Dimension Reduction in Statistical Simulation of Digital Circuits

Armin Alaghi and John P. Hayes
Advanced Computer Architecture Laboratory
Department of Electrical Engineering and Computer Science
University of Michigan, Ann Arbor, MI, 48109, USA
{alaghi, jhayes}@eecs.umich.edu

ABSTRACT
Statistical analysis tasks are increasingly encountered in the design of digital circuits that implement complex Boolean functions. Examples include analyzing the impact of soft errors, estimating power consumption, and stochastic computation. Common simulation techniques such as quasi-Monte Carlo simulation often converge too slowly for large circuits with many inputs, i.e., many dimensions. We analyze the probabilistic properties of Boolean functions, and identify new function classes called correlation insensitive (CI) which tolerate input dependencies. Correlation insensitivity enables dimension reduction, thereby significantly improving simulation quality. We investigate the theory of CI functions, present a new dimension-reduction algorithm, and show useful links to some well-known circuit design problems.

Author Keywords
Statistical simulation; logic simulation; dimension reduction; logic design; Boolean functions; correlation analysis.

INTRODUCTION
Integrated circuits (ICs) have long benefited from manufacturing improvements that have steadily increased their structural and functional complexity. As a result, ICs are becoming more sensitive to environmental interference and small processing variations, which can cause non-deterministic behavior [3]. Furthermore, IC design styles like stochastic computing are emerging where non-deterministic features are intentionally introduced to trade accuracy for power, area or speed [1][7][16]. Such circuits require probabilistic methods of analysis and simulation, unlike conventional approaches which are strictly deterministic.

Statistical methods such as Monte Carlo (MC) simulation [8] are widely used, mainly due to their high accuracy. Quasi-Monte Carlo (QMC) is a variant of MC that employs carefully selected deterministic samples instead of random samples [13]. It is preferred over MC in statistical analysis of digital circuits where it tends to converge faster to a solution [18]. Figure 1 compares the convergence rates (in terms of mean square error MSE) of MC and QMC for statistical simulation of the circuit in Figure 2. As can be seen, QMC converges much faster than MC, and exhibits an order-of-magnitude lower error rate.

To address current and future IC design needs, ever faster simulation techniques are needed. A major factor influencing QMC performance is the number of dimensions of the problem under consideration, i.e., the number of input parameters.

TMS/DEVS 2015, April 12 - 15, 2015, Alexandria, VA, USA

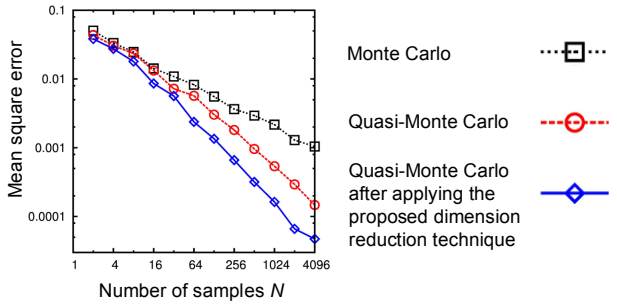

Figure 1. MC and QMC simulation of a 6-dimensional problem before and after applying dimension reduction.

Figure 1 previews of a key result of this paper: the convergence rate of QMC after applying the dimension-reduction technique proposed here is much faster than MC and QMC. In this example, the circuit dimension is reduced from 6 to 3, and convergence is significantly improved. It can be shown that the convergence rate of MC is of order $O(N^{-1/2})$, while that of QMC is approximately $O(N^{-1}\log^{D}(N))$, where N is the number of samples and D is the number of dimensions [13].

We propose a method to effectively reduce the dimensions of QMC in the statistical simulation of digital circuits. The key idea is to detect *correlation insensitivity* (CI) within the target circuit. Functions with CI are not affected by input dependencies. Hence independent dimensions associated with inputs can be merged into one dimension without affecting the circuit's output probability distribution. This reduces the circuit's dimensionality, leading to significant improvement in simulation performance and quality. A similar dimension reduction technique was employed by Singhee et al. in statistical timing analysis of digital circuits [18]. Their technique, however, does not apply to statistical simulation.

Figure 2 illustrates the proposed dimension reduction method in a basic form. The device shown is a 4-to-1 multiplexer, which connects one of the four data inputs x_3,x_4,x_5,x_6 to the output z under the control of the select inputs x_1,x_2. A naïve approach to statistical simulation of this circuit is to assign six individual dimensions, (independent random sources), to the six input variables. However, the four data inputs never affect the output simultaneously, so any correlation between them is not reflected in the output z, which is therefore CI with respect to x_3,x_4,x_5,x_6. This means that we can assign one random source to all four data inputs and effectively reduce the dimensions of the statistical simulation from six to three.

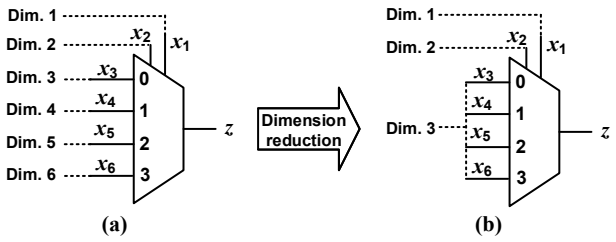

Figure 2. Illustration of dimension reduction: z is unaffected by correlations among data inputs x_3,x_4,x_5,x_6.

The paper is organized as follows. First, relevant aspects of digital logic circuits and probabilistic analysis are reviewed in the Background section. Then, CI is analyzed along with its use to reduce dimensionality. A helpful connection with existing logic design algorithms is established, and CI is shown to occur often in benchmark circuits. Finally, CI's utility in several applications is demonstrated.

The main contributions of this paper are:

1. Introduction and classification of CI Boolean functions
2. Demonstration of the role of CI in reducing the dimensionality of logic circuits
3. Application of CI-based dimension reduction to some statistical simulation tasks

BACKGROUND

This section establishes a mathematical framework for statistical simulation (SS) in the digital circuit context. The target circuits consist of logic gates (AND, OR, NOT, etc.), other common components (multiplexers, comparators, etc.), and their inter-connecting wires. The signals being processed take their values from the bit set {0,1}. Following engineering convention, the logic operations AND, OR and NOT on these signals are denoted by juxtaposition, + and ′, respectively. It will usually be clear from the context when juxtaposition and + refer to the arithmetic operations multiply and add, e.g., when dealing with probabilities and probability distributions.

