A Stochastic Computational Approach for Accurate and Efficient Reliability Evaluation

Jie Han, Hao Chen, Jinghang Liang, Peican Zhu, Zhixi Yang and Fabrizio Lombardi

Abstract— Reliability is fast becoming a major concern due to the nanometric scaling of CMOS technology. Accurate analytical approaches for the reliability evaluation of logic circuits, however, have a computational complexity that generally increases exponentially with circuit size. This makes intractable the reliability analysis of large circuits. This paper initially presents novel computational models based on stochastic computation; using these stochastic computational models (SCMs), a simulation-based analytical approach is then proposed for the reliability evaluation of logic circuits. In this approach, signal probabilities are encoded in the statistics of random binary bit streams and non-Bernoulli sequences of random permutations of binary bits are used for initial input and gate error probabilities. By leveraging the bit-wise dependencies of random binary streams, the proposed approach takes into account signal correlations and evaluates the joint reliability of multiple outputs. Therefore, it accurately determines the reliability of a circuit; its precision is only limited by the random fluctuations inherent in the stochastic sequences. Based on both simulation and analysis, the SCM approach takes advantages of ease in implementation and accuracy in evaluation. The use of non-Bernoulli sequences as initial inputs further increases the evaluation efficiency and accuracy compared to the conventional use of Bernoulli sequences, so the proposed stochastic approach is scalable for analyzing large circuits. It can further account for various fault models as well as calculating the soft error rate (SER). These results are supported by extensive simulations and detailed comparison with existing approaches.

Index Terms— B.2.3 Reliability, Testing, and Fault-Tolerance, B.7.2 Reliability and Testing, Error-checking, Fault injection, G.3 Probabilistic algorithms, Random number generation.

1 Introduction

THE nanometric scaling of CMOS technology has introduced substantial challenges in circuit design; the higher integration density and lower voltage/current thresholds have increased the likelihood of soft errors. Process variations have prominently emerged to impact the performance and degrade the reliability of electronic circuits [1]. Process variations are due to random dopant fluctuation or manufacturing imprecision in the CMOS fabrication process. These physical-level characteristics have subsequently resulted in probabilistic device and circuit behavior. Novel nanoelectronic devices (such as carbon nanotubes, silicon nanowires, graphene and molecular electronics) have non-deterministic characteristics due to the uncertainty inherent in their operational behavior, so emerging technologies have significant limitations for reliable operation. Reliability has, therefore, become a major concern and probabilistic design methodologies are needed for assembling reliable circuits and systems out of unreliable devices [2-4].

- *J. Han, H. Chen, J. Liang, P. Zhu and Z. Yang are with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada T6G 2V4. E-mail: {jhan8, hc5, jinghang, peican, zhixi} @ualberta.ca.*
- *F. Lombardi is with the Department of Electrical and Computer Engineering, Northeastern University, Boston, USA. E-mail: lombardi@ece.neu.edu.*

Manuscript received on June 16, 2011.

To meet this increasing demand on reliable design, several analytical approaches have been proposed for the reliability evaluation [5-15] and soft error rate (SER) analysis of logic circuits [16-26]. Soft errors are typically caused by temporary environmental phenomena, such as external radiation or power supply noise [23]. In contrast to the general definition of reliability, i.e., the probability of the correct functioning of a circuit, SER has been used as a measure on the vulnerability of a circuit under the influence of soft errors. While reliability evaluation techniques are essential at the core of an SER analysis, an SER analyzer often considers various technology-dependent factors such as the electrical and timing effects of single event upsets.

An analytical evaluation can be readily accomplished for small circuits with no loss of accuracy. As a circuit becomes large, it becomes difficult, if not impossible, to implement an exact analysis of its reliability. This is due to the signal correlation caused by reconvergent fanouts in combinational circuits and/or feedback loops in sequential circuits. Usually, a compromise is made on the accuracy of the evaluation, and simulation has emerged as a possible solution. In a simulation-based approach, experimental data are gathered to characterize the behavior of a circuit by randomly sampling its activity. As an example, Monte Carlo simulation (MCS) has been widely used when an analytical approach is not available or easy to use. A disadvantage of simulation using random vectors is that numerous pseudo-random numbers need to be generated and a large number of simulation runs must be executed to reach a stable output, so making the evaluation of large circuits a very time-consuming pro-

cess.

The design of nanometric integrated circuits requires tools that accurately and efficiently compute reliability; this is a stringent requirement in mission critical applications. For space systems, SER is often concerned. Hence, there is an urgent need to develop a unifying technical framework by which reliable design can be assessed with respect to different metrics (such as reliability and SER), while still retaining flexibility (as technology independence). A computational framework should also be applicable to a variety of fault models that can be encountered when designing such systems. Hence, permanent faults (such as stuck-at) and errors (such as of a transient/soft nature) should be handled. The proposed approach meets these objectives.

In this paper, a *simulation-based analytical approach* is presented for an accurate and efficient evaluation of the reliability of a circuit. Using random binary bit streams to encode signal probabilities, this approach originates from the mathematical formulations of stochastic computation [27, 28] and probabilistic gate models (PGMs) [7-9]. Stochastic computational models (SCMs) are constructed to implement the probabilistic analysis performed by PGMs, thus enabling an accurate analysis of circuit reliability. Differently from a traditional application of stochastic computation [29-32], this approach employs and leverages the bit-wise dependencies in the random binary streams to efficiently handle signal correlations caused by recovergent fanouts or feedback loops in logic circuits. Hence, this approach avoids the large complexity typically encountered in a traditional analytical approach. Although the SCM approach implements accurate analytical algorithms with an efficient simulation, its evaluation precision is limited by inherent features of stochastic computation, such as the quantization and resolution in the representation of the binary bit streams and the random permutation in the stochastic sequences. A detailed analysis and an extensive comparison with existing approaches show that the SCM approach offers considerable advantages with respect to accuracy, execution efficiency and flexibility, especially in the evaluation of large circuits.

In contrast to methods based only on the simulation of random vectors, SCMs explicitly carry out the computation of signal probabilities, so they are generic and versatile for use in both algorithmic development and applications. This feature is shown by modeling various fault types and evaluating the joint signal probability. In contrast to the conventional use of Bernoulli sequences in stochastic computation, non-Bernoulli sequences (as random permutations of fixed numbers of 1's and 0's) are used in this paper for initial input and gate error probabilities. It is shown by both analysis and simulation that the use of non-Bernoulli sequences significantly increases the efficiency and accuracy of the stochastic approach. The proposed approach is therefore advantageous in terms of evaluation efficiency and accuracy when compared to a random sampling method, such as the Monte Carlo simulation.

This paper is a significant expansion of [15] and is organized as follows. Section 2 reviews PGMs. Section 3 presents the new stochastic logic. Section 4 discusses the stochastic computational approach for circuit reliability analysis. Its efficiency and accuracy are assessed in Sections 5 and 6 respectively. Extensive simulation results are provided in Section 7 with a detailed comparison with existing approaches. Section 8 presents a discussion and Section 9 concludes the paper.

2 Probabilistic Gate Models (PGMs)

2.1 Probabilistic Logic

Most faults in nanometric logic circuits either are inherently probabilistic, or can be modeled probabilistically. Therefore, the reliability analysis of logic circuits has been based on the probabilistic treatment of signals [2]. The *signal probability* of an input or output of a logic gate is usually defined as the probability that the signal is logical "1." A *logic function* transforms its inputs to its output probability. The *reliability of an output* is defined as the probability of the output with an expected logic value of "1," or its complement otherwise. Given independent inputs, Boolean functions can be mapped to arithmetic operations of signal probabilities, by the following rules [2, 3]:

Rule I: Boolean "NOT," or $B = \bar{A}$, corresponds to

$$b = 1 - a, \tag{1}$$

where $b = P(B = 1)$ and $a = P(A = 1)$.

Rule II: Boolean "AND," or $C = AB$, corresponds to

$$c = a \cdot b, \tag{2}$$

where $c = P(C = 1)$, $b = P(B = 1)$ and $a = P(A = 1)$.

Rule III: Boolean "OR," or $C = A + B$, corresponds to

$$c = a + b - a \cdot b, \tag{3}$$

where $c = P(C = 1)$, $b = P(B = 1)$ and $a = P(A = 1)$.

However, if the input signals are not mutually independent, then the corresponding probability function may change. For example, the following rule maps the Boolean "AND" with two input signals that are totally dependent.

Rule IV: Boolean "AND" of a signal A with itself, or $C = AA$, corresponds to

$$c = a, \tag{4}$$

where $c = P(C = 1)$, and $a = P(A = 1)$.

Proof:

$c = P(C = 1) = P(A = 1, A = 1) = P(A = 1) = a.$ □

By applying "AND," "OR" and "NOT," any Boolean logic function can be mapped to an arithmetic equation of signal probabilities.

2.2 PGMs for Reliability Evaluation

A *probabilistic gate model* (PGM) relates the output probability of a gate to its input and error probabilities; this is accomplished according to the function and malfunction (such as in the presence of an error) of the gate [9]. In general, the output probability of a gate can be calculated by the following equation,

Z = P(output "1" | gate faulty)• P(gate faulty) +
P(output "1" | gate not faulty)• P(gate not faulty). (5)

Consider a *von Neumann fault*, i.e., a fault that flips the correct output of a gate and resembles the behavior of a soft error. Let ε denote the error rate, i.e., $\varepsilon = P(gate\ faulty)$, and p, the fault-free output probability, i.e., $p = P(output\ \text{“1”}|gate\ not\ faulty)$. The following equation is then applicable to any logic gate/function for the calculation of its output probability,

$$Z_v = (1-p)\cdot\varepsilon + p\cdot(1-\varepsilon). \tag{6}$$

Stuck-at faults can also be modeled in a PGM. For a stuck-at-1 fault, (6) becomes

$$Z_{SA1} = \varepsilon + p\cdot(1-\varepsilon). \tag{7}$$

For a stuck-at-0 fault, this is given by

$$Z_{SA0} = p\cdot(1-\varepsilon). \tag{8}$$

An accurate algorithm using the PGMs accounts for signal dependencies in a circuit [9]. If all inputs are mutually independent, reconvergent fanouts are the only topological structures that introduce signal correlations in a circuit with no feedback. Signal correlations can be eliminated by decomposing a circuit into two subcircuits for each reconvergent fanout. When all reconvergent fanouts are eliminated by this fanout decomposition, the gate PGMs can then be applied to obtain the reliability of the original circuit. As the required computation almost doubles for each reconvergent fanout, however, the PGM algorithm has a computational complexity that increases exponentially with the number of dependent reconvergent fanouts [9]. As applicable to any analytical approach, the accurate analysis of large circuits is therefore likely to be intractable due to its very large computational overhead.

