PAPER

A Feasibility Study of Multi-Domain Stochastic Computing Circuit

Tati ERLINA[†a)], *Nonmember*, Renyuan ZHANG[†], *Member*, *and* Yasuhiko NAKASHIMA[†], *Fellow*

SUMMARY An efficient approximate computing circuit is developed for polynomial functions through the hybrid of analog and stochastic domains. Different from the ordinary time-based stochastic computing (TBSC), the proposed circuit exploits not only the duty cycle of pulses but also the pulse strength of the analog current to carry information for multiplications. The accumulation of many multiplications is performed by merely collecting the stochastic-current. As the calculation depth increases, the growth of latency (while summations), signal power weakening, and disparity of output signals (while multiplications) are substantially avoidable in contrast to that in the conventional TBSC. Furthermore, the calculation range spreads to bipolar infinite without scaling, theoretically. The proposed multi-domain stochastic computing (MDSC) is designed and simulated in a 0.18 μm CMOS technology by employing a set of current mirrors and an improved scheme of the TBSC circuit based on the Neuron-MOS mechanism. For proof-of-concept, the multiply and accumulate calculations (MACs) are implemented, achieving an average accuracy of 95.3%. More importantly, the transistor counting, power consumption, and latency decrease to 6.1%, 55.4%, and 4.2% of the state-of-art TBSC circuit, respectively. The robustness against temperature and process variations is also investigated and presented in detail.

key words: *stochastic computing, multi-domain, approximate computing, MAC*

1. Introduction

Along with the explosion of artificial intelligence (AI), the demands of the massive computations grow excessively in recent years. At the application-end, one of the critical challenges lies in implementing sophisticated algorithms such as an artificial neural network (ANN) onto the Internet of Things (IoT) devices [1]. It is evident that most of the computations for ANN, such as multiply and accumulate calculations (MACs), are time- and resource-hungry [2] and usually expected to perform in parallel for the high speed. Unfortunately, the circuit integration hardly keeps growing to increase computing units since the benefits from Moore's Law will vanish soon. Therefore, alternative computational implementations with low cost are seriously on demand. Trading the computational accuracy (reasonable for most of AI tasks) by compact hardware resources and low power, the approximate computing (AC) technology is known as one of the prospective solutions which offer an efficient implementation and acceptable quality of services. A common strategy of the AC is accomplished widely by building specific analog computing circuits [3], which provide various beneficial features such as compact size and low power. However, most of the analog computing circuits are sensitive to the process, voltage, and temperature (PVT) variations [4], which strictly limits the practical application fields of analog computing.

To achieve robust noise-tolerance, the stochastic computing (SC) was proposed decades ago [5]–[7]. Early efforts of SC were mainly made based on bit-stream. Representing the data by possibility, the multiplication and summation are implemented by extremely compact circuits (even one single logic gate). Although the bit-stream type of SC is low cost and noise-tolerance, it can hardly be considered as a perfect solution because of the long latency, reduced data range, and the inherent limitations on computational depth. Besides, a necessary module called stochastic number generator (SNG) is quite expensive in hardware resources and power [8], eating up the benefit on the low cost of the SC. As an improved scheme of SC, the time-based representation carries information by leveraging pulse width (in terms of duty cycle) instead of counting bits in a stream [9], [10]. Indeed, the time-based stochastic computing (TBSC) technology helps to shorten the latency and reduce the power consumption somehow. Our early reports [11], [12] show that a well designed TBSC circuit wins the costly trade-off between efficiency and accuracy compared to various AC strategies. However, conventional TBSC, including our early works, still suffer from reduced range and limitations on computational depth. When performing a complex function such as polynomial, the latency increases significantly, and the signal becomes sparse. In real-world application of the MAC, for example, it is difficult to practically apply SC (regardless of schemes) without additional buffering mechanisms.

In this work, a hybrid implementation of stochastic computing is proposed across analog and time-based stochastic domains. By tuning the duty cycle and pulse strength, the signal in terms of the analog current pulse is entangled as the multiplication; the summation/accumulation is realized by simply collecting the entangled current. In this manner, the implementation of an extended polynomial function does not lead to the sparse signal; and latency does not significantly increase since it is unnecessary to vary the frequency of pulse for each item-operation. Naturally, the representation range is elongated to infinite by accumulating current. By employing the Neuron-MOS mechanism [13],

Manuscript received March 25, 2020.
Manuscript revised July 21, 2020.
Manuscript publicized October 29, 2020.

[†]The authors are with the Graduate School of Information Science, Nara Institute of Science and Technology (NAIST), Ikoma-shi, 630–0192 Japan.
a) E-mail: tati.erlina.su4@is.naist.jp

DOI: 10.1587/transele.2020ECP5015

a duty cycle tuning module is designed for switching the current source representing the specific analog value. The summation is achieved by accumulating multiple entangled items. For the convenience of observation, a detector circuit is also designed to translate the MDSC values into voltage mode. The entire circuit is designed and simulated in a 0.18 μm CMOS technology. The MAC with four variables is introduced for proof-of-concept. From the circuit simulation results, an average accuracy of 95.3% is achieved. The transistor counting, power, and latency of the proposed MDSC circuit are 152, 179.6 μW, and 90 ns, which are 6.1%, 55.4%, and 4.2% of the state-of-art TBSC circuit in a 45 nm technology, respectively. To investigate robustness against variations, the random changes on transistors' threshold, size, and temperature are simulated in Monte Carlo analysis. The distribution features and analysis of sensitivity are presented in detail with various input patterns.

The rest of this paper is organized as follows: in Sect. 2, the fundamentals of SC is reviewed; Sect. 3 presents our proposed MDSC circuit and its operational principle; in Sect. 4, a toy-example of MAC is implemented, and the circuit simulation results are shown; the conclusion is made in Sect. 5.

2. Preliminary: Stochastic Computing Fundamentals

2.1 Bit-Stream-Based Stochastic Computing

Originally, SC encodes a value as a sequence of randomized bitstreams called a stochastic number (SN) [14]. The data being represented are dependent upon the occurrence of logic '1' [15] and the format being used in the bit streams. For instance, in unipolar format where representation of a number x is bounded by ($0 \leq x \leq 1$), all distinct arrangement bits of SNs $\{1,1,0,0,0\}$, $\{0,1,1,0,0\}$, $\{0,0,1,1,0\}$, $\{0,0,0,1,1\}$ and $\{1,0,0,0,1\}$ describe the same real number, i.e., $x = 0.4$. On the other hand, the exact composition of the sequences are interpreted differently, i.e., $x = -0.2$, in bipolar format where ($-1 \leq x \leq 1$). Thus, the probability of each bit to be 1 in bit streams of unipolar and bipolar format are $P(X = 1) = x$, and $P(X = 1) = \frac{x+1}{2}$, respectively. In order to generate the SNs, an SNG comprises of a random number generator (RNG) and a comparator are usually exploited. Figure 1 shows the SNG of the bit-stream based SC and two commonly used elements in the stochastic operations.

