A Survey of Power Estimation Techniques in VLSI Circuits

Farid N. Najm

Coordinated Science Laboratory
University of Illinois at Urbana-Champaign
Urbana, IL 61801

Abstract

With the advent of portable and high-density microelectronic devices, the power dissipation of very large scale integrated (VLSI) circuits is becoming a critical concern. Accurate and efficient power estimation during the design phase is required in order to meet the power specifications without a costly redesign process. In this paper, we present a review/tutorial of the power estimation techniques that have recently been proposed.

Invited, IEEE Trans. on VLSI, Dec. 1994.

1. Introduction

The continuing decrease in feature size and the corresponding increase in chip density and operating frequency have made power consumption a major concern in VLSI design [1, 2]. Modern microprocessors are indeed *hot*: the PowerPC chip from Motorola consumes 8.5 Watts, the Pentium chip from Intel consumes 16 Watts, and DEC's alpha chip consumes 30 Watts. Excessive power dissipation in integrated circuits not only discourages their use in a portable environment, but also causes overheating, which degrades performance and reduces chip lifetime. To control their temperature levels, high power chips require specialized and costly packaging and heat-sink arrangements. This, combined with the recently growing demand for low-power portable communications and computing systems, has created a need to limit the power consumption in many chip designs. Indeed, the Semiconductor Industry Association has identified low-power design techniques as a critical technological need [3].

Managing the power of an IC design adds to a growing list of problems that IC designers and design managers have to contend with. Computer Aided Design (CAD) tools are needed to help with the power management tasks. Specifically, there is a need for CAD tools to estimate power dissipation *during* the design phase in order to meet the power specifications without a costly redesign process.

In CMOS and BiCMOS technologies, the chip components (gates, cells) draw power supply current only during a logic transition (if we ignore the small leakage current). While this is considered an attractive low-power feature of these technologies, it makes the power-dissipation highly dependent on the *switching activity* inside these circuits. Simply put, a more active circuit will consume more power. This complicates the power estimation problem because the power becomes a moving target - it is *input pattern-dependent*.

Thus the simple and straight-forward solution of estimating power by using a simulator is severely complicated by this pattern-dependence problem. Input signals are generally unknown during the design phase because they depend on the system (or chip) in which the chip (or functional block) will eventually be used. Furthermore, it is practically impossible to estimate the power by simulating the circuit for all possible inputs. Recently, several techniques have been proposed to overcome this problem by using *probabilities* to describe the set of *all possible logic signals*, and then studying the power resulting from the collective influence of all these signals. This formulation achieves a certain degree of *pattern-independence* that allows one to efficiently estimate and manipulate the power dissipation.

The rest of this paper is organized as follows. In the next section, we explain the power estimation problem in more detail and introduce a number of probabilistic measures that have been used to estimate power. Section 3 contains a literature survey of power estimation techniques and the following two sections focus in more detail on the two main types of approaches. We discuss sequential circuit power estimation in section 6, and provide a summary and conclusions in section 7.

2. Detailed Problem Description

By *power estimation* we generally refer to the problem of estimating the *average* power dissipation of a digital circuit. This is different from estimating the *worst case* instantaneous power, often referred to as the *voltage drop problem* [4–6]. Chip heating and temperature are directly related to the *average* power.

We have already alluded to a most straight-forward method of power estimation, namely by simulation: perform a *circuit simulation* of the design and monitor the power supply current waveform. Subsequently, the average of the current waveform is computed and used to provide the average power. The advantages of this technique are mainly its accuracy and generality. It can be used to estimate the power of any circuit, regardless of technology, design style, functionality, architecture, etc. The simulation results, however, are directly related to the specific input signals used to drive the simulator. Furthermore, *complete and specific* information about the input signals is required, in the form of voltage waveforms. Hence we describe these simulation-based techniques as being *strongly pattern-dependent.*

The pattern-dependence problem is serious. Often, the power of a functional block needs to be estimated when the rest of the chip has not yet been designed, or even completely specified. In such a case, very little may be known about the inputs to this functional block, and *complete and specific* information about its inputs would be impossible to obtain. Even if one is willing to guess at specific input waveforms, it may be impossible to assess if such inputs are *typical.* Large numbers of input patterns would have to be simulated, and this can become computationally very expensive, practically impossible for large circuits.

Most other (more efficient) power estimation techniques that have been proposed start out by simplifying the problem in three ways. Firstly, it is assumed that the power supply and ground voltage levels throughout the chip are fixed, so that it becomes simpler to compute the power by estimating the current drawn by every sub-circuit assuming a given fixed power supply voltage. Secondly, it is assumed that the circuit is built of logic gates and latches, and has the popular and well-structured design style of a *synchronous sequential circuit*, as shown in Fig. 1. In other words, it consists of latches driven by a common clock and combinational logic blocks whose inputs (outputs) are latch outputs (inputs). It is also assumed that the latches are edge-triggered and, with the use of a CMOS or BiCMOS design technology, the circuit draws no steady-state supply current. Therefore, the average power dissipation of the circuit can be broken down into (1) the power consumed by the latches and (2) that consumed by the combinational logic blocks. This provides a convenient way to decouple the problem and simplify the analysis. And, finally, it is commonly accepted that, in accordance with the results of [7], it is enough to consider only the charging/discharging current drawn by a logic gate, so that the short-circuit current during switching is neglected.

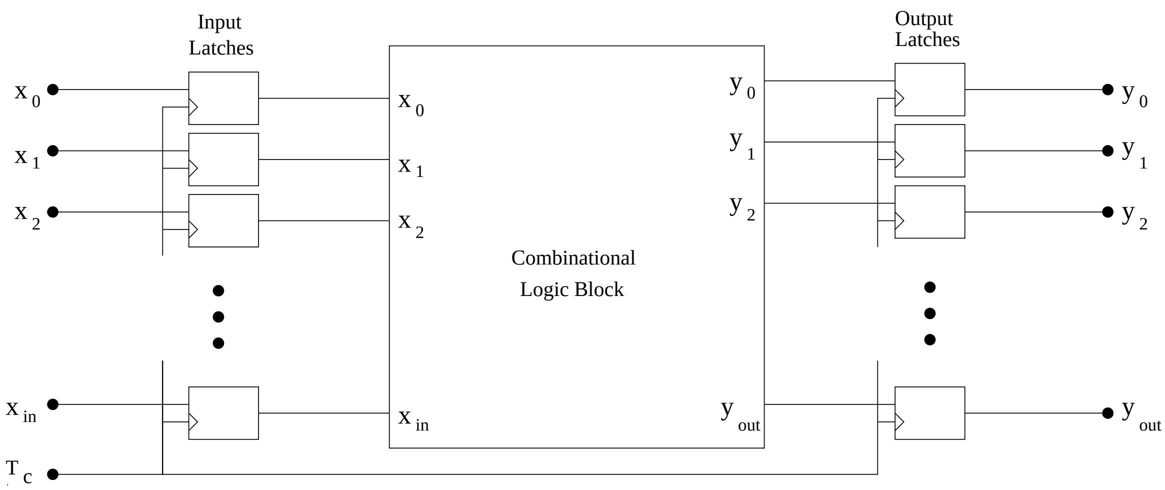

Figure 1. A combinational circuit embedded in a synchronous sequential design.

