Correctness Analysis and Power Optimization for Probabilistic Boolean Circuits

Ching-Yi Huang, Zheng-Shan Yu, Yung-Chun Hu, Tung-Chen Tsou, Chun-Yao Wang, *Member, IEEE*, and Yung-Chih Chen

***Abstract*—Traditionally, we expect that circuit designs can be executed without errors. However, for error resilient applications such as image processing, 100% correctness is not necessary. By pursuing less than 100% correctness, power consumption can be significantly reduced. Recently, probabilistic CMOS and probabilistic Boolean circuits (PBCs) have been proposed to deal with power consumption issue. However, to the best of our knowledge, no correctness analysis and power optimization algorithms have been proposed for PBCs. Thus, in this paper, we first propose a statistical approach for evaluating the correctness of PBCs. Then, we propose strategies for power optimization of PBCs. Finally, we integrate these strategies with the correctness analysis as a power optimization algorithm for PBCs. The experimental results show that the proposed correctness analysis method is highly efficient and accurate, and that the power optimization algorithm saves 36% of total power-delay-product on average under a correctness constraint of 90% on a set of International Workshop on Logic and Synthesis (IWLS) 2005 benchmarks.**

***Index Terms*—Analysis, logic synthesis, low-power design, power optimization, synthesis for low power.**

I. Introduction

DRIVEN by Moore's law, MOSFET devices have scaled into the nanometer regime. With faster clock frequencies, VLSI designs containing these devices encounter certain inevitable impediments such as high power consumption. Therefore, in recent years, reducing power consumption has become an important issue in VLSI designs.

To accommodate this power issue, at the system level, there has been a design paradigm shift from a single high performance processor to multiple moderate processors [22], [37]. At the device level, many new devices have been explored and proposed to reduce the required energy [21], [29].

Manuscript received May 28, 2014; revised November 13, 2014; accepted January 6, 2015. Date of publication January 21, 2015; date of current version March 17, 2015. This work was supported in part by the Ministry of Science and Technology of Taiwan under under Grant MOST 103-2221-E-007-125-MY3 and Grant MOST 103-2221-E-155-069, in part by the National Sciene Council of Taiwan under Grant NSC 102-2221-E-007-140-MY3, Grant NSC 102-2221-E-155-087, Grant NSC 101-2221-E-155-077, Grant NSC 101-2628-E-007-005, and Grant NSC 100-2628-E-007-031-MY3 and in part by the National Tsing Hua University under Grant NTHU 102N2726E1. This paper was recommended by Associate Editor L. Benini.

C.-Y. Huang, Z.-S. Yu, Y.-C. Hu, and C.-Y. Wang are with the Department of Computer Science, National Tsing Hua University, Hsinchu 30013, Taiwan (e-mail: s986516@m98.nthu.edu.tw).

T.-C. Tsou is with the Institute of Statistics, National Tsing Hua University, Hsinchu 30013, Taiwan.

Y.-C. Chen is with the Department of Computer Science and Engineering, Yuan Ze University, Chungli 32003, Taiwan.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCAD.2015.2394378

The miniaturization of transistors means that circuit designers must also deal with reliability issues. That is, noise, process variation, and other device perturbations can affect the behavior of transistors or even change logic gates from deterministic to probabilistic. To deal with this probabilistic behavior, an International Technology Roadmap for Semiconductors report [19] has predicted that a design paradigm shift, from deterministic to probabilistic, will be necessary. In other words, the future chip design will correlate with certain probabilistic behavior instead of correlating with specific metrics of performance.

Although the behavior in probabilistic designs might be erroneous, it is usable in error resilient applications. Error resilient applications are generally divided into two categories. The first one is algorithmic resilience to errors, such as in machine learning [3] applications. These applications do not require 100% precision in searching for or identifying objects from a large volume of data. The other category is perceptual resilience to errors, such as in audio and video applications. These applications tolerate errors that are imperceptible by a human being. For example, a peak-signal-to-noise-ratio that is greater than 30 dB is acceptable in lossy images and video compression [1].

Chakrapani *et al.* [8] and Palem *et al.* [28] revealed that if 100% correctness for the behavior of logic gates is not necessary, the energy requirement can be significantly reduced. Typically, the amount of energy reduction depends on the values of correctness probability p, as shown in Fig. 1 [12]. In Fig. 1, we can see that a lower p requires less energy. Thus, probabilistic Boolean circuits (PBCs), which contain probabilistic logic gates, are a possible alternative to achieving power minimization in designs [5], [9], [16], [23], [28].

References [6], [11], [12], and [23] demonstrated a method for designing CMOS logic gates with probabilistic behavior, named probabilistic CMOS or PCMOS, through voltage scaling with noise interference. By treating probabilistic behavior as a resource rather than an impediment, PCMOS approaches can realize low-power designs.

For a PBC, its output is changed from logic 1 or 0 to the probability of 1 or 0. Although the netlist in the PBC is still the same as the original netlist, its functionality is changed due to the probabilistic behavior of the gates. Therefore, when we take advantage of the PBCs' low-power feature, two issues have to be addressed.

1) We have to examine whether the circuits' behavior significantly varies, or even exceeds correctness constraints in the power optimization flow.
2) When designing PBCs for reducing power consumption with respect to a given correctness constraint, we

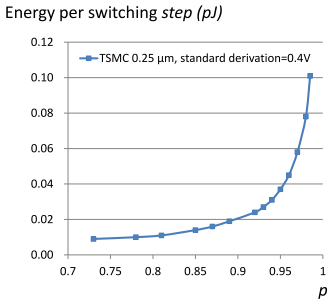


Fig. 1. Energy-probability model on a TSMC 0.25 μm inverter with standard deviation (SD) of noise 0.4 V [12].

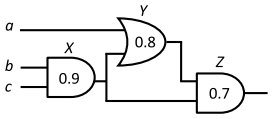


Fig. 2. Example of PBC with three probabilistic gates.

have to determine which gates should be replaced with probabilistic ones.

A. Correctness Analysis

In Boolean logic, evaluating the output values of a Boolean circuit under an input pattern is trivial due to determinism. However, computing the output probability of 1 in a PBC under an input pattern would be computation-intensive. This is because the time complexity of its truth value evaluation is exponential to the number of probabilistic gates in the circuit. For example, as shown in Fig. 2, to compute the probabilistic output value of a three-gate PBC, we have to consider $2^3 = 8$ combinations of correct and incorrect assumptions to these gates. Furthermore, to evaluate the correctness of the circuit, we have to consider all the input patterns. Thus, the time complexity is $O(2^{n+m})$ for acquiring the exact correctness of a PBC, where n and m are the number of primary inputs (PIs) and probabilistic gates, respectively.

Chakrapani [7] directly applied the probability values into logic gates using the probabilistic Boolean logic (PBL) formulas as shown in Fig. 3 for evaluating the output probability of PBCs. However, this approach is inaccurate when the circuits have many reconvergent-fan-out structures. This is because the formulas in the PBL assume that their input variables are signal-independent, while the inputs in the reconvergent-fan-out structures are signal-correlated. Hence, the PBL formulas are not applicable to PBCs that have many reconvergent-fan-out structures. Additionally, this approach still needs to explore 2^n input patterns to evaluate the output correctness of the whole circuit.

Thus, in this paper, we propose a correctness analysis method that is efficient, accurate, and scalable. The method exploits a statistical model for evaluating the correctness of PBCs without involving overwhelming computations [2].

B. Power Optimization

For power optimization, since a probabilistic gate can help save power, it is desirable to have a greater number of probabilistic gates assigned in the PBC under the correctness constraint. We observe that testability can be used to guide this assignment. This is because a node with low testability means that an error occurring on it cannot be observed at

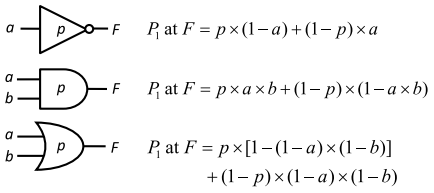


Fig. 3. Formulas for probabilistic gate simulation.

primary outputs (POs) easily. Therefore, a low testability node is a good candidate for having probabilistic behavior. However, traditional testability analysis methods are unsuitable for direct application due to their neglect of affected PO numbers. Thus, in this paper, we also propose a PO-aware testability-based replacement strategy to determine the replacement locations under the correctness constraint.

Additionally, given two connected gates, we observed that when the voltage domain of a driver-gate is far less than that of a driven-gate, the driven-gate consumes a large amount of leakage power. As a result, we also propose a level-up strategy for reducing leakage power if the design allows multiple voltage domains. Finally, we integrate the strategies and the correctness analysis method into a power optimization algorithm.

We conduct the experiments on a set of IWLS 2005 benchmarks [36]. The SPICE model is Predictive Technology Model (PTM) 45 nm [35]. We also utilize Synopsys HSPICE [39] and Synopsys Liberty NCX [40] to realize cell characterization, and use Synopsys Design Compiler (DC) [38] to compute the delay and power information of the synthesized PBCs. For the correctness analysis, the experimental results show that the proposed approach has an average speedup of more than two orders of magnitude compared to the previous approaches while having less than 0.0051 error. For power optimization, the experimental results show that the proposed power optimization algorithm gives a power-delay-product (PDP) saving of 36% on average under a correctness constraint of 90%.

II. Preliminaries

A. Related Works

Utilizing error resilience characteristic of applications to reduce the power consumption of a design has emerged as a new research direction in logic synthesis. There are two major classes in this research area. The first class focuses on functionally approximate circuit synthesis [18], [27], [30], [32], [33]. Gupta *et al.* [18] utilized approximate adders with reduced complexity to design low-power architecture. Miao *et al.* [27] and Shin and Gupta [30] proposed logic synthesis algorithms, which reduced the circuit area subject to different error metrics. Different from these works, SALSA [32] constructed a quality constraint circuit (QCC) to represent the target quality (error metric), and utilized the idea of approximation do not cares (ADCs) to simplify the approximate circuits based on the QCC structure. With a larger design paradigm, a novel substitute-and-simplify technique was proposed to synthesize approximate circuits, and also extended the technique to quality configurable circuits [33].

The other class is about PBC designs, which utilize PCMOS to reduce power consumption of designs [5], [9], [16], [23]. In [5] and [23], the concept of probabilistic system-on-a-chip (PSoC) architecture was

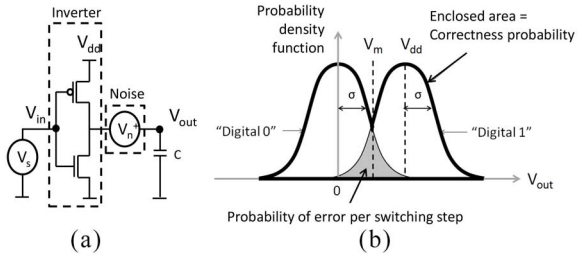


Fig. 4. (a) Inverter coupled with noise at its output. (b) Probability density function of (a).

