Gate-Level Power Estimation Using Tagged Probabilistic Simulation

Chih-Shun Ding, Chi-Ying Tsui, *Member, IEEE*, and Massoud Pedram, *Member, IEEE*

***Abstract*—In this paper, we present a probabilistic simulation technique to estimate the power consumption of a CMOS circuit under a general delay model. This technique is based on the notion of a tagged (probability) waveform, which models the set of all possible events at the output of each circuit node. Tagged waveforms are obtained by partitioning the logic waveform space of a circuit node according to the initial and final values of each logic waveform and compacting all logic waveforms in each partition by a single tagged waveform. To improve the efficiency of tagged probabilistic simulation, only tagged waveforms at the circuit inputs are exactly computed. The tagged waveforms of the remaining nodes are computed using a compositional scheme that propagates the tagged waveforms from circuit inputs to circuit outputs. We obtain significant speed up over explicit simulation methods with an average error of only 6%. This also represents a factor of 2–3× improvement in accuracy of power estimates over previous probabilistic simulation approaches.**

***Index Terms*—Power modeling and estimation, very-large-scale integration (VLSI).**

I. Introduction

POWER dissipation, along with circuit area and speed, emerges as one of the important objectives of IC design. As a result, we see a greater demand for efficient power-estimation techniques and tools. In CMOS circuits, power is largely consumed during charging and discharging of load capacitances. This has led to a simple, yet accurate, power-consumption model at the gate level. Several gate-level power-estimation techniques with reasonable accuracy and high efficiency have been proposed. These techniques can be classified into two categories: *dynamic* (or *simulative*) [1], [2] and *static* (or *nonsimulative*) [3]–[5].

Dynamic techniques explicitly simulate the circuit under a "typical" input vector stream. Their main shortcoming is, however, that they are very slow. Moreover, their results are highly dependent on the simulated sequence. To produce a meaningful power estimate, the required number of simulated vectors is usually high. To address this problem, Monte Carlo simulation techniques are proposed in [1] and [2]. These techniques use an input model based on a Markov process to generate the input stream for simulation. The main difficulty is that it is not clear how the input stream can be efficiently generated when the circuit inputs exhibit complex correlations.

The static techniques [3]–[5] rely on statistical information (such as the mean activity of the input signals and their correlations) about the input stream to estimate the internal switching activity of the circuit. In [3] $(CREST)$, the concept of *probability waveforms* is proposed to estimate the mean and variance of the current drawn by each circuit node. During the simulation, the logic waveforms are compactly represented by probability waveforms, which consist of an initial signal probability and a sequence of events occurring at different time instances. The propagation mechanism for the transition events and their associated transition probabilities is event-driven in nature. In [4] $(DENSIM)$, the notion of *transition density*, which is the average number of transitions per second, is proposed. An efficient algorithm based on Boolean difference operation is proposed to propagate the transition densities from circuit inputs throughout the circuit. Although both of these two techniques can be performed efficiently, the accuracy in general is only moderate, mainly due to the lack of an efficient mechanism to account for the signal correlations among circuit nodes. An implicitly enumeration approach based on *symbolic simulation* is proposed in [5]. While the efficiency of this technique has been improved using ordered binary decision diagram (OBDD) [6], the space complexity is its major limitation.

In this paper, we propose an efficient power-estimation technique, called *tagged probabilistic simulation* (TPS), based on the notion of tagged (probability) waveforms [7], [8]. It works under the general delay models, and therefore does account for the power due to glitches. The tagged probability waveforms are formed by partitioning the waveform space such that logic waveforms produced by all members in a partition are collectively represented by a single tagged waveform. The purpose of this partitioning is to put all logic waveforms with similar properties into the same partition so that the accuracy and efficiency of simulation can be significantly improved. TPS uses the following partitioning strategy: for each node n, all input vectors that produce the same initial and final values in the logic waveforms at node n are put in the same partition, and their corresponding logic waveforms are collectively represented by the same tagged waveform. An efficient waveform propagation scheme is developed to propagate the tagged waveforms from circuit inputs to circuit

Manuscript received September 3, 1997; revised May 12, 1998. This work was supported in part by the Defense Advanced Research Project Agency under Contract F33615-95-C1627 and by the National Science Foundation under PECASE award MIP-9628999. This paper was recommended by Associate Editor E. Macii.

C.-S. Ding is with Rockwell Semiconductor Systems, Newport Beach, CA 92660 USA.

C.-Y. Tsui is with the Department of Electrical and Electronic Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Hong Kong.

M. Pedram is with the Department of Electrical Engineering Systems, University of Southern California, Los Angeles, CA 90089 USA.

Publisher Item Identifier S 0278-0070(98)08588-1.

outputs. In particular, the correlation between the tagged waveforms at the inputs of a gate can be effectively accounted for through the use of correlation coefficients. When the inputs to the circuits are specified in terms of statistical measurements (e.g., signal probabilities, transition probabilities), TPS uses an efficient local OBDD-based technique to extract the correlation coefficients. When the inputs to the circuits are specified in terms of vector streams, a bit-parallel simulation approach is used to extract the correlation coefficients. The advantages of this technique are 1) it is very efficient, as only four tagged waveforms are simulated, and 2) since the logic waveforms represented by each tagged waveform exhibit similar properties, issues such as signal correlations among circuit nodes and glitch generation/propagation can be effectively accounted for. In summary, TPS is intended as a combination of static approaches, which are efficient as only a small number of waveforms are simulated, and dynamic approaches in which the spatiotemporal correlation in input streams is fully captured.

The organization of this paper is as follows. In Section II, we briefly describe the background and terminology. The notion of tagged probability waveforms is introduced in Section III. In Section IV, the waveform computation scheme in TPS is described. The issue of computing correlation coefficients is examined in Section V. The issue of glitch handling is discussed in Section VI, followed by the experimental results and conclusion in Sections VII and VIII, respectively.

II. Background and Terminology

The dominant sources in a correctly designed CMOS circuit are the charging/discharging current of the loading capacitance and the short-circuit current. The former depends on the loading capacitance, while the latter can be modeled as the charging/discharging current of an "equivalent capacitance." Therefore, the power consumption of each node n in the circuit is given by

$$P_n = \frac{1}{2T_c} V_{dd}^2 C_n sw_n \tag{1}$$

where T_c is the clock period, V_{dd} is the supply voltage, C_n is the sum of load capacitance and equivalent short-circuit capacitance at node n, and sw_n is the average switching activity at the output of node n (i.e., the expected number of signal changes) per clock cycle. In (1), we assume that the signal transition is from zero to V_{dd} or vice versa and ignore the impact of signal slew rate on the short-circuit current.

