ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/322750168

A Low-Power High-Speed Accuracy-Controllable Approximate Multiplier Design

Conference Paper · January 2018

DOI: 10.1109/ASPDAC.2018.8297389

CITATIONS
2

READS
431

3 authors:

Tongxin Yang
Fukuoka University
13 PUBLICATIONS **12** CITATIONS
SEE PROFILE

Tomoaki Ukezono
Fukuoka University
17 PUBLICATIONS **13** CITATIONS
SEE PROFILE

Toshinori Sato
Fukuoka University
181 PUBLICATIONS **775** CITATIONS
SEE PROFILE

Some of the authors of this publication are also working on these related projects:

Approximate arithmetic circuits View project

End of Life IC re-making View project

All content following this page was uploaded by Tongxin Yang on 28 January 2018.

The user has requested enhancement of the downloaded file.

A Low-Power High-Speed Accuracy-Controllable Approximate Multiplier Design

Tongxin Yang[1] Tomoaki Ukezono[2] Toshinori Sato[3]
[1] Graduate School of Information and Control Systems, Fukuoka University, Japan
[2,3]Department of Electronics Engineering and Computer Science, Fukuoka University, Japan
[1]Email: td166502@cis.fukuoka-u.ac.jp [2]Email: tukezo@fukuoka-u.ac.jp [3]Email: toshinori.sato@computer.org

***Abstract*—Multiplication is a key fundamental function for many error-tolerant applications. Approximate multiplication is considered to be an efficient technique for trading off energy against performance and accuracy. This paper proposes an accuracy-controllable multiplier whose final product is generated by a carry-maskable adder. The proposed scheme can dynamically select the length of the carry propagation to satisfy the accuracy requirements flexibly. The partial product tree of the multiplier is approximated by the proposed tree compressor. An 8×8 multiplier design is implemented by employing the carry-maskable adder and the compressor. Compared with a conventional Wallace tree multiplier, the proposed multiplier reduced power consumption by between 47.3% and 56.2% and critical path delay by between 29.9% and 60.5%, depending on the required accuracy. Its silicon area was also 44.6% smaller. In addition, results from an image processing application demonstrate that the quality of the processed images can be controlled by the proposed multiplier design.**

I. INTRODUCTION

Many increasingly popular applications, such as image processing and recognition, are inherently tolerant of small inaccuracies. These applications are computationally demanding and multiplication is their fundamental arithmetic function, which creates an opportunity to trade off computational accuracy for reduced power consumption. Approximate computing is an efficient approach for error-tolerant applications because it can trade off accuracy for power, and it currently plays an important role in such application domains [1].

Different error-tolerant applications have different accuracy requirements, as do different program phases in an application. If multiplication accuracy is fixed, power will be wasted when high accuracy is not required. This means that approximate multipliers should be dynamically reconfigurable to match the different accuracy requirements of different program phases and applications.

This paper focuses on an approximate multiplier design that can control accuracy dynamically. A carry-maskable adder (CMA) is proposed that can be dynamically configured to function as a conventional carry propagation adder (CPA), a set of bit-parallel OR gates, or a combination of the two. This configurability is realized by masking carry propagation: the CPA in the last stage of the multiplier is replaced by the proposed CMA. An approximate tree compressor is utilized to reduce the accumulation layer depth of the partial product tree. Our approach introduces a term representing the power and accuracy requirements which simplifies the partial product reduction (PPR) component as needed. An approximate multiplier is designed using the proposed adder and compressor. This multiplier, together with a conventional multiplier and the previously studied approximate multipliers, was implemented in Verilog HDL using a 45-nm library to evaluate the power consumption, critical path delay, and design area. Compared with the conventional Wallace tree multiplier, the proposed approximate multiplier reduced power consumption by between 47.3% and 56.2% and the critical path delay by between 29.9% and 60.5%, depending on the required computational accuracy. In addition, its design area was 44.6% smaller. Comparisons with the established approximate multipliers, none of which have any dynamic reconfigurability, demonstrate that the proposed multiplier provided the best trade-off of power and delay against accuracy. All the multiplier designs are then evaluated in a real image processing application.

The remainder of this paper is organized as follows. Section II reviews previous works. Section III introduces the accuracy-controllable approximate multiplier after explaining the tree compressor and the CMA. Section IV evaluates the multipliers experimentally and then evaluates the proposed approximate multiplier using an image processing application. Section V presents our conclusions.

