IEICE TRANSACTIONS

on Fundamentals of Electronics, Communications and Computer Sciences

VOL. E106-A NO. 3
MARCH 2023

A PUBLICATION OF THE ENGINEERING SCIENCES SOCIETY
The Institute of Electronics, Information and Communication Engineers
Kikai-Shinko-Kaikan Bldg., 5-8, Shibakoen 3 chome, Minato-ku, TOKYO, 105-0011 JAPAN

IEICE TRANS. FUNDAMENTALS, VOL.E106–A, NO.3 MARCH 2023

PAPER Special Section on VLSI Design and CAD Algorithms

An Accuracy Reconfigurable Vector Accelerator based on Approximate Logarithmic Multipliers for Energy-Efficient Computing*

Lingxiao HOU[†], ***Nonmember***, **Yutaka MASUDA**[†], ***and*** **Tohru ISHIHARA**[†a)], ***Members***

SUMMARY The approximate logarithmic multiplier proposed by Mitchell provides an efficient alternative for processing dense multiplication or multiply-accumulate operations in applications such as image processing and real-time robotics. It offers the advantages of small area, high energy efficiency and is suitable for applications that do not necessarily achieve high accuracy. However, its maximum error of 11.1% makes it challenging to deploy in applications requiring relatively high accuracy. This paper proposes a novel operand decomposition method (OD) that decomposes one multiplication into the sum of multiple approximate logarithmic multiplications to widely reduce Mitchell multiplier errors while taking full advantage of its area savings. Based on the proposed OD method, this paper also proposes an accuracy reconfigurable multiply-accumulate (MAC) unit that provides multiple reconfigurable accuracies with high parallelism. Compared to a MAC unit consisting of accurate multipliers, the area is significantly reduced to less than half, improving the hardware parallelism while satisfying the required accuracy for various scenarios. The experimental results show the excellent applicability of our proposed MAC unit in image smoothing and robot localization and mapping application. We have also designed a prototype processor that integrates the minimum functionality of this MAC unit as a vector accelerator and have implemented a software-level accuracy reconfiguration in the form of an instruction set extension. We experimentally confirmed the correct operation of the proposed vector accelerator, which provides the different degrees of accuracy and parallelism at the software level.

key words: *approximate computing, energy-efficient computing, low power design, vector acceleration, single instruction multiple data*

1. Introduction

Vector multiplication units highly contribute to improving the performance of data-intensive applications such as image processing and real-time robotics. However, the accurate conventional multipliers are known as power-hungry and area-expensive. In applications involving large amounts of multi-bit data or floating-point multiplication, multipliers dominate the area and power dissipation.

On the other hand, multiplication does not need to be accurate in many practical applications. A multiplication unit with a smaller area and higher speed is needed for applications that require massive computational parallelism [2]. For example, some approximate multipliers have been used in applications like image sharpening and smoothing [3]–[6]. However, these proposals have focused on bit truncation or compression of the partial products, with limited effect on reducing the multipliers' area and energy consumption.

The approximate logarithmic multiplier proposed by Mitchell converts multiplication to more uncomplicated shift and addition operations [7]. [8] experimentally demonstrated that it reduces the area to as low as 34% of an accurate multiplier. The approximate multiplier is very effective for deep neural networks (DNN) since DNN includes massively parallel multiply-accumulation but does not necessarily achieve high accuracy [8], [9]. However, for applications that require both high parallelism and a specific degree of accuracy, the accuracy of the Mitchell multiplier becomes a bottleneck. Moreover, there is a naive scheme integrating both Mitchell and accurate vector multiplication units and using them on demand. However, this strategy would compromise Mitchell multipliers' high area efficiency advantage and is not worthwhile.

This paper first proposes a novel operand decomposition (OD) method to combine with the Mitchell approximate multiplication in [7] that improves the accuracy of [7]. Based on the proposed OD method, this paper proposes an accuracy reconfigurable multiply-accumulate (MAC in the following) unit for massive parallel computation. Note that the functional units, including multipliers and MAC units, presented in this paper are all integer functional units. The proposed MAC unit achieves the following key contributions: 1). The accuracy is significantly improved over the original Mitchell approximation. 2). The area is reduced dramatically compared to the accurate unit, effectively improving the computational parallelism. 3). Multiple accuracies can be reconfigured by different OD modes to suit various scenarios.

This paper is an extension of our previous work [1]. The previous work [1] validates the accuracy reconfiguration for several practical applications like image smoothing and Simultaneous Localization and Mapping (SLAM) in robotics. However, it does not show the execution cycle reduction brought by the proposed MAC unit. This paper newly discusses in Sect. 5 how the proposed MAC unit reduces the execution cycles in practical applications. This paper also newly shows a prototype of a RISC processor integrating the proposed MAC unit as a vector accelerator in Sect. 6. With RTL simulation results for the prototype processor running several parallel programs, we show the processor effectively

Manuscript received March 10, 2022.
Manuscript revised July 6, 2022.
Manuscript publicized September 2, 2022.

[†]The authors are with Graduate School of Informatics, Nagoya University, Nagoya-shi, 464-0814 Japan.

*This work was partly presented in the conference [1].

a) E-mail: ishihara@ertl.jp

DOI: 10.1587/transfun.2022VLP0005

reduces the execution cycles if a specific degree of accuracy degradation is acceptable.

The rest of this paper is organized as follows: Section 2 reviews the original Mitchell multiplier and the original operand decomposition (OOD) method. Section 3 proposes the novel OD method, which is essential for the proposed MAC unit, then highlights the "accuracy reconfigurability" feature. The experimental results regarding the accuracy improvement and area reduction are shown in Sect. 4. Section 5 describes the applicability of the proposed MAC unit to image smoothing and robot SLAM applications. Section 6 describes a prototype open-source RISC processor integration for the proposed MAC unit as a vector accelerator and shows the execution cycle reduction in practical applications. Lastly, concluding remarks are given in Sect. 7.

2. Related Works

This section summarizes several related works, including the original Mitchell approximate multiplier and the original operand decomposition (OOD) method. We analyze their principles, advantages, and disadvantages.

