Effect of Bit-Level Correlation in Stochastic Computing

Megha Parhi, Marc D. Riedel, Keshab K. Parhi
Dept. of Electrical and Computer Engineering
University of Minnesota, Minneapolis, MN 55455
{parhi002, mriedel, parhi}@umn.edu

***Abstract*—Simple stochastic logic gates can compute complex functions using stochastic computing. A stochastic number is encoded by a unary bit stream where each bit is 0 or 1. The value of the number is represented by the percent of 1's in the number, and is interpreted as a probability. Each bit of the stochastic number can be modeled as a Bernoulli random variable, and each stochastic number can be represented by a binomial random variable. The variance of a stochastic number is given by $p(1-p)/N$ where N represents the number of bits in the sequence, and p represents the mean value of the number. For long word-lengths, a binomial random variable behaves as a Gaussian random variable. The mean and variance of a two-input stochastic logic gate are dependent on the bit-level correlation of the two inputs. This paper derives closed-form expressions for mean and variance of two-input stochastic logic gates with correlated inputs. An approach to synthesize correlated stochastic bit streams with specified correlation from uncorrelated bit streams is also presented. Using the proposed synthesis method, stochastic logic gates are simulated with correlated inputs. The simulated values of means and variances are shown to be the same as the theoretical values; thus, the closed-form expressions are validated.**

I. Introduction

Stochastic Computing (SC) was proposed in 1967 by Gaines [1] [2]. SC was proposed as an alternative approach to binary computing. SC uses a unary system as opposed to a weighted system. A number is represented by a string of 1's and 0's where the percent of 1's in the number represents the value of the number as a probability. Computing using stochastic logic requires significantly less hardware compared to traditional binary arithmetic. However, these circuits suffer from a significant increase in latency. This is because the number of bits used to represent a number is typically very large. For example, a 10-bit binary number can represent 1024 levels. To achieve the same resolution, a stochastic number should be represented using 1024 bits. The stochastic computing systems are ideal for low-speed applications such as neural networks [3], for decoding error-control codes such as low-density parity check codes and polar codes [4]–[6], for digital filters [7]–[9], cyber-physical systems operating at very low rates, and biomedical applications. Stochastic logic gates compute an *approximate* value of the result, as opposed to the exact value. In many applications, such as machine learning, where a decision is made based on a thresholding operation, the decisions may not be affected by approximate computing. A main advantage of stochastic computing is its inherent tolerance to faults in CMOS circuits.

The stochastic bit stream is represented using a string of 1's and 0's where each bit is assumed to be independent of the other bits, i.e., the bits are assumed to be random. This randomness corresponds to temporal independence, i.e., the bits at different positions (or times) are uncorrelated. Closed-form expressions for stochastic logic outputs have been derived for uncorrelated inputs in [10]. Synthesis of logic gates to compute specified probabilities from uncorrelated bit streams has been proposed in [11] [12]. However, the bit-level correlation between the two inputs to a stochastic logic gate affects the expected value of the output. Recently, the notion of *stochastic correlation* has been introduced in [13] and has been used to generate correlated bit streams using probabilistic transfer matrices. In contrast, the proposed work presents a method to generate correlated bit streams using Pearson correlation. Although the Pearson correlation coefficient, ρ, lies between -1 and 1, the ρ of two stochastic bit streams is restricted to a narrower range; this range is dependent on the expected values of the stochastic inputs.

This paper makes two key contributions. First, closed-form expressions are derived for mean and variance of outputs of two-input stochastic logic gates as functions of the input probabilities and the spatial correlation between the inputs. To prove the theoretical expressions by simulations, it is necessary to generate correlated bit streams for a specified correlation. However, generation of correlated stochastic bit streams has not been addressed in prior work. The second contribution of this paper is a novel approach to generate correlated bit streams. Finally the validity of the theoretical expressions is demonstrated through simulations of stochastic logic gates using correlated bit streams.

This paper is organized as follows. In Section II closed-form expressions for outputs of stochastic logic gates with correlated inputs are derived. Section III presents an approach to synthesize correlated stochastic bit streams from uncorrelated bit streams. Section IV compares stochastic outputs computed theoretically and from simulations.

II. Analysis of Stochastic Logic with Correlated Inputs

Each input bit of a stochastic bit stream is assumed to be an independent identically distributed (IID) Bernoulli random variable with the probability of one equal to p. Thus, the stochastic number is represented by a Binomial random variable with variance, $\sigma^2 = p(1-p)/N$ where N represents the number of bits in the word. Thus, the variance is maximum at $p = 0.5$ and is minimum at $p = 0$ or $p = 1$.

