Probabilistic Gate-level Power Estimation using a Novel Waveform Set Method

Saeeid Tahmasbi Oskuii
Norwegian University of
Science and Technology
7491 Trondheim, Norway
oskuii@iet.ntnu.no

Per Gunnar Kjeldsberg
Norwegian University of
Science and Technology
7491 Trondheim, Norway
pgk@iet.ntnu.no

Einar J. Aas
Norwegian University of
Science and Technology
7491 Trondheim, Norway
ejaas@iet.ntnu.no

ABSTRACT

A probabilistic power estimation technique for combinational circuits is presented. A novel set of simple waveforms is the kernel of this technique. The transition density of each circuit node is estimated. Existing methods have local glitch filtering approaches that fail to model this phenomenon correctly. Glitches originated from a node may be filtered in some, but not necessarily all, of its successor nodes. Our waveform set approach allows us to utilize a global glitch filtering technique that can model the removal of glitches in more detail. It produces error free estimates for tree structured circuits. For other circuit, experimental results using the ISCAS'85 benchmarks show that the waveform set method generally provides significantly better estimates of the transition density compared to previous techniques.

Categories and Subject Descriptors

B.6.3 [**LOGIC DESIGN**]: Design Aids—*Switching theory*; B.6.1 [**LOGIC DESIGN**]: Design Styles—*Combinational logic*

General Terms

Algorithms

Keywords

probabilistic power estimation, gate-level, probability waveform, combinational logic, transition density

1. INTRODUCTION

The demand for fast and accurate power estimation at early design stages are steadily increasing as energy consumption is becoming a more important design factor. The total power consumption in CMOS circuits results from a combination of dynamic and static sources. In this paper we focus on the dynamic part, caused by charging and discharging of the load capacitances at the outputs of the nodes, i.e., logic gates, in the circuit. Several power estimation techniques have been proposed [1], [2], [3], [4], [5], [6], [7], [8], [9] and [10].

GLSVLSI'07, March 11–13, 2007, Stresa-Lago Maggiore, Italy.

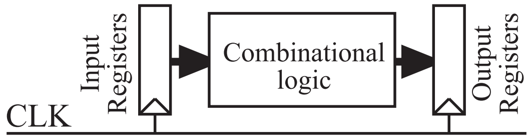


Figure 1: Combinational logic with synchronous primary inputs as the target architecture of power estimator

The average dynamic power consumed in a combinational circuit is given by [11]:

$$P_{av} = \frac{1}{2} V_{DD}^2 \sum_{i=1}^{N} C_i D_i \tag{1}$$

where V_{DD} is the supply voltage, N is the total number of nodes, C_i is the total load capacitance at node i and D_i is the transition density at node i defined as [12]:

$$D_i = \lim_{T \to \infty} \frac{n_i(T)}{T} \tag{2}$$

$n_i(T)$ is the number of transitions in a time interval of length T. Blocks of combinational logic within a synchronous system are very common in digital circuits (Fig. 1). The transitions on the primary combinational inputs then typically appear on the clock edge and the total delay of the combinational circuit is less than the clock period. With these assumptions the transition density D_i in a pure combinatorial circuit can be rewritten as:

$$D_i = f_{clk} \overline{n_i} \tag{3}$$

where $\overline{n_i}$ is the average number of transitions at node i in one clock cycle and f_{clk} is the clock frequency.

Deterministic delay assignments for the gates lead to a finite set of transition times for each node. Assuming a given transition probability at the primary inputs, we get a set S_i of transition times with their probabilities at each node i. D_i can then be specified as follows:

$$D_i = f_{clk} \sum_{j \in S_i} p_j \tag{4}$$

where p_j is the occurrence probability of a transition j. Our objective is to efficiently compute the set of all possible transition times and their corresponding probabilities which can easily provide an estimate of the power consumption.

The probability waveform concept is introduced in a probabilistic current estimator in [13]. The probability waveform is a compact representation of logic waveforms and events happening at different time instances. Tagged probabilistic simulation (TPS) [4] presents an estimation technique based on the notion of tagged waveforms which models the set of all possible events at the output of each circuit node. The origin of the estimation errors in the probabilistic power estimation methods is the glitch filtering and interdependency issues. In the original TPS a simple glitch filtering approach is followed. That is, if a rising transition appears close enough to a falling transition in a tagged waveform, they will both be removed from computations. The glitch filtering of TPS has been improved in [5] by introducing a dual-transition method to consider different combinations of tagged waveforms at a certain node, and filter out extra transitions if necessary. The dual transition method is improved further in [6] by reusing supergate notion from [14] (enclosing reconvergent fanouts) and improving interdependency issues in a circuit.