INV

Voltage		1.1V	1.0V	0.9V	0.8V
Vm		0.474	0.446	0.418	0.390
Probability	SD=0.22V	0.991	0.986	0.979	0.965
	SD=0.20V	≈1	0.992	0.987	0.977
	SD=0.18V	≈1	≈1	0.993	0.987

(a)

NAND2

Voltage		1.1V	1.0V	0.9V	0.8V
Vm		0.509	0.465	0.422	0.378
Probability	SD=0.22V	0.993	0.988	0.979	0.965
	SD=0.20V	≈1	0.993	0.987	0.977
	SD=0.18V	≈1	≈1	0.993	0.986

(b)

Fig. 5. Voltage, V_m, and probability relationship with respect to SD of noise = 0.22 V $\sim$ 0.18 V. (a) Inverter gate (INV). (b) Two-input NAND gate (NAND2).

proposed to realize a low-power system. The proposed PSoC contains a host processor and a co-processor, where the co-processor is realized using PCMOS and plays a role of energy-performance accelerator to compute the probabilistic content of an algorithm or an application. References [9] and [16] then focused on the realization of probabilistic arithmetic blocks using PCMOS.

There are two major differences between approximate circuits and PBCs. From the viewpoint of the mechanism of power reduction, approximate circuits reduce the power consumption through circuit simplification. PBCs, however, reduce the power by replacing some conventional CMOS devices with low-power PCMOS ones. From the viewpoint of circuit behavior under input patterns, the approximate circuits behave the same as the original circuit under most of the input patterns, but behave differently under a few input patterns due to circuit approximation. In contrast, PBCs have a small probability to behave differently for every input pattern.

This paper belongs to the class of PBC designs. Although the power benefits of PBCs have been demonstrated in the prior works, those PBCs were only realized in specific architecture or small arithmetic blocks.

B. PCMOS

To achieve reliable devices, traditional CMOS technology adopts a high enough voltage to avoid noise interference at the output. However, as device size keeps shrinking, the impact of noise rises and the device would more likely have probabilistic behavior if the voltage is not high enough.

In recent years, with the design paradigm shift from deterministic to probabilistic, researchers have introduced a PCMOS technology to achieve low-power devices through scaling down the supply voltage and viewing noise as a resource on CMOS [12], [28]. The energy consumption and the behavior of the PCMOS are determined by the supply voltage and the noise model. Lowering supply voltage reduces the energy consumption but also decreases the correctness probability due to noise interference [12]. Therefore, designers determine the supply voltage of PCMOS according to the tradeoff between energy saving and correctness suffering.

Fig. 4(a) shows a PCMOS inverter that is modeled as an inverter coupled with noise at its output [12], [23]. The probability density function of this PCMOS is shown in Fig. 4(b) represented as Gaussian distribution. In Fig. 4(b), σ, V_{dd}, and V_m represent the SD of noise, supply voltage, and voltage that can distinguish 1 from 0, respectively. The shaded region in Fig. 4(b) stands for the probability of error per switching step. If we lower the V_{dd}, the probability of error per switching step will increase. The relationship between noise and voltage is important in PCMOS technology because it directly influences the correctness probability in PCMOS.

Cheemalavagu *et al.* [12] and Korkmaz *et al.* [23] concluded that the enclosed area of Fig. 4(b), which represents the correctness probability, can be expressed as

$$p = \frac{1}{4} + \frac{1}{4}\text{erf}\left(\frac{V_m}{\sqrt{2}\sigma}\right) + \frac{1}{4} + \frac{1}{4}\text{erf}\left(\frac{V_{dd} - V_m}{\sqrt{2}\sigma}\right) \quad (1)$$

where erf is the error function [31]. For an ideal symmetric inverter, i.e., $V_m = V_{dd}/2$, the energy consumption per switching step can be expressed as follows:

$$E = 4C\sigma^2\left[\text{erf}^{-1}(2p-1)\right]^2. \quad (2)$$

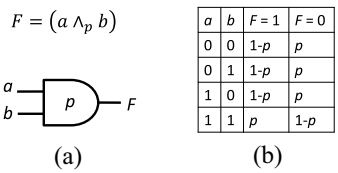
Equation (2) describes the relationship between the correctness probability p and the energy E given the load capacitance C of the inverter and the SD of noise σ. Fig. 1 also shows this relationship, in which the required energy is drastically reduced when lowering the correctness probability in PCMOS [12], [23]. Although this amount of reduction is only for dynamic energy and does not consider leakage energy, it still highlights the potential for minimizing energy by using probabilistic gates in a design.

However, in practice, we usually determine the voltages, but not the probabilities for the gates in a design. Thus, according to the cell property of PTM 45 nm SPICE model and (1), we can derive the V_m and probabilities of gates under different supply voltages and SDs of the noise. The results for an inverter and a NAND2, are shown in Fig. 5, where 1.1 V is the nominal voltage for this technology [35]. For example, the V_m of NAND2 with supply voltage 1.0 V is 0.465, and the probability of this NAND2 with 0.20 V SD of noise is 0.993.

References [9] and [16] showed that multiple supply voltages are allowed in a circuit. The available voltage domains and possible SDs of noise, which are user-defined parameters in the proposed approaches, influence the probabilities of gates. For ease of discussion, in this paper, we assume that four voltage domains are available: 1) 1.1 V; 2) 1.0 V; 3) 0.9 V; and 4) 0.8 V. We also assume that the SD of noise is 0.20 V, which corresponds to proper probabilities among different voltage domains of gates, as shown in Fig. 5.

C. PBL

PBL is a logic that focuses on studying the behavior of the primitive operations, e.g., AND, with a correctness probability p, which indicates the probability of evaluating the correct



a	b	F = 1	F = 0
0	0	1-p	p
0	1	1-p	p
1	0	1-p	p
1	1	p	1-p

Fig. 6. (a) Probabilistic formula and the corresponding probabilistic gate. (b) Truth table of the probabilistic gate of (a).

value of this operation [6]. A conventional logic operation can be regarded as a probabilistic operation with $p = 1$. Fig. 6(a) shows an example of a probabilistic formula and its probabilistic gate. This formula is an AND operation with a probability parameter p, denoted as $\wedge_p$. Fig. 6(b) lists the probability of the truth values of the formula under each input pattern. Since p is the probability of evaluating the correct value, the output F of the AND gate has a probability p of 1 under the input pattern $ab = 11$ and has a probability $(1 - p)$ of 1 under the other patterns.

D. Definitions of Correctness

We first define the correctness of an output under an input pattern. If the golden output value under an input pattern i is 1, the correctness c_i is defined as

$$c_i = P_1 \times 100\% \tag{3}$$

where P_1 represents the output probability of 1 under the input pattern. If the golden output value under an input pattern is 0, the correctness is defined as follows:

$$c_i = (1 - P_1) \times 100\%. \tag{4}$$

The correctness for a PO j under all input patterns, Corr_j, is defined as

$$\text{Corr}_j = \frac{1}{2^{|\text{PI}|}} \sum_{i=1}^{2^{|\text{PI}|}} c_i \tag{5}$$

which is the average correctness among all input patterns.

In this paper, we discuss two correctness constraints. One is the minimum correctness ($C_{\min}$) constraint, defined as

$$C_{\min} = \min\{\text{Corr}_i, i = 1, 2, \ldots, |\text{PO}|\} \tag{6}$$

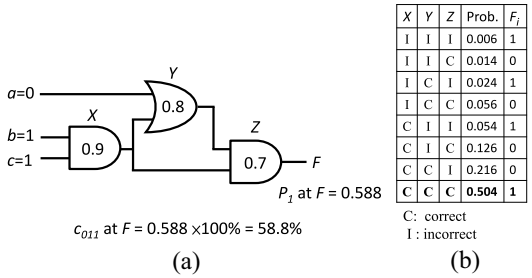
where min represents the minimum operator, and |PO| represents the total number of POs. The other one is the average correctness (C_{avg}) constraint, defined as follows:

$$C_{\text{avg}} = \frac{\sum_{i=1}^{|\text{PO}|} \text{Corr}_i}{|\text{PO}|}. \tag{7}$$

III. Correctness Analysis

In this section, we first review the naive approaches, and then present our statistical approach to evaluating the correctness of PBCs.

Problem Formulation 1: Given a PBC, report $C_{\min}$ and C_{avg} of the PBC.



X	Y	Z	Prob.	F_i
I	I	I	0.006	1
I	I	C	0.014	0
I	C	I	0.024	1
I	C	C	0.056	0
C	I	I	0.054	1
C	I	C	0.126	0
C	C	I	0.216	0
C	**C**	**C**	**0.504**	**1**

Fig. 7. Example for the exact method under input $abc = 011$. (a) PBC. (b) Truth table of (a).

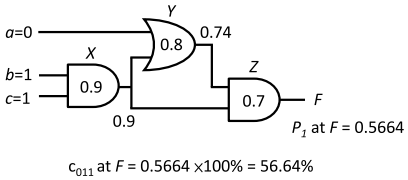


Fig. 8. Example for formula-based method.

A. Naive Approaches

In this subsection. we introduce the exact method and the formula-based method to evaluate the output correctness.

1) Exact Method: For an input pattern, the exact method first considers the combinations of correct and incorrect conditions to all the gates, and calculates the probability of each combination. Then, the input pattern is coupled with the corresponding combination. For example, in Fig. 7(a), we evaluate the correctness of a PBC at the output F under $abc = 011$. It first calculates the probability of each combination of correctness to gates, as shown in Fig. 7(b), e.g., the probability that all gates are correct is $0.9 \times 0.8 \times 0.7 = 0.504$. After coupling the circuit with $abc = 011$, we can obtain the truth value of F under each combination, represented as F_i for the combination i. For example, $F_{CCC} = 1$ when all gates are correct (C); $F_{CCI} = 0$ when X, Y are correct (C), but Z is incorrect (I). Then, the probability of 1 at F under $abc = 011$ can be evaluated as $\sum_{i=1}^{2^{|\text{gate}|}} \text{Prob}_i \times F_i = 0.588$. By (3), c_{011} is $0.588 \times 100\% = 58.8\%$.

Using this method, the correctness at the output under one input pattern can be obtained. However, the method is impractical when the number of probabilistic gates is enormous. To evaluate the output correctness under all input patterns, the method has to be repeated for $2^{|\text{PI}|}$ runs exhaustively. Therefore, this method requires exponential complexity in computation.

2) Formula-Based Method: Another method for evaluating the correctness is to apply logic values to the PBCs directly under an input pattern. Fig. 3 lists the formulas for the probabilistic gates. For example, if $p = 0.9$ and the input $a = 1$, the output probability of 1 for an INV, i.e., P_1 at F of the INV, is $0.9 \times (1 - 1) + (1 - 0.9) \times 1 = 0.1$. Then, by (4), the output correctness c_{011} is $(1 - 0.1) \times 100\% = 90\%$.