2.1 Mitchell Approximate Logarithmic Multiplier

The Mitchell approximate logarithmic multiplier converts a multiplication into an anti-logarithm of the sum of two approximate logarithms [7]. Suppose two inputs to a Mitchell multiplier are A and B:

$$\begin{aligned} A &= 2^{k_A}(1+m_A), m_A \in [0,1), \\ B &= 2^{k_B}(1+m_B), m_B \in [0,1), \end{aligned} \tag{1}$$

where the positions of the leading "1" bit of both inputs are k_A and k_B, respectively. The remaining bits of both inputs after the leading "1" are the mantissas m_A and m_B. The Mitchell multiplier approximates the logarithms of the two inputs as (2):

$$\begin{aligned} \log_2 A &= k_A + m_A, \\ \log_2 B &= k_B + m_B. \end{aligned} \tag{2}$$

Next, we add the two approximate logarithms for deriving $\log_2(A \times B)$. Note that the carry from $m_A + m_B$ is the carry-in for $k_A + k_B$. Lastly, we do anti-logarithm to $\log_2(A \times B)$ to get the approximate result of $A \times B$ by concatenating the mantissa m after a leading "1". Then we shift left to make the leading "1" to position k, where k and m are the leading "1" position and mantissa of $\log_2(A \times B)$, respectively.

Whereas the Mitchell multiplier significantly reduces the area compared to the accurate multiplier, it suffers from computational accuracy. The maximum error of 11.1% occurs when both m_A and m_B are 0.5. The approximate product, which is always smaller than the actual value, leads to the accumulation and widening of the error.

Babić et al. use correction circuits to reduce the maximum error to less than 0.5% (3 correction circuits) [10]. However, the parallel implementation with only one correction circuit doubles the area compared to the Mitchell multiplier. Ansari et al. proposed to round the operand to its nearest power of 2 [11]. Alla et al. combined the Mitchell multiplier with a hardware pruning technique, producing a more area-efficient multiplier without compromising accuracy [12]. Pilipović et al. further reduced the area and power consumption of the Mitchell multiplier by combining a two-stage operand trimming [13]. Still, as a cost, their proposal has a more significant error than the Mitchell multiplier.

2.2 Original Operand Decomposition (OOD)

Mahalingam et al. proposed the operand decomposition method (OOD) [14] to improve the accuracy of the Mitchell multiplier. Assume that the two n-bit inputs are X and Y. Decompose X and Y into four operands $A = X \bigvee Y$, $B = X \bigwedge Y$, $C = \overline{X} \bigwedge Y$, and $D = X \bigwedge \overline{Y}$. The approximate product of $X \times Y$ is calculated by (3):

$$X \times Y \approx (A \times B) + (C \times D), \tag{3}$$

where the "$\times$" here indicates the Mitchell approximate multiplication.

The OOD method reduces the number of "1" bits in every decomposed operand, making the switching power dissipated for the multiplication reduced. However, it does not reduce the maximum error, and the reduction in the average error is only slight. Moreover, its fixed "decompose into the sum of two approximate multiplications" is also detrimental to the area-saving advantage of the Mitchell multiplier. Nandan et al. proposed a variant of the OOD method [15] whose primary goal is to improve the delay and power consumption, but with only a slight accuracy improvement.

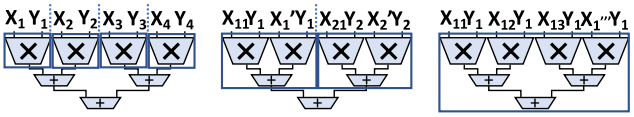
The above solutions solve the problem of the excessive area of the accurate multiplier, and some other methods offer a few degrees of improvement in accuracy. However, they do not meet the requirements of multiple accuracy reconfigurability and cannot be adapted to application scenarios with high parallelism and various accuracies.

3. Proposed Approach

In this section, first, we propose the novel OD method. Next, we introduce the operating modes of the proposed accuracy reconfigurable MAC unit and then highlight the advantages of the accuracy reconfigurability feature.

3.1 Proposed Operand Decomposition (OD)

The proposed accuracy reconfigurable MAC unit has three operating modes corresponding to the OD method OD-1, OD-2, and OD-4, respectively. Figure 1(a) shows a non-decomposition configuration that calculates four multiply-accumulation in parallel using a 4-Mitchell multiplication unit. Since each multiplication remains as it is, we refer to this mode as OD-1. The proposals to decompose



(a) OD-1 (b) OD-2 (c) OD-4

Fig. 1 The proposed MAC unit providing different degrees of accuracy and parallelism by switching the OD modes.

one multiplication into two and four ways are referred to as OD-2 and OD-4, respectively, which are shown in Fig. 1(b) and Fig. 1(c). We now explain how our proposed OD method improves the accuracy by reconfiguring the configuration shown in Fig. 1(a), which is equivalent to the original Mitchell approximation in computation.

First, we describe the OD-2 method as illustrated in Fig. 1(b). For given two binary inputs X_1 and Y_1, the OD-2 method finds the position $k_{X_{11}}$ of the leading "1" bit of X_1, and fills $k_{X_{11}}$ "0"s after "1". We refer to this new number as X_{11}. The second number is the remaining bits of X_1 without the leading "1" bits, and we refer to it as X_1'. The X_1' can be obtained by (4):

$$X_1' = \overline{X_{11}} \bigwedge X_1. \tag{4}$$

Then the approximation of $X_1 \times Y_1$ is calculated by (5):

$$X_1 \times Y_1 = (X_{11} \times Y_1) + \left(X_1' \times Y_1\right), \tag{5}$$

where the "$\times$" here indicates the Mitchell approximate multiplication. Since X_{11} is the exponent of 2, the mantissa of the approximate $\log_2 X_{11}$ is 0. Therefore, antilog $\left(\log_2 X_{11} + \log_2 Y_1\right)$ is equivalent to shifting Y_1 to the left by $k_{X_{11}}$ bits, which does not produce any error. Meanwhile, since X_1' is a smaller number than X_1, $X_1' \times Y_1$ produces a smaller error than applying Mitchell approximation to $X_1 \times Y_1$ directly. Therefore, the proposed OD-2 method reduces the overall error.

The OD-4 similarly decomposes operand X_1 into four operands, which corresponds to Fig. 1(c). In this case $X_1 \times Y_1$ is calculated by (6):

$$X_1 \times Y_1 = X_{11} \times Y_1 + X_{12} \times Y_1 + X_{13} \times Y_1 + X_1''' \times Y_1, \tag{6}$$

where X_{11}, X_{12}, and X_{13} hold the three most significant "1" bits of X_1, respectively. The approximation of $X_1 \times Y_1$ consists of a sum of four original Mitchell approximations. Similar to the discussion for OD-2 mode, the first three approximation does not produce any error, while the last one produces a much smaller error, significantly making the overall result more accurate.

