Framework for Quantifying and Managing Accuracy in Stochastic Circuit Design

Florian Neugebauer[€], Ilia Polian[€] and John P. Hayes[$]

[€]Faculty of Computer Science and Mathematics
University of Passau,
Innstr. 43, D-94032 Passau, Germany
{florian.neugebauer|ilia.polian}@uni-passau.de

[$]Computer Engineering Laboratory
University of Michigan,
Ann Arbor, MI 48109, USA
jhayes@eecs.umich.edu

***Abstract*—Stochastic circuits (SCs) offer tremendous area- and power-consumption benefits at the expense of computational inaccuracies. Managing accuracy is a central problem in SC design and has no counterpart in conventional circuit synthesis. It raises a basic question: how to build a systematic design flow for stochastic circuits? We present, for the first time, a systematic design approach to control the accuracy of SCs and balance it against other design parameters. We express the (in)accuracy of a circuit processing *n*-bit stochastic numbers by the numerical deviation of the computed value from the expected result, in conjunction with a confidence level. Using the theory of Monte Carlo simulation, we derive expressions for the stochastic number length required for a desired level of accuracy, or vice versa. We discuss the integration of the theory into a design framework that is applicable to both combinational and sequential SCs. We show that for combinational SCs, accuracy is independent of the circuit's size or complexity, a surprising result. We also show how the analysis can identify subtle errors in both combinational and sequential designs.**

Keywords: Emerging technologies, error analysis, simulation, stochastic computing.

I. INTRODUCTION

Stochastic computing [2][9] has proven effective for a number of applications including neural networks [4], decoding of low-density parity codes [14], image-processing [3][12], vector quantization [19] and filter design [8]. A recent advance is the development of generic synthesis methods which take a functional description and automatically generate a stochastic circuit that realizes this function [1][15][18]. Stochastic computing provides extremely compact, error-tolerant and low-power implementations of complex functions, but at the expense of long computation times and some degree of inaccuracy.

Most current approaches focus on the design of stochastic circuits (SCs) that are functionally correct under idealized assumptions. It is often assumed that increasing the length n of stochastic numbers (SNs) used for a computation will improve its accuracy at the expense of the required time and energy. In prior work, this trade-off was controlled by running simulations for different values of n and observing the achieved accuracy. However, a systematic, mathematical foundation for quantifying and managing accuracy is lacking.

In this paper, we introduce a framework which formally establishes a relationship between the accuracy of a stochastic computation and the SN length n based on the theory of (direct) Monte Carlo simulation [11][13]. Accuracy is quantified by mean-square error (MSE) Δ^2 and the corresponding error ε, in combination with a confidence level γ, The framework can be used to determine the accuracy of a given stochastic computation, or to calculate the minimum n (denoted n^*) needed to achieve user-specified ε and γ. The methodology can be integrated into an SC synthesis flow and provide a foundation for design state exploration, i.e., systematic comparison of different circuit realizations of the desired function in order to meet accuracy specification with minimal cost.

The accuracy quantification and control methodology is applicable to combinational and sequential SCs. Many practical SCs are affected by undesired correlations [17], and a few decorrelation techniques are known, notably regeneration and isolation [6]. These techniques add memory elements, often in large numbers, to the target circuit and thus make it sequential. Therefore, the ability to handle sequential circuits is essential even if only functions realizable by combinational SCs are considered. The accuracy is harder to model in the sequential case because the individual bits of a stochastic number are no longer independent of each other. We solve this problem by extending the proposed theory to time-frame expansions of sequential circuits. Consequently, our approach is readily applicable to circuits employing decorrelation techniques. It can even provide a systematic assessment of whether a chosen decorrelation strategy provides a satisfactory level of accuracy.

We report the application of our framework to several combinational and sequential stochastic circuits. An unexpected finding is that the accuracy of a combinational SC is completely determined by its output value and the SN length; it is independent of the circuit's size or complexity. For sequential circuits, the length-accuracy trade-off depends on the output variance, which must be determined from the circuit's time-frame expansion. Output variance is represented by a symbolic expression, which can be generated automatically; this task is particularly simple for circuits without feedback. We demonstrate the validity of our framework by comparing its predictions with simulations for different classes of circuits.

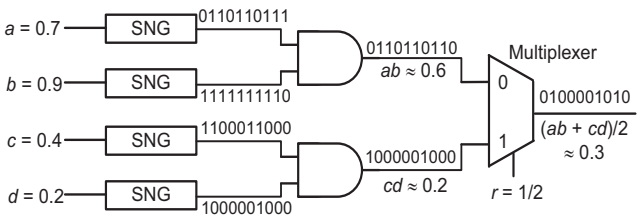

Fig. 1: Example stochastic calculation using unipolar SNs.