One key benefit of the stochastic domain is the capability to utilize a simple logic to conduct arithmetic operations, which implies low hardware cost and power consumption. In accomplishing an addition, for example, SC can exploit a multiplexer (MUX) [16] and an OR gate [17]. The MUX performs a scaled addition by randomly selecting one input at a time as part of the resulted output and ignoring all other inputs. Thus, when the input size is large, the operation incurs a significant accuracy loss. On the contrary, the output of an addition using the OR gate is not scaled, but the result saturates at 1. In order to perform a multiplication of two unipolar and bipolar stochastic streams, an AND gate and an exclusive NOR (XNOR) circuit are exploited [18], respectively.

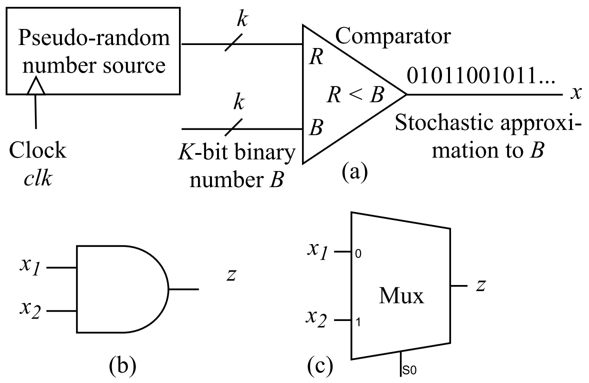


Fig. 1 Key stochastic components: (a) the stochastic number generator (SNG), (b) the multiplier, (c) the scaled adder [16]

Another compelling advantage is fault tolerance; the SC endures a huge number of errors while maintaining equivalent performance. It corresponds to the unary representation of the SNs where every bit in a stream is equally weighted, suggesting that a bit flip would only cause a small error. The maximum error from one bit flip in the SC is $\frac{1}{n}$ where n is the bitstream length of an SN. As a comparison, the same number of soft errors trigger inaccuracy as much 2^{n-1} [19] in n size binary radix. Hence, stochastic circuits can provide more accurate results than conventional binary circuits under severe error conditions [20]. Additionally, SC offers the capability of changing the computation time and precision dynamically without any hardware modifications [21]; the accuracy increases as the length of bitstreams rises.

Unfortunately, SC also brings about some drawbacks; for example, it suffers from some accuracy lost when signal correlation requirement or randomness is not adequately satisfied, or simply because of the existence of random fluctuations. Besides, representing sufficiently accurate numbers requires a lengthy bitstream, which undoubtedly causes a high latency in the system.

2.2 Time-Based Stochastic Computing (TBSC)

A TBSC circuit represents an SN using the duty cycle, i.e., the ratio between pulse width and the period of a PWM signal [10]. Figure 2 (a) and (b) show examples of two signals with duty cycle of 80% (5 ns period) and 50% (6 ns period), which portray certain values of SNs, namely $S_1 = 0.8$ and $S_2 = 0.5$ respectively. A time-based SNG, so-called PWM generator [9], as illustrated in Fig. 3, produces the SNs by regulating the duty cycle and fine-tuning the frequency of each signal. More specifically, a duty cycle is determined based on the amount of current received from the sensing circuit. At the same time, the clock generator adjusts the frequency by activating a certain number of inverters in the ring oscillator structure. In this case, as many as 89 inverters are mobilized to produce a periodic signal which has a 1 ns period.

Similar to the conventional SC, the TBSC circuits perform multiplication and addition of unipolar SNs by utilizing an AND gate and a MUX, respectively. However, to produce a highly accurate resulted output, the dependency and correlation between input signals must be avoided by employing distinctive SNGs for each input signal. At the same time, the output signal has to be observed for the LCM (least common multiple) period of its input signals. For instance, if input signals of a multiplication operation have 5 ns (Fig. 2 (a)) and 6 ns (Fig. 2 (b)) periods, then the resulted output with the highest accuracy can only be achieved by observing the output signal at least for 30 ns. The ratio between the total time of the output signal is high (12 ns) and the LCM time (30 ns) of input signal periods, assemble a resulted output of 0.4, as presented in Fig. 2 (c).

In contrast to multiplication, dependency and correlation between signal representing operands are highly desirable in performing addition. These input signals, however, should be independent and uncorrelated with the select signal of the MUX circuit performing the addition. In such a way, the resulted output of the scaled addition eventually is also examined at least for the LCM period of input signals and the select signal.

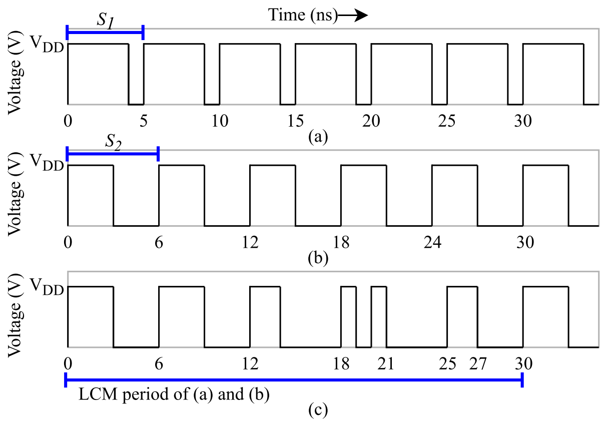


Fig. 2 PWM signals with 80% and 50% duty cycle represent operands (a) $S_1 = 0.8$, (b) $S_2 = 0.5$, respectively, and (c) the output signal of ($S_1 \times S_2$) in TBSC

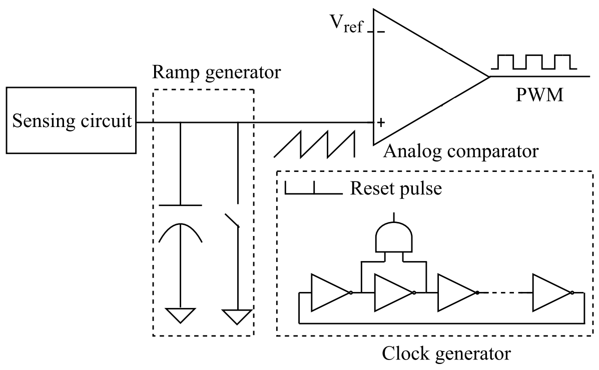


Fig. 3 Design of the PWM Generator in TBSC [9]

3. The Proposed Method

3.1 Representations and Operations

The MDSC designates numbers by utilizing the analog and stochastic domains. The former expresses a value through the pulse strength of an analog signal, which is the ratio between pulse height (P_{height}) and a predetermined reference current (I_{ref}); $X = \frac{P_{height}}{I_{ref}}$. The latter denotes a number with a duty cycle or proportion of pulse width (P_{width}) to the period (t) of a PWM signal; $W = \frac{P_{width}}{t}$. An example of signals representing numbers in the analog (X) and stochastic domain (W) is shown in Fig. 4 (a) and (b), respectively.

