1	2	0.089
MLAFA33	4	2	2	2	0.080

3) COMPARISON AND DISCUSSION

The proposed 2-bit approximate hybrid adders based on MLAFA1 and MLAFA2 introduce errors for 14 of the 32 input cases; the design based on truth table reduction generates errors for 16 of the 32 input cases. The MAE and NMED of these MLAFAs are provided in Table 4. MLAFA22 shows the best performance in delay; the errors due to the inverters have more significance in a multi-bit design. Moreover, hybrid MLAFAs designed by cascading two of the same type of 1-bit MLAFAs have larger errors than cascading two different types of 1-bit MLAFAs. Consider the number of required gates, MLAFA12 requires one less inverter than MLAFA21; in terms of delay, MLAFA21 incurs in 1 less delay than MLAFA12.

For MLAFA33, two additional majority gates are needed than other 2-bit MLAFAs; however, the NMED is decreased by 10 percent compared with MLAFA12 and MLAFA21.

C. PROPOSED MULTI-BIT MLAFAS

In this section, multi-bit MLAFAs are considered (including 4-bit and 8-bit designs) by cascading 2-bit MLAFAs.

1) PROPOSED 4-BIT MLAFAS

Consider a 4-bit MLAFA with inputs given by $A = a_3a_2a_1a_0$, $B = b_3b_2b_1b_0$, C_{in} and outputs given by $S = s_3s_2s_1s_0$, C_4. Similar to the proposed hybrid 2-bit MLAFAs, 4-bit MLAFAs can be designed by cascading two 2-bit MLAFAs (MLAFA12 and MLAFA21).

Table 5 shows that the proposed designs require fewer gates than an EFA, but at the cost of a reduced accuracy. An improvement of up to 50 percent in delay is achieved. Although MLAFA1221 has advantages in terms of the reduced number of gates and delay, its MED/NMED is the largest. MLAFA2121 and MLAFA2112 have the same MED/NMED, but MLAFA2121 has less delay. Compared with MLAFA2112, MLAFA1212 requires one less inverter with a reduction in MED. Therefore, MLAFA2121 is the best design which contributes to a reduction of 67 percent in majority gates and delay. Moreover, the schemes in which two of the same type of the proposed 2-bit MLAFAs are cascaded have better performance than cascading different types of MLAFAs.

MLAFA33 is employed to control the introduced error into the most significant bits (MSBs), as shown in Figure 6(e)-(f). From Table 5, the designs using MLAFA33 decrease the

TABLE 5. **Comparison of 4-bit MLAFAs.**

Types of 4-bit Adders	MV	INV	D	MAE	NMED
CFA4[18]	12	8	6	0	0
RCA4[19]	12	4	7	0	0
MLAFA1212	4	2	3	10	0.091
MLAFA2121	4	3	2	10	0.092
MLAFA2112	4	3	3	10	0.092
MLAFA1221	4	2	2	10	0.175
MLAFA1233	6	2	3	9	0.083
MLAFA2133	6	3	3	10	0.081

NMED below 0.084 at a small hardware utilization. Improvements of up to 50 percent in delay and majority gates can be achieved compared with the exact counterparts.

2) PROPOSED 8-BIT MLAFAS

Consider an 8-bit MLAFA with inputs $A = a_7a_6a_5a_4a_3a_2a_1a_0$, $B = b_7b_6b_5b_4b_3b_2b_1b_0$, C_{in} and outputs $S = s_7s_6s_5s_4s_3s_2s_1s_0$, C_8. 8-bit MLAFAs are designed by cascading two 4-bit MLAFAs by using MLAFA1212 and MLAFA2121.

The comparison results are presented in Table 6. The proposed designs significantly reduce the number of gates and delay but at a decrease in accuracy. In terms of gates, MLAFA1212-1212 and MLAFA1212-2121 require one less inverter than the other adders; MLAFA2121-2121 and MLAFA1212-2121 incur in a smaller delay than the other adders. Compared with MLAFA2121-2121, MLAFA1212-1212 is superior; the design is also better than the other two designs, resulting in an improvement of 67 percent in majority gates and 50 percent in delay. So the designs whose LSBs are processed by MLAFA1212 show considerable advantages.

For a higher accuracy, we take advantage of MLAFA33 to substitute the MSBs of MLAFA1212-1212, whose NMED is the smallest. From Table 6, MLAFA33 improves the accuracy by decreasing the NMED to approximately 0.08. MLAFA1212-1233 (with a reduction of 58 percent in majority gates as well as 50 percent in delay) and MLAFA1212-3333 (with a reduction of 50 percent in majority gates as well as 50 percent in delay) are superior to other designs. Moreover when more than 2 MLAFA33 are used, the MAE increases. Therefore, MLAFA1212-3333 and MLAFA1212-1233 are superior to other designs. As for those applications which have relatively low error-tolerance, exact adders can substitute MLAFA33 in MLAFA1212-3333 or MLAFA1212-3333, with relatively smaller errors.

TABLE 6. **Comparison of 8-bit MLAFAs.**

Types of 8-bit Adders	MV	INV	D	MAE	NMED
CFA8[18]	24	16	10	0	0
RCA8[19]	24	8	11	0	0
MLAFA1212-1212	8	4	5	170	0.090
MLAFA2121-2121	8	5	4	170	0.092
MLAFA2121-1212	8	5	5	170	0.091
MLAFA1212-2121	8	4	4	170	0.092
MLAFA1212-1233	10	5	5	149	0.082
MLAFA1212-3333	12	5	5	145	0.081
MLAFA1233-3333	14	5	5	169	0.080
MLAFA3333-3333	16	5	5	170	0.080

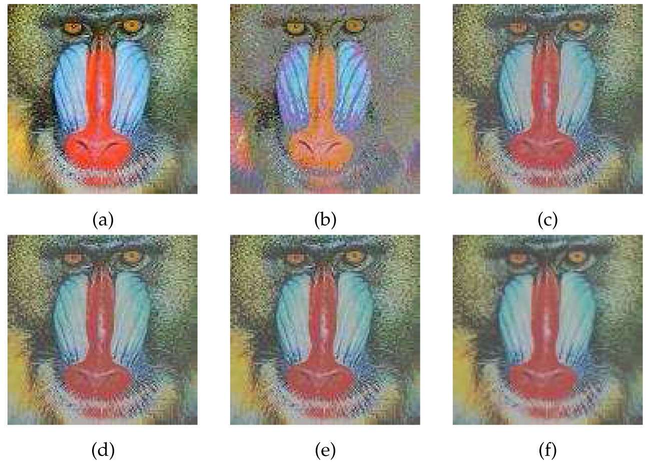

FIGURE 6. **Image processing of 8-bit MLAFAs: (a) original, (b) MLAFA1212-1212, (c) MLAFA1212-1233, (d) MLAFA1212-3333, (e) MLAFA1233-3333, and (f) MLAFA3333-3333.**

D. IMAGE PROCESSING APPLICATION WITH MLAFAS

The proposed 8-bit MLAFAs are applied to image processing when adding two of the same images pixel by pixel and combining them into a single output image. To evaluate the perceived quality of the output, the structural similarity (SSIM) [20] and Peak Signal to Noise Ratio (PSNR) [21] are calculated for each image. The SSIM index takes a decimal value between -1 and 1, and the value of 1 is reached only when the two inputs are the same. The PSNR is the logarithm of the squared error between the original image and the processed image relative to the square of the maximum value of the signal; its unit is dB. The greater the PSNR value is, the less distortion it represents.

The obtained results are shown in Figure 6 and Table 7. As the number of MLAFA33 used increases, the SSIM or the PSNR increases. MLAFA1212-3333 and MLAFA1212-1233 are the best designs for 8-bit MLAFAs. If a higher accuracy is required, exact adders can substitute the MSBs, which can increase SSIM to over 0.9.

IV. ML-BASED APPROXIMATE MULTIPLIERS

The designs of ML based approximate multipliers are studied in this section based on 2×2 MLAMs. The so-called

TABLE 7. **Image processing results of 8-bit MLAFAs.**

Types of 8-bit Adders	SSIM	PSNR(dB)
MLAFA1212-1212	0.2969	28.69
MLAFA1212-1233	0.7314	32.03
MLAFA1212-3333	0.8085	32.30
MLAFA1233-3333	0.8087	32.31
`MLAFA3333-3333	0.8090	32.31

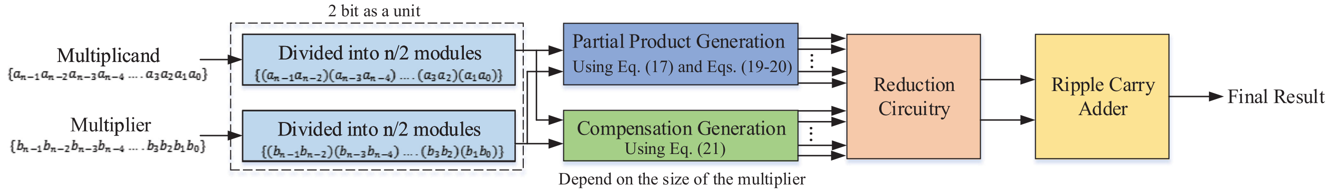

FIGURE 7. Proposed Design Flow of $n \times n$ MLAMs.

complement bit is introduced through a selection scheme to compensate errors.