From (1), the problem of estimating the average power can be reduced to one estimating the average switching activity at each node. Therefore, the focus of this paper is the estimation of average switching activities. sw_n in (1) is related to the timing relationship (or delay) of input signals of each gate. Indeed, the output signal of a node may change more than once due to unequal signal arrival times at its inputs. The power consumed by these extra signal changes is generally referred to as the *glitch power*. Here we assume that all signal changes are between V_{dd} and ground and therefore do not consider partial swing. CMOS gates have inertial delays. Only glitches with adequate strength (i.e., glitch width) can overcome the gate inertia and propagate through the gate to its gate output. These two issues are referred to as the modeling of glitch generation and glitch filtering. A delay model that accounts for gate inertia is referred to as the *inertial delay model*. In this paper, we assume the inertial gate delay model. All the glitches with width less than a value specified in the cell library will be suppressed from the waveforms. For the sake of simplicity, we assume that the gate inertial is the same as the gate delay in the discussion of this paper.

One way to cope with the complexity of power simulation is through the concept of probability and tagged (probability) waveforms, as described next.

A. Probability Waveforms

Let us examine the operation of the circuit under a sequence of two input vectors V_{-1}, V_0. Let V_{-1} be the vector applied at time $-\infty$ and V_0 be the vector applied at time zero. Clearly, when vector V_0 is applied, all gates have stablized to their values under V_{-1}. During the time interval from zero to ∞, the logic value of each gate varies over time depending on the signal propagation paths from primary inputs to that gate. Therefore, a large number of distinct logic waveforms may be generated at the output of a gate.

Probability waveforms [3] represent the logic waveforms of each gate collectively using probabilistic measurements. A probability waveform is a sequence of *transition edges* (or *events*) over time where each event is annotated with an occurrence probability. In a probability waveform w, two concepts are employed: *signal probability* and *transition probability*, which are defined as follows. *Signal probability* $sp_n(t)$ is defined as the probability that a node n assumes logic one at time t. The transition probability of an event that changes from zero (one) to one (zero) is defined as the *upward* or *rising transition probability* $tu_n(t)$ (*downward* or *falling transition probability* $td_n(t)$)

The probability waveform of a node is a compact representation of the set of all logical waveforms at that node under the input stimuli between two consecutive clock cycles. In this sense, it is an abstraction of the logical waveform space. If we enumerate all the distinct logic waveforms and their occurrence probabilities for a node n, its probability waveform can be easily constructed.

A probability waveform w can also be represented by an initial signal probability followed by a sequence of transition probabilities in temporal order, that is

$$w = \{sp_n(0), tu_n(t_0), td_n(t_0), \cdots, tu_n(t_m), td_n(t_m)\}.$$

Note that $sp_n(t_+) = sp_n(t_-) + tu_n(t) - td_n(t)$.The main shortcoming of simulation techniques based on these probability waveforms is that signal correlations caused by reconvergent signal paths in the circuit are difficult to account for. The notion of tagged waveforms is proposed here to alleviate this shortcoming.

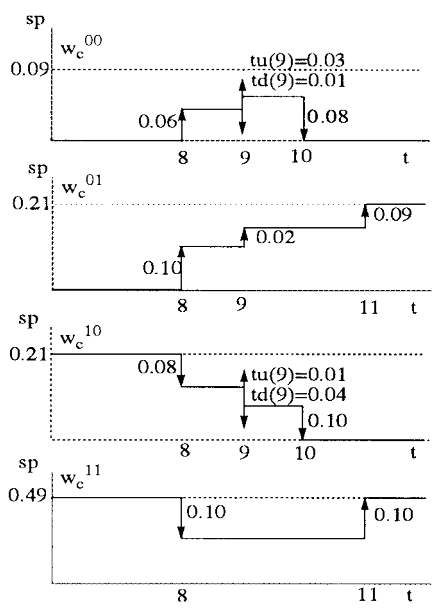

Fig. 1. Tagged waveforms.

III. Tagged Waveforms

A *tagged (probability) waveform* is obtained by partitioning a probability waveform according to the initial and final steady-state logic values of the logical waveforms that have contributed to this probability waveform.[1] That is, four tagged waveforms can be defined at each node n: $w_n^{00}, w_n^{01}, w_n^{10}$, and w_n^{11}. All logic waveforms at the output of node n with initial state x and final state y are compactly represented by the tagged waveform w_n^{xy}. Furthermore, $tu_n^{xy}(t)(td_n^{xy}(t))$ represents rising (falling) transition probability at time t in the tagged waveform w_n^{xy}. Similarly, $sp_n^{xy}(t)$ represents the signal probability at time t in tagged waveform w_n^{xy}. See Fig. 1 for an example of the tagged waveforms at some node c.

Definition III.1: Tagged waveform probability, denoted as $P(w_n^{xy})$, is the sum of occurrence probabilities of all logic waveforms represented by w_n^{xy}.

Note that the initial and final state values are not affected by the delay model. As a result, $P(w_n^{xy})$ can be computed using the zero delay logic simulation.

Let W denote the maximum number of events in any w_n^{xy} where $x, y \in \{0, 1\}$. From the tagged waveforms of node n, its power consumption can be computed as

$$P_{av}(n) = \frac{1}{2T_c} V_{dd}^2 C_n \sum_{x=0}^{1} \sum_{y=0}^{1} \sum_{k=0}^{W} (td_n^{xy}(t_k) + tu_n^{xy}(t_k)) \quad (2)$$

where T_c, V_{dd}, C_n, and sw_n are as defined in (1).

A. Extremal Condition

Definition III.2: A tagged waveform w_c^{xy} is said to satisfy the *extremal condition* at time t if $sp_c^{xy}(t) = 0$ or $sp_c^{xy}(t) = P(w_c^{xy})$.

[1]This definition can be generalized to any partitioning scheme of the probability waveforms.

When a tagged waveform w^{xy} satisfies extremal condition at both times t_- and t_+, there are four possible cases, which will be referred to as cases E1)–E4) in this paper.

Proposition III.1: If a tagged waveform w^{xy} satisfies the extremal condition at times t_- and t_+, then $tu^{xy}(t)$ and $td^{xy}(t)$ are uniquely determined as follows.

E1) *0-holding* case: if $sp^{xy}(t_-) = sp^{xy}(t_+) = 0$, then $tu^{xy}(t) = td^{xy}(t) = 0$.

E2) *1-holding* case: if $sp^{xy}(t_-) = sp^{xy}(t_+) = P(w^{xy})$, then $tu^{xy}(t) = td^{xy}(t) = 0$.

E3) *Rising-transition* case: if $sp^{xy}(t_-) = 0$ and $sp^{xy}(t_+) = P(w^{xy})$, then $td^{xy}(t) = 0$ and $tu^{xy}(t) = P(w^{xy})$.