For SS purposes, each wire x in a circuit is associated with a Bernoulli random variable (BRV) X with parameter p_X, which is the probability of seeing a 1 on x. The probability mass function (pmf) of X is $f_X(k) = P[X = k]$, where P[A] is the probability of event A. Since f_X is a binary function, $f_X(1)$ uniquely defines it. Note that $f_X(1) = p_X$.

The BRVs $X_1,\ldots,X_n$ associated with $x_1,\ldots,x_n$ may be correlated. Hence, their probabilistic behavior must be viewed as a joint probability distribution (or a joint pmf) $f_{X_1\ldots X_n}$.

$$f_{X_1\ldots X_n}(k_1,\ldots,k_n) = P[X_1 = k_1,\ldots,X_n = k_n]$$

The joint pmf specifies the probability of each of the 2^n possible combinations of X_i's. In the special case where all the X_i's are independent, the joint pmf reduces to the product of the *marginal* distributions of the individual BRVs.

$$f_{X_1\ldots X_n}(k_1,\ldots,k_n) = f_{X_1}(k_1) \times \ldots \times f_{X_n}(k_n) \qquad (1)$$

Definition 1: Let $z(x_1,\ldots, x_n)$ be the Boolean function realized by combinational circuit C. *Statistical simulation* (SS) is the process of estimating the pmf f_Z by applying samples from a joint pmf $f_{X_1\ldots X_n}$ to C. Note that f_Z can be calculated *exactly* using the following equation:

$$f_Z(1) = \sum_{k_1=0}^{1}\sum_{k_2=0}^{1}\ldots\sum_{k_n=0}^{1}[z(k_1,k_2,\ldots,k_n) \times f_{X_1X_2\ldots X_n}(k_1,k_2,\ldots,k_n)] \qquad (2)$$

which, along with $f_Z(0) = 1 - f_Z(1)$, completely specifies f_Z. Evaluating Eq. (2) is not practical, however, because of its exponential growth with n. SS is a practical alternative because it uses only a subset of the input combinations to estimate f_Z.

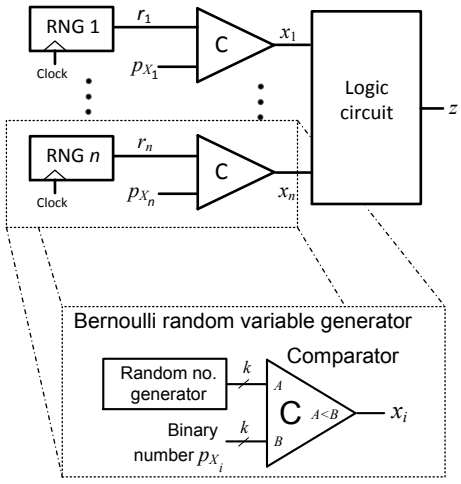

Figure 3. Statistical simulation set-up; random samples are generated at $r_1,\ldots,r_n$ to estimate probability distribution f_Z.

As an example, consider again circuit C_1 in Figure 2. If the inputs are independent BRVs with parameter 1/2, i.e., unbiased BRVs, then SS of C_1 involves generating samples from six independent random sources, recording the frequency of 1s and 0s on z, and estimating f_Z. As discussed earlier, we can use the same random source for the multiplexer's data inputs and reduce the number of sources by three. Each random source can be thought of as the toss of a fair coin. Linear feedback shift registers (LFSRs) are typically used to implement random bit generation [7]. Strictly speaking, LFSRs are pseudo-random generators, but they work well as random sources in practice.

In general, the input BRVs of a circuit can be biased, meaning that they can have a parameter other than 1/2, and they also can be non-independent (correlated). In such cases, the joint pmf of the inputs cannot be expressed in terms of individual pmfs, as in Eq. (1). So for SS purposes, we must sample directly from the joint pmf. The theoretical analysis of this paper applies to the general case. However, for ease of presentation, and since most applications employ independent inputs, we normally use examples that have independent RVs at their inputs.

Figure 3 shows a typical set-up for SS of a circuit with n independent inputs. The *Bernoulli random variable generator* (BG) produces BRVs with arbitrary parameter p_{X_i}. It includes a comparator (C) and an LFSR-based random number generator (RNG). In each clock cycle, C compares two k-bit binary numbers: the desired parameter p_{X_i} and a number r_i

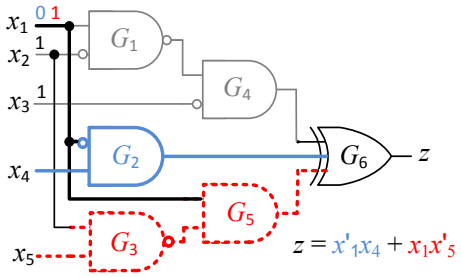

Figure 4. Five-input circuit; paths from x_4 (blue) and x_5 (red, dashed) are activated by different values of x_1.

generated by the RNG. If $r_i < p_{X_i}$, then $x_i = 1$; otherwise, $x_i = 0$. Hence, $x_i = 1$ has probability p_{X_i}, approximately. To generate n independent BRVs, n BGs with independent RNGs are used.

The *dimension* D of a circuit is defined as the number of independent random sources it needs for SS, and corresponds to the number of inputs. In general, the larger D, the harder the simulation problem. In methods like QMC [13], D directly affects the convergence rate. Regular MC is unaffected by dimension, but generally converges more slowly.

Methods of estimating the output pmf are known. An analytical approach is the most straightforward way to find the exact output pmf [14]. This is not practical for large circuits, as its complexity grows exponentially with circuit size. Heuristic methods exist that are quite fast but are inaccurate [6]. MC and QMC are considered to be accurate methods of estimating the output pmf, with accuracy increasing with number of samples.

CORRELATION INSENSITIVITY

We first illustrate CI using the function $z(x_1,x_4,x_5) = x_1'x_4 + x_1x'_5$ obtained by setting $x_2 = x_3 = 1$ in Figure 4. This is essentially the 2-to-1 multiplexer function, so it has the special property that inputs x_4 and x_5 cannot propagate to the output z simultaneously. In other words, x_4 and x_5 never affect z at the same time. When $x_1 = 1$, the path from x_5 to z via G_3 and G_5 (highlighted in red, dashed) becomes active, and the path from x_4 through G_2 (highlighted in blue) is blocked. If $x_1 = 0$, the opposite happens. Assuming X_1 is independent of the other variables, we can write

$$f_Z(1) = f_{X_1}(0)f_{Z_{x'_1}}(1) + f_{X_1}(1)f_{Z_{x_1}}(1)$$

which is the probabilistic version of Boole-Shannon expansion (Theorem 4 in the Appendix). $z_{x'_1} = x_4$ is the negative cofactor of z with respect to x_1 obtained by setting $x_1 = 0$ in z; $z_{x_1} = x'_5$ is the positive cofactor of z obtained by setting $x_1 = 1$. Hence,

$$f_Z(1) = f_{X_1}(0).f_{X_4}(1) + f_{X_1}(1).f_{X_5}(0)$$

implying that the output is a function of the marginal distributions f_{X_4} and f_{X_5} only, and not of their joint pmf $f_{X_4X_5}$, as if X_4 and X_5 were independent BRVs. This is because x_4 and x_5 do not appear in the x_1 cofactors of z simultaneously. A function like z is CI with respect to x_4 and x_5 because its output pmf is unaffected by correlations between X_4 and X_5.