II. PREVIOUS WORK

The adder is a basic element of most multipliers. Mahdiani et al. [2] proposed the lower-part-OR adder, which utilizes OR gates for addition of the lower bits and precise adders for addition of the upper bits. It is similar to our proposed CMA in that it uses OR gates to generate the sum approximately, but our CMA is also dynamically reconfigurable.

Liu et al. [3] utilized an approximate adder to reduce carry propagation delay in partial product accumulation. They also proposed a recovery vector to improve accuracy. The bit width of the error recovery vector can be selected by the designer to satisfy accuracy requirements. Hashemi et al. [4] proposed a technique that reduces the size of the multiplier by detecting the leading one bit of the input operands and selecting the following k bits as abridged operands for both inputs, where k is a designer-defined value that specifies the bandwidth used in the core accurate multiplier. Both [3] and [4] allow a static trade-off between power consumption and accuracy. The bit lengths

of the recovery vector [3] and the input operands [4] are determined during the design process and the accuracy is not dynamically controllable, unlike with our proposed multiplier. Moons et al. [5] proposed a system-level technique that disables part of the combinational logic and reconfigures the pipelined registers and combinational logic. It can trade off accuracy for power dynamically by changing the numbers of pipeline stages and voltage-accuracy scaling modes. Our proposed multiplier also disables part of the combinational logic in the CPA to achieve lower power consumption, but ours does not require a pipeline system or control circuits for voltage scaling.

III. ACCURACY-CONTROLLABLE MULTIPLIER

A typical multiplier consists of three parts: (i) partial product generation using an AND gate; (ii) PPR using an adder tree; and (iii) addition to produce the final result using a CPA. Power consumption and circuit complexity are dominated by the PPR [6], and the multiplier's critical path is dominated by the propagated carry chain in the CPA [7].

This section is organized as follows. Section III-A explains how the partial product layer is simplified by the approximate tree compressor. Section III-B introduces the CMA. Finally, Section III-C presents the overall structure of the accuracy-controllable approximate multiplier, which uses the proposed adder and tree compressor.

A. Approximate Tree Compressor

Figure 1(a) shows an accurate half adder, for which the following equation can be obtained:

$$\{c, s\} = a + b = 2c + s = (c + s) + c,$$

where {,} and + denote concatenation and addition, respectively. The value c is generated by a AND b and s is generated by a XOR b, so $(c + s)$ can be generated by a OR b. Based on the above, consider the basic logic cell shown in Fig. 1(b), for which the following equations can be obtained:

$$p = c + s,$$
$$q = c,$$
$$\{c, s\} = a + b = p + q.$$

This is called an incomplete adder cell (iCAC). Table I shows the truth tables for an accurate half adder and an iCAC. Note that the bit position of c and that of s, p, and q are different. As can be seen, q is equal to c. While p is not equal to s, the precise sum can be obtained by adding p and q, so the iCAC is not an approximate adder but an element of a precise adder.

By extending the above equation to m bits, the following equation can be obtained:

$$S = A + B = P + Q.$$

where A, B, P, and Q are m-bit values, the bits of which correspond to a, b, p, and q, respectively. A row of eight iCACs, used for 8-bit inputs, is shown in Fig. 2.

Consider the example of an 8-bit adder with the two inputs A = 01011111 and B = 00110110. The accurate sum S is 10010101, while the row of iCACs produces P = 01111111 and Q = 00010110. Again, it is evident that the following holds:

$$S = P + Q. \quad (1)$$

While S is obtained from P and Q, P can be used as an approximation for S, and Q can be used as an error recovery vector for the approximate sum P.

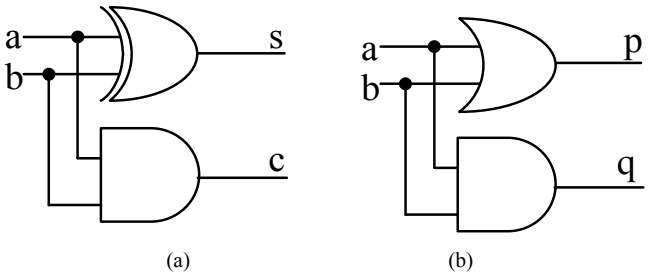

Fig. 1. (a) Accurate half adder and (b) incomplete adder cell.

TABLE I. TRUTH TABLES FOR ACCURATE HALF ADDER AND INCOMPLETE ADDER CELL.