In this work we present a method that models the set of all possible transitions at the output of each node of the circuit and their occurrence probabilities. We improve glitch filtering to take into account the successor nodes when a glitch is generated, as a glitch might be filtered out in some of the successor nodes. Unlike the previous methods, which consider the glitch filtering locally, our technique considers the glitch filtering globally. This results in accurate estimations for tree-structured circuits and relatively smaller errors for general circuits with reconvergent fanouts.

2. BACKGROUND AND TERMINOLOGY

A digital circuit consists of combinational and sequential parts. The power estimation technique we present here focuses on the combinational circuits as it is shown in Fig. 1. We employ a set of simple waveforms, which is described in this section. As can be seen from Eq. 4, the probability computations are limited to one clock cycle. The clock period is assumed larger than the maximum delay of the combinational logic. When the occurrence probabilities (p_js) in Eq. 4 are being computed, the clock period is set to be infinitely large. The clock's active edge is assumed to appear at time 0. This infers that all nodes have settled to their final values after the previous clock edge at time 0.

Throughout this paper we utilize the term waveform with a special meaning. A waveform W is assumed to have exactly one rising edge and one falling edge. We represent a waveform as a pair of time stamps $\Psi(t_r, t_f)$, where t_r and t_f are the times of the rising edge and falling edge respectively. An occurrence probability is also associated with each waveform denoted as $p(W)$ for waveform W. Figure 2 shows some example waveforms. The zero-holding and one-holding are in our method represented as $\Psi(+\infty, -\infty)$ and $\Psi(-\infty, +\infty)$, respectively. $+\infty$ shows that the transition appears after any imaginable time and will not be considered in power computations. Similarly $-\infty$ shows that the transition appears before any imaginable time and will not be considered in power computations. A waveform W is simple if either t_r or t_f is $+\infty$. A waveform is non-simple if both t_r and t_f are non-infinite times. A non-simple waveform is originally a glitch. A Simple Waveform Set S is a set of simple waveforms and is complete if the sum of the waveform occurrence probabilities is equal to 1, i.e:

$$\sum_{W \in S} p(W) = 1 \tag{5}$$

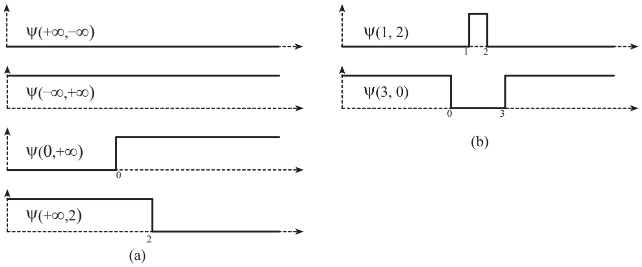


Figure 2: (a) Examples of simple transitions. (b) Examples of non-simple transitions

For simplicity we refer to Simple Waveform Set as SWS. In order to keep SWS small, waveforms with equal rise and fall times are combined, i.e the occurrence probabilities are summed.

The probabilistic power estimation commences by assigning SWSs to input nodes. Four waveforms are included for each node:

$$\text{SWS} = \{W_{11}, W_{10}, W_{01}, W_{00}\} \tag{6}$$

where $W_{11} = \Psi(-\infty, +\infty)$, $W_{00} = \Psi(+\infty, -\infty)$, $W_{01} = \Psi(0, +\infty)$ and $W_{10} = \Psi(+\infty, 0)$ are holding one, holding zero, zero-one transition and one-zero transition respectively. We assume that all the input transitions appear at time 0. Under the independence assumption for input values before and after time 0, occurrence probabilities can be assigned to each waveform. Let p be one-probability at the corresponding input node. Then $1-p$ will be zero-probability at this node and $p(W_{11}) = p^2$, $p(W_{00}) = (1-p)^2$ and $p(W_{01}) = p(W_{10}) = p(1-p)$. Once the SWSs are assigned to the inputs of a certain logic gate, the SWS at the output of that gate can be computed by computing the output waveform for each combination of the input waveforms. In some cases the output waveform may contain two non-infinite transition edges which cannot be considered as simple waveforms. In such cases the waveform will be decomposed into simple waveforms.