Further, as shown in Eq. (1), a resulted output (O) of operation in the MDSC is governed by the ratio between the so-called duty area (A_{duty}) and reference area (A_{ref}), in which the P_{height} and the P_{width} of an output signal are taken into account, simultaneously.

$$O = \frac{A_{duty}}{A_{ref}} = \frac{P_{height} \times P_{width}}{I_{ref} \times LCM}, \qquad (1)$$

where LCM is the least common multiple of input signals.

The concurrent engagement of both domains in designating the resulted output allows the latency in the MDSC is shorter than the TBSC counterpart. As an illustration, the resulted output of 2-operand multiplication in the MDSC is

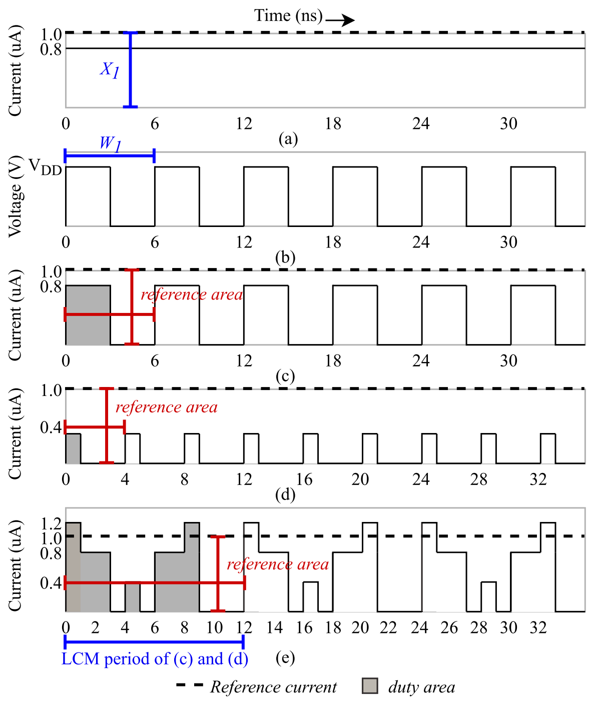


Fig. 4 Example of signals representing numbers in the MDSC: (a) signified in analog domain ($X_1 = 0.8$) (b) denoted in stochastic domain ($W_1 = 0.5$), output signals of (c) $X_1 \times W_1$, (d) ($X_2 = 0.4$) $\times$ ($W_2 = 0.25$) (input signals not shown), (e) The output signal of MAC operation $((X_1 \times W_1) + (X_2 \times W_2))$

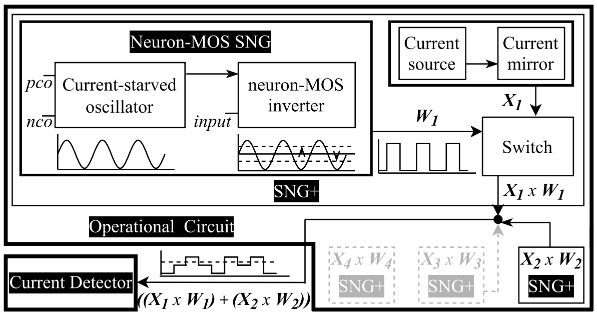


Fig. 5 Block diagram of the MDSC circuit for MAC

merely dependent on the period of one input signal, (W_1 = 6 ns) and (W_2 = 4 ns) for resulted output in Fig. 4 (c) and (d), respectively. It is contradictory to an equivalent operation in the TBSC shown previously in Fig. 2 (c), in which all input signals are in the stochastic domain. Hence, each period (S_1 = 5 ns, S_2 = 6 ns) contributes to the LCM period (30 ns) of signal observation to gain a reasonably accurate output, leading to high latency in the TBSC.

A similar principle is also valid in the 4-variable MAC operation, where the output signal of the MDSC in Fig. 4 (e) only need to be observed for the LCM period (12 ns) of two signals denoting W_1 and W_2. The corresponding operation in the TBSC, as a comparison, should be assessed for the LCM period of four input signals, causing an explosion in the latency and sparser output signal. Subsequently, the MDSC scheme presents more prevalent merits as the number of operands escalates, overcoming some essential problems in the TBSC, namely the rapid growth of latency and the disparity of output signals.

Basically, the proposed method capable of conducting all operations possible in either conventional (bit stream-based) or time-based stochastic computing. One significant difference is that the resulted outputs from summations in the proposed method are not scaled. Hence, the range of resulted values can theoretically be extended to any range of real numbers.

3.2 Hardware Design

Primarily, the MDSC hardware scheme is organized as an operational circuit and a current detector, as shown in Fig. 5. The former aims to produce SNs as well as perform arithmetic operations, while the latter functions to ease the observation of the output signals. Further, the operational circuit comprises of a collection of SNG+, which exploit the neuron-MOS SNG, a current source and a current mirror. In this manner, one particular SNG+ produces two numbers, i.e., in stochastic and the analog domain, concurrently and conducts the multiplication between both operands.

The stochastic element is generated by fine-tuning the frequency of oscillation based on *pco* and *nco* voltage, and changing the DC level of a particular *input* in the neuron-MOS SNG sub-circuit. This scheme avoids the complex implementation of a comparator and massive utilization of

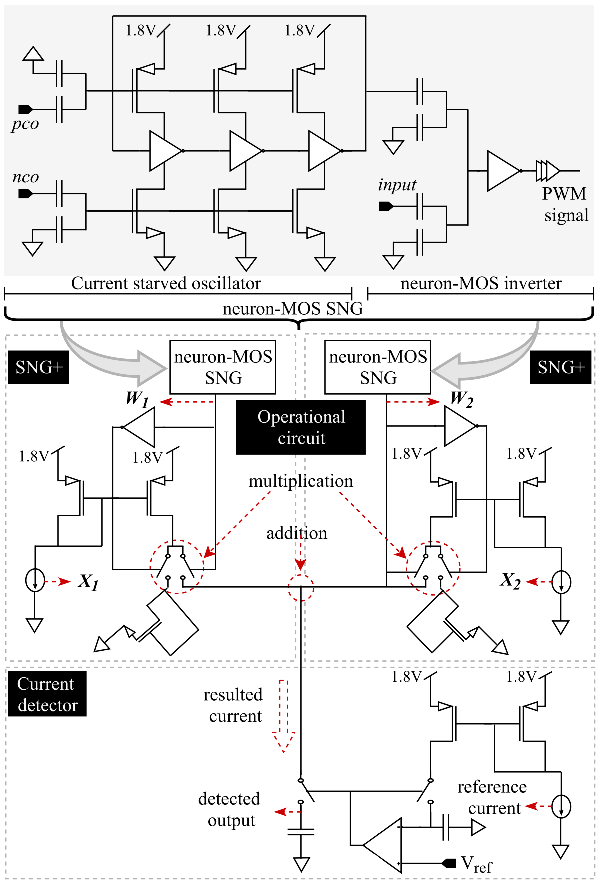


Fig. 6 A configuration of the MDSC circuit consisting of an operational circuit with two SNG+ and a current detector

inverters in regulating the frequency of the PWM signal representing SNs as that in the TBSC. The analog item, on the other side, derives from the current source, which is connected to the current mirror component. Eventually, the SNG+ conducts the multiplication between both numbers by switching the analog-based input signals.

