Static Transition Probability Analysis Under Uncertainty

Siddharth Garg, Siddharth Tata and Ravishankar Arunachalam
Department of Electrical Engineering
Indian Institute of Technology-Madras
s_garg1982@yahoo.com, tsiddharth@yahoo.com, ravia@ee.iitm.ernet.in

Abstract

Deterministic gate delay models have been widely used to find the transition probabilities at the nodes of a circuit for calculating the power dissipation. However, with progressive scaling down of feature sizes, the variations in process parameters increase, thereby increasing the uncertainty in gate delay. In this work, we propose a novel non-simulative scheme to compute the transition probability waveforms (TPWs) in a single pass of the circuit for continuous gate delay distributions. These TPWs are continuous functions of time as opposed to the deterministic delay case where transitions are constrained to occur at discrete time points. The TPWs are then used to calculate the dynamic power dissipation in a circuit. We show that the corresponding power estimates obtained from deterministic delay models can be off by as much as 75%. Our method has an average error of only 6% and a speed up of 232$\times$ when compared to logic simulations. Another important application of our TPWs is in the area of crosstalk noise where the likelihood of signals switching within a certain timing window is required.

1. Introduction

Logic transitions in a circuit affect circuit performance characteristics like timing and power dissipation. It is therefore necessary to know when and how often these transitions occur and quantify them using probabilistic techniques. Early methods to calculate the probability of logic transitions (transition probability) assumed a zero-delay model [9, 7]. These models do not account for transitions due to glitching activity in the circuit which is caused due to unbalanced path delays. To account for these glitches, deterministic delay models that assigned a fixed delay to every gate in the circuit were introduced in [2, 11]. In [10, 6], the authors use this model to obtain a tagged transition probability waveform at each node. These waveforms represent the probability of a transition occurring at any time instant. In these waveforms, transitions can only occur at discrete time points as the delay for every gate is a fixed number.

Progressive scaling down of feature sizes has meant an increase in manufacturing process variations. The variability introduced in device characteristics has correspondingly increased, thus affecting the overall circuit performance. Variations in process parameters like effective gate length, oxide thickness, and threshold voltage affect the gate delay, making it vary over a range of values. Since the actual gate delay can take *any* value within this specified range, gate delays need to be modeled as random variables with a known probability density function (pdf). In this work, we propose a novel methodology to obtain precise transition probability waveforms (TPWs) at all nodes in a combinational circuit. Standard approaches to model continuous time functions in a computer would suffer from an exponential increase in time-points with increasing complexity of the circuit. We introduce a new approach to sample what we call the integrated probability waveform (IPW), to keep the storage and error within bounds. Having obtained these TPWs we apply them to two crucial tasks in the circuit design process- power estimation and the analysis of the effect of crosstalk induced noise on circuit delay.

The dynamic power dissipated by a logic gate is given by the relation

$$P_x = 0.5\, C_L\, V_{dd}^2\, f_{clk}\, E_{sw} \tag{1}$$

where C_L is the effective parasitic capacitance at the gate output, V_{dd} is the supply voltage,f_{clk} is the clock frequency and E_{sw} is the average switching activity of the gate output per clock cycle. Since V_{dd},f_{clk} and C_L are known, the task of dynamic power estimation usually reduces to the problem of finding the average switching activity at each node. The average switching activity at each node can be found by integrating the TPWs obtained from our proposed scheme. We show that power estimation tools that use deterministic delay models can give incorrect estimates that are off by as much as 75%. To alleviate this problem [4] proposes a Monte Carlo simulation based scheme to estimate the power where the gate delay is modeled as a random variable with a

Gaussian pdf. However, the scheme suffers from excessive run time and memory requirements as it is simulative in nature. Our approach of estimating the power using the TPWs is more efficient than simulative methods and has an average error of only 6%.