In Fig. 8, we can obtain the output correctness c_{011} according to the probabilistic formulas. However, this c_{011} is not equal to that in Fig. 7(a). This is because the probabilistic formulas assume that the fan-in signals are independent, and they fail to deal with the circuits that have reconvergent-fan-outs [13], [24]. For example in Fig. 8, the

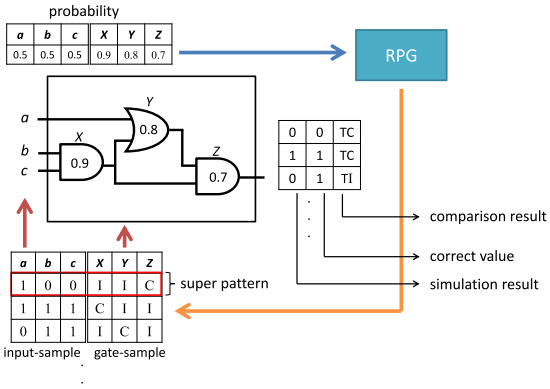


Fig. 9. RPG scheme of the proposed approach.

fan-in signals of gate Z are not independent since they can be traced back to the common gate X. The exact method does not have this issue because it uses logic simulation rather than arithmetic calculation. In summary, the formula-based method has a fast calculation process due to one pass calculation. However, inaccuracy is its major drawback, and this inaccuracy cannot be controlled.

Like the exact method, when considering all input patterns, this method can be repeated for $2^{|\mathrm{PI}|}$ runs exhaustively for getting the Corr_j of the output j. Obviously, the complexity also grows exponentially.

As a result, to tackle the practicability and accuracy issues simultaneously, we propose a statistical approach, which can deal with large general PBCs.

B. Proposed Approach

In practice, it is not necessary to compute the correctness exactly. An estimate suffices if the estimation error can be bounded. That is, if we can confine the error within an error resilient range, the approximate results are still acceptable. Thus, we propose a statistical approach—Monte Carlo approach, which both overcomes the problem of high time complexity, and also has controllable accuracy. Although the Monte Carlo method is not a novel approach, and has been used in many different studies, this paper customizes this approach to our problem and justifies its effectiveness.

The Monte Carlo approach consists of four parts, which will be elaborated in the following subsections.

1) Random Pattern Generation (RPG): RPG is performed to generate samples for simulation. Here, we generate a "super pattern," consisting of one input-sample and one gate-sample. An input-sample is composed of "1" and "0" logic values assigned in the PIs. A gate-sample is composed of "C" and "I," where C represents a probabilistic gate performed correctly and I represents one performed incorrectly. Thus, we can use logic simulation to obtain the sampling result of a super pattern. During simulation, the RPG in Fig. 9 will generate a super pattern according to the probability distributions of PIs and gates where the PI probability is determined according to the given input statistics (IS), and the gate probability is determined by designers. In this paper, for simplification, the probability for each PI is assigned as 0.5. Note that the probability distributions of PIs can be extended to any application specific IS.

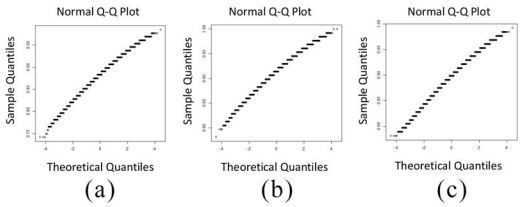


Fig. 10. *q-q* plots of some circuits with different percentages of probabilistic gates in different supply voltages. (a) apex7_50%_0.8 V. (b) C1355_75%_0.8 V. (c) C1355_100%_0.9 V.

2) Sampling Rule: Each sampling will return a result according to the sampling rule. In this problem, we first obtain the output of logic simulation according to the input pattern and the gate behavior. If the simulation result is the same as the correct value, the comparison result is "totally correct" (TC); otherwise, it is "totally incorrect" (TI). Fig. 9 shows examples of judging TC or TI. After a random pattern simulation trial, we count the number of TCs at the POs, and the correctness of the jth PO for the ith sampling, $\mathrm{SampleCorr}_j^{(i)}$, is calculated as

$$\mathrm{SampleCorr}_j^{(i)} = \frac{|\mathrm{TC}|}{|\mathrm{ParaPatt}|} \times 100\% \tag{8}$$

where |TC| is the number of TC at the jth PO, and |ParaPatt| is the number of random parallel patterns for collecting a sampling result. In this paper, 128-bit parallel simulation is adopted, i.e., |ParaPatt| = 128. The comparison result, i.e., TC or TI, is a binary random variable for a $\mathrm{SampleCorr}_j^{(i)}$.

3) Scoring: The next important issue that we have to deal with is calculating the sufficient number of samplings for obtaining an accurate enough result of Corr_i, which relates to the applied statistical model. Note that in the calculation of Corr_i, $\mathrm{SampleCorr}_j^{(i)}$ is a real-valued random variable and $\in [0\%, 100\%]$. In (6) and (7), we assume that the sampling distribution is normal since it utilizes the linearity property. To validate this, a statistical plot, *q-q* plot [25], is used to draw figures for some circuits. From statistics, the sampling data will form the normal-distribution if and only if the ith quantile of the sampling data and the ith quantile of the standard normal-distribution have a linear relationship [25]. Hence, we plot the figure that shows the quantiles of the sampling data and the standard normal-distribution, called normal *q-q* plot, and see if the result behaves like a straight line.

Fig. 10(a)–(c) show the results of *q-q* plotting for some circuits. Since the data behave like a straight line, the samples generated by our approach are considered normal-distribution.

In our approach, t-distribution [15], [20], [26] statistical model is adopted to deal with the error estimation. t-distribution is used for estimating the mean of a normally distributed population in situations where the sample size is small and the SD is unknown. If the number of samplings is sufficient, applying the t-distribution model will produce an approximate but fairly accurate result from averaging the sampling results. Corr_j is approximated as

$$\widehat{\mathrm{Corr}_j} = \frac{\sum_{i=1}^{|\mathrm{Sampling}|} \mathrm{SampleCorr}_j^{(i)}}{|\mathrm{Sampling}|} \tag{9}$$

where |Sampling| is the total number of samplings.

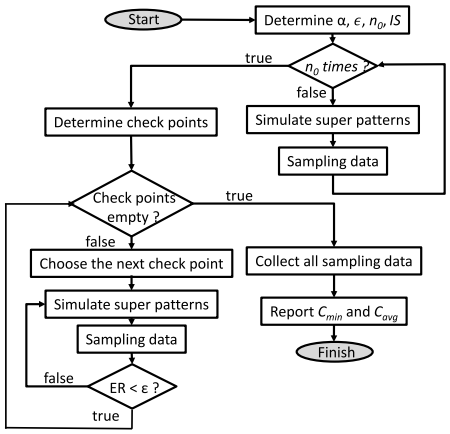


Fig. 11. Flow of the proposed approach.

4) Error Estimation: We also use the confidence interval in statistics to estimate the sampling error (ER) of the correctness. The confidence interval is a statistical measure indicating the percentage of the expected results that falls within a specified range. Assume that we perform N samplings for a PO. We can compute the sample SD, denoted as SD, of these samplings. Given the confidence level α of the predefined confidence interval, $(1-\alpha)\times 100\%$ is the confidence interval. Then we look the value $t_{\alpha/2}$ up in the t-distribution table with $(N-1)$ degrees of freedom.

Thus, the ER of the output correctness is expressed as

$$\mathrm{ER} = \frac{t_{\frac{\alpha}{2}} \times \mathrm{SD}}{\sqrt{N}}. \tag{10}$$

In (10) [26], ER decreases as the sampling number N increases. Thus, given a desired ER ϵ, we keep sampling until the termination condition is reached. The termination condition is expressed as follows:

$$\frac{t_{\frac{\alpha}{2}} \times \mathrm{SD}}{\sqrt{N}} < \epsilon. \tag{11}$$

Checking whether the correctnesses of all the POs meet the termination condition is time-consuming. Hence, we heuristically select $\lfloor 5 + 0.1 \times |\mathrm{PO}| \rfloor$ POs whose SDs are the highest among all the POs as the check points after simulating n_0 initial patterns. When the selected check points satisfy (11), we terminate the simulation.

5) Overall Flow: Since the proposed approach uses confidence level α, ER ϵ, and n_0 to control the accuracy and efficiency, and uses IS to determine the probability distribution for PIs, the formulation of correctness analysis in PBCs is slightly modified as follows.

Problem Formulation 2: Given a PBC with IS, confidence level α, ER ϵ, and n_0, report $C_{\min}$ and C_{avg} of the PBC.

The flow of the proposed approach is shown in Fig. 11. Given confidence level α, ER ϵ, the number of initial patterns n_0, and IS, the approach generates super patterns for simulating the given PBC. After simulating n_0 super patterns, the check points are determined. If a check point has met the termination condition, we proceed to the next one. When all the check points have reached the termination condition, we collect the results and report $C_{\min}$ and C_{avg} of the PBC.

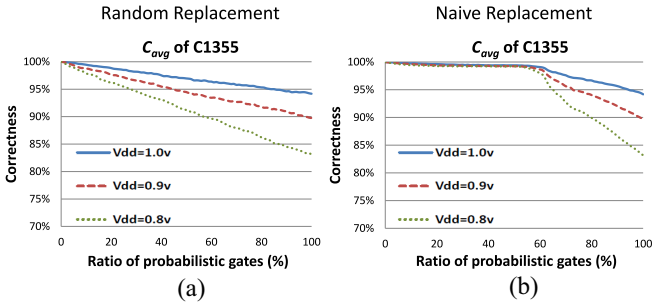


Fig. 12. Variation of C_{avg} with respect to the ratio of probabilistic gates. (a) Random and (b) naive replacement.

IV. Power Optimization

The major motivation of adopting PBC as a design alternative is the advantage of low-power consumption in operation. Therefore, in this section, we discuss the power consumption issue of PBCs and propose algorithms for power optimization under the correctness constraint.

The amount of power reduction in PBCs strongly depends on both the number and the probability values of probabilistic gates in the circuits. Intuitively, replacing more gates with probabilistic ones and adopting lower probability values reduce more power. However, the correctness constraint has to be taken into account when minimizing power consumption. Additionally, the locations of probabilistic gates significantly influence the number of probabilistic gates that can be assigned. Thus, an approach for determining the locations of probabilistic gates in PBCs is crucial.

For ease of discussion, we first assume that a library of probabilistic gates with the same probability value are available to realize a PBC. Note that a probability value represents a corresponding supply voltage in a probabilistic gate. Thus, two voltage domains are used in this PBC, one for general logic gates, and the other for probabilistic gates. The problem formulation of power optimization in PBCs is as follows.