978-3-9815370-8-6/17/$31.00 ©2017 IEEE

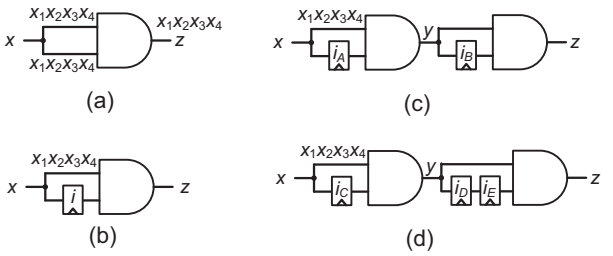

Fig. 2: (a) Incorrect squarer; (b) decorrelated squarer; (c) incorrect quartic circuit; (d) decorrelated quartic circuit.

We also observe that non-obvious design errors can be identified with the help of our framework, thus suggesting its usefulness for stochastic circuit validation.

The paper is organized as follows. Sec. II discusses accuracy issues in SC, namely errors due to approximation, quantization, random fluctuations, and undesired correlations. It also proposes a general accuracy-driven design flow. Sec. III presents the paper's mathematical foundations. Sec. IV describes the proposed framework for quantifying and managing accuracy, and experimental results are reported in Sec. V.

II. Accuracy Issues in Stochastic Computing

The accuracy of a stochastic computation is affected by several mechanisms. First, SCs can only represent certain classes of functions. For example, combinational SCs can only implement certain multilinear polynomials, and if the desired function does not have a suitable polynomial form, it has to be first approximated by one. Qian et al. [15] distinguish between *approximation*, *quantization* and *random-fluctuation* errors. We do not consider approximation errors; we assume that each SC is a suitable, but possibly inaccurate, realization a desired function *f*. We do consider inaccuracies induced by *undesired correlations* [6]. Correlation errors occur when the same random source is used for generating multiple SNs, when a single SN is used multiple times during a computation (as in circuits with re-convergent fanout), or in sequential circuits with feedback. In the remainder of this section, we discuss quantization, random-fluctuation and correlation errors and provide illustrative examples.

A. Accuracy in the absence of correlation

An *n*-bit *unipolar SN* is a stream of *n* bits whose *value* is n_1/n if n_1 out of *n* bits are 1. Bipolar representation extends the SN range to [–1,1]. A *bipolar SN* with n_1 1s and n_0 0s represents the value $(n_1 - n_0)/n$. For example, an SN having four 1's and six 0's stands for 4/(4 + 6) = 0.4 in unipolar and for (4 – 6)/(4 + 6) = –0.2 in bipolar. All concepts in this paper apply to both representations. We use the circuit of Fig. 1 to summarize stochastic computing and various types of errors that can occur. This circuit takes four numbers $a, b, c, d \in [0, 1]$ and calculates the normalized sum of products $(ab + cd)/2$ using unipolar SNs of length $n = 10$. The binary numbers applied to the circuit inputs are converted to unipolar stochastic form using *stochastic-number generators* (SNGs). Multiplication is performed by AND gates and scaled addition is realized by a multiplexer with its select input attached to a random source, i.e., an SN of value 0.5. The outcome can be converted back to binary using a counter, or it can be further processed in the stochastic domain.

Quantization errors occur when processing numbers that cannot be exactly represented by an SN of the current length. For example, the exact value of *ab* in Fig. 1 is 0.63, but no 10-bit SN can represent this number (only 0.6 or 0.7 are possible). *Random fluctuation errors* occur when calculating *cd* in Fig. 1: its exact value is $0.08 \approx 0.1$, but due to the specific positions of 1s in the SN representation, a (rather inaccurate) approximation by 0.2 is computed instead.

B. Correlation-induced inaccuracy

Consider realizing $f(x) = x^2$ by an SC (a squarer). Fig. 2a implements it by feeding two copies of the SN representing *x* into an AND gate, which might be expected to multiply them. However, this circuit will keep every bit position of *x* unchanged (illustrated by the 4-bit SN $x_1x_2x_3x_4$ in the figure), resulting in the incorrect function $f'(x) = x$ instead of $f(x) = x^2$. This problem can be resolved by generating an independent SN with the same value as *x*, e.g., by converting *x* to binary and back to stochastic using a new SNG, however the cost of such regeneration is extremely high. A lower-cost approach is to add an isolator flip-flop *i* to shift one of the SNs by one bit-position, as shown in Fig. 2b [6]. Then, the SNs at the AND gate's inputs are $x_1x_2x_3x_4$ and $ix_1x_2x_3$, where *i* is the initial value of the isolator. Consequently, the circuit will compute $x_j \cdot x_{j-1}$ in clock cycle $j > 1$. The 1-probability of every x_j is the same as for the entire SN *x*, and all bit-positions of *x* are independent of each other; therefore, the computed value is indeed x^2.