Logic transitions on a node (aggressor) can affect neighbouring nodes (victim) due to capacitive coupling. Traditionally, timing windows obtained from static timing analysis have been used to determine the effect of noise due to capacitive coupling [1, 3, 8]. If the timing windows overlapped, it was assumed that the aggressor would switch simultaneously with the victim. However, it is possible that even for an overlapping timing window, the likelihood of simultaneous switching is low. We provide a framework for using the TPWs obtained from our model to estimate the likelihood of delay noise.

The rest of the paper is organized as follows. Section 2 formally defines the TPWs and explains the terminology used in the paper, section 3 develops the equations used to propagate the TPWs, section 4 describes the method of representing the TPWs, section 5 deals with reconvergent fanout, section 6 discusses the application of TPWs to the problems of power estimation and delay noise due to capacitive coupling. In section 7 we present our experimental results on the ISCAS'85 set of benchmark circuits followed by conclusions in section 8.

2. Background and Terminology

In this work, we propose a novel scheme where we represent the transition probabilities as *continuous* functions of time as against discrete valued functions that earlier methods use. We begin by defining the probability waveforms we use in our scheme.

Definition 2.1 (Rising Transition Probability Waveform) *$p^x_{01}(t)$ is a waveform such that $p^x_{01}(t)\,\Delta t$ is the probability that the signal 'x' will make a transition from 0→ 1 in a small time interval $(t, t+\Delta t)$*

Definition 2.2 (Falling Transition Probability Waveform) *$p^x_{10}(t)$ is a waveform such that $p^x_{10}(t)\,\Delta t$ is the probability that the signal 'x' will make a transition from 1→ 0 in a small time interval $(t, t+\Delta t)$*

Definition 2.3 (Signal Probability Waveform) *$sp^x(t)$ is the probability that the signal 'x' has a value of logic 1 at the time instant t.*

The first two waveforms are collectively referred to as the transition probability waveforms (TPWs), and if the signal probability waveform is included, the three waveforms together are called the probability waveforms at the node 'x'.

The effect of process parameter variations is captured in the gate delay model. In this work, we assume that the gate delay is a random variable with a known pdf. While our scheme works for any general delay distribution, we implement it for the case of a truncated Gaussian distribution which is used in [4, 5]. We use the notation $D^x(t)$ to denote the pdf of the delay distribution for a gate with output node 'x'.

3. Waveform Propagation Scheme

Given that the probability waveforms are specified at the primary inputs of a circuit, we derive equations to propagate these waveforms to every node in the circuit. Without loss of generality, we consider a two input AND gate with input nodes 'a' and 'b' and output node 'c'. We assume that the two input lines 'a' and 'b' are independent for the current discussion.

To obtain the probability waveforms at the node 'c', we split up the AND gate into three different stages. The first stage is an AND gate with zero delay, the second stage is the glitch filter and the third stage is a BUFFER with the delay distribution, $D^c(t)$, of the original AND gate. The intermediate nodes are labeled 'q' and 'g' as shown in Figure 1. In the following three sub-sections we develop equations to propagate the TPWs across each of the three stages. The final sub-section derives equations for the signal probability waveform at the output node.

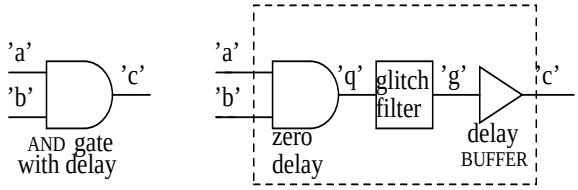

Figure 1. Three stages of an AND gate

3.1. Zero-Delay Gate

An up transition on node 'q' can occur in a small time interval $(t, t+\Delta t)$ in three mutually exclusive ways that are listed in Table 1.