Definition - Assume that $W = \Psi(t_r, t_f)$ is a non-simple waveform with $t_r \neq t_f$. DECOMP(W) decomposes W into three simple waveforms:
If $t_r < t_f$, these simple waveforms are $W_1 = \Psi(t_r, +\infty)$, $W_2 = \Psi(+\infty, t_f)$ and $W_3 = \ominus\Psi(-\infty, +\infty)$.
If $t_r > t_f$, they are $W_1 = \Psi(t_r, +\infty)$, $W_2 = \Psi(+\infty, t_f)$ and $W_3 = \ominus\Psi(+\infty, -\infty)$.
Occurrence probabilities for all decomposed simple waveforms are equal to the occurrence probability $p(W)$ of the original non-simple waveform. Waveforms marked by $\ominus$ will be considered with negated occurrence probabilities and they are used to remove the extra transitions generated by the other two waveforms. The negative sign for the occurrence probability nullifies the extra waveforms, as two waveforms with equal rise edge and fall edge can be combined by adding the occurrence probabilities.

An example of decomposition is illustrated in Figure 4. We use a fanout delay model and the numbers inside each gate in Figure 3 represent the inertial delay of that gate. When $\Psi(+\infty, 2)$ and $\Psi(1, +\infty)$ are applied to inputs f and e, respectively, the output at node g will be $\Psi(3, 2)$. This waveform is non-simple and should be decomposed to simple waveforms $\Psi(3, +\infty)$, $\Psi(+\infty, 2)$, and $\ominus\Psi(-\infty, +\infty)$.

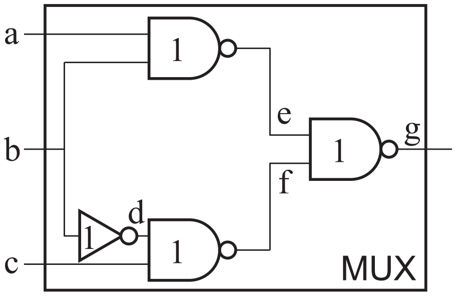


Figure 3: A 2-to-1 MUX example

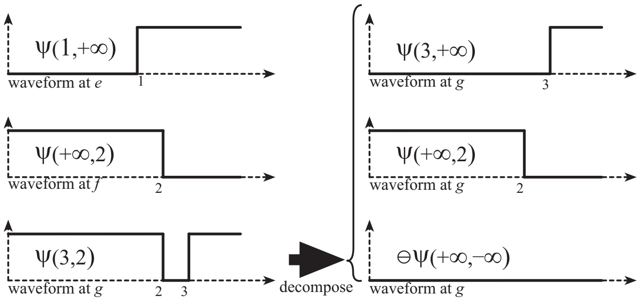


Figure 4: An example non-simple waveform generation and its decomposition to simple waveforms

3. COMPUTATION OF SIMPLE WAVEFORM SETS

For a given node in a combinational circuit, the output SWS is computed using its input SWSs as shown in Algorithm I in Figure 5. The computation for a node can commence if all input SWSs are computed earlier. The list of available nodes is initialized to the primary inputs in the beginning. After completion of the computations for each node, the list of available nodes should be updated by adding the new nodes that can be computed and removing the computed node. After computation of the SWS of a given node i, its transition density is calculated. When SWSs are computed for all nodes having node i as input, the SWS for node i can be removed from memory to reduce the estimator's run-time memory requirement. The waveform $W = \Psi(t_r, t_f)$ resulting from applying W_1 and W_2 to a logic gate f, has the occurrence probability of $p(W_1)p(W_2)$ where $p(W_1)$ and $p(W_2)$ are the occurrence probabilities of W_1 and W_2, respectively. This assumption is correct if W_1 and W_2 are independent. The inertial and transport delay of the logic gate f, denoted as d_i^f and d_t^f respectively, are taken into account by adding the delay values d_i^f and d_t^f to the timetamps of the resulted waveform's rise and fall edges, i.e. t_r and t_f. For non-simple output waveform, if the glitch length $|t_r - t_f|$ is smaller than the inertial delay d_i^f, it will be replaced by constant zero or one depending on the waveform value at $+\infty$. Although our proposed technique is not limited to a particular delay model, we will utilize a fanout delay model. The delays of the gates are assumed to be proportional to their fanout number. The `Simplify` procedure in Algorithm I locates waveforms with equal rise and fall times within the given SWS and replaces them with a waveform with the same rise and fall times and an occurrence probability equal to the sum of their occurrence probabilities.

```
function ComputeSWS ($S_1, S_2$)
OutputSTS=∅;
for all $W_1 \in S_1$
   for all $W_2 \in S_2$
      Compute $W = f(W_1, W_2)$;
      if $W$ is simple then
         Add $W$ to OutputSWS;
      else
         $[W_1, W_2, W_3]$=DECOMP($W$);
         Add $W_1$, $W_2$ and $W_3$ to OutputSWS;
simplify OutputSWS;
return OutputSWS;
```