Whenever the clock triggers the latches, some of them will make a transition and will draw power. Thus latch power is drawn in synchrony with the clock. The same is not true for gates inside the combinational logic. Even though the inputs to a combinational logic block are updated by the latches (in synchrony with the clock), the internal gates of the block may make several transitions before settling to their steady state values for that clock period.

These additional transitions have been called hazards or glitches. Although unplanned for by the designer, they are not necessarily design errors. Only in the context of low-power design do they become a nuisance, because of the additional power that they dissipate. It has been observed [8] that this additional power dissipation is typically 20% of the total power, but can be as high as 70% of the total power in some cases such as combinational adders. We have observed that in a 16-bit multiplier circuit, some nodes make as many as 20 transitions before reaching steady state. This component of the power dissipation is computationally expensive to estimate, because it depends on the timing relationships between signals inside the circuit. Consequently, many proposed power estimation techniques have ignored this issue. We will refer to this elusive component of power as the *toggle power*. Computing the toggle power is one main challenge in power estimation.

Another challenge has to do with *independence* when signals are represented with probabilities. The reason for introducing probabilities is to solve the pattern-dependence problem, as follows. Instead of simulating the circuit for a large number of patterns and then averaging the result, one can simply compute (from the input pattern set, for instance) the fraction of cycles in which an input signal makes a transition (a *probability* measure) and use that information to estimate (somehow) how often internal nodes transition and, consequently, the power drawn by the circuit. Conceptually, this idea is shown in Fig. 2, which depicts both the conventional path of using circuit simulation and the alternative path of using probabilities. In a sense, one performs the averaging *before*, instead of after, running the analysis. Thus, a single run of a probabilistic *analysis tool* replaces a large number of circuit simulation runs, provided some loss of accuracy can be tolerated. The issues are exactly what probabilities are required, how they are to be obtained and, most importantly, what sort of analysis should be performed.

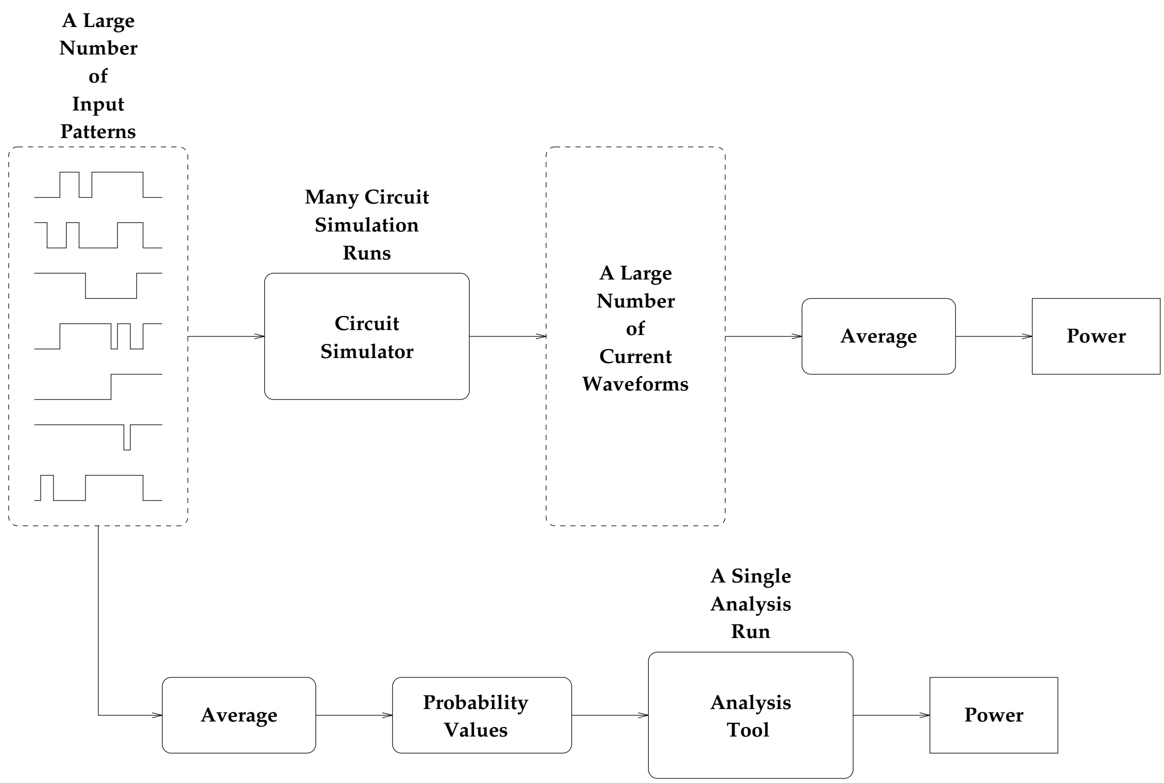

Figure 2. An alternative flow for power estimation.

In practice, one can directly provide the required input probabilities, eliminating the need for a large set of specific input patterns. The results of the analysis will depend on the supplied probabilities. Thus, to some extent the process is still pattern-dependent and the user must supply information about the *typical* behavior at the circuit inputs, in terms of probabilities. However, since one is not required to provide complete and specific information about the input signals, we call these approaches *weakly pattern-dependent.*

There are many ways of defining probability measures associated with the transitions made by a logic signal, be it at the primary inputs of the combinational block or at an internal node. We start with the following two:

Definition 1. (signal probability): *The signal probability $P_s(x)$ at a node x is defined as the average fraction of clock cycles in which the steady state value of x is a logic high.*

Definition 2. (transition probability): *The transition probability $P_t(x)$ at a node x is defined as the average fraction of clock cycles in which the steady state value of x is different from its initial value.*

The signal probability is a relatively old concept that was first introduced to study circuit testability [9]. It is important to note that both these probability measures are unaffected by the circuit internal delays. Indeed, they remain the same even if a zero-delay timing model is used. When this is done, however, the toggle power is automatically excluded from the analysis. This is a serious shortcoming of techniques that are based on these measures, as we will point out below.