Definition 2: A Boolean function $z(x_1,\ldots,x_n)$ is *correlation insensitive* with respect to variables x_1 and x_2 if the distribution (pmf) of Z only depends on the marginal probability distributions $f_{X_2X_3\ldots X_n}$ and $f_{X_1X_3\ldots X_n}$ and not the joint pmf $f_{X_1\ldots X_n}$. We refer to x_1 and x_2 as *CI inputs* of z.

Equivalently, we could define $z(x_1,\ldots,x_n)$ as CI with respect to x_1 and x_2, if x_1 and x_2 do not appear in the cofactors of z with respect to $x_3,\ldots,x_n$, simultaneously. This equivalence is shown in the proof of Theorem 1 in the Appendix.

In a similar SS context, Yu et al. [19] define *compatibility*, which is a special case of CI. The compatible inputs of [19] are those with independent and identically distributed BRVs, in which case they can be tied together for SS purposes. Definition 2, on the other hand, allows *any* joint distribution among the inputs.

Example 1: Consider again the function $z(x_1,x_4,x_5) = x_1'x_4 + x_1x_5'$ in Figure 4. We will show that correlations between X_4 and X_5 do not affect the output pmf, whereas correlations between X_1 and X_4 do. According to Eq. (2),

$$f_Z(1) = \sum_{k_1=0}^{1}\sum_{k_4=0}^{1}\sum_{k_5=0}^{1} z(k_1,k_4,k_5)f_{X_1X_4X_5}(k_1,k_4,k_5)$$
$$= f_{X_1X_4X_5}(0,1,0) + f_{X_1X_4X_5}(0,1,1) + f_{X_1X_4X_5}(1,0,0) + f_{X_1X_4X_5}(1,1,0)$$

The first two terms marginalize X_5 and the last two terms marginalize X_4, yielding

$$f_Z(1) = f_{X_1X_4}(0,1) + f_{X_1X_5}(1,0)$$

Z's pmf is a function of $f_{X_1X_4}$ and $f_{X_1X_5}$ only, so by Definition 2, z is CI with respect to x_4 and x_5. The function z is *not* CI with respect to x_1 and x_4 (or x_1 and x_5), because terms like $f_{X_1X_4}$ (or $f_{X_1X_5}$) appear in the pmf of Z. The rest of the example shows how correlations between X_4 and X_5 leave Z unaffected, but those between X_1 and X_4 alter the distribution of Z.

Case 1: All the input BRVs X_1, X_4, X_5 are identically distributed and independent with $f_{X_i}(1) = 0.5$. Consequently,

$$f_Z(1) = f_{X_1X_4}(0,1) + f_{X_1X_5}(1,0)$$
$$= f_{X_1}(0)f_{X_4}(1) + f_{X_1}(1)f_{X_5}(0) = 0.5 \quad (3)$$

Case 2: The input BRVs have the same marginal distribution as before ($f_{X_i}(1) = 0.5$), but now X_4 and X_5 are highly correlated with the following distribution, while X_1 remains independent.

$f_{X_4X_5}(0,0) = f_{X_4X_5}(1,1) = 0.5$ and $f_{X_4X_5}(0,1) = f_{X_4X_5}(1,0) = 0$

or, equivalently, in vector notation, $\vec{f}_{X_4X_5} = [0.5 \quad 0 \quad 0 \quad 0.5]$. The correlation here leaves Z unaffected, following (3). This shows that Z is insensitive to correlations between X_4 and X_5.

Case 3: Again $f_{X_i}(1) = 0.5$ and X_5 is independent. This time assume X_1 and X_4 are highly correlated with joint distribution $\vec{f}_{X_1X_4} = [0.5 \quad 0 \quad 0 \quad 0.5]$. The pmf of Z changes to

$$f_Z(1) = f_{X_1X_4}(0,1) + f_{X_1X_5}(1,0) = 0 + 0.25 = 0.25$$

indicating that z is *not* CI with respect to x_1 and x_4. □

When a circuit's output is CI with respect to two variables, SS can be performed as if those inputs were highly correlated, even when they are independent. For example, if the two inputs have the same probability, we can tie them together as if they were one. This observation is the basis of our proposed dimension-reduction approach, which is summarized below.

Dimension reduction: In SS of a Boolean function z that is CI with respect to a subset S of its inputs, RNGs can be shared by the members of S. This may introduce correlation among those members, but the probability distribution of z is unaffected. As a result, the dimension D of z, is reduced.

Next, we show that CI can be expressed in terms of the *Boolean difference* (BD). The BD of z with respect to x_i is $dz/dx_i = z_{x_i'} \oplus z_{x_i}$, where $\oplus$ denotes XOR (exclusive-OR), and z_{x_i} and $z_{x_i'}$ are z's cofactors [9]. For the function $z = x_1'x_4 + x_1x_5'$ of Example 1, $dz/dx_1 = x_4 \oplus x'_5$, $dz/dx_4 = x'_1$ and $dz/dx_5 = x_1$. If $dz/dx_1 = 0$, then x_1 is *redundant*, meaning that it has no influence on z. This enables X_1 to be marginalized out of Eq. (2), making f_Z a function of $f_{X_2 \ldots X_n}$ only. In this case, according to Definition 2, z is CI with respect to x_1 and x_j for any $j = 2,3,\ldots,n$. This points to a method for identifying CI functions.