Figure 5: Algorithm I for computing SWSs

4. DEALING WITH INTERDEPENDENCIES

The problem of interdependencies between different nodes arises in the presence of reconvergent fanouts in a combinational logic circuit. For circuits with reconvergent fanout paths, the exact computation of transition probabilities is a problem of NP-complexity. Several methods try to overcome this problem. [13] uses the supergate concept introduced in [14] and tries to estimate spatial dependencies, while [4] uses macroscopic correlations. Correlation coefficients can also be approximated using local ordered binary decision diagrams (LOBDDs) [15] and [16]. [17] and [10] use Markov Chain models to capture correlations. [7] employs Bayesian networks to simplify the heavy global computation needed to overcome the problem of reconvergent fanouts. The computations are localized to small units, referred to as cliques, assuming zero-delay model. We adopt the concept of macroscopic correlation. Macroscopic correlations are correlations between the different nodes for fixed input values and zero-delay model. Since the values of nodes are assumed to be settled and delay independent at times $+\infty$ and $-\infty$, we use these values for capturing macroscopic correlations. These macroscopic correlations are encapsulated by correlation coefficients, which are defined as:

$$\kappa_{A,B}^{a,b} = \frac{p(A = a \wedge B = b)}{p(A = a)p(B = b)} \tag{7}$$

where A and B are two logic nodes in the circuit and a and b are two logic values ($a, b \in \{0, 1\}$). We can rewrite the correlation coefficients $\kappa_{A,B}^{0,0}$, $\kappa_{A,B}^{1,0}$ and $\kappa_{A,B}^{0,1}$ as a function of $\kappa_{A,B} = \kappa_{A,B}^{1,1}$.

$$\begin{aligned} \kappa_{A,B}^{0,0} &= 1 + \tfrac{p_A p_B (\kappa_{A,B} - 1)}{(1-p_A)(1-p_B)} \\ \kappa_{A,B}^{1,0} &= \tfrac{1 - p_B \kappa_{A,B}}{1 - p_B} \\ \kappa_{A,B}^{0,1} &= \tfrac{1 - p_A \kappa_{A,B}}{1 - p_A} \end{aligned} \tag{8}$$

where p_A and p_B are the one-probability of node A and B, respectively. These correlation coefficients can be computed under the zero-delay model. In [17], Marculescu et al. give a technique for computation of temporal and spatial correlations. This is, however, computationally complex. A method for computing pairwise correlation coefficients recursively is presented by Ercolani et al. in [18]. This method approximates joint probabilities using only pairwise correlation coefficients of all involved signals and ignores higher orders of correlations. However signal probabilities computed using introduce errors to the estimations because of their deviation from accurate computation of signal probabilities. Another source of the error is introduced by utilization of macrocorrelations instead of microcorrelations and ignoring temporal correlations. Microcorrelation provides the accurate correlation of two signals at a particular time with real delay models. We will discuss these errors in Section 6.

In our power estimation method, the correlation coefficients are multiplied with the occurrence probabilities. Assume that W_A and W_B are two waveforms at inputs A and B of gate f. The occurrence probability for the waveform at the output of f is $\kappa p(W_A)p(W_B)$ where

$$\kappa = \kappa_{A,B}^{W_A|_{+\infty}, W_B|_{+\infty}} \cdot \kappa_{A,B}^{W_A|_{-\infty}, W_B|_{-\infty}} \tag{9}$$

As an example let us consider the multiplexer circuit in Figure 3 again. Waveforms $W_1 = \Psi(1, +\infty)$ and $W_2 = \Psi(+\infty, 2)$ are applied to inputs e and f, respectively. A one-probability assignment of 0.5 at the primary inputs gives occurrence probabilities of each of these waveform of $\frac{3}{16}$. The reconverging fanout from node b to node g means that the waveforms at the inputs of node g are not independent. Using Ercolani's method and Eq. 8, gives the following correlation coefficients:

$$\kappa_{e,f}^{1,1} = \frac{8}{9},\ \kappa_{e,f}^{1,0} = \kappa_{e,f}^{0,1} = \frac{4}{3} \text{ and } \kappa_{e,f}^{0,0} = 0 \tag{10}$$