3 Stochastic Logic using Non-Bernoulli Sequences

3.1 Stochastic Logic

In stochastic computation, signal probabilities are encoded into binary bit streams, i.e., serially in the time domain. Randomly generated bit streams are used to encode signal probabilities; a specific probability is represented by a number of bits set to a value that is usually in proportion to the mean number of 1's in a bit stream. Fig. 1 shows a stochastic encoding and an inverter. As Boolean operations can be mapped to arithmetic operations, the inverter probabilistically implements the complement operation of *Rule I*. Note that in Fig. 1, a sequence length of 10 bits is used for illustration purposes; a larger sequence length is usually needed in practice.

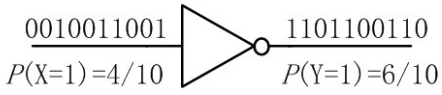


Fig. 1. An inverter and a stochastic encoding.

Stochastic computation transforms Boolean logic operations into probabilistic computations in the real domain. Although each binary bit is processed by a Boolean gate, signal operations are no longer Boolean in nature, but they are arithmetic computations by stochastic logic. Bernoulli sequences are often used as binary bit streams in stochastic computation [28, 29]. In a Bernoulli sequence, every bit is independently generated with a probability p. The mean and variance of the number of 1's in an N-bit Bernoulli sequence are respectively given by

$$\mu = Np, \tag{9}$$

and

$$v = Np(1-p). \tag{10}$$

For the inverter of Fig. 1 , if the input probability is a, the mean number of 1's in its output sequence is

$$\mu_1 = N(1-a), \tag{11}$$

and the variance is

$$v_1 = Na(1-a). \tag{12}$$

This is the same as the variance of the input sequence.

Complex arithmetic operations can be implemented by simple stochastic logic. According to *Rule II*, for instance, multiplication can be implemented by an AND gate, as shown in Fig. 2(b). In this multiplication, the input binary streams must not be correlated for a correct computation. However, the bit-wise dependencies between the input random binary streams can be used to yield new stochastic logic models that account for the statistical correlation in input signals. This is shown in Fig. 2(a) as a general stochastic model of AND in which the two input signals may be correlated.

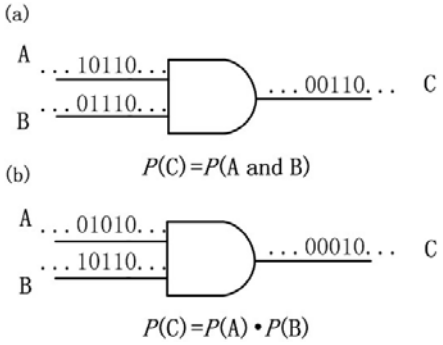


Fig. 2. Stochastic AND logic: (a) the general model; (b) the special case of multiplication, when the two inputs are statistically independent.

If the inputs of the AND are two independent Bernoulli sequences with generating probabilities a and b respectively, the mean number of 1's in the output sequence is:

$$\mu_2 = Nab, \tag{13}$$

and the variance is given by:

$$v_2 = Nab(1-ab). \tag{14}$$

For the AND gate in Fig. 2(a) with possibly correlated inputs,

$$\begin{aligned} P(C=1) &= P(A=1, B=1) \\ &= P(A=1)P(B=1|A=1). \end{aligned} \tag{15}$$

Let $a = P(A=1)$, $b = P(B=1)$ and $p_c = P(B=1|A=1)$; then

$$P(C=1) = ap_c. \tag{16}$$

The use of Bernoulli sequences as inputs results in a Bernoulli sequence at the output with a generating probability given by (16); therefore, the mean number of 1's in the output sequence and its variance are given by:

$$\mu_{2,g} = Nap_c, \quad (17)$$

and

$$v_{2,g} = Nap_c(1 - ap_c), \quad (18)$$

respectively.

The use of Bernoulli sequences however incurs a large computational overhead that severely limits its application for reliability analysis. This aspect is addressed through the use of non-Bernoulli sequences, as discussed next.

3.2 Non-Bernoulli Sequences

In this work, non-Bernoulli sequences are used for reducing the computational overhead and inaccuracy. Specifically, each initial input stochastic sequence contains a fixed number of 1's and the positions of the 1's are determined by a random permutation. For a given probability p and a sequence length of N bits, the number of 1's to be generated is given by Np. When Np is not an integer, it must be rounded to an integer, thus introducing a quantization error into the representation. The effect of quantization errors is discussed in a later section; the output distributions of the inverter and AND gate when non-Bernoulli sequences are used as inputs, are treated in more detail next.

For an inverter, assume that the input has a probability of a to be "1"; so Na is the number of 1's in the input sequence of N bits. Then the expected value of 1's in the output sequence is given by:

$$\mu_1' = N(1 - a). \quad (19)$$

Since there is no variation in the input, the variance in the output is considered to be 0, i.e.,

$$v_1' = 0. \quad (20)$$

For an AND gate, the use of the non-Bernoulli sequences resembles von Neumann's NAND multiplexing technique, as discussed in [33 – 37, 40] for fault-tolerant logic design. The following Lemma shows that its output follows approximately a Gaussian distribution when the sequence length N is large.

Lemma 1: For an AND gate, assume that the two inputs are "1" with probabilities a and b and represented by non-Bernoulli sequences of N bits (as random permutations of fixed numbers of 1's and 0's). For a large N, the output sequence follows a Gaussian distribution with a mean number of 1's given by:

$$\mu_2' = Nab, \quad (21)$$

and a variance:

$$v_2' = Na(1 - a)b(1 - b). \quad (22)$$

Proof: The two input probabilities a and b give r=aN and s=bN as the numbers of 1's in the input sequences. In these two inputs, the numbers of possible permutations are:

$$C_a = \binom{N}{r} = \frac{N!}{r!(N-r)!}, \quad (23)$$

and

$$C_b = \binom{N}{s} = \frac{N!}{s!(N-s)!}, \quad (24)$$

respectively. Assume that the AND gate produces t 1's in the output sequence; then, the number of permutations that causes this occurrence, C_o, can be obtained by combinatorial analysis [33, 40]. This leads to:

$$C_o = \binom{N}{t}\binom{N-t}{r-t}\binom{N-r}{s-t} = \frac{N!}{t!\,(r-t)!\,(s-t)!\,(N-r-s+t)!} \quad (25)$$

The probability that t 1's result in the output sequence, is given by the number of output permutations divided by the total possible number of input permutations, i.e.,

$$P(t) = \frac{C_o}{C_a C_b} = \frac{r!(N-r)!s!(N-s)!}{t!(r-t)!(s-t)!(N-r-s+t)!N!}. \quad (26)$$

Assume that the expected output probability is z, and therefore

$$z = \frac{t}{N}. \quad (27)$$

As per [33], the application of Stirling's formula results in:

$$P(z) \sim \frac{1}{\sqrt{2\pi N}}\sqrt{\beta}e^{-\theta N}, \quad (28)$$

where

$$\beta \sim \frac{1}{a(1-a)b(1-b)}, \quad (29)$$

$$\theta \sim \frac{(z-ab)^2}{2a(1-a)b(1-b)}. \quad (30)$$

(28), (29) and (30) indicate that the output sequence follows approximately a Gaussian distribution with a mean number of 1's given by (21) and a variance given by (22). □

3.3 Non-Bernoulli vs. Bernoulli Sequences

Next, the comparison between the use of Bernoulli and non-Bernoulli input sequences in stochastic logic is pursued. For an inverter, it is easy to find that (19) = (11) and (20) = 0. This indicates that the use of non-Bernoulli input sequences results in a deterministic output value equal to the mean value of the one by using Bernoulli input sequences. For an AND gate, the following theorem applies for independent inputs.

Theorem 1: Compared to the case when Bernoulli sequences are used to represent input probabilities, the use of large non-Bernoulli sequences as random permutations of fixed numbers of 1's and 0's results in an output sequence with the same mean number of 1's and a smaller variance for an AND gate when its inputs are independent.

Proof: From Lemma 1, it can be seen that (21) = (13) and

$$v_2 - v_2' = Nab(1 - ab) - Na(1 - a)b(1 - b) = Nab(a(1 - b) + b(1 - a)) \geq 0, \quad (31)$$

so proving the theorem. □

The general case of correlated inputs is considered as follows. When non-Bernoulli sequences are used as inputs, the random permutation allows for some randomness in the inputs, albeit with a correlation between them. Without loss of generality, assume that input A is first generated; input B is then generated conditionally on A. For the 1's in the sequence of A, further assume that the corresponding bits in B are generated as a Bernoulli sequence with probability p_c. For the 0's in the sequence of A, subsequently, the number of 1' in the corresponding bits in B is actually determined due to the nature of the non-Bernoulli sequence used to represent the input B. Since the number of 1's in the sequence of A is Na, the mean number of 1's in the corresponding bits in B and its variance are given by:

$$\mu_{2,g}' = Nap_c, \quad (32)$$

and

$$v'_{2,g} = Nap_c(1-p_c). \quad (33)$$

The combinations of 1's in inputs A and B produce the 1's in the output sequence, so the mean number of 1's at the output and the variance are given by (32) and (33) respectively for an AND gate with non-Bernoulli input sequences that may be correlated.