Inputs		Outputs			
		Accurate half adder		iCAC	
a	b	c	s	q	p
0	0	0	0	0	0
0	1	0	1	0	1
1	0	0	1	0	1
1	1	1	0	1	1

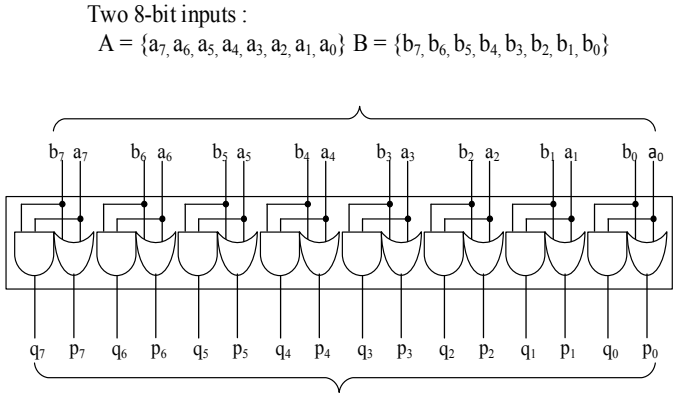

Fig. 2. A row of incomplete adder cells with two 8-bit inputs.

By extending the row of iCACs from two to n inputs, $n/2$ Ps and $n/2$ Qs are obtained. If the sum of the $n/2$ Qs is used instead of the $n/2$ Qs themselves, the number of Qs is reduced to one. Remember that P is always greater than or equal to S, and Q is equal to C. By exploiting these facts, OR gates can be used to generate the approximate sum of the $n/2$ Qs without significant loss of accuracy. This approximate sum is called the accuracy compensation vector and is denoted by V. This method is named approximate tree compressor (ATC). An ATC with n inputs is called an ATC-n, and the structure of an ATC with eight inputs (ATC-8) is shown in Fig. 3. The rectangles represent rows of iCACs and the number of iCACs in each row (rectangle) is dependent on the bit width of the inputs. For example, if there are eight m-bit inputs (D1, D2, …, D8), four rows of m iCACs are required to build a m-bit ATC-8. This reconstruction generates four approximate sums, P1, P2, P3, and P4, and four error recovery vectors, Q1, Q2, Q3, and Q4. OR gates generate the accuracy compensation vector V. As a result, the eight inputs have been reduced to five.

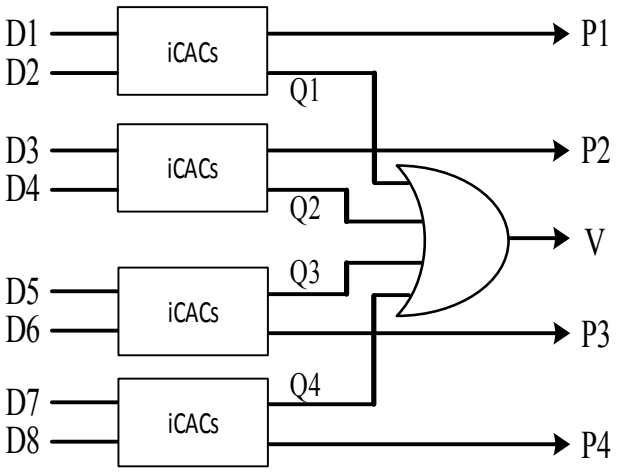

Fig. 3. Structure of an approximate tree compressor with eight inputs.

B. Carry-maskable Adder

A CMA is proposed to control the accuracy flexibly and dynamically. A k-bit CMA comprises $(k-1)$ carry-maskable full adders and one carry-maskable half adder, and its structure is similar to that of a k-bit CPA.

The structures of the proposed carry-maskable half and full adders are shown in Fig. 4. In the proposed half adder, when mask_x is 0, S is equal to x OR y and C_{out} is equal to 0. Otherwise, when mask_x is 1, S is equal to x XOR b and C_{out} is equal to x AND y. In other words, the operation of the proposed half adder can be controlled by the active-low signal mask_x. When mask_x is disabled (=1), it functions as an accurate half adder, and when mask_x is enabled (=0), C_{out} is masked to 0 and it functions as an OR gate with output S. The operation of the proposed full adder is similar to the half adder: when mask_x is disabled (=1), it functions as an accurate full adder, and when mask_x is enabled (=0), C_{out} is equal to C_{in} and S is the output of an OR gate.