Waveform W_1 at node e is equal to 0 at $-\infty$ and 1 at $+\infty$ since it rises at time 1. Waveform W_2 at node f is equal to 1 at $-\infty$ and 0 at $+\infty$ since it falls at time 2. This gives the following coefficient κ:

$$\kappa = \kappa_{e,f}^{1,0} \kappa_{e,f}^{0,1} = \frac{16}{9} \tag{11}$$

The occurrence probability of the resulting waveform $W = \Psi(3, 2)$ will therefore be

$$p(W) = \kappa p(W_1)p(W_2) = \frac{1}{16} \tag{12}$$

As discussed earlier, the waveform W is non-simple and should be decomposed to simple waveforms.

5. GLITCH FILTERING

A logic gate will remove short glitches due to its inertial delay. Overestimation of power is resulted if glitch filtering is ignored or is imperfect. A glitch will be removed at a certain gate if the glitch length is smaller than the inertial delay of that gate. Glitches generated in a certain node of the circuit might be filtered out in some of the successor nodes. Due to differences in the inertial delays of the successor nodes, we can also have the situation where the glitch is filtered out in some, but not all, of the successor nodes. In order to include glitch filtering in the our computations, we assign a mask tag to each one of the waveforms in the SWS. Let M^W be the mask tag for waveform W. This tag is an array of boolean with the size of the total number of nodes in the system. M_i^W is FALSE if the waveform W will reach node i in the future and will be filtered out at node i; otherwise it is TRUE. Several changes in Algorithm I are needed in order to include glitch filtering. Algorithm II in Figure 6 takes into account the glitch filtering. The `and` operator used for mask computations in Algorithm II is a bitwise `and` operation on two vectors.

```
function ComputeSWS ($S_1, S_2$)
OutputSTS=∅;
i=Current Node ID;
for all $W_1 \in S_1$
   for all $W_2 \in S_2$
      if ($M_i^{W_1}$ and $M_i^{W_2}$)
         Compute $W = f(W_1, W_2)$;
         if $W$ is simple then
            $M^W = M^{W_1}$ and $M^{W_2}$;
            Add $W$ to OutputSWS;
         else
            Compute $newM^W$;
            $[W_1, W_2, W_3, W_4, W_5]$=DECOMP2($W$);
            $M_{1..4}^W = M^{W_1}$ and $M^{W_2}$ and $newM^W$;
            $M_5^W = M^{W_1}$ and $M^{W_2}$;
            Add $W_{1..5}$ to OutputSWS;
simplify OutputSWS;
return OutputSWS;
```

Figure 6: Algorithm II for computing SWSs with glitch filtering

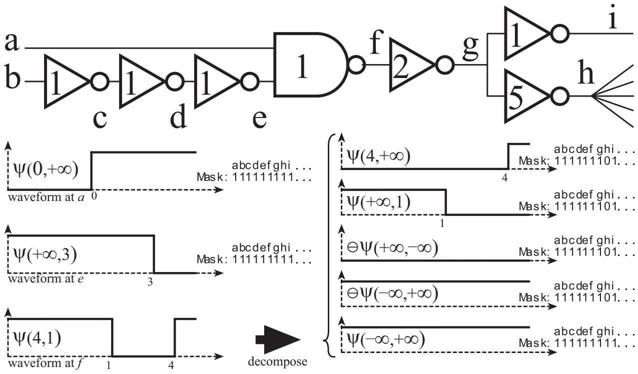


Figure 7: An example of glitch filtering

If $W = \Psi(t_r, t_f)$ is a non-simple waveform, DECOMP2(W) decomposes the non-simple waveform, W, into simple waveforms similar to DECOMP(W). However DECOMP2(W) generates five waveforms $W_{1..5}$. If $t_r < t_f$, they are $W_1 = \Psi(t_r, +\infty)$, $W_2 = \Psi(+\infty, t_f)$, $W_3 = \ominus\Psi(-\infty, +\infty)$, $W_4 = \ominus\Psi(+\infty, -\infty)$ and $W_5 = \Psi(+\infty, -\infty)$. And if $t_r > t_f$, the decomposed waveforms are $W_1 = \Psi(t_r, +\infty)$, $W_2 = \Psi(+\infty, t_f)$, $W_3 = \ominus\Psi(+\infty, -\infty)$, $W_4 = \ominus\Psi(-\infty, +\infty)$ and $W_5 = \Psi(-\infty, +\infty)$.

The `Simplify` procedure is altered to take into account the mask tags as well. Two waveforms with equal rise and fall times can be combined only if they have equal mask tags for the successor nodes of the current node.