E4) *Falling-transition* case: if $sp^{xy}(t_-) = P(w^{xy})$ and $sp^{xy}(t_+) = 0$, then $td^{xy}(t) = P(w^{xy})$ and $tu^{xy}(t) = 0$.

Proof: E1) and E2) are easy to prove. We thus show the proof for E3) and E4) only. If $sp^{xy}(t_-) = 0$ $(sp^{xy}(t_-) = P(w^{xy}))$, all logic waveforms represented by w^{xy} must assume logic 0 (1) at time t_-. Therefore, w^{xy} can only have a rising (falling) transition. ■

As the transition probabilities in the above cases are uniquely determined and are either zero or the tagged waveform probability, one may ask if the transition probabilities are also exact. In Section IV-C, we will show that extremal conditions indeed imply the exactness of tagged waveforms. In the next section, we describe the waveform propagation scheme in TPS.

IV. Waveform Propagation

The waveform propagation mechanism used in TPS is similar to the one used for probability waveforms except that, at each node, TPS deals with four tagged waveforms instead of a single probability waveform. Consider a two-input AND gate with inputs a and b and gate output c. During the waveform propagation, there are four tagged waveforms w_a^{xy} (w_b^{wz}) at input a (b), where $x, y, w, z \in \{0, 1\}$. In other words, there are 16 tagged waveform combinations at the gate inputs. Each of these combinations is referred to as the (input) *joint tagged waveform* and denoted by $w_c^{xy,wz}$. They form a disjoint partitioning of the logic waveforms produced at the gate output c. For instance, $w_c^{01,10}$ represents the logic waveforms produced at gate output c when logic waveforms at input a assume initial state 0 and final state 1 and logic waveforms at input b assume initial state 1 and final state 0.

In the following, we consider two cases: 1) tree circuits and 2) circuits with reconvergent paths to illustrate the computation of transition probabilities in TPS. In these cases, we assume no gate inertial and ignore the effect of glitch filtering, which will be discussed in Section VI.

A. Tree Circuits

For circuits with a tree structure, signals connected to two inputs of the AND gate are uncorrelated if the circuit inputs are uncorrelated. Therefore, the transition probabilities can be

TABLE I
THE FORCING SET TABLE FOR A TWO-INPUT AND GATE

OUTPUT TAGS	INPUT TAGS
00	(00,00), (00,01), (00,10),(00,11),(01,00) (01,10),(10,00),(10,01),(11,00)
01	(01,01), (01,11),(11,01)
11	(11,11)
10	(10,10), (10,11), (11,10)

calculated as

$$tu_c^{xy,wz}(t+d) = tu_a^{xy}(t)sp_b^{wz}(t_+) + tu_b^{wz}(t)sp_a^{xy}(t_+) - tu_a^{xy}(t)tu_b^{wz}(t) \quad (3)$$

$$td_c^{xy,wz}(t+d) = td_a^{xy}(t)sp_b^{wz}(t_-) + td_b^{wz}(t)sp_a^{xy}(t_-) - td_a^{xy}(t)td_b^{wz}(t) \quad (4)$$

where d is the gate delay, a and b are inputs of an AND gate, and c is the gate output. The reason to have t_+ and t_- on the right-hand side of the above equations will become clear later in the proof.

The procedure for combining 16 joint tagged waveforms can be described as follows. Since each joint tagged waveform already specifies the initial and final states of the logic waveforms at each gate input, we can easily derive the initial and final states of the logic waveforms produced at the gate output. All the joint tagged waveforms that produce the same initial and final state values at the gate output are added together. After this procedure, four tagged waveforms are formed and are propagated to the fanout gates. Table I, which is referred to as the *forcing set table,* lists the tags of the joint tagged waveforms (for a two-input AND gate) that should be combined into each output tagged waveform.

An OR gate is equivalent to an AND gate with input and output phase inversion. With phase inversion, we mean that w^{00} becomes w^{11}, all rising transition events become falling transition events, all falling transition events becomes rising transition events, etc.

To avoid an exponential increase in the number of joint tagged waveforms, gates with more than two inputs are decomposed into subnetworks of two-input gates AND gates and inverters. This subsection will be concluded with the following theorem.

Theorem IV.1: If the circuit has a tree structure that consists of only simple gates and inverters and circuit inputs are spatially uncorrelated, the transition probability calculations based on (3) and (4) and phase inversion are exact.

Proof: We prove this theorem by induction on levels of nodes. Simple gates with more than two inputs are decomposed into a subnetwork of two-input simple gates. The delay of the root gate in each subnetwork is taken to be the same as the delay of an original gate, while the delays of remaining gates in the subnetwork are assigned to be zero. In the following, we only prove the case for two-input AND gates; the proof for inverters is trivial as it only involves phase inversion.

We assume that all the circuit input nodes are at level $l = 1$. Since the tagged waveforms are computed exactly on these nodes, the theorem holds for $l = 1$. Assume that the theorem holds for $l = k$. Let c be a gate at level $l = k + 1$, with gate inputs a and b. Without loss of generality, consider the joint tagged waveform $w_c^{xy,wz}$. d is the gate delay of c. For input a, there are four possible combinations of values that a can assume at time t_- and t_+. Let p_a^{00}, p_a^{01}, p_a^{10}, and p_a^{11} represent the occurrence probabilities of these four combinations. From the transition probabilities $tu_a^{xy}(t), td_a^{xy}(t)$, and $sp_a^{xy}(t_-)$, these four occurrence probabilities can be computed as

$$\begin{aligned} p_a^{00}(t) &= 1 - sp_a^{xy}(t_-) - tu_a^{xy}(t) \\ p_a^{01}(t) &= tu_a^{xy}(t) \\ p_a^{10}(t) &= td_a^{xy}(t) \\ p_a^{11}(t) &= sp_a^{xy}(t_-) - td_a^{xy}(t). \end{aligned}$$

Similarly for input b. Since the circuit inputs are spatially uncorrelated and the circuit has a tree structure, a and b also must be spatially uncorrelated. $tu_c^{xy}(t+d)$ and $td_c^{xy}(t+d)$ are computed as

$$\begin{aligned} tu_c^{xy,wz}(t+d) &= p_a^{01}(t)p_b^{11}(t) + p_a^{01}(t)p_b^{01}(t) + p_a^{11}(t)p_b^{01}(t) \\ &= tu_a^{xy}(t)sp_b^{wz}(t_+) + tu_b^{wz}(t)sp_a^{xy}(t_+) \\ &\quad - tu_a^{xy}(t)tu_b^{wz}(t) \\ td_c^{xy,wz}(t+d) &= p_a^{10}(t)p_b^{11}(t) + p_a^{10}(t)p_b^{10}(t) + p_a^{11}(t)p_b^{10}(t) \\ &= td_a^{xy}(t)sp_b^{wz}(t_-) + td_b^{wz}(t)sp_a^{xy}(t_-) \\ &\quad - td_a^{xy}(t)td_b^{wz}(t). \end{aligned}$$

Therefore, it proves the case for $l = k + 1$, which in turn proves the theorem by mathematical induction. ■

B. Circuits with Reconvergent Fanout Paths

For circuits with reconvergent fanout paths, exact computation of the transition probability of an event is very difficult, as explained next. A tagged waveform describes the occurrence probabilities of four different states—staying at zero, making a rising transition, making a falling transition, and staying at one—at any time instance. To exactly calculate the output tagged waveform from a joint input tagged waveform in a two-input AND gate, we need to know the correlation between any two states at the same time instance of the input tagged waveforms from different gate inputs.