Instead of relying upon a specific circuit explicitly, the accumulation operation merely takes advantage of the Kirchhoff Current Law (KCL). Hence, the system avoids obtaining a scaled, or an approximated resulted output. These types of outcomes have become a critical source of failure in either bit-stream based SC and TBSC implementation, especially when applying the NN algorithm, which requires many additions.

A MAC operation in the MDSC is carried out by exploiting multiple SNG+. Specifically, the number of SNG+ circuits activated at a particular time is dependent upon a target number of operands involved in the operations. The MAC of four variables illustrated in Fig. 4 (e), for example, can be achieved by using an arrangement containing two SNG+ circuits shown in Fig. 6. For completing an operation consisting of more significant operand numbers, one could activate an additional SNG+ and plug it to a predetermined node.

For convenience reason, a current detector interprets a signal from the operational circuit as a detected output by converting the output current into voltage. A comparator and two capacitors support the translation mechanism. The comparator serves as a controller of the charging time of two identical yet separated capacitors. On the one hand, the reference current of the detector charges the first capacitor connected to the positive input of the comparator through a simple current mirror and a switch. On the other hand, the resulted current from the operational circuit loads the second capacitor. Once the voltage in the positive input is equal to a reference voltage in the negative input of the comparator, the charging process in both capacitors stops momentarily. Then, the voltage on the second capacitor becomes the detected output of the operations.

Adjusting the arrangement of the SNG+, adding and adding few current mirrors to facilitate variables with negative values, the operational circuit part of the MDSC can be exploited to implement a more complicated polynomial function such as Maclaurin polynomial. Referring to [22], a nonlinear function such as $ln(1 + ax)$ can be approximated using a 5th-order Maclaurin polynomial as presented in Eq. (2).

Figure 7 shows the Maclaurin polynomial implementation of $ln(1 + ax)$, where $a = 1$. Each grayed portion of the operational circuit realizes every term in the equation. Coefficients in the terms are represented by the current component, while the stochastic domain of the SNG+ denotes the variable. An additional component such as a distinctive form of the current mirror in the circuit facilitate the multiplication between terms and support representation of negative value in the equation.

$$ln(1 + ax) \approx ax - \frac{a^2x^2}{2} + \frac{a^3x^3}{3} - \frac{a^4x^4}{4} + \frac{a^5x^5}{5} \qquad (2)$$

Similar to the MAC operation, every term in the Maclaurin polynomial is implemented by exploiting SNG+. Following the basic structure of the SNG+, each term in the equation is treated as a multiplication between coefficient and variable. High order variables are multiplied repeatedly using the current mirror component of the SNG+. For instance, the term $\frac{a^2x^2}{2}$ is multiplied as $(\frac{1}{2} \times x) \times x$. Afterward, the resulted output of all terms is integrated through a set of current mirrors that enable charging and discharging states, thus facilitating terms with negative values. The total current from the integration is then connected to the current detector.

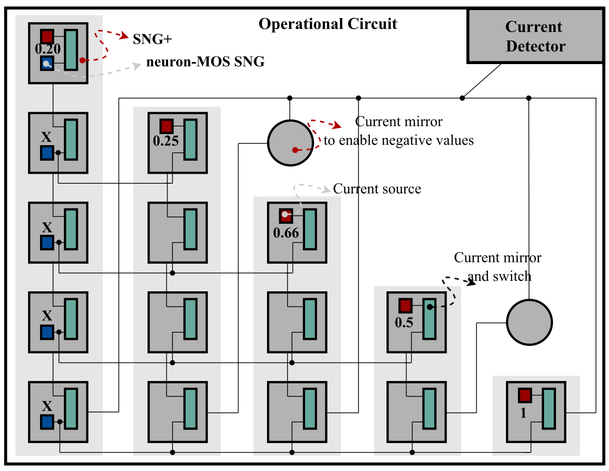


Fig. 7 Block diagram of the MDSC circuit for $5 - th$ order Maclaurin polynomial implementing $ln(1 + x)$

4. Experimental Results

For proof-of-concept, the proposed circuit was implemented in a 0.18 μm CMOS technology using HSPICE. The performance in terms of accuracy and the robustness of the circuit in response to temperature and process variations were verified by conducting numerous operations.

4.1 Accuracy Assessment and Sampling Method

In assessing the accuracy, the experiment was managed hierarchically. For every stage, the performance is expressed in terms of the average accuracy as a reverse of the mean absolute error (MAE), which is calculated using Eq. (3). First, observing signals which depict 11 distinctive values in both analog and stochastic domains, we obtained results shown in Fig. 8 with the average accuracy of 98.6% and 96.4%, respectively.

$$MAE = \frac{\sum_{i=1}^{n} |z_i - y_i|}{n} \qquad (3)$$

where *MAE*, n, z_i and y_i are the *mean absolute error*, *number of sample*, *expected output* and *detected output*, respectively.

Secondly, 2-variable multiplication was performed by exploiting a single SNG+. Assuming that each representation consists of 11 members (from 0 to 1 with step 0.1), the operation was carried out for every possible combination of the values in both representations, making 121 diverse operations with the average accuracy of 96.6% or MAE of 3.4%, which distribution is shown in Fig. 9.

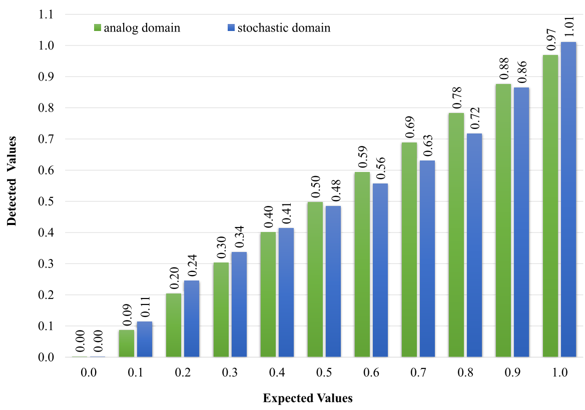


Fig. 8 Detection of 11 distinctive values in the analog and stochastic domains

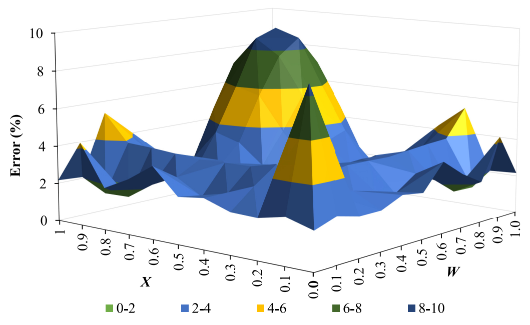


Fig. 9 The distribution of MAE in 121 multiplication operations

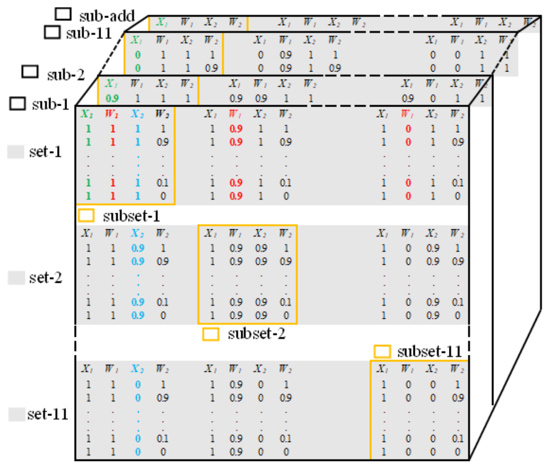


Fig. 10 Organization of population and the selected samples of the 4-variable MAC operations