Hence, it can be seen that (17) = (32) and from (18) and (33),

$$v_{2,g} - v'_{2,g} = Nap_c(1-ap_c) - Nap_c(1-p_c) = Nap_c{}^2(1-a) \geq 0. \quad (34)$$

This indicates that, when compared to Bernoulli input sequences, the use of non-Bernoulli input sequences as random permutations of fixed numbers of 1's and 0's results in an output sequence with the same mean number of 1's and a smaller variance for an AND gate when its inputs may be correlated.

As in [29], the accuracy of an evaluation result can be measured by the coefficient of variation (CV). The CV is defined as the ratio between the standard deviation and the mean, i.e.,

$$CV = \frac{\sigma}{\mu}. \quad (35)$$

For an AND gate, the following corollary applies for independent inputs.

Corollary 1: For a specific accuracy given by a CV, the use of large non-Bernoulli sequences as random permutations of fixed numbers of 1's and 0's requires a smaller sequence length than using Bernoulli sequences for the independent input probabilities of an AND gate.

Proof: As per (21), (22) and (35), the required sequence length for the use of non-Bernoulli sequences is given by:

$$N_{NB} = \frac{(1-a)(1-b)}{ab(CV)^2}. \quad (36)$$

Similarly, the required sequence length for using Bernoulli sequences is given by:

$$N_B = \frac{1-ab}{ab(CV)^2}. \quad (37)$$

It is easy to show that $N_B > N_{NB}$ for $a \neq 1$ or $b \neq 1$, thus proving the corollary. □

Based on the analysis leading to (34), it can be shown that the use of non-Bernoulli sequences requires also a smaller sequence length than using Bernoulli sequences for an AND gate with correlated inputs.

Any logic function can be implemented with inverters and AND gates; so, a smaller variance in the output of AND gates (as achieved by using the non-Bernoulli inputs) will result in a smaller variance in the output of a function implemented with inverters and AND gates. Also, the same mean value results from the use of non-Bernoulli and Bernoulli inputs. In a logic network, therefore, the use of non-Bernoulli and Bernoulli sequences as initial inputs will produce evaluation results with the same mean, but different variance; the former method results in a smaller variance than the latter method.

We conjecture this result as follows: compared to the case when Bernoulli sequences are used to represent initial input probabilities, the use of large non-Bernoulli sequences as random permutations of fixed numbers of 1's and 0's results in an output sequence with the same mean number of 1's and a smaller variance for a combinational logic network. Therefore, the use of non-Bernoulli sequences requires a smaller sequence length at a desired accuracy.

3.4 Signal Correlations in Stochastic Logic

Signal correlations are accounted in stochastic logic, as shown as follows. If an AND gate has two independent random bit streams X_1 and X_2 as inputs, then its output will be a sequence encoding $Z = X_1X_2 = 0.81$ for $X_1 = X_2 = 0.9$. If the inputs are not independent, the output will depend on the correlation of the two input signals. If the two inputs are fully dependent, as shown in Fig. 3, then it results in $Z = X_1 = X_2 = 0.9$. This complies with *Rule IV*.

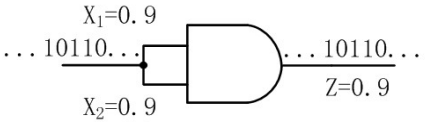


Fig. 3. Signal correlation in stochastic logic processing: X_1 and X_2 are fully correlated inputs.

A more general case of signal correlation is caused by a reconvergent fanout, as shown in Fig. 4. In this case, even though the inputs A, B and C are independent, the random binary streams carrying the signals of X_1 and X_2 originate from the same root (i.e. B); therefore, they are statistically correlated. Due to the bit-wise dependency in the two correlated stochastic sequences, the signal correlation is accounted in the output sequence when the signals reconverge at the output D. This output probability can be calculated as $D = P(X_1 and\ X_2) = P\big((A\ and\ B)\ and\ (B\ and\ C)\big) = P(A\ and\ B\ and\ C) = 0.5 \times 0.8 \times 0.5 = 0.2$. This is different from the independent case; if X_1 and X_2 were independent, the output probability would be $D = X_1X_2 = 0.1$.

This feature of stochastic computation is applicable to any logic function. For example, a stochastic OR gate implements the probabilistic analysis given by *Rule III* as a special case for independent inputs; in the general case, it computes $P(C) = P(A\ or\ B)$.

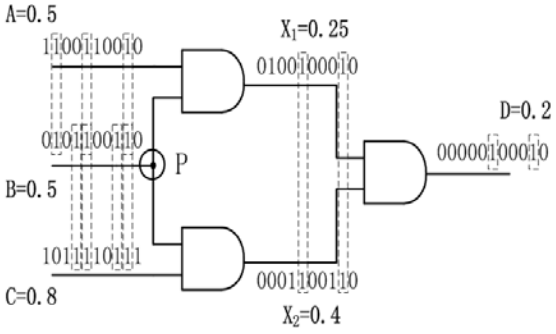


Fig. 4. A general case of signal correlation in stochastic logic processing: the correlated signals are X_1 and X_2.

A stochastic XOR gate computes $P(C=1) = P((A=1\ and\ B=0)\ or\ (A=0\ and\ B=1))$ in general; for independent inputs, it computes $P(C=1) = P(A=1)\cdot\big(1-P(B=1)\big) + (1-P(A=1))\cdot P(B=1)$ as

a special case. Hence, stochastic logic implements a corresponding probabilistic operation as dictated by a mapping rule or a combination of rules; at the same time, it maintains the signal correlations present in the random binary bit streams.

4 A STOCHASTIC APPROACH FOR CIRCUIT RELIABILITY EVALUATION

4.1 Stochastic Computational Models (SCMs)

This computational capability of stochastic logic allows the numerical evaluation of circuit reliability using *stochastic computational models* (SCMs). SCMs are based on the operation of stochastic logic and the notions of PGMs. As discussed previously, any gate affected by a von Neumann fault can be modeled by (6). In fact, (6) can be implemented by the stochastic logic of an XOR gate [15], as follows:

$$XOR_{sto}(p, \varepsilon) = p(1 - \varepsilon) + (1 - p)\varepsilon, \quad (38)$$

where p is the fault-free output probability and ε is the gate error rate. The special case of a stochastic XOR is used to compute (6) because gate errors are assumed to occur independently. The general model must be used if there is a correlation between the gate error and the input signals. (38) shows that *regardless* of the type of logic gate modeled by PGM, (6) can be implemented by a stochastic XOR logic. Therefore, an SCM can be obtained by adding an XOR gate to an unreliable gate and using an input of XOR to implement the gate error rate. This is shown in Fig. 5, in which an unreliable AND gate (Fig. 5(a)) is implemented by a general stochastic structure (Fig. 5(b)) and an SCM with an XOR gate (Fig. 5(c)). In this case,

$$p = P(X_1 = 1 \; and \; X_2 = 1), \quad (39)$$

and the XOR gate computes (38).

In addition to the von Neumann fault that was originally modeled in [15], the stuck-at faults can also be modeled by SCMs. For (7) considering a stuck-at-1 fault, an SCM can be constructed by adding an OR gate to the unreliable gate and using an input of the OR to implement the gate error rate, as

$$OR_{sto}(p, \varepsilon) = p + \varepsilon - p \cdot \varepsilon = \varepsilon + p \cdot (1 - \varepsilon). \quad (40)$$

For a stuck-at-0 fault, AND and NOT gates are used to implement the function of (8):

$$AND_{sto}(p, \vec{\varepsilon}) = p \cdot (1 - \varepsilon). \quad (41)$$

The SCMs for an unreliable AND gate affected by stuck-at-1 and stuck-at-0 faults are shown in Fig. 5 (d) and (e) respectively.

As indicated in (38), (40) and (41), an SCM is universal, because it can be constructed for an arbitrary logic gate. Moreover, it also masks errors through the function of a logic gate; so logic masking is explicitly considered in an SER analysis. Signal correlations are inherently accounted, so the use of SCMs significantly reduces the computational complexity of a probabilistic analysis by using redundancy in the time domain and stochastic logic for processing and calculating the gate error rate.

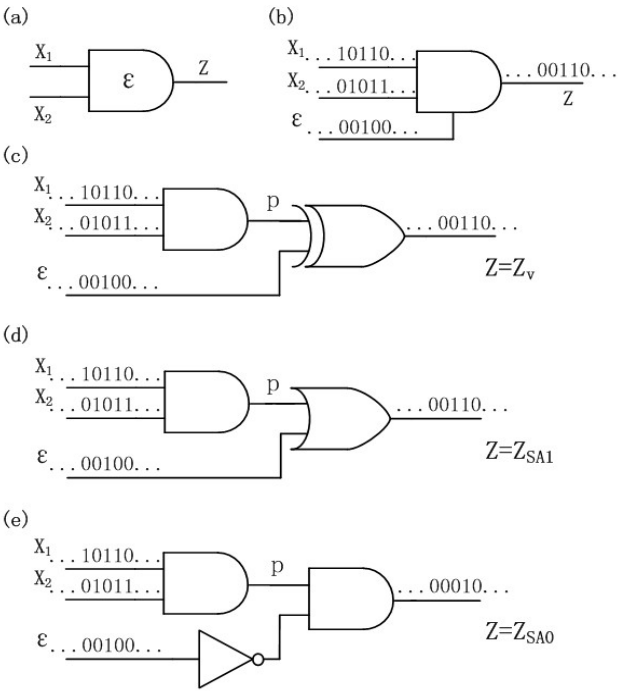


Fig. 5. (a) An unreliable AND gate; (b) A stochastic logic implementation; (c) A stochastic computational model (SCM) for the von Neumann fault; (d) An SCM for the stuck-at-1 fault; (e) An SCM for the stuck-at-0 fault.