C. Overall Structure

An n-bit multiplier consists of n rows, each of which has n partial products (PP), so there are $n \times n$ PPs in total. Using the ATC-n introduced in the previous section, the n rows can be replaced by $n/2+1$ rows. Figure 5 shows an example of an 8-bit multiplier with 8×8 PPs. The PPR is performed in three stages (Stage 1, Stage 2, and Stage 3) and the CPA is performed in Stage 4. The PP generation step is not shown. Each dot represents a PP. The least significant bit (right side) is bit 0, and the most significant bit (left side) is bit 14. The solid rectangles in Stage 1 represent ATCs and the dashed rectangles represent rows of seven iCACs. Every row of iCACs includes PPs that are not processed: for example, the PP at position 0 in the first row and the one at position 8 in the second row of the first iCAC block in ATC-8 are not processed.

In Stage 1, eight rows of PPs are reduced to four rows (P1, P2, P3, and P4) and one accuracy compensation vector (V1) by an ATC-8. The four rows are further reduced to two rows (P5 and P6) and another accuracy compensation vector (V2) by an ATC-4. A final row of iCACs then processes P5 and P6 and generates P7 and Q7. In summary, Stage 1 uses an ATC-8, an ATC-4, and a row of seven iCACs to compress the 8×8 PPs to four rows (P7, V1, V2, and Q7).

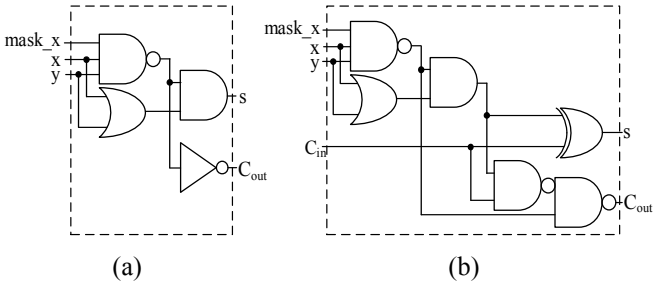

Fig. 4. (a) Carry-maskable half adder, (b) Carry-maskable full adder.

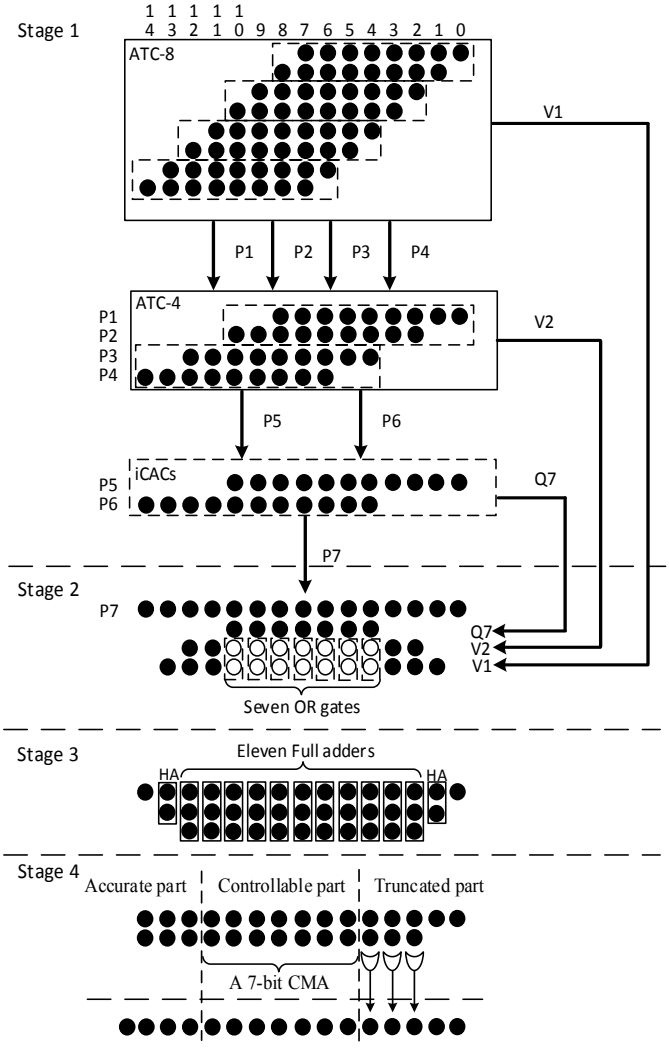

Fig. 5. Structure of an 8-bit multiplier with 8×8 partial products.

In Stage 2, there are four PPs for each of bits 4 to 10. In order to achieve a lower path delay, OR gates are used to sum V1 and V2 approximately. The empty circles for V1 and V2 represent the bits which are summed using OR gates. Seven OR gates are required in total and the four rows are compressed to three.

In Stage 3, full adders and half adders are used to compress the three rows to two. Two half adders are required for bits 1 and 13, and eleven full adders are required for bits 2 to 12.