Optionally, decomposing a smaller operand of X_1 and Y_1 can further improve the accuracy since we expect the mantissa m to contain fewer "1" bits, resulting in a minor error in the approximate logarithm.

Another key point of the proposed OD method is that most parts shown in Fig. 1(a) can be reused in the configuration shown in Fig. 1 and(b) Fig. 1(c). In other words, just providing the decomposed operands to the configuration

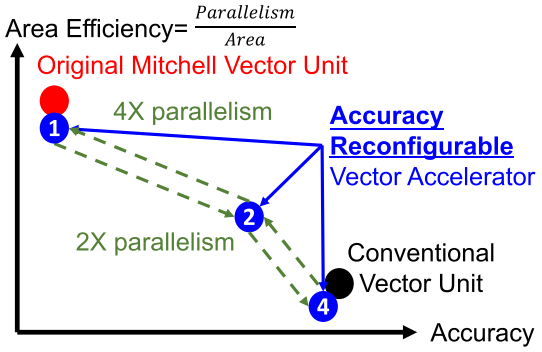


Fig. 2 The proposed vector accelerator makes a trade-off between accuracy and parallelism to achieve on-demand accuracy reconfiguration.

improves the accuracy. The accuracy reconfigurability in Sect. 3.2 is achieved by switching among the three modes. Section 4 experimentally shows the accuracy improvement thanks to the proposed OD method.

3.2 Accuracy Reconfigurability Feature

By integrating the accuracy reconfigurable MAC unit into a dedicated co-processor, this paper also proposes a vector accelerator which satisfies high accuracy and high parallelism and supports on-demand accuracy reconfiguration with only a tiny area overhead. A prototype of the vector accelerator is presented in Sect. 5. The key idea is illustrated in Fig. 2. The conventional accurate multiplier (black) has 100% accuracy but occupies the most significant area. That means only a few accurate multipliers can be integrated as a vector multiplication unit with a limited circuit area. In contrast, the original Mitchell multiplier (red) significantly reduces the area, increasing the parallelism of a vector multiplication unit. However, its error is challenging to cope with high accuracy in some scenarios. Considering the area and accuracy of these vector multiplication units, integrating any of the two vector units alone cannot meet the needs for both high parallelism and high accuracy. Moreover, the area overhead of integrating both makes this not worthwhile.

Our proposal (blue) has the advantage that, by making a trade-off between accuracy and parallelism, the vector accelerator can meet the usage requirements of various accuracies, keeping high parallelism with only a small area overhead. The key idea is the critical proposal of the OD method described in Sect. 3.1. The accuracy of OD-1 mode ("1" in Fig. 2, corresponding to Fig. 1(a)) is identical with that of the original Mitchell multiplier, but the parallelism is slightly decreased due to the decomposition circuit. OD-2 mode ("2" in Fig. 2, corresponding to Fig. 1(b)) makes one multiplication to use two approximate multipliers, significantly improving the accuracy. OD-4 mode ("4" in Fig. 2, corresponding to Fig. 1(c)) gives the accuracy comparable to that of the accurate multiplier, but at the expense of the most parallelism. By taking the advantages of the proposed vector accelerator, both problems of the Mitchell multiplication unit and the accurate unit mentioned above can be overcome.

Also, by implementing as an instruction set extension, the different degrees of accuracy can be provided at the software level, making the proposed vector accelerator more generalizable to any usage scenario and decoupling it from specific hardware design.

4. Evaluation of the MAC Unit

This section experimentally evaluates the accuracy, area, power, and critical path delay of the proposed accuracy reconfigurable MAC unit.

First, we evaluate the accuracy. Since the approximation is introduced into only the multiplier part in our accuracy reconfigurable MAC unit, we evaluate the accuracy of the multiplier part only. We implement our proposal of OD-1, OD-2, and OD-4 in C++. We also implement the OOD method in [14] for comparison. The original Mitchell method in [7] is the same in computation as OD-1. We evaluate the above methods with 8-bit, 16-bit, and 32-bit unsigned integers, respectively. Maximum error and *Mean Relative Error Distance (MRED)* [16] are used as criteria for the accuracy evaluation, which is calculated by (7):

$$\text{MRED} = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|P_i - P_i'\right|}{P_i}, \tag{7}$$

where N is the number of multiplication test cases, i is the index of each test case, P_i is the accurate product of the i-th test case, and P_i' is the approximate product of the i-th test case.

In the evaluation, all input patterns were given to the 8-bit multipliers, and 1 million patterns of pseudo-random numbers were given to the 16-bit and 32-bit multipliers. Although the test coverage for the 16-bit and 32-bit multipliers is far less than 1%, we experimentally confirmed in this simulation that the values of the maximum error and MRED are unchanged in the value at 3 significant digits even if the number of input patterns given to the multipliers was doubled from the 1 million patterns. We also confirmed that the values of the maximum error and MRED for the original Mitchell multiplier in Table 1 are consistent with the values which are analytically derived in [7]. Table 1 shows the accuracy evaluation results in various situations. Compared to the original Mitchell multiplier (OD-1), the proposed OD-2 method provides about three times improvement in MRED, while the OD-4 method provides even up to 30 times improvement in MRED. The maximum error is also improved by a factor of 2 and 10, respectively.

Next, we evaluate the circuit area and critical path delay of the proposed reconfigurable MAC unit. For comparison, we also implemented the vector multiplication units consisting of the accurate multipliers, original Mitchell multipliers in [7], and OOD multipliers in [14], respectively. All of the above implementations contains eight parallel accurate or approximate multipliers. We use a commercial logic synthesis tool to obtain the area, power, and timing reports for the above hardware designs. We use a commercial foundry-provided standard cell library based on 55 nm CMOS process technology.

Table 1 Accuracy of 8-bit, 16-bit, and 32-bit unsigned integer multiplication.