Consider Figure 7 and the proposed design flow of $n \times n$ MLAMs. The multiplicand $a_{n-1}a_{n-2}a_{n-3}a_{n-4}\cdots a_3a_2a_1a_0$ and the multiplier $b_{n-1}b_{n-2}b_{n-3}b_{n-4}\cdots b_3b_2b_1b_0$ are first divided into N/2 modules (each of 2 bits as a unit); then, these modules are substituted into the expression to calculate the partial product, while at the same time, selectively adding the compensation bits as per the size of the multiplier. Next, for efficient compression, a partial product reduction (PPR) circuitry which uses exact or approximate compression is employed. This depends on the distribution of the generated partial products (PPs) and the compensation bits, such that the PP of two rows (or a carry in the lowest order) can be obtained. Finally, the final product can be calculated by the final exact adder.

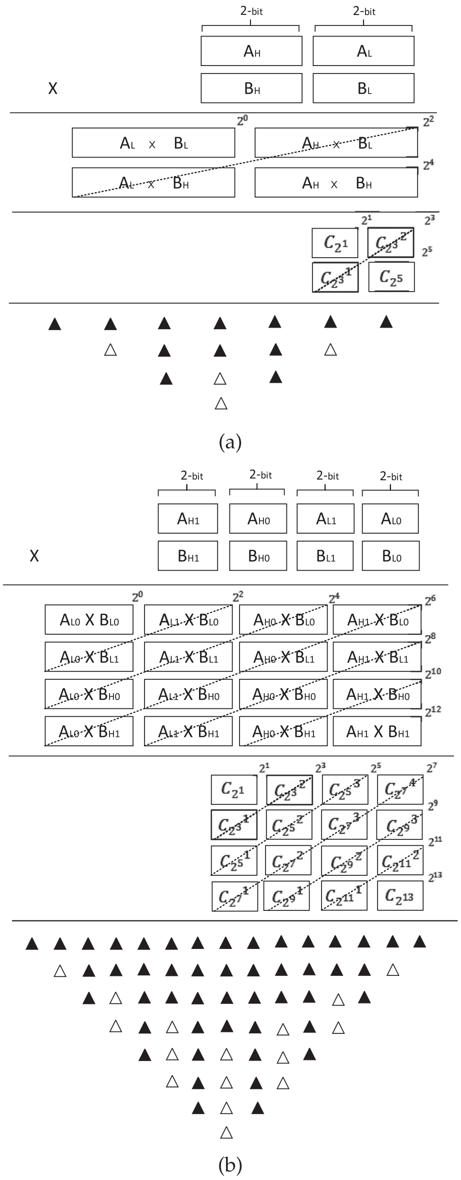

FIGURE 8. PP generation and complement bit generation of MLAMs: (a) 4×4 MLAM, and (b) 8×8 MLAM.

A. 2×2 *MLAM*

By mapping the 2×2 AM design [22] into ML (as per Eqs. (18), (19), and (20)), out_1 requires three majority gates, which is two more than out_0 and out_2.

$$out_0 = M(A_0, B_0, 0) \tag{18}$$

$$out_1 = M(M(A_1, B_0, 0), M(A_0, B_1, 0), 1) \tag{19}$$

$$out_2 = M(A_1, B_1, 0). \tag{20}$$

Therefore, this should be further improved; furthermore, with an increase of design scale, errors will increase substantially, so unacceptable in most cases. Taking these issues into account, the initial expression is split into two parts (i.e., Eqs. (21) and (22)), one is employed as the out_1 of the 2×2 MLAM, the other is used as a compensation bit (denoted as $\triangle$). In this paper, we just take one case into consideration. The other case follows the same rules.

$$out_1 = M(A_0, B_1, 0) \tag{21}$$

$$\triangle = M(A_1, B_0, 0). \tag{22}$$

By considering the 2×2 MLAM as a module, larger multipliers can be constructed by dividing the operands into several units (Figure 8), where $\triangle$ represents a complement bit. Figure 8 shows the operations of the 4×4 and 8×8 MLAMs with all complement bits that need to be further reduced.

B. COMPLEMENT BIT SELECTION

Too many compensation bits will lead to a larger overhead when calculating the subsequent compression; however, too few compensation bits will cause the final result to lose accuracy. Therefore, a tradeoff must be assessed when selecting an appropriate number of compensation bits.

When selecting the compensation bits and to control the error within a reasonable bound, the compensation bits for the lowest weight are ignored. Moreover, the MSB result will be affected by the LSBs, and the hardware overhead for the MSB compensation bit is not less than the LSB compensation bit. Therefore, when the MSB compensation bits are removed, the LSB compensation bits are also ignored.

In this paper, the complement bits are denoted as ${C_{2^i}}^x$, where 2^i represents the weight of the complement bit and x denotes the signed number of the complement bit if multiple complement bits exist under the current weight. If only a single bit exists for the same weight, the superscript is defaulted; for example, ${C_{2^3}}^1$ denotes the weight of the complement bit (2^3), and 1 is its number (i.e., ${C_{2^3}}^1$ and ${C_{2^3}}^2$ for two items for the same weight).

To measure the importance of each complement bit ${C_{2^i}}^x$ as well as to determine the ignored items, an influence factor denoted by ${P_{{C_{2^i}}^x}}^n$ is defined; this is required to show the impact of a complement bit on the final NMED. For an $n \times n$ MLAM, it can be expressed as follows:

$$ {P_{{C_{2^i}}^x}}^n = \frac{N \times 2^i}{2^n \times 2^n \times (2^n - 1) \times (2^n - 1)}, \tag{23} $$

where 2^i denotes the weight of the complement bit; n denotes the size of the multiplier; N denotes the number of cases that the output ($\triangle$) is 1 when traversing all possible cases (as a function of n). Only when the inputs are both 1, $\triangle$ is effective (and equal to 1). Except the two inputs of $\triangle$, there are $2^{(n-1)} \times 2^{(n-1)}$ possible cases for the inputs of multipliers. Thus, the equation can be written as Eq. (24).

$$ N = 1 \times 2^{(n-1)} \times 2^{(n-1)} = 2^{(2n-2)}. \tag{24} $$

Consequently, Eq. (23) can been further simplified into Eq. (25).

$$ {P_{{C_{2^i}}^x}}^n = \frac{2^{i-2}}{(2^n - 1)^2}. \tag{25} $$

The influence factor ${P_{{C_{2^i}}^x}}^n$ has the following properties:

Property 1: For ${C_{2^i}}^{x_1}$ and ${C_{2^i}}^{x_2}$ of different signed numbers but under the same weight and same size of multiplier,

$$ {P_{{C_{2^i}}^{x_1}}}^n = {P_{{C_{2^i}}^{x_2}}}^n. $$

Proof: From Eq. (24), the value of ${P_{{C_{2^i}}^x}}^n$ is independent of x. Therefore, ${P_{{C_{2^i}}^x}}^n$ remains constant when only x changes. □

Property 2: For ${C_{2^{i_1}}}^{x_1}$ and ${C_{2^{i_2}}}^{x_2}$ of different weight but same size of multiplier,

$$ \frac{{P_{{C_{2^{i_1}}}^{x_1}}}^n}{{P_{{C_{2^{i_2}}}^{x_2}}}^n} = \frac{2^{i_1}}{2^{i_2}}. $$

Proof: From Eq. (24),

$$ \frac{{P_{{C_{2^{i_1}}}^{x_1}}}^n}{{P_{{C_{2^{i_2}}}^{x_2}}}^n} = \frac{2^{i_1-2}}{2^{i_2-2}} = \frac{2^{i_1}}{2^{i_2}}. $$

□

Property 3: For different size of multipliers, as $n_1 \times n_1$ multiplier and $n_2 \times n_2$ multiplier respectively,

$$ (2^{n_1} - 1)^2 \times {P_{{C_{2^i}}^{x_1}}}^{n_1} = (2^{n_2} - 1)^2 \times {P_{{C_{2^i}}^{x_2}}}^{n_2}. $$

Proof: From Eq. (24),

$$ \frac{{P_{{C_{2^i}}^{x_1}}}^{n_1}}{{P_{{C_{2^i}}^{x_2}}}^{n_2}} = \frac{(2^{n_2} - 1)^2}{(2^{n_1} - 1)^2}. $$

□

Property 4: For same size of multiplier, if $i_1 < i_2 < i_3 \cdots < i_{m-1} < i_m$,

$$ {P_{{C_{2^{i_1}}}^{x_1}}}^n < {P_{{C_{2^{i_2}}}^{x_2}}}^n \cdots < {P_{{C_{2^{i_{m-1}}}}^{x_{m-1}}}}^n < {P_{{C_{2^{i_m}}}^{x_m}}}^n. $$