If a zero-delay model is assumed and the transition probabilities are computed, then the power can be computed as:

$$P_{av} = \frac{1}{2T_c} V_{dd}^2 \sum_{i=1}^{n} C_i P_t(x_i) \tag{1}$$

where T_c is the clock period, C_i is the total capacitance at node x_i, and n is the total number of circuit nodes that are outputs of logic gates or cells. Since this assumes at most a single transition per clock cycle, then this is actually a *lower bound* on the true average power.

We can now discuss the signal independence issue. In practice, signals may be correlated so that, for instance, two of them may never be simultaneously high. It is computationally too expensive to compute these correlations, so that the circuit input and internal nodes are usually assumed to be *independent*. We refer to this as a *spatial independence* assumption. Another independence issue is whether the values of the same signal in two consecutive clock cycles are independent or not. If assumed independent, then the transition probability can be easily obtained from the signal probability according to:

$$P_t(x) = 2P_s(x)P_s(\overline{x}) = 2P_s(x)\left[1 - P_s(x)\right] \tag{2}$$

We refer to this as a *temporal independence* assumption.

Other recent power measures are based on the *transition density* formulation [10, 25]. The transition density at node x is the average number of transitions per second at node x, denoted $D(x)$. Formally:

Definition 3. (transition density) *If a logic signal $x(t)$ makes $n_x(T)$ transitions in a time interval of length T, then the transition density of $x(t)$ is defined as:*

$$D(x) \triangleq \lim_{T\to\infty} \frac{n_x(T)}{T} \tag{3}$$

The density provides an effective measure of switching activity in logic circuits. If the density at every circuit node is made available, the overall average power dissipation in the circuit can be computed as:

$$P_{av} = \frac{1}{2} V_{dd}^2 \sum_{i=1}^{n} C_i D(x_i) \tag{4}$$

In a synchronous circuit, with a clock period T_c, the relationship between transition density and transition probability is:

$$D(x) \geq \frac{P_t(x)}{T_c} \tag{5}$$

where equality occurs in the zero-delay case. Thus the transition probability gives a lower bound on the transition density.

Let $P(x)$ denote the *equilibrium probability* [25] of a logic signal $x(t)$, defined as the *average fraction of time that the signal is high.* Formally:

Definition 4. (equilibrium probability) *If $x(t)$ is a logic signal (switching between 0 and 1), then its equilibrium probability is defined as:*

$$P(x) \triangleq \lim_{T \to \infty} \frac{1}{T} \int_{\frac{-T}{2}}^{\frac{+T}{2}} x(t) dt \tag{6}$$

In contrast to the signal probability, the equilibrium probability depends on the circuit internal delays since it describes the signal behavior over time and not only its steady state behavior per clock cycle. In the zero-delay case, the equilibrium probability reduces to the signal probability.

In the remainder of this paper, we will make use of the probability measures defined above in discussing the various recently proposed power estimation techniques.

3. Brief Overview

The earliest proposed techniques of estimating power dissipation were strongly patter-dependent circuit simulation based [11, 12]. One would simulate the circuit while monitoring the supply voltage and current waveforms, which are subsequently used to compute the average power. Besides being strongly pattern dependent, these techniques are too slow to be used on large circuits, for which high power dissipation is a problem.

In order to improve computational efficiency, other simulation based techniques were also proposed using various kinds of timing, switch-level, and logic simulation [13–18]. These techniques generally assume that the power supply and ground voltages are fixed, and only the supply current waveform is estimated. While they are indeed more efficient than traditional circuit simulation, at the cost of some loss in accuracy, they remain strongly pattern-dependent.

In order to overcome the shortcomings of simulation-based techniques, other specialized approaches have been proposed with a focus on combinational digital CMOS circuits embedded in a synchronous design environment, as described above. For the rest of this section, therefore, we will be concerned with the power consumed in a combinational circuit whose inputs switch in synchrony.

The use of probabilities to estimate power was first proposed in [19]. In this work, a zero-delay model was assumed and a temporal independence assumption was made so that the transition probabilities could be estimated using signal probabilities based on (2). Signal probabilities supplied by the user at the primary inputs are propagated into the circuit assuming spatial independence and the power was computed based on (1). Since a zero-delay model was used, the toggle power was ignored.

A probabilistic power estimation approach that does compute the toggle power and does not make the zero-delay or temporal independence assumptions, called *probabilistic simulation* was proposed in [20–22]. In this technique, the use of probabilities was expanded to allow the specification of *probability waveforms*, as described in more detail in the next section. This approach assumed spatial independence, and was not restricted to only synchronous circuits. Improvements on this technique were proposed in [23, 24], where the accuracy and the correlation handling were improved upon.

Another probabilistic approach was proposed in [25–27], where the *transition density* measure of circuit activity was introduced. An algorithm was also presented for propagating the transition density into the circuit. This approach does not make a zero-delay assumption and makes only the *spatial* independence assumption, as will be discussed in more detail in section 4. Nevertheless, the result of this independence assumption is to make the computed density values insensitive to the internal circuit delays.

Yet another probabilistic approach was presented in [28], where Binary Decision Diagrams (BDDs) [35] were used to take into account internal node correlations and toggle power, at the cost of increased computation. This approach can become computationally expensive, especially for circuits where toggle power is dominant. It will be reviewed in more detail in section 4.

We refer to the above approaches as *probabilistic* because probabilistic information is *directly* propagated into the circuit. To perform this, special models for circuit blocks (gates) must be developed and stored in the cell library. In contrast, other techniques, that we will refer to as *statistical*, do not require specialized circuit models. Instead, they use traditional simulation models and simulate the circuit for a limited number of randomly generated input vectors while monitoring the power. These vectors are generated from user-specified probabilistic information about the circuit inputs. Using statistical estimation techniques, one can determine when to stop the simulation in order to obtain a certain specified error bound. Details of these techniques can be found in [29–32], and will be summarized below.

All of the above probabilistic and statistical techniques are applicable only to combinational circuits. They require the user to specify information on the activity at the latch outputs. Power estimation in sequential circuits will be discussed in section 6.

4. Probabilistic Techniques

Recently, several power estimation approaches have been proposed that use probabilities in order to solve the pattern-dependence problem. In practice, all are applicable only to combinational circuits and require the user to specify typical behavior at the combinational circuit inputs. We will compare and contrast these techniques based on the six criteria of: (1) Whether they include the toggle power, (2) If they handle temporal correlation, (3) Complexity of the required input specification, (4) Whether they provide the power consumed by individual gates, (5) If they handle spatial correlation, and (6) Speed. We will discuss five different approaches, for which the comparisons are shown in Table 1.

Table 1. Probabilistic techniques.