Addition using a CPA is required after PPR to produce the final result. For an 8-bit Wallace tree multiplier, the length of

the CPA is 11 [7]. In our proposed multiplier, the length of the CPA is 13. In Stage 4, the CPA is divided into three parts in order to reduce the length of the carry propagation. Since the lower bits are not significant for accuracy, bits 0 to 4 are defined as the truncated part and three OR gates are used to generate the values for bits 2, 3, and 4 of the final result. Because there is no carry out from the truncated part, the length of the CPA is reduced to 10. Since the upper bits are the most significant for accuracy, bits 12 to 14 are defined as the accurate part, and three accurate adders are used to generate the values for these bits of the final result.

The accuracy-controllable part lies between the truncated and accurate parts. This part is important for both critical path delay and accuracy. In Stage 4, bits 5 to 11 in the CPA are replaced by a 7-bit CMA. Note that every 1-bit CMA has a mask_x signal. Given a value for u, the u upper bits in the accuracy-controllable part are configured as a u-bit CPA and the lower bits are configured as $(7-u)$ 2-input OR gates by managing the seven mask_x signals appropriately. When u = 7, it functions as a 7-bit CPA, and when u = 0, it functions as seven 2-input OR gates. For each bit of S that is generated by a 2-input OR gate, power consumption is reduced because the switching activity is reduced in some of the logic gates. Furthermore, the maximum delay of the CMA is reduced.

IV. Experimental Results

A. Experimental Setup

In this section, the proposed multiplier is evaluated in terms of power consumption, critical path delay, design area, and computational accuracy. To clarify the ability of the approximate multiplier to save power, shorten critical path delay, and control the accuracy, a conventional Wallace tree multiplier and the previously-proposed approximate multipliers [3] [8] were implemented for comparison. The approximate multiplier [3] can be configured at design time, and its accuracy is controlled by the length of the recovery vector. Four different approximate multipliers were implemented, using 10-bit, 8-bit, 6-bit, and 4-bit recovery vectors, and are referred to as AMER_10b, AMER_8b, AMER_6b, and AMER_4b, respectively. Note that the accuracy of AMER_XX is not dynamically controllable, unlike that of our proposed multiplier. The ACCI2 approximate multiplier [8] is one of the most accurate approximate multipliers and is referred to as ACCI_M2. The multipliers with eight different accuracy settings (values of u) are referred to as m_7b, m_6b, …, m_0b. Multiplier m_ub utilized an approximate adder for the final results from the PPR consisting of a $(3+u)$-bit CPA and $\{(7-u)+3\}$ 2-input OR gates. For example, the approximate adder for m_6b consisted of a 9-bit CPA and four 2-input OR gates.

All the approximate multipliers, as well as the conventional Wallace tree multiplier, were eight bits and coded using Verilog HDL. The Synopsys VCS was used to simulate the designs and generate value change dump (VCD) files to evaluate the power consumption precisely. The Synopsys Design Compiler was used to synthesize the multipliers with the NanGate 45nm Open Cell Library [9]. The power consumption was evaluated at a frequency of 0.5GHz. The operating conditions for synthesis were typical (a 1.00 process factor, 1.1 V power supply, and 25°C operating temperature). All designs were synthesized and optimized using the default compiler options. The Synopsys Power Compiler was used to estimate power consumption from switching activity interchange format files generated from the VCD files. The Synopsys VCS was used to evaluate the numerical outputs of all the multipliers. Because 8-bit multipliers were evaluated, the total number of test patterns was 65,536.

TABLE II. Accuracy Comparison.

	NMED (%)	MRED (%)	ER (%)
m_7b	0.25	0.85	36.16
m_6b	0.26	0.99	43.46
m_5b	0.29	1.31	52.07
m_4b	0.35	1.93	61.05
m_3b	0.49	3.05	69.61
m_2b	0.71	4.57	74.93
m_1b	1.05	6.50	78.10
m_0b	1.64	9.02	80.02
AMER_10b	0.20	0.62	31.59
AMER_8b	0.24	1.16	55.44
AMER_6b	0.46	3.23	71.12
AMER_4b	1.20	7.53	79.54
ACCI_M2	0.04	0.62	72.29