No. of bits	Method	Max. Err. (%)	MRED (%)
8	**Mitchell (OD-1) [7]**	11.11	3.76
	OOD [14]	11.11	2.01
	Proposed OD-2	4.53	1.11
	Proposed OD-4	0.64	0.09
16	**Mitchell (OD-1) [7]**	11.11	3.84
	OOD [14]	11.11	2.17
	Proposed OD-2	4.81	1.17
	Proposed OD-4	1.09	0.12
32	**Mitchell (OD-1) [7]**	11.11	3.84
	OOD [14]	11.11	2.18
	Proposed OD-2	4.81	1.17
	Proposed OD-4	1.10	0.12

Table 2 Area, Power, and Critical Path Delay Evaluation with Foundry-Provided Cell Library.

Type of MAC Unit	Area (μm^2)	Power (mW)	Delay (ps)
Accurate	117,768.85	24.53	1,438.19
Mitchell [7]	36,581.71	5.05	1,567.51
OOD [14]	41,234.96	5.70	1,631.31
Proposed	50,504.28	9.73	1,942.02
Proposed[1]	55,071.58	12.59	2,190.28
Proposed[2]	41,766.81	8.08	2,026.37
Proposed[3]	40,236.36	7.42	1,934.48

[1] With input comparison
[2] OD-1 and OD-4 only
[3] OD-1 and OD-2 only

Table 2 shows the evaluation results of the area, power, and critical path delay. We run gate-level simulation for 10,000 pseudo random input patterns and get switching activity information as a SAIF format to calculate the power consumption values. The clock frequency assumed in the simulation is 200 MHz. Therefore, the power values shown in Table 2 correspond to the power consumption of the MAC units when running at 200 MHz. The area of the vector Mitchell multiplication units is only 31.06% of the accurate multiplication unit. The area of the unit using OOD multipliers is close to the Mitchell unit, but it fixedly halved the degree of parallelism. The proposed accuracy reconfigurable MAC unit area is increased compared to the Mitchell unit, but still only 42.88% of the accurate unit, achieving a significant reduction. Even with the input comparator mentioned in Sect. 3.1, the area only grows to 46.76% of the accurate unit, which is still less than half. If we keep only one mode with the highest degree of computational parallelism (OD-1) and the mode with the highest accuracy (OD-4) for switching, the area will be reduced to 35.47% of the accurate unit. If we keep only the OD-2 mode in addition to the OD-1 mode as specified as Proposed[3], the area can be reduced to 34.17% of the accurate counterpart. Thanks to this significant area reduction, we can accelerate vector multiplication or multiply-accumulation by integrating higher parallel vector units on a chip. On the other hand, the critical path delay is increased by up to 52.29% compared to the accurate unit due to the decomposition circuit, which is one of our main

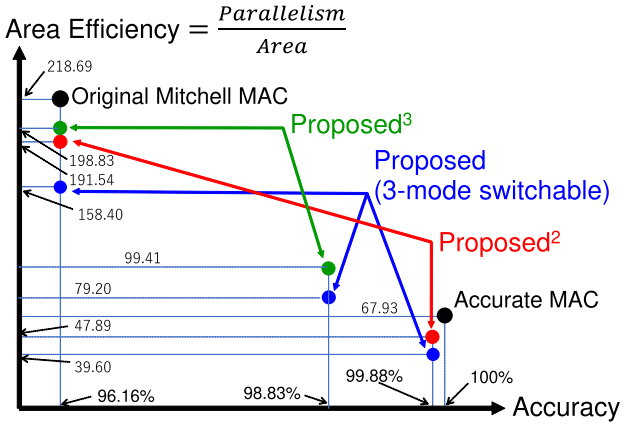


Fig. 3 Quantitative analysis on the trade-off between accuracy and area-efficiency to achieve on-demand accuracy reconfiguration.

future work.

The quantitative analysis based on Table 1 and Table 2 is summarized in Fig. 3. The proposed 3-mode switchable configuration which can reconfigure OD-1, OD-2, and OD-4 modes has relatively low area efficiency. For example, the area efficiency of this configuration which corresponds to the blue dots in Fig. 3 is lower than that of the original Mitchell MAC unit by 28% when it uses the OD-1 mode, which is considerably large. However, if we target applications that do not need accuracy of better than 98.8% for example, we can employ the 2-mode switchable configuration which uses the OD-1 and OD-2 modes only. This configuration of the Proposed^3 in Fig. 3, which corresponds to the green dots in Fig. 3 improves the area efficiency by 19% over the 3-mode switchable configuration when using the OD-1 mode, which is only 9% lower area efficiency than that of the original Mitchell MAC unit. Even for applications that need accuracy of up to 99.8% in some cases, we can satisfy the accuracy requirement by employing the configuration of the proposed^2 which can use the OD-1 and OD-4 modes while enjoying the high parallelism by using the OD-1 mode when the accuracy requirement is lower than 96%.

5. Application Case Studies

This section discusses the applicability of the proposed accuracy reconfigurable MAC unit to the image smoothing and robotic Simultaneous Localization and Mapping (SLAM) applications. We compare the applicability of the proposed OD-2 and OD-4 methods with OD-1 (which is equivalent to the original Mitchell multiplier in computation) and the accurate multiplication unit. All evaluations are implemented using C++.

5.1 Image Smoothing

Image Smoothing [17] mainly removes the noise from an image by convolving the image with a smoothing kernel, which is a square matrix with each value conforming to a Gaussian distribution. A smoothed pixel's new value is calculated by multiplying-accumulating the pixels around the original pixel (including itself) with the smoothing kernel. The size of pixels involved in every computation is the same as the size of the smoothing kernel.

Table 3 SSIM Index and PSNR of image smoothing.