Approach	Handle Toggle Power?	Handle Temporal Correlation?	Input Specification	Individual Gate Power?	Handle Spatial Correlation?	Speed
Signal Probability	No	No	Simple	Yes	No	Fast
CREST	Yes	Yes	Moderate	Yes	No	Fast
DENSIM	Yes (approximately)	Yes	Simple	Yes	No	Fast
BDD	Yes	Yes, Internally	Simple	Yes	Yes, Internally	Slow
Correlation Coefficients	Yes	Yes	Moderate	Yes	Yes, Internally (approximately)	Moderate

These techniques all use simplified delay models for the circuit components and require user-supplied information about typical input behavior. Thus, their accuracy is limited by the quality of the delay models and the input specification. Nevertheless, some are more accurate than others, and this may be gauged by looking at criteria (1), (2), and (5) in the table.

4.1. Using signal probability

In [19], a zero-delay model is used and temporal as well as spatial independence is assumed. The user is expected to provide signal probabilities at the primary inputs. These are then propagated into the circuit to provide the probabilities at every node. In the paper, the propagation of probabilities is performed at the switch-level, but this is not essential to the approach. The simplest way to propagate probabilities is to work with a gate-level description of the circuit. Thus if $y = AND(x_1, x_2)$, then it follows from basic probability theory [34] that $P_s(y) = P_s(x_1)P_s(x_2)$, provided x_1 and x_2 are (spatially) independent. Similarly, other simple expressions can be derived for other gate types. Once the signal probabilities are computed at every node in the circuit, the power is computed by making use of (1) and (2), based on the temporal independence assumption.

In general, if the circuit is built from Boolean components that are not part of a pre-defined gate library, the signal probability can be computed by using a BDD [35] to represent the Boolean functions, as proposed in [10] and [37]. As an example to illustrate the BDD representation, consider the Boolean function $y = x_1x_2 + x_3$, which can be represented by the BDD shown in Fig. 3. The Boolean variables x_i are *ordered*, and each *level* in the BDD corresponds to a single variable. Each level may contain one or more BDD nodes at which one can branch in one of two directions, depending on the value of the relevant variable. For example, suppose that $x_1 = 1$, $x_2 = 0$, and $x_3 = 1$. To evaluate y, we start at the top node and branch to the right since $x_1 = 1$, then branch to the left since $x_2 = 0$, and finally branch to the right since $x_3 = 1$ to reach the terminal node "1." Thus the corresponding value of y is 1.

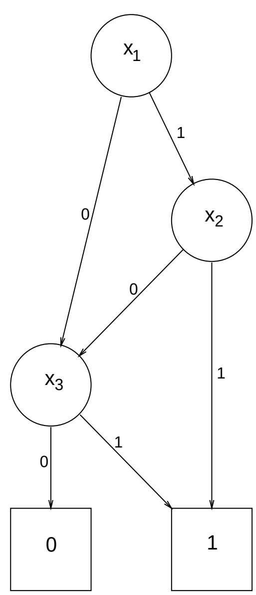

Figure 3. Example BDD representation.

In general, let $y = f(x_1, \ldots, x_n)$ be a Boolean function. If the inputs x_i are independent, then the signal probability of f can be obtained in *linear time* (in the size of its BDD representation), as follows. If $f_{x_1} = f(1, x_2, \ldots, x_n)$ and $f_{\overline{x_1}} = f(0, x_2, \ldots, x_n)$ are the *cofactors* of f with respect to x_1, then:

$$P(y) = P(x_1)P(f_{x_1}) + P(\overline{x_1})P(f_{\overline{x_1}}) \tag{7}$$

This equation shows how the BDD can be used to evaluate $P(y)$. The two nodes that are descendants of y in the BDD correspond to the cofactors of f. The probability of the cofactors can then be expressed in the same way, in terms of their descendants. Thus a depth-first-traversal of the BDD, with a post-order evaluation of $P(\cdot)$ at every node is all that is required. This can be implemented using the "scan" function of the BDD package [36].

4.2. Probabilistic Simulation (CREST)

This approach [20–22] requires the user to specify typical signal behavior at the circuit inputs using *probability waveforms*. A probability waveform is a sequence of values indicating the probability that the signal is high for certain time intervals, and the probability that it makes low-to-high transitions at specific time points. The transition times themselves are not random. This allows the computation of the average, as well as the variance, of the current waveforms drawn by the individual gates in the design in one simulation run. The average current waveforms can then be used to compute the average power dissipated in each gate and the total average power of the circuit.

An example of a probability waveform is shown in Fig. 4. In this example, the signal is high with probability 0.5, to begin with. It then transitions low-to-high with probability 0.2 at t_1, to become high with probability 0.25 between t_1 and t_2, etc. At every transition time point, the signal may also make a high-to-low transition, the probabilities of which can be computed from the other probabilities specified in the waveform. Notice that, at t_1, $0.2 \neq (1.0 - 0.5) \times 0.25$ which illustrates that temporal independence is not assumed. Given such waveforms at the primary inputs, they are *propagated* into the circuit to compute the corresponding probability waveforms at all the nodes.

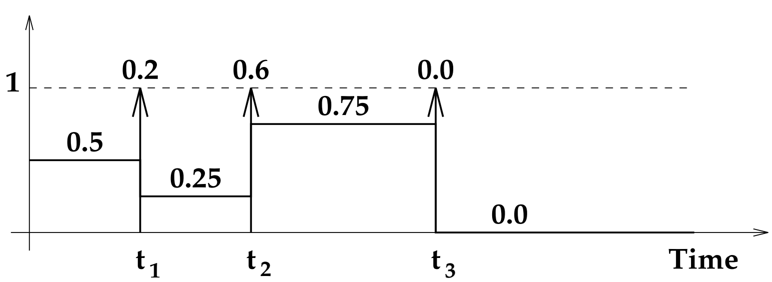

Figure 4. Example probability waveform.

The propagation algorithm is very similar to event driven logic simulation with an assignable delay model. The only difference is that the simulation algorithm and simulation model for each gate deal with the probability of making a transition rather than the definite occurrence of a transition. The events are propagated one at a time, using an event queue based mechanism. Whenever an event occurs at the input to a gate, the gate makes a contribution to the overall average current that is being estimated, and generates an output event that is scheduled after some time delay. In the original implementation of CREST, a transistor level netlist was used to compute the average current pulse and delay of every gate. The same can be achieved using gate level models, provided they are pre-characterized to estimate the current pulse and delay.

4.3. Transition density (DENSIM)

The average number of transitions per second at a node in the circuit has been called the transition density in [25–27], where an efficient algorithm is presented to propagate the density values from the inputs throughout the circuit. This was implemented in the program DENSIM for which the required input specification is a pair of numbers for every input node, namely the equilibrium probability and transition density. In this case, both signal values and signal transition times are random.