Summing up the probabilities of the three mutually exclusive events, the expression for the TPW at node 'q' can be written as,

$$p^q_{01}(t)\Delta t = p^a_{01}(t)sp^b(t)\Delta t + p^b_{01}(t)sp^a(t)\Delta t + p^a_{01}(t)p^b_{01}(t)\Delta t^2 \quad (2)$$

In the limit $\Delta t \to 0$ and the input TPWs being continuous functions of time, (2) reduces to

$$p^q_{01}(t) = p^a_{01}(t)sp^b(t) + p^b_{01}(t)sp^a(t) \quad (3)$$

Event	Probability
Node 'a' 0→1 and Node 'b' at logic 1	$p^a_{01}(t)sp^b(t)\Delta t$
Node 'b' 0→1 and Node 'a' at logic 1	$p^b_{01}(t)sp^a(t)\Delta t$
Node 'a' 0→1 and Node 'b' 0→1	$p^a_{01}(t)p^b_{01}(t)\Delta t^2$

Table 1. Mutually Exclusive ways for an up transition

The corresponding equation for the falling TPW can be written as,

$$p^q_{10}(t) = p^a_{10}(t)sp^b(t) + p^b_{10}(t)sp^a(t) \tag{4}$$

3.2. Glitch Filter

Short glitches at the input of a gate do not propagate as valid logic transitions at the output due to the inertial delay of the gate. Glitch filtering refers to the process of adjusting the TPWs to account for this. Determining the minimum glitch width at the input that can propagate to the output of the gate is not an easy task. In [4], the minimum glitch width is assumed to be equal to half the transport delay of the gate. In our case, since the delay of the gate is variable, the minimum glitch width is also variable. This makes the glitch filtering scheme equations intractable. Instead we assume that the minimum glitch width is a constant which is equal to half the mean of the gate delay probability density function $D^c(t)$, denoted by d^c.

For a rising transition between $(t, t+\Delta t)$ at node 'a', all the falling transitions at node 'b' which lie between $(t, t+d^c)$ produce glitches at the output node which need to be filtered. Similarly for a rising transition at node 'b' in the interval $(t, t+\Delta t)$, all falling transitions in the interval $(t, t+d^c)$ are subject to glitch filtering. Under the assumption that the inputs are free from glitches of width less than d^c, these two events are mutually exclusive. Their respective probabilities can be subtracted from $p^q_{01}(t)$ to get $p^g_{01}(t)$.

$$\begin{aligned} p^g_{01}(t) = p^q_{01}(t) - p^a_{01}(t)\int_t^{t+d^c} p^b_{10}(\tau)\,d\tau \\ - p^b_{01}(t)\int_t^{t+d^c} p^a_{10}(\tau)\,d\tau \end{aligned} \tag{5}$$

Similar reasoning can be used to obtain $p^g_{10}(t)$.

3.3. The Delay BUFFER

To obtain the TPWs at the node 'c', once $p^g_{01}(t)$ and $p^g_{10}(t)$ have been computed, we observe that,

$$p^c_{01}(t)\Delta t = \lim_{\Delta k\to 0}\sum_{i=-\infty}^{\infty} p^g_{01}(k_i)\Delta k D^c(t-k_i)(\Delta t - \Delta k) \tag{6}$$

where $k_i = i\Delta k$. In effect, we have discretized the waveform $p^g_{01}(t)$ into small intervals of length Δk to arrive at (6). We can show that (6) reduces to,

$$p^c_{01}(t) = \int_{-\infty}^{\infty} p^g_{01}(k)D^c(t-k)\,dk \tag{7}$$

or

$$p^c_{01}(t) = p^g_{01}(t) * D^c(t) \tag{8}$$

where * represents the convolution operation. Similarly,

$$p^c_{10}(t) = p^g_{10}(t) * D^c(t) \tag{9}$$

The above set of equations assumes that the gate delay is characterized by a single random variable with the same delay distribution for rise and fall transitions at the output. A more general approach would be to use different distributions for rising and falling output transitions as well as different delays for different inputs. If $D^{ac}_r(t)$ ($D^{ac}_f(t)$) is the distribution of the delay from a→c for a rising (falling) transition at c, and similarly $D^{bc}_r(t)$ and $D^{bc}_f(t)$ are the distributions for input b, the propagation equation is given under as

$$\begin{aligned} p^c_{01}(t) = \left[p^a_{01}(t)\left(sp^b(t) - \int_t^{t+d^{ac}} p^b_{10}(\tau)\,d\tau\right)\right] * D^{ac}_r(t) + \\ \left[p^b_{01}(t)\left(sp^a(t) - \int_t^{t+d^{bc}} p^a_{10}(\tau)\,d\tau\right)\right] * D^{bc}_r(t) \end{aligned} \tag{10}$$