	Color		Grayscale	
	SSIM	**PSNR**	**SSIM**	**PSNR**
OD-1	0.989	29.511	0.994	34.793
OD-2	0.997	39.638	0.998	45.005
OD-4	0.998	53.868	0.999	56.860

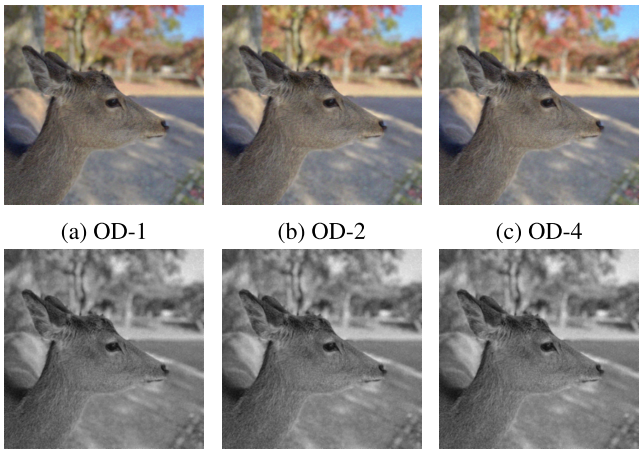
We quantify the floating-point smoothing kernel to the unsigned fixed-point number with eight fractional bits due to the nature of the proposed MAC unit:

$$K_q = \frac{\lfloor K \cdot 2^8 \rfloor}{2^8}, \tag{8}$$

where K is the floating-point value of the smoothing kernel. As image quality metrics, we select the *Structural Similarity Index (SSIM Index)* [18] and the *Peak Signal-to-Noise Ratio (PSNR)*. We compare the SSIM Index and PSNR of the smoothed color and grayscale images using the three operating modes of the proposed MAC unit to the benchmark using the accurate multipliers, respectively.

Table 3 shows the results of SSIM Index and PSNR of the smoothed images. Note that the SSIM index and PSNR values shown in Table 3 are calculated with the smoothing results obtained with the accurate integer MAC unit as the reference value. Compared to the images smoothed using OD-1 mode, OD-2, and OD-4 modes significantly improve SSIM Index and PSNR. Figure 4 visualizes the smoothed color and grayscale images. Suppose there is a requirement that the SSIM Index must not be lower than 0.99. For color images, OD-2 and OD-4 modes meet this requirement, and Fig. 4(a) shows a relatively darker tone of the OD-1 smoothed color image. While for grayscale images, all three modes meet this requirement. There is no significant difference among three grayscale images shown in Fig. 4(d), Fig. 4(e), and Fig. 4(f).

For a single pixel of smoothing, the number of multiplication in the convolution depends on the size of the smoothing kernel. Suppose we have a 32-parallel accuracy reconfigurable MAC unit (i.e., with 32 approximate multipliers), and the size of the smoothing kernel is 10×10. The number of instructions executed for the processing can be calculated by Eq. (9) where N_p represents the degree of parallelism in the MAC unit. When using scalar instructions where the degree of parallelism N_p in Eq. (9) is 1, it is estimated that 400 instructions will be spent since each multiply-accumulation is regarded as 4 scalar instructions consisting of "two loading instructions + one multiplication instruction + one addition instruction". If we use the highest-parallelism OD-1 mode, the value of N_p in Eq. (9) can be 32. In this case, the number of instructions executed is expected to reduce to 16 since parallel multiply-accumulation will be implemented as 32 parallel "2 vector loading instructions + 1 vector multiply-accumulation instruction + 1 addition



(a) OD-1 (b) OD-2 (c) OD-4

(d) OD-1 (e) OD-2 (f) OD-4

Fig. 4 Smoothed color and grayscale images. If SSIM Index must not be lower than 0.99, for color images, OD-2 and OD-4 modes meet the requirement, while for grayscale images, all three modes meet the requirement.

instruction":

$$(\text{vload} \times 2 + \text{vmac} \times 1 + \text{add} \times 1) \times \left\lceil \frac{10 \times 10}{N_p} \right\rceil. \quad (9)$$

If we implement the accurate MAC unit based on the traditional accurate multiplier so that the area is equal to that of the 32-parallel accuracy reconfigurable MAC unit, the degree of parallelism is about 14 since the area of the accurate MAC unit is about 2.3 times bigger than our accuracy reconfigurable MAC unit as shown in Table 2. In this case, the value of N_p is 14, and the number of instructions required for the smoothing kernel of 10×10 is 32 which is double of the OD-1 mode. If we implement the area-efficient version which is specified as Proposed^3 in Table 2, the degree of the parallelism can be 40. In this case, the number of instructions required can be reduced to 12 which is 62% lower than the case that the accurate MAC unit is used for the image smoothing. In this case study, we can reconfigure the operating mode on demand to obtain good smoothing results for color and grayscale images, respectively, while ensuring high computational parallelism.

5.2 Simultaneous Localization and Mapping (SLAM)

Simultaneous Localization and Mapping (SLAM) is the computational problem of constructing or updating a map of an unknown environment while simultaneously estimating the robot's pose (position and orientation, etc.). The study of [19] has a high expectation of the possibility of applying approximate computing to the field of SLAM. Oh et al. [20] first applied approximate computing to matrix multiplication of two major parts of RGB-SLAM: feature extraction for visual observation and robot localization using an iterative algorithm. We use a laser-based SLAM application named LittleSLAM proposed by Tomono [21] to discuss the applicability of the proposed MAC unit. As the name implies, SLAM contains two core components: namely localization and mapping (or map construction). The core approach to robot localization in LittleSLAM is the *Iterative Closest Point (ICP)* method based on *Scan Matching* first proposed in [22]. It estimates the rigid body transformation of the robot by iteratively approximating the minimal value of the cost function by the gradient descent method. Here, the cost function is the sum of the distances between each matched point pair of the environmental landmark point cloud and the reference point cloud obtained by the robot's laser scanner. This iterative process contains extremely intensive multiply-accumulate operations and is suitable for approximation computing, so we deploy the MAC unit in the square operations in the cost function. After the robot has been localized, the mapping part converts the currently scanned environmental landmark point cloud from the robot coordinate system to the global coordinate system in dense matrix multiplication. We choose to deploy the vector accelerator in the matrix multiplication. We use two default data sets, i.e., Corridor and Hall, that come with the program as test inputs.

Table 4 Maximum and mean difference of poses in localization compared to accurate multiplication.

Data Set	Hall		Corridor	
Mode	Max (cm)	Mean (cm)	Max (cm)	Mean (cm)
OD-1	19.27	10.74	30.36	18.19
OD-2	13.40	9.51	15.72	5.74
OD-4	21.97	14.04	11.92	4.49

Table 4 shows the maximum and mean distances between the robot's pose trajectory estimated by applying the proposed MAC unit and the accurate multipliers. The choice of the OD modes has an irregular effect on the pose trajectory. In general, OD-1 mode yields a pose trajectory close enough to using the accurate multipliers. In the data set Hall, the maximum error of 19.27 cm is only 0.43% of the total walking distance of 44.64 m, while in the data set Corridor, the maximum error of 30.36 cm is only 0.46% of the total walking distance of 66.05 m. Figure 5 shows the pose trajectory estimated by OD-1 mode in data set Hall. It can be seen that the OD-1 trajectory and that using the accurate multipliers overlap to a high degree so that sufficient accuracy can be achieved by choosing OD-1 mode in the localization part of SLAM.