B. Accuracy Results

The error distance (ED) and mean ED (MED) measures have been proposed to evaluate the performance of approximate arithmetic circuits [10]. For multipliers, the ED is defined as the arithmetic difference between the accurate product (S) and the approximate product (S′): $\mathrm{ED} = |\mathrm{S} - \mathrm{S}'|$. The MED is the average ED for a set of outputs. In [3], the mean relative ED (MRED) and normalized MED (NMED) are proposed to evaluate approximate multipliers. The relative ED (RED) is the ED divided by the accurate output: $\mathrm{RED} = |\mathrm{S} - \mathrm{S}'|/\mathrm{S}$, and the MRED is the average RED, which can be obtained similarly to the MED. The NMED is defined as $\mathrm{NMED} = \mathrm{MED}/\mathrm{S}_{\max}$, where $\mathrm{S}_{\max}$ is the maximum output magnitude of an accurate multiplier. The error rate (ER) is the percentage of inaccurate outputs among all outputs generated from all combinations of inputs. These three metrics (NMED, MRED, and ER) were used to evaluate the proposed multiplier.

Table II compares the accuracy results. It can be seen that the accuracy of the proposed multiplier changes widely according to its setting. While the NMED and MRED values of the most accurate configuration of the proposed multiplier are larger than those of the most accurate AMER configuration and ACCI_M2, its controllability is better than that of AMER. Remember that the proposed multiplier is dynamically controllable, unlike AMER.

C. Power, Critical Path Delay, and Design Area Results

Comparisons of the power consumption and critical path delay for the different multipliers relative to accuracy are shown in Fig. 6 and Fig. 7, respectively, where the x-axis indicates the MRED. The circles, triangles, asterisk, and square represent the proposed accuracy-controllable multiplier with different dynamic configurations (m_7b, m_6b, …, m_0b), the

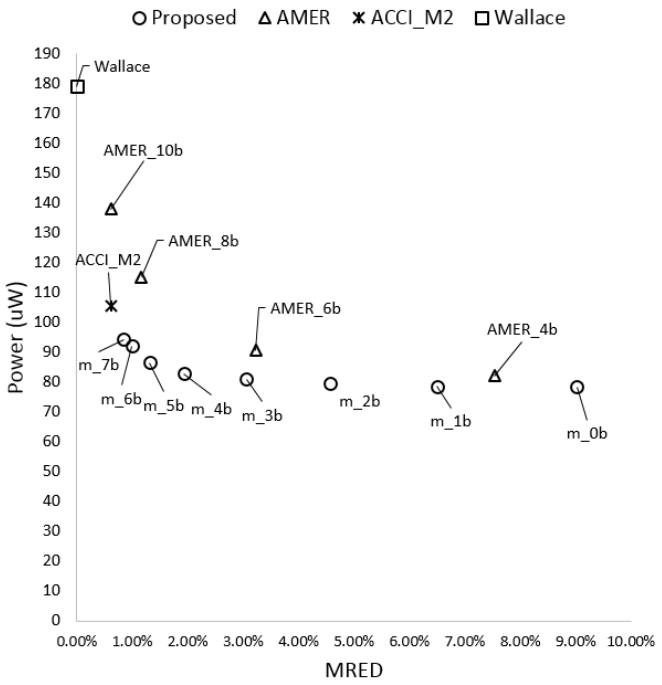

Fig. 6. Power consumption results relative to the MRED.

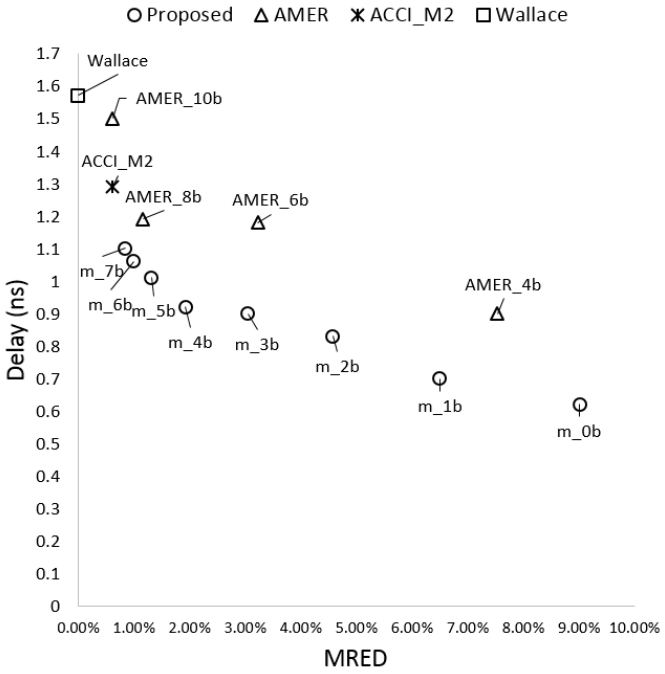

Fig. 7. Critical path delay results relative to the MRED.