Theorem 1: Function $z(x_1,x_2,\ldots,x_n)$ with $n > 2$ variables is CI with respect to x_1 and x_2 if and only if for every cube (product term) c containing only variables $x_3,x_4,\ldots,x_n$, at least one input (x_1 or x_2) is redundant in the cofactor $z_c(x_1, x_2)$ or, equivalently,

$$dz_c/dx_1 = 0 \quad \text{or} \quad dz_c/dx_2 = 0$$

A proof can be found in the Appendix. For Example 1, $dz_{x_1}/dx_4 = 0$, and $dz_{x_1'}/dx_5 = 0$, so in all the cofactors of z with respect to the remaining variable x_1, at least one of the variables x_4 and x_5 is redundant, confirming that z is CI with respect to x_4 and x_5.

It is possible to have Boolean functions that are CI with respect to more than two variables. In the extreme case, a function can be CI with respect to all its inputs, in which case, the function is just a constant and the output probability is either 0 or 1.

Definition 3: $z(x_1,\ldots,x_n)$ is (*strongly*) *correlation insensitive* (CI) *with respect to variable set* $\mathbb{X} = \{x_1 \ldots, x_k\}$, if it is CI with respect to every pair x_i,x_j in $\mathbb{X}$.

Example 2: The circuit in Figure 5 realizes $z = x_1(x_2' + x_3' + x_4 + x_5)(x_4' + x_6' + x_7')(x_4' + x_6 + x_8')$ which is CI with respect to $\{x_5, x_7\}$, $\{x_5, x_8\}$ and $\{x_7, x_8\}$, among others. Hence, z is CI with respect to $\mathbb{X} = \{x_5, x_7, x_8\}$. Other CI sets present in this function are $\{x_2, x_7, x_8\}$ and $\{x_3, x_7, x_8\}$. □

Correlation insensitivity occurs in several other ways. A function may not be CI with respect to any pair of variables, but have a larger subset of variables that do not affect the output at the same time.

Definition 4: $z(x_1, \ldots, x_n)$ is *weakly correlation insensitive* (WCI) with respect to variables $\mathbb{X} = \{x_1, \ldots, x_k\}$ if the distribution of the BRV Z is only a function of the k marginal probability distributions in which one of the BRVs $X_1,\ldots,X_k$ is

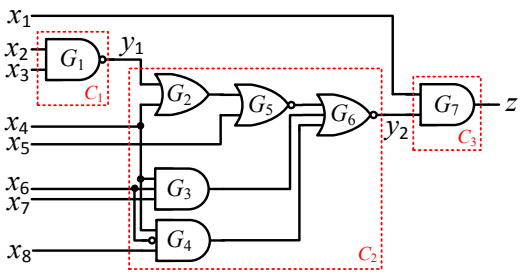

Figure 5. Circuit used in Examples 2 and 4.

marginalized out, i.e., $f_{X_2X_3\ldots X_kX_{k+1}\ldots X_n}$, $f_{X_1X_3X_4\ldots X_kX_{k+1}\ldots X_n}$, $\cdots$ $f_{X_1X_2X_3\ldots X_{k-1}X_{k+1}\ldots X_n}$. $\mathbb{X}$ is called a *WCI set* of z.

Weak CI generalizes CI, so WCI sets can be expected to occur more often than (strong) CI sets. A CI input-pair is a WCI set of size 2, because if the function z is CI with respect to x_1 and x_2, then the BRV Z is a function of $f_{X_2\ldots X_n}$ and $f_{X_1X_3\ldots X_n}$ only, so by Definition 4, $\{x_1, x_2\}$ is also a WCI of z. Similarly, (strong) CI sets of variables are special cases of WCI sets.

Identifying WCI sets is harder than finding CI pairs because more variables are involved. For simplicity, we only consider WCI sets of size $k = 3$. If $z(x_1,\ldots,x_n)$ is WCI with respect to $\{x_1, x_2, x_3\}$, then for every cube c containing $x_4,\ldots,x_n$, at least one variable from the set $\{X_1, X_2, X_3\}$ must be marginalized out in the corresponding iteration of Eq. (2). This is only possible if some variable is redundant in z_c, or if z_c is CI with respect to at least one pair of its variables. Hence, z is WCI with respect to $\{x_1, x_2, x_3\}$ if and only if for every cube c containing the variables $x_4,\ldots,x_n$, the cofactor z_c is CI with respect to $\{x_1, x_2\}$, $\{x_1, x_3\}$ or $\{x_2, x_3\}$.

Theorem 2: z is WCI with respect to $\mathbb{X} = \{x_1, \ldots, x_k\}$ if and only if for every cube c containing the variables $x_{k+1},\ldots,x_n$, zc is WCI with respect to at least one subset of size $k - 1$ of $\mathbb{X}$.

Example 3: The function $z = x_1x_2x_4 + x_1x_3x'_4x'_5 + x_2x_3x'_4x_5$ (Figure 6) has no pair of CI inputs. However, z is WCI with respect to $\{x_1, x_2, x_3\}$ as it has the cofactors $z_{x'_4x'_5} = x_1x_3$, $z_{x'_4x_5} = x_2x_3$ and $z_{x_4x'_5} = z_{x_4x_5} = x_1x_2$, which are CI with respect to $\{x_1, x_2\}$, $\{x_1, x_2\}$ and $\{x_1, x_3\}$, resp. Like strong CI sets, WCI sets provide dimension reduction in SS. □

Finally, we observe that correlation insensitivity is not directly related to the similar-sounding concept of correlation *immunity* used in cryptography [17]. The latter measures the extent to which a Boolean function's outputs are statistically independent of (uncorrelated with) subsets of its inputs. Correlation insensitivity is only concerned with dependencies among the inputs, and those dependencies are of a more restricted type.

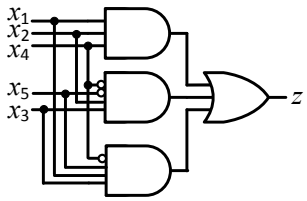

Figure 6. Circuit with weak CI set $\{x_1, x_2, x_3\}$.

DETECTING CORRELATION INSENSITIVITY

Theorem 1 implicitly provides a way to detect CI. To construct a more practical method, we relate CI detection to a well-known problem in the digital circuit field, automatic test pattern generation (ATPG) [5]. In order to find CI inputs of a circuit, we only need to perform ATPG for the faults that occur on the inputs. (See Theorem 5 in the Appendix.)

To gauge CI's frequency in real circuits, we examined the individual output functions of the widely-used ISCAS suite of benchmark circuits [11], and also a 16-bit binary adder circuit. We counted their CI input pairs and WCI sets of size three via an ATPG-based program; Table 1 shows the results. Several outputs were sampled from each circuit. For example, *adder*16_*cout* refers the adder's final carry-out function, while *c*432_*o*1 refers to the first output of the *c*432 benchmark. These results show that CI is commonplace. It varies significantly from circuit to circuit, but generally increases with circuit size.