To see how the propagation algorithm works, recall the concept of *Boolean difference*: if y is a Boolean function that depends on x, then the Boolean difference of y with respect to x is defined as:

$$\frac{\partial y}{\partial x} \triangleq y|_{x=1} \oplus y|_{x=0} \tag{8}$$

where $\oplus$ denotes the exclusive-or operation. It was shown in [25] that, if the inputs x_i to a Boolean module are (spatially) *independent*, then the density of its output y is given by:

$$D(y) = \sum_{i=1}^{n} P\left(\frac{\partial y}{\partial x_i}\right) D(x_i) \tag{9}$$

The *simplicity* of this expression allows very efficient CAD implementations. Given the probability and density values at the primary inputs of a logic circuit, a single pass over the circuit, using (9), gives the density at every node. In order to compute the Boolean difference probabilities, one must also propagate the equilibrium probabilities $P(x)$ from the primary inputs throughout the circuit, using the same BDD algorithm for signal probability propagation described above.

As an example, consider the simple case of a 2-input logic AND gate: $y = x_1 x_2$. In this case, $\partial y/\partial x_1 = x_2$ and $\partial y/\partial x_2 = x_1$, so that:

$$D(y) = P(x_2)D(x_1) + P(x_1)D(x_2) \tag{10}$$

In more complex cases, where f is a general Boolean function, Binary Decision Diagrams can be used [25] to compute the Boolean difference probabilities. Recently, specialized BDD-based techniques have been proposed to facilitate this [39].

4.4. Using a BDD

The technique proposed in [28] attempts to handle both spatial and temporal correlations by using a BDD to represent the successive Boolean functions at every node in terms of the primary inputs, as follows. The circuit topology defines a Boolean function corresponding to every node that gives the *steady state* value of that node in terms of the primary inputs. The intermediate values that the node takes before reaching steady state are not represented by this function. Nevertheless, one can construct Boolean functions for them by making use of the circuit delay information, assuming the delay of every gate is a specified fixed constant.

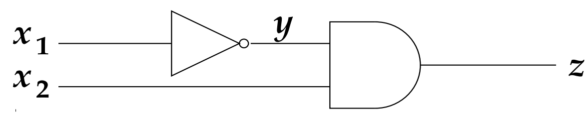

Figure 5. A simple test case circuit.

To illustrate, consider the circuit in Fig. 5, and let the values of x_1 and x_2 in two consecutive clock cycles be denoted $x_1(1)$, $x_1(2)$, and $x_2(1)$, $x_2(2)$. Assuming the AND gate and the inverter have comparable delays, a typical timing diagram is shown in Fig. 6, where it can be seen that node z may make two transitions before reaching steady state. The intermediate and steady state values of y and z can be expressed as follows:

$$y(1) = \overline{x_1(1)}, \qquad \text{and} \qquad y(2) = \overline{x_1(2)} \tag{11}$$

$$z(1) = \overline{x_1(1)}x_2(1), \qquad z(2) = \overline{x_1(1)}x_2(2), \qquad \text{and} \qquad z(3) = \overline{x_1(2)}x_2(2) \tag{12}$$

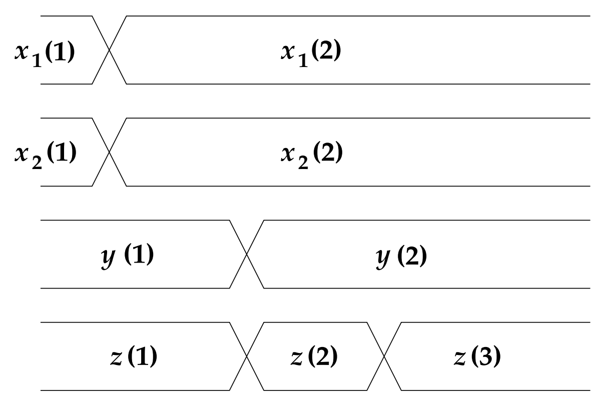

Figure 6. Timing diagram.

In this way, one can express the intermediate values of every node in terms of the two sets of values at the primary inputs. If a BDD is built for these functions, then the intermediate state probabilities can be accurately computed. In order to compute the probabilities of internal transitions, one can use the BDD to construct the exclusive-OR function of two consecutive intermediate states. Thus, in the above example, the probability that the first transition of z occurs is $P(z(1) \oplus z(2))$ and the probability that the second transition occurs is

$P(z(2) \oplus z(3))$. Once these XOR functions have been constructed, both of these probabilities can be computed from the BDD. The expected number of transitions at z in a clock cycle is, therefore, $E[n_z(T_c)] = P(z(1) \oplus z(2)) + P(z(2) \oplus z(3))$, and the transition density at z is $D(z) = E[n_z(T_c)]/T_c$.

Using a BDD to perform these tasks implicitly means that the BDD variables are assumed independent. In the above example, this means that $x_1(1)$, $x_1(2)$, $x_2(1)$, and $x_2(2)$ are independent. Thus, while some temporal correlation between $z(1)$ and $z(2)$ is taken care of (through the $\overline{x_1(1)}$ term), no temporal correlation between $y(1)$ and $y(2)$ is possible. The reason is that temporal and spatial independence are effectively assumed at the primary inputs. Hence the qualifications "Internally" in Table 1.

One disadvantage of this technique is that it is computationally expensive. Since the BDD is built for the whole circuit, there will be cases where the technique breaks down because the required BDD may be too big. As a result, this approach is limited to moderate sized circuits. The situation is actually potentially worse than this, because a BDD function must be built for every intermediate state and for their pairwise XOR functions. In cases where many intermediate transitions occur, even moderate sized circuits may be too big to handle. In absolute terms, and by way of comparison, the previous three techniques can run on circuits with a few thousand gates in a matter of seconds, while the one large circuit (with 2779 gates) reported for this approach takes over half an hour (on a DEC-station 5900). Nevertheless, the technique has many desirable and interesting features.

4.5. Correlation coefficients

Another probabilistic approach that is similar to probabilistic simulation was proposed in [24] whereby the correlation coefficients between steady state signal values are used as approximations to the correlation coefficients between the intermediate signal values. This allows spatial correlation to be handled approximately, and is much more efficient than trying to estimate the dynamic correlations between intermediate states. The steady state correlations are estimated from the BDD by constructing the function for the AND of two signals. The reported results have good accuracy, but the technique does require building the BDD for the whole circuit, which may not always be feasible.