Problem Formulation 3: Given a Boolean circuit and a library of probabilistic gates with the same probability value, determining the locations of probabilistic gates such that the power consumption of the corresponding PBC is minimized under the $C_{\min}$ and C_{avg} constraints.

A. Naive Replacement Strategy

As mentioned, to achieve power minimization in PBCs, more gates replaced by probabilistic gates are desired. A naive approach is to replace the gates according to their levels, from the PIs to the POs of the circuit. We call this approach a naive replacement strategy. The intuition behind this strategy is that the error effect caused by these probabilistic gates that are located near the PIs would be relatively hard-to-detect at the POs. This is because these error effects are likely blocked during the long propagation to the POs. Hence, the correctness of the whole PBC would be less suffered when applying this approach. The advantage of this strategy is its efficiency.

Fig. 12 shows the C_{avg} of some benchmarks with respect to the ratio of assigned probabilistic gates using different replacement strategies. For each benchmark, the experiments were conducted under three voltage domains, i.e., 1.0 V, 0.9 V, and 0.8 V, for probabilistic gates. Fig. 12(a) shows the results from

using a random replacement strategy, and Fig. 12(b) shows the results from using the naive replacement strategy. From Fig. 12, we can see that the correctness at the beginning (0% of probabilistic gates) and at the end (100% of probabilistic gates) are the same for each voltage domain from using these two strategies. However, with the random strategy, Fig. 12(a) shows that correctness consistently decreases when the ratio of probabilistic gates in a PBC increases. On the other hand, with the naive replacement strategy, Fig. 12(b) shows that the correctness remains stable when the first 50% of gates are replaced with probabilistic gates. Thus, when a correctness constraint is specified, say 95%, the naive replacement strategy can accommodate more probabilistic gates and, accordingly, save more power.

B. PO-Aware Testability-Based Replacement Strategy

Testability of a node in a Boolean circuit measures the difficulty of simultaneously setting a value to a node from the PIs and observing the node value change in the POs [4], [10], [14], [17]. In PBCs, since we can consider the erroneous behavior caused by a probabilistic gate as a fault, the testability of nodes can be used as a guide to determine the locations of probabilistic gates. That is, the gates with lower testability are preferred for probabilistic gate replacement. This observation is similar to the observation mentioned in SALSA [32]. The difference between SALSA and our approach is that SALSA simplifies the approximate circuit by the aid of ADCs based on the QCC structure, and our approach adopts a greedy algorithm to determine the locations of probabilistic gate replacement directly on the circuit. We call this approach the testability-based replacement strategy. We reimplement a testability algorithm proposed in [14], and use it as a guide to determine locations of the probabilistic gates.

In the conventional testability analysis just discussed, however, designers do not care about the number of POs influenced. Designers only care about whether the fault effect can be propagated to the POs. In fact, from the viewpoint of correctness, when the number of influenced POs is larger, the correctness suffers more. Thus, we propose a PO-aware testability-based replacement strategy, which takes the number of influenced POs into account when analyzing testability. We skip the detailed algorithm here since it is an extension of conventional testability analysis [14], [17]. In brief, we extend the number of criticality vectors at each signal wire to the number of POs, i.e., each criticality vector records the number of detectable faults corresponding to a certain PO, in the proposed approach. The experimental results of this strategy will be shown in Section V-B.

Fig. 13 shows the synthesis flow of PBCs, which combines the PO-aware testability-based replacement strategy with the correctness analysis in Section III-B under two voltage domains. In the flow, given a Boolean logic circuit with IS, a voltage domain for probabilistic gates V_p, the SD of noise σ, $C_{\min}$ or C_{avg} constraint *corr_con*, confidence level α, ER ϵ, and n_0, we first calculate the PO-aware testability of each gate. After sorting the gates according to their PO-aware testability values in an ascending order, we replace the logic gates with probabilistic ones in this order. When replacing, we use binary search to determine the number of probabilistic gates that can be assigned under the correctness constraint. After creating the PBC, we utilize DC [38] with a recharacterized cell library to report timing information of the synthesized PBC. The timing information then is used to compute the power information by using DC. Finally, we compare the power and timing of the derived PBC to that of the original circuit.

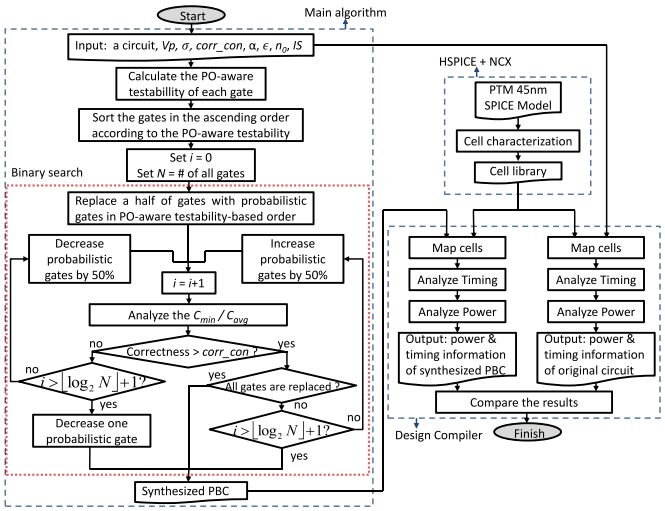


Fig. 13. Flow chart of the proposed PO-aware testability-based replacement strategy.

Since lowering the voltage of a gate will increase gate delay, we also developed a simple timing-aware replacement method based on the PO-aware testability-based replacement strategy to minimize the delay in PBCs for timing-critical designs. This timing-aware replacement method was modified from the original algorithm by checking whether the circuit delay of the PBC became worse after the replacement. If so, we restore the replacement and select the next gate in the replacement order. For simplification, we adopt an approximate timing analysis. The circuit delay was analyzed by topologically computing the arrival time of the gates. The gate delay was obtained from a look-up table that considers the output loading and voltage. The final circuit delay is calculated by DC.

C. Level-Up Replacement Strategy

We observe that if the voltage domain of a driver-gate is smaller than the driven-gate, the driven-gate will suffer larger leakage power though the driver-gate saves some dynamic power. For example, Fig. 14(a) and (b) shows an inverter chain and power report under a single voltage domain. Fig. 14(c) shows a PBC that replaces gate $g1$ in Fig. 14(a) with a probabilistic gate using 0.8 V voltage. According to HSPICE simulation, the dynamic power of gate $g1$ under the frequency of 500 MHz is reduced from 1.6010 to 0.8977 u watts. However, the leakage power of gate $g2$ is increased from 0.0005 to 2.0686 u watts. Based on this observation, the increased leakage power may compromise total power reduction. In this example, the total power is even increased. Thus, replacing a gate with a lower-voltage gate does not always save power.

In fact, since the locations of probabilistic gates are determined according to the PO-aware testability-based order, the voltage domains of two connected gates are frequently different. Thus, two experiments are conducted to observe the leakage power suffered by a driven-gate when two connected gates have different voltage domains.

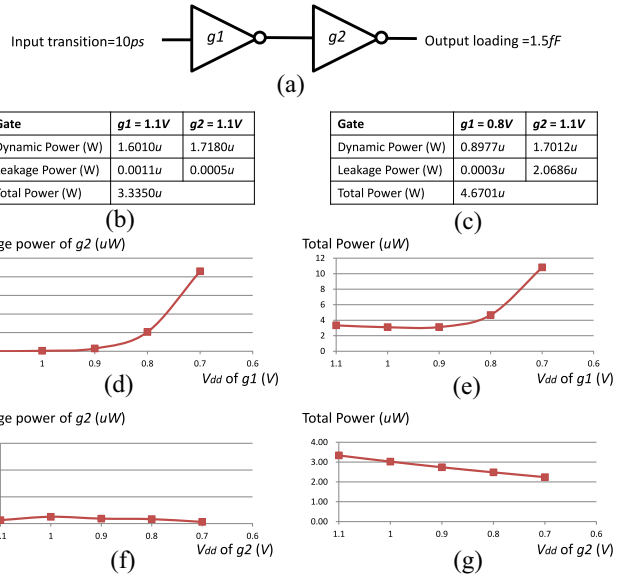


Fig. 14. (a) Inverter chain. (b) Power report of (a) under a voltage domain. (c) Power report of (a) after replacing $g1$ with a probabilistic gate. The power profile of (a) with respect to different voltage domains of a driver-gate. (d) Leakage power profile in $g2$. (e) Total power profile. The power profile of (a) with respect to different voltage domains of a driven-gate. (f) Leakage power profile in $g2$. (g) Total power profile.

Fig. 14(d) and (e) shows the power profiles of Fig. 14(a), in which the voltage domain of $g2$ stays at 1.1 V, with respect to different voltage domains of the driver-gate $g1$ under the frequency of 500 MHz. Fig. 14(d) and (e) shows the leakage power profile in $g2$ and the total power profile of this inverter chain, respectively, with respect to different voltage domains in $g1$. We can see that when the voltage domain of $g1$ is decreased to around 0.9 V, the leakage power of $g2$ starts to increase, and total power consumption also increases. We suggest that the voltage difference between a lower-voltage driver-gate and a higher-voltage driven-gate should be less than or equal to 0.1 V to minimize the leakage power of the driven-gate.

On the other hand, Fig. 14(f) and (g) shows the power profiles of Fig. 14(a), in which the voltage domain of $g1$ stays at 1.1 V, with respect to different voltage domains of the driven-gate $g2$ under the frequency of 500 MHz. Fig. 14(f) and (g) show the leakage power profile in $g2$ and the total power profile of this inverter chain, respectively, with respect to different voltage domains in $g2$. We can see that although the voltage domain change in $g2$ influences the leakage power of $g2$ slightly, it does not increase the total power. Therefore, we do not need to deal with the voltage difference between a higher-voltage driver-gate and a lower-voltage driven-gate.

We also conducted experiments on PTM 32 nm, 65 nm, and 90 nm SPICE models, and obtained the similar observation. Based on this observation, we propose a level-up replacement strategy to make the power reduction possible if more than two voltage domains are available in PBCs. The main idea of the level-up replacement strategy is to maximize dynamic power saving with the PO-aware testability-based strategy while minimizing leakage power by decreasing the voltage difference between two connected gates where the driver-gate is in a lower voltage domain. This circuit structure is called the "level-up" structure.