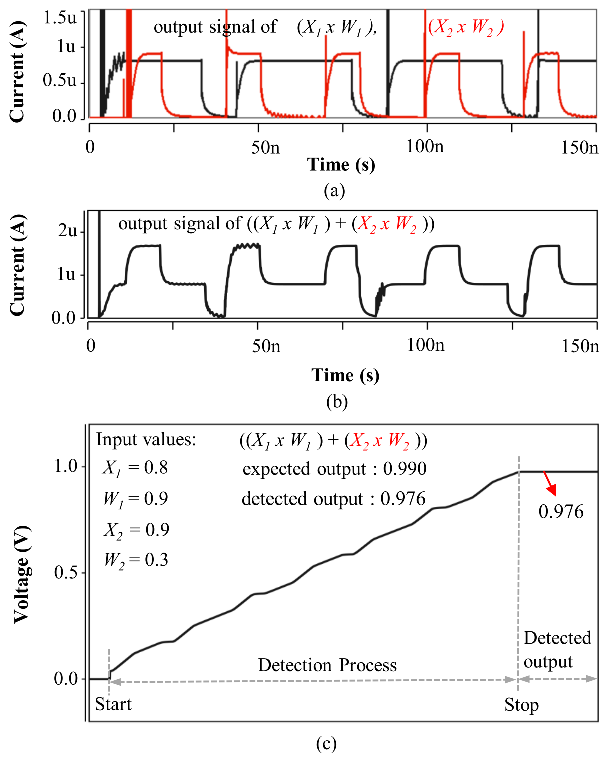


Fig. 11 Example of the simulation signals: (a) two output signals of multiplication between two pairs of input values (analog and stochastic domain), (b) the output signal of MAC operation of the two signals in (a), (c) a signal representing the detected output of the current detector

Furthermore, the performance of the proposed circuit in completing 4-variable MAC operations was evaluated. Since each variable in the MAC operations could represent any SNs between 0 and 1, the overall combination of variables as the population members in operation becomes unlimited, testing all population members would be impossible. Therefore, we accomplished the experiment in a set of selected samples by following the three stages of the statistical method called systematic sampling [23] and regularizing the values from 0 to 1 with step 0.1, resulting 11 numbers for each variable.

Calculating the number of population. Given that each variable has 11 primary elements, possible combination of values in $((X_1 \text{ x } W_1) + (X_2 \text{ x } W_2))$, are $P_n = n^k = 11^4$ embodying 14641 population members.

Identifying and organizing the population. The population members were identified and organized in such a way to avoid redundancy. We grouped the members into *sub*, *set*, and *subset*, as exhibited in Fig. 10. These operations are then organized into 11 subs (by changing the value of one particular operand at a time) of 1331 operations each (organized as 11 sets of 11 subsets which is further elaborated to 11 operations in each subset).

Selecting samples. Statistically, two determining factors in producing representative samples are the sample frame (the origin of samples in an organized population) and the sample size. Aiming to select the samples which evenly distributed in each hierarchy of the entire population, we select 121 operations in the diagonal positions of every sub (marked by the yellow rectangle in Fig. 10), harvesting 1331 operations in which is later added by another 121 operations from an additional sub in order to make up 1452 operation (about 10% of the identified population which statistically is considered as a reasonable proportion of representative samples).

Figure 11 illustrates an example of a 4-variable MAC operation. Two output signals of operation $(X_1 \text{ x } W_1)$ and $(X_2 \text{ x } W_2)$ is provided in Fig. 11 (a). Both signals are accumulated and measured for the LCM period (90 ns) of stochastic domain-based (W_1 = 30 ns, W_2 = 45 ns) representation. The output signal of the MAC operation is shown in Fig. 11 (b) and interpreted by converting the resulted current into voltage using the current detector. The output signal of the detector appears in Fig. 11 (c), showing the detected output 0.976.

The distribution of the MAE in every *subset* constituting the total sample, are shown in Fig. 12. In general, the inaccuracy is diverse and highly dependent on the expected output; the higher the expected output of operations, the higher the error rate. Besides that, many other factors could contribute to the variation of error; one of them is the detection mechanism in the current detector. Overall, the

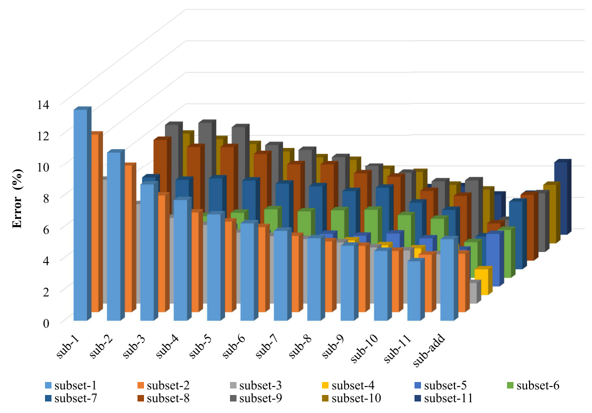


Fig. 12 Mean absolute error (MAE) of 1452 MAC operations organized in 12 *subs*

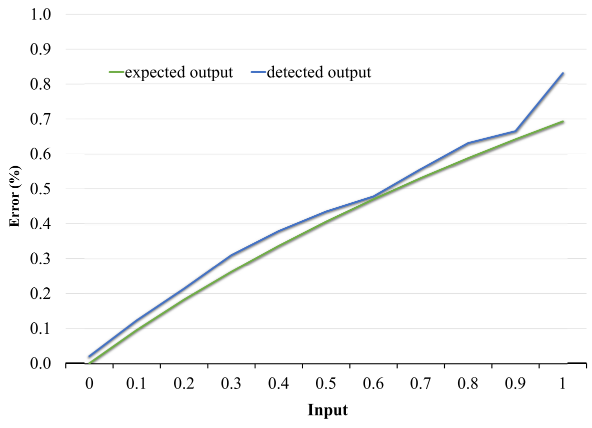


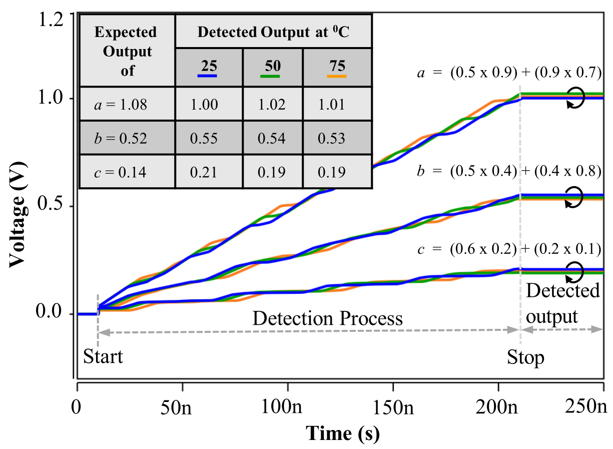
Fig. 13 Expected and detected output of $5 - th$ Maclaurin polynomial implementing $ln(1 + x)$

MAE of the operations is 4.7% or the average accuracy of 95.3%.