Proof: From Property 2,

$$ \frac{{P_{{C_{2^{i_1}}}^{x_1}}}^n}{{P_{{C_{2^{i_2}}}^{x_2}}}^n} = 2^{i_1 - i_2}, $$

if $i_1 < i_2$, then

$$ \frac{{P_{{C_{2^{i_1}}}^{x_1}}}^n}{{P_{{C_{2^{i_2}}}^{x_2}}}^n} < 1. $$

Thus, ${P_{{C_{2^i}}^x}}^n$ increases with an increase of i. □

Property 5: For different size of multipliers, if $n_1 < n_2 < n_3 \cdots < n_{m-1} < n_m$,

$$ {P_{{C_{2^i}}^{x_1}}}^{n_1} > {P_{{C_{2^i}}^{x_2}}}^{n_2} \cdots > {P_{{C_{2^i}}^{x_{m-1}}}}^{n_{m-1}} > {P_{{C_{2^i}}^{x_m}}}^{n_m}. $$

Proof: From Property 3,

$$ \frac{{P_{{C_{2^i}}^{x_1}}}^{n_1}}{{P_{{C_{2^i}}^{x_2}}}^{n_2}} = \left(\frac{2^{n_2} - 1}{2^{n_1} - 1}\right)^2, $$

if $n_1 < n_2$, then

$$ \frac{{P_{{C_{2^i}}^{x_1}}}^{n_1}}{{P_{{C_{2^i}}^{x_2}}}^{n_2}} > 1. $$

Thus, ${P_{{C_{2^i}}^x}}^n$ decreases with an increase of n. □

Assume that the number of ignored compensation bits is p; as ${P_{{C_{2^i}}^x}}^n$ of different complement bits are mutually independent and linearly superposed, the NMED of a $n \times n$ approximate multiplier is given by

$$ NMED_{n\times n} = \underbrace{{P_{C_{2^1}}}^n + {P_{{C_{2^3}}^1}}^n + {P_{{C_{2^3}}^2}}^n + \cdots}_{p}. \tag{26} $$

For a 4×4 multiplier, there are four complement bits which can be selected (denoted as C_{2^5}, ${C_{2^3}}^1$, ${C_{2^3}}^2$, C_{2^1}); similarly, an 8×8 multiplier has 16 items, including $C_{2^{13}}$, ${C_{2^{11}}}^1$, ${C_{2^{11}}}^2$, ${C_{2^9}}^1$, ${C_{2^9}}^2$, ${C_{2^9}}^3$, ${C_{2^7}}^1$, ${C_{2^7}}^2$, ${C_{2^7}}^3$, ${C_{2^7}}^4$, ${C_{2^5}}^1$, ${C_{2^5}}^2$, ${C_{2^5}}^3$, ${C_{2^3}}^1$, ${C_{2^3}}^2$, C_{2^1}. Based on Property 2, the items of different signed numbers but under the same weight and size of

TABLE 8. The Value of Influence Factor when n = 4 and n = 8.

n	$P_{C_{2^1}x^n}$	$P_{C_{2^3}x^n}$	$P_{C_{2^5}x^n}$	$P_{C_{2^7}x^n}$	$P_{C_{2^9}x^n}$	$P_{C_{2^{11}}x^n}$	$P_{C_{2^{13}}x^n}$
n = 4	2.22×10^{-3}	8.88×10^{-3}	3.56×10^{-2}	-	-	-	-
n = 8	7.69×10^{-6}	3.07×10^{-5}	1.23×10^{-4}	4.92×10^{-4}	1.97×10^{-3}	7.87×10^{-3}	3.14×10^{-2}

multiplier share the same value. Table 8 shows the value of the influence factors when $n = 4$ and $n = 8$.

In Figure 9, the NMED increases as p increases. For different size of multipliers, the largest value of the NMED due to the complement bits is nearly the same. The increase of weight will ultimately increase the NMED; so, when the weight of the compensation bits remains unchanged, the NMED will increase linearly. For a 4×4 multiplier, when p changes from 3 to 4, the NMED increases sharply; for an 8×8 multiplier, when p is smaller than 10, the NMED increases slowly but when p is larger than 10, the NMED increases rapidly.

Similar to the above analysis, the expression for MAE can be found. Another influence factor denoted by $E_{C_{2^i}x^n}$ is defined; this is required to show the impact of a complement bit on the final MAE. For an $n \times n$ MLAM, it can be expressed as follows:

$$E_{C_{2^i}x^n} = 2^i. \tag{27}$$

The MAE of a $n \times n$ approximate multiplier is given by

$$MAE_{n\times n} = \underbrace{E_{C_{2^1}}{}^n + E_{C_{2^3}1}{}^n + E_{C_{2^3}2}{}^n + \cdots}_{p}. \tag{28}$$

In Figure 10, the MAE increases as p increases. Same as the analysis of the NMED, for a 4×4 multiplier, when p changes from 3 to 4, the MAE sharply increases; for an 8×8 multiplier, when p changes from 10 to 11, the MAE sharply increases.

Various approximate compression schemes are studied in the next section for the design of the PPR circuitry so that it can be designed with suitable complement bits to meet specific accuracy constraints.

C. DESIGN OF MLACS

In this section, few approximate 4:2 compressors are designed based on 1-bit MLAFAs (the inputs are P_5, P_4, P_3, P_2, P_1 and the outputs are Sum , C_{out}, $Carry$) and a K-Map simplification (the inputs are P_5, P_4, P_3, P_2, P_1 and the outputs are Sum, $Carry$), respectively.

1) MLACS BASED ON 1-BIT MLAFAS

In 4:2 compressors, there are two full adders, referred to as models 1 and 2 from upside to downside (as defined in [16]). 1-bit MLAFAs are used to replace the exact versions. Six different designs are investigated by employing through various combinations of MLAFA1 and MLAFA2 (Figure 11). For MLAC21, module 1 utilizes MLAFA2 while the other uses MLAFA1. MLAC11 has been proposed in [15]; there are two schemes when MLAFA2 is employed as module 2. So, MLAC22-1 and MLAC12-1 employ the input of the compressor as *Carry* and use its negation to calculate *Sum*, while

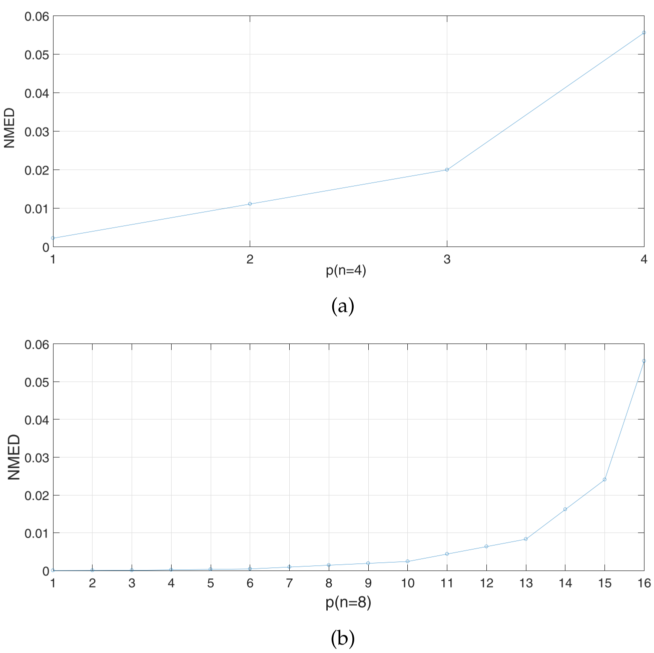

FIGURE 9. Complement bit selection (NMED versus p): (a) 4×4 MLAM, and (b) 8×8 MLAM.

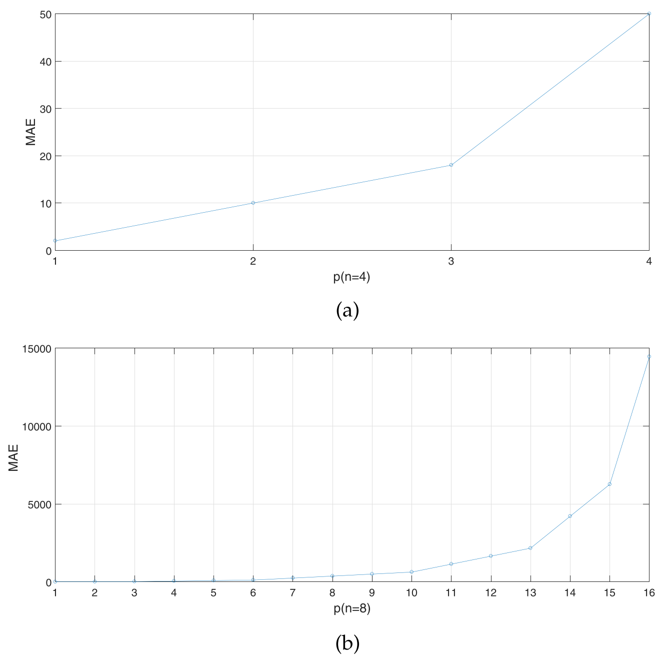

FIGURE 10. Complement bit selection (MAE versus p): (a) 4×4 MLAM, and (b) 8×8 MLAM.