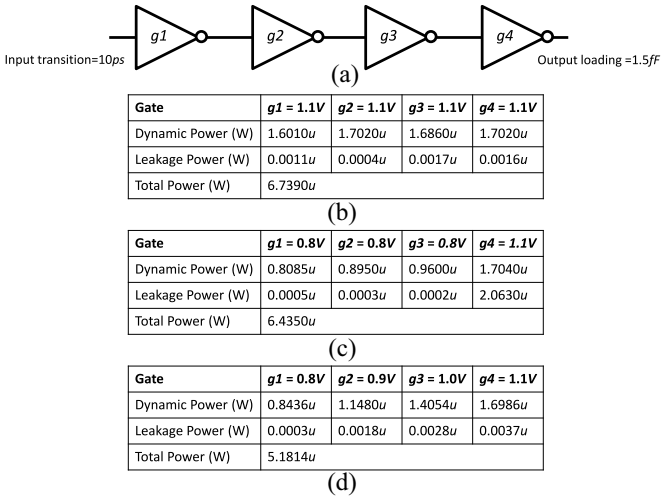


Fig. 15. (a) Four-gate inverter chain. (b) Power report of the inverter chain with $V_{dd} = 1.1$ V. (c) Power report after the replacement with 0.8 V probabilistic gates. (d) Power report after the level-up replacement.

We use an example to introduce the level-up structure. In Fig. 15, the available voltage domains are {0.8 V, 0.9 V, 1.0 V, 1.1 V (nominal voltage)}, and the frequency is 500 MHz. Fig. 15(a) and (b) shows an inverter chain and power information with $V_{dd} = 1.1$ V. In Fig. 15(c), the inverter chain is composed of three probabilistic inverters with $V_{dd} = 0.8$ V followed by a normal inverter $g4$ with $V_{dd} = 1.1$ V. Although the dynamic power and leakage power of $g1 \sim g3$ are significantly reduced by probabilistic gate replacement, the leakage power of $g4$ is drastically increased from 0.0016 to 2.0630 u watts according to the HSPICE simulation. However, if we modify the inverter chain with a level-up structure such as the one shown in Fig. 15(d), i.e., 0.8 V $\rightarrow$ 0.9 V $\rightarrow$ 1.0 V $\rightarrow$ 1.1 V, its total power is less than that of Fig. 15(c).

Given a correctness constraint and a set of available voltage domains, it is not straightforward to assign the lowest voltage to probabilistic gates for maximally reducing dynamic power. This is because using the lowest voltage domain for probabilistic gates might decrease the number of probabilistic gates that can be assigned in a circuit within a correctness constraint. Therefore, prior to embedding the level-up structures into the circuit, we first conduct the flow shown in Fig. 13 for each available voltage domain. For these voltage domains, we select the one that has the smallest switching energy consumption. The switching energy is calculated as (dynamic power $\times$ circuit delay). Rather than dynamic power, the switching energy is used in the measurement since the dynamic power is affected by the frequency, which is determined by the circuit delay. We name this selected voltage domain the desired voltage domain of the circuit. Next, we replace the logic gates with the probabilistic gates using the desired voltage domain according to the PO-aware testability-based order. When replacing a gate, we also change the voltage domains of the gates in the fan-out cone of the probabilistic gates using higher voltage domains to construct the level-up structures.

We define two terms, voltage level and voltage level distance, for explaining the level-up structure construction.

Definition 1: Given a set of available voltage domains sorted in the ascending order $\{v_1, v_2, \ldots, v_i, v_{i+1}, \ldots, v_{\text{nom}}\}$, where v_1 is the lowest voltage and v_{nom} is the largest voltage or

Level_Up_Replace (Circuit C, Target_Gate g_t, Desired_Voltage_Domain v_i, Sorted_Voltage_Domain_List $V = \{v_1, v_2, ..., v_i, ..., v_{nom}\}$)
1. $C^* \leftarrow C$.
2. $g_s \leftarrow$ The probabilistic gate version of g_t with v_i.
3. $C^* \leftarrow$ Replace g_t with g_s.
For each gate $fanout_i$ in the fan-out cone of g_s
If voltage_level_distance(g_s, $fanout_i$) > 1
$C^* \leftarrow$ **Recursive_Replace**(C^*, V, v_{i+1}, $fanout_i$)
Return C^*

Fig. 16. Algorithm for constructing the level-up structures.

Recursive_Replace (Circuit C^*, Sorted_Voltage_Domain_List V, Voltage_Domain v_k, Target_Gate g_t)
If $v_k \neq v_{nom}$
(1) $g_s \leftarrow$ The probabilistic gate version of g_t with v_k.
(2) $C^* \leftarrow$ Replace g_t with g_s.
For each gate $fanout_i$ in the fan-out cone of g_s
If voltage_level_distance(g_s, $fanout_i$) > 1
$C^* \leftarrow$ **Recursive_Replace**(C^*, V, v_{k+1}, $fanout_i$)
Return C^*

Fig. 17. Subfunction of Recursive_Replace() in the algorithm for constructing the level-up structures.

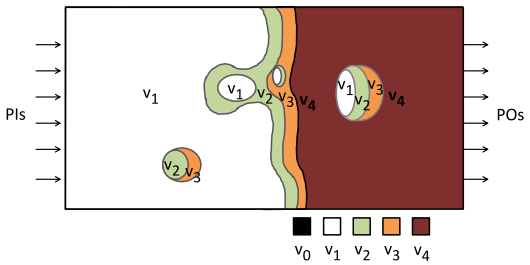


Fig. 18. Possible voltage domain distribution of a PBC with level-up structures, where the available voltage domains are $v_0 < v_1$ (desired voltage domain) $< v_2 < v_3 < v_4$ (nominal voltage domain).

the nominal voltage, the voltage level of a gate with the voltage domain v_i is i. The voltage level distance between gate g_a and gate g_b (voltage_level_distance (g_a, g_b)) is $(j - i)$, where i and j are the voltage levels of gate g_a and g_b, respectively.

Figs. 16 and 17 show the algorithms for constructing the level-up structures. After a gate g_t is replaced with a probabilistic gate with the desired voltage domain v_i, we check whether the voltage level distance from g_t to its fan-out gate fanout$_i$ exceeds 1. If so, we replace the fan-out gate fanout$_i$ with a probabilistic gate with $v_k = v_{i+1}$. Then, we recursively check and replace the fan-out gates of fanout$_i$ until v_k reaches v_{nom}. This algorithm can ensure that the voltage difference is minimized between two connected gates where the driver-gate is in a lower voltage domain.

Fig. 18 shows a figure that depicts a possible voltage domain distribution resulting from the algorithms in Figs. 16 and 17. In Fig. 18, we assume that the available voltage domains are $v_0 < v_1 < v_2 < v_3 < v_4$ and also assume that the desired voltage domain of the circuit is v_1 and the nominal voltage domain is v_4. We can see that v_0 does not appear in Fig. 18 even though it is the lowest voltage domain. This is because assigning v_1 causes the smallest switching energy consumption. Also, we can see many level-up structures in the circuit, e.g., $v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow v_4$ between v_1 and v_4. However, the high to low situations such as $v_4 \rightarrow v_1$ or $v_3 \rightarrow v_1$ are still allowed in our approach since they do not affect power consumption significantly, as shown in Fig. 14(f) and (g).

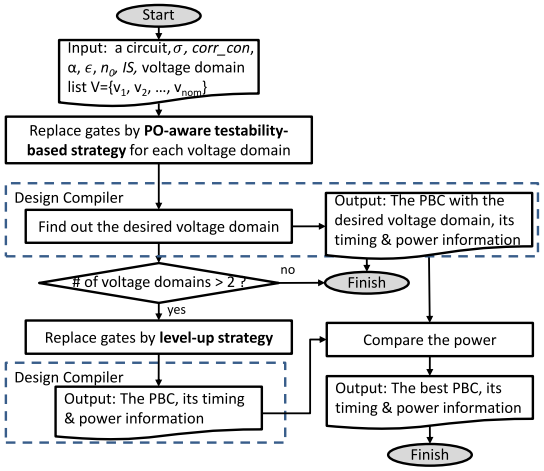


Fig. 19. Overall flow for power optimization of PBCs.

Note that the optimization flow of level-up replacement strategy is almost the same as the flow in Fig. 13. The only difference is that the level-up replacement algorithms in Figs. 16 and 17 are applied after a gate is replaced with a probabilistic gate in the PO-aware testability-based order. As a result, the synthesized PBC is still ensured to meet the correctness constraint based on the flow.

D. Overall Flow of Power Optimization

Although the level-up replacement strategy can reduce the leakage power suffered by a circuit, it saves less dynamic power than the PO-aware testability-based replacement with two voltage domains saves. Therefore, it is difficult to predict which approach saves much more total power. As a result, in the power optimization flow, we calculate the power information of PBCs from these two strategies, and report the PBC that has a smaller power consumption.

The overall flow for power optimization is shown in Fig. 19. We first determine the desired voltage domain by running the PO-aware testability-based replacement flow shown in Fig. 13 for each available voltage domain except for the nominal voltage domain. If only two voltage domains are available, the desired voltage domain is definitely the lower one because the higher one is the nominal voltage domain. In this case, we directly output the PBC and corresponding information as the final result. If more than two voltage domains are available, we keep the results from the PO-aware testability-based strategy with the desired voltage domain and the nominal voltage domain. Next, the level-up replacement is conducted based on the desired voltage domain. Finally, we compare the total power of the PBC by the level-up replacement strategy to that by the PO-aware testability-based strategy with two voltage domains, and then output the best PBC and corresponding power information.

V. Experimental Results

We implemented the proposed algorithms in C++ language. The experiments were conducted over a set of IWLS 2005 benchmarks [36] on a 3.0 GHz Linux platform. These benchmarks were originally in *.blif* format, and they were transformed into the circuits that consisted only of two primitive

TABLE I
CORRECTNESS DIFFERENCE (CD) BETWEEN THE EXACT AND OUR APPROACH

|Patt.|	Maximum correctness differences (CDs) among all POs		
	f51m (247 gates)	alu4 (2015 gates)	t481 (4336 gates)
128	0.028125	0.063282	0.033659
1280	0.010938	0.010938	0.011003
12800	0.005798	0.006302	0.000721
128000	0.003436	0.001219	0.000638
1280000	0.000627	0.000763	0.000240
12800000	0.000159	0.000191	0.000062

gates—NAND2 and INV. This is because cell recharacterization efforts will increase if more cells are available in the library. In this paper, we only recharacterized these two gates for demonstrating a preliminary implementation. The used SPICE model was from PTM 45 nm model [35], and only size 1 cells, i.e., INV_*X*1 and NAND*2*_*X*1, were considered in the experiments. In the experiments, we assumed that four voltage domains {0.8 V, 0.9 V, 1.0 V, 1.1 V} are available, where 1.1 V is the nominal voltage domain for this technology [35]. Since having multiple voltage domains, cells with different combinations of power domains in one cell itself and its fan-in cells were recharacterized using Synopsys Liberty NCX [40] and HSPICE simulator [39]. With the four available voltage domains for an INV gate, we recharacterized $4 \times 4 = 16$ INV cells with respect to different fan-in cell voltage domains. Similarly, for a NAND2 gate, since it has two fan-ins, $4 \times 4 \times 4 = 64$ NAND2 cells were recharacterized. Finally, Synopsys DC [38] was used to replace the gates in PBCs with the recharacterized cells and report the power and timing information.