3.4. Signal Probability Waveform

(3), (4), (5), (8) and (9) yield the two TPWs at output node 'c'. Now to calculate the signal probability waveform at node 'c' we observe that the *change* in signal probability in the time interval $(t, t+\Delta t)$ depends on the number of signals undergoing a transition, either from $0 \to 1$ or $1 \to 0$ in that interval, i.e.

$$sp^c(t+\Delta t) - sp^c(t) = p^c_{01}(t)\Delta t - p^c_{10}(t)\Delta t \tag{11}$$

Again, under the limit $\Delta t \to 0$, (11) reduces to,

$$sp^c(t) = sp^c(0) + \int_0^t \left[p^c_{01}(\tau) - p^c_{10}(\tau)\right]\,d\tau \tag{12}$$

where,

$$sp^c(0) = sp^a(0)sp^b(0) \tag{13}$$

4. Reconvergent Fanout

The previous section assumed that the two inputs 'a' and 'b' were independent. However if the inputs 'a' and 'b' fanout from a common node, they are correlated.The equations derived in section 3 will therefore have to be multiplied by time varying correlation coefficients. Calculating

these correlation coefficients is computationally expensive and is infeasible for most circuits. Past methods either neglect them or approximate them with zero delay correlation coefficients [6].

It has been observed that the effect of reconvergent fanout reduces as the fanout node moves away from the point of reconvergence while using a deterministic delay model. For example, [6] neglects reconvergent fanout if the common node is more than six levels away as it is uncertain whether transition at the point of reconvergence has been caused by a transition at the common node. Moreover, in the variable delay model, the effect of reconvergent fanout is of lesser importance since further uncertainty in the paths is injected by the uncertainty in the delay after the very first level. In our scheme, we therefore assume all inputs to a gate to be independent.

5. Piece-wise Linear Implementation

The continuous time TPWs need to be represented on a computer. Sampling the TPWs at discrete time points is one way to do so. However, since the width of the TPWs will increase linearly with the levels in the circuits, the storage requirements and computation costs become prohibitive. Instead we propose a novel method to represent the TPWs in which they are first integrated to give the integrated probability waveforms (IPWs) as given in (14) and (15)

$$I^x_{01}(t) = \int_{-\infty}^{t} p^x_{01}(\tau)\, d\tau \tag{14}$$

$$I^x_{10}(t) = \int_{-\infty}^{t} p^x_{10}(\tau)\, d\tau \tag{15}$$

The IPWs are monotonically increasing functions of time which are upper bounded by the average switching activity at the corresponding circuit node[1] . In our scheme, the IPWs are stored as piece-wise linear (PWL) functions which are obtained by sampling them at equally spaced *y-axis* values (where the x-axis is the time axis). This method has the advantage that fewer points are used to represent the time intervals in which the node is dormant, while time intervals with greater switching will be sampled more frequently. In order to keep the complexity of the scheme within bounds, we allocate a fixed number of points (N) to represent *every* IPW in the circuit. Alternately we could have decided to sample the waveform at equally spaced intervals along the *y-axis* but since this does not offer any significant benefit we choose the number of points (N) to be constant. Figure 2 shows a PWL representation of an IPW according to our method with N=7. Since the IPWs, and not

1 The maximum value of the IPW at any node gives it's average switching activity. Therefore, the maximum value of the IPW need not be one, unlike a conventional cumulative distribution function (CDF).