IEEE TRANSACTIONS ON
EMERGING TOPICS
IN COMPUTING

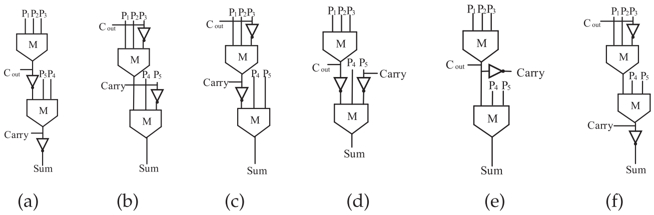

FIGURE 11. Schematic diagrams of proposed approximate 4:2 compressor: (a) MLAC11 (MLAC2 [15]), (b) MLAC22-1, (c) MLAC22-2, (d) MLAC12-1, (e) MLAC12-2, and (f) MLAC21.

TABLE 9. K-Map of proposed MLAC4 after step 1.

$P_5\,P_4$ \ $P_3\,P_2\,P_1$	000	001	011	010	110	111	101	100
00	00	01	10	01	10	11	10	01
01	01	10	11	10	11	11	11	10
11	10	11	11	11	11	11	11	11
10	01	10	11	10	11	11	11	10

MLAC22-2 and MLAC12-2 employ the output of module 1 as *Carry*.

2) MLAC BASED ON K-MAP SIMPLIFICATION

To further decrease the hardware complexity, an additional MLAC, namely MLAC4, is proposed; this circuit uses 2 outputs (rather than 3) for the final result. The design of this MLAC is accomplished as per the following 3-step process:

Step 1: Approximation on the number of outputs. Use 2 outputs to denote the results so the binary 11 is employed to represent results larger than 11.

Step 2: Approximation of the expression for the K-map in Step 1. Depending on the properties of the majority logic, simplified equations can be found by introducing few errors.

Step 3: Computing the error distance by combining Step 1 and Step 2.

In Step 1, an approximate K-Map is found (Table 9); this step introduces 6 errors due to the shortened bit length. The final expression can be then obtained after further simplification in Step 2; this is shown in Table 10, leading to 7 additional errors, as in Eqs. (29) and (30). Figure 12 gives the schematic diagram.

TABLE 10. K-Map of proposed MLAC4 after step 2.

$P_5\,P_4$ \ $P_3\,P_2\,P_1$	000	001	011	010	110	111	101	100
00	00	10	10		10	10	10	
01	10	10	11	10	11	11	11	10
11	11	11	11	11	11	11	11	11
10	10	10	11	10	11	11	11	10

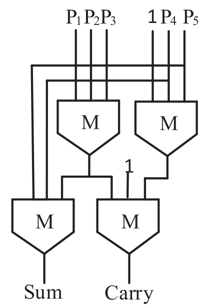

FIGURE 12. The schematic diagram of proposed MLAC4.

$$Carry = M(M(P_3, P_2, P_1), M(P_5, P_4, 1), 1) \tag{29}$$

$$Sum = M(M(P_3, P_2, P_1), P_5, P_4). \tag{30}$$

3) COMPARISON AND DISCUSSION OF MLACS

A comprehensive comparison of these MLACs is provided in Table 11. The designs based on 1-bit MLAFAs are similar, in terms of majority gate consumption and delay. In terms of NMED, the MED of MLAC2 [15], MLAC22-2, MLAC12-1 are 25 percent smaller than the remaining designs and their MAEs are 50 percent smaller than the remaining designs; moreover, the carry chain delays of MLAC22-2 and MLAC12-1 are shorter than MLAC2 [15]. The delay of MLAC12-1 is the shortest. The overall delay can be reduced by connecting the output to the input of the next units. Therefore, MLAC22-2 and MLAC12-1 show the best overall performance among these MLACs based on MLAFAs. Compared with the exact design, the proposed designs save 67 percent of majority gates, 50 percent of inverters, 50 percent of delay and 67 percent of carry chain. MLAC1 [14] requires no delay for carry chain; and a reduction of up to 25 percent in NMED can be achieved by the proposed designs.

TABLE 11. Comparison of 4:2 MLACs.

Types of Compressor		MV	INV	D	D_{carry}	MAE	NMED
Exact compressor		6	4	4	3	0	0
MLAC1 [14]		2	2	2	0	2	0.1
1-bit MLAFAs based	MLAC2[15]	2	2	2	2	1	0.075
	MLAC22-1	2	2	2	1	2	0.1
	MLAC22-2	2	2	2	1	1	0.075
	MLAC12-1	2	2	2	1	1	0.075
	MLAC12-2	2	1	2	1	2	0.1
	MLAC21	2	2	2	2	2	0.1
K-map simplification based	MLAC4	4	0	2	2	3	0.087

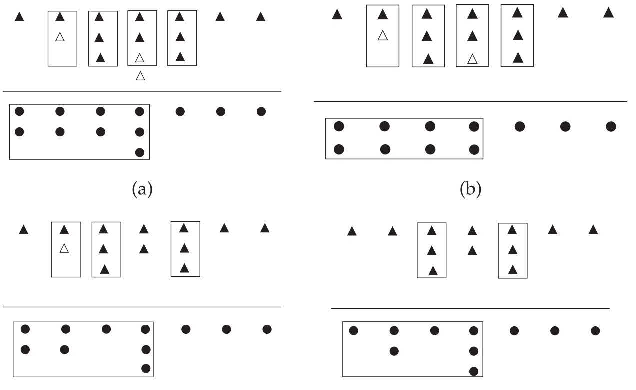

FIGURE 13. **Exact PPR circuitry design for 4×4 MLAMs: (a) MLAM-EC (p = 1), (b) MLAM-EC (p = 2), (c) MLAM-EC (p = 3), and (d) MLAM-EC (p = 4).**

For the design based on K-map simplification, compared with the above designs, the delay is slightly smaller and the error (NMED and MAE) is slightly larger; although MLAC4 requires two additional 3-input majority gates compared with the proposed MLACs based on 1-bit MLAFAs, it uses no inverter and only generates two outputs. Moreover, compared with the exact design, it saves 33 percent of the majority gates, all inverters, 50 percent of the delay and 33 percent of the carry chain.

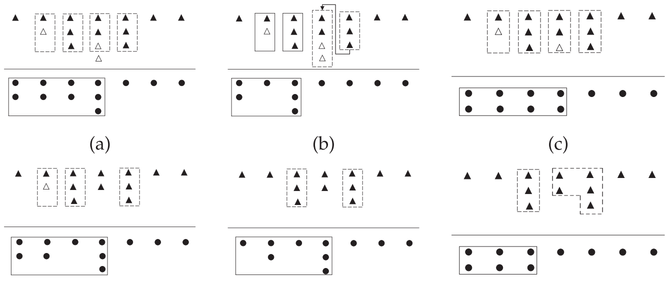

FIGURE 14. **Approximate PPR circuitry design for 4×4 MLAMs: (a) MLAM-AC1 (p = 1), (b) MLAM-AC2 (p = 1), (c) MLAM-AC (p = 2), (d) MLAM-AC2 (p = 3), (e) MLAM-AC1 (p = 4), and (f) MLAM-AC2 (p = 4).**

D. DESIGN OF MLAMS WITH PROPOSED PPR CIRCUITRY

During compression, this design uses the distribution of the PPs for compression to shorten the length of the critical path. By using a pipeline, an efficient design of the PPR circuitry is studied for 4×4 and 8×8 MLAM designs with different number of added complement bits for exact and approximate compression, respectively, to obtain only 2 rows (or a carry in the lowest order). In particular for an 8×8 MLAM design in addition to the approach mentioned above, there is yet another structure in which 4×4 MLAMs are used to design 8×8 MLAMs.