To make the CI detection algorithm more scalable, divide-and-conquer can be used. Our approach is to partition the circuit into *supergates* [19], which confine reconvergent fanout to single-output sub-circuits. The overall circuit then becomes a tree of supergates with no high-level fanout. Hence, we only need to search within the supergates in order to find CI. Furthermore, it can be shown that in such tree-structured circuits, the CI of the intermediate signals propagates to their fan-in cones and to the output, leading to the following result.

Corollary 3: Consider the Boolean function $z(g, h)$, in which $g(x_1,\ldots,x_n)$ and $h(y_1,\ldots,y_m)$ are sub-functions. If z is CI with respect to g and h, then z is also CI with respect to x_i and y_j for any i and j. Further, if g is CI with respect to x_1 and x_2, then z is also CI with respect to x_1 and x_2.

Example 4: The circuit of Figure 5 can be partitioned into three supergates C_1, C_2 and C_3 as indicated by dashed lines.

Circuit	No. of inputs	No. of gates	No. of CI input-pairs	No. of WCI sets of size 3
adder16_cout	32	125	0	0
c432_o1	18	20	0	0
c432_o2	27	58	0	0
c880_o1	36	80	1	38
c880_o2	32	63	1	34
c1196_o1	21	180	24	377
c1196_o2	20	118	12	204
c1196_o3	19	132	16	222
c1196_o4	23	187	36	586
c1196_o5	22	203	58	652
c1238_o1	21	203	23	358
c1238_o2	23	209	36	586
c1238_o3	15	52	9	99
c1238_o4	10	39	0	0
c2670_o1	11	24	28	140
c2670_o2	12	22	32	180
c5315_o1	14	50	41	279
c5315_o2	24	78	72	1,044
c6288_o1	24	630	0	0
c6288_o2	26	747	0	0
c7552_o1	29	342	108	1,817
c7552_o2	29	153	108	1,818
c7552_o3	20	97	54	591

Table 1. Presence of CI in some benchmark circuits.

Supergate C_2 realizes $y_2 = y_1x'_4 + x'_4x_5 + x_4x_6x'_7 + x_4x'_6x'_8$ and is the only one containing reconvergent fanout. Since C_2 is smaller than the original circuit, its CI sets can be detected more quickly. Analysis shows that y_2 is CI with respect to seven pairs: $\{y_1, x_6\}$, $\{y_1, x_7\}$, $\{y_1, x_8\}$, $\{x_5, x_6\}$, $\{x_5, x_7\}$, $\{x_5, x_8\}$ and $\{x_7, x_8\}$. According to Corollary 3, all CI relations of y_1 propagate to its fan-in cone x_2 and x_3, and all the CI properties of y_2 propagate to the output z. Thus we can conclude that z is CI with respect to x_2 and x_8, a result already seen in Example 2. □

The scalability of the proposed approach depends on the number and the size of the supergates of the circuit. Supergates are often much smaller than the circuits that contain them. Min and Park [11] show that for the ISCAS circuits, more than 87% of the supergates have 10 or fewer gates, and over 99% have 100 or fewer gates. The average supergate size is around 8 gates. These numbers suggest that supergate partitioning is effective at detecting CI in larger circuits. CI detection for *adder*16_*cout* was reduced from 4 minutes to a fraction of a second using this divide-and-conquer approach.

CASE STUDIES

This section demonstrates several applications of the proposed dimension-reduction technique to circuit design problems.

Dimension reduction in c1196

As reported in Table 1, many CI pairs and WCI sets exist in c1196, the combinational part of ISCAS benchmark s1196. Consider the single-output sub-circuit denoted c1196_o1 in Table 1. It has 21 inputs x_1, …, x_{21} and 24 CI pairs. Figure 7 illustrates c1196's CI relations in graph form. The vertices correspond to the inputs, and a solid edge between x_i and x_j means that the output is CI with respect to them.

Even though c1196 has 24 CI pairs, it is not possible to merge them all. For instance, the output is CI with respect to $\{x_5, x_{19}\}$ and $\{x_6, x_{19}\}$, but we cannot combine these three inputs because the output is still sensitive to correlations between x_5 and x_6. By Definition 3, in order to have a (strong) CI set, every pair of inputs must be CI. This corresponds to a fully-connected sub-graph or *clique* in the graph representation. Finding the maximal CI sets can now be done by clique partitioning [9]. Figure 7 shows one of many such partitions. The sets $\{x_{18}, x_{19}, x_{20}, x_{21}\}$ (shown in blue) and $\{x_{16}, x_{17}\}$ (red) are identified as (strong) CI sets and can be merged during statistical simulation.

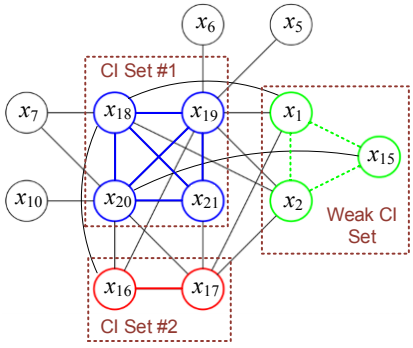

Figure 7. Graph illustrating CI sets of the c1196 benchmark; the clustered nodes can be merged during SS.

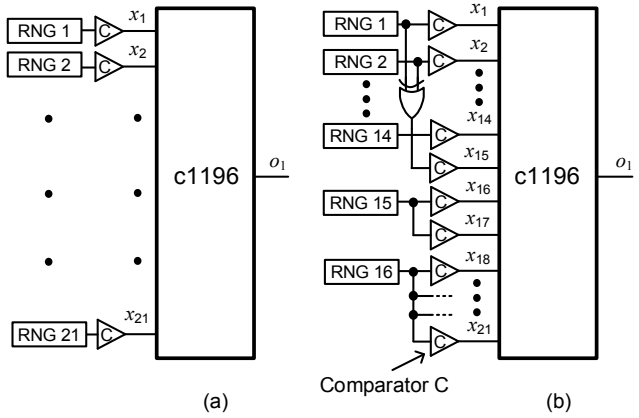

Figure 8. Set-up for SS of the c1196_o1 benchmark circuit (a) before and (b) after applying dimension reduction.