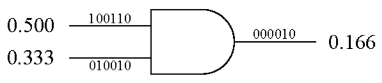

Fig. 1: An example of multiplication using an AND gate

Correlation in bit streams alters the expected output of a stochastic logic circuit from that with independent bit streams. This will be discussed further in this section. Consider the multiplication of two stochastic numbers, X_1 and X_2 using an AND gate as shown in Figure 1. The probabilities of the two bit streams, denoted as p_1 and p_2, are assumed to be 0.5 and 0.3333, with word-length of 6 bits. Let Y denote the output. If X_1 and X_2 are uncorrelated, the output probability is given by $E(Y) = p_1 \times p_2$. In this example the output value is 0.1666. If the bits of the two inputs are correlated, the expected value of the output can be derived from the definition of the *bit-level* correlation coefficient:

$$\rho = \frac{E(Y) - p_1 p_2}{\sigma_1 \sigma_2} \tag{1}$$

where p_1 and p_2 represent the mean values of the two stochastic inputs X_1 and X_2, and σ_1 and σ_2, respectively, represent the *bit-level* standard deviations of X_1 and X_2. The expected value of the output, Y, is given by:

$$\begin{aligned} E(Y) &= E(X_1 X_2) \\ &= p_1 p_2 + \rho \sigma_1 \sigma_2 \\ &= p_1 p_2 + \rho \sqrt{p_1 p_2 (1 - p_1)(1 - p_2)} \end{aligned} \tag{2}$$

If the two bit-streams, X_1 and X_2 are identical, then $p_1 = p_2$ and ρ is one, and $E(Y) = p_1 = p_2$.

Consider a two-input OR gate with uncorrelated inputs. The mean value of its output is given by $p_1 + p_2 - p_1 p_2$ where p_1 and p_2, respectively, represent the expected values of X_1 and X_2. If X_1 and X_2 are correlated with correlation coefficient ρ, using (1), the mean output of the OR gate can be calculated as:

$$\begin{aligned} E(Y) &= E(X_1 + X_2 - X_1 X_2) \\ &= p_1 + p_2 - p_1 p_2 - \rho \sigma_1 \sigma_2. \end{aligned} \tag{3}$$

The expected output of an XOR gate can be similarly computed as:

$$E(Y) = p_1 + p_2 - 2 p_1 p_2 - 2 \rho \sigma_1 \sigma_2.$$

The output of a MUX is unaffected by correlation of input signals. Let the input signals be denoted as X_1 and X_2, and the select signal be denoted as X_3. Let the stochastic values of these three signals be, respectively, given by p_1, p_2, and p_3. The mean output of the MUX, Y is given by:

$$E(Y) = p_1 - p_1 p_3 + p_2 p_3. \tag{4}$$

Although the mean output is unaffected by correlation between X_1 and X_2, it can be affected by the correlation between the control signal, X_3, and either or both input signals, X_1 or X_2.

The mean outputs of other stochastic logic gates can be derived in a similar manner. Table I lists the mean outputs of various stochastic logic gates without and with input correlation.

III. Synthesizing Correlated Bit Streams

To verify the results of Table I, it is necessary to synthesize two correlated bit streams for specified mean values and correlation coefficient. An approach to synthesize correlated stochastic bit streams from uncorrelated bit streams is presented in this section.

Let p_1, p_2 and ρ, respectively, represent the mean values of stochastic numbers X_1, X_2, and their *bit-level* correlation coefficient. A joint probability mass function (pmf) of X_1 and X_2 is described in Table II, where the only unknown is a. The parameter a describes the probability that $X_1 = 1$ and $X_2 = 1$, and can be derived from the correlation coefficient, ρ.

TABLE II: The joint Probability Mass Function (PMF) for X_1 and X_2

Input Signal	Probability
$x_1 = 0, x_2 = 0$	$1 - p_1 - p_2 + a$
$x_1 = 0, x_2 = 1$	$p_2 - a$
$x_1 = 1, x_2 = 0$	$p_1 - a$
$x_1 = 1, x_2 = 1$	a

Since all values in Table II represent probabilities, these must be non-negative. This is guaranteed by the constraints:

- $a > 0$
- $a < p_2$
- $a < p_1$
- $a > p_1 + p_2 - 1$

From the definition of the *bit-level* correlation coefficient, ρ, given by:

$$\rho = \frac{a - p_1 p_2}{\sigma_1 \sigma_2} \tag{5}$$

the parameter a can be obtained as:

$$\begin{aligned} a &= p_1 p_2 + \rho \sigma_1 \sigma_2 \\ &= p_1 p_2 + \rho \sqrt{p_1 p_2 (1 - p_1)(1 - p_2)}. \end{aligned} \tag{6}$$

Note that although the correlation coefficient, ρ, lies between -1 and 1, the above constraints limit the allowable range of ρ. For example, if $p_1 = .3$, and $p_2 = 0.8$, the range of a is constrained to the range $[0.1, 0.3]$. This limits the range of ρ to $[-0.7638, 0.3273]$.