Isolators have to be placed carefully. Fig. 2c shows a "quartic" circuit intended to implement $x^4 = (x^2)^2$ by cascading two (correct) squarers from Fig. 2b; the isolators in the two copies are called i_A and i_B. For 4-bit SNs, the first squarer calculates $y = y_1y_2y_3y_4 = x_1x_2x_3x_4 \cdot i_Ax_1x_2x_3$. The second squarer will calculate $z = z_1z_2z_3z_4 = y_1y_2y_3y_4 \cdot i_By_1y_2y_3 = (x_1x_2x_3x_4 \cdot i_Ax_1x_2x_3) \cdot (i_Bx_1x_2x_3 \cdot i_Bi_Ax_1x_2)$. It can be seen that, for all output bits $j > 2$, $z_j = x_j \cdot x_{j-1} \cdot x_{j-1} \cdot x_{j-2}$, which is identical to $z_j = x_j \cdot x_{j-1} \cdot x_{j-2}$. Consequently, the circuit calculates the function $z = x^3$ instead of x^4. A sufficiently decorrelated version of the circuit with three isolators i_C, i_D and i_E is shown in Fig. 2d.

Adding isolators turns a combinational circuit into a sequential circuit without feedback (a pipeline). Individual bits of an SN can be considered independent events (Bernoulli experiments) in the combinational but not in the sequential case. For example, the first two bits $z_1 = x_1 \cdot i_A$ and $z_2 = x_2 \cdot x_1$ calculated by the squarer of Fig. 2b are not independent because both depend on x_1. This further highlights the importance of applying accuracy management to sequential circuits.

C. Accuracy-driven SC design exploration

In the past, most SC-related research tackled specific problems, like artificial neural network simulation [4] or LDPC decoding [14]. More recently, interest has shifted towards the definition of synthesis flows that are as generic as possible. They aim to take a user-specified target function as input and automatically produce an SC that implements this function. A number of different approaches are now available, including the reconfigurable architecture based on Bernstein polynomials [15], the combinational synthesis algorithm STRAUSS [1], and the ma-

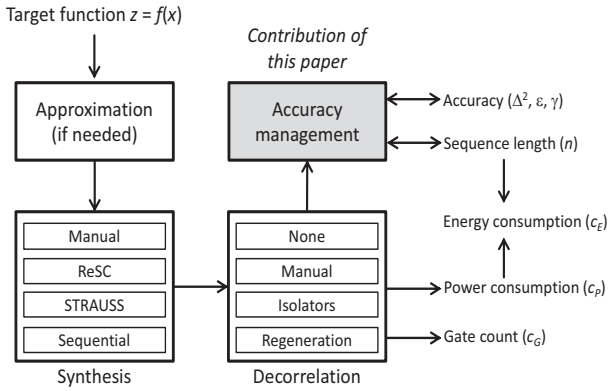

Fig. 3: Accuracy-driven synthesis flow.

pping procedure in [18]. These methods allow further design choices: it may be possible to share some SN generators, thus radically reducing the cost of the circuit but introducing undesired correlations, which can be in turn addressed by the decorrelation methods mentioned above. Moreover, some designs deliberately introduce carefully crafted correlations and by doing so achieve cost savings [3][6].

Fig. 3 shows the role of the accuracy-management framework in the overall SC design flow. The specification (target function *f*) is first approximated by a multilinear polynomial, a Bernstein polynomial, a polynomial quotient, or a different applicable formalism. (If the function is in the required shape already, this step is omitted.) Then, a circuit is generated either manually or using one of the above-mentioned automatic synthesis procedures. Finally, a decorrelation strategy is decided and appropriate circuitry is inserted.

Each combination of design choices made so far results in a digital circuit characterized by the cost measures gate count c_G and power consumption c_P. At this point, the accuracy management framework from this paper can be applied to each circuit under consideration to establish the relationship between accuracy (mean-squared error Δ^2, error ε, and confidence γ) and SN length n. For each circuit a number of consistent combinations of Δ^2, ε, γ and n will be generated. Now, the designer can explore the design space by balancing the accuracy numbers against cost measures c_G, c_P and, if needed, derived cost metrics such as energy consumption per computation.