4.2 Reliability Evaluation by SCMs

A stochastic computational network can be constructed using the SCMs for reliability evaluation of a circuit. The output probabilities are obtained by using stochastic sequences as inputs and propagating them from the primary inputs to the outputs. A logic circuit may contain more than one primary output. Individual output reliabilities have been considered in [15]; in this paper, we consider the joint output reliability. As output signal probabilities are encoded by the proportion of 1's in the output stochastic sequences, signal correlation is preserved in the distribution pattern. Let *A, B, C, D* be the output signals that may be correlated, then the output of a stochastic AND logic is given by

$$AND_{sto}(A, B, C, D) = P(A = 1, B = 1, C = 1, D = 1). \quad (42)$$

By (42), a stochastic AND gate produces an output sequence containing the common 1's in all outputs; so it evaluates the *joint output probability* of *A, B, C* and *D*, i.e., the probability that all outputs are "1." A *joint probability* of the outputs can thus be calculated by applying a stochastic AND, that takes into account the signal correlations among output signals. (42) is still applicable when signal correlations among a few (but not all) outputs are of interest [22], i.e., by specifying the irrelevant outputs as sequences of all 1's.

As the output signal probability is the probability of the output being "1," the output reliability is the output signal probability if the fault-free output is expected to be 1 (or, the complement of the output signal probability otherwise). Hence, a stochastic XOR gate with one inverted input is used to convert the output probability

into reliability by either keeping or flipping the output sequence from each output of an unreliable circuit according to the correct output value (as generated by the equivalent fault-free circuit). The *joint output reliability* can then be obtained by the output sequence of the AND gate that takes the outputs of the XOR gates as (correlated) input probabilities. The proposed SCM approach is demonstrated by an example of the benchmark circuit C17, as shown in Fig. 6. The von Neumann fault is used for illustration purposes.

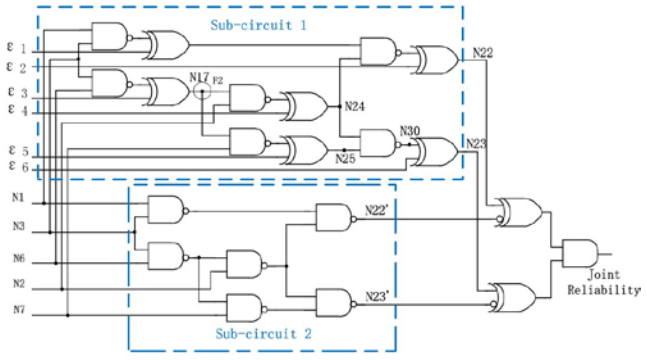


Fig. 6. A stochastic architecture using SCMs for the evaluation of circuit reliability (for C17). Sub-circuit 1: the stochastic computational circuit; sub-circuit 2: the original fault-free circuit.

A general evaluation procedure is as follows:

1. Construct the stochastic computational architecture by adding stochastic logic gates according to a specific type of fault (Fig. 5), as well as XOR and AND gates for obtaining the joint output reliability;
2. Generate initial random bit streams encoding signal probabilities of the primary inputs and gate error rates in the circuit;
3. Propagate the binary streams from the primary inputs to the output and obtain the stochastic bit stream at the output;
4. Decode the signal probability as the joint output reliability of the circuit from the obtained output sequence.

In SCMs, signal probabilities are carried in the random binary bit streams, while signal dependencies are preserved in the stochastic logic network. Hence, the reliability found using the SCM approach is accurate. However, the precision of the obtained result is limited due to the inevitable random fluctuations in stochastic computation. The efficiency and accuracy of the SCM approach are discussed next.

5 Efficiency

In this section, the efficiency of the proposed SCM approach is analyzed. Efficiency is affected by various features as related to the circuit structure and the simulation process itself.

5.1 Signal Correlation

Due to signal correlation, an accurate analytical approach requires a computational complexity that increases exponentially with the number of reconvergent fanouts. This makes it difficult, if not impossible, to evaluate the reliability of large circuits. A significant feature of the SCM approach is that it efficiently handles signal correlation introduced by reconvergent fanouts, as discussed in Section 3, thus significantly reducing the computational complexity. Although not explicitly presented, signal correlation due to feedbacks in sequential circuits can similarly be handled; hence, the proposed SCM approach is also applicable to the analysis of sequential circuits.

5.2 Simulation Efficiency

5.2.1 Pseudo-random number generation

A simulative approach such as Monte Carlo simulation (MCS) is based on fault injections and random pattern simulations. In MCS, pseudo-random numbers are independently generated for each gate and input to produce a random pattern that mimics the behavior of an erroneous circuit. Typically, a large number of random patterns must be simulated; the circuit reliability is then calculated based on the resulting statistical outcome. The efficiency of MCS is therefore limited by two factors: 1) the required number of pseudo-random number generations in a single simulation run and 2) the required number of simulation runs to obtain a stable output. As an example, consider a circuit with N_{in} primary inputs and N_g gates; if a total number of M simulations are required to achieve convergence in the statistical output, then the required number of pseudo-random number generations is given by $N_{MCS} = (N_g + N_{in}) \times M$.

In contrast, the proposed SCM approach *explicitly* and *efficiently* implements analytical algorithms as well as taking advantage of statistical randomness in simulation. As discussed in Section 3, however, it is sufficient to use non-Bernoulli sequences with fixed numbers of 1's (and 0's) as initial random binary streams rather than the Bernoulli sequences usually required in a random simulation approach. For a sequence length of N bits and a gate error rate ε, only $N\varepsilon$ pseudo-random numbers must be generated (for the positions of 1's) in the representation of the error probability ε. An equivalent MCS would require the generation of N pseudo-random numbers. As ε usually has a (very) small value, the number of required generations of pseudo-random numbers is significantly reduced in the SCM approach.

5.2.2 Convergence rate

The use of non-Bernoulli sequences as random permutations of fixed numbers of 1's (and 0's) does not only produce accurate evaluation results (as discussed in Section 3), but also leads to a smaller variance and thus, to a faster convergence of the output value compared to the conventional use of Bernoulli sequences in MCS.

Fig. 7 shows a comparison of the simulation results for C432 using SCM with the non-Bernoulli sequences and MCS (as equivalent to the use of Bernoulli sequences). While both approaches produce the same evaluation result (given by the mean value of 0.7969 for SCM and 0.7970 for MCS), there is a clear difference in the output distributions of the evaluation results: each approach leads to a Gaussian distribution with a different standard deviation. Specifically, the SCM approach results in a standard deviation of 0.0054 (as shown in Fig. 7(a)), while for MCS, 0.0128 (as shown in Fig. 7(b)). This indicates that for the same number of simulations, it is likely

to have a more accurate evaluation result by using SCM compared to MCS. Equivalently, a shorter sequence length in SCM is required to achieve an evaluation accuracy by utilizing an equivalent number of MCS runs. Hence, the SCM approach is more efficient because it reduces both the number of required pseudo-random number generations and the total number of required simulation runs (as equivalent to the required sequence length). Therefore, it requires a significantly shorter runtime compared to MCS.

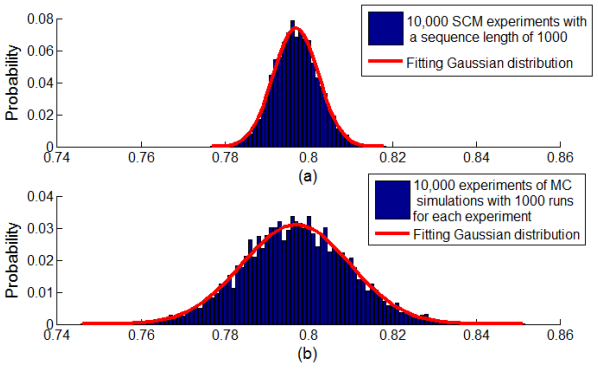


Fig. 7. Comparisons of the simulation results for C432 with an error rate of 0.005. (a) 10,000 SCM experiments with a sequence length of 1000 bits; (b) 10,000 Monte Carlo simulations with a number of 1000 runs in each simulation.

6 ACCURACY

The accuracy of the SCM is analyzed in terms of various precision errors as well as random fluctuations.

6.1 Error Analysis

6.1.1 Quantization Error

A *quantization error* is due to the loss of precision in the process of converting a probability into the representation of a stochastic binary sequence [31]. It is determined by the length of the sequence, i.e., the number of bits used in a stochastic sequence. For a sequence length of N bits, a real value is rounded to the nearest value that can be represented by this sequence (in units of $1/N$). Therefore, the maximum error due to quantization (by using an appropriate rounding scheme) is limited by $1/2N$, i.e.,

$$e_q \le \frac{1}{2N}. \tag{43}$$

In the SCM approach, the stochastic representation of inputs and the gate error rate can both lead to quantization errors. Some of these errors are propagated through the circuits, while others are attenuated or masked due to the logic operations in the evaluation process. The quantization error at the output can be estimated as follows. Initially assume that the circuit reliability represented by the output sequence is a function of the input and gate error probabilities, i.e.,

$$R = F_q(p_1, p_2, \dots p_{N_{in}}, \varepsilon_1, \varepsilon_2, \dots, \varepsilon_{N_g}), \tag{44}$$

where p_i is the input probability of the ith input, ε_j is the error rate of the jth gate, and N_{in} and N_g are the input and gate numbers, respectively.

The quantization error in the output reliability is determined by the collective effect of the inputs and the gate error rates; therefore it is given by a first-order approximation as follows:

$$E_q \approx \sum_{i=1}^{N_{in}} \frac{\partial F_q}{\partial p_i} \cdot e_{q,p_i} + \sum_{j=1}^{N_g} \frac{\partial F_q}{\partial \varepsilon_j} \cdot e_{q,\varepsilon_j}, \tag{45}$$

where e_{q,p_i} is the quantization error in the ith input and e_{q,ε_j} is the quantization error in the error rate of the jth gate. Furthermore, assume that each quantization error has a maximum negative effect on the evaluation result; so as per (45),

$$E_q \le \frac{1}{2N}(N_{in} + N_g). \tag{46}$$

(46) gives an upper bound of the quantization error as dependent on the numbers of inputs and gates, i.e. the bound is larger for a larger circuit. However, this bound can only be reached by circuits in which an output is dependent on every input and internal signal. Therefore, this estimate of the quantization error is pessimistic for most practical applications. In practice, however, most quantization errors can be avoided by using an appropriate sequence length if the input probability and gate error rate can be represented by rational numbers. For example, for perfect inputs and a gate error rate of 0.05, a sequence length of 1000 bits suffices to avoid the generation of quantization errors.