Fig. 2(a) shows a simple circuit with reconvergent fanout paths. All logic waveforms at nodes e and f and their occurrence probabilities are shown in Fig. 2(b) and (c), assuming 0.5 signal and transition probabilities at nodes a, b, and c. Fig. 2(d) shows the joint tagged waveform $w_{ef}^{01,10}$. Fig. 2(e) shows joint logic waveforms with nonzero occurrence probability that are represented by $w_{ef}^{01,10}$. To compute the output tagged waveform at g under this joint tagged waveform, we need to know the correlations of the following joint states: 1) at time 2, the joint state in which e makes a rising transition and f stays at one, and 2) at times 3 and 4, the joint state in which e stays at one and f makes a falling transition. Although the joint states under consideration at times 3 and 4 are the same, their correlations are different as shown in Fig. 2(e).

The above example demonstrates that the correlation of the same joint state may change with time. Unless all joint logic waveforms exhibit the same correlation, it is difficult accurately to estimate the correlations of the joint states. These possibly time-variant correlations will be referred to

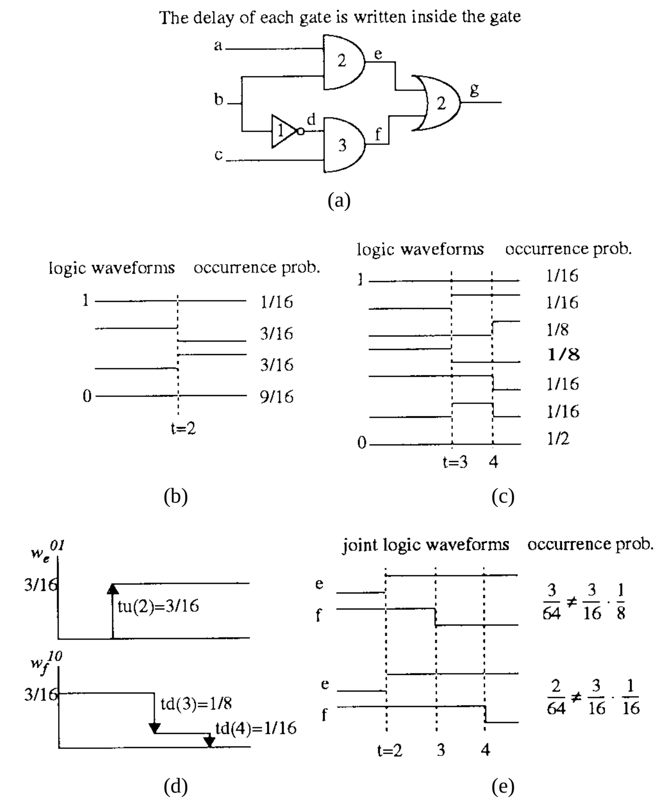

Fig. 2. An example.

as *microscopic correlations*. More precisely, microscopic correlations refer to the correlations between joint events that are present on different logic waveforms. As the time changes, these correlations may change because the joint event under consideration may move from one set of logic waveforms to another.

Modeling of microscopic correlations is very difficult for circuits with reconvergent paths. In contrast, *macroscopic correlations* are the correlations between the tags of the input tagged waveforms that constitute the joint tagged waveforms. These macroscopic correlations are encapsulated by *correlation coefficients,* which are defined as, for w_a^{xy} and w_b^{wz}

$$\kappa^{xy,wz} = \frac{P(w_a^{xy} \wedge w_b^{wz})}{P(w_a^{xy})P(w_b^{wz})} \tag{5}$$

where $w, x, y,$ and $z \in \{0, 1\}$.

TPS uses macroscopic correlations to approximate microscopic correlations. That is, we essentially assume that correlations between any element from set $\{tu_a^{xy}(t), td_a^{xy}(t), sp_a^{xy}(t)\}$ and any element from set $\{tu_b^{wz}(t), td_b^{wz}(t), sp_b^{wz}(t)\}$ are equal to $\kappa^{xy,wz}$. Therefore, the transition probabilities can be computed as

$$\begin{aligned} tu_c^{xy,wz}(t+d) = \kappa^{xy,wz}\{&tu_a^{xy}(t)sp_b^{wz}(t_+) \\ &+ tu_b^{wz}(t)sp_a^{xy}(t_+) - tu_a^{xy}(t)tu_b^{wz}(t)\} \end{aligned} \tag{6}$$

$$\begin{aligned} td_c^{xy,wz}(t+d) = \kappa^{xy,wz}\{&td_a^{xy}(t)sp_b^{wz}(t_-) \\ &+ td_b^{wz}(t)sp_a^{xy}(t_-) - td_a^{xy}(t)td_b^{wz}(t)\} \end{aligned} \tag{7}$$

where d is the gate delay, $w, x, y, z \in \{0, 1\}, a, b$ are the inputs of an AND gate, and c is the output.

The rest of the propagation scheme is exactly the same as in the case of tree circuits. While the above scheme is only an approximation to model the complex microscopic correlations among logic waveforms, it has a few desirable properties. First, the correlation coefficient $\kappa^{xy,wz}$ can be computed under the zero-delay model using approaches proposed in [9] and [10], which can account for both spatial and temporal correlations in circuit inputs. In particular, $\kappa^{xy,wz}$ is identical to $TC^{xy,wz}$, defined in [9].

Second, if the circuit inputs are free of any temporal correlations, calculation of correlation coefficients can be greatly simplified to no more than that of calculating the signal probabilities under the zero-delay model. Under the temporal independence assumption, (5) can be rewritten as

$$\kappa^{xy,wz} = \frac{P(a = x \wedge b = w)P(a = y \wedge b = z)}{P(a = x)P(b = w)P(a = y)P(b = z)}. \tag{8}$$

$P(a = x \wedge b = w)$ can be calculated directly from signal probabilities of $a, b,$ and c using a tabular method [8].