5. Statistical Techniques

The idea behind these techniques is quite simple and appealing: simulate the circuit repeatedly, using some timing or logic simulator, while monitoring the power being consumed. Eventually, the power will converge to the average power, based on (3) and (4). The issues are how to select the input patterns to be applied in the simulations and how to decide when the measured power has *converged* close enough to the true average power. Normally, the inputs are randomly generated and statistical mean estimation techniques [38] are used to decide when to stop, essentially a *Monté Carlo* method. We will review the two main approaches that have been proposed, whose characteristics are compared in Table 2.

Table 2. Statistical techniques.

Approach	Handle Toggle Power?	Handle Temporal Correlation?	Input Specification	Individual Gate Power?	Handle Spatial Correlation?	Speed
McPower	Yes	Yes	Simple	No	Only Internally	Fast
MED	Yes	Yes	Simple	Yes	Only Internally	Moderate

5.1. Total power (McPower)

This approach [29–31] uses Monté Carlo simulation to estimate the total average power of the circuit. It consists of applying randomly-generated input patterns at the primary inputs and monitoring the energy dissipated per clock cycle using a simulator. If the successive input patterns are independently generated, a number N of such measurements is called a *random sample* whose average (divided by T_c) approaches the desired average power for large N. In order to stop the simulation when one is *close enough* to the average power, we need a so-called *stopping criterion.*

It was found experimentally [31] that the power consumed by a circuit over a period T has a distribution that is very close to *normal.* This allows one to use the following stopping criterion. Let $\bar{p}$ and s be the measured average and standard deviation of the random sample of the power, measured over a period T. Then we have $(1-\alpha) \times 100\%$ *confidence* that $|\bar{p} - P_{av}| < t_{\alpha/2}\ s/\sqrt{N}$, where $t_{\alpha/2}$ is obtained from the t-distribution [38] with $(N-1)$ degrees of freedom. This result can be rewritten as :

$$\frac{|P_{av} - \bar{p}|}{\bar{p}} < \frac{t_{\alpha/2}\ s}{\bar{p}\sqrt{N}} \tag{13}$$

Therefore, for a desired *percentage error* ϵ in the power estimate, and for a given confidence level $(1-\alpha)$, we must simulate the circuit until :

$$\frac{t_{\alpha/2}\ s}{\bar{p}\sqrt{N}} < \epsilon \tag{14}$$

Which means that the number of required simulations is:

$$N \geq \left(\frac{t_{\alpha/2}\ s}{\epsilon\ \bar{p}}\right)^2 \tag{15}$$

In practice, this technique was found to be very efficient. Typically, as few as 10 vectors may be enough to estimate the power of a large circuit with thousands of gates. But perhaps the most useful feature of this technique is that the user can specify the required accuracy and confidence level *up-front*. Thus, it retains the accuracy of deterministic simulation-based approaches, while achieving speeds comparable to probabilistic techniques. It also does not require an independence assumption for internal nodes; it only requires the primary inputs to be independent. The approach can be extended to model and take into account the correlations between input nodes.

Perhaps the only disadvantage of this approach is that, while it provides an accurate estimate of the total power, it does not provide the power consumed by individual gates or small groups of gates. It would take many more transitions to estimate (with the same accuracy) the power of individual gates, because some gates may switch very infrequently. This point will be further clarified below.

5.2. Power of individual gates (MED)

This recent technique [32] is a modification of the McPower approach that provides both the total and individual-gate power estimates, with user-specified accuracy and confidence. One reason why one may want to estimate the power consumed by individual gates is to be able to *diagnose* a high power problem, and find out which part of the circuit consumes the most power. Other reasons have to do with the fact that estimating gate power is essentially equivalent to estimating the transition density at every node. Indeed, the implementation of this technique in the program MED provides the transition density at every gate output node, in addition to the total power. These density values can then be used to estimate circuit reliability [25].

The main difference between this and the above approach is in the stopping criterion. Suppose we simulate the circuit for a time interval T, N times, and measure the number of transitions at a node every time, call this n_i. Then, according to the *Central Limit Theorem* [38], the average $\bar{n} = \sum_{i=1}^{N} n_i/N$ has a distribution which is close to normal for large N. If η is the true expected number of transitions in T, and s is the measured standard deviation of the N values n_i, then it can be shown that with confidence $(1-\alpha) \times 100\%$:

$$\frac{|\eta - \bar{n}|}{\bar{n}} \leq \frac{z_{\alpha/2}\ s}{\bar{n}\sqrt{N}} \tag{16}$$

provided N is larger than about 30 transitions, where $z_{\alpha/2}$ is obtained from the normal distribution [38]. The ratio $\bar{n}/T$ approaches the transition density $D = \eta/T$. Thus if a percentage error ϵ is tolerated in the density, then the number of required simulations is:

$$N \geq \left(\frac{z_{\alpha/2}\ s}{\epsilon\ \bar{n}}\right)^2 \tag{17}$$

It should be clear from (17) that for small values of $\bar{n}$ the number of samples required can become too large. It thus becomes too expensive to guarantee a percentage accuracy for low-density nodes. This is why the McPower approach cannot be used as is to measure node densities. The modification proposed in [32] is to use an absolute, rather than percentage, error bound for low-density nodes, as follows. A node is classified as *a low-density node* if it has $\bar{n} < \eta_{\min}$, where η_{min} is user-specified. For these nodes, if we use the modified stopping criterion:

$$N \geq \left(\frac{z_{\alpha/2}\ s}{\eta_{\min}\epsilon}\right)^2 \tag{18}$$

then with $(1-\alpha)$ confidence:

$$|\eta - \bar{n}| \leq \frac{z_{\alpha/2}\ s}{\sqrt{N}} \leq \eta_{\min}\epsilon \tag{19}$$

Thus $\eta_{\min}\epsilon$ becomes an *absolute* error bound that characterizes the accuracy for low-density nodes. Although the percentage error for low-density nodes sharply increases as $\bar{n} \rightarrow 0$, the absolute error remains relatively fixed. In fact, it can be shown that the absolute error bounds for low-density nodes are always *less than* the absolute error bounds for other nodes. Although these nodes require the longest time to converge, they have the least effect on circuit power and reliability. Therefore the above strategy reduces the execution time, with little or no penalty.

A weakness of this approach may be its speed (currently, a circuit with 16000 gates requires about 2 hours on a SUN sparc ELC). Further development may improve this performance.

6. Sequential Circuits

The above techniques do not apply directly to sequential circuits. While the CREST approach *can* be used to simulate a circuit with feedback, the resulting loss of accuracy due to the independence assumption, especially when recursively applied in a feedback loop, renders the results somewhat suspect. As for [28], although the title includes "sequential circuits," it is assumed that all states are equally probable, which is not true in practice.