Note that in this paper, the discussion of quantization errors is limited to those due to the stochastic representation of the initial inputs and gate error rates; there are no further sources of quantization errors. The errors that occur during the evaluation of a circuit and are also due to the effect of quantization, are referred to as *resolution errors* and discussed next.

6.1.2 Resolution Error

The sequence length is an important parameter because it determines the resolution of the computation result. Due to a limited resolution, a resolution error occurs in the result of a stochastic computation. As an example for a sequence length of 10 bits, the resolution is 0.1, i.e., any probability with a precision higher than that of 0.1 cannot be represented. A computational error due to a limited resolution is illustrated in Fig. 8. Fig. 8 (a) shows a scenario in which there are two independent inputs: X_1=0.2 and X_2=1. As $Z=X_1X_2$, it can be found from the output binary stream that Z=0.2. This is an accurate result. Fig. 8 (b) shows a scenario in which X_1=0.8 and X_2=0.8. The correct output is therefore 0.64. However, due to the limited resolution, it is found that Z=0.6 by this computation.

As a further example, a sequence of 1000 bits can be used to give a resolution of 10^{-3}. A computational result is rounded to its nearest available representation, so the maximum error due to this resolution is 5×10^{-4}. This indicates that the result obtained in a single computation can have a precision error of up to $1/2N$ for a sequence length of N bits, i.e.,

$$e_r \le \frac{1}{2N}, \tag{47}$$

where e_r is the resolution error in a single computation.

For a given input vector and a gate error rate, the resolution error incurred at each evaluation step (i.e., for each logic gate) contributes to the precision error in the evaluation result. The error due to this precision loss in the

computation at the *i*th logic gate is given by:

$$E_{r,i} = \frac{\partial R}{\partial r_i} \cdot e_{r,i}\,, \tag{48}$$

where R is the reliability function, r_i is the evaluation result at the *i*th gate, and $e_{r,i}$ is the incurred resolution error at this evaluation step.

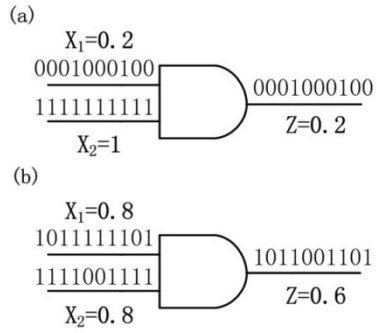


Fig. 8. Resolutions in stochastic computation: (a) The desired output; (b) An imprecise output due to a limited resolution.

This process must also account for the stochastic computational structures used in the SCM. In Fig. 5, an unreliable gate is modeled by two equivalent fault-free gates for each fault model (as an inverter does not introduce a resolution error). So, the total precision loss in the final evaluation result is given by a first-order estimate as:

$$E_r \approx \sum_{i=1}^{2N_g} \frac{\partial R}{\partial r_i} \cdot e_{r,i}. \tag{49}$$

Since each single bit flip in the output of the *i*th gate may or may not cause a change in the final output sequence, then

$$\left|\frac{\partial R}{\partial r_i}\right| \leq 1. \tag{50}$$

and thus

$$E_r \leq \frac{1}{N} N_g. \tag{51}$$

Again, (51) gives an upper bound and is often pessimistic when used as an error estimate. Therefore, a stochastic sequence with a length that satisfies the maximum resolution error requirement of (51), is usually sufficient for the evaluation of a circuit of certain size (as determined by the number of gates N_g).

For example, if quantization errors can be avoided and a maximum precision loss of 0.01 is acceptable due to resolution errors, the minimum sequence length can be determined by (51), i.e.,

$$N \geq 100 N_g. \tag{52}$$

A sequence length of one million bits would meet this precision requirement for a circuit with thousands of gates. In practice, a shorter sequence length can often produce a result at a desired accuracy. This is shown by the simulation results in Section 7.

6.1.3 Random Permutation

Errors can also be caused by the *random permutation of bits* in a sequence. Fig. 9 illustrates an example of a randomized permutation; the logic operation in Fig. 9 (a) gives the desired output value, while the operation in Fig. 9 (b) gives an output that is considered to be in error. In general, longer sequences tend to be better randomized; however, random permutations are probabilistic in nature and therefore, they do not always provide the desired results. The error due to a random permutation occurs in each stochastic computation and is considered as "noise." It also contributes to the stochastic fluctuation in the evaluation result of a probabilistic logic network.

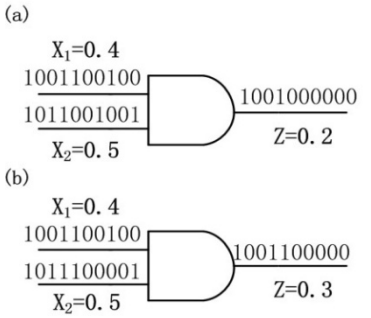


Fig. 9. Random permutations in stochastic computation: (a) The desired permutation; (b) A permutation resulting in an error.

6.2 Random Fluctuations

A *random fluctuation* is a result of the aforementioned precision errors; it is an inherent feature of stochastic computation. For example, simulation of C17 has shown that the result of each experiment fluctuates around the expected mean value [15]; the output follows approximately a Gaussian distribution, similar to the one shown in Fig. 7(a) for C432. This fluctuation can then be analyzed quantitatively by investigating the mean and variance of the output distribution. A detailed discussion of different relations (such as between sequence length and accuracy, error rate and circuit size) is presented in the next section.

7 Simulation Results

In this section, the proposed SCM approach is compared to the following reliability evaluation techniques: the accurate PGM algorithm, the PTM approach, the signature-based SER analyzer and the Monte Carlo simulation. Simulations have been performed on a computer with a 2.66-GHz Pentium microprocessor and a 2 GB memory.

7.1 SCM vs. PGM and PTM

To validate the model, the proposed SCM approach is initially compared with the accurate PGM and PTM approaches on several small circuits. While individual output reliabilities are reported in [15], only joint output reliabilities are reported in this section. Table 1 shows the evaluated circuit reliability and the relative error for a von Neumann fault with ε=0.05. The *relative error* is defined as the ratio of the difference between an approximate value and the accurate value over the accurate value. A maximum of 1000 randomly-selected inputs are used in the simulation. As shown in Table 1, the SCM approach yields highly accurate results; the maximum relative error is around 0.2% by using a sequence length of 1000 bits. The results for stuck-at faults are shown in Table S1 in the supplementary material, with a maximum relative error around 0.1% for both stuck-at-1 and stuck-at-0 faults.

Although the runtime of the SCM approach appears mostly longer than that of the PGM and PTM approaches, the longer time required for identifying and decomposing reconvergent fanouts in PGM (required in excess of several hours), as well as the time required for analyzing the circuit structure in PTM, is not included in the runtime reported in the tables. For the SCM approach, accuracy can further be improved by increasing the sequence length. As the stochastic simulations are independent processes, it would be possible to further reduce the runtime by using a parallelized procedure.

7.2 SCM vs. Signature-based approach

The SCM approach is further compared with the signature-based approach [25] for SER analysis of the LGSynth91 benchmarks [41]. For a gate affected by soft errors, the SER is defined as its output error probability while for a circuit, the SER is defined as the complement of its joint output reliability. The simulation results are shown in Table 2. The signature-based SER analyzer finds the testability of a circuit by using bit-parallel simulation for SER calculation; this process is similar to the stochastic simulation as used in the SCM approach. Only temporary single stuck-at (TSA) faults are analyzed in [25].

In most previous studies, the SER is expressed in Failures in Time (or FIT, usually in the number of failures in 10^9 hours) [16-19], [25], [26]. Since the effect of multiple errors is considered in the SCM approach, probabilities are instead used in this section as SER values. In Table 2, the stuck-at faults are assumed to be uniformly distributed with a gate SER $\varepsilon = 10^{-6}$ and $\varepsilon = 10^{-2}$. Although an SER is generally considered small, a rather large value (10^{-2}) is used in the simulation to show the difference between the approaches. The overall circuit SER is given by the sum of the stuck-at-0 and stuck-at-1 SER values. Table 2 shows that the two approaches produce very close results for most circuits when $\varepsilon = 10^{-6}$; however, the relative differences of the SER increases at $\varepsilon = 10^{-2}$. A signature-based approach considers the sensitivity of each gate separately and sums over the resulting circuit SER for each gate. This is different for the SCM approach that accounts for multiple error occurrences as well as their correlation by considering the joint effects of multiple errors in the evaluation of a circuit.

These two approaches model a similar error scenario as long as the node (or gate) SER is small. In this case, the probability that more than one error occur is even smaller and thus negligible. This is confirmed by the simulation results for $\varepsilon = 10^{-6}$. As the circuit size or the SER of each node increases, however, the probability of independent multiple-error occurrence increases. This may result in large discrepancies using these two evaluation methods, as indicated in the reported simulation results for $\varepsilon = 10^{-2}$.

In the scenario of multiple dependent transient faults caused by a single radiation upset [22], the SCM approach can readily be adapted to model the correlated multiple errors by using dependent stochastic sequences. Although the SCM approach needed a longer runtime due to its use of the redundant stochastic sequences, the signature-based approach required a greater effort in execution due to the complicated programming and algorithms involved. The SER analysis by both approaches primarily considers logic masking; however the results obtained can be enhanced by modeling technology-dependent factors such as the electrical and timing effects on SER.

