Stochastic Number Generation with Internal Signals of Logic Circuits

Naoya Kubota Hideyuki Ichihara Tsuyoshi Iwagaki Tomoo Inoue

Graduate School of Information Sciences, Hiroshima City University
3-4-1 Ozukahigashi, Asaminami, Hiroshima, JAPAN
kubota@cd.info.hiroshima-cu.ac.jp, {ichihara, iwagaki, tomoo}@hiroshima-cu.ac.jp

Abstract— Stochastic computing (SC) is an approximate computation with random numbers. In this study, we propose a new SC scheme in which internal signals of logic circuits are exploited for generating random numbers. This scheme can eliminate random number generators, such as linear-feedback shift registers, and accordingly reduce the hardware cost for SC without losing the accuracy.

I. Introduction

Stochastic Computing (SC) [1] has attracted attention due to its high error tolerance and low hardware cost. In SC, bit sequences, which are called stochastic numbers (SNs), are used for calculation. An SN is generated from a given deterministic (or binary) number by a converter, called a stochastic number generator (SNG). In general, numerous SNGs are required for SC, so that they account for a large fraction of the circuit; it is important to reduce the hardware size of SNGs.

In this study, we propose a new SNG design scheme without random number generators (RNGs). Since an RNG occupies about half of an SNG, our scheme can reduce the area size of an SNG by half. This scheme exploits, for generating random numbers to an SC circuit, part of internal signals of logic circuits, which may be embedded other than the SC circuit. We also propose a heuristic algorithm for selecting internal signals so as to keep the accuracy of SC high.

II. Stochastic Computing

A stochastic number is a bit sequence where the probability of one occurring corresponds to a number to be computed. For example, a stochastic number 11001111 means 0.75. In SC, computation with SNs can be achieved by simple logic circuits. Fig. 1 shows an SC circuit for multiplication of two numbers 0.50 and 0.25 with an AND gate.

A given deterministic number (DN) is transformed into an SN by an SNG. As shown in Fig. 1, it is composed of an RNG and a comparator. A given DN is compared with a random number generated by the RNG; the comparison result corresponds to an SN. In Fig. 1, DNs $100_{(2)}$ and $010_{(2)}$ are converted into SNs 00110011 and 00011000. Note that the resultant SNs can be converted into DNs with a counter (CNT).

The RNGs in SNGs tend to occupy a considerably large portion of an SC circuit. For example, an SC circuit for KDE-based image segmentaion [2] requires 96 RNGs, which occupy the half of the whole area of the SC circuit. Hence, it is important to reduce the area size of RNGs.

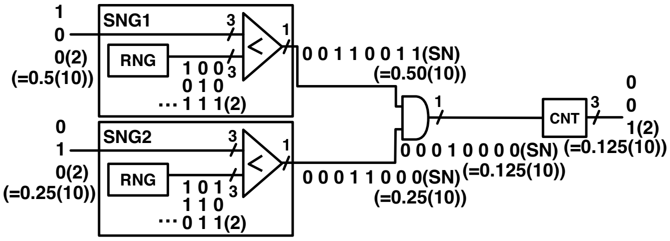


Fig. 1. SC circuit.

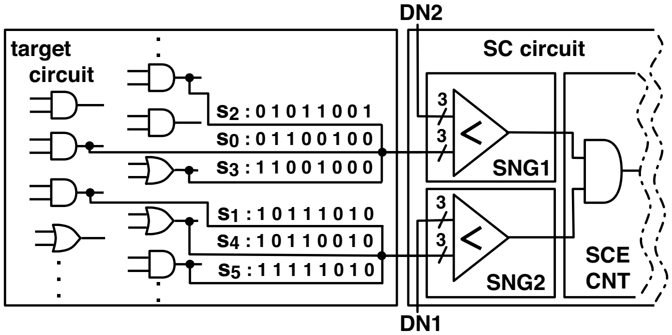


Fig. 2. SNGs with internal signals.

III. Stochastic Number Generation with Internal Signals of Logic Circuits

In general, an SC circuit is used with other logic circuits. For example, when an SC-based edge detection circuit is used for image processing, some converters (e.g., RGB-YUV conversion) and image filters (e.g., gamma correction) are also employed with the SC circuit. Here, we assume that such a logic circuit simultaneously operates with an SC circuit. Under this assumption, we propose a scheme in which internal signals of the logic circuit are exploited for random number generation. This scheme can eliminate RNGs, so that it can drastically reduce the hardware cost. We call a logic circuit from which internal signals are selected a *target circuit.* Fig. 2 shows an example of our scheme. We select six signal lines from a target circuit, instead of employing RNGs.

The conversion error, which is the difference between a given DN and the corresponding SN, depends on which signal lines are selected from a target circuit. In Table I, the second and third columns show two signal line selections from the target circuit in Fig. 2: $\boldsymbol{b_1} = (s_2, s_0, s_3)$ and $\boldsymbol{b_2} = (s_1, s_4, s_5)$. Here, for convenience, s_i represents a signal line or the bit sequence observed at the signal line. For instance, s_2 means not only a signal line in the

TABLE I
Two selections of internal signals and uniform random sequence.

t	$\boldsymbol{b_1} = (s_2, s_0, s_3)$	$\boldsymbol{b_2} = (s_1, s_4, s_5)$	(u_0, u_1, u_2)
0	1 1 1	1 0 1	0 0 0
1	0 1 0	1 1 0	0 0 1
2	0 0 0	1 0 0	0 1 0
3	1 0 1	0 1 1	0 1 1
4	1 0 1	0 0 1	1 0 0
5	1 1 0	1 0 1	1 0 1
6	0 0 0	0 0 1	1 1 0
7	0 1 1	1 1 1	1 1 1
avg. error	0.08	0.17	0
$\sum F/n$	0.50	0.58	0.50
$\sum \chi^2$	2.00	7.00	0

target circuit but also a sequence 10011100. As shown in these columns, two selections $\boldsymbol{b_1}$ and $\boldsymbol{b_2}$ generate different random sequences, so that the average conversion error of selection $\boldsymbol{b_1}$ is smaller than that of selection $\boldsymbol{b_2}$, as shown in row *avg. error*.