A new mask, $newM^W$, is computed for the non-simple waveforms. $newM_i^W$ is FALSE if W can reach node i but cannot pass this node (the inertial delay of this node is larger than the glitch size of W) otherwise $newM_i^W$ will be TRUE. This new mask affects waveform $W_{1..4}$ from the decomposed waveforms, but not waveform W_5. Assume that node j and node k are successors of node i and that node j will pass the non-simple wave W, while node k will filter it out. As a result of Algorithm II, the j node mask tags of the decomposed waveforms $W_{1..5}$ will all be TRUE. Therefore, at node j, waveform W_4 and W_5 will be removed because of the negative sign in the occurrence probability of W_4. Only W_1, W_2 and W_3 will be left which is exactly as algorithm I. However, the node k mask tags are FALSE for waveforms

$W_{1..4}$ so that they will be removed, leaving only waveform W_5. The glitch is consequently filtered out. Let us consider the example shown in Figure 7. A glitch with length 3 will be created at node f due to unequal delays of input paths of node f. The successor nodes i and h have different inertial delays. The glitch with length 3 created in node f will pass through node g and i, while it will be filtered out in node h. Therefore the node h mask tag of waveform $W_{1..4}$ generated at node f will be FALSE while all other mask tags will be FALSE. During the SWS computation for node h, all waveforms with FALSE mask tag value for node h will be removed.

6. EXPERIMENTAL RESULTS

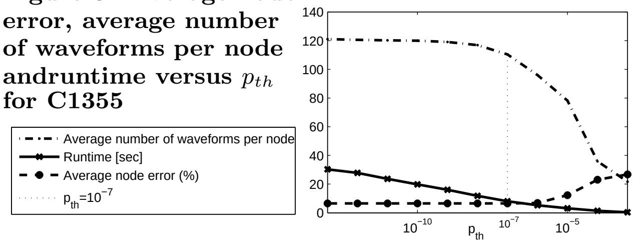
The proposed technique is implemented in C++. This prototype tool includes the procedure for signal probability and correlation coefficient estimation. The inputs to the tool are the netlist, the delay model and one-probabilities of primary inputs. We want to emphasize that any realistic delay model can be used in our system and it not restricted to the fanout model. One-probabilities for primary inputs are in our experiments set to 0.5, but can be any arbitrary value between 0 and 1. The reference transition densities for all experiments are acquired from a circuit logic simulation with 100000 random input vectors that satisfy the one-probabilities of the primary inputs (0.5). A sample tree structured circuit is analyzed in [5] and computation errors are reported. Node errors are up to 42% for the probabilistic simulation method [13] and up to 23% for the TPS method [4]. Dual-transition glitch filtering [5] reduces the computation error to maximum 3%. As opposed to these methods, the computation error for tree structured circuits (no reconvergent fanouts) using our estimation technique is zero. This is because of the complete glitch filtering consideration for tree-structured circuits.

Although our proposed technique has zero computation error for circuits with no reconvergent fanouts, errors are introduced into computations in the presence of reconvergent fanouts due to imperfect dependency considerations. More experiments are carried out using ISCAS'85 benchmark circuits. In order to be able to compare our results with those previously published, we use the fanout delay model: The delay of each gate is proportional to its number of fanouts. For each circuit average node error (E_{av}), standard deviation (σ) and total power error (E_{tot}) are reported in Table 1. E_{av} is the average relative error in node transition density estimation and E_{tot} is relative error in total power. The errors are reported in percentage. The error numbers for TPS, dual transition TPS (referred as TPS-DT) and Enhanced dual transition TPS (referred as TPS-EDT) from [4], [5] and [6] are reported for comparison. On average, and for most cases better results are achieved using our technique. One measurement of quality of an estimate is average error. Another important measure is standard deviation (σ), which measures the uncertainty of the estimate. Our improvement over TPS-EDT, namely σ equals 12.2% versus 16.1%, is significant, because large errors contribute more to the sigma value. TPS-EDT provides better estimation results in some cases which is mainly because of better dependency coverage in this method. As a future extension to SWS we can enclose reconvergent fanouts to form supergates similar to [6]. This will reduce our estimation errors further. From Table 1 and 2 it can be seen that SWS (as well as TPS, TPS-DT and TPS-EDT) in general gives better estimates for circuits with small logic depth. The waveform set at the output of a gate is computed based on its input waveform sets, therefore errors might propagate and accumulate from earlier stages to later stages. However, one should note that practical circuits are typically optimized to avoid excessively large logic depths, so the circuits with large logic depths are mainly used to evaluate the performance of the estimators.