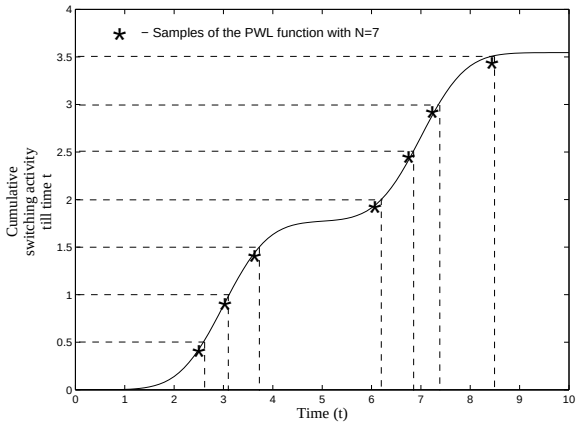

Figure 2. PWL representation of an integrated probability waveform

the TPWs, are stored at every node in the circuit, the propagation equations derived in section 3 need to be restated in terms of the IPWs at nodes 'a' and 'b'.

5.1. Propagation Equations

Rewriting the equations derived in section 3 [(3), (5), (8) and (12)] in terms of the IPWs we get,

$$p^q_{01}(t) = \frac{dI^a_{01}(t)}{dt} sp^b(t) + \frac{dI^b_{01}(t)}{dt} sp^a(t) \tag{16}$$

$$\begin{aligned} p^g_{01}(t) = \; & p^q_{01}(t) - p^a_{01}(t)(I^b_{10}(t+d) - I^b_{10}(t)) \\ & - p^b_{01}(t)(I^b_{10}(t+d) - I^b_{10}(t)) \end{aligned} \tag{17}$$

$$I^c_{01}(t) = I^g_{01}(t) * D^c(t) \tag{18}$$

$$sp^c(t) = sp^c(0) + I^c_{01}(t) - I^c_{10}(t) \tag{19}$$

Only the equations for the rising transitions are presented, similar equations for the falling transitions can be easily derived. Only (16),(17), (18) and (19) will be used explicitly in our implementation. Note that we have presented the equations for the case where the gate delay is characterised by a single random variable. The equations for a more general case with different rise/fall delays and transport delays can be easily derived from (10).

Since the IPWs at nodes 'a' and 'b' are PWL, the signal probability waveforms, $sp_a(t)$ and $sp_b(t)$ will also be PWL. Furthermore, the derivatives of the IPWs will be piecewise constant (PWC). This information, along with (16) implies that the computed TPWs, $p^g_{01}(t)$ and $p^g_{10}(t)$, are PWL. These waveforms are now integrated and are sampled at the

preset y-axis values to give the IPWs, $I^g_{01}(t)$ and $I^g_{10}(t)$. The time taken to perform the various operations on the IPWs - addition, subtraction and multiplication, varies linearly with the number of segments in the IPWs.

5.2. The Convolution Operation

Having obtained the IPWs at node 'g' we need to propagate them across the delay BUFFER as per (18). To perform the convolution operation, (18) is re-written as,

$$I^c_{01}(t) = \int_{-\infty}^{t} \frac{dI^g_{01}(t)}{dt} * D^c(t)dt \qquad (20)$$

where $\frac{dI^g_{01}(t)}{dt}$ is a PWC function. The delay distribution, $D^c(t)$, is also represented as a PWC function as shown in Figure 3.

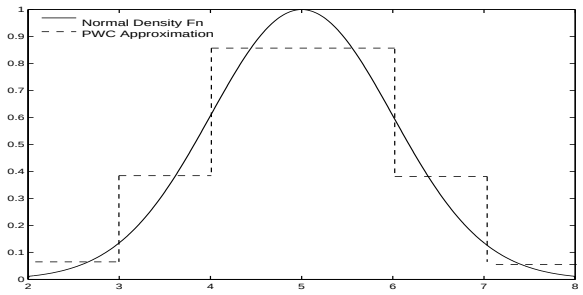

Figure 3. Truncated Gaussian pdf and it's PWC representation

A PWC function is expressible as a sum of a number of rectangular functions. If $\frac{dI^g_{01}(t)}{dt}$ has n_1 rectangular functions and $D^c(t)$ has n_2 rectangular functions, n trapezoidal functions need to be added to yield the result of the convolution operation, where $n = n_1 n_2$. Figure 4 shows the convolution of two rectangular functions, $R_1(t)$ and $R_2(t)$ to yield a trapezoidal function T(t).