Third, the scheme is consistent in the sense that initial and final signal probabilities of each tagged waveform are exactly calculated if $\kappa^{xy,wz}$'s are exact. Moreover, $sp_c^{xy,wz}(t+d)$ can be calculated as

$$sp_c^{xy,wz}(t+d) = \kappa^{xy,wz} sp_a^{xy}(t) sp_b^{wz}(t). \tag{9}$$

Last, but not least, TPS calculates the power consumption due to functional activity exactly, since only the glitch power is approximated. Therefore, the power estimates obtained by TPS are never less than those obtained from the zero-delay logic simulation. Note that without using tagged waveforms, there is no assurance that the estimated power will be greater than the zero-delay power.

C. Exactness Conditions of TPS

We show that if an output tagged waveform w^{xy} satisfies extremal condition at time t, signal probability $sp^{xy}(t)$ is exact.

Lemma IV.2: Let c be a two-input AND gate with gate inputs a and b.[2] In tagged waveform w_c^{xy}, where $x, y \in \{0, 1\}$, the signal probability $sp^{xy}(t+d)$ is exactly calculated using (9) when w_c^{xy} satisfies the extremal condition at time $t+d$, and κ's are calculated exactly for all transitive fanin gates of c.

Proof: We prove this lemma by induction on levels of nodes l. We assume that all the circuit input nodes are at level $l = 1$. Since extremal conditions are true on these nodes at all times other than time 0, and the tagged waveforms are computed exactly on these nodes, the theorem is true for $l = 1$. Assume that the theorem is true for $l = k$. Let us now consider a node c at level $l = k + 1$ with gate inputs a and b. First, consider the case when $sp_c^{xy}(t+d) = 0$. Note that this implies that each of the joint tagged waveforms $w_c^{pq,rs}$'s that are combined into w_c^{xy} will have $sp_c^{pq,rs}(t+d) = 0$.

[2]We assume that the network has been decomposed into two-input AND gates and inverters. The theorem applies to the network after decomposition.

From (9), it implies that either $\kappa^{pq,rs} = 0$ or at least one of $sp_a^{pq}(t)$ and $sp_b^{rs}(t)$ is zero. For the former case, $sp_c^{pq,rs}(t+d)$ is exact. For the later case, Without loss of generality, we assume $sp_a^{pq}(t) = 0$. Since node a is at level k or less and $sp_a^{pq}(t) = 0$ satisfies extremal condition, tagged waveform w_a^{pq} is exact at time t, and all logic waveforms represented by w_a^{pq} assume zero at time t. As a result, no logic waveforms at c represented by $w_c^{pq,rs}$ can produce logic 1 at time $t+d$, and $sp_c^{pq,rs}(t+d) = 0$ is exact. The proof for the case when $sp_c^{xy}(t+d) = P(w_c^{xy})$ is similar.

We only prove the case for AND gates, as inverters only invert the tagged waveforms and phase inversion does not affect the exactness of the signal probability calculation. It completes the proof for $l = k+1$. The theorem is true by mathematical induction. ■

Corollary IV.3: Let c be a two-input AND gate with gate inputs a and b. In tagged waveform w_c^{xy}, where $x, y \in \{0, 1\}$, the transition probabilities of transition events at time $t+d$ are exactly calculated using (6) and (7), when w_c^{xy}'s at time $t+d$ satisfy one of the conditions described in Proposition III.1, and κ's are calculated exactly for all transitive fanin gates of c.

Proof: From Lemma IV.2, signal probabilities $sp^{xy}(t_- + d)$ and $sp^{xy}(t_+ + d)$ are exact. From Proposition III.1, the transition probabilities $tu^{xy}(t+d)$ and $td^{xy}(t+d)$ are unique and exact. ■

An interesting case is when the total switching activity sw_c of a node c estimated from TPS equals its functional switching activity, e.g., $sw_c = 2 \cdot P(w_c^{01})$. Consider Fig. 1, where w_c^{00} and w_c^{11} are replaced by constant $sp = 0$ and $sp = 0.49$ lines, respectively. In $w_c^{10}, tu(9) = 0$ and $td(9) = 0.03$, whereas w_c^{01} remains unchanged. First, note that w_c^{01} and w_c^{10} may not be exact. However, we only need to show there is no one to zero (zero to one) transition at any time in the logic waveforms represented by w_c^{01} (w_c^{10}), as in the following theorem, to prove sw_c is exact.

Theorem IV.4: Let c be a two-input AND gate with gate inputs a and b. If $sw_c = 2 \cdot P(w_c^{01})$, then sw_c is exact provided that κ's are calculated exactly for all transitive fanin gates of c.

Proof: Note that $sw_c = 2 \cdot P(w_c^{01})$ is the minimum activity that c can have. Consequently, there are no glitch activities in the tagged waveforms. We need to show that there will be no rising (falling) transition in the logic waveforms at time t if $tu(t) = 0\,(td(t) = 0)$. We prove this theorem by contradiction.

The theorem is true for all circuit inputs, which are at level $l = 1$. Assume this theorem is true for $l = k$. Consider a node c at level $l = k+1$. and only w_c^{01} (the proof for the other tagged waveforms can be similarly derived). Assume there exists a falling transition in the logic waveform that occurs at time $t+d$, where d is the gate delay of c. There are two possible cases: 1) both inputs have a falling transition at time (t) or 2) only one input has a falling transition at time t. From the proof for Theorem IV.1

$$p_a^{10}(t)p_b^{11}(t) + p_a^{10}(t)p_b^{10}(t) + p_a^{11}(t)p_b^{10}(t) = 0$$

where (xy, wz) is one of the joint input tags that produces the output tag 01.

For case 1), the term $p_a^{10}(t)p_b^{10}(t)$ needs to be zero. Without loss of generality, assume $p_a^{10}(t) = 0$. Since input a is at level k or less, no logic waveform of a can have a falling transition at time (t), a contradiction.

For case 2), without loss of generality, assume that input a has a falling transition at time t. Again, the term $p_a^{10}(t)p_b^{10}(t)$ needs to be zero. If $p_a^{10}(t) = 0$, it leads to contradiction for the reason described in case 1). Therefore, $p_a^{10}(t) \neq 0$ and $p_b^{10}(t) = 0$. Since $td_c^{xy,wz}(t) = 0$, we must have $p_b^{11}(t) = 0$. Consequently, $sp_b^{wz}(t_-) = p_b^{11}(t) + p_b^{10}(t) = 0$, and it satisfies the extremal condition at time (t_-), which implies all logic waveforms represented by w_b^{wz} will assume logic 0 at time (t_-), and c will assume logic 0 at time t_- and cannot make a falling transition at time $t+d$. This completes the proof. ■

V. Computing the Correlation Coefficient

The correlation coefficient $\kappa^{xy,wz}$ in (5) can be computed exactly or approximately as described next.