Table 1: Error comparison for ISCAS'85 benchmark circuits with fanout delay assignment. All errors are in percentage.

	TPS			TPS-DT			TPS-EDT			SWS		
Circuit	Eav	σ	Etot	Eav	σ	Etot	Eav	σ	Etot	Eav	σ	Etot
C17	2.3	2.6	0.1	2.3	2.6	0.1	2.3	2.6	0.1	0.7	1.1	0.7
C432	29.9	38.8	35.8	9.5	11.8	6.5	11.5	16.6	11.5	5.2	9.6	2.8
C499	6.8	14.0	7.0	3.6	8.2	0.6	2.3	3.0	3.0	1.0	1.6	0.3
C880	8.3	15.3	1.6	8.0	15.7	5.2	4.8	9.0	0.0	4.4	7.9	2.7
C1355	24.2	31.6	32.9	5.8	11.2	5.4	5.0	9.5	0.5	6.3	9.5	0.2
C1908	15.0	23.1	4.1	17.7	27.9	11.2	7.0	16.3	2.0	7.8	11.1	2.3
C2670	16.6	29.8	7.2	16.7	28.3	9.9	13.2	23.6	6.2	9.7	16.7	2.9
C3540	13.8	26.3	9.8	10.3	25.6	2.4	10.5	26.4	3.7	7.3	17.9	1.5
C5315	11.8	24.4	2.3	13.4	31.5	10.1	11.3	27.0	3.4	7.1	11.5	2.1
C6288	27.4	27.5	32.1	15.7	18.8	4.1	12.7	15.4	0.2	15.7	20.1	4.9
C7552	14.5	27.5	3.2	14.8	31.4	7.8	14.1	27.6	1.3	11.5	27.7	0.9
AVE.	15.5	23.7	12.4	10.7	19.4	5.8	8.6	16.1	2.9	7.0	12.2	1.9

Figure 8: Average node error, average number of waveforms per node andruntime versus p_{th} for C1355



During the computation phase, waveforms with extremely small occurrence probabilities can be ignored in order to achieve higher speed. All the generated waveforms are compared to an experimentally determined threshold p_{th}; if the occurrence probability is smaller than p_{th} this waveform will be ignored. The value of p_{th} affects the runtime and accuracy of computations. Figure 8 shows the variation of the runtime, the average number of waveforms, and the average error for the C1355 circuit. We have chosen $p_{th} = 10^{-7}$ since this value provides a very small degradation from the case with $p_{th} = 0$ while still achieving a relatively short runtime.

Table 2 summarizes the estimation errors from computing correlation coefficients as discussed in Sec. 4. E_{av2} and E_{tot2} in Table 2 are average and total relative estimation errors respectively where the correlation coefficients are acquired from zero-delay logic simulations. In this case the correlation coefficients are error-free. However errors still exist in the estimations because microcorrelations are approximated by macrocorrelations. E_{av1} and E_{tot1} in Table 2 are average and total relative estimation errors when correlation coeffi-

Table 2: Runtime and computation complexity for ISCAS'85 benchmark circuits with fanout delay assignment. All errors are in percentage.

	Runtime (sec)	Ave. # of waveforms	Ave. logic depth	Maximum logic depth	Number of nodes	E_{av1}	E_{tot1}	E_{av2}	E_{tot2}
C17	0.003	8.4	1.1	3	11	0.7	0.7	0.7	0.7
C432	7	130.6	11.2	29	252	5.2	2.8	5.1	1.0
C499	0.3	23.1	5.9	13	287	1.0	0.3	0.9	0.1
C880	7	110.7	8.7	30	495	4.4	2.7	4.5	2.0
C1355	8	110.4	12.7	26	631	6.3	0.2	6.3	0.4
C1908	77	168.6	16.1	44	1090	7.8	2.3	6.4	1.2
C2670	205	125.8	8.4	39	1633	9.7	2.9	9.3	2.5
C3540	881	275.2	13.4	56	2033	7.3	1.5	5.8	0.7
C5315	570	141.1	9.5	52	3151	7.1	2.1	6.8	2.0
C6288	3984	1268.3	42.1	124	2448	15.7	4.9	14.9	3.5
C7552	1201	154.4	11.1	45	4249	11.5	0.9	10.0	0.8

cients are computed using the method in [18]. The computation complexity of the SWS method is highly dependent on the circuit structure. Table 2 summarizes the runtime and average number of waveforms per node for the ISCAS'85 benchmark circuits. An Intel Xeon processor 3GHz is used for computations. Our experiments show that the product of the total number of nodes and the average logic depth can in general give a relatively good estimation of the runtime of our power estimation algorithm.