The n trapezoidal functions are grouped into sets of two each and added. The resulting functions are then further grouped and added, and this process is recursively repeated to yield the final result. The resulting PWL waveform is integrated and sampled at the preset y-axis values to give $I^c_{01}(t)$. A similar process is used to obtain $I^c_{10}(t)$. The time complexity of the convolution operation is O(nlogn).

6. Applications of the TPWs

The TPW at any circuit node can be explicitly evaluated as a PWC function by differentiating the corresponding IPW. The TPW is used for both dynamic power estimation and analysis of noise due to capacitive coupling.

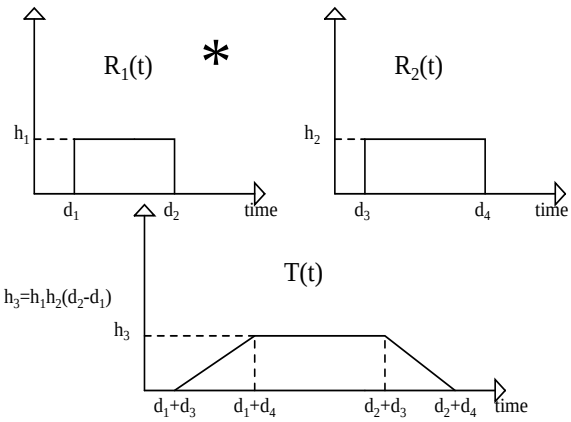

Figure 4. Convolution of two rectangular functions

6.1. Dynamic Power Estimation

The dynamic power dissipation at a circuit node is given by

$$P_x = 0.5\, C_L\, V^2_{dd} f_{clk} \int_0^{\infty} (p^x_{01}(t) + p^x_{10}(t))\, dt \qquad (21)$$

P_x is summed up over all nodes to give the total dynamic power dissipation in the circuit.

6.2. Capacitive Coupling Noise

To analyze noise due to crosstalk consider the aggressor and victim timing windows obtained from static timing analysis as shown in Figure 5. Note that the rising window has been shown for the aggressor and the falling window for the victim, since the aggressor and the victim should switch in different directions for worst case delay noise. Without information of the TPWs at the aggressor and victim nodes ('a' and 'v' respectively), an arbitrary distribution would have to be assumed (for e.g. a uniform distribution) for the time varying transition probability in the timing window interval. This would then be used to find the probability of both the aggressor and victim switching in the overlap interval.

For the same nodes 'a' and 'v', if the TPWs are known, as shown in Figure 5, the probability that both the aggressor and victim switch in the overlap interval (T_{a1}, T_{v2}), denoted by $P^n_{a,v}$ can be computed as

$$p^n_{a,v} = \int_{T_{a1}}^{T_{v2}} \int_{T_{a1}}^{T_{v2}} p^a_{01}(t_1) p^v_{10}(t_2)\, dt_1\, dt_2 \qquad (22)$$