TABLE 12. **Comparison of 4×4 MLAMs.**

Types of 4×4 Multiplier				MV	INV	D	NMED(10^{-3})	MAE
Exact Multiplier		4×4 Array I based[23]		52	24	14	0	0
		4×4 Array II based[23]		64	32	14	0	0
		4×4 Wallace based[24]		52	24	10	0	0
		4×4 Dadda based[24]		52	24	12	0	0
Approximate Multiplier	Exact compression	4×4 MLAM-EC (p=1)		39	16	8	2.22	2
		4×4 MLAM-EC (p=2)		38	16	8	11.10	10
		4×4 MLAM-EC (p=3)		34	14	8	19.98	18
		4×4 MLAM-EC (p=4)		30	12	8	55.58	50
	Approximate compression	4×4 MLAM-AC1 (p=1)		31	12	7	28.61	40
		4×4 MLAM-AC2 (p=1)	MLAC4	35	11	7	18.89	18
			MLAC22-2	33	13	7	14.44	12
		4×4 MLAM-AC (p=2)		30	12	7	34.65	48
		4×4 MLAM-AC (p=3)		28	11	7	36.52	54
		4×4 MLAM-AC1 (p=4)		26	10	7	59.76	66
		4×4 MLAM-AC2 (p=4)		24	9	7	61.59	66

TABLE 13. **Comparison of 8×8 MLAMs using 4×4 MLAMs (Note that items include PPs and complement bits).**

Types of 8×8 Multiplier		Items Production		PPR Circuitry		Final Adder	D	NMED (10^{-2})	MAE
		MV	INV	MV	INV				
Exact compression	4×4 MLAM-AC1 (p=1)	124	48	24	16	11	21	2.7	11560
	4×4 MLAM-AC2 (p=1)-MLAC22-2	132	52	24	16	11	21	1.3	3468
	4×4 MLAM-AC (p=2)	120	48	24	16	11	21	3.2	13872
	4×4 MLAM-AC (p=3)	112	44	24	16	11	21	3.5	15606
	4×4 MLAM-AC2 (p=4)	96	36	24	16	11	21	5.8	19074
Approximate compression	4×4 MLAM-AC1 (p=1)	124	48	8	8	11	20	3.1	13602
	4×4 MLAM-AC2 (p=1)-MLAC22-2	132	52	8	8	11	20	1.9	6572
	4×4 MLAM-AC (p=2)	120	48	8	8	11	20	3.4	14334
	4×4 MLAM-AC (p=3)	112	44	8	8	11	20	3.6	17110
	4×4 MLAM-AC2 (p=4)	96	36	8	8	11	20	5.8	19226

IEEE TRANSACTIONS ON
EMERGING TOPICS IN COMPUTING

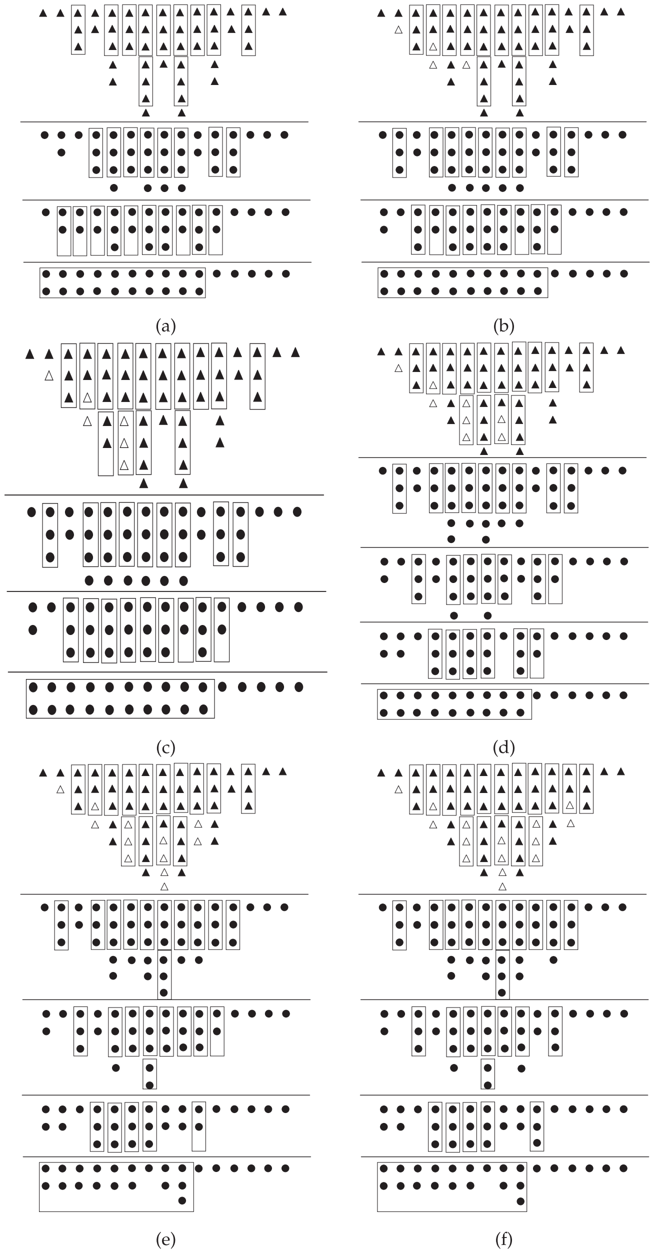

FIGURE 15. Exact PPR circuitry design for 8×8 MLAMs: (a) p = 16, (b) p = 12, (c) p = 10, (d) p = 8, (e) p = 4, and (f) p = 1.

1) 4×4 MLAMS

A. Exact Compression (EC)

1-bit adders are used for PP compression. Four different compression schemes are presented depending on the number of complement bits (which require one stage of compression as shown in Figure 13). The PPs within the solid line represent an arithmetic unit. The number of complement bits does not affect the critical path. Independently of the selection of the complement bits, a 4-bit adder is required to compute the final result. The difference between these four schemes is the number of 1-bit adders used for compression. Figures 13(a) and 13(b) require four 1-bit adders in Stage 1; Figure 13(c) needs three while Figure 13(d) needs only two.

B. Approximate Compression (AC)

To further reduce hardware and delay, the proposed MLAFAs and compressors can be utilized in the reduction circuitry. In Figures 14(a), 14(c), 14(d), and 14(e), the exact adders can be changed to approximate adders for approximate compression; the PPs circled by dotted lines indicate the approximate arithmetic units.

The proposed compressors are employed when p is 1; an approximate compressor whose inputs come from the output of 1-bit MLAFA, is shown in Figure 14(b). When considering the delay, the PPs on the right side of the middle line are better compressed by the arithmetic circuits (they have the same delay as the left compression circuits). Two 1-bit EFAs (rather than a 2-bit MLAFA) are selected to better control the errors; if an approximate compressor with 2 outputs is employed, a 3-bit half adder is required for the final result. Otherwise, a 3-bit full adder is required when using an approximate compressor with 3 outputs. However when p increases, the errors increase; consequently, a further approximation is proposed for larger values of p. When p is 4, Figure 14(f) shows another method for compression (i.e., to replace one of the 1-bit MLAFAs of lower weight with a 2-bit MLAFA).

C. Comparison and Discussion of 4×4 MLAMs

Table 12 gives the comparison between various PPR circuitry designs and the exact designs from [23], [24]. The proposed MLAFA2, MLAC22-2, MLAC21 or MLAC4 are used for approximate compression; approximate compression reduces the accuracy but it decreases the hardware overhead compared with an exact compression. Approximate compression is less pronounced when p is larger. For example, when p is 1, approximate compression significantly affects the NMED and the MAE; however, when p is 4, approximate compression increases the NMED by 7.5 percent and the MAE 24 percent only. As mentioned previously, when p is 4, the use of 4×4 MLAM-AC2 (p = 4) results in a modest NMED and MAE increase. This implies that for an application that can tolerate relatively large errors, a larger approximation can be used.

When p is equal to 1, the proposed MLACs show excellent performance so improving the overall accuracy while adding just few majority gates. When comparing 4×4 MLAM-EC (p = 3) with 4×4 MLAM-AC2 (p = 1), although more complement bits are needed for 4×4 MLAM-AC2 (p = 1), they not only introduce fewer errors, but also incur in less hardware overhead and delay. Compared with the exact designs from [23], [24], the proposed design significantly reduces the hardware overhead and delay. For example, 4×4 MLAM-AC2 (p = 1) which uses MLAC22-2 saves at least 37 percent of the number of majority gates, 46 percent of the number of inverters and 50 percent of delay.

2) 8×8 MLAMS

A. 8×8 MLAMs Using 4×4 MLAMs

8×8 MLAMs can be designed using 4×4 MLAMs as a module, so generating 4 rows of PPs. Independently of the selection of the complement bits and PPR circuitry, 4×4 MLAMs generate an 8-bit result. Therefore, the PPs can be

TABLE 14. Comparison of 8×8 MLAMs using 2×2 MLAMs with different number of complement bits (Note that items include PPs and complement bits).

Types of 8×8 Multiplier	Items Production MVs	PPR Circuitry		Final Adder	D	NMED	MAE
		FAs	HAs				
8×8 MLAM (p = 16)	48	23	6	10	18	5.54×10^{-2}	14,450
8×8 MLAM (p = 15)	49	24	5	10	18	2.41×10^{-2}	6,258
8×8 MLAM (p = 14)	50	26	4	10	18	1.62×10^{-2}	4,210
8×8 MLAM (p = 13)	51	26	4	10	18	8.31×10^{-3}	2,162
8×8 MLAM (p = 12)	52	27	3	10	18	6.34×10^{-3}	1,650
8×8 MLAM (p = 11)	53	28	4	10	18	4.37×10^{-3}	1,138
8×8 MLAM (p = 10)	54	29	3	10	18	2.41×10^{-3}	626
8×8 MLAM (p = 9)	55	32	4	9	18	1.91×10^{-3}	498
8×8 MLAM (p = 8)	56	34	2	9	18	1.42×10^{-3}	370
8×8 MLAM (p = 7)	57	34	3	9	18	9.30×10^{-4}	242
8×8 MLAM (p = 6)	58	35	3	9	18	4.38×10^{-4}	114
8×8 MLAM (p = 5)	59	36	2	9	18	3.15×10^{-4}	82
8×8 MLAM (p = 4)	60	36	3	9	18	1.92×10^{-4}	50
8×8 MLAM (p = 3)	61	37	2	9	18	6.91×10^{-5}	18
8×8 MLAM (p = 2)	62	38	2	9	18	3.84×10^{-5}	10
8×8 MLAM (p = 1)	63	39	1	9	18	7.69×10^{-6}	2

compressed by utilizing eight 1-bit full adders to generate 2 lines requiring an additional 11-bit full adder. To further reduce the hardware, these eight 1-bit full adders can be replaced by approximate adders too.