To simplify the discussion, we will assume that the sequential circuit implements a finite state machine (FSM) with a connected state space. Another simplifying assumption that has been made by most researchers is to say that the FSM is *Markov* [34] (so that its future is independent of its past once its present state is specified). If the signal and transition probabilities at the *present state inputs* of the FSM (i.e., the latch outputs) are known, then, with some approximation, any of the above combinational circuit techniques can be used to compute the power.

Several approaches [40–43] have been proposed for sequential circuits, all of which make use of the above Markov assumption. Some of these compute only the probabilities (signal and transition) at the latch outputs, while others also compute the power. The approach in [40] solves directly for the transition probabilities on the present state lines using the Chapman-Kolmogorov equations [33, 34], which is computationally too expensive. Another approach that also attempts a direct solution of the Chapman-Kolmogorov equations was given in [41]. While it is more efficient, it remains quite expensive, so that the largest test case presented contains less than 30 latches.

Better solutions are offered by two recent papers [42, 43], which are based on solving a non-linear system that gives the present state line probabilities, as follows. Let a vector of input probabilities $P_{in} = [p_1, p_2, \ldots, p_n]$ be applied to the combinational logic block *and* let the n input signals be independent. At the outputs of the logic, let the corresponding output node probability vector be P_{out}. The mapping from P_{in} to P_{out} is some non-linear function that is determined by the Boolean function implemented by the logic. We denote this vector-valued function by $F(\cdot)$, so that $P_{out} = F(P_{in})$.

If we now assume that $P_{in} = P$ is the vector of present state probabilities, then we should also have $P_{out} = P$, because the steady-state state line probabilities are constant. If we assume that the state lines are *independent*, this translates to $P = F(P)$. The solution of this non-linear system gives the required state line probability vector P. It is solved using the Newton-Raphson method in [42], and using the Picard-Peano iteration method in [43].

Both techniques also try to correct for the state line independence assumption. In [42], this is done by accounting for m-wise correlations between state bits when computing their probabilities. This requires 2^m additional gates and can get very expensive. Nevertheless, they show good experimental results. The approach in [43] is to *unroll* the combinational logic block k times. This is less expensive than [42], and the authors observe that with $k = 3$ or so, good results can be obtained.

7. Summary and Conclusions

Power estimation tools are required to manage the power consumption of modern VLSI designs during the design phase, so as to avoid a costly redesign process. Since average power dissipation is directly related to the average switching activity inside a circuit, it would not make sense to expect to estimate power without some information about the circuit input patterns. Yet this is what one would like to do in order to qualify a chip with a certain *power rating* that is expected to hold irrespective of the application. We have presented a number of power estimation techniques that are designed to alleviate this *strong pattern-dependence* problem.

The presented techniques are *weakly* pattern dependent since the user is expected to supply some information on the typical behavior at the circuit inputs. This information is usually in the form of probability (average fraction of time that a signal is high) and density (average number of transitions per second). This information is usually much more readily available to designers than specific input patterns are. For instance, it is relatively easy for a designer to estimate average input switching frequencies, say by looking at test vector sets, or simply by assuming some nominal average switching frequency based on the known clock frequency. The proposed techniques are effective ways of using this information to find the circuit power.

All these techniques use simplified delay models, so that they do not provide the same accuracy as, say, circuit simulation. But they are fast, which is very important because VLSI designers are interested in the power dissipation of large designs. Within the limitations of the simplified delay models, some of these techniques, e.g., the *statistical* techniques, can be very accurate. In fact the desired accuracy can be specified up-front. The other class of techniques, i.e., the *probabilistic* techniques, are not as accurate but can be faster. Two of the proposed probabilistic techniques use BDDs and achieve very good accuracy, but they can be slow and may not be feasible for larger circuits.

From an implementation standpoint, one major difference between probabilistic and statistical techniques is that statistical techniques can be built around existing simulation tools and libraries, while probabilistic techniques cannot. Typically, probabilistic techniques require specialized simulation models. In general, it is not clear that any one approach is best in all cases, but we feel that the second statistical approach (MED) offers a good mix of accuracy, speed, and ease of implementation. It may be that a combination of the different techniques can be used for different circuit blocks. Tables 1 and 2 compare the different characteristics of these techniques.

References

[1] R. W. Brodersen, A. Chandrakasan, S. Sheng, "Technologies for personal communications," *1991 Symp. on VLSI circuits*, Tokyo, Japan, pp. 5–9, 1991.

[2] A. P. Chandrakasan, S. Sheng, and R. W. Brodersen, "Low-power CMOS digital design," *IEEE Journal of Solid-State Circuits*, vol. 27, no. 4, pp. 473–484, April 1992.

[3] Workshop Working Group Reports, Semiconductor Industry Association, pp. 22–23, Nov. 17–19, 1992, Irving, Texas.

[4] S. Chowdhury and J. S. Barkatullah, "Estimation of maximum currents in MOS IC logic circuits," *IEEE Transactions on Computer-Aided Design*, vol. 9, no. 6, pp. 642–654, June 1990.

[5] S. Devadas, K. Keutzer, and J. White, "Estimation of power dissipation in CMOS combinational circuits using Boolean function manipulation," *IEEE Transactions on Computer-Aided Design*, vol. 11, no. 3, pp. 373–383, March 1992.

[6] H. Kriplani, F. Najm, and I. Hajj, "Maximum current estimation in CMOS circuits," *29th ACM/IEEE Design Automation Conference*, Anaheim, CA, pp. 2–7, June 8-12, 1992.

[7] H. J. M. Veendrick, "Short-circuit dissipation of static CMOS circuitry and its impact on the design of buffer circuits," *IEEE Journal of Solid-State Circuits*, vol. SC-19, no. 4, pp. 468–473, Aug. 1984.

[8] A. Shen, A. Ghosh, S. Devadas, and K. Keutzer, "On average power dissipation and random pattern testability of CMOS combinational logic networks," IEEE/ACM International Conference on Computer-Aided Design, pp. 402–407, Santa Clara, CA, November 8–12, 1992.

[9] K. P. Parker and E. J. McCluskey, "Probabilistic treatment of general combinational networks," *IEEE Transactions on Computers*, vol. C-24, pp. 668–670, June 1975.

[10] F. Najm, "Transition density, a stochastic measure of activity in digital circuits," *28th ACM/IEEE Design Automation Conference*, San Francisco, CA, pp. 644–649, June 17–21, 1991.

[11] S. M. Kang, "Accurate simulation of power dissipation in VLSI circuits," *IEEE Journal of Solid-State Circuits*, vol. SC-21, no. 5, pp. 889–891, Oct. 1986.