The memory requirement can grow for large circuits due to the large mask tags and large number of waveforms. However the memory size can be suppressed significantly by dynamically removing the SWSs for the nodes that are not required for future computations. This reduces the maximum required memory for the C6288 circuit from about 2.1GB to about 200MB.

7. CONCLUSION

A novel probabilistic power estimation method has been presented. The previous methods of probabilistic power estimation have weak modeling of glitch filtering. The set of simple waveforms technique is capable of accounting for the successor nodes' behavior with respect to introduced glitches. The estimations for a number of benchmark circuits are compared to the estimations obtained from logic simulations. Error-free estimations are obtained for tree structured circuits. Significant improvements in the power estimation of general circuits are achieved compared to earlier techniques.

8. REFERENCES

[1] M.A. Cirit. Estimating dynamic power consumption of CMOS circuits. In *Proc. ICCAD*, pages 534–537, November 1987.

[2] F. Najm, R. Burch, P. Yang, and I. Hajj. Crest - a current estimator for cmos circuits. In *Proc. IEEE Intl. Conf. Computer-Aided Design*, pages 204–207, November 1988.

[3] M.G. Xakellis and F.N. Najm. Statistical estimation of the switching activity in digital circuits. In *Proc. of DAC*, pages 728–733, 1994.

[4] C.S. Ding, C.Y. Tsui, and M. Pedram. Gate-level power estimation using tagged probabilistic simulation. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 17:1099–1107, November 1998.

[5] F. Hu and V.D. Agrawal. Dual-transition glitch filtering in probabilistic waveform power estimation. In *Proc. 15th ACM Great Lakes symposium on VLSI*, pages 357–360, April 2005.

[6] F. Hu and V.D. Agrawal. Enhanced dual-transition probabilistic power estimation with selective supergate analysis. In *Proc. Intl. Conference on Computer Design*, pages 366–369, October 2005.

[7] S. Bhanja and N. Ranganathan. Swtiching activity estimation of VLSI circuits using bayesian networks. *IEEE Trans. Very Large Scale Integr. Syst.*, 11(4):558–567, 2003.

[8] W.C. Tsai, C.B. Shung, and D.C. Wang. Accurate logic-level power simulation using glitch filtering andestimation. In *Circuits and Systems, IEEE Asia Pacific Conf. on*, pages 314–317, Nov 1996.

[9] P. Israsena and S. Summerfield. Novel pattern-based power estimation tool with accurate glitch modeling. In *Proc. ISCAS*, pages 721–724, May 2000.

[10] S. Theoharis, G. Theodoridis, D. Soudris, C. Goutis, and A. Thanailakis. A fast and accurate delay dependent method for switching estimation of large combinational circuits. *Journal of systems architecture*, 48(4-5):113–124, 2002.

[11] F.N. Najm. Power estimation techniques for integrated circuits. In *Proc. Intl. Conf. on Computer-aided design*, pages 492–499, 1995.

[12] F. Najm. Transition density: A new measure of activity in digital circuits. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 12:310–323, February 1992.

[13] F.N. Najm, R. Burch, P. Yang, and I. Hajj. Probabilistic simulation for reliability analysis of CMOS VLSI circuits. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 10:1372–1381, November 1990.

[14] S.C. Seth, L. Pan, and V.D. Agrawal. Predict - probabilistic estimation of digital circuit testability. In *Proc. Fault Tolerant Computing Symposium*, pages 220–225, June 1985.

[15] T. Stornetta and F. Brewer. Implementation of an efficient parallel bdd package. In *Proc. Design Automation Conference*, pages 641–644, June 1996.

[16] B. Kapoor. Improving the accuracy of circuit activity measurement. In *Proc. Design Automation Conference*, pages 734–739, June 1994.

[17] R. Marculescu, D. Marculescu, and M. Pedram. Switching activity analysis considering spatiotemporal correlations. In *Proc. IEEE/ACM Intl. Conference on Computer Aided Design*, pages 294–299, November 1994.

[18] S. Ercolani, M. Favalli, M. Damiani, P. Olivo, , and B. Ricc. Estimate of signal probability in combinational logic networks. In *Proc. 1st European Test Conf.*, pages 132–138, 1989.