7.3 SCM vs. Monte Carlo simulation

Finally, large benchmarks of ISCAS'85 are simulated to compare the efficiency of the SCM approach with Monte Carlo simulation (MCS). As more computational power is required, these simulations were performed on a computer with a 3.10 GHz i3-2100 microprocessor and a 6 GB memory.

For MCS (equivalent to the use of Bernoulli sequences), the number of simulation runs required at a given CV can be estimated by considering (9), (10) and (35), i.e.,

$$N_{mcs} = \frac{1-R}{R} \cdot \frac{1}{(CV)^2}, \tag{53}$$

where R is the output reliability of a circuit. Since reliability usually decreases with circuit size (at a constant component failure rate), (53) indicates that the number of required MCS runs may increase with circuit size at a given accuracy. Next, simulations are performed at a specified accuracy to find the least required sequence length for SCM, as well as the least number of runs for MCS. The results at a CV of 0.001 are shown in Table 3 for $\varepsilon = 10^{-3}$ and in Table S2 in the supplementary material for $\varepsilon = 5\times10^{-4}$ respectively. The same randomly-selected input vector is used for an accurate and fair comparison. Twenty experiments are run for each circuit to obtain the reported statistics. It can be seen that both methods produce similar results; however, the SCM requires a significantly shorter runtime (by several orders of magnitude) compared to MCS.

As the number of MCS runs is a function of circuit reliability, then the relationship between the sequence length of SCM (as well as the number of MCS runs) and circuit reliability is plotted in Fig. 10(a) and compared to the analytical results provided by (53). It can be seen that the MCS results are in good agreements with those found by (53). Also, the sequence length of SCM is significantly smaller than the number of MCS runs; this is consistent with the analysis in Section 3. The runtimes of these two approaches are compared in Fig. 10(b); a significantly longer runtime (by several orders of magnitude) is encountered for MCS compared to SCM at the desired evaluation accuracy.

Table 1. Accuracy comparison of the SCM, PGM and PTM approaches for the von Neumann fault.

Circuit	*Characteristics*			*SCM* $\varepsilon = 0.05$ $N = 1000$ *bits*			*Accurate PGM* $\varepsilon = 0.05$		*PTM* $\varepsilon = 0.05$	
	Gates	*PIs*	*POs*	*R*	*Time (s)*	*Rel. error*	*R*	*Time (s)*	*R*	*Time (s)*
C17	*6*	*5*	*2*	0.7830	0.06	0.11%	0.7839	0.002	0.7839	0.001
Majority	*10*	*5*	*1*	0.8634	0.15	0.13%	0.8623	0.002	0.8623	0.05
Full adder (Majority)	*8*	*3*	*2*	0.7896	0.02	0.1%	0.7904	0.0007	0.7904	0.18
Full adder (XOR/NAND)	*6*	*3*	*2*	0.8016	0.01	0.2%	0.8000	0.001	0.8000	0.001
Full adder (NAND)	*12*	*3*	*2*	0.6547	0.03	0.21%	0.6533	0.008	0.6533	0.002
Comparator	*4*	*2*	*3*	0.8265	0.008	0.01%	0.8264	0.006	0.8264	0.0009
Decoder2	*6*	*2*	*4*	0.7385	0.01	0.09%	0.7392	0.03	0.7392	0.009
MUX4	*7*	*6*	*1*	0.8228	0.14	0.09%	0.8221	0.001	0.8221	0.52

Table 2. SER analysis using the SCM and signature-based approaches.

Circuit	*No. Gates*	*Signature-based SER analyzer* $Signature\ length = 10,000$		*SCM approach* $Sequence\ length = 1,000,000$, $10,000$ *input vectors*			
		SER $\varepsilon = 10^{-6}$	***SER*** $\varepsilon = 10^{-2}$	***SER*** $\varepsilon = 10^{-6}$	***Relative difference***	***SER*** $\varepsilon = 10^{-2}$	***Relative difference***
majority	10	3.4503×10^{-6}	3.4473×10^{-2}	3.4374×10^{-6}	0.38%	3.3185×10^{-2}	3.9%
parity	15	1.5×10^{-5}	1.5×10^{-1}	1.4976×10^{-5}	0.16%	1.3977×10^{-1}	7.3%
decod	22	1.9884×10^{-5}	1.9873×10^{-1}	2.0012×10^{-5}	0.63%	1.8642×10^{-1}	6.6%
x2	38	1.8547×10^{-5}	1.8519×10^{-1}	1.8463×10^{-5}	0.45%	1.7563×10^{-1}	5.4%
pm1	41	1.8923×10^{-5}	1.8871×10^{-1}	1.8897×10^{-5}	0.14%	1.7965×10^{-1}	5.0%
cu	43	1.6577×10^{-5}	1.6578×10^{-1}	1.6631×10^{-5}	0.32%	1.5816×10^{-1}	4.8%
z4ml	45	2.6005×10^{-5}	2.6010×10^{-1}	2.6105×10^{-5}	0.38%	2.4208×10^{-1}	7.4%
mux	50	6.6941×10^{-6}	6.7689×10^{-2}	6.7369×10^{-6}	0.64%	6.4965×10^{-2}	4.0%
pcle	61	2.8963×10^{-5}	2.8896×10^{-1}	2.9061×10^{-5}	0.34%	2.6632×10^{-1}	8.5%

For SCMs, an empirical function based on (53) is used to fit the experimental results for $\varepsilon = 10^{-3}$ (in Fig. 10(a)); this is given by

$$N_{mcs} = (5.15R^3 - 9.52R^2 + 4.65R - 0.19)\left(\frac{1-R}{R}\right)\frac{1}{(CV)^2}, \quad (54)$$

where R is the circuit reliability and CV gives the desired accuracy. Then the simulation results for $\boldsymbol{\varepsilon = 5\times10^{-4}}$ are used for validation of (54). As shown in Fig. 10(a), (54) fits the data very well. This indicates that the application of (54) is not limited to a specific gate error rate, ε. Hence, the sequence length to achieve a specific accuracy for an arbitrary circuit using SCMs can be obtained as follows. Initially, a short sequence can be used to obtain an estimate of the circuit reliability; then, an empirical formula (such as (54)) can be used to predict the minimum sequence length for evaluating the reliability. As the predictive function is not unique, the required sequence length may deviate from the predicted value. Therefore, the evaluation results must be carefully analyzed and a different sequence length may be eventually needed to ensure the desired accuracy.

8 Discussion

8.1 Sequence length vs. Circuit reliability

It has been observed that a longer sequence is required for evaluating a circuit with lower reliability to achieve a specific accuracy. This is primarily due to the use of the coefficient of variation (CV) as metric of accuracy, because CV is determined by the standard deviation and the mean value of the obtained circuit reliability. The reliability of a circuit usually decreases by increasing its size (i.e., the number of gates) when a constant gate error rate is assumed [36], so it is generally believed that a longer sequence is required for evaluating a larger circuit. In fact, this is a result of the decreasing reliability (given by μ) and increasing random fluctuations due to various analysis errors (as indicated by σ). However, a larger circuit may not necessarily have a reduced reliability and therefore, it may not require a larger sequence length in SCM. Hence, reliability appears to be a more relevant factor when determining the minimum length of the required sequence, because it considers many other factors, such as the logic function, the circuit topology and the number of inputs and outputs of a circuit.

Another factor that determines the minimum sequence length, is the gate error rate, ε. To allow for minimum representation, at least $1/\varepsilon$ bits are required in the sequences. Albeit not included (due to space limitation), the simulation results for $\varepsilon = 10^{-4}$ show that this minimum requirement in sequence length has already met the accuracy requirement for most of the benchmark circuits. However, longer sequences (such as those of k/ε bits with $k > 1$) may be desired for a better resolution and/or proper statistical operation. When the use of this sequence length does not produce satisfactory results, the procedure outlined in the previous section can be

utilized to find the minimum sequence length at a given accuracy. Nevertheless, it remains a challenge to efficiently determine the required minimum sequence length when no prior knowledge of the circuit is available. It is also an open question if further theoretical results can be obtained on this respect.

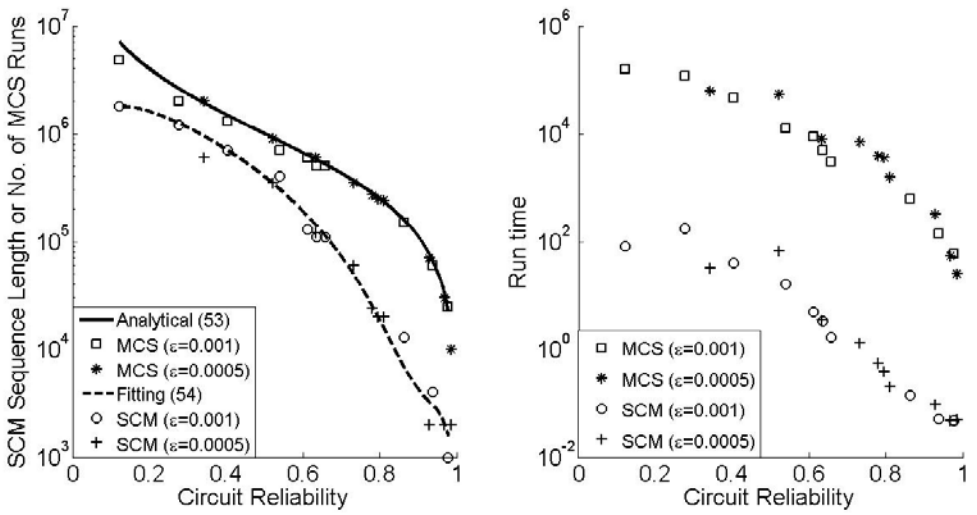
8.2 Electrical and latching-window masking

So far we have only considered logic masking in the modeling of SERs. In [22], an analysis framework using BDDs and ADDs has been developed. The attenuation of electrical signals is computed using a glitch propagation model [42], in which the amplitude and duration of a glitch at a gate's output are determined for several different scenarios related to the input parameters of the glitch. The probability that the propagated glitch is latched by a flip-flop, is then determined by the expected value of the duration and the timing information of the glitch.