AMERs [3] with different static configurations (AMER_10b, AMER_8b, AMER_6b, AMER_4b), ACCI_M2 [8], and the Wallace tree multiplier, respectively. The power values used in this evaluation are the sum of the dynamic and static power consumptions.

As can be seen in Fig. 6 and Fig. 7, the proposed multiplier achieves good results, both in terms of power consumption and critical path delay. For all accuracies (MRED), the proposed accuracy-controllable multiplier achieves the smallest power consumption and delay results. For example, if MRED of around 9% is required, m_0b delivers the lowest power consumption and the shortest critical path delay.

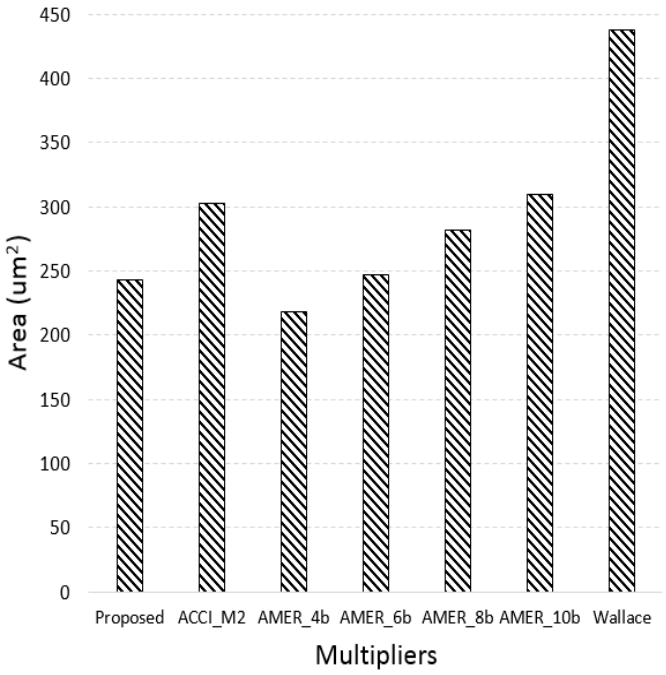

Fig. 8. Design area results.

TABLE III. INPUT IMAGES.

Image No.	Description
1	Lena
2	Some peppers
3	A bridge
4	A domed palace
5	A truck on grassland
6	A bird standing in a stream
7	A view of a small town
8	A house and a car

A comparison of the design area results is shown in Fig. 8. Note that the accuracy setting does not have any effect on the design area of the proposed multiplier because it is dynamically configurable, and thus only one design area result is shown. In contrast, AMER produces different results for different accuracy settings because it is not dynamically configurable. While our proposed multiplier is larger than AMER_4b, it consumes less power and has a shorter critical path delay than AMER_4b does. In addition, the proposed multiplier has a smaller design area than AMER_6b, while having lower power consumption and critical path delay and a higher accuracy.

D. Image Processing

An image processing application was also evaluated. An image sharpening algorithm [11] was used, which is popular in the evaluation of approximate multipliers. Eight images collected from the Internet were used, all 512×512 8-bit grayscale bitmap images, and these are summarized in Table III. Only the multiplications were approximate; all the other operations (addition, subtraction, and division) were accurate.

TABLE IV. PSNR RESULTS OF THE APPROXIMATE MULTIPLIERS, IN DB.

Image No.	m_7b	m_6b	m_5b	m_4b	m_3b	m_2b	m_1b	m_0b	AMER 10b	AMER 8b	AMER 6b	AMER 4b	ACCI_M2
1	53.1	50.9	45.8	40.4	34.4	28.6	25.1	15.5	57.0	44.6	31.7	22.8	49.4
2	54.1	51.6	47.6	42.0	35.8	30.1	27.4	17.9	57.0	46.5	33.8	24.9	51.1
3	56.0	52.6	48.8	40.2	33.3	31.6	27.0	27.0	56.8	46.8	33.7	25.3	50.9
4	52.9	49.3	45.9	39.6	32.1	30.3	25.5	24.1	55.3	45.3	32.0	23.5	49.1
5	50.3	49.1	46.7	41.3	33.9	32.9	23.9	23.9	57.3	49.4	31.4	22.7	51.4
6	50.9	49.3	46.8	39.8	32.9	30.4	25.1	24.1	52.3	46.6	31.8	23.7	49.4
7	49.4	48.3	46.2	40.4	33.7	29.7	25.1	13.6	57.0	46.5	31.5	18.8	49.4
8	53.6	51.1	47.6	42.8	35.4	27.3	24.8	11.5	56.7	46.8	31.7	20.3	51.0