Furthermore, the performance of the proposed method in terms of accuracy, is evaluated in resolving a complex polynomial function. The expected and detected output of the $5 - th$ order Maclaurin polynomial approximating $ln(1 + ax)$ are illustrated in Fig. 13. Overall, the resulted output is considerably accurate, except in the point where the input value is the highest. It contributes significantly to the MAE of the calculation (3.9%). Presumably, the variation of error in several points is the effect of the mechanism applied in the current detector.

4.2 Verification of Circuit Robustness

In order to evaluate the robustness, we observe the circuit's response to the temperature variations and the extremes of the manufacturing process. The circuit tenacity is measured based on the detected output standard deviations. When the variation in one particular condition is observed, other conditioning variables are kept at the default values. For example, the effect of transistor length variation to the detected output is inspected by setting the temperature to 25^0C and



Expected Output of	Detected Output at ^{0}C		
	25	50	75
a = 1.08	1.00	1.02	1.01
b = 0.52	0.55	0.54	0.53
c = 0.14	0.21	0.19	0.19

Fig. 14 The detected output of three distinctive operations under the effect of temperature variations

providing the supply voltage of 1.8 V.

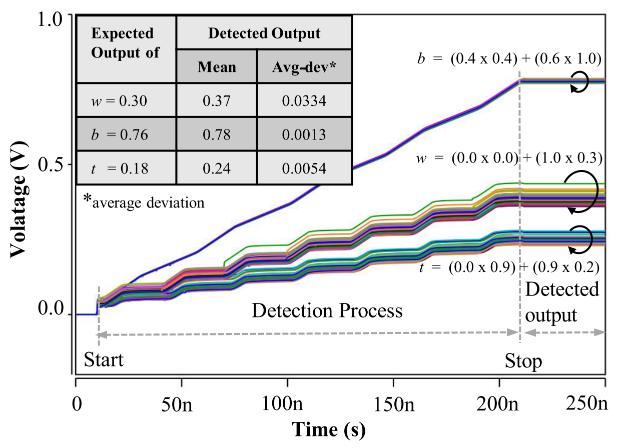
4.2.1 Temperature Variations

It is worth noting that the average error rate shown previously in Fig. 12 reveals the performance of the circuit operating in the default temperature of the simulation environment (25^0C). In order to identify the influence of heat fluctuations on the detected output, the simulation was conditioned to run in 50^0C and 75^0C. Inspecting the detected outputs of 1452 operations under each temperature, we found that the average standard deviation was only 0.017 V, suggesting that the circuit was reasonably robust to temperature variations. Three examples of operations with the expected and detected output of circuit operating in the three different temperatures, provided in Fig. 14.

If the error is calculated using a typical error percentage formula, the error appears reasonably significant. As far as we are concerned, however, such a method is arguably unjustifiable in dealing with relatively small numbers like stochastic computing, where values only range from 0 to 1. In this regard, the error of the detected output in all tested temperatures is calculated using the mean absolute error (MAE), resulting in a 4.8% error. This percentage is slightly lower than the maximum acceptable error (5%) [24] of approximate computing in which stochastic computing is categorized.

4.2.2 Process Variations

The response of the circuit to process variations was verified by varying the transistor length and threshold voltage. The simulation utilized a statistical model, i.e., the Monte Carlo analysis. The sample was 121 operations derived from samples in the prior temperature variation experiment by selecting operation with the most significant standard deviation in every *subset*. Referring to [25], we conducted 121 iterations and used three sigmas Gaussian distribution as a process parameter for each operation. Thus, the confidence level of the



Expected Output of	Detected Output	
	Mean	Avg-dev*
$w = 0.30$	0.37	0.0334
$b = 0.76$	0.78	0.0013
$t = 0.18$	0.24	0.0054

*average deviation

Fig. 15 The best-case, typical-case and the worst-case of average standard deviations in the detected output of MACs due to transistor length variations, simulated by Monte Carlo analysis with 121 iterations

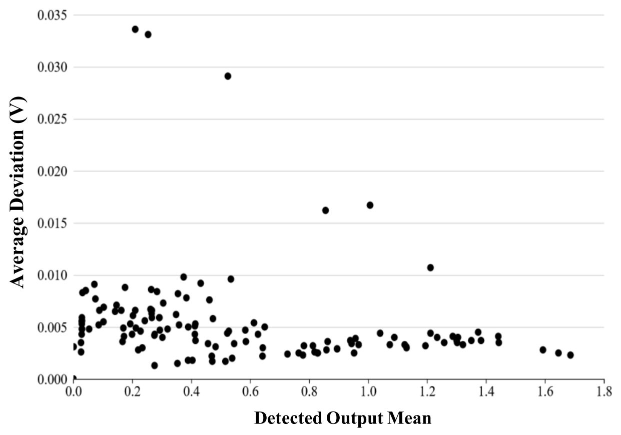
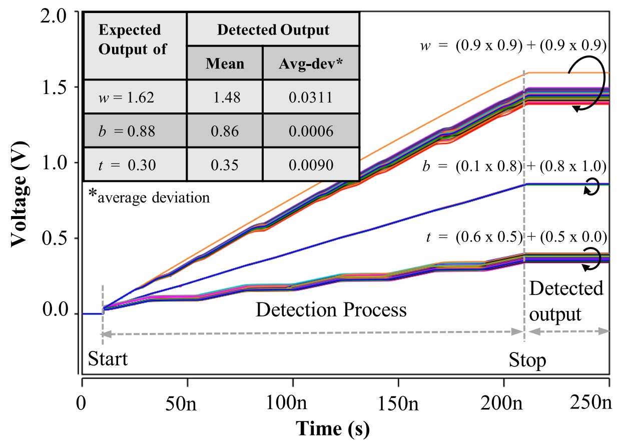


Fig. 16 Distribution of average standard deviations of 121 distinctive detected output as transistor length varies, each operation simulated by 121 iteration Monte Carlo analysis



Expected Output of	Detected Output	
	Mean	Avg-dev*
$w = 1.62$	1.48	0.0311
$b = 0.88$	0.86	0.0006
$t = 0.30$	0.35	0.0090

*average deviation

Fig. 17 The best-case, typical-case and the worst-case of average standard deviations in the detected output of MACs due to threshold voltage variations, simulated by Monte Carlo analysis with 121 iterations

simulation is 99.7%, and estimated yields are 90%, while the lower bound, and upper bound of actual yields are 83% and 97%, respectively.

Since the simulation of each operation produced 121 discrete detected outputs, the disparity is presented in the form of average standard deviations. Further, to avoid confusion, we present the least, average, and the most significant average standard deviations among all operations as the best, typical, and worst-case variations, accordingly.