The two correlated stochastic bit streams, X_1 and X_2, can be synthesized from two uncorrelated bit streams, denoted as R_1 and R_2, where each bit of R_1 and R_2 is a uniform random variable between 0 and 1. The proposed approach makes use

TABLE I: Closed form Expressions for Single Input Gates

Gate type	Independent	Correlated
AND	p_1p_2	$p_1p_2+\rho\sigma_1\sigma_2$
AND (X_1 inverted)	$p_2(1-p_1)$	$p_2-p_1p_2-\rho\sigma_1\sigma_2$
NAND	$1-p_1p_2$	$1-p_1p_2-\rho\sigma_1\sigma_2$
OR	$p_1+p_2-p_1p_2$	$p_1+p_2-p_1p_2-\rho\sigma_1\sigma_2$
OR (X_1 inverted)	$p_2+(1-p_1)-p_2(1-p_1)$	$1-p_1+p_1p_2+\rho\sigma_1\sigma_2$
NOR	$(1-p_1)(1-p_2)$	$1-p_1-p_2+p_1p_2+\rho\sigma_1\sigma_2$
XOR	$p_1+p_2-2p_1p_2$	$p_1+p_2-2p_1p_2-2\rho\sigma_1\sigma_2$
XOR (X_1 inverted)	$1-p_1-p_2+2p_1p_2$	$1-p_1-p_2+2p_1p_2+2\rho\sigma_1\sigma_2$
XNOR	$1-p_1-p_2+2p_1p_2$	$1-p_1-p_2+2p_1p_2+2\rho\sigma_1\sigma_2$
MUX (X_3 select signal)	$p_1-p_1p_3+p_2p_3$	$p_1-p_1p_3+p_2p_3$
MUX (X_3 select signal with X_1 inverted)	$1-p_1-p_3+p_1p_3+p_2p_3$	$1-p_1-p_3+p_1p_3+p_2p_3$

of marginal probability and conditional marginal probability. First R_1 is used to generate X_1 using marginal probability of X_1. Then, X_2 is generated from R_2 using conditional marginal probability of X_2 conditioned to selection of X_1. The approach to generating X_1 follows the principle of stochastic number generator (SNG). If the probability of first uncorrelated bit stream, R_1, is greater then p_1, then $X_1 = 0$, otherwise $X_1 = 1$. X_2 is synthesized by using the conditional probability of X_2 given X_1, which is:

$$P(X_2|X_1) = \frac{P(X_1, X_2)}{P(X_1)}. \tag{7}$$

Assume that $X_1 = 1$ is given. we obtain $P(X_2 = 1|X_1 = 1) = a/p_1$. If $R_2 > a/p_1$, then $X_2 = 0$. Otherwise, $X_2 = 1$. Given that $X_1 = 0$ is generated, the conditional probability will be $P(X_2 = 1|X_1 = 0) = (p_2 - a)/(1 - p_1)$. Then $X_2 = 0$ if $R_2 > (p_2 - a)/(1 - p_1)$. Otherwise $X_2 = 1$. Figure 2 shows the decision tree to synthesize two correlated stochastic bit steams with specified p_1, p_2 and ρ. Every bit of two stochastic sequences is generated using this decision tree.

IV. Simulation Results

Stochastic bit streams with and without correlation are input stochastic logic gates. The difference in their outputs is referred as *Deviation D*. Using Monte Carlo experiments, a histogram of D is constructed and its mean and standard deviation are obtained from simulation. These are then compared with theoretical values predicted from Table I.

TABLE III: Error Analysis

Gate type	Error
AND	$\rho\sigma_1\sigma_2$
NAND	$-\rho\sigma_1\sigma_2$
OR	$-\rho\sigma_1\sigma_2$
NOR	$\rho\sigma_1\sigma_2$
XOR	$-2\rho\sigma_1\sigma_2$
XNOR	$2\rho\sigma_1\sigma_2$

The mean values of the deviation for each gate are shown in Table III. This deviation is found by subtracting expected values with correlation from without using Table I. The standard deviation can be computed as a function of the length of the bit stream. Consider an AND gate as an example. Let the two mean values be given by 0.4 and 0.5 and the correlation coefficient be given by 0.2. The mean value of the output is simulated with and without correlation for a word-length of 100 bits with 2000 Monte Carlo runs. Figure 3 illustrates the histogram of the output value for correlated inputs.