Four 4×4 MLAMs are used so 4×4 combinations (based on the number of complement bits) are possible; however, as in Section IV-B, complement bits of the MSBs can be omitted or reserved for the LSBs in this scheme, as reduction of hardware is not the primary objective. Only the cases of using the same type of 4×4 MLAM with approximate compression are presented because they have better performance according to the previous discussion; so 4×4 MLAM-AC2 (p = 1) using MLAC22-2 and 4×4 MLAM-AC2 (p = 4) are selected.

As shown in Table 13, when p is more than 3, approximate compression has little influence on the NMED; when p is equal to 1, the multiplier with approximate compression made of 4×4 MLAM-AC2 (p = 1) shows better performance than the one with exact compression made by 4×4 MLAM-AC1 (p = 1) in terms of NMED, MAE, hardware as well as delay. Compared with the multiplier with the exact compression made of 4×4 MLAM-AC (p = 2) or 4×4 MLAM-AC (p = 3) with approximate compression made of 4×4 MLAM-AC1 (p = 1), the latter has better performance than all other cases.

B. 8×8 MLAMs Using 2×2 MLAMs

Compared with using 4×4 MLAMs as a module, the design using a 2×2 MLAM as a module can generate all PPs at once, so requiring fewer clock cycles in execution. Depending on the selection of the complement bits, different PPR circuitry designs are proposed (Figure 15). Only 1-bit adders are considered in all cases. From Table 14, the NMED and the MAE are consistent with the analysis above. Ry decreasing p, the required hardware increases; when p is larger than 10, the compression just needs 3 stages. Else, 4 stages are necessary for compression; however, the delay is not affected by the number of complement bits.

As discussed, the approximate PPR circuitry using the proposed MLAFA2 and MLACs is assessed at p = 10. Compression is analyzed using MLACs with 3 outputs and the new design using MLACs with 2 outputs (Figure 16). The red connecting line denotes the critical path in the first compression stage.

Table 15 shows the comparison of 8×8 MLAMs with an approximate PPR circuitry design. Compared with an exact compression as discussed previously, hardware and delay have been significantly reduced. Approximate compression not only results in a smaller delay, but also in a saving of more than 13 percent in the number of majority gates with only a small loss in accuracy. Furthermore, the use of MLAC4 not only reduces the required hardware, but it also incurs in a smaller inaccuracy because a smaller number of approximate operations is performed, so showing the best overall performance.

C. Comparison and Discussion of 8×8 MLAMs

The best designs are selected and compared with the exact designs from [23] (Table 16). The 4×4 MLAM-AC2 (p = 1) using MLAC22-2 reduces by at least 25 percent the number of majority gates, by 27 percent the number of inverters and by 33 percent the delay, compared with the exact designs. The designs based on 4×4 multipliers are not as good as the designs using 2×2 multipliers. As for the designs using 2×2 multipliers, a significant decrease of hardware is achieved without incurring in large errors. The MLAC4 based design using 2×2 multipliers reduces the number of majority gates by up to 48 percent, the number of inverters by up to 67 percent, and the delay by up to 47 percent; the MLAC22-2 based design using 2×2 multipliers reduces the number of majority gates by up to 50 percent, the number of inverters up to 53 percent, and the delay up to 47 percent.

To further verify the feasibility of the proposed designs using QCA as an emerging technology, a MLAC4 based approximate multiplier design has been implemented using QCADesigner (multi-layer crossing is used and the following

IEEE TRANSACTIONS ON
EMERGING TOPICS IN COMPUTING

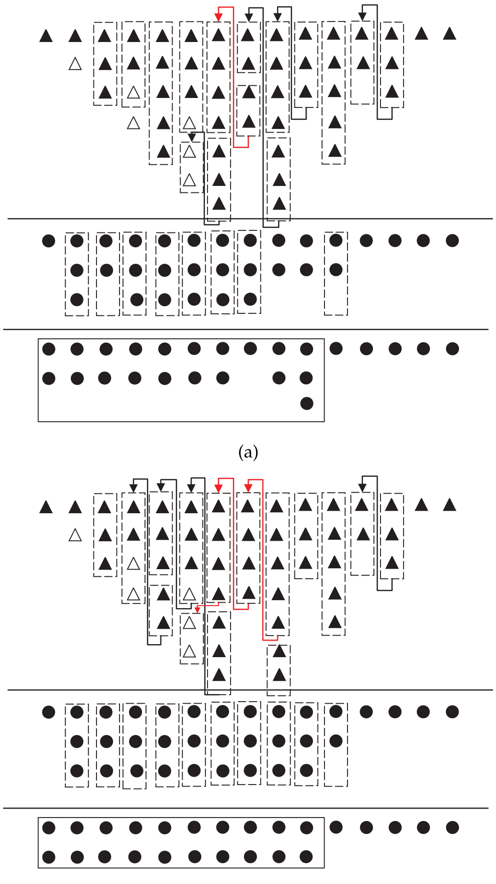

FIGURE 16. Approximate PPR circuitry design for 8×8 MLAMs: (a) MLAC (2 outputs) based (p = 10), and (b) MLAC (3 outputs) based (p = 10).

parameters have been used for the coherence vector simulation engine: Number of Samples: 128,000; Convergence Tolerance: 0.00001; Radius of Effect: 55 nm. The rest of the parameters are set as the default values). Table 17 shows the comparison between the 8-bit MLAC4 based approximate multiplier design and the latest CMOS based approximate multipliers (using 45 nm technology) [25]; note that there is no direct comparison between majority logic based arithmetic circuits and non-majority based arithmetic circuits.

TABLE 17. Results of 8×8 approximate multipliesr (Majority based design versus CMOS based design).

Types	Delay	Area (μm^2)
MLAC4 based	20 clock zones	19.57
R4ABM1(p = 8)[25]	0.58 ns	581.7
R4ABM2(p = 8)[25]	0.58 ns	538.6

E. IMAGE PROCESSING APPLICATION WITH MLAMS

The proposed 8×8 MLAMs are applied to image processing; the multipliers are used to multiply the same two images on a pixel basis so combining the two input images into a single output image. In this section, the impact of MLAMs is assessed with different numbers of complement bits as well as the proposed MLACs and approximate PPR circuit.

1) 8×8 MLAMS WITH DIFFERENT NUMBERS OF COMPLEMENT BITS

As shown in Figure 17, when the number of complement bits changes, the generated image will not change dramatically. Table 18 show that even when p is equal to 16, the processed image retains a high quality (the SSIMs are all above 0.95 and the PSNRs are all above 45dB). Although small changes occur for a different value; when p changes from 10 to 12, then a relatively sharp decrease of the SSIM occurs. To ensure a reasonable accuracy using approximate compression, it is better to choose a value for p smaller than 10. This is consistent with the discussion in Section IV-B.

TABLE 15. Comparison of 8×8 MLAMs using 2×2 MLAMs with approximate PPR circuit (p = 10) (Note that items include PPs and complement bits).

Types of 8×8 Multiplier	Items Production	PPR Circuitry		Final Adder	D	NMED	MAE
	MV	AFAs	ACs				
MLAC4 based	54	16	5	10	16	0.0267	9,120
MLAC22-2 based	54	18	7	10	16	0.0318	9,476
MLAC12-1 based	54	18	7	10	16	0.0346	9,820

TABLE 16. Comparison of 8×8 MLAMs.