Transistor Length Variations. In assessing the circuit's response to variability of the transistor length, the resulted experiment depicted in Fig. 15 was retrieved by conditioning each transistor's length in the circuit to vary by 2%. Despite this variation, other conditioning variables such as transistor's corner conditions, temperature, and supply voltage remain at the default values, namely 25^0C for the temperature and 1.8 V for the supply voltage. In this way, any discrepancy in the detected output is more likely to be affected by transistor length variations. Typically, such variations in transistor length cause a significant disparity in the detected output, even when executed with a small number of iterations of MC analysis. Since the experiment involving numerous samples with a distinctive magnitude of values, those samples get affected by the variations to a different degree. Thus, we use the term of the best case, typical case and the worst case to reflect the least affected (smallest variations), the average affected (typical average variations) and the most affected (the most significant variations) samples by the corresponding conditions.

Figure 15 presents the best-case (0.0013 V), typical-case (0.0054 V), and the worst-case (0.0334 V) of average standard deviations in the detected output of MACs due to the transistor length variations. It shows that, even when observed in a large number of iterations, the circuit is sufficiently robust to the fluctuation of transistor length, demonstrated by a relatively narrow disparity even in the worst-case. More importantly, as shown in Fig. 16, about 95% of the operation samples have an average deviation of less than 0.01 V, which is much lower than the worst-case, 0.034 V, and involving only about 5% of the sample operations.

Threshold Voltage Variations. By assuming that absolute variation in the threshold voltage of the transistors in the circuit was 2%, we also simulated 121 iterations Monte Carlo analysis on 121 operations. As shown in Fig. 17, the pattern of the best-case (0.0006 V), typical-case (0.009 V), and the worst-case (0.0311 V) of average standard deviations are almost similar to those in transistor length variation, where the detected output for many iterations still produces relatively close variations. The distribution of the average deviations of the detected output, though, is slightly sparser (only about 90% of average deviation is below 0.015 V) than that of the transistor length variations, as shown in Fig. 18.

Comparing the best-case, typical-case, and the worst-case of detected values which are conditioned to run in a wide range of transistor's length and threshold voltage, the detected output in the proposed circuit seems to be dispersed. However, by keeping in mind that an analog circuit

Table 1 Comparison of hardware implementation

Categories	Bit-stream SC [7]	Conventional TBSC [9]	Our TBSC [12]	MDSC (this work)
IC Technology	40 nm	45 nm	180 nm	180 nm
Representation	bit-stream	duty cycle	duty cycle	pulse strength & duty cycle
Input	digital signal	analog current	analog current	analog current & analog voltage
Multiplier	AND gate	AND gate	AND gate	switch
Adder	MUX (scaled)	MUX (scaled)	MUX (scaled)	KCL (not scaled)
Output	[0, 1]	[0, 1]	[0, 1]	[-∞, +∞]
SNG Power (μW)	13350	323.5	299	179.6 *
Average Accuracy (%) of Mul. & MAC**	N/A	98.6 & 98.6	96.6 & 96.6	96.7 & 95.3
#SNG for Mul. & MAC**	2 & 4	2 & 5	2 & 5	1 & 2
#transistor for Mul. & MAC**	1559 & 3118	967 & 2479	140 & 350	76 & 152
Latency (ns) for Mul. & MAC**	N/A	7 & 245***	7 & 245***	45 & 90

* The SNG+ works on a lower frequency (10MHz-40Mhz)
** Mul. (2 variable multiplication) & MAC (4-variable multiply and accumulate calculations)
*** In order to achieve highly accurate output, all signals represent operands in the TBSC must be independent and uncorrelated, e.g., 2.24, 3.13, 5, 7ns period, and the resulted output is observed for the LCM period of all of the signals.

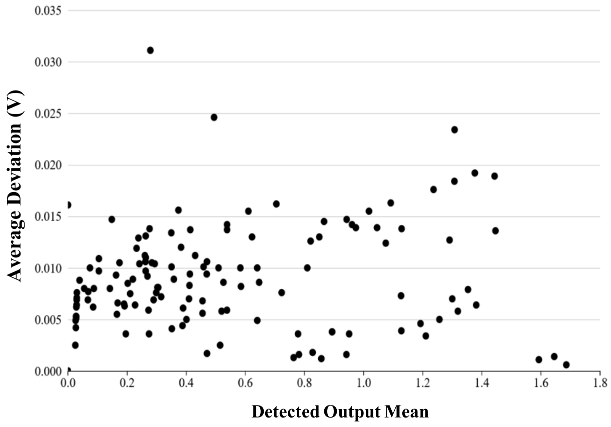


Fig. 18 Distribution of average standard deviations of 121 distinctive detected output as transistor threshold voltage varies, each operation simulated by 121 iteration Monte Carlo analysis

is commonly acknowledged as highly sensitive to the process variation, the disparity in the proposed circuit is acceptable. Besides, the circuit was tested in numerous iteration of the Monte Carlo simulation. Among 121 iterations for every operation, few iterations have significant variations, affecting the detected voltage's overall mean. For these reasons, while we agree that the accuracy and the variation of the proposed method should be improved, we believe that the disparity of error in the process variation of the proposed circuit is acceptable.

4.3 Comparison

Table 1 presents the characteristics of the hardware implementation of various strategies. The diverse input and unique representation of the proposed circuit allow the multiplication and addition to be completed simply by the mean of switches and the KCL, accordingly. Hence, the resulted output is not scaled or approximated, and potentially unlimited [-∞, +∞], achieved by accumulating and changing the direction of the current flow. Furthermore, to accomplish the operations with the same number of operands, the proposal exploits fewer SNGs as half of the operands are in the analog domain. Besides, its summation mechanism does not need an additional SNG to produce the select signal as that in the MUX scaled adder of the TBSC. Eventually, these lead to a fewer number of transistors is required in the MDSC. Likewise, the total latency in the proposed circuit takes the LCM period of signals of operands in the stochastic domain only, instead of for the entire operand's signal LCM period (which also must be independent and uncorrelated) as that in the TBSC. As the operand numbers grow, the proposal quickly outperforms the latency of earlier methods, even though the MDSC operates at the lowest frequency among all. The growth in latency resulting from the increase in LCM, though, is inevitable and can not be totally eliminated because it is an intrinsic consequence of the proposed mechanism. For the same reason, prior determination of the maximum period in operation would also be highly unlikely to do.

5. Conclusion

In this paper, we proposed a multi-domain stochastic computing circuit. The multiply operation is implemented by analog current pulses, where the strength and duty cycle represent two variables. The summation is performed by simply accumulating the current pulses. Employing the original Neuron-MOS based TBSC module and current mirrors, the entire MDSC circuit is designed and simulated in a 0.18 μm CMOS technology. A toy-example is demonstrated for MAC operations. From the circuit simulation results, average accuracy of 95.3% is achieved. The performances over data range, circuit complexity, power consumption, and latency are all improved in contrast to the state-of-art TBSC implementations. Besides, the robustness against temperature and process variations is investigated and analyzed through Monte Carlo simulations.