Electrical and latching-window masking can be included in the proposed stochastic approach by integrating the attenuation and latching models developed in [22, 42] with the current SCM framework. For instance, the initial glitch duration and its distribution can be represented by a stochastic sequence of multiple values based on the discrete sampling over the full range of the possible glitch length. The attenuation process can then be implemented using stochastic logic designs [29] that selectively compute the output amplitude according to different input parameters. As logic masking is also inherently accounted for, the SCM approach can provide a *unified treatment* of these masking features. Due to space limitation, this topic is not further covered in this paper; it will be investigated in future work.

Table 3. Simulation results of ISCAS'85 benchmarks by SCM and MCS at a fixed CV of 0.001 ($\varepsilon = 10^{-3}$).

Circuit	Characteristics			MCS ($\varepsilon = 10^{-3}$)			SCM ($\varepsilon = 10^{-3}$)		
	gates	inputs	outputs	Number of Simulations	Reliability	Runtime (s)	Sequence length (bits)	Reliability	Runtime (s)
C432	250	36	7	25,000	0.9773	58.592	1,000	0.9777	0.048695
C499	202	41	32	60,000	0.9372	142.28	4,000	0.9375	0.051646
C880	383	60	26	150,000	0.8622	623.48	13,000	0.8624	0.14022
C1355	546	41	32	500,000	0.6580	3019.5	110,000	0.6582	1.6674
C1908	880	33	25	500,000	0.6358	4882.1	110,000	0.6355	3.3146
C2670	1193	157	64	600,000	0.6120	8904.6	130,000	0.6118	4.9834
C3540	1669	50	22	700,000	0.5398	12582	400,000	0.5396	16.409
C5315	2307	178	123	1,300,000	0.4039	45500	700,000	0.4034	40.604
C6288	2416	32	32	4,800,000	0.1207	158146	1,800,000	0.1207	81.382
C7552	3512	207	108	2,000,000	0.2777	115724	1,200,000	0.2772	177.29



(a) **(b)**

Fig. 10. Comparisons of SCM and MCS for different gate error rate, ε, at a fixed CV of 0.001: (a) Required sequence length in SCM and number of MCS runs vs. circuit reliability, fitted using (53) and (54); (b) Required runtime for the two methods. In (a), (54) is fitted using the data for $\varepsilon = 10^{-3}$ and validated by the data for $\varepsilon = 5\times10^{-4}$.

9 Conclusion

Advances of VLSI circuits and systems into the nanometric regimes require accurate and efficient reliability evaluation techniques. In this paper, a novel stochastic approach is proposed as a computational framework for the reliability evaluation of logic circuits. Using stochastic computational models (SCMs), this approach

accurately evaluates the reliability of a circuit with a precision limited by the inherent randomness of the binary bit streams used in stochastic computation. Compared to accurate analytical approaches found in the technical literature, the proposed SCM approach *efficiently handles* signal correlations introduced by reconvergent fanouts and thus *significantly reduces* the computational complexity.

Compared to the simple simulation of random vectors, the proposed approach has the following distinguishing features: 1) *Versatility*. The SCM is flexible due to its pronounced arithmetic nature. 2) *Generality*. The SCM approach has been developed as a general computational framework to efficiently implement analytical algorithms. 3) *Scalability*. Compared to Monte Carlo simulation, the SCM approach is more efficient because it benefits from the use of non-Bernoulli sequences thus requiring a reduced number of pseudo-random number generations. At a specified evaluation accuracy, a procedure based on an empirical (curve-fitting) formula has been proposed for finding the required minimum sequence length in SCM for evaluating a circuit.

The runtime can further be reduced through a parallelization of the stochastic sequences, so the proposed approach is potentially useful in the design and test of reliable VLSI circuits and systems. It is shown that it is potentially applicable to account for electrical and latching-window masking in the evaluation of SERs; this topic will be pursued in future investigation.

ACKNOWLEDGEMENT

This work was supported in part by the Natural Sciences and Engineering Research Council (NSERC) of Canada in a Discovery Grant.

REFERENCES

[1] S. Borkar, "Designing Reliable Systems from Unreliable Components: The Challenges of Transistor Variability and Degradation," in IEEE Micro, vol. 25, no. 6, pp. 10-16, Nov. 2005.

[2] K. P. Parker and E. J. McCluskey, "Probabilistic Treatment of General Combinational Networks", IEEE Trans. on Computers, vol. C, no. 24, pp. 668–670, June 1975.

[3] I. Bahar, J. L. Mundy and J. Chen, "A probabilistic-based design methodology for nanoscale computation," in Proc. Int. Conf. Comput.-Aided Des., 2003, pp. 480–486.

[4] J. Han and P. Jonker, "A defect- and fault-tolerant architecture for nanocomputers," Nanotechnology, vol. 14, no. 2, pp. 224–230, 2003.

[5] S. Krishnaswamy, G. F. Viamontes, I. L. Markov, and J. P. Hayes, "Probabilistic transfer matrices in symbolic reliability analysis of logic circuits," ACM Trans. Des. Autom. Electron. Syst., vol. 13, no. 1, pp. 1–35, 2008.

[6] C. Yu and J. P. Hayes, "Scalable and Accurate Estimation of Probabilistic Behavior in Sequential Circuits," Proc. VTS, pp. 165-170, 2010.

[7] J. Han, E.R. Taylor, J. Gao and J.A.B. Fortes, "Faults, Error Bounds and Reliability of Nanoelectronic Circuits," in Proc. IEEE ASAP 2005, pp. 247-253.

[8] E.R. Taylor, J. Han, J.A.B. Fortes, "Towards Accurate and Efficient Reliability Modeling of Nanoelectronic Circuits," in Proc. IEEE-NANO 2006, Vol. 1, pp. 395-398.

[9] J. Han, H. Chen, E. Boykin, J. Fortes, "Reliability evaluation of logic circuits using probabilistic gate models," Microelectronics Reliability, vol. 51, no. 2, 2011, pp. 468-476.

[10] T. Rejimon and S. Bhanja, "Scalable probabilistic computing models using Bayesian networks," in Proc. Int. Midwest Symp. Circuits Syst., 2005, pp. 712–715.

[11] A. Abdollahi, "Probabilistic Decision Diagrams for Exact Probabilistic Analysis," Proc. Int'l Conference on Computer Aided Design, 2007.

[12] N. Mohyuddin, E. Pakbaznia, M. Pedram, "Probabilistic Error Propagation in Logic Circuits Using the Boolean Difference Calculus," in IEEE Intl. Conf. on Comp. Des., Lake Tahoe, CA, USA, pp. 7-13 (2008)

[13] S. Sivaswamy, K. Bazargan, M. Riedel, "Estimation and Optimization of Reliability of Noisy Digital Circuits," in International Symposium on Quality Electronic Design, San Jose, CA, USA, pp. 213-219 (2009)

[14] M. R. Choudhury and K. Mohanram, "Reliability analysis of logic circuits," IEEE TCAD, vol. 28, no. 3, pp. 392–405, March 2009.

[15] H. Chen, J. Han, "Stochastic Computational Models for Accurate Reliability Evaluation of Logic Circuits", Proc. Great Lakes Symp. VLSI (GLVLSI), Providence, RI, USA, pp. 61-66 (2010)

[16] P. Shivakumar et al., "Modeling the Effect of Technology Trends on the Soft Error Rate of Combinatorial Logic," Proc. Int"l Conf. Dependable Systems and Networks, IEEE CS Press, 2002, pp. 389-398.

[17] M. Zhang and N. R. Shanbhag, "A soft error rate analysis (SERA) methodology," in Proc. ICCAD, 2004, pp. 111–118.

[18] B. Zhang, W. S. Wang, and M. Orshansky, "FASER: Fast analysis of soft error susceptibility for cell-based designs," in Proc. ISQED, 2006, pp. 755–760.

[19] R. Rao, K. Chopra, D. Blaauw, and D. Sylvester, "An efficient static algorithm for computing the soft error rates of combinational circuits," in Proc. DATE, 2006, pp. 164–169.

[20] N. Miskov-Zivanov and D. Marculescu, "MARS-C: Modeling and reduction of soft errors in combinational circuits," in Proc. DAC, 2006, pp. 767–772.

[21] N. Miskov-Zivanov and D. Marculescu, "Soft error rate analysis for sequential circuits," in Proc. DATE, 2007, pp. 1436–1441.

[22] N. Miskov-Zivanov, D. Marculescu, "Multiple Transient Faults in Combinational and Sequential Circuits: A Systematic Approach," IEEE TCAD, vol. 29, no. 10, pp. 1614–1627, 2010.

[23] J. P. Hayes, I. Polian, and B. Becker, "An analysis framework for transient-error tolerance," In VLSI Test Symp., pages 249–255, 2007.

[24] I. Polian, J.P. Hayes, S.M. Reddy and B. Becker, "Modeling and mitigating transient errors in logic circuits," IEEE Trans. on Dependable & Secure Computing, 2010.

[25] S. Krishnaswamy, S. M. Plaza, I. L. Markov, and J. P. Hayes, "Signature-based SER analysis and design of logic circuits," IEEE Trans. Comput.-Aided Design Integr. Circuits, vol. 28, no. 1, pp. 74–86, Jan. 2009.

[26] Y-H Kuo, H-I Peng, Wen, C.H.-P., "Accurate statistical soft error rate (SSER) analysis using a quasi-Monte Carlo framework with quality cell models," in Proc. ISQED, 2010, pp. 831–838.

[27] W. J. Poppelbaump, C. Afuso and J. W. Esch, "Stochastic Computing Elements and Systems", Proceedings of the Fall Joint Computing Conf. 1967, pp. 631-644.