Three sets of experimental results are shown in this section. The first subsection shows the efficiency and accuracy of the proposed correctness analysis method. The second one compares the switching energy among different replacement strategies for probabilistic gates under two voltage domains. The final subsection shows the effectiveness of the level-up replacement strategy.

A. Correctness Analysis

In this subsection, two experiments were conducted. The first one shows the accuracy of Monte Carlo simulation when the number of random patterns increases from 128 to 12 800 000. Since a 128-bit parallel simulator was used in our Monte Carlo simulation, we set the number of random patterns as a multiple of 128. In the first experiment, we set the parameters of $\epsilon = 0.01$ and $\alpha = 0.001$. The initial sampling number n_0 is 6400, and the voltage of probabilistic gates was set to 0.8 V. The percentage of probabilistic gates was 25% of the total gate count of a circuit such that the golden result of the exact approach can be obtained under this setting within a time limit.

Table I summarizes the results of the first experiment. Column 1 lists the number of random patterns in the Monte Carlo simulation. Columns 2-4 list the maximum CD among all the POs compared to the golden result under different random patterns for three selected circuits.

According to Table I, we can find that the Monte Carlo simulation is more accurate when the number of random patterns is large enough. Since the exact method costs enormous CPU time to compute the golden result, for experimental purposes, we considered the result of the Monte Carlo simulation with 12 800 000 random patterns as the "golden result" in the succeeding experiments.

The second experiment shows the CD and the CPU time of different approaches when considering all input patterns. Since the computation time of the formula-based method grows exponentially with the number of PIs, exhaustive simulation is applied when the number of PIs is less than or equal to 21. Otherwise, we simulate $2^{21} = 2\,097\,152$ patterns to approximate the result.

Table II summarizes the results of the second experiment. Columns 1-3 list the circuit information. Column 4 lists the CPU time for calculating the "golden result." Columns 5–7 list the CPU time, its maximum CD among all the POs, and the pattern number in the approximate formula-based method. Columns 8–10 show the results of our approach. The last two columns list our speedup over the other two approaches.

According to Table II, the golden result method costs much more time than the others. From an accuracy perspective, our approach has an averaged CD of 0.0051 for all the benchmarks. The approximate formula-based method results in a CD about one order of magnitude larger on average than ours. Also, our approach has averaged speedups of 1140 and 752 times compared to the golden result method and the approximate formula-based method, respectively. Since the exact method costs even more time than the golden result method, we can see that our approach is a promising alternative to trade a little correctness for large savings in CPU time.

B. PO-Aware Testability-Based Replacement Strategy

In this subsection, we conducted four experiments that show the trends in PBCs from the aspects of energy and correctness. Note that in this subsection, we first discuss the reduction in switching energy in PBCs. The issue of leakage energy will be discussed in the next subsection.

In the first experiment, the proposed PO-aware testability-based replacement strategy was applied to replace general logic gates with probabilistic ones to form PBCs. Fig. 20 shows the results demonstrating the ratio of consumed switching energy of PBCs with respect to the switching energy of the original circuits under the C_{avg} constraint for the benchmarks *alu*4 and *dalu*. Hundred percent of the correctness means that no gate was replaced, and 100% of the ratio of switching energy means that the energy consumption of the PBC is the same as that of the original circuit. Each curve in Fig. 20 represents the change of switching energy under a voltage domain for probabilistic gates. For example, in Fig. 20(a), the ratio of switching energy between the resultant PBC and the original *alu*4 benchmark is 81% under 90% C_{avg} constraint when 0.8 V voltage domain is used for probabilistic gates. The end points of these three curves in Fig. 20 represent the minimal correctness and maximal switching energy saving that the PBC will reach when 100% gates in the PBC are probabilistic. Having all probabilistic gates in a PBC is impractical, however, because the corresponding correctness is not always satisfactory. Thus, we set 90% as the C_{avg} constraint in the succeeding experiments.

In the second experiment, we show the ratios of probabilistic gates and switching energy consumption of a PBC under the

TABLE II
EXPERIMENTAL RESULTS OF THE GOLDEN, FORMULA-BASED, AND OUR APPROACHES CONSIDERING ALL INPUT PATTERNS

Bench	|PI|	|Gate|	Golden	Approx. Formula-based			Ours			Speedup	
			T(s)	T(s)	CD	|Patt.|	T(s)	CD	|Patt.|	$\frac{\text{Golden}}{\text{Ours}}$	$\frac{\text{Formula}}{\text{Ours}}$
pm1	16	127	47.15	0.86	0.0076	65536	0.02	0.0018	6400	2357.50	43.00
comp	32	263	96.26	58.22	0.0543	2097152	0.06	0.0036	6784	1604.33	970.33
stepper.	29	388	142.73	92.46	0.0121	2097152	0.08	0.0060	7168	1784.12	1155.75
cht	47	466	171.44	106.82	0.0093	2097152	0.11	0.0057	8192	1558.55	971.09
apex7	49	592	217.62	139.01	0.0274	2097152	0.11	0.0052	6400	1978.36	1263.73
C880	60	615	228.98	145.93	0.0213	2097152	0.46	0.0044	25856	497.78	317.24
apex6	135	1222	564.92	381.96	0.0168	2097152	0.35	0.0052	8064	1614.06	1091.31
sasc	133	1384	509.85	354.25	0.0267	2097152	0.36	0.0053	9088	1416.25	984.03
simple_spi	148	1858	686.12	474.34	0.0166	2097152	0.67	0.0062	12672	1024.06	707.97
C3540	50	1921	702.43	445.70	0.1195	2097152	1.45	0.0061	26368	484.43	307.38
alu4	14	2015	918.98	4.42	0.0260	16384	1.51	0.0043	20992	608.60	2.93
i9	88	2237	817.23	490.17	0.1015	2097152	1.78	0.0066	27776	459.12	275.38
C6288	32	2464	892.05	629.49	0.3474	2097152	2.09	0.0068	30080	426.82	301.19
x3	135	2814	1033.50	659.83	0.0146	2097152	0.53	0.0049	6528	1950.00	1244.96
x1	51	2984	1088.38	648.01	0.0071	2097152	0.55	0.0027	6400	1978.87	1178.20
pair	173	3598	1323.58	867.79	0.0521	2097152	0.87	0.0055	8448	1521.36	997.46
frg2	143	3669	1347.81	877.48	0.0248	2097152	0.96	0.0050	9088	1403.97	914.04
t481	16	4336	1583.37	30.75	0.0158	65536	2.14	0.0007	17280	739.89	14.37
systemcdes	322	5729	2111.52	1447.69	0.0428	2097152	3.70	0.0068	22400	570.68	391.27
i10	257	5753	2118.88	1391.37	0.1944	2097152	3.73	0.0050	22528	568.06	373.02
i8	133	7674	2815.13	1746.15	0.0734	2097152	4.06	0.0053	18432	693.38	430.09
des_area	368	7707	2833.99	1967.92	0.0426	2097152	6.86	0.0071	30848	413.12	286.87
des	256	9246	3422.86	2219.78	0.0330	2097152	8.43	0.0106	31360	406.03	263.32
too_large	38	22219	10662.90	18091.10	0.0265	2097152	5.37	0.0015	6400	1985.65	3368.92
mem_ctrl	256	9246	3422.86	2219.78	0.0330	2097152	8.43	0.0106	31360	406.03	263.32
Average	-	-	2032.57	2520.38	**0.0544**	-	3.11	**0.0051**	-	1140.22	752.00
Ratio	-	-	654.15	811.14	10.6008	-	1	1	-	-	-

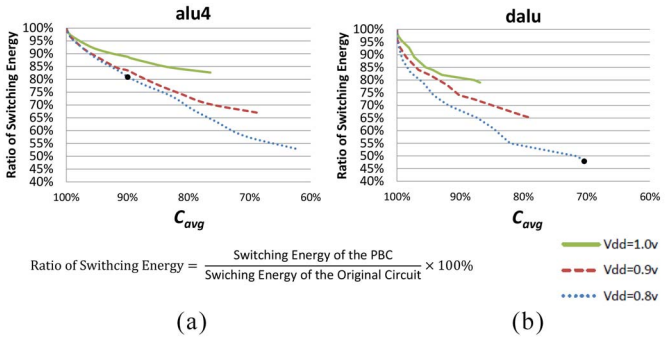


Fig. 20. Relationship between switching energy consumption and correctness constraint under (a) C_{avg} for *alu*4 and (b) C_{avg} for *dalu*.

C_{avg} constraint of 90% using the proposed replacement strategy, as shown in Fig. 21. In each benchmark, three results are shown for different voltage domains of probabilistic gates, namely, 1.0 V, 0.9 V, and 0.8 V.

In Fig. 21, the gray bar and black bar represent the ratios of the probabilistic gates and the switching energy consumption of the PBCs. On average, the ratios of probabilistic gates are 89.95%, 82.92%, and 71.84% for 1.0 V, 0.9 V, and 0.8 V, respectively. The ratios of switching energy consumption are 81.67%, 68.68%, and 62.26% for 1.0 V, 0.9*V* V, and 0.8 V, respectively. According to Fig. 21, we can see that the ratio of probabilistic gates is smaller when a lower voltage domain is adopted. This is because probabilistic gates with lower voltage domains are more likely to be erroneous such that the resultant PBC violates the correctness constraint more easily. Furthermore, we can see that the switching energy consumption of PBCs is usually smaller when probabilistic gates are adopted in the 0.9 V or 0.8 V voltage domains. Most results show that the 0.8 V voltage domain results in a larger savings in switching energy saving than for 0.9 V. However, it is not always the case, since switching energy consumption is related to the voltage domain as well as to the number of probabilistic gates replaced under the correctness constraint.

To address the timing issue caused by lowering the supply voltage, the third experiment compares the delay change in the PBCs with the original PO-aware testability-based replacement strategy to the delay change with the timing-aware replacement method under the 0.8 V voltage domain for probabilistic gates.

In Fig. 22, the names followed by _TA represent the results obtained using the timing-aware method. For example, the circuit delay of the PBC *sasc*, as highlighted in the red box, was about 107% of that of the original circuit after having 89% probabilistic gates under the C_{avg} constraint of 90% for the 0.8 V voltage domain. On average, the circuit delays of the PBCs for the original replacement and the timing-aware replacement methods were about 124% and 105% of the original circuit after having 72% and 61% probabilistic gates, respectively.