III. Direct Simulation

The theoretical basis for our analysis is standard Monte-Carlo simulation, referred to as *direct simulation* (DS) in [11][13]. It provides a well-understood mathematical framework for stochastic processes and is especially well suited to stochastic computation. Its basic notion is a *base experiment* (BE) that is repeated a given number of times in order to calculate a desired value. In the context of this paper, an n-bit SN is modeled as a series of BEs. An introduction to DS can be found in [13], which is the source for the following definitions and equations.

A real-valued and integrable random variable Y with expected value $E(Y) = f(x)$, where $f(x)$ is the target function for a specific input value x, is a base experiment for the computation of $f(x)$. Let $Y_1, \dots, Y_n$ be independent identically distributed random variables with probability distribution $P_{Y_1} = P_Y$. The random variable

$$D_n = \frac{1}{n} \sum_{i=1}^{n} Y_i \tag{1}$$

is a direct simulation for the computation of $f(x)$ with n repetitions. For example, the target function in Fig.1 is $(ab + cd)/2$ and the BE $Y = rAB + (1 - r)CD$, where r is a Bernoulli variable with probability 0.5 and A, B, C, D are Bernoulli variables with probabilities a, b, c and d.

The *mean-square error* (MSE) Δ^2 of a DS can be directly deduced from the variance σ^2 of its BE:

$$\Delta^2(D_n) = \frac{1}{n} \cdot \sigma^2(\mathrm{Y}) \tag{2}$$

The proof for this and the following equations can be found in [13]. For the circuit of Fig. 1, $\sigma^2(Y) = z - z^2$ where z is the expected result of the computation and $n = 10$, which leads to $\Delta^2(D_n) = 0.023$.

MSE describes the average error if a stochastic circuit is run a large number of times with the same input values. In the SC design context, the accuracy of a *single* stochastic computation is more relevant. This is captured by two parameters: *precision* (absolute error) ε and *confidence level* 1 – γ. Direct simulation relates ε and γ to n and provides several techniques to estimate one of the three parameters ε, γ and n from the other two [13]. We use the method based on the Central Limit Theorem (CLT), which provides approximations for the parameters but can slightly overestimate the circuit accuracy. However, our simulations show that the approximations are very good for commonly used SN lengths and the possible overestimation is very small. We also investigated further methods based on the Chebyshev and Hoeffding inequalities [13] but found that they perform worse than CLT in the SC context.

The parameter $p_n(\varepsilon)$ describes the probability that a single DS with n repetitions differs from its expected value (as defined by target function *f*) by more than ε. In other words, it is the confidence level for a given precision ε. The CLT method estimates this value as

$$p_n(\varepsilon) \approx 2 \cdot \left(1 - \Phi\left(\frac{\sqrt{n} \cdot \varepsilon}{\sigma(Y)}\right)\right) \tag{3}$$

with the standard normal cumulative distribution Φ.

Conversely, given a confidence level $1 - \gamma$, the precision ε is estimated by substituting γ for $p_n(\varepsilon)$ in the above equation and solving for ε, which leads to

$$\varepsilon \approx \Phi^{-1}\left(1 - \frac{\gamma}{2}\right) \cdot \frac{\sigma(Y)}{\sqrt{n}} \tag{4}$$

which can also be used for constructing confidence intervals $[D_n - \varepsilon, D_n + \varepsilon]$. Furthermore, it is possible to estimate the minimum length n^* of SNs required to achieve a desired precision ε with a confidence level $1 - \gamma$ by solving the above equation for n:

$$n^* \approx \left(\Phi^{-1}\left(1 - \frac{\gamma}{2}\right) \cdot \frac{\sigma(Y)}{\varepsilon}\right)^2 \tag{5}$$

These parameters are important to properly estimate and manage the accuracy of an SC, and the equations provide an understanding of their interaction, which is not seen in previous work on this subject. For example, if one is interested in the

confidence level with which the circuit of Fig. 1 achieves the precision $\varepsilon = 0.1$ using SNs of length $n = 100$, (3) yields $p_{100}(0.1) \approx 0.037$, which corresponds to a confidence level of $1 - 0.037 = 0.963$. To estimate the precision achievable with a confidence level of 90%, (4) should be used; the result is $\varepsilon \approx 0.079$. Finally, the designer might want to know the minimum SN length that will provide precision $\varepsilon = 0.1$ with a confidence of 90%; (5) gives $n^* \approx 62$.

