A New Gate Delay Model for Simultaneous Switching and Its Applications*

Liang-Chi Chen, Sandeep K. Gupta, Melvin A. Breuer
Department of EE - Systems, University of Southern California, Los Angeles, CA 90089-2562
+1 (213) 740-4460, +1 (213) 740-2251, +1 (213) 740-4469
{lichen, sandeep, mb}@poisson.usc.edu

ABSTRACT

We present a new model to capture the delay phenomena associated with simultaneous to-controlling transitions. The proposed delay model accurately captures the effect of the targeted delay phenomena over a wide range of transition times and skews. It also captures the effects of more variables than table lookup methods can handle. The model helps improve the accuracy of static timing analysis, incremental timing refinement, and timing-based ATPG.

1. INTRODUCTION

Static timing analysis (STA) [1] is widely used for validating circuit performance. It provides min-max timing ranges (also called timing windows) for rising and falling transitions on each line in a circuit without explicitly considering any vectors. The accuracy of STA depends heavily on the delay model used for each gate. Although SPICE-like models [2][3][4] provide good timing accuracy, they can not be used in STA because they require fully specified input waveforms.

Pin-to-pin delay models [5] are hence used for STA. One main deficiency of pin-to-pin delay models is that simultaneous switching delay [6][7] is not captured. *Simultaneous to-controlling transitions* at inputs of a primitive gate decrease gate delay due to activation of multiple charge/discharge paths (Figure 1). *Simultaneous to-non-controlling transitions* at inputs of a primitive gate increase gate delay due to the Miller effect. The former is a first-order effect and the later is a second order effect.

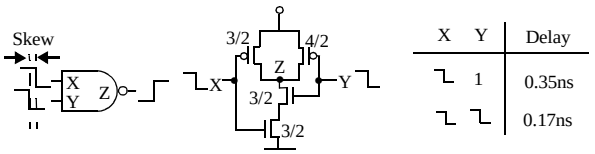

FIGURE 1. Single vs. multiple to-controlling-value transitions at gate inputs.

We have developed a delay model to capture the delay due to simultaneous to-controlling transitions at the inputs of a primitive gate that is more accurate than pin-to-pin model. At this time we continue to use pin-to-pin delay model for simultaneous to-non-controlling transitions.

For a delay model to be adopted in STA, the delay model should have certain characteristics that enable identification of the combinations of transition and arrival times at a gate's inputs that lead to each individual extreme value for a timing range at its output. The proposed delay model has these characteristics.

We show how this delay model can be used by STA to improve its accuracy and also demonstrate that our delay model is useful in *incremental timing refinement (ITR)* [8][9][10], which can provide tighter min-max ranges of delay values than STA when a partially specified vector is given. As will be discussed below, ITR provides a new approach to prune the search space for timing-based ATPG.

The proposed delay model is compared with existing models and is shown to accurately capture the effect of the target delay phenomena over a wide range of transition times and skews (differences between arrival times) associated with transitions at inputs of a single gate. The significance of the proposed model is shown via experiments on ISCAS85 benchmarks.

In STA, the input vectors are completely unspecified. In *timing simulation (TS)*, the input vectors are completely specified. During test generation for a target, values are specified incrementally and this framework enables refinement of timing windows. ITR was proposed to compute more accurate timing ranges during test generation, where line values are specified incrementally. As all delay models can be used by TS, we demonstrate that the proposed model is suitable for STA, ITR and timing-based test generation.

In Section 2, previous delay models are reviewed. In Section 3, the approach employed to develop our delay model is introduced, and the assumptions validated. In Section 4, operations for static timing analysis on our delay model are developed. In Section 5, these operations are extended to perform incremental timing refinement. Results of experiments are shown in Section 6. Concepts for constructing timing-based ATPG utilizing ITR framework are proposed in Section 7.

2. PREVIOUS DELAY MODELS

Simulators have been developed for digital circuits with different accuracy/computation cost trade-offs. *Timing simulators* [2][3] generate voltage waveforms more efficiently (lower computation costs) than SPICE-like *circuit simulators* [4], but are less accurate. *Delay calculators* are very efficient in determining circuit delay. Several approaches for delay calculation have been developed, including *resistance-capacitance (RC) based systems* [11], *equation solving systems* [12], *analytical delay function systems* [13], and *empirical delay based systems* that use *lookup tables* [14][15][16][17] or *empirical delay functions* [6][18]. Some of these methods do not provide sufficient accuracy. For others it is difficult to identify the combinations of transition and arrival times at gate inputs that lead to extreme values of timing ranges at gate outputs, unless all possible pairs of vectors are simulated. So these methods can not be used in STA or ITR for large circuits.

To accurately model the effects of simultaneous input transitions [6][17][18][19], both input transition time and input skew must be considered. Often a multi-input gate is modeled as an "equivalent" inverter, and the multiple input transitions are mapped into a single transition at the inverter's input. In some approaches researchers have obtained an equivalent inverter for a gate by replacing (collapsing) parallel transistors by a single transistor whose width is the sum of the widths of the transistors in parallel. In [6] and [18], the authors provided better models for finding an equivalent inverter, but their models can result in significant errors because certain combinations of *input transition time* and *input skew* are ignored.

DAC 2001, June 18-22, 2001, Las Vegas, Nevada, USA.

*. This work was supported in part by the Semiconductor Research Corp. under contract No. 98-TJ-646, and by Intel Corporation.

3. PROPOSED DELAY MODEL

In this paper we propose a new delay model to handle simultaneous to-controlling transitions. Using the same input variables as [17], our model gives more accurate empirical formulas than those presented in [6][18], which have significant errors for many cases. The model has been validated using arbitrary skews over a typical range of input transition times.

A NAND gate, with output Z and two inputs X and Y (Figure 1), is used as the example for illustrating the definitions. Here Z represents the gate output and also the gate. The **controlling value** of a multi-input gate Z, $\boldsymbol{CV}^{\mathbf{Z}}$, is the value when applied to any of the gate's