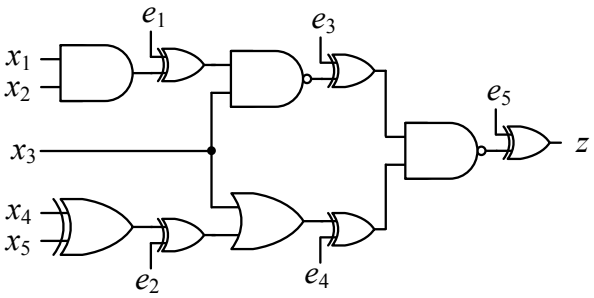

Figure 9. 5-input circuit with XOR gates to model soft errors.

Besides strong CI sets, c1196 has many weak CI sets that may or may not overlap strong ones. One non-overlapping WCI set is depicted in Figure 7 (green dashed lines). Thus, the WCI set $\{x_1, x_2, x_{15}\}$ can also be used to further reduce the dimensions of c1196 simulation. The three CI sets of Figure 7 reduce the dimensions of c1196 from 21 to 16 for SS purposes. Figure 8 shows a simulation set-up for c1196: only 16 independent RNGs are needed after applying dimension reduction. Note that the random number sources for x_{15} are obtained by XOR-ing the RNGs of x_1 and x_2; this is explained later.

Soft error analysis

Next, we show how our dimension reduction technique can improve soft-error analysis. Figure 9 gives a 5-input circuit in which an XOR gate is added to the output of each gate in order to model soft errors affecting the circuit, a standard technique in simulation-based reliability analysis. Since this falls into the SS category it is possible to exploit correlation insensitivity. For example, z is CI with respect to $\{x_1, x_4\}$, $\{x_2, x_5\}$, and $\{e_1, e_2\}$, so we can share independent random sources among them. Such dimension reduction can improve simulation quality, especially if QMC is used. Figure 10 shows that after exploiting CI, the simulation quality improves significantly

Stochastic circuits

Stochastic computing (SC) is an unconventional computing style that represents a number by a (pseudo) random bit-stream in which the fraction of 1's denote a probability value [1][7]. For example, X = 00001101 is one of many possible representations of the probability p_X = 3/8. Computation is performed by applying such bit-streams to a conventional

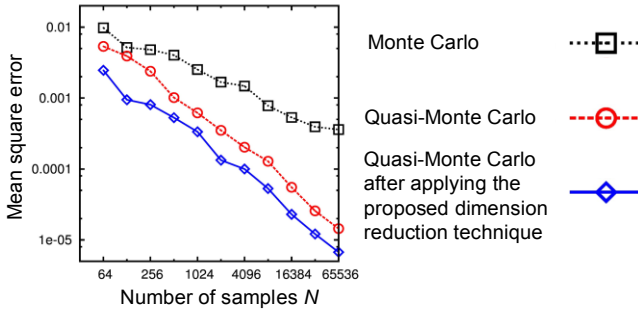

Figure 10. Simulation data for the circuit of Figure 9 showing that dimension reduction significantly decreases MSE.

logic circuit and estimating the probability of each output bit-stream. This process is, in fact, a kind of hardware-based SS.

The key advantage of SC is that it uses extremely small and error-tolerant logic circuits to implement arithmetic operations which require orders of magnitude more hardware with conventional binary numbers. For example, a single AND gate performs the multiplication $p_X p_Y$, while a multiplexer performs the scaled addition $(p_X + p_Y)/2$; see Figure 11. An error that flips a bit of an N-bit stream X has little effect on p_X provided N is sufficiently large. On the other hand, SC tends to be much slower and less accurate than conventional binary arithmetic. Nevertheless, SC is an attractive technology for certain applications such as implantable biomedical devices [2].

An AND gate with N-bit inputs X and Y computes $p_X p_Y$ only if the inputs are statistically independent, i.e., uncorrelated, and N is sufficiently large; see Figure 11a. Figure 11b shows a case where the bit-streams for X and Y have identical bit patterns as well as values (they are maximally correlated). This produces the output p_X instead of p_X^2, a big inaccuracy. In general the inputs to a stochastic circuit need to be statistically independent. Generating many independent inputs for a stochastic circuit has significant area cost [15]. If, however, the output of the circuit is CI with respect to some inputs, the random number sources can be shared among those inputs thereby reducing the overall cost. Note that the stochastic adder is unaffected by correlations among its data inputs, as seen in Figure 11d.

Consider again the Boolean function $z(r, x, y) = r'x + ry$ which implements a stochastic adder; see Figure 11c. Assuming the three inputs are BRVs with parameter 0.5, we need three independent random sources. These are usually obtained from a 3-bit LFSR [7]. The joint pmf of X and Y then is $\vec{f}_{XY}$ = [0.25 0.25 0.25 0.25] . However, as seen in a similar

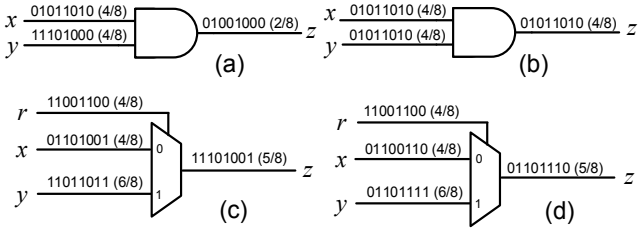

Figure 11. (a) Stochastic multiplier with independent and (b) correlated input bit-streams. Stochastic adder with (c) independent and (d) correlated input bit-streams.

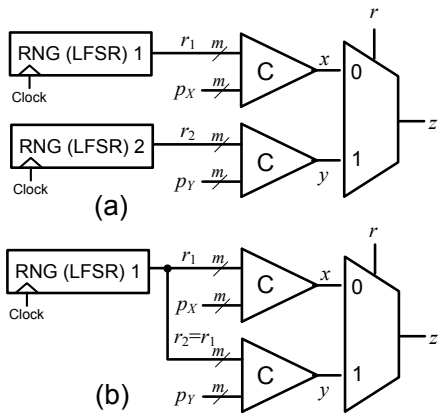

Figure 12. Dimension reduction in a stochastic adder; (a) with independent RNGs and (b) with shared RNGs.

function in Example 1, we can replace this with $\vec{f}^*_{XY}$ = [0.5 0 0 0.5] without altering the output pmf. Having $\vec{f}^*_{XY}$ means that X and Y have the same value; so inputs x and y can be connected.