Table 5 shows the maximum and mean distances between the map point cloud generated by the proposed MAC unit and the accurate multipliers. The accuracy improvement brought by the proposed OD-2 and OD-4 methods is evident compared to the OD-1 mode. For the data set Hall, the map error using OD-1 mode is up to more than 50 cm. Figure 6 shows the map generated by OD-1 mode in the data set Hall. We can see the obstacles in the environment are not clear enough. The maximum error (59.79 cm) occurs at the two biggest red dots in the lower right corner. The proposed OD-2 mode reduces the mean error to 28.8% of the OD-1, making the map much closer to which generated by the accurate multipliers. The OD-4 mode even reduces

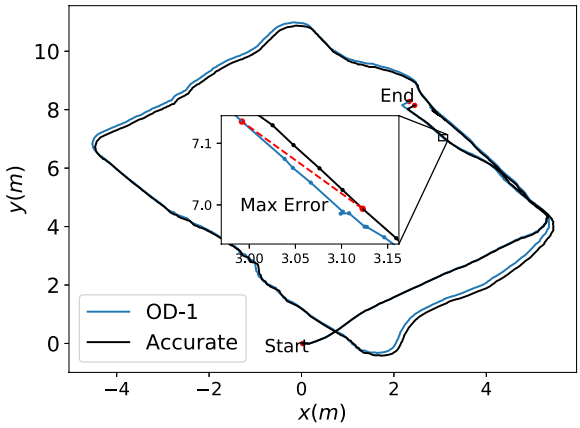


Fig. 5 Robot's pose trajectory (44.64 m) of data set Hall. The maximum error in OD-1 mode is only 0.43% of the total walking distance, which is small enough for localization.

Table 5 Maximum and mean difference of map points in mapping compared to accurate multiplication.

Data Set	Hall		Corridor	
Mode	Max (cm)	Mean (cm)	Max (cm)	Mean (cm)
OD-1	59.79	11.27	46.42	1.83
OD-2	27.50	3.25	25.12	1.00
OD-4	5.77	0.29	6.32	0.52

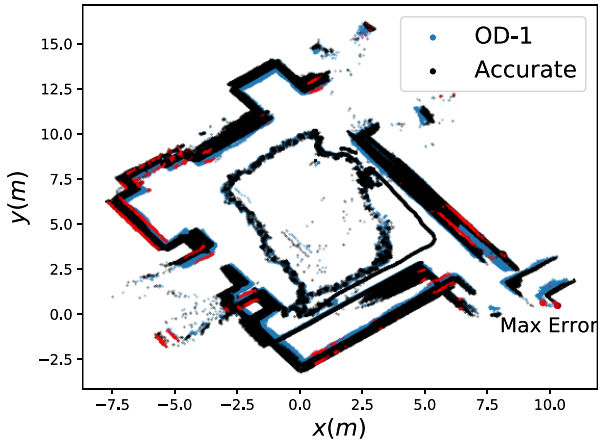


Fig. 6 Generated map of data set Hall. The red part indicates the error is over 50 cm. The obstacles are not fine enough so the higher-accuracy modes are required.

the mean error to within 1 cm. Therefore, to build more acceptable maps without losing high parallelism, the OD-2 or OD-4 modes of the proposed MAC unit are recommended.

The number of consumed multiplications in localization depends on the number of point pairs and iterations for the gradient descent method. We experimentally demonstrate that the number of iterations required to obtain the robot's pose trajectories using the proposed OD method is essentially the same or even slightly lower than the conventional method using the accurate multiplication, which is shown in Table 6. For the data set Hall, the cost function in localization contains the sum of the distances of 323 point pairs

Table 6 Iteration numbers of robot's pose trajectories with accurate multiplication and OD methods.

Data Set	Hall	Corridor
Accurate	2,047,300	4,249,198
OD-1	1,994,403	4,107,685
OD-2	1,930,185	4,046,058
OD-4	1,744,407	3,664,914

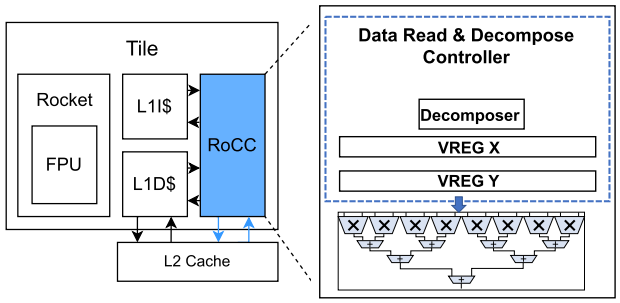
on average, i.e., 646 sets of multiply-accumulation inputs. Therefore, similar to the discussion of image smoothing, the 32-parallel MAC unit with OD-1 mode is expected to consume only 84 instructions according to Eq. (9), while in the case of using accurate MAC unit, the number will be 188 since the degree of parallelism in the accurate one is 14. If we implement the more area-efficient version which is specified as Proposed[3] in Table 2, the number of instructions expected to consume can be reduced to 68 which is 64% lower than the case that the accurate MAC unit is used for the localization in the SLAM application. In this case study, high parallelism will be limited by the circuit area if we use accurate multipliers, which will inevitably increase the execution cycles and consume more energy. Meanwhile, the high-accuracy localization and mapping will not be obtained at the same time if only the Mitchell multiplication unit is used, even though the computation is performed at a high degree of parallelism. By applying the proposed accuracy reconfigurable MAC unit in different parts of the SLAM application and reconfiguring the operating modes on-demand, it is possible to achieve high parallelism and obtain high accuracy for both localization and mapping at the same time.

6. Open Source RISC Processor Integration

This section describes an implementation of a prototype open-source RISC processor that integrates the minimum functionality of the proposed MAC unit as a vector accelerator. We also implement a software-level accuracy reconfiguration in the form of an instruction set extension.