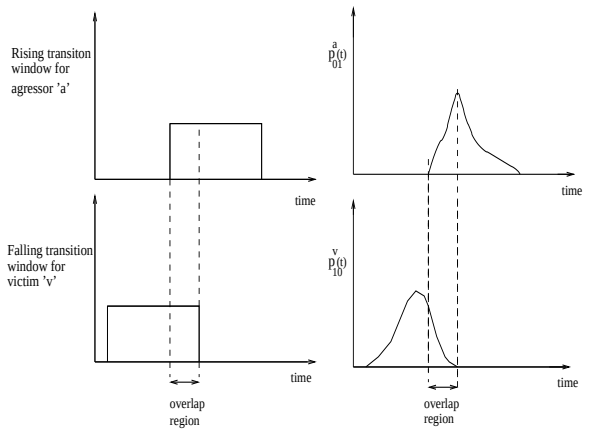

Figure 5. Aggressor-Victim crosstalk analysis

Since a victim can have more than one aggressor, the probability that a particular aggressor and the victim switch in the corresponding overlap interval can be calculated for each of the aggressors. The noise contribution of the aggressors can then be weighted with their respective simultaneous switching probabilities and superposed to give a more realistic picture of noise due to crosstalk.

7. Experimental Results

To validate our proposed scheme, we tested our algorithm on the ISCAS'85 set of benchmark circuits. The benchmark circuits are mapped to a recent technology library and the gate delay information is extracted using a commercial logic synthesis tool. We assume that the variance of the delay distribution $\sigma_d = 0.3\mu_d$ where μ_d is the mean value of the distribution, as assumed in [4]. Primary inputs are assumed to be unbiased and are assumed to switch at the same time instant. In order to compare the accuracy of our scheme, we also performed explicit logic simulations. The logic simulation results are reported for 100,000 simulation runs. We assume a V_{dd} value of 3.3 V and a clock frequency of 500 MHz. The experiments were run on a 2 GHz Intel Pentium processor running on 256 MB of RAM.

7.1. Transition Probability Waveforms

Since there is no accurate way of obtaining TPWs through logic simulation to a high degree of accuracy in a reasonable amount of time, we compare switching activity figures between the logic simulation and our scheme. Table 2 reports the average node by node error in switching activity obtained in our scheme compared to switching activity obtained from extensive logic simulations.

Circuit	Average Percentage Error
c432	4.7
c499	2.1
c880	7.9
c1355	6.6
c1908	10.6
c2670	16.2
c3540	15.9
c6288	13.8
Mean	9.73

Table 2. Average node by node percentage error

The mean average node-by-node error over all circuits is only 9.73%, thereby re-enforcing the validity of our scheme as an accurate method to compute the TPWs. Figure 6 shows the rising TPW ($p_{01}(t)$) obtained by our method for a randomly picked node (node 791) in the c880 benchmark circuit. The number of pieces (N) in the PWL representation is set at N=50 for every circuit node. From the waveform, one can clearly infer three regions of increased switching activity on the node. Timing windows obtained from static timing analysis cannot yield this information.

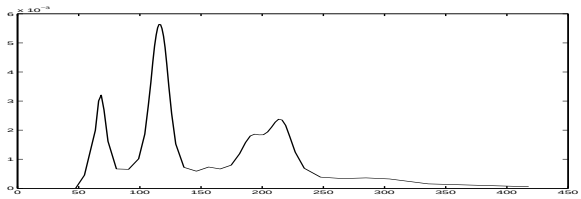

Figure 6. $p_{01}(t)$ at node 791 in c880 circuit

7.2. Power Estimation

Table 3 shows the percentage error in power dissipation by using a deterministic delay model, in which the mean values of the delay distribution are used as the gate delay numbers, with reference to the power dissipation obtained from the variable delay model. These figures are generated by running extensive logic simulations for the deterministic delay case and the variable delay case.

From Table 3, it is observed that a deterministic delay model can yield an overestimate in power dissipation of upto 75% (c6288) and an underestimate of upto -25%

Circuit	Variable Delay Power(in mW)	Fixed Delay Power(in mW)	Error Percentage
c432	6.55	6.80	+3.88
c499	13.76	10.28	-25.28
c880	20.98	20.93	-0.26
c1355	37.27	36.63	-1.71
c1908	80.52	93.69	+16.35
c2670	114.84	115.25	+0.36
c3540	149.72	150.14	+0.28
c6288	1565.17	2746.87	+75.5

Table 3. Error between deterministic and variable delay models

(c499). Methods based on deterministic delay models are therefore inadequate to estimate the power dissipation under delay uncertainty.