If a stochastic circuit needs arbitrary input values, we must add Bernoulli random variable generators (BGs) to the inputs. Figure 12a shows two BGs connected to a multiplexer-based stochastic adder. Two independent m-bit LFSRs are used as RNGs in this example. Counting the comparators we get a circuit with $2m + 1$ inputs (r_1, r_2 and the multiplexer's select input r), each with value 0.5. As noted earlier, correlation insensitivity propagates through the fan-in cones of x and y, i.e., the comparators. So the newly obtained function z is CI to every input pair $r_1[i]$ and $r_2[j]$ taken from bits of r_1 and r_2. Since all the inputs have value 0.5, we can connect each bit of r_1 to a corresponding bit at r_2. This yields the circuit in Figure 12b which uses only one LFSR.

This idea can be immediately extended to strongly CI sets. For example, the reconfigurable stochastic architecture described by Qian et al. [15] is similar to that of Figure 2 in that its main component is a k-to-1 multiplexer, where k is the degree of a polynomial being implemented. Since the multiplexer is CI with respect to all its data inputs, one RNG can be shared among all k inputs, leading to significant cost reduction.

Weakly CI sets can also be exploited to reduce the cost of SC. Consider Figure 6 again, and assume that all the inputs are SNs of value 0.5. The joint pmf of X_1, X_2, and X_3 is thus

$$\vec{f}_{X_1X_2X_3} = [0.125 \;\; 0.125 \;\; 0.125 \;\; 0.125 \;\; 0.125 \;\; 0.125 \;\; 0.125 \;\; 0.125]$$

However, since z is weakly CI with respect to $\{x_1, x_2, x_3\}$, we can replace the input pmf with the following:

$$\vec{f}^*_{X_1X_2X_3} = [0.25 \;\; 0 \;\; 0 \;\; 0.25 \;\; 0 \;\; 0.25 \;\; 0.25 \;\; 0]$$

without altering the output pmf. Note that even though the joint pmf is changed, the marginal distributions of each pair, i.e., $\vec{f}^*_{X_1X_2}$, $\vec{f}^*_{X_1X_3}$, and $\vec{f}^*_{X_2X_3}$, are the same as before. Since the output pmf is only a function of the marginal distributions, it remains unaffected. The new pmf $\vec{f}^*_{X_1X_2X_3}$ can be generated by assigning two random bits to x_1 and x_2, and assigning $x_3 = x_1 \oplus x_2$. This guarantees that each pair remains independent. Extending the foregoing concepts to inputs of arbitrary value, we can replace one RNG by a bit-wise XOR function of the other RNGs if weakly CI sets are present in the circuit. In Figure 8b, the RNG used for x_{15} is obtained by XORing the RNGs of x_1 and x_2, because $\{x_1, x_2, x_{15}\}$ form a WCI set.

Circuit	Initial area (μm^2)	No. of CI input-pairs	Area after dimension reduction (μm^2)
Example from [15]	2,393	6	818
Gamma correction [15]	10,524	21	3,291
Gaussian function [10]	10,436	120	4,039
Degree-3 *CheckNode* [12]	2,649	0	2,649
4-input SC neuron* [4]	15,426	24	8,932
8-input SC neuron* [4]	30,906	112	15,752
Circuit of Figure 6**	9,495	0 **	9,046
Stochastic adder* [7]	3,925	1	2,842

* Eight-bit BGs are included in the area calculations.
** This circuit has no CI pairs but has one weakly CI set of size 3.

Table 2. Impact of dimension reduction on circuit area.

To illustrate the effectiveness of CI-based dimension reduction in stochastic circuits, we compare the area of several existing SC designs before and after applying our method. All the designs were mapped to ICs using a standard layout tool, the Berkeley *sis* program, Table 2 summarizes the results, and indicates that dimension reduction can save up to 68% in IC area. In the case of *CheckNode*, no correlation insensitivity was detected, so no area reduction was achieved.

CONCLUSIONS

We have introduced and analyzed a property of Boolean functions called correlation insensitivity which is very useful for statistical simulation. Correlation-insensitive (CI) functions are unaffected by statistical dependencies among certain inputs. This makes it possible to merge those inputs and so reduce the dimensions of the simulation problem. We presented rigorous definitions of basic CI properties including strong and weak correlation insensitivity. We also described a practical way of detecting them via test pattern generation. We found that some degree of CI is widespread in circuits like the ISCAS bench-marks. Finally, we showed, that our dimension reduction method can significantly improve simulation quality, and also improve the design of stochastic circuits.

ACKNOWLEDGEMENT

This work was supported by Grant CCF-1318091 from the National Science Foundation.

REFERENCES

1. A. Alaghi and J.P. Hayes. Survey of stochastic computing. *ACM Trans. Embed. Comp. Sys.*, *12* (2013), art. 92.
2. A. Alaghi, C. Li, and J.P. Hayes. Stochastic circuits for real-time image-processing applications. *Proc. DAC 2013*, paper 136.
3. S. Borkar. Designing reliable systems from unreliable components. *IEEE Micro*, *25* (2005), 10-16.
4. B.D. Brown and H.C. Card. Stochastic neural computation. I. Computational elements. *IEEE Trans. Comp. 50* (2013), 891-905.

5. M.L. Bushnell and V.D. Agrawal. *Essentials of Electronic Testing*, New York: Springer, 2000.
6. S. Ercolani el al. Estimate of signal probability in combinational logic networks. *Proc. ETC 1989*, 132-138.
7. B.R. Gaines. Stochastic computing systems. *Advances in Information Systems Science*, *2* (1969), 37-172.
8. J.M. Hammersley and D.C. Handscomb. *Monte Carlo Methods*. London: Methuen, 1964.
9. G.D. Hachtel and F. Somenzi. *Logic Synthesis and Verification Algorithms*. Boston: Kluwer, 1996.
10. P. Li et al. The synthesis of complex arithmetic computation on stochastic bit streams using sequential logic. *Proc. ICCAD 2012*, 480-487.
11. H.B. Min and E.S. Park. Graph-theoretic algorithm for finding maximal supergates in combinational logic circuits. *IEE Proc. Circuits, Devices and Syst.*, *143* (1996), 313-318.
12. A. Naderi et al. Delayed stochastic decoding of LDPC codes. *IEEE Trans. Signal Proc*. (2011), 5617-5626.
13. H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*, SIAM Series App. Math., *32* (1992).
14. K.P. Parker and E.J. McCluskey. Probabilistic treatment of general combinational networks. *IEEE Trans. Comp.* (1975) 668-670.
15. W. Qian et al. An architecture for fault-tolerant computation with stochastic logic, *IEEE Trans. Comp.*, (2011), 93-105.
16. N.R. Shanbhag et al. Stochastic computation. *Proc. DAC 2010*, 859-864.
17. T. Siegenthaler. Correlation-immunity of nonlinear combining functions for cryptographic applications. *IEEE Trans. Info. Th.*, (1984), 776–780.
18. A. Singhee et al. Practical, fast Monte Carlo statistical static timing analysis: Why and how. *Proc. ICCAD 2008*, 190-195.
19. C.C. Yu et al. Scalable sampling methodology for logic simulation: Reduced-Ordered Monte Carlo. *Proc. ICCAD 2012*, 195-201.