A. Exact Techniques

Global OBDD's refer to the OBDD's that represent the logic function of a circuit node in terms of the variables associated with circuit inputs. Using the technique proposed in [11], the temporal correlation of circuits can be accounted for exactly. Reference [9] gives an exact technique that accounts for temporal and pairwise spatial correlations among circuit inputs. However, the space complexity of global OBDD's limits these techniques to small circuit sizes.

Very often, the input stimuli are specified by a vector stream (e.g., from simulation results obtained at register-transfer level or higher). In this case, a bit-parallel simulation technique under the zero-delay model [12] can be employed to calculate the correlation coefficients. On a SUN SS-20 machine, the run time of a bit-parallel algorithm that computes all 16 correlation coefficients can run as fast as 600-K gate-vector/s. Bit-parallel simulation can directly capture both temporal and spatial correlations in the circuit inputs. However, the main drawback is that it cannot handle extremely long streams in a reasonable time.

B. Approximate Technique

Local OBDD's refer to OBDD's that represent the logic function of a circuit node in terms of the variables associated with some set of intermediate variables (associated with nodes in the fanin cone of the node in question). When the signal probability is computed from a local OBDD, the OBDD variables are assumed to be spatially uncorrelated. Several researchers [13], [9], [8] have concluded that the error caused by this assumption is negligible as long as the OBDD variable support set is selected far away (in terms of the level distance) from the node for which the local OBDD is built. This is mainly because signal correlations are the result of reconvergent paths in the circuit; these correlations are stronger for short reconvergent paths compared to long reconvergent paths.

In [14], a technique for computing the signal probabilities using local OBDD's is proposed. The idea is to break the circuit nodes into disjoint parts so that the number of input variables (in the support set) to each part is less than or equal to some user-specified value. However, this approach compromises the accuracy on the nodes at the first few levels of each part since these nodes are too close to the support set. As a result, the effect of short reconvergent paths that pass through the input variables in the support set is not properly captured for the nodes at the first few levels. In summary, this partitioning technique is proposed to limit the size of the support set, and it overlooks the impact of short reconvergent paths on the signal correlations.

In our scheme [8], nodes in the network are first levelized. Then the OBDD variables for each local OBDD (associated with some node n) are selected from the transitive fanins of n that are at least l levels away from n, where l is a user-supplied number. Nodes in the circuit are processed level by level. The advantages of this approach are the following.

1) It maximizes the computation caching (sharing of intermediate results) of OBDD's, as the nodes on the same level will share the same set of variables in the proposed scheme and can be constructed by the same set of OBDD variables and the same variable ordering.
2) It addresses the signal correlation problem, as the function of a node is built in terms of variables that are at least l levels away from it, and thus the short reconvergents are always taken into account and the computation accuracy is roughly the same for all the nodes in the circuit.

In practice, we have found that $l = 6$ on the decomposed network (see Section IV) works very well for most of the circuits that we have tested. Increasing l beyond six does not improve the accuracy of the total network power estimate by much.

VI. Dealing with Glitches

Glitch filtering refers to the process of "adjusting the transition probabilities" in a joint tagged waveform to account for the fact that short glitches in the logic waveforms do not pass through gates due to the inertial delay of these gates. From our experience, as well as that reported in [15], the power dissipation of some data-path circuits (e.g., multipliers) without any glitch filtering can be overestimated by as much as 200%. Glitch filtering is a complicated task, as two transitions in a probability waveform that constitute the same glitch may be correlated.

Another importance issue in glitch filtering is to determine the minimum glitch width that enables it to propagate through a gate. This information can be obtained from the propagation delay and rise/fall times of a gate. In general, we found that using gate delay as the minimum glitch width yields satisfactory results.

The proposed glitch-filtering scheme is based on observations from logic simulations. Consider a two-input AND gate with inputs a, b and output c. Without loss of generality, we assume a single $1 \rightarrow 0$ or $0 \rightarrow 1$ transition at each of a and b. Furthermore, we assume that the arrival time of the transition at input a is earlier than that at input b, and the skew between the two transitions is smaller than the gate inertial delay. Out of the four possible transition combinations, only the case where input a has a rising transition and its arrival time is earlier than that of a falling transition at input b requires filtering.

Glitch filtering on joint tagged waveform $w_c^{xy,wz}$ of an AND gate c with delay d can be described as follows. For a rising transition event $tu_a^{xy}(t_1)$, all of the falling transition events $td_b^{wz}(t_2)$ that 1) come from the other gate input and 2) $t_1 < t_2 \leq t_1 + d$, where d is the gate delay, are subject to glitch filtering. By glitch filtering, we mean that $\kappa^{xy,wz} tu_a^{xy}(t_1) td_b^{wz}(t_2)$ is subtracted from both $tu_c^{xy,wz}(t_1 + d)$ and $td_c^{xy,wz}(t_2 + d)$. In essence, we assume that $tu_a^{xy}(t_1)$ and $td_b^{wz}(t_2)$ are also correlated by macroscopic correlations of the joint tagged waveforms. This scheme is applied to each joint tagged waveform before they are merged into four tagged waveforms using the forcing set table. Note that this scheme does not perform any filtering on short glitches that come from the same gate input. In practice, we found that it produces reasonably good results.

Compared to other probabilistic techniques [3] that only use one probability waveform for each node, TPS offers better opportunity for glitch filtering due to the disjointness property of the joint tagged waveforms. To be more precise, during waveform propagation, the 16 joint tagged waveforms form a disjoint partition of the logic waveform space, and only the transitions in the same joint tagged waveform are subject to glitch filtering.

VII. Experimental Results

The proposed algorithm has been implemented under the SIS environment [16]. TPS can be divided into three phases: network decomposition, correlation coefficient calculation, and waveform propagation. During the correlation coefficient calculation phase, one can use either a local OBDD or bit-parallel simulation approach.

In the experiments, we compare the power estimates of TPS to those obtained from gate-level logic simulation. Circuits used in the experiments are taken from ISCAS85 and MCNC91 benchmarks and are first mapped to the *lib2.sis* library. The delay of each gate is calculated using the delay calculator in SIS [16]. The glitch-filtering scheme in both the logic simulator and TPS is such that all glitches with width smaller than the gate delay are filtered out. The experiments are performed on two types of sequences. The first sequence—an input sequence of 40 000 vectors—is generated randomly by assuming 0.5 signal and transition probabilities at circuit inputs. The second sequence of length 4000 is a nonrandom one obtained from industry and will be referred to as the biased sequence in this section. The average bit activity for this sequence is 0.32.