Types of 8×8 Multiplier			MV	INV	D	NMED(10^{-2})	MAE
Exact Multiplier		8×8 Array I based[23]	232	112	30	0	0
		8×8 Array II based[23]	256	128	30	0	0
		8×8 Wallace based[23]	256	128	44	0	0
		8×8 Dadda based[23]	232	112	47	0	0
Approximate Multipliers	Using 4×4 Multipliers (Approximate Compression)	MLAM-AC2 (p=1) MLAC22-2	173	82	20	0.0191	6572
		MLAM-AC (p=2)	128	56	20	0.0342	14334
	Using 2×2 Multipliers(p=10) (Approximate Compression)	MLAC4 based	120	36	16	0.0267	9120
		MLAC22-2 based	116	52	16	0.0318	9476

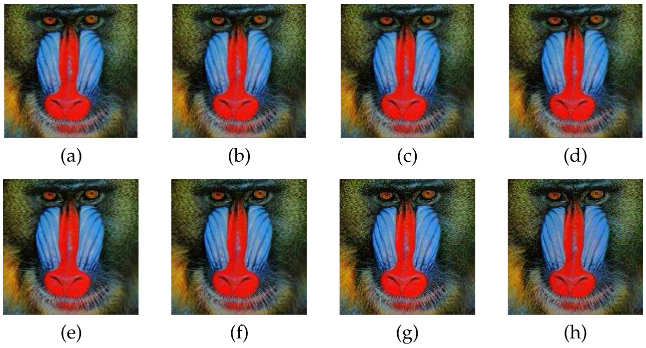

FIGURE 17. **Image processing of 8×8 MLAMs with different numbers of complement bits: (a) p = 2, (b) p = 4, (c) p = 6, (d) p = 8, (e) p = 10, (f) p = 12, (g) p = 14, and (h) p = 16.**

TABLE 18. **Image processing results of 8×8 MLAMs with different numbers of complement bits.**

Value of P	SSIM	PSNR(dB)
2	1	69.93
4	0.9999	62.61
6	0.9997	60.04
8	0.9994	56.17
10	0.9990	54.74
12	0.9954	51.77
14	0.9861	48.47
16	0.9597	45.08

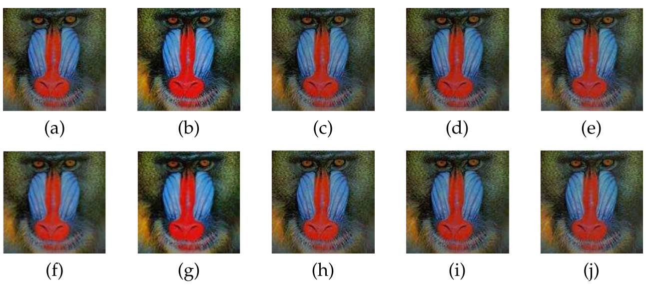

FIGURE 18. **Image processing of 8×8 MLAMs using 4×4 MLAMs as a module: exact compression based (a) 4×4 MLAM-AC1 (p = 1), (b) 4×4 MLAM-AC2 (p = 1)-MLAC22-2, (c) 4×4 MLAM-AC (p = 2), (d) 4×4 MLAM-AC (p = 3); approximate compression based (f) 4×4 MLAM-AC1 (p = 1), (g) 4×4 MLAM-AC2 (p = 1)-MLAC22-2, (h) 4×4 MLAM-AC (p = 2), (i) 4×4 MLAM-AC (p = 3), and (j) 4×4 MLAM-AC (p = 4).**

2) 8×8 MLAMS WITH PPR CIRCUITRY

A. 8×8 MLAMs Using 4×4 MLAMs

When 4×4 MLAMs are used as a module to assemble 8×8 MLAMs, the results of the PSNR, and SSIM indicate that there is no significant difference between exact compressions and approximate compressions (Figure 18 and Table 19). Therefore, approximate compression and a value of p smaller than 3 are preferred to further reduce the required hardware. Accordingly, the proposed MLACs can be utilized in image processing applications for reduced delay and power dissipation at a low inaccuracy with a SSIM of up to 0.97 and a PSNR of up to 48dB.

TABLE 19. **Image Processing Results of 8×8 MLAMs Using 4×4 MLAMs as a Module.**

Types of 8×8 Multiplier		SSIM	PSNR(dB)
Exact Compression	MLAM-AC1(p = 1)	0.8880	37.40
	MLAM-AC2 (p = 1)-MLAC22-2	0.9782	48.36
	MLAM-AC (p = 2)	0.8868	36.67
	MLAM-AC (p = 3)	0.8718	36.36
	MLAM-AC2 (p = 4)	0.8623	35.68
Approximate Compression	MLAM-AC1(p = 1)	0.8879	37.48
	MLAM-AC2 (p = 1)-MLAC22-2	0.9753	47.52
	MLAM-AC (p = 2)	0.8802	36.54
	MLAM-AC (p = 3)	0.8632	36.32
	MLAM-AC2 (p = 4)	0.8472	35.44

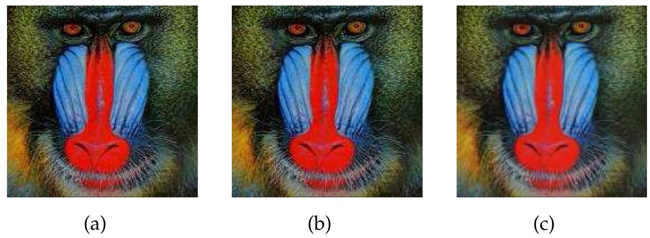

FIGURE 19. **Image processing of 8×8 MLAMs using 2×2 MLAMs as a module: (a) MLAC4 based, (b) MLAC22-2 based, and (c) MLAC12-1 based.**

TABLE 20. **Image processing results of 8×8 MLAMs using 2×2 MLAMs as a module.**

Types of 8×8 Multiplier	SSIM	PSNR(dB)
MLAC4 based	0.9759	47.33
MLAC22-2 based	0.9614	45.17
MLAC12-1 based	0.8971	41.31

B. 8×8 MLAMs Using 2×2 MLAMs

The use of 2×2 MLAMs as a module for 8×8 MLAMs as an approximate compression designs shows excellent performance, with a SSIM of at least 0.89 and the PSNR of at least 41 dB (Figure 19 and Table 20). When comparing the MLAC22-2 and MLAC12-1 based designs, MLAC22-2 improves by 7 percent in terms of SSIM and 9 percent in terms of PSNR. The MLAC4 based design is superior to others with improvements in SSIM and PSNR, so consistent with the original image. Compared with other designs, the utilization of a 2×2 approximate multiplier as a module and the proposed approximate compression result in the best performance.

V. CONCLUSION

This paper has presented a design, analysis and evaluation of majority logic based approximate adders and approximate multipliers. ML based 1-bit, 2-bit and multi-bit AFAs have been proposed; these deigns have a reduced circuit complexity and reduced delay compared to the exact counterpart while only incurring in a modest loss in accuracy. By combining multiple approximate techniques (such as the proposed MLACs and approximate PPR circuitry) with the so-called complement bits, ML based multi-bit AMs have been proposed: an influence factor has been defined to measure the importance of different complement bits; selection of the complement bits has also been pursued by an in-depth analysis depending on the size of multipliers; multiple MLACs has been proposed based on MLAFAs or K-Map simplification, and has been employed in the approximate PPR circuitry design for 8×8 MLAMs. The following conclusions can be drawn.

1) By combining the proposed 1-bit MLAFA and the existing 1-bit MLAFA, multi-bit MLAFAs can be designed with a relatively large error. The proposed MLAFA33 designed by truth table reduction can provide the solution for improving accuracy. For 8-bit adders, MLAFA1212-1233 (with a reduction of 58 percent in majority gates as well as 50 percent in delay) and MLAFA1212-3333 (with a reduction of 50 percent in majority gates as well as 50 percent in delay) are superior to other designs.
2) Based on the proposed 2×2 MLAM, we can selectively design multi-bit multipliers by adding the complement bits. From the presented theoretical analysis, for a 4×4 multiplier, when p changes from 3 to 4, the NMED increases sharply; for an 8×8 multiplier, when p is smaller than 10, the NMED increases slowly but when p is larger than 10, the NMED increases rapidly. So p = 3 and p = 10 are the best choices for the 4×4 multiplier and the 8×8 multiplier, respectively.
3) MLAC22-2 and MLAC12-1 show the best overall performance among these MLACs based on MLAFAs. Compared with the exact design, the proposed designs save 67 percent of majority gates, 50 percent of inverters, 50 percent of delay and 67 percent of carry chain. Although MLAC4 requires two additional 3-input majority gates, MLAC4 only has 2 outputs, which is likely to reduce the approximate operation in the PPR circuitry. The MLAC4 based 8×8 multiplier using 2×2 multipliers (p = 10) reduces the number of majority gates by up to 48 percent, the number of inverters by up to 67 percent, and the delay by up to 47 percent compared to the exact counterpart, achieving a significant decrease of hardware without incurring in large errors.

The proposed approximate adders and multipliers have been shown to be good for applications requiring low inaccuracy and high speed. Case studies of error-resilient applications have shown the validity of the proposed designs.

ACKNOWLEDGMENTS

This work is supported by a grant from the National Natural Science Foundation China (No. 61871216), Fundamental Research Funds for the Central Universities China (NE2019102), Six Talent Peaks Project in Jiangsu Province (XYDXX-009) and the US National Science Foundation (No. 1812467).