According to Fig. 22, we can see that when more gates are replaced with probabilistic gates, the circuit delay usually increases; this could be up to 150% of that of the original circuit. However, we also see that for some circuits like *simple_spi, alu*4, and *x*1, the circuit delay does not suffer much. The circuit delay might be even smaller than the original one due to changes in critical paths, e.g., *steppermotodrive*, *simple_spi*, *i9*, and *i8* circuits. On the other hand, although the timing-aware replacement method could keep the circuit delay of PBCs close to the original circuit delay, it reduces the number of probabilistic gates in most circuits. However, for *cht*_TA or *alu*4_TA, as highlighted in the red boxes, circuit delay was effectively reduced while the ratios of probabilistic gates are almost the same. We believe that this timing suffering

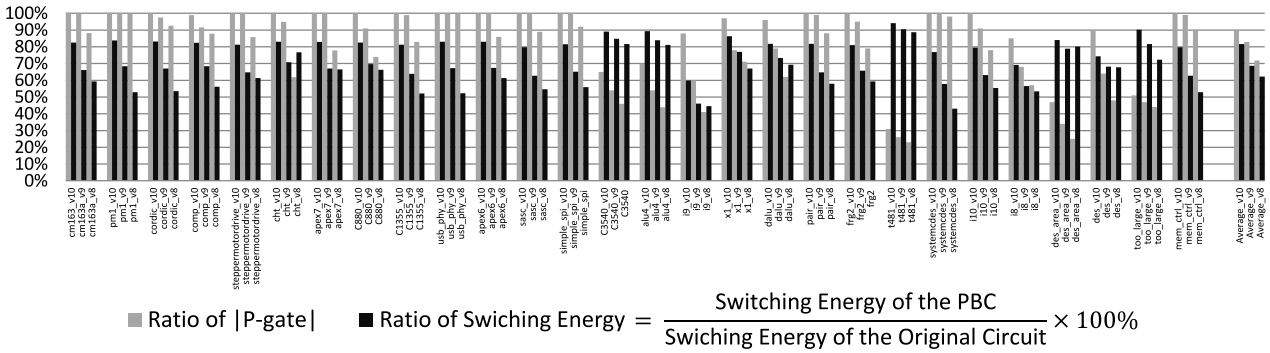


Fig. 21. Ratios of probabilistic gates and switching energy consumption in PBCs with voltage domains of 1.0 V, 0.9 V, or 0.8 V under the C_{avg} constraint of 90%.

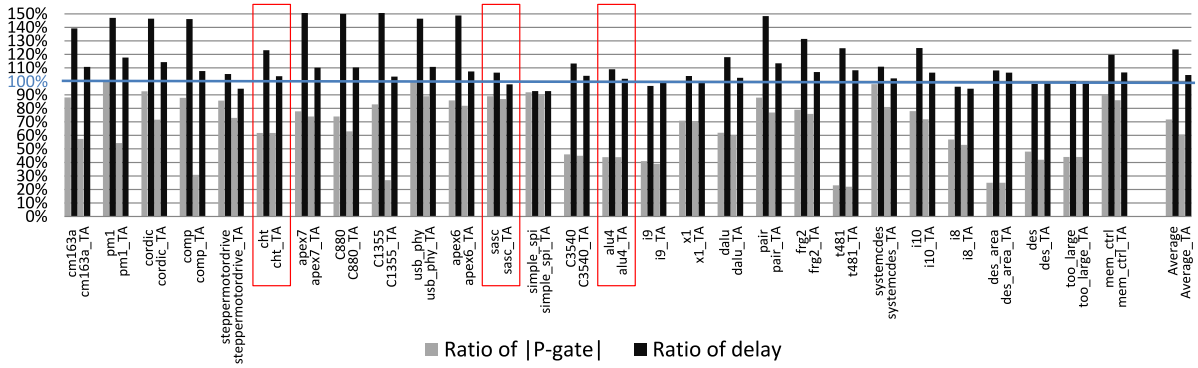


Fig. 22. Ratios of circuit delay under C_{avg} constraint of 90%.

would be alleviated if more accurate timing models were used in the timing-aware method. Since the timing issue is not the major concern of this paper, here we just demonstrate the practicability of PBC for timing-critical designs and do not compare different timing analysis methods.

The fourth experiment shows the comparison between different replacement strategies. In this experiment, the adopted voltage domain for probabilistic gates was 0.8 V, and the C_{avg} constraint was 90%. Here, the ratio of probabilistic gates is used as a measurement, i.e., a higher ratio means the corresponding replacement order is better. The bar charts shown in Fig. 23 summarize the results under the C_{avg} constraint of 90%. In the figure, the light gray bars, gray bars, and black bars represent the results from the naive replacement strategy, testability-based replacement strategy, and PO-aware testability-based replacement strategy, respectively. According to Fig. 23, the testability-based replacement strategy performed equal to or better than the naive replacement strategy for all cases. On the other hand, the PO-aware testability-based replacement strategy performed equal to or better than testability-based replacement strategy for all cases except *dalu*, as highlighted in Fig. 23, under the C_{avg} constraint. On average, the ratios of probabilistic gates for naive replacement strategy, testability-based replacement strategy, and PO-aware testability-based replacement strategy were 49.31%, 69.24%, and 71.84%, respectively. According to these experimental results, the proposed PO-aware testability-based replacement strategy is a promising approach.

C. Level-Up Replacement Strategy

The final experiment shows the comparison of the PDP of PBCs between the PO-aware testability-based replacement strategy (two voltage domains) and the level-up replacement strategy (more than two voltage domains). Since the circuit delays vary with different replacement strategies, PDP is a fair metric correlated with the energy efficiency of PBCs.

The effect of level-up structure can be clearly seen in Fig. 24. The black and light gray bars in the figure represent the ratios of the dynamic PDP and leakage PDP of a circuit to the total PDP of the original circuit, respectively. The *x*-axis shows the benchmarks, followed by *_ori,* _PO, and *_level_up* to distinguish the PDP of the original circuit, of the PBC by the PO-aware testability-based replacement strategy, and of the PBC by the level-up replacement strategy, respectively. The last four bars show the average ratios, where the last blue one shows the best ratio of the total PDP between the PO-aware testability-based replacement strategy and the level-up replacement strategy on average.

For example, as highlighted in the red box in Fig. 24, when using the level-up replacement strategy, the ratios of dynamic PDP and the leakage PDP of *C*1355*_level_up* were 50.62% and 0.35% of the total PDP of *C*_ori, respectively. On average, the ratios of dynamic PDP and the leakage PDP for the original circuit were 97.20% and 2.80%, respectively. The ratios of dynamic PDP and the leakage PDP for the PO-aware testability-based replacement strategy were 59.76% and 8.57%, respectively. The ratios of dynamic PDP and the leakage PDP for level-up replacement strategy were 62.01% and 2.53%, respectively.

According to Fig. 24, we can see that although the dynamic PDP of the PBCs by the PO-aware testability-based replacement strategy were reduced, the leakage PDP of the PBCs were increased g reatly. By means of level-up replacement strategy, however, the leakage PDP could be effectively reduced for some circuits. Nevertheless, the level-up replacement might result in larger PDP than the PO-aware testability-based

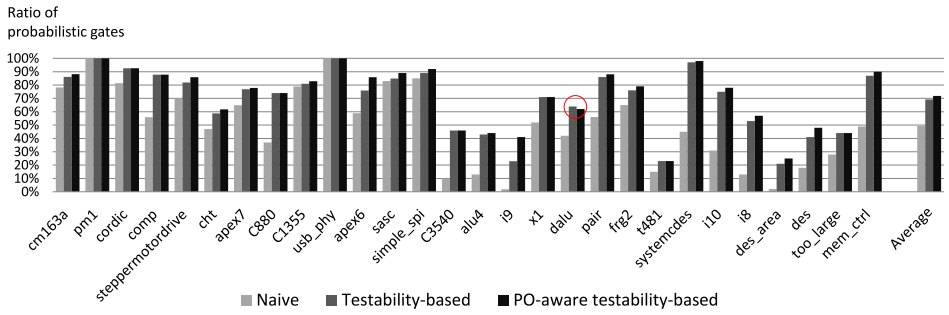


Fig. 23. Probabilistic gate number comparison between different strategies under C_{avg} constraint of 90%.

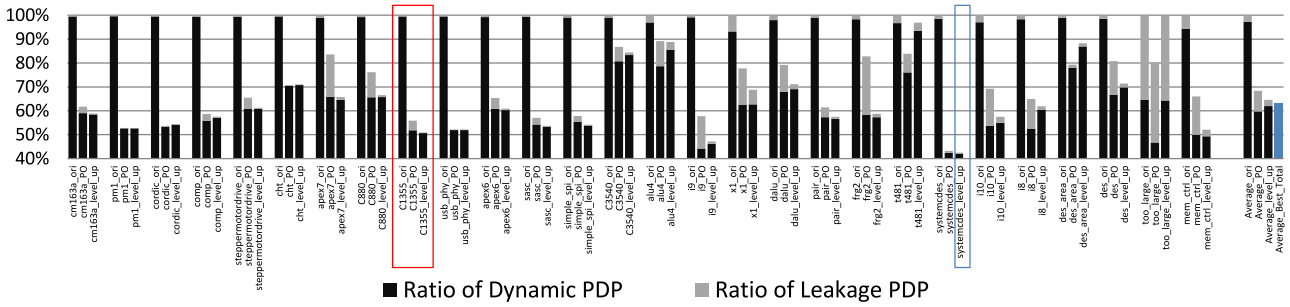


Fig. 24. PDP comparison among different strategies under C_{avg} constraint of 90%.

replacement strategy for some circuits. Thus, to ensure to have a PBC with a lower PDP, we choose the better result after conducting both strategies in the flow shown in Fig. 19. In summary, the proposed algorithm reduces PDP by an average of 36.99% (100%–63.01%), as shown with the last blue bar in Fig. 24, and by up to 57.52% (100%–42.48%), as highlighted in the blue box in Fig. 24, when C_{avg} constraint is set to 90%.

VI. Conclusion

Evaluating the correctness of PBCs is a crucial procedure in the analysis of PBCs. This paper presents a statistical approach that accurately and efficiently evaluates the correctness of PBCs. This paper also presents several strategies for effectively minimizing the power consumption in PBC. The experimental results show that the proposed power optimization flow gives an averaged PDP saving of 36.99% under the C_{avg} constraint of 90%. Using these techniques, the analysis and optimization of PBCs become tractable and facilitate the PBC designs. Our future work will focus on extending the proposed approaches to sequential circuits, considering more types of probabilistic gates in the library, and considering different correctness metrics. Additionally, we are going to develop an algorithm for dealing with timing issue of PBC designs.

References

[1] R. S. Asamwar, K. Bhurchandi, and A. S. Gandhi, "Successive image interpolation using lifting scheme approach," *J. Comput. Sci.*, vol. 6, no. 9, pp. 969–978, 2010.