IV. Algorithm for Selecting Internal Signals

We propose an algorithm for selecting internal signals from a given target circuit in order to minimize the average conversion error. This algorithm receives a set of bit sequences $\{s_i\}$ that result from a logic simulation.

Our strategy is to make the random sequence composed of the selected signals close to a uniform sequence. The last column of Table I shows a 3-digit uniform random sequence (u_0, u_1, u_2). If this uniform sequence is used for SNGs, the average conversion error becomes zero. To make the selected bit sequence close to a uniform one, we consider two properties that should be satisfied by the selected n sequences $\boldsymbol{b} = (s'_0, s'_1, \ldots, s'_{n-1})$:

(1) the probability of one occurrence in $\boldsymbol{b}$ is close to 0.5. With the probability $F(s'_i)$ of one occurrence of a sequence s'_i, it can be denoted by $\sum_{s'_i \in \boldsymbol{b}} F(s'_i)/n \simeq 0.5$, and

(2) the chi-square-value $\chi^2(s'_i, s'_j)$ of any pair s'_i and s'_j of $\boldsymbol{b}$ is close to 0. Here, $\chi^2(s'_i, s'_j) = \sum_{x \in \{00,01,10,11\}}((f_x(s'_i, s'_j) - L/4)^2)/(L/4)$, where L is the length of bit sequences (eight in our example) and $f_{pq}(s'_i, s'_j)$ means the frequency of appearance of a logic value pair (p, q) in a bit sequence pair (s'_i, s'_j) ($f_{11}(s_2, s_3) = 3$) [3].

The overview of our algorithm is shown in Fig. 3. The algorithm selects signal lines based on F and χ^2 in a greedy manner. For the six signals of the target circuit in Fig. 2, this algorithm selects $\boldsymbol{b_1}$ of Table I, whose $\sum_{s'_i \in \boldsymbol{b}} F(s'_i)/n$ and $\sum_{\forall (s'_i, s'_j) \in \boldsymbol{b}^2} \chi^2(s'_i, s'_j)$ are closer to 0.5 and 0 than $\boldsymbol{b_2}$, respectively, as shown in the bottom of the table.

V. Experimental Results and Conclusions

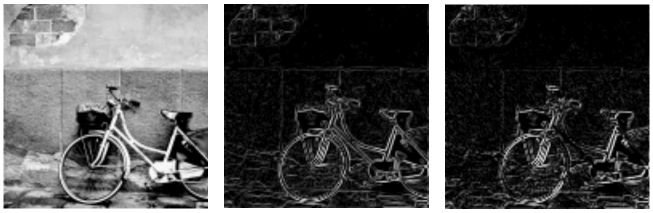
We applied our scheme to an image processing system. This system includes an SC-based edge detection circuit and a deterministic gamma correction and scaling circuit, which was regarded as a target circuit. For comparison, we implemented an SC-based edge detection circuit with RNGs (linear-feedback shift-registers (LFSRs)).

```
Input: simulation results $\{s_i\}$ of each signal line, the number $n$ of signal lines to be seleceted.
Output: selected signal lines $\boldsymbol{b} = (s'_0, s'_1, \ldots, s'_{n-1})$.
(1) Select a signal line $s_i$ whose $F(s_i)$ is the closest to 0.5 of all. $s'_0 = s_i$.
(2) For $k = 1$ to $k = n - 1$.
    Select a signal line $s_i$ such that $\sum_{j=0}^{k-1} \chi^2(s'_j, s_i)$ is the smallest of all. If there are two more signal lines with the smallest summation, select a signal line $s_i$ such that $(\sum_{j=0}^{k-1} F(s'_j) + F(s_i))/k$ is the closest to 0.5 of them. $s'_k = s_i$.
(3) Return $\boldsymbol{b}$.
```

Fig. 3. Internal signal selection algorithm.

TABLE II
processing result

	area [μm^2]	PSNR [dB]
with LFSRs	337.022	57.575
proposed scheme	175.826	54.307



(a) original (b) with LFSRs (c) proposed scheme

Fig. 4. Edge detection result (128 × 128).

Table II shows the area size of the SC circuit and peak signal-to-noise ratio (PSNR). As shown in this table, the area size can be reduced by 48% with keeping high PSNR. Fig. 4 shows edge detection results for an image. From these results, we can see that our scheme can keep high-quality edge detection.

Acknowledgements

This work is supported by JSPS KAKENHI Grant Number JP25330072 and VLSI Design and Education Center(VDEC), The University of Tokyo with the collaboration with Synopsys Corporation.

References

[1] A. Alaghi and J. P. Hayes, "Survey of stochastic computing," *Trans. on Embedded Computing Systems*, vol. 12, no. 2, pp. 1–19, May 2013.

[2] H. Ichihara, S. Ishii, D. Sunamori, T. Iwagaki and T. Inoue, "Compact and accurate stochastic circuits with shared random number sources," *Proc. ICCD*, pp. 361–366, 2014.

[3] R. A. Fisher, "On the interpretation of χ^2 from contingency tables, and the calculation of P," *Journal. Roy. Stat. Soc*, vol. 85, no. 1, pp. 87–94, 1922.