	Logic Simulation		TPW based Estimates		
	Power (mW)	Time (sec)	Power (mW)	Error (%)	Speed Up
c432	6.55	0.6	7.07	7.90	277×
c499	13.76	12.7	13.92	1.14	252×
c880	20.98	15.1	20.38	2.88	185 ×
c1355	37.27	30.4	37.06	0.56	177×
c1908	80.52	45.0	70.01	13.05	200×
c2670	114.84	81.7	101.64	11.50	247 ×
c3540	149.72	218.6	138.62	7.41	243×
c6288	1565.07	409.1	1582.15	1.09	274×
Mean				**5.69**	**232** ×

Table 4. Comparison of power Estimates from logic simulation and the proposed algorithm

Table 4 reports the power estimated obtained from our scheme in comparison with results obtained from extensive logic simulations for the variable delay case. For our scheme, the number of segments (N) in the PWL waveform at every node is kept at a value of N=50. It can be observed that the average error is only 5.69% and the worst case error is 13%. The speed-up obtained from our scheme over the extensive logic simulations is 232 × on an average. Moreover, the speed-up does not worsen with increase in circuit size.

8. Conclusions

In this paper we present a novel scheme to represent the transition probability waveforms (TPWs) which captures uncertainty in gate delays. To the best of our knowledge this is the first time that transition probabilities have been represented as continuous functions of time. We propose a fast and efficient implementation to propagate these TPWs across all nodes of a circuit. We use TPWs to estimate dynamic power dissipation under delay uncertainty and show that previously used deterministic delay models can give an error of upto 75%, while our maximum error is only 13%. Moreover, we obtain substantial speed-up over simulative schemes. We also provide a framework to use our TPWs to analyze the effect of noise due to crosstalk.

References

[1] R. Arunachalam, K. Rajagopal, and L. T. Pillegi. TACO: Timing Analysis with COupling. In *Proceedings of the Design Automation Conference*, 2000.

[2] R. Burch, F. N. Najm, P. Yang, and T. N. Trick. A Monte Carlo approach for power estimation. *IEEE Transactions on Very Large Scale Integration Systems*, pages 63–71, March 1993.

[3] P. Chen and K. Keutzer. Towards true crosstalk noise analysis. In *Proceedings of the International Conference on Computer Aided Design*, pages 132–137, 1999.

[4] T.-L. Chou and K. Roy. Statistical estimation of combinational and sequential CMOS digital circuit activity considering uncertainty of gate delays. In *Proceedings of the Asia and South Pacific Design Automation Conference*, pages 95–100, 1997.

[5] A. Devgan and C. Kashyap. Block-based static timing analysis under uncetainty. In *Proceedings of the International Conference of Computer Aided Design*, pages 607–614, November 2003.

[6] C.-S. Ding, C.-Y. Tsui, and M. Pedram. Gate-level power estimation using tagged probabilistic simulation. *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, 17(11):1099–1107, November 1998.

[7] H. Goldstein. Controllability/observability of digital circuits. *IEEE Transactions on Circuits and Systems*, 26(9):685–693, September 1979.

[8] Y. Sasaki and G. DeMichelli. Crosstalk delay analysis using relative window method. In *Proceedings of the ASIC/SoC Conference*, pages 9–13, 1999.

[9] J. Savir, G. Ditlow, and P. Bardell. Random pattern testability. *IEEE Transactions on Computing*, 33(1), January 1984.

[10] C.-Y. Tsui, M. Pedram, and A. Despain. Efficient estimation of dynamic power consumption under a real delay model. In *Proceedings of the International Conference on Computer Aided Design*, 1993.

[11] M. G. Xakellis and F. N. Najm. Statistical estimation of the switching activity on digital circuits. In *Proceedings of the 31st Design Automation Conference*, 1994.