Appendix: Theorem Proofs

This appendix proves Theorem 1. Theorem 2 is a straightforward generalization of Theorem 1, so its proof is omitted. Two additional theorems (Theorems 4 and 5) are also presented and proven.

Theorem 1: *Proof*: (Sufficiency) For every combination of $x_3 \ldots x_n$, i.e., for every cube c containing only the variables $x_3, x_4, \ldots, x_n$, the cofactor z_c is a function of one of the variables x_1 or x_2, but not both. This means that $dz_c/dx_1 = z_c(0,x_2) \oplus z_c(1,x_2) = 0$ or $dz_c/dx_2 = z_c(x_1,0) \oplus z_c(x_1,1) = 0$, implying

$$z_c(0,0) = z_c(1,0) \text{ and } z_c(0,1) = z_c(1,1)$$
or
$$z_c(0,0) = z_c(0,1) \text{ and } z_c(1,0) = z_c(1,1) \quad (4)$$

Next, we express $f_Z(1)$ using Eq. (2) and unroll the first two sums (with respect to k_1 and k_2) to enumerate all the four possible cases of $z_c(0,0)$, $z_c(0,1)$, $z_c(1,0)$ and $z_c(1,1)$:

$$\sum_{k_3=0}^{1} \ldots \sum_{k_n=0}^{1} [z(0,0,k_3,\ldots,k_n) f_{X_1X_2\ldots X_n}(0,0,k_3,\ldots,k_n) + z(0,1,k_3,\ldots,k_n) f_{X_1X_2\ldots X_n}(0,1,k_3,\ldots,k_n) + z(1,0,k_3,\ldots,k_n) f_{X_1X_2\ldots X_n}(1,0,k_3,\ldots,k_n) + z(1,1,k_3,\ldots,k_n) f_{X_1X_2\ldots X_n}(1,1,k_3,\ldots,k_n)] \quad (5)$$

Equations (4) implies that for each iteration of the preceding summation, we can marginalize either X_1 or X_2, yielding

$$z(0,0,k_3,\ldots,k_n) f_{X_2\ldots X_n}(0,k_3,\ldots,k_n) + z(1,1,k_3,\ldots,k_n) f_{X_2\ldots X_n}(1,k_3,\ldots,k_n)$$

or

$$z(0,0,k_3,\ldots,k_n) f_{X_1X_3\ldots X_n}(0,k_3,\ldots,k_n) + z(1,1,k_3,\ldots,k_n) f_{X_1X_3\ldots X_n}(1,k_3,\ldots,k_n)$$

So the distribution of Z is a function of the marginal distributions $f_{X_2\ldots X_n}$ and $f_{X_1X_3\ldots X_n}$. From Definition 2, z is CI with respect to x_1 and x_2.

(Necessity) By way of contradiction, if z is CI with respect to x_1 and x_2, but there is a cube c with $dz_c/dx_1 \neq 0$ and $dz_c/dx_2 \neq 0$, then z_c is a non-degenerate function of both x_1 and x_2. By enumerating all possibilities for z_c—there are only 10 non-degenerate functions of two variables—we see that for the particular iteration of (5) that corresponds to c, none of the variables X_1 and X_2 can be marginalized. Hence, z is not CI with respect to x_1 and x_2, a contradiction from which the necessary condition follows. □

Theorem 4: (*Probabilistic Boole-Shannon expansion*) For a Boolean function $z(x_1,\ldots,x_n)$, if BRV X_1 is independent of the remaining variables, then

$$f_Z(1) = f_{X_1}(0) f_{Z_{x'_1}}(1) + f_{X_1}(1) f_{Z_{x_1}}(1)$$

where Z_{x_1} and $Z_{x'_1}$ are the BRVs corresponding to the positive and negative cofactors, respectively, of z with respect to x_1.

Proof: The events $X_1 = 0$ and $X_1 = 1$ are complementary, so

$$f_Z(1) = f_{Z|X'_1}(1).P[X_1 = 0] + f_{Z|X_1}(1).P[X_1 = 1] = f_{Z|X'_1}(1).f_{X_1}(0) + f_{Z|X_1}(1).f_{X_1}(1) \quad (6)$$

$f_{Z|X_1}$ ($f_{Z|X'_1}$) is the conditional distribution of Z with respect to the event $X_1 = 1$ ($X_1 = 0$), and is defined as $f_{Z|X_1}(1) = f_{ZX_1}(1,1)/f_{X_1}(1)$. Since X_1 is independent of the remaining variables, we have $f_{Z|X_1}(1) = f_{Z_{x_1}}(1)$. Similarly, $f_{Z|X'_1}(1) = f_{Z_{x'_1}}(1)$, so by Eq. (6)

$$f_Z(1) = f_{X_1}(0).f_{Z_{x'_1}}(1) + f_{X_1}(1).f_{Z_{x_1}}(1) \quad \square$$

Theorem 5: $z(x_1,\ldots,x_n)$ is CI with respect to x_1 and x_2 if and only if there are no test patterns that detect a single stuck-at fault on x_1 and a single stuck-at fault on x_2.

Proof: The solutions to $dz/dx_1 = 1$ are all possible test patterns that detect the faults on x_1. Similarly, the solutions to $dz/dx_2 = 1$ are all tests for the faults on x_2. An input pattern that detects both fault sets must be a solution to both equations. However, if z is CI with respect to x_1 and x_2, then according to Theorem 1, for every cube c containing the variables $x_3,\ldots,x_n$, we have $dz_c/dx_1 = 0$ or $dz_c/dx_2 = 0$. This means there is no input combination for which both $dz_c/dx_1 = 1$ and $dz_c/dx_2 = 1$, implying that no test pattern exists that detects both faults sets.

Conversely, if a test pattern exists, then for the cube c (containing the variables $x_3,\ldots,x_n$) taken from the test pattern, both $dz_c/dx_1 \neq 0$ and $dz_c/dx_2 \neq 0$, so z is not CI. □