We first assess the node-by-node accuracy of three different approaches on the random sequence.

1) TPS_NC (no correlation): the transition probabilities at each gate are calculated directly from the transition probabilities at gate inputs. Correlations between gate inputs are ignored. This approach is similar to the one

TABLE II
NODE-BY-NODE PERCENTAGE ERROR COMPARISON (RANDOM SEQUENCES)

	TPS_NC		TPS_BDD		TPS_BP	
ckt	avg	rms	avg	rms	avg	rms
apex6	9.0	17.5	4.8	9.9	3.0	6.9
dalu	16.7	26.9	4.8	10.0	3.2	6.7
des	10.8	17.6	4.4	9.2	2.9	6.1
i8	8.9	20.6	6.1	16.2	1.9	4.0
i10	15.4	24.1	9.4	18.2	6.3	13.1
pair	8.5	18.5	4.8	10.3	3.0	8.0
t481	31.5	47.3	4.9	12.1	2.0	4.8
C432	11.0	15.8	5.9	11.6	1.4	3.2
C880	8.6	15.4	5.4	10.2	2.1	6.1
C1355	8.9	16.5	5.0	10.3	3.6	7.5
C1908	17.3	28.6	12.0	19.1	7.1	12.7
C2670	8.3	14.7	4.4	9.5	2.9	6.2
C3540	27.8	47.7	12.7	29.3	7.8	16.9
C5315	14.2	21.3	9.0	14.7	5.5	10.0
C6288	27.2	34.2	20.1	27.9	12.5	21.3
C7552	13.2	22.2	9.9	17.5	7.7	14.4
avg	14.8	24.3	7.7	14.7	4.5	9.2

TABLE III
THE RUN TIME AND TOTAL POWER-ESTIMATION ERROR FOR BENCHMARK CIRCUITS (RANDOM SEQUENCES)

ckt	logic sim.		TPS_NC	TPS_BDD		TPS_BP	
	power (mW)	run time	err. (%)	err. (%)	speed up	err. (%)	speed up
apex6	10.5	146	4.7	1.7	182	0.4	19
dalu	10.7	178	8.4	0.1	93	1.7	18
des	48.4	861	0.6	1.3	132	0.3	22
i8	15.3	205	6.0	3.3	128	1.9	19
i10	37.8	559	13.0	4.4	82	1.0	22
pair	24.2	385	2.3	0.1	124	0.4	23
t481	4.5	112	25.5	1.3	93	0.4	17
C432	3.9	44	2.8	0.7	73	0.2	19
C880	7.2	105	4.2	2.1	116	0.1	27
C1355	8.9	127	12.7	8.7	158	5.2	24
C1908	9.8	134	19.6	17.2	95	8.7	25
C2670	14.6	214	8.0	3.5	152	0.4	25
C3540	25.4	263	7.3	0.1	71	0.7	21
C5315	37.0	452	15.6	11.4	132	5.0	25
C6288	245.6	865	30.9	10.4	38	6.7	17
C7552	57.5	641	12.4	4.3	112	0.3	27
avg			10.8	4.4	111	2.1	22

TABLE IV
RESULTS FOR BIASED SEQUENCES

ckt	logic sim.		TPS_BP		
	power (mW)	run time	error total	error node	speedup
apex6	6.78	14.3	1.0	3.7	5.3
dalu	4.26	15.5	1.0	1.2	4.0
des	28.05	82.4	0.1	3.9	5.5
i8	9.29	19.2	0.1	1.9	4.7
i10	18.09	51.0	2.0	7.0	4.3
pair	16.08	37.1	0.1	4.9	5.5
t481	2.01	10.7	0.1	1.2	4.3
C432	2.11	4.3	6.8	8.1	4.7
C880	4.11	7.9	1.6	4.1	4.3
C1355	5.13	10.6	4.2	4.2	5.3
C1908	6.32	9.8	4.1	5.7	4.0
C2670	8.31	20.8	3.2	5.2	6.1
C3540	15.18	24.6	2.0	8.8	3.8
C5315	20.71	42.3	3.2	7.5	5.5
C6288	65.53	67.6	10.3	25.2	2.6
C7552	34.32	58.3	1.9	11.3	5.0
avg			2.6	6.4	4.7

used in $CREST$, which is a probabilistic simulation technique.

2) TPS_BDD: TPS using local OBDD's with $l = 6$.

3) TPS_BP: TPS using a bit-parallel simulation scheme.

We do not use TPS with global OBDD's due to the excessive memory requirement of this approach. The power estimates from the above three approaches are compared against those obtained from logic simulation over the entire vector sequence.

The results on random sequences are summarized in Tables II and III. In Table II, we report average relative errors on nodes with switching activity (sw) greater than or equal to 0.1. TPS in general does not guarantee good relative errors on low-activity nodes (that is, $sw < 0.1$). However, note that even large average errors on low-activity nodes only result in a relatively small total error in the power estimates. The TPS_BP approach simulates all 40 000 vectors. The errors for the C6288 circuit are high. This is because very high glitch power is observed in this circuit (five times the zero-delay power).

Table III shows run times and accuracy in estimating the total circuit power consumption. The run times in seconds are reported on a SUN UltraSparc2. The operating frequency of the circuits is 20 MHz, and V_{dd} is set to 5 V. Not surprisingly, the error is much smaller than the node-by-node error due to the averaging effect.

It may appear that one can also apply bit-parallel techniques to logic simulation, and therefore the speedup reported in Table III may not be valid. However, applying parallel simulation techniques to logic simulation is possible if no glitch filtering is performed. When implementing the inertial delay model described by this paper to bit-parallel simulation techniques, it requires the 32-bit (or 64-bit) data path to be unpacked to be processed for correct filtering, and this procedure alone may negate the advantage of the bit-parallel simulation technique. Unfortunately, we found (using PowerMill results as the reference) that logic simulation without any glitch filtering considerably overestimates the power dissipation in the circuit, and therefore, bit-parallel simulation cannot be applied to logic simulation engines.

For the biased sequence (Table IV), only the results from TPS_BP are reported. All errors of circuit power estimates are smaller than 10% except for C6288. Again, the glitch power in this circuit is still very high even under the biased sequence. The speedup in TPS is about five, except for C6288, since only 4000 vectors are simulated. The speedup will be higher if longer vectors are simulated as the run time of waveform computation in TPS is relatively insensitive to the vector length. As the vector length increases, the run time of bit-parallel simulation becomes dominant.

To investigate the impact of the sequence length of the biased sequence on the estimation accuracy, we compared results of logic simulation and TPS_BP on the first N clock cycles, with N varying from 93 to 2000. The results are plotted in Fig. 3, where the solid line shows the results of logic simulations. This figure shows that TPS is robust even for short input sequences.