The processed image quality was measured using the peak signal-to-noise ratio (PSNR). This is usually used to measure the quality of reconstructive processes that involves information loss and is defined in terms of the mean squared error (MSE) [6]. The MSE and PSNR were defined in [6] as

$$\text{MSE} = \frac{1}{mp}\sum_{i=0}^{m-1}\sum_{j=0}^{p-1}[I(i,j) - K(i,j)]^2, \qquad (2)$$

$$\text{PSNR} = 10\log_{10}\left(\frac{MAX_I^2}{\text{MSE}}\right), \qquad (3)$$

where $I(i,j)$ and $K(i,j)$ are the correct and obtained values, respectively, of each pixel, m and p are the image dimensions, and MAX_I represents the maximum value of each pixel (255 here, as the images are 8-bit).

Table IV shows the PSNR results of the approximate multipliers, in dB. Larger values represent better quality images. As can be seen, different PSNR values are found for the different images on each column of the table. This confirms that the dynamic reconfigurability is necessary for the situations where different qualities are required. In addition, the proposed accuracy-controllable multiplier produced a wide range of PSNR values, with its largest values being comparable to those of the other approximate multipliers.

V. CONCLUSION

An accuracy-controllable approximate multiplier has been proposed in this paper that consumes less power and has a shorter critical path delay than the conventional design. Its dynamic controllability is realized by the proposed CMA. The multiplier was evaluated at both the circuit and application levels. The experimental results demonstrate that the proposed multiplier was able to deliver significant power savings and speedups while maintaining a significantly smaller circuit area than that of the conventional Wallace tree multiplier. Furthermore, for the same accuracy, the proposed multiplier delivered greater improvements in both power consumption and critical path delay than other previously studied approximate multipliers. Finally, the ability of our proposed multiplier to control accuracy was confirmed by an application-level evaluation.

ACKNOWLEDGMENT

Thanks are due to Katsuhiko Wakasugi of Logic Research Co., Ltd. for assistance with the experiments. This work was supported by JSPS KAKENHI Grant Number JP17K00088 and by funds (No.175007 and No.177005) from the Central Research Institute of Fukuoka University. This work is supported by VLSI Design and Education Center (VDEC), the University of Tokyo in collaboration with Synopsys, Inc.

REFERENCES

[1] S. Venkataramani, V. K. Chippa, S. T. Chakradhar, K. Roy, and A. Raghunathan. "Quality programmable vector processors for approximate computing," *46th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO),* pp. 1-12, Dec. 2013.

[2] 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 Transactions on Circuits and Systems I: Regular Papers,* vol. 57, no. 4, pp. 850-862, Apr. 2010.

[3] C. Liu, J. Han, and F. Lombardi, "A Low-Power, High-Performance approximate multiplier with configurable partial error recovery," *Design, Automation & Test in Europe Conference & Exhibition (DATE),* Mar. 2014.

[4] S. Hashemi, R. I. Bahar, and S. Reda, "DRUM: A Dynamic Range Unbiased Multiplier for approximate applications," *IEEE/ACM International Conference on Computer-Aided Design (ICCAD),* pp. 418-425, Nov. 2015.

[5] B. Moons, M. Verhelst, "DVAS: Dynamic Voltage Accuracy Scaling for increased energy-efficiency in approximate computing," *IEEE/ACM International Symposium on Low Power Electronics and Design (ISLPED),* Jul. 2015.

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

[7] K. C. Bickerstaff, E. E. Swartzlander, and M. J. Schulte, "Analysis of column compression multipliers," *15th IEEE Symposium on Computer Arithmetic,* pp. 33-39, Jun. 2001.

[8] Z. Yang, J. Han, and F. Lombardi, "Approximate compressors for Error-Resilient multiplier design," *IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS),* pp. 183-186, Oct. 2015.

[9] NanGate, Inc. NanGate FreePDK45 Open Cell Library, http://www.nangate.com/?page_id=2325, 2008

[10] J. Liang, J. Han, and F. Lombardi, "New metrics for the reliability of approximate and probabilistic adders," *IEEE Transactions on computers,* vol. 62, no. 9, pp. 1760-1771, Sep. 2013.

[11] M. S. Lau, K. V. Ling, and Y. C. Chu, "Energy-Aware probabilistic multiplier: Design and Analysis," *2009 international Conferrence on Compliers, architeture, and synthesis for embedded systems,* pp. 281-290, Oct. 2009.

View publication stats