6.1 Implementation Approach

The prototype processor that integrates the proposed vector accelerator is based on the *Rocket Chip Generator* [23], which is an open-source System-on-Chip generator that emits synthesizable RTL based on the RISC-V Instruction Set Architecture. The proposed vector accelerator is integrated in the form of *Rocket Chip Coprocessor (RoCC)*. As shown in Fig. 7(a), a tile, which is the core part in a Rocket chip, contains a Rocket base core (which supports basic RISC-V instructions), L1 cache, and a highly customizable coprocessor (RoCC). This RoCC communicates with the base core and L1 cache via defined interfaces. The data width of the L1 data cache interface is equal to the word length of the chip, and the data is read in a "request-response" way. When a cache miss occurs, carrying data takes up to 30 clock cycles. Since our implementation is a vector accelerator, we mount a 4-channel DMA node for



(a) RoCC in Rocket Tile. (b) RoCC implementation.

Fig. 7 The RoCC uses interfaces to communicate with other parts in the tile, and use DMA node to speed up data reading directly from L2 cache.

the RoCC to read data directly from the larger L2 cache to speed up the data loading process. When the amount of parallel data is high, the read speed of the DMA method will be much faster than the L1 interface reading. The specific RoCC implementation of the proposed vector accelerator is shown in Fig. 7(b). The decomposer is used to decompose the contiguous data read by the DMA node based on the instruction machine code passed in the L1 instruction cache interface. The decomposed (or in OD-1 mode, not decomposed) data is filled into two vector registers, respectively. The multiplication/multiply-accumulation unit consisting of Mitchell's approximate multipliers with area-saving advantage calculates the results of operations on the data in the two vector registers.

In terms of instructions, we implement an OD mode control instruction that determines the operating mode of the proposed vector accelerator and the use of vector registers. In addition, vector loading instruction, vector multiplication instruction, and vector multiply-accumulation instruction are also implemented, respectively.

6.2 Clock Cycle Reduction Verification

This section describes the clock cycle reduction of the practical applications provided by the RTL design. We set the parallelism of the hardware design to 32, i.e., the vector accelerator contains 32 parallel Mitchell approximation multipliers. The Verilator simulator converts the hardware design into a clock-accurate program modeled in C, and its expected behavior will be the same as the physical processor. We use the same C program of image smoothing application in Sect. 5.1 as an example to show the effect of the clock cycle reduction of the prototype processor. We compare the clock cycles consumed by smoothing the image using the three modes of operation and the clock cycles consumed without the accelerator (i.e., using scalar instructions). Due to the performance limitation of the test platform, the image sizes being processed are 10×10 and 100×100 pixels.

First of all, using the cycle-accurate simulation, we confirmed the correct operation of the accuracy reconfigurable vector accelerator implemented based on the RoCC. It correctly works over the 3 operation modes corresponding to OD-1, OD-2, and OD-4. The number of clock cycles for the 3 modes are shown in Table 7. In addition, the clock cycles for the scalar MAC unit which is originally built in the Rocket core are also shown for reference. For a 10×10 image, the highest-parallelism OD-1 mode reduces the clock cycles by up to 77.39% over the result of the scalar MAC unit. The OD-2 mode reduces the consumed cycles by 69.94%. Even the OD-4 mode, which sacrifices the most parallelism, reduces the clock cycles by 52.14%. A similar reduction also occurs in the 100×100 image, where the clock cycle reductions of OD-1, OD-2, and OD-4 modes are 77.89%, 68.88%, and 51.53%, respectively. Ideally, the clock cycles could have been reduced more significantly since the number of multipliers integrated in the RoCC is 32. One of the reasons why the parallelism of the 32-parallel MAC unit is not fully exploited is that the memory bandwidth between the data cache and the RoCC is limited. The reduction in the number of cycles to application execution will be improved if the concatenation of data I/O of the coprocessor is implemented, i.e., high bandwidth data can be read and written as a vector per cycle.

Table 7 Clock cycles consumed in image smoothing with the proposed accelerator prototype and scalar instructions.

Image Size	10×10	100×100
Scalar	35,927	9,134,537
OD-1	8,124	2,019,374
OD-2	10,799	2,842,359
OD-4	17,195	4,427,465

7. Conclusions

This paper proposed a novel operand decomposition (OD) method that improves the accuracy of the existing original approximate multiplier. Based on the proposed OD method, we also proposed an accuracy reconfigurable vector accelerator. It can reconfigure multiple accuracies on demand by a trade-off to meet various accuracy requirements while keeping high computational parallelism. The hardware evaluation shows the proposed MAC unit significantly reduced the area to only 42.88% of the accurate multiplication unit. The deployment on practical applications like image smoothing and SLAM shows excellent accuracy performance of the proposed MAC unit. We can reconfigure the operating mode on-demand to obtain results with slightly lower accuracy but the highest computational parallelism or sacrifice a certain degree of parallelism to get high accuracy. The proposed MAC unit addresses the low parallelism of accurate multipliers and the low accuracy of original approximate logarithmic multipliers. It achieves on-demand accuracy and high parallelism computation with only a small area overhead. We also designed a prototype processor that integrates the minimum functionality of the proposed MAC unit as a vector accelerator based on the Rocket Chip Coprocessor (RoCC). Using the cycle-accurate simulation, we confirmed the correct operation of the accuracy reconfigurable vector accelerator implemented based on the RoCC. It correctly works over the

3 operation modes corresponding to OD-1, OD-2, and OD-4. We also experimentally confirmed the OD-1 mode with the highest computational parallelism reduced the number of clock cycles by up to 77.89% compared to the original scalar instructions. However, the results show that the OD-1 mode does not fully exploit the parallelism of the accelerator because of the limitation of the memory bandwidth between the data cache and the RoCC. Our future work is focused on improving the memory bandwidth by implementing vector load and store instructions where high bandwidth data can be read and written.

Acknowledgments

This work has been supported by JST CREST Grant Number JPMJCR18K1, and VLSI Design and Education Center (VDEC), the University of Tokyo with collaboration with SYNOPSYS Corporation and Mentor Graphics Corporation. The authors thank Dr. Jun Shiomi for assistance with the physical layout design of the vector accelerator.