[2] K. Binder, *Monte Carlo Methods in Statistical Physics*. Berlin, Germany: Springer, 1988.

[3] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: NY, USA: Springer, 2006.

[4] F. Brglez, "On testability of combinational networks," in *Proc. Int. Symp. Circuits Syst.*, 1984, pp. 221–225.

[5] L. N. Chakrapani *et al.*, "Ultra efficient (embedded) SoC architectures based on probabilistic CMOS technology," in *Proc. Design Autom. Test Europe (DATE)*, Munich, Germany, 2006, pp. 1110–1115.

[6] L. N. B. Chakrapani and K. V. Palem, "A probabilistic Boolean logic and its meaning," Dept. Comput. Sci., Rice Univ., Houston, TX, USA, Tech. Rep. TR08-05, 2008.

[7] L. N. B. Chakrapani, "Probabilistic Boolean logic, arithmetic and architectures," Ph.D. dissertation, Dept. Comput., Georgia Tech., Atlanta, GA, USA, 2008.

[8] L. N. B. Chakrapani, J. George, B. Marr, B. E. S. Akgul, and K. V. Palem, "Probabilistic design: A survey of probabilistic CMOS technology and future directions for terascale IC design," in *VLSI-SoC: Research Trends in VLSI and Systems on Chip*, vol. 249. Boston, MA, USA: Springer, 2008, pp. 101–118.

[9] L. N. B. Chakrapani, K. K. Muntimadugu, A. Lingamneni, J. George, and K. V. Palem, "Highly energy and performance efficient embedded computing through approximately correct arithmetic: A mathematical foundation and preliminary experimental validation," in *Proc. Int. Conf. Compilers Archit. Syn. Embedded Syst.*, Atlanta, GA, USA, 2008, pp. 187–196.

[10] S. C. Chang, W.-B. Jone, and S.-S. Chang, "TAIR: Testability analysis by implication reasoning," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 19, no. 1, pp. 152–160, Jan. 2000.

[11] S. Cheemalavagu, P. Korkmaz, and K. V. Palem, "Ultra low-energy computing via probabilistic algorithms and devices: CMOS device primitives and the energy-probability relationship," in *Proc. Int. Conf. Solid State Devices Mater.*, San Francisco, CA, 2004, pp. 402–403.

[12] S. Cheemalavagu, P. Korkmaz, K. V. Palem, B. E. S. Akgul, and L. N. Chakrapani, "A probabilistic CMOS switch and its realization by exploiting noise," in *Proc. 14th Int. Conf. Very Large Scale Integr. Syst. Chip (VLSI-SoC)*, Nice, France, 2005, pp. 452–457.

[13] M. R. Choudhury and K. Mohanram, "Reliability analysis of logic circuits," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 28, no. 3, pp. 392–405, Mar. 2009.

[14] C.-C. Chiou *et al.*, "A statistic-based approach to testability analysis," in *Proc. 9th Int. Symp. Qual. Electron. Design (ISQED)*, San Jose, CA, USA, 2008, pp. 267–270.

[15] W. G. Cochran, "The distribution of quadratic forms in a normal system, with applications to the analysis of covariance," *Math. Proc. Cambridge Philosoph. Soc.*, vol. 30, no. 2, pp. 178–191, 1934.

[16] J. George, B. Marr, B. E. S. Akgul, and K. V. Palem, "Probabilistic arithmetic and energy efficient embedded signal processing," in *Proc. Int. Conf. Compilers Archit. Syn. Embedded Syst.*, Seoul, Korea, 2006, pp. 158–168.

[17] L. H. Goldstein, "Controllability/observability analysis of digital circuits," *IEEE Trans. Circuits Syst.*, vol. 26, no. 9, pp. 685–693, Sep. 1979.

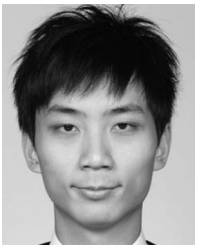
[18] V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, and K. Roy, "IMPACT: IMPrecise adders for low-power approximate computing," in *Proc. Int. Symp. Low Power Electron. Design (ISLPED)*, Fukuoka, Japan, 2011, pp. 409–414.
[19] International Technology Roadmap for Semiconductors, "International technology roadmap for semiconductors 2007 edition," p. Design 5, 2007.
[20] N. L. Johnson, S. Kotz, and N. Balakrishnan, *Continuous Univariate Distributions*. New York, NY, USA: Wiley, 1995.
[21] M. Keating, D. Flynn, R. Aitken, A. Gibbons, and K. Shi, *Low Power Methodology Manual: For System-on-Chip Design*. New York, NY, USA: Springer, 2007.
[22] S. W. Keckler, K. Olukotun, and H. P. Hofstee, *Multicore Processors and Systems*. New York, NY, USA: Springer, 2009.
[23] P. Korkmaz, B. E. S. Akgul, K. V. Palem, and L. N. Chakrapani, "Advocating noise as an agent for ultra-low energy computing: Probabilistic CMOS devices and their characteristics," *Jpn. J. Appl. Phys.*, vol. 45, no. 4B, pp. 3307–3316, 2006.
[24] S. Krishnaswamy, G. F. Viamontes, I. L. Markov, and J. P. Hayes, "Probabilistic transfer matrices in symbolic reliability analysis of logic circuits," *ACM Trans. Design Autom. Electron. Syst.*, vol. 13, no. 1, 2008, Art. ID 8.
[25] R. V. Hogg, J. W. McKean, and A. T. Craig, *Introduction to Mathematical Statistics*. Boston, MA, USA: Pearson, 2012.
[26] I. R. Miller and J. E. Freund, *Probability and Statistics for Engineers*. Englewood Cliffs, NJ, USA: Prentice Hall, 1990.
[27] J. Miao, A. Gerstlauer, and M. Orshansky, "Approximate logic synthesis under general error magnitude and frequency constraints," in *Proc. IEEE/ACM Int. Conf. Comput.-Aided Design (ICCAD)*, San Jose, CA, USA, 2013, pp. 779–786.
[28] K. Palem *et al.*, "Ten years of building broken chips: The physics and engineering of inexact computing," *ACM Trans. Embedded Comput. Syst.*, vol. 12, no. 2s, May 2013, Art. ID 87.
[29] C. Piguet, *Low-Power CMOS Circuits: Technology, Logic Design and CAD Tools*. Boca Raton, FL, USA: CRC Press, 2006.
[30] D. Shin and S. K. Gupta, "A new circuit simplification method for error tolerant applications," in *Proc. Design Autom. Test Europe Conf. Exhibit. (DATE)*, Grenoble, France, 2011, pp. 1–6.
[31] A. J. Strecok, "On the calculation of the inverse of the error function, mathematics of computation," *Math. Comput.*, vol. 22, no. 101, pp. 144–158, 1968.
[32] S. Venkataramani, A. Sabne, V. Kozhikkottu, K. Roy, and A. Raghunathan, "SALSA: Systematic logic synthesis of approximate circuits," in *Proc. 49th ACM/EDAC/IEEE Design Autom. Conf. (DAC)*, San Francisco, CA, USA, 2012, pp. 796–801.
[33] S. Venkataramani, K. Roy, and A. Raghunathan, "Substitute-and-simplify: A unified design paradigm for approximate and quality configurable circuits," in *Proc. Design Autom. Test Europe Conf. Exhibit. (DATE)*, Grenoble, France, 2013, pp. 1367–1372.
[34] D. T. Wang, "An algorithm for the generation of test sets for combinational logic network," *IEEE Trans. Comput.*, vol. C-24, no. 7, pp. 742–746, Jul. 1975.
[35] (Sep. 1, 2013). *Predictive Technology Model*. [Online]. Available: http://ptm.asu.edu/
[36] (May 22, 2013). *IWLS 2005 Benchmarks*. [Online]. Available: http://iwls.org/iwls2005/ benchmarks.html
[37] (Feb. 13, 2013). *INTEL*. [Online]. Available: http://www.intel.com/go/terascale/
[38] (Aug. 28, 2013). *Synopsys Design Compiler*. [Online]. Available: http://www.synopsys.com/
[39] (Aug. 28, 2013). *Synopsys HSPICE*. [Online]. Available: http://www.synopsys.com/
[40] (Aug. 28, 2013). *Synopsys Liberty NCX*. [Online]. Available: http://www.synopsys.com/

Ching-Yi Huang received the B.S. degree from the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, in 2009, where he is currently pursuing the Ph.D. degree from the Department of Computer Science.

His current research interests include logic synthesis, optimization, verification for very large scale integration (VLSI) designs, and automation for emerging technologies.

Zheng-Shan Yu received the B.S. and M.S. degrees from the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, in 2011 and 2013, respectively.

His current research interests include logic synthesis, optimization, verification for VLSI designs, and automation for emerging technologies.



Yung-Chun Hu received the B.S. and M.S. degrees from the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, in 2012 and 2014, respectively.

His current research interests include physical synthesis for VLSI designs and logic synthesis for emerging technologies.

Tung-Chen Tsou received the B.S. degree from the Department of Mathematics, National Tsing Hua University, Hsinchu, Taiwan, in 2012, and the M.S. degree from the Institute of Statistics, National Tsing Hua University, Hsinchu, Taiwan, in 2015.

His current research interests include statistical computing and statistical timing analysis for VLSI designs.

Chun-Yao Wang (S'00–M'03) received the B.S. degree from the Department of Electronics Engineering, National Taipei University of Technology, Taipei, Taiwan, and the Ph.D. degree from the Department of Electronics Engineering, National Chiao Tung University, Hsinchu, Taiwan, in 1994 and 2002, respectively.

Since 2003, he has been an Assistant Professor at the Department of Computer Science, National Tsing Hua University, Hsinchu, where he is currently a Professor. His current research interests include logic synthesis, optimization, and verification for very large-scale integrated/system-on-chip designs and emerging technologies. He has published over 50 technical papers in the above areas and holds eight patents.

Prof. Wang was the recipient of the best paper nomination in the 2009 IEEE Asia and South Pacific Design Automation Conference and the 2010 IEEE/ACM Design Automation Conference, respectively.

Yung-Chih Chen received the B.S., M.S., and Ph.D. degrees from the Department of Computer Science, National Tsing Hua University, Hsinchu, Taiwan, in 2003, 2005, and 2011, respectively.

He was a Visiting Student at the Department of Computer Science and Engineering, Pennsylvania State University, State College, PA, USA, from 2010 to 2011. He then joined the Department of Electronic Engineering, Chung Yuan Christian University, Taoyuan, Taiwan, where he was an Assistant Professor from 2011 to 2012. He is currently an Assistant Professor with the Department of Computer Science and Engineering, Yuan Ze University, Chung-Li, Taiwan. His current research interests include logic synthesis, design verification, and design automation for emerging technologies.