IV. Quantifying and Controlling Accuracy

A single stochastic computation with n-bit SNs can be seen as the average of n independent computations with SNs of length one, which is the basic principle of the underlying simulation theory. For our accuracy management techniques, we consider three SC categories: combinational circuits, sequential circuits without feedback, and sequential circuits with feedback.

A. Combinational circuits

In combinational SCs, each output bit is independent of all output and input bits in previous clock cycles. The BE Y of a circuit C is therefore equal to the value of the output Z of the circuit after one clock cycle, which represents an SN of length one. In the unipolar case, a bit Z is equal to its value as a SN. If bipolar representation is used, a function v maps Z to its SN value, as follows:

$$v(Z) = \begin{cases} 1, & \text{if } Z = 1 \\ -1, & \text{if } Z = 0 \end{cases} \quad (6)$$

Hence, $Y = v(Z)$ and

$$\sigma^2(Y) = E(Y^2) - E^2(Y) = 1 - z^2 \quad (7)$$

where z is the expected output of C, i.e., the value that is to be computed. For unipolar representation, this equation changes to

$$\sigma^2(Y) = z - z^2$$

It is surprising to see that the accuracy of a combinational SC depends only on its expected output value, whereas parameters such as its number of inputs and gate count have no influence. In particular, different combinational SC implementations of the same target function always have the same accuracy, and it is impossible to improve accuracy by re-synthesis. Note, however, that the value z is normally input-dependent; therefore, the accuracy of the circuit is determined by the worst possible z that can occur.

Using the equations from Sec. III, we can derive the MSE of any combinational stochastic circuit as

$$\Delta^2(C) = \frac{1 - z^2}{n} \quad (8)$$

Further, using the equations for $p_n(\varepsilon) = \gamma$, ε and n^*, one can manage a circuit's accuracy by tweaking the values γ, ε and n^* in a way that the desired precision and/or SN length is reached.

Table 1 gives minimum sequence lengths in the unipolar and bipolar domains calculated using (5); note the remarkable fact that it applies to *arbitrary* combinational circuits. Conversely, if SN length is fixed, we can calculate pairs of consistent (ε, γ) pairs, and thereby manage the tradeoff between precision and confidence. For example, if the circuit in Fig.1 is simulated with $n = 100$, a precision $\varepsilon = 0.079$ can be reached with confidence 90%, or $\varepsilon = 0.094$ with confidence 95%.

Table 1: Approximate minimum SN lengths for combinational SCs to reach precision ε with confidence $1 - \gamma$.

Accuracy		n^*	
ε	γ	Unipolar	Bipolar
0.2	0.1	16	68
0.2	0.05	22	97
0.1	0.1	62	271
0.1	0.05	88	385
0.05	0.1	248	1083
0.05	0.05	352	1537

B. Sequential circuits without feedback

A major class of sequential SCs without feedback is formed by combinational circuits with isolators added for decorrelation, as in Fig. 2d. Direct simulation is not readily applicable to such circuits because different clock cycles are no longer independent and so do not satisfy the BE assumption.

We solve this problem by constructing a time-frame expansion [5] of the circuit (shown in Fig. 4 for three time frames), which is a combinational equivalent of the sequential one and construct the BE for this circuit. In the general case of k time frames, this leads to

$$Y = \frac{(z_1 + z_2 + \ldots + z_k)}{k} \quad (9)$$

with $z_1 = x_1 i_C i_E$, $z_2 = x_1 x_2 i_D$, $z_3 = x_1 x_2 x_3 i_C$, and $z_j = x_{j-3} x_{j-2} x_{j-1} x_j$ for $j \in \{4, \ldots, k\}$. After an initial warm-up time of three clock cycles it can be assumed that $E(i_C) = x$ and $E(i_D) = E(i_E) = x^2$, because the initial states of the isolators have been shifted out. All inputs are independent of each other and $E(x_1) = E(x_2) = \ldots = E(x_k) = x$. At this point,

$$\sigma^2(Y) = \frac{E((z_1 + z_2 + \ldots + z_k)^2)}{k^2} - x^8 \quad (10)$$

which leads to

$$\sigma^2(Y) = \frac{k x^4 + 2(k-3)x^5 + 2(k-1)x^6 + 2(k-2)x^7 + (12 - 7k)x^8}{k^2} \quad (11)$$

DS for sequential circuits without feedback therefore consists of only a single execution of the BE. In this case, n (and n^*) in Eqs. 3–5 become 1, and the SN length is represented by k, the number of time frames. With this modification, managing the circuit's accuracy becomes similar to the combinational case.