References

[1] L. Hou, Y. Masuda, and T. Ishihara, "An accuracy reconfigurable vector accelerator based on approximate logarithmic multipliers," Proc. 27th ASP-DAC, pp.568–573, Jan. 2022.
[2] W. Liu, F. Lombardi, and M. Shulte, "A retrospective and prospective view of approximate computing," Proc. IEEE, vol.108, no.3, pp.394–399, March 2020.
[3] R. Zendegani, M. Kamal, M. Bahadori, A. Afzali-Kusha, and M. Pedram, "RoBA multiplier: A rounding-based approximate multiplier for high-speed yet energy-efficient digital signal processing," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol.25, no.2, pp.393–401, Feb. 2016.
[4] H. Jiang, C. Liu, F. Lombardi, and J. Han, "Low-power approximate unsigned multipliers with configurable error recovery," IEEE Trans. Circuits Syst. I, Reg. Papers, vol.66, no.1, pp.189–202, 2018.
[5] T. Yang, T. Ukezono, and T. Sato, "Low-power and high-speed approximate multiplier design with a tree compressor," Proc. IEEE International Conference on Computer Design (ICCD), pp.89–96, Nov. 2017.
[6] D. Esposito, A.G.M. Strollo, E. Napoli, D. De Caro, and N. Petra, "Approximate multipliers based on new approximate compressors," IEEE Trans. Circuits Syst. I, Reg. Papers, vol.65, no.12, pp.4169–4182, 2018.
[7] J.N. Mitchell, "Computer multiplication and division using binary logarithms," IRE Trans. Electron. Comput., vol.EC-11, no.4, pp.512–517, Aug. 1962.
[8] M.S. Kim, A.A. Del Barrio, R. Hermida, and N. Bagherzadeh, "Low-power implementation of Mitchell's approximate logarithmic multiplication for convolutional neural networks," Proc. 23rd Asia and South Pacific Design Automation Conference (ASP-DAC), pp.617–622, Jan. 2018.
[9] M.S. Kim, A.A. Del Barrio, L.T. Oliveira, R. Hermida, and N. Bagherzadeh, "Efficient Mitchell's approximate log multipliers for convolutional neural networks," IEEE Trans. Comput., vol.68, no.5, pp.660–675, 2018.
[10] Z. Babić, A. Avramović, and P. Bulić, "An iterative logarithmic multiplier," Microprocessors and Microsystems, vol.35, no.1, pp.23–33, 2011.
[11] M.S. Ansari, B.F. Cockburn, and J. Han, "A hardware-efficient logarithmic multiplier with improved accuracy," Proc. Design, Automation and Test in Europe Conference (DATE), pp.928–931, March 2019.
[12] N. Alla and S.E. Ahmed, "An area and delay efficient logarithmic multiplier," Proc. International Conference on Contemporary Computing and Applications (IC3A), pp.169–174, Feb. 2020.
[13] R. Pilipović, P. Bulić, and U. Lotrič, "A two-stage operand trimming approximate logarithmic multiplier," IEEE Trans. Circuits Syst. I, Reg. Papers, vol.68, no.6, pp.2535–2545, June 2021.
[14] V. Mahalingam and N. Ranganathan, "Improving accuracy in Mitchell's logarithmic multiplication using operand decomposition," IEEE Trans. Comput., vol.55, no.12, pp.1523–1535, Dec. 2006.
[15] D. Nandan, J. Kanungo, and A. Mahajan, "An efficient VLSI architecture design for logarithmic multiplication by using the improved operand decomposition," Integration, vol.58, pp.134–141, June 2017.
[16] C. Liu, J. Han, and F. Lombardi, "A low-power, high-performance approximate multiplier with configurable partial error recovery," Proc. Design, Automation and Test in Europe Conference (DATE), pp.1–4, March 2014.
[17] R.C. Gonzalez, R.E. Woods, and B.R. Masters, Digital Image Processing, 2009.
[18] Z. Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," IEEE Trans. Image Process, vol.13, no.4, pp.600–612, April 2004.
[19] A. Agrawal, J. Choi, K. Gopalakrishnan, S. Gupta, R. Nair, J. Oh, D.A. Prener, S. Shukla, V. Srinivasan, and Z. Sura, "Approximate computing: Challenges and opportunities," Proc. IEEE International Conference on Rebooting Computing (ICRC), pp.1–8, Oct. 2016.
[20] J. Oh, J. Choi, G.C. Januario, and K. Gopalakrishnan, "Energy-efficient simultaneous localization and mapping via compounded approximate computing," Proc. IEEE International Workshop on Signal Processing Systems, pp.51–56, Oct. 2016.
[21] M. Tomono, "A scan matching method using euclidean invariant signature for global localization and map building," Proc. IEEE International Conference on Robotics and Automation, pp.866–871, April 2004.
[22] Y. Chen and G. Medioni, "Object modelling by registration of multiple range images," Image and Vision Computing, vol.10, no.3, pp.145–155, 1992.
[23] K. Asanović, R. Avižienis, J. Bachrach, S. Beamer, D. Biancolin, C. Celio, H. Cook, D. Dabbelt, J. Hauser, A. Izraelevitz, et al., "The rocket chip generator," Technical Report, UCB/EECS-2016-17, EECS Dept., UC Berkeley, April 2016.

Lingxiao Hou is a master's student majoring in computer and software systems at Nagoya University. His main research focus is on designing high-speed and energy-efficient hardware based on approximate computing. His current work is the design of highly parallel approximate multiplication/multiplication-accumulation acceleration units that save area and power. His work also focuses on general-purpose processor integration of this accelerator unit and instruction set level software toolchain development. He aims to earn a master's degree in March 2022.

Yutaka Masuda received the B.E., M.E., and Ph.D. degrees in Information Systems Engineering from the Osaka University, Osaka, Japan, in 2014, 2016, and 2019, respectively. He is currently an Assistant Professor in Center for Embedded Computing Systems, Graduate School of Informatics, Nagoya University. His research interests include low-power circuit design. He serves on the Technical Program Committee of international conferences including ASP-DAC. He is a member of IEEE, IEICE, and IPSJ.

Tohru Ishihara received his Dr. Eng. degree in computer science from Kyushu University in 2000. For the next three years, he was a Research Associate in the University of Tokyo. From 2003 to 2005, he was with Fujitsu Laboratories of America as a Research Staff of an Advanced CAD Technology Group. From 2005 to 2011, he was with Kyushu University and for the next seven years he was with Kyoto University as an Associate Professor. In October 2018, he joined Nagoya University where he is currently a Professor in the Department of Computing and Software Systems. His research interests include low-power design methodologies and power management techniques for embedded systems. Dr. Ishihara is a member of the IEEE, ACM and IPSJ.