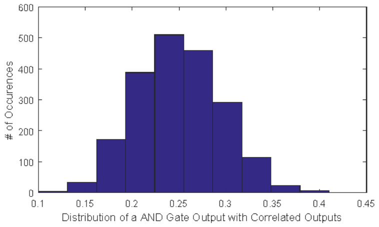

Fig. 3: The histogram for a 100 bits and 2000 experiments for independent bit streams.

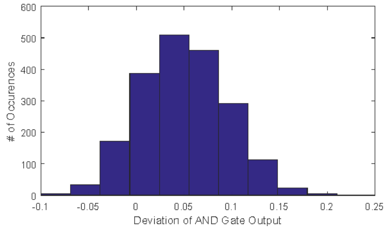

Fig. 4: The histogram shows the error of the correlated bit streams.

The expected output for the AND gate with correlated inputs is given by:

$$\hat{y} = p_1p_2 + \rho\sigma_1\sigma_2 = 0.2 + 0.2 \times \frac{\sqrt{0.4 \times 0.5 \times 0.6 \times 0.5}}{100} = .2489$$

Using (8):

$$E(Deviation) = \hat{y} - y = .2489 - .2 = .0489 \tag{8}$$

This can also be written as:

$$E(Deviation) = p_1p_2 + \rho\sigma_1\sigma_2 - p_1p_2 = \rho\sigma_1\sigma_2 \tag{9}$$

The standard deviation of the Deviation can be calculated using $\sqrt{p(1-p)/N}$ using $p = 0.2489$ and $N = 100$, and is given by 0.0432. The Deviation histogram is shown in Fig. 4. This histogram is in agreement with the predicted value.

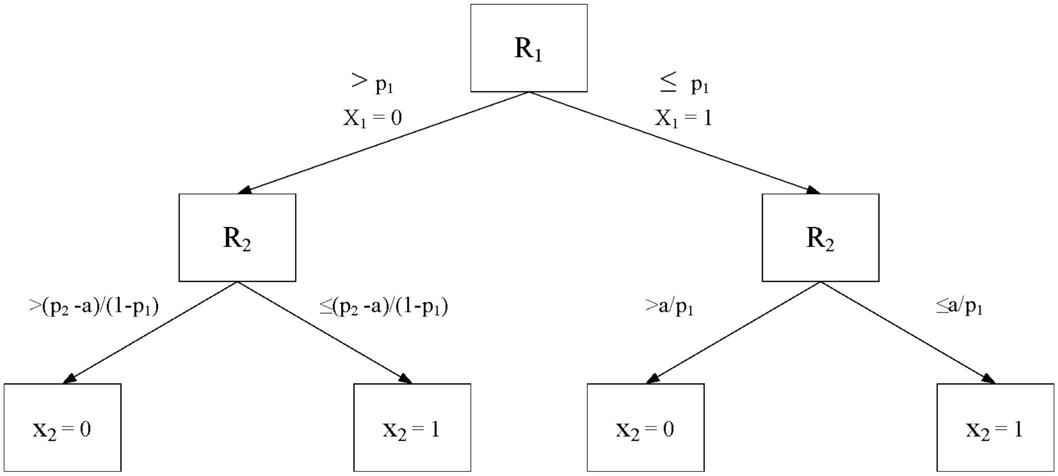

Fig. 2: Method of synthesizing two correlated bit streams.

V. Analysis of Two-Level Stochastic Logic

This section illustrates theoretical computation of mean output of two-level stochastic logic circuits with correlated inputs. Consider the two-level stochastic logic example shown in Figure 5. In this figure, X_2 is a common input to the two AND gates in the first level. We also assume x_2 to be independent of x_1 and x_3. Let x_1 and x_3 be correlated with a correlation coefficient ρ. Let X_1, X_2, and X_3 have probabilities of p_1, p_2, and p_3, respectively. Using Figure 5, we can find the $E(Z)$ by the following derivation:

$$E(Y_1) = E(X_1X_2) = p_1p_2$$
$$E(Y_2) = E(X_2X_3) = p_2p_3$$

Using the equation for the output of the AND gate with correlation:

$$\begin{aligned} E(Z) &= E(Y_1Y_2) \\ &= p_1p_2^2p_3 + \rho_{Y_1Y_2}\sqrt{p_1p_2^2p_3(1-p_1p_2)(1-p_2p_3)} \\ &= p_2\left(p_1p_2p_3 + \rho_{Y_1Y_2}\sqrt{p_1p_3(1-p_1p_2)(1-p_2p_3)}\right) \end{aligned}$$

Note that the bit-level correlation between Y_1 and Y_2 is given by $\rho_{Y_1Y_2}$. It can be shown that:

$$\rho_{Y_1Y_2} = \frac{p_1p_3(1-p_2) + \rho\sqrt{p_1p_3(1-p_1)(1-p_3)}}{\sqrt{p_1p_3(1-p_1p_2)(1-p_2p_3)}}$$

Further note if ρ is 0, then $E(Z) = p_1p_2p_3$. Fig. 6 shows a circuit that is equivalent to Fig. 5.