C. Sequential circuits with feedback

Application to sequential circuits with feedback follows the same principle as in Sec. IV.*B*. However, in the feedback-free case, only sets of at most $l + 1$ terms (with l being the pipeline depth) are needed. In general circuits with feedback, all outputs may depend on inputs from all previous cycles. However, the calculation of variance (needed to compute MSE, p_n, ε and γ) is still possible, even though the equations are more complex.

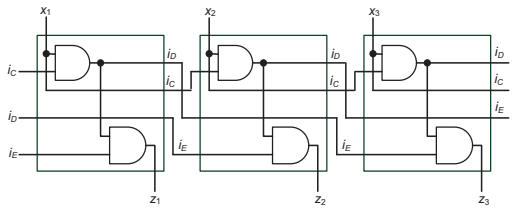

Fig. 4: Time-frame expansion for the circuit in Fig. 2d.

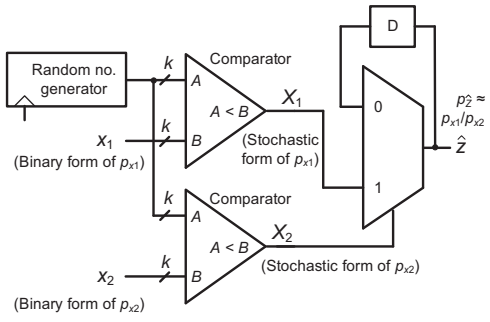

Fig. 5: Unipolar divider [7]: feedback is via the D flip-flop.

As an example of a sequential circuit with a feedback, we use the unipolar CORDIV divider [7] shown in Fig. 5. We constructed the time-frame expansion of CORDIV as in Fig. 4 and calculated the expression for variance using the computer algebra system SAGE. The results presented in detail in Sec. V.*C* demonstrate a good match between theoretical predictions and simulation for sequential circuits with feedback.

V. Experimental Results

We validated our theoretical predictions and the accuracy management by simulating various circuits from the three categories: combinational, sequential without feedback, and sequential with feedback. All simulations have been performed in the programming language R

A. Combinational circuits

We simulated the circuit for complex-valued matrix multiplication from [16] as an example of a combinational SC; see Fig. 6. We discovered that our method is not only useful for designing and analyzing SCs, but also for detecting design errors. To validate the circuit of Fig. 6, we simulated it with n = 271, (which implies a precision of 0.1 and a confidence level of 90% according to Table 1), and a set of randomly generated input values. We counted the ratio of simulations out of 10,000 repetitions with independently generated SNs that differed by more than ε from the expected result. If the circuit is correctly specified, this ratio should be around 0.1 (or less because worst-case inputs are assumed). However, in our sample simulation the ratios for the outputs (o1i, o1r, o2i, o2r) were (0.15, 0.16, 0.41, 0.40), which points to a design error. We checked the circuit by hand and found that the multiplexers before each output were misconnected. For example, output o1i should compute ((ar·i1i + ai·i1r) + (br·i2i + bi·i2r)).

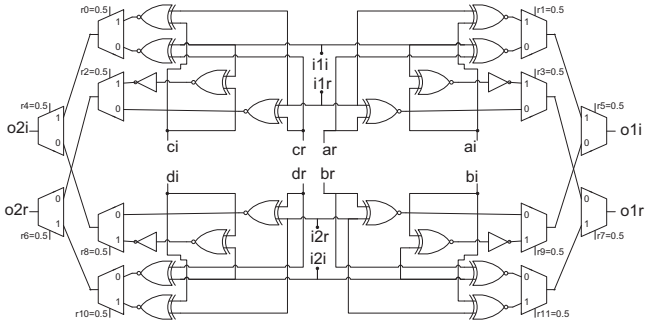

Fig. 6: Circuit for complex vector-matrix multiplication [16].

To do so, the multiplexer that drives o1i should be connected to the multiplexers with the select inputs r1 and r11, but it is connected to the multiplexers with select inputs r1 and r9. Simulating the corrected circuit with the same set of input values led to ratios of (0.09, 0.10, 0.09, 0.10). (We contacted the authors of [16] who confirmed that the observed error was introduced when the circuit diagram was incorrectly transcribed from a correct design.)

It might also be possible to identify a design error by running simulations with different input values and SN lengths without quantifying accuracy beforehand. However, if the impact of the design error on the output value is small, the error can easily be overlooked or attributed to the "natural" inaccuracy of stochastic computation. Knowing the expected error and the associated confidence level beforehand makes it far easier to trigger an alarm and invest effort in validation when suspicious simulation results are observed.