Acknowledgment

This work supported by JST, PRESTO Grant Number JP-MJPR18M7, Japan. The authors would like to thank the VLSI Design and Education Center (VDEC) of the University of Tokyo in collaboration with Rohm Corporation, Toppan Printing Corporation, Synopsys, Inc., Cadence Design Systems, Inc., and Mentor Graphics, Inc.

References

[1] Z. Li, A. Ren, J. Li, Q. Qiu, B. Yuan, J. Draper, and Y. Wang, "Structural design optimization for deep convolutional neural networks using stochastic computing," Proceedings of the Conference on Design, Automation & Test in Europe, pp.250–253, European Design and Automation Association, 2017.

[2] M. Yamaguchi, G. Iwamoto, H. Tamukoh, and T. Morie, "An energy-efficient time-domain analog vlsi neural network processor based on a pulse-width modulation approach," arXiv preprint arXiv:1902.07707, 2019.

[3] C. Popa, "Improved accuracy current-mode multiplier circuits with applications in analog signal processing," IEEE transactions on very large scale integration (VLSI) systems, vol.22, no.2, pp.443–447, 2013.

[4] C.C. Enz and E.A. Vittoz, "Cmos low-power analog circuit design," Emerging Technologies: Designing Low Power Digital Systems, pp.79–133, IEEE, 1996.

[5] B.R. Gaines, "Stochastic computing," Proceedings of the April 18-20, 1967, spring joint computer conference, pp.149–156, ACM, 1967.

[6] A. Alaghi and J.P. Hayes, "Survey of stochastic computing," ACM Transactions on Embedded computing systems (TECS), vol.12, no.2s, pp.1–19, 2013.

[7] B. Moons and M. Verhelst, "Energy-efficiency and accuracy of stochastic computing circuits in emerging technologies," IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol.4, no.4, pp.475–486, 2014.

[8] W. Qian, X. Li, M.D. Riedel, K. Bazargan, and D.J. Lilja, "An architecture for fault-tolerant computation with stochastic logic," IEEE transactions on computers, vol.60, no.1, pp.93–105, 2010.

[9] M.H. Najafi, S. Jamali-Zavareh, D.J. Lilja, M.D. Riedel, K. Bazargan, and R. Harjani, "Time-encoded values for highly efficient stochastic circuits," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol.25, no.5, pp.1644–1657, 2017.

[10] M.H. Najafi, S. Jamali-Zavareh, D.J. Lilja, M.D. Riedel, K. Bazargan, and R. Harjani, "An overview of time-based computing with stochastic constructs," IEEE Micro, vol.37, no.6, pp.62–71, 2017.

[11] T. Erlina, R. Zhang, and Y. Nakashima, "An efficient multiplier employing time-encoded stochastic computing circuit," CPSY2018-41, vol.118, no.339, pp.47–52, 2018.

[12] T. Erlina, Y. Chen, R. Zhang, and Y. Nakashima, "An efficient time-based stochastic computing circuitry employing neuron-mos," Proceedings of the 2019 on Great Lakes Symposium on VLSI, pp.51–56, ACM, 2019.

[13] T. Shibata and T. Ohmi, "A functional mos transistor featuring gate-level weighted sum and threshold operations," IEEE Transactions on Electron devices, vol.39, no.6, pp.1444–1455, 1992.

[14] T.-H. Chen and J.P. Hayes, "Analyzing and controlling accuracy in stochastic circuits," Computer Design (ICCD), 2014 32nd IEEE International Conference on, pp.367–373, IEEE, 2014.

[15] H. Ichihara, S. Ishii, D. Sunamori, T. Iwagaki, and T. Inoue, "Compact and accurate stochastic circuits with shared random number sources," 2014 IEEE 32nd International Conference on Computer Design (ICCD), pp.361–366, IEEE, 2014.

[16] J.P. Hayes, "Introduction to stochastic computing and its challenges," Proceedings of the 52nd Annual Design Automation Conference, pp.1–3, ACM, 2015.

[17] J.A. Dickson, R.D. McLeod, and H.C. Card, "Stochastic arithmetic implementations of neural networks with in situ learning," IEEE International Conference on Neural Networks, pp.711–716, IEEE, 1993.

[18] P. Li, D.J. Lilja, W. Qian, M.D. Riedel, and K. Bazargan, "Logical computation on stochastic bit streams with linear finite-state machines," IEEE Transactions on Computers, vol.63, no.6, pp.1474–1486, 2012.

[19] J.M. de Aguiar and S.P. Khatri, "Exploring the viability of stochastic computing," 2015 33rd IEEE International Conference on Computer Design (ICCD), pp.391–394, IEEE, 2015.

[20] T.-H. Chen, A. Alaghi, and J.P. Hayes, "Behavior of stochastic circuits under severe error conditions," it-Information Technology, vol.56, no.4, pp.182–191, 2014.

[21] K. Kim, J. Kim, J. Yu, J. Seo, J. Lee, and K. Choi, "Dynamic energy-accuracy trade-off using stochastic computing in deep neural networks," 2016 53nd ACM/EDAC/IEEE Design Automation Conference (DAC), pp.1–6, IEEE, 2016.

[22] K.K. Parhi and Y. Liu, "Computing arithmetic functions using stochastic logic by series expansion," IEEE Transactions on Emerging Topics in Computing, vol.7, no.01, pp.44–59, Jan. 2019.

[23] R. Iachan, "Systematic sampling: A critical review," International Statistical Review/Revue Internationale de Statistique, vol.50, no.3, pp.293–303, 1982.

[24] S. Mittal, "A survey of techniques for approximate computing," ACM Computing Surveys (CSUR), vol.48, no.4, pp.1–33, 2016.

[25] D.E. Hocevar, M.R. Lightner, and T.N. Trick, "A study of variance reduction techniques for estimating circuit yields," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol.2, no.3, pp.180–192, 1983.

Tati Erlina received her S. Kom. and MInfTech (Comp.) degrees from Universitas Putra Indonesia "YPTK" and The Flinders University of South Australia in 2001 and 2008, respectively. She is currently a Doctorate student at the Nara Institute of Science and Technology. Her research interests include stochastic computing and analog VLSI design.

Renyuan Zhang received his M.E. and Ph.D. degrees from Waseda University and the University of Tokyo in 2008 and 2013, respectively. He has been an Assistant Professor with the Nara Institute of Science and Technology since 2017. His research interests include analog-digital-mixed circuits and approximate computing. He is a member of IEEE and IEICE.

Yasuhiko Nakashima received B.E., M.E., and Ph.D. degrees in Computer Engineering from Kyoto University in 1986, 1988 and 1998, respectively. He was a computer architect in the Computer and System Architecture Department, FUJITSU Limited from 1988 to 1999. From 1999 to 2005, he was an associate professor at the Graduate School of Economics, Kyoto University. Since 2006, he has been a professor in the Graduate School of Information Science, Nara Institute of Science and Technology. His research interests include computer architecture, emulation, circuit design, and accelerators. He is a member of IEEE CS, ACM, and IPSJ.