For Figure 6, $E(Z)$ can be computed as follows:

$$E(Y_1) = p_1p_3 + \rho\sqrt{p_1p_3(1-p_1)(1-p_3)}$$

$$E(Z) = E(X_2Y) = p_2(p_1p_3 + \rho\sqrt{p_1p_3(1-p_1)(1-p_3)})$$

If ρ is 0, then $E(Z) = p_1p_2p_3$.

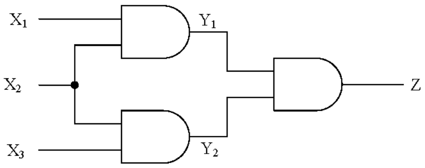

Fig. 5: Two-level Stochastic Logic Gate

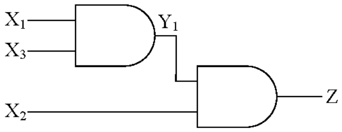

Fig. 6: Two-level Stochastic Logic Gate

VI. Conclusion

This paper has presented closed-form expressions for mean values of two-input stochastic logic gates with correlated inputs. A method to generate two correlated bit streams has been presented. Simulation results using synthesized correlated bit streams validate the theoretical expressions derived for the mean and variance values. Future work will be directed towards derivation of mean values of three-input logic gates, and for multi-level stochastic logic.

VII. Acknowledgment

This was supported in parts by an Undergraduate Research Opportunity Program (UROP) from the University of Minnesota, by the National Science Foundation (NSF) Career Award 0845650, and the NSF grant CCF-1319107.

REFERENCES

[1] B.R. Gaines. Stochastic computing. In *in Proc. AFIPS Spring Joint Computer Conference*, pages 149–156, 1967.

[2] B.R. Gaines. Stochastic computing systems. In *Advances in Information Systems Science*, pages 37–172. Springer US, 1969.

[3] B.D. Brown and H.C. Card. Stochastic neural networks part I:computational elements. *IEEE Transactions on Computers*, 50(9):891–905, Sept 2001.

[4] S.S. Tehrani, A. Naderi, G.-A Kamendje, , S. Hemati, S. Mannor, and W.J. Gross. Majority-based tracking forecast memories for stochastic LDPC decoding. *IEEE Transactions on Signal Processing*, 58(9):4883–4896, 2010.

[5] S.S. Tehrani, S. Mannor, and W.J. Gross. Fully parallel stochastic LDPC decoders. *IEEE Transactions on Signal Processing*, 56(11):5692–5703, 2008.

[6] B. Yuan and K.K. Parhi. Successive cancellation decoding of polar codes using stochastic computing. In *Proc. 2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015.

[7] Y. Liu and K.K. Parhi. Lattice FIR digital filters using stochastic computing. In *Proc. 2015 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2015.

[8] K.K. Parhi and Y. Liu. Architectures for IIR digital filters using stochastic computing. In *Proc. 2014 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 373–376, 2014.

[9] Y.N. Chang and K.K. Parhi. Architectures for digital filters using stochastic computing. In *Proc. 2013 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pages 2697–2701, 2013.

[10] K.P. Parker and E.J. McCluskey. Probabilistic treatment of general combinational networks. *IEEE Transactions on Computers,*, C-24(6):668–670, June 1975.

[11] W. Qian, M.D. Riedel, H. Zhou, and J. Bruck. Transforming probabilities with combinational logic. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 30(9):1279–1292, Sept 2011.

[12] W. Qian, M.D. Riedel, K Bazargan, and D.J. Lilja. The synthesis of combinational logic to generate probabilities. In *Proceedings of the 2009 International Conference on Computer-Aided Design (ICCAD)*, ICCAD '09, pages 367–374, New York, NY, USA, 2009. ACM.

[13] A. Alaghi and J.P. Hayes. Exploiting correlation in stochastic circuit design. In *Proc. 2013 IEEE International Conference on Computer Design (ICCD)*, pages 39–46, Oct 2013.