B. Sequential circuits without feedback

We determined confidence intervals for the x^4 circuit of Fig. 2d by using (4) with various values for γ. The input value x was chosen randomly in the range $[0, 1]$. Fig. 8 shows an example for values of $\gamma = 0.1$ and a maximum length $n = 150$ with input value $x \approx 0.77$. The initial states of the isolators were random with probability 0.5, and no warm-up period was allowed. Confidence intervals were calculated every 5 bits. The predicted confidence intervals describe the circuit's behavior very well for small SN length n.

Our methods can also help to detect design errors in sequential circuits. Suppose a designer builds the improperly decorrelated circuit in Fig. 2c, and wrongly assumes that it computes x^4. Then suppose (s)he wants to calculate a minimum length n^* using the equations from Sec. III. For this purpose, the designer constructs the circuit's k-time-frame expansion and derives an expression for σ^2 as the function of k, as in (10): $\sigma^2(Y) = ((k-1)x^3 + (2k-3)x^4 + 2(k-2)x^5 + (k-4)(k-3)x^6 + 2x^6 + 2(k-3)x^7) / k^2 - x^8$. As in Section IV.*B*, the SN length is given by k. It is obtained by setting n^* in (5) to 1 and solving the resulting equation for k:

$$\left(\Phi^{-1}\left(1-\frac{\gamma}{2}\right)\cdot\frac{\sigma(Y)(k)}{\varepsilon}\right)^2 \approx 1 \qquad (12)$$

Eq. 12 can be rearranged into a quadratic equation of the form $ak^2 + bk + c \approx 0$ with $a = 1 - gx^6 + gx^8$, $b = -gx^3 - 2gx^4 - 2gx^5 + 7gx^6 - 2gx^7$, $c = gx^3 + 3gx^4 + 4gx^5 - 14gx^6 + 6gx^7$ and

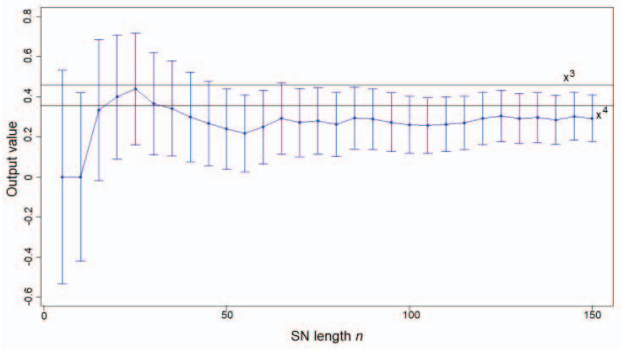

Fig. 7: Confidence intervals for the quartic circuit of Fig. 2d. Blue dots show the simulated output for n bits. Black lines show the expected values of x^3 and x^4.

Table 2: MSE for the unipolar CORDIV divider comparing the predicted and simulated results.

Input values		MSE	
X_1	X_2	Predicted	Simulated
0.2780	0.6197	0.0108	0.0106
0.1370	0.2656	0.0305	0.0306
0.0368	0.3707	0.0075	0.0077
0.3890	0.9477	0.0054	0.0053
0.7180	0.7933	0.0026	0.0026

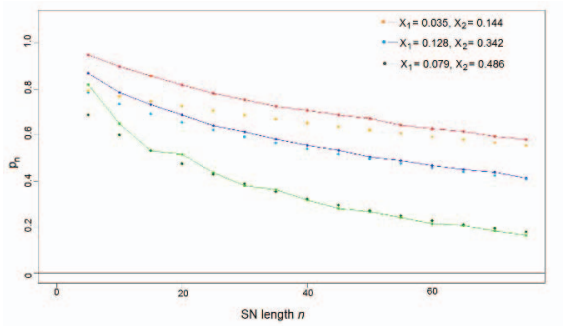

Fig. 8: Comparison between predictions and simulations for p_n in the case of the CORDIV divider.

$g = (\Phi^{-1}(1 - \gamma/2))^2 / \varepsilon^2$. However, solving this equation exposes inconsistencies. In this case, most input values x lead to either complex or negative solutions. For example, if $x = 0.18$ and $\varepsilon = \gamma = 0.1$, solving (12) yields the complex result $k \approx (2.19 \pm \sqrt{-5.19})/1.98$. This clearly tells the designer that a mistake has been made.

C. Sequential circuits with feedback

In this category we simulated the unipolar divider from Fig. 5. We compared its predicted MSE to the simulation results for random inputs in the interval $[0, 1]$ for both X_1 and X_2 with the restriction $X_1 < X_2$. Fifty sets of input values were generated and 10,000 repetitions were simulated for each set with independently generated input SNs. The SN length was 50 and the initial state of the flip-flop was randomly set with probability 0.5. No warm-up period was included. Some sample results are given in Table 2.