REFERENCES

[1] S. L. Lu, "Speeding up processing with approximation circuits," *Comput.*, vol. 37, no. 3, pp. 67–73, 2004.
[2] J. Han and M. Orshansky, "Approximate computing: An emerging paradigm for energy-efficient design," in *Proc. Eur. Test Symp.*, 2013, pp. 1–6.
[3] S. Hashemi, R. Bahar, and S. Reda, "DRUM: A dynamic range unbiased multiplier for approximate applications," in *Proc. IEEE/ACM Int. Conf. Comput. Des.*, 2015, pp. 418–425.
[4] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 32, no. 1, pp. 124–137, Jan. 2013.
[5] S. Rehman, W. El-Harouni, M. Shafique, A. Kumar, and J. Henkel, "Architectural-space exploration of approximate multipliers," in *Proc. Int. Conf. Comput.-Aided Des.*, 2016, pp. 1–6.
[6] 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, Apr. 2015.
[7] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," *IEEE Trans. Comput.*, vol. 62, no. 9, pp. 1760–1771, Sep. 2013.
[8] K. Walus and G. Jullien, "Design tools for an emerging SoC technology: Quantum-dot cellular automata," *Proc. IEEE*, vol. 94, no. 6, pp. 1225–1244, Jun. 2006.
[9] C. Lent and P. Tougaw, "A device architecture for computing with quantum dots," *Proc. IEEE*, vol. 85, no. 4, pp. 541–557, Apr. 1997.
[10] M. Vacca, M. Graziano, J. Wang, F. Cairo, G. Causapruno, G. Urgese, A. Biroli, and M. Zamboni, "NanoMagnet logic: An architectural level overview," in *Field-Coupled Nanocomputing*, Berlin, Germany: Springer, 2014, pp. 223–256.
[11] A. Khitun and K. L. Wang, "Nano scale computational architectures with spin wave bus," *Superlattices Microstructures*, vol. 38, no. 3, pp. 184–200, 2005.
[12] C. Labrado, H. Thapliyal, and F. Lombardi, "Design of majority logic based approximate arithmetic circuits," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2017, pp. 2122–2125.
[13] T. Zhang, W. Liu, E. McLarnon, M. O'Neill, and F. Lombardi, "Design of majority logic (ML) based approximate full adders," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2018, pp. 1–5.
[14] M. H. Moaiyeri, F. Sabetzadeh, and S. Angizi, "An efficient majority-based compressor for approximate computing in the nano era," *Microsyst. Technol.*, vol. 4, pp. 1–13, 2017.
[15] S. Angizi, H. Jiang, R. F. DeMara, J. Han, and D. Fan, "Majority-based spin-CMOS primitives for approximate computing," *IEEE Trans. Nanotechnol.*, vol. 17, no. 4, pp. 795–806, Jul. 2018.
[16] C. Chang, J. Gu, and M. Zhang, "Ultra low-voltage low-power CMOS 4–2 and 5-2 compressors for fast arithmetic circuits," *IEEE Trans. Circuits Syst. I*, vol. 51, no. 10, pp. 1985–1997, Oct. 2004.
[17] V. Pudi, K. Sridharan, and F. Lombardi, "Majority logic formulations for parallel adder designs at reduced delay and circuit complexity," *IEEE Trans. Comput.*, vol. 66, no. 10, pp. 1824–1830, Oct. 2017.
[18] H. Cho and E. E. Swartzlander, "Adder and multiplier designs in quantum dot cellular automata," *IEEE Trans. Comput.*, vol. 58, no. 6, pp. 721–727, Jun. 2009.
[19] V. Pudi and K. Sridharan, "Low complexity design of ripple carry and Brent-Kung adders in QCA," *IEEE Trans. Nanotechnol.*, vol. 11, no. 1, pp. 105–119, Jan. 2012.
[20] S. Mittal, "A survey of techniques for approximate computing," *ACM Comput. Surv.*, vol. 48, no. 4, 2016, Art. no. 62.
[21] S. L. Eddins and M. T. Orchard, "Using MATLAB and C in an image processing lab course," in *Proc. First Int. Conf. Image Process.*, 1994, vol. 1, pp. 515–519.
[22] P. Kulkarni, P. Gupta, and M. Ercegovac, "Trading accuracy for power with an underdesigned multiplier architecture," *Proc. 24th Int. Conf. VLSI Des.*, 2011, pp. 346–351.
[23] S. W. Kim and E. E. Swartzlander, "Multipliers with coplanar crossings for quantum-dot cellular automata," in *Proc. 10th IEEE Conf. Nanotechnol.*, 2010, pp. 953–957.
[24] S. W. Kim and E. E. Swartzlander, "Parallel multipliers for quantum-dot cellular automata," in *Proc. Nanotechnol. Materials Devices Conf.*, 2009, pp. 68–72.
[25] 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, no. 8, pp. 1435–1441, Aug. 2017.

WEIQIANG LIU (S'90-M12-SM15) received the BSc degree in information engineering from the Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China, and the PhD degree in electronic engineering from the Queen's University Belfast (QUB), Belfast, United Kingdom, 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 and the leader of Emerging Integrated Circuit and System (EICAS) Group. He has published one research book by Artech House and more than 70 leading journal and conference papers. He serves as an associate editor of the *IEEE Transactions on Computers (TC)* and the *IEEE Transactions on Emerging Topics in Computing (TETC)*, a Steering Committee member of the *IEEE Transactions on Multi-Scale Computing Systems (TMSCS)*, the guest editors of the *IEEE Transactions on Emerging Topics in Computing* and the *Microelectronics Journal*. He has been a program committee member for several conferences including ARITH, ISCAS, ASAP, ISVLSI, NANOARCH, AisaHOST, SiPS and ICONIP. He is a member of both CASCOM and VSA Technical Committee, IEEE CAS Society. His research interests include computer arithmetic, approximate computing, emerging technologies in computing systems and hardware security. He is a senior member of the IEEE.

TINGTING ZHANG received the BSc degree in electronic information science and technology from the Nanjing University of Aeronautics and Astronautics (NUAA), Nanjing, China, 2016, and the MEng degree in circuits and system from NUAA, Nanjing, China, in 2019. She is currently a research assistant with the Emerging Integrated Circuit and System (EICAS) Group, NUAA. Her research interests include computer arithmetic, approximate computing, and emerging technologies in computing systems.

EMMA MCLARNON received the graduate degree from Queen's University, Belfast, in 2011, with a joint honours in mathematics, statistics and operational research. She began researching in the area of Quantum-dot Cellular Automata, in 2013, focusing on power dissipation and its impacts on the security of QCA devices, as well as design and implementation of QCA adders and multipliers, along with bit erasure analysis. Her research interests include post-CMOS nanotechologies, quantum computing and physical unclonable functions.

MAIRE O'NEILL (M'03-SM'11) received the MEng degree with distinction and the PhD degree in electrical and electronic engineering from Queen's University Belfast (QUB), in 1999 and 2002, respectively. She is currently research director of the Centre for Secure Information Technologies (CSIT) at QUB. She previously held a prestigious UK Engineering and Physical Sciences Research Council (EPSRC) Leadership Fellowship (2008-2015) and was a former holder of a UK Royal Academy of Engineering (RAEng) research fellowship (2003-2008). She has received numerous awards for her research to date which include a 2014 UK Royal Academy of Engineering Silver Medal, the Womens Engineering Society (WES) prize at the 2006 IET Young Woman Engineer of the Year awards and she was named British Female Inventor of the Year, in 2007. She has authored two research books and has more than 125 international peer-reviewed conference and journal publications. She is associate editor for the *IEEE Transactions on Computers* and the *IEEE Transactions on Emerging Topics in Computing* and has acted as guest editor for a number of journals. She has been a technical program committee member for many international conferences, including DAC, CHES, DATE, SOCC, and ISCAS. She is an IEEE Circuits & Systems for Communications (CASCOM) Technical committee member and was treasurer of the Executive Committee of the IEEE United Kingdom and Ireland (UKRI) Section, 2008-09. She is a member of the Royal Irish Academy and a Fellow of the Irish Academy of Engineering. She is a senior member of the IEEE.

PAOLO MONTUSCHI (M'90-SM'07-F'14) is a full professor with the Department of Control and Computer Engineering and a member of the Board of Governors (BoG) at Politecnico di Torino. Previously, he served as chair of the Department from 2003 to 2011. His research interests include computer arithmetic and architectures, computer graphics, electronic publications, semantics and education. He is an IEEE fellow, an IEEE Computer Society (CS) Golden Core member, a recipient of the "Distinguished Service" and the "Spirit of the Computer Society" awards. He is serving as the acting Editor-in-Chief of the *IEEE Transactions on Emerging Topics in Computing* (2019), as the chair of the CS Awards Committee (2017-19), and as a member of the IEEE Publications Services and Products Board (2018-20). He was the Editor-in-Chief of the *IEEE Transactions on Computers* (2015-18). Previously, he served in a number of Boards and Committees, including the IEEE Products and Services Committee and the BoG of the CS. He is a life member of the International Academy of Sciences of Turin and of Eta Kappa Nu (the Honor Society of IEEE). In March 2017, he co-founded the first HKN Student Chapter in Italy.

FABRIZIO LOMBARDI (M81-SM02-F'09) received the graduate degree from the University of Essex, United Kingdom, with a BSc (Hons.) degree in electronic engineering, in 1977. In 1977, he joined the Microwave Research Unit at University College London, where he received the master's degree in Microwaves and Modern Optics, in 1978, the Diploma degree in microwave engineering, in 1978 and the PhD degree from the University of London, in 1982. He is currently the holder of the International Test Conference (ITC) Endowed chair Professorship at Northeastern University, Boston. In the past he 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*. Since 2019, he serves 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. He has extensively published in these areas and coauthored/edited seven books. He is a fellow of the IEEE.