Although the run times of TPS are higher than the probability waveform approach [3] and transition density approach

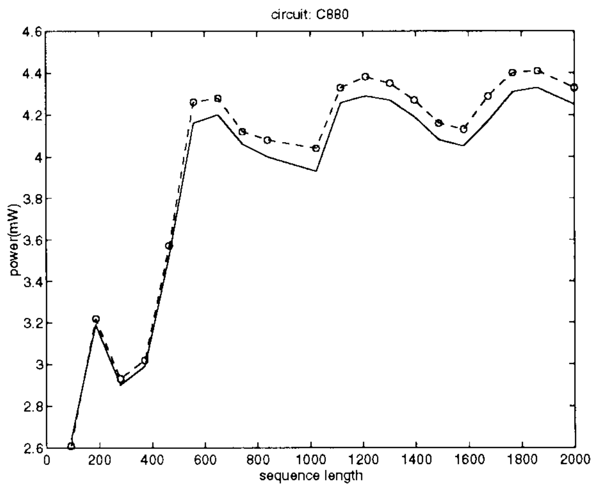

Fig. 3. Impact of the simulation length on total power estimates for the biased sequence.

[4] that consider no gate input correlations, TPS has higher accuracy and the ability to account for the spatiotemporal correlations in the input streams.

VIII. Conclusion

Efficient power-estimation techniques have become very important in today's electronic design automation environments. While this issue can be addressed at different abstraction levels of design, gate-level techniques provide good balance between accuracy and efficiency. In this paper, we presented TPS as an accurate and efficient gate-level power-estimation technique. TPS is based on the notion of tagged waveforms, which divide the logic waveform space into a small number of disjoint parts, and then represents all the logic waveforms in a part by a probability waveform. The advantage of this simulation strategy is that the correlations among circuit internal nodes can be effectively accounted for. Due to the disjointness of the tagged waveforms, an effective filtering scheme was developed and used in TPS. We described the fundamental properties and the waveform propagation schemes of tagged waveforms. We also presented necessary conditions for the transition probability calculation to be exact. Last, the experimental results showed that TPS is more accurate than a probabilistic simulation without considering gate input correlations, and is a factor of ten more efficient than explicit gate-level simulation. TPS is also space-efficient and can easily handle circuits of large size. Although in this paper we did not discuss the application of TPS to FSM circuits, TPS can easily be extended to do so.

References

[1] R. Burch, F. N. Najm, P. Yang, and T. Trick, "A Monte Carlo approach for power estimation," *IEEE Trans. VLSI Syst.*, vol. 1, pp. 63–71, Mar. 1993.

[2] M. Xakellis and F. Najm, "Statistical estimation of the switching activity in digital circuits," in *Proc. 31st Design Automation Conf.*, 1994, pp. 728–733.

[3] F. N. Najm, R. Burch, P. Yang, and I. N. Hajj, "Probabilistic simulation for reliability analysis of CMOS circuits," *IEEE Trans. Computer-Aided Design*, vol. 9, pp. 439–450, Apr. 1990.

[4] F. N. Najm, "Transition density: A new measure of activity in digital circuits," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 310–323, Feb. 1993.

[5] A. A. Ghosh, S. Devadas, K. Keutzer, and J. White, "Estimation of average switching activity in combinational and sequential circuits," in *Proc. 29th Design Automation Conf.*, June 1992, pp. 253–259.

[6] R. Bryant, "Graph-based algorithms for Boolean function manipulation," *IEEE Trans. Comput.*, vol. C-35, pp. 677–691, Aug. 1986.

[7] C-Y. Tsui, M. Pedram, and A. M. Despain, "Efficient estimation of dynamic power dissipation under a real delay model," in *Proc. IEEE Int. Conf. Computer Aided Design*, Nov. 1993, pp. 224–228.

[8] C.-S. Ding and M. Pedram, "Tagged probabilistic simulation provides accurate and efficient power estimates at the gate level," in *Proc. Symp. Low Power Electronics*, Sept. 1995.

[9] D. Marculescu, R. Marculescu, and M. Pedram, "Switching activity analysis considering spatiotemporal correlation," in *Proc. IEEE Int. Conf. Computer Aided Design*, June 1994, pp. 294–299.

[10] S. Ercolani, M. Favalli, M. Damiani, P. Olivo, and B. Ricco, "Estimate of signal probability in combinational logic networks," in *Proc. 1st Eur. Test Conf.*, 1989, pp. 132–138.

[11] P. Schneider and U. Schlichtmann, "Decomposition of Boolean functions for low power based on a new power estimation technique," in *Proc. 1994 Int. Workshop Low Power Design*, Apr. 1994, pp. 123–128.

[12] P. H. Schneider, "PAPSAS: A fast switching activity simulator," in *Proc. PATMOS*, 1995, pp. 350–360.

[13] A. Majumdar and S. Sastry, "Probabilistic characterization of controllability in general homogeneous circuits," *IEEE Trans. Computer-Aided Design*, vol. 25, pp. 76–93, Feb. 1993.

[14] B. Kapoor, "Improving the accuracy of circuit activity measurement," in *Proc. 31st Design Automation Conf.*, 1994, pp. 734–739.

[15] F. N. Najm, "Low-pass filter for computing the transition density in digital circuits," *IEEE Trans. Computer-Aided Design*, vol. 13, pp. 1123–1131, Sept. 1994.

[16] E. M. Sentovich, K. J. Singh, L. Lavagno, C. Moon, R. Murgai, A. Saldanha, H. Savoj, P. R. Stephan, R. K. Brayton, and A. L. Sangiovanni-Vincentelli, "SIS: A system for sequential circuit synthesis," Electronics Research Lab, University of California at Berkeley, Tech. Rep. UCB/ERL M92/41, 1992.

Chih-Shun Ding, for a photograph and biography, see p. 471 of the June 1998 issue of this Transactions.

Chi-Ying Tsui (S'94–M'95) received the B.S. degree in electrical engineering from the University of Hong Kong and the Ph.D. degree in computer engineering from the University of Southern California, Los Angeles, in 1994.

He is an Assistant Professor in the Department of Electrical and Electronic Engineering at the Hong Kong University of Science and Technology. His current work focuses on designing lower power VLSI architectures for multimedia and wireless applications and developing computer-aided design methodologies and techniques for low-power synthesis and power estimation.

Dr. Tsui received the Best Paper Award from the IEEE Transactions on Very Large Scale Integration (VLSI) Systems in 1995. He has served on the technical program committee of a number of conferences and symposiums, including ILSPED, ASP-DAC, and IEEE VLSI.

Massoud Pedram (S'88–M'90), for a photograph and biography, see p. 83 of the February 1998 issue of this Transactions.