Concerning predictions for single simulations, the predicted confidence level $p_n(\varepsilon)$ was compared to the simulated level. For this experiment, input values were generated according to the previous simulation, and SN lengths up to 75 bit were simulated. For each set of input values 10,000 repetitions with independently generated SNs were performed, and the ratio of simulations where the output value differed by more than ε from the expected result was determined for various SN lengths. Fig. 8 compares the simulated (colored lines) and predicted values of $p_n(\varepsilon)$ (dots) for a sample value of $\varepsilon = 0.1$ and three sets of input values. Because our prediction method is based on the CLT, which is a statement about convergence, significant deviations between prediction and simulation for very small n are common but, in general, the predictions converge quickly toward the simulated results.

VI. Conclusions

We have investigated a central issue in stochastic circuit synthesis, which has received very little attention in the past: namely, the measurement and control of accuracy. Using Monte Carlo simulation theory, we developed a generic and practical way to relate accuracy to the stochastic number length. This method serves as a foundation for a synthesis framework that makes it possible for a designer to systematically balance accuracy against other design parameters. We derived precision-versus-length relationships that are universally valid for combinational circuits. Moreover, we demonstrated the application of our techniques to the important but little-understood area of sequential stochastic circuit design, which includes decorrelated combinational circuits. The validity of the theoretical predictions was confirmed by simulations for representative classes of circuits. Moreover, we showed their usefulness in the context of stochastic circuit validation by providing well-grounded hints to indicate hard-to-detect design errors.

Acknowledgments: John P. Hayes was supported in part by a Research Award from the Humboldt Foundation and by NSF grant CCF-1318091.

References

[1] A. Alaghi and J.P. Hayes. STRAUSS: Spectral transform use in stochastic circuit synthesis. *IEEE Trans. CAD,* 34: 1770-1783, 2015.

[2] A. Alaghi and J.P. Hayes. Survey of stochastic computing. *ACM Trans. Embedded Comp. Sys.*, 12: article 92, May 2013.

[3] A. Alaghi, C. Li and J.P. Hayes Stochastic circuits for real-time image-processing applications. *Proc. DAC*, article 236, 2013.

[4] B. D. Brown and H. C. Card, Stochastic neural computation I: Computational elements. *IEEE Trans. Comp.*, 50: 891-905 2001.

[5] M. Bushnell and V. Agrawal, *Essential of Electronic Testing*, Springer, 2002.

[6] T.-H. Chen and J. P. Hayes. Analyzing and controlling accuracy in stochastic circuits. *Proc. ICCD,* 367–373, 2014.

[7] T.-H. Chen and J. P. Hayes. Design of division circuits for stochastic computing.*Proc. ISVLSI*, 116-121, 2016.

[8] J. Chen and T. Hu. A novel FIR filter based on stochastic logic. *Proc. ISCAS*, 2050 - 2053, 2013.

[9] B.R. Gaines. Stochastic computing systems. *Advances in Info. Sys. Sci.*, 2: 37-172, 1969.

[10] V.C. Gaudet and A.C. Rapley. Iterative decoding using stochastic computation. *Electron. Letters*, 39: 299-301, 2003.

[11] J.M. Hammersley and D.S. Handscomb, *Monte Carlo Methods*, Methuen, 1964

[12] P. Li and D.J. Lilja. Using stochastic computing to implement digital image processing algorithms. *Proc. ICCD*, 154-161, 2011.

[13] T. Müller-Gronbach, E. Novak and K. Ritter. *Monte Carlo-Algorithmen*. Springer, 2012.

[14] A. Naderi et al. Delayed stochastic decoding of LDPC codes. *IEEE Trans. Signal Proc.*, 59: 5617-5626, 2011.

[15] W. Qian et al. An architecture for fault-tolerant computation with stochastic logic. *IEEE Trans. Comp.,* 60: 93-105, 2011.

[16] A. Paler et al. Approximate simulation of circuits with probabilistic behavior. *Proc. DFTS*, 95-100, 2013.

[17] M. Parhi, M. D. Riedel and K. K. Parhi. Effect of bit-level correlation in stochastic computing. *Proc DSP*, 465-467, 2015.

[18] N. Saraf and K. Bazargan. Polynomial arithmetic using sequential stochastic logic. *Proc. GLSVLSI*, 245-250, 2016.

[19] R. Wang et al. Stochastic circuit design and performance evaluation of vector quantization for different error measures. *IEEE Trans. VLSI*, early access article, 2