[12] G. Y. Yacoub and W. H. Ku, "An accurate simulation technique for short-circuit power dissipation based on current component isolation," *IEEE International Symposium on Circuits and Systems*, pp. 1157–1161, 1989.

[13] A-C. Deng, Y-C. Shiau, and K-H. Loh, "Time domain current waveform simulation of CMOS circuits," *IEEE International Conference on Computer-Aided Design*, Santa Clara, CA, pp. 208–211, Nov. 7–10, 1988.

[14] R. Tjarnstrom, "Power dissipation estimate by switch level simulation," *IEEE International Symposium on Circuits and Systems*, pp. 881–884, May 1989.

[15] U. Jagau, "SIMCURRENT - an efficient program for the estimation of the current flow of complex CMOS circuits," *IEEE International Conference on Computer-Aided Design*, Santa Clara, CA, pp. 396–399, Nov. 11-15, 1990.

[16] T. H. Krodel, "PowerPlay - fast dynamic power estimation based on logic simulation," *IEEE International Conference on Computer Design*, pp. 96–100, October 1991.

[17] L. Benini, M. Favalli, P. Olivo, and B. Ricco, "A novel approach to cost-effective estimate of power dissipation in CMOS ICs," *European Design Automation Conference*, pp. 354–360, 1993.

[18] F. Dresig, Ph. Lanches, O. Rettig, and U. G. Baitinger, "Simulation and reduction of CMOS power dissipation at logic level," *European Design Automation Conference*, pp. 341–346, 1993.

[19] M. A. Cirit, "Estimating dynamic power consumption of CMOS circuits," *IEEE International Conference on Computer-Aided Design*, pp. 534-537, Nov. 9-12, 1987.

[20] R. Burch, F. Najm, P. Yang, and D. Hocevar, "Pattern-independent current estimation for reliability analysis of CMOS circuits," *25th ACM/IEEE Design Automation Conference*, Anaheim, CA, pp. 294–299, June 12–15, 1988.

[21] F. Najm, R. Burch, P. Yang, and I. Hajj, "CREST - a current estimator for CMOS circuits," *IEEE International Conference on Computer-Aided Design*, Santa Clara, CA, pp. 204–207, November 7–10, 1988.

[22] F. Najm, R. Burch, P. Yang, and I. Hajj, "Probabilistic simulation for reliability analysis of CMOS VLSI circuits," *IEEE Transactions on Computer-Aided Design*, vol. 9, no. 4, pp. 439–450, April 1990 (Errata in July 1990).

[23] G. I. Stamoulis and I. N. Hajj, "Improved techniques for probabilistic simulation including signal correlation effects," *30th ACM/IEEE Design Automation Conference*, pp. 379–383, 1993.

[24] C-Y. Tsui, M. Pedram, A. M. Despain, "Efficient estimation of dynamic power consumption under a real delay model," *IEEE International Conference on Computer-Aided Design*, Santa Clara, CA, pp. 224–228, November 7–11, 1993

[25] F. Najm, "Transition density : a new measure of activity in digital circuits," *IEEE Transactions on Computer-Aided Design*, vol. 12, no. 2, pp. 310–323, February 1993.

[26] F. Najm, "Improved estimation of the switching activity for reliability prediction in VLSI circuits," *IEEE Custom Integrated Circuits Conference*, pp. 17.7.1–17.7.4, San Diego, CA, May 1–4, 1994.

[27] F. Najm, "Low-pass filter for computing the transition density in digital circuits," *IEEE Transactions on Computer-Aided Design*, vol. 13, no. 9, pp. 1123–1131, September 1994.

[28] A. Ghosh, S. Devadas, K. Keutzer, and J. White, "Estimation of average switching activity in combinational and sequential circuits," *29th ACM/IEEE Design Automation Conference*, pp. 253–259, Anaheim, CA, June 8–12, 1992.

[29] C. M. Huizer, "Power dissipation analysis of CMOS VLSI circuits by means of switch-level simulation," *IEEE European Solid State Circuits Conference*, pp. 61–64, Grenoble, France, 1990.

[30] R. Burch, F. Najm, P. Yang, and T. Trick, "McPOWER: A Monte Carlo approach to power estimation," *IEEE/ACM International Conference on Computer-Aided Design*, Santa Clara, CA, pp. 90-97, November 8–12, 1992.

[31] R. Burch, F. Najm, P. Yang, and T. Trick, "A Monte Carlo approach for power estimation," *IEEE Transactions on VLSI Systems*, vol. 1, no. 1, pp. 63–71, March 1993.

[32] M. Xakellis and F. Najm, "Statistical Estimation of the Switching Activity in Digital Circuits," *31st ACM/IEEE Design Automation Conference*, San Diego, CA, pp. 728–733, 1994.

[33] S. M. Ross, *Stochastic Processes*, New York, NY: John Wiley & Sons, 1983.

[34] A. Papoulis, *Probability, Random Variables, and Stochastic Processes*, 2nd Edition. New York, NY: McGraw-Hill Book Co., 1984.

[35] R. E. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Transactions on Computer-Aided Design*, pp. 677–691, August 1986.

[36] K. S. Brace, R. L. Rudell, and R. E. Bryant, "Efficient implementation of a BDD package," *27th ACM/IEEE Design Automation Conference*, pp. 40–45, June 1990.

[37] S. Chakravarty, "On the complexity of using BDDs for the synthesis and analysis of Boolean circuits," *27th Annual Allerton Conference on Communication, Control, and Computing*, pp. 730–739, Monticello, IL, September 27–29, 1989.

[38] I. Miller and J. Freund, *Probability and Statistics for Engineers*, 3rd edition. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1985.

[39] B. Kapoor, "Improving the accuracy of circuit activity measurement," *31st ACM/IEEE Design Automation Conference*, San Diego, CA, pp. 734–739, June 6–10, 1994

[40] A. A. Ismaeel and M. A. Breuer, "The probability of error detection in sequential circuits using random test vectors," *Journal of Electronic Testing*, vol. 1, pp. 245–256, January 1991.

[41] G. D. Hachtel, E. Macii, A. Pardo, and F. Somenzi, "Probabilistic analysis of large finite state machines," *31st ACM/IEEE Design Automation Conference*, San Diego, CA, pp. 270–275, June 6–10, 1994.

[42] J. Monteiro and S. Devadas, "A methodology for efficient estimation of switching activity in sequential logic circuits," *ACM/IEEE 31st Design Automation Conference*, San Diego, CA, pp. 12–17, June 6–10, 1994.

[43] C-Y Tsui, M. Pedram, and A. M. Despain, "Exact and approximate methods for calculating signal and transition probabilities in FSMs," *ACM/IEEE 31st Design Automation Conference*, San Diego, CA, pp. 18–23, June 6–10, 1994.