[28] B. R. Gaines, "Stochastic Computing Systems," Advances in Information Systems Science, Vol. 2, pp. 37-172, 1969.

[29] B. Brown and H. Card, "Stochastic neural computation I: Computational elements," IEEE Tran. Computers, vol. 50, pp. 891–905, Sept. 2001.

[30] X. Li, W.K. Qian, M. Riedel, K. Bazargan, D. Lilja, "A Reconfigurable Stochastic Architecture for Highly Reliable Computing," Proc. Great Lakes Symp. VLSI (GLVLSI), Boston, MA, USA, pp. 315-320, 2009

[31] W. Qian, X. Li, M. D. Riedel, K. Bazargan, and D. J. Lilja, "An architec-

ture for fault-tolerant computation with stochastic logic," IEEE Tran. Computers, vol. 60, pp. 93–105, Jan. 2011.

[32] N. Shanbhag, R. Abdallah, R. Kumar, and D. Jones, "Stochastic computation," in Proc. ACM/IEEE DAC, 2010.

[33] J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," Automata Studies, Shannon C.E. & McCarthy J., eds., Princeton University Press, pp. 43-98, 1956.

[34] J. Han and P. Jonker, "A system architecture solution for unreliable nanoelectronic devices," IEEE Trans. on Nanotechnology, vol. 1, no. 4, pp. 201–208, December 2002.

[35] S. Roy and V. Beiu, "Majority multiplexing–economical redundant fault-tolerant design for nano architectures." IEEE Trans. On Nano, 4, 4, 2005.

[36] J. Han, J. Gao, Y. Qi, P. Jonker, J.A.B. Fortes. "Toward Hardware- Redundant, Fault-Tolerant Logic for Nanoelectronics," IEEE Design and Test of Computers, July/August 2005, vol. 22, no. 4, 328-339.

[37] G. Roelke, R. Baldwin, D. Bulutoglu, "Analytical Models for the Performance of von Neumann Multiplexing," IEEE Trans. on Nanotechnology, vol. 6, no. 1, pp. 75–89, 2007.

[38] S. S. Tehrani, S. Mannor, and W. J. Gross, "Survey of stochastic computation on factor graphs," in Proc. 37th IEEE Int. Symp. Multiple-Valued Logic, Oslo, Norway, May 2007, pp. 54–59.

[39] C. Winstead, V. C. Gaudet, A. Rapley, and C. B. Schlegel, "Stochastic iterative decoders," In Proc. Intl Symp. Info. Theory, pp. 1116-1120, 2005.

[40] Jie Han, "Fault-Tolerant Architectures for Nanoelectronic and Quantum Devices", Universal Press, Veenendaal, The Netherlands, 2004. A Ph.D. dissertation of the Delft University of Technology, 1-135. ISBN: 90-9018888-6

[41] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Muigai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "SIS: A System for Sequential Circuit Synthesis." Technical Report UCB/ERL M92/41, Electronics Research Lab, Univ. of California, Berkeley, CA 94720, May 1992.

[42] M. Omana, G. Papasso, D. Rossi, and C. Metra. "A Model for Transient Fault Propagation in Combinatorial Logic." In Proc. of the 9th IEEE International On-Line Testing Symposium, IOLTS'03, pp. 111-115, July 2003.

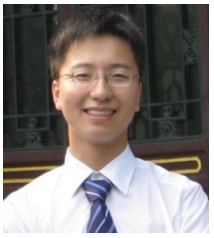
Jie Han (S'02–M'05) received the B.Sc. degree in electronic engineering from Tsinghua University, Beijing, China, in 1999 and the Ph.D. degree from Delft University of Technology, The Netherlands, in 2004. He is currently an assistant professor in the Department of Electrical and Computer Engineering at the University of Alberta, Edmonton, AB, Canada.

His research interests include reliability and fault tolerance, nanoelectronic circuits and systems, novel computational models for nanoscale and biological applications. Dr. Han served as a Technical Program Chair in IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT 2012) and as a Technical Program Committee Member in several other international symposia and conferences.

Hao Chen received the B.Sc. degree in electronic engineering from Zhejiang University, Hangzhou, China, in 2009 and the M.Sc. degree from the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada, in 2012.

His research interests include stochastic modeling of nanoscale devices and circuits, and reliability-aware design of nanoscale circuits and systems.

.



Jinghang Liang received the B.Eng. degree in Institute of Microelectronics from Tsinghua University, China, in 2009 and the M.Sc. degree from the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada, in 2012.

His research interests include fault-tolerant circuits and systems, RF circuit design and genetic network models.

Peican Zhu received the B.S. degree in 2008 and the M.Sc. degree in 2011, both from the Northwestern Polytechnical University (NWPU), Xi'an, Shaanxi, China. He is currently working towards the Ph.D. degree in the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada.

His current research interests include stochastic computing models for circuit and system reliability analysis, biological system modeling and analysis.

.

Zhixi Yang received the B.S. degree from National University of Defense Technology (NUDT), Changsha, Hunan, China, in 2011. He is currently working towards the Ph.D. degree in the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada. His current research interests include stochastic timing analysis and soft error models for analyzing circuit reliability.

Fabrizio Lombardi (M'81–SM'02-F'09) graduated in 1977 from the University of Essex (UK) with a B.Sc. (Hons.) in Electronic Engineering. In 1977 he joined the Microwave Research Unit at University College London, where he received the Master in Microwaves and Modern Optics (1978), the Diploma in Microwave Engineering (1978) and the Ph.D. from the University of London (1982).

He is currently the holder of the International Test Conference (ITC) Endowed Chair Professorship at Northeastern University, Boston. His research interests are bio-inspired and nano manufacturing/computing, VLSI design, testing, and fault/defect tolerance of digital systems. He has extensively published in these areas and coauthored/edited seven books.

A Stochastic Computational Approach for Accurate and Efficient Reliability Evaluation

Jie Han, Hao Chen, Jinghang Liang, Peican Zhu, Zhixi Yang and Fabrizio Lombardi

Supplementary Material

Table S1. Accuracy comparison of the SCM, PGM and PTM approaches for stuck-at faults.

Circuit	*Stuck-at-1 (TSA-1) Fault*							*Stuck-at-0 (TSA-0) Fault*						
	SCM $\varepsilon = 0.05$ $N = 1000$			*Accurate PGM* $\varepsilon = 0.05$		*PTM* $\varepsilon = 0.05$		*SCM* $\varepsilon = 0.05$ $N = 1,000$			*Accurate PGM* $\varepsilon = 0.05$		*PTM* $\varepsilon = 0.05$	
	R	***Time (s)***	***Relative error***	***R***	***Time (s)***	***R***	***Time (s)***	***R***	***Time (s)***	***Relative error***	***R***	***Time (s)***	***R***	***Time (s)***
C17	0.9152	0.068	0.03%	0.9149	0.003	0.9149	0.001	0.8548	0.068	0.01%	0.8549	0.003	0.8549	0.001
Majority	0.9298	0.158	0.04%	0.9294	0.003	0.9294	0.047	0.9175	0.165	0.04%	0.9171	0.003	0.9171	0.047
Full adder	0.8835	0.023	0.01%	0.8834	0.0005	0.8834	0.175	0.8839	0.023	0.09%	0.8834	0.0005	0.8834	0.189
Full adder (XOR/NAND)	0.9054	0.015	0.09%	0.9046	0.002	0.9046	0.001	0.8836	0.015	0.06%	0.8831	0.002	0.8831	0.001
Full adder	0.8250	0.033	0.11%	0.8241	0.008	0.8241	0.002	0.7732	0.033	0.12%	0.7741	0.008	0.7741	0.003
Comparator	0.8918	0.007	0.07%	0.8912	0.006	0.8912	0.001	0.9263	0.007	0%	0.9263	0.007	0.9263	0.001
Decoder2	0.8157	0.015	0.08%	0.8150	0.028	0.8150	0.009	0.9038	0.016	0.08%	0.9031	0.029	0.9031	0.009
MUX4	0.9408	0.153	0.02%	0.9406	0.001	0.9406	0.523	0.8679	0.151	0.01%	0.8678	0.001	0.8678	0.536

Table S2. Simulation results of ISCAS'85 benchmarks by SCM and MCS at a fixed CV of 0.001 ($\varepsilon = 5\times10^{-4}$).

Circuit	Characteristics			MCS ($\varepsilon = 5\times10^{-4}$)			SCM ($\varepsilon = 5\times10^{-4}$)		
	gates	inputs	outputs	Number of Simulations	Reliability	Runtime (s)	Sequence length (bits)	Reliability	Runtime (s)
C432	250	36	7	10000	0.9884	24.896	2,000	0.9885	0.050223
C499	202	41	32	30000	0.9686	54.105	2,000	0.9684	0.047493
C880	383	60	26	70000	0.9285	316.46	2,000	0.9284	0.094566
C1355	546	41	32	240000	0.8100	1545.5	20,000	0.8102	0.20498
C1908	880	33	25	250000	0.7963	3541.9	20,000	0.7960	0.38428
C2670	1193	157	64	270000	0.7805	3871.4	24,000	0.7808	0.56057
C3540	1669	50	22	350000	0.7332	7047.5	60,000	0.7329	1.3109
C5315	2307	178	123	600000	0.6327	7980.3	120,000	0.6327	3.4718
C6288	2416	32	32	2000000	0.3426	61351	600000	0.3427	32.640
C7552	3512	207	108	900000	0.5223	53833	350000	0.5212	65.199

- *J. Han, H. Chen, J. Liang, P. Zhu and Z. Yang are with the Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada T6G 2V4. E-mail: {jhan8, hc5, jinghang, peican, zhixi} @ualberta.ca.*
- *F. Lombardi is with the Department of Electrical and Computer Engineering, Northeastern University, Boston, USA. E-mail: lombardi@ece.neu.edu.*

Manuscript received on June 16, 2011. Please note that all acknowledgments should be placed at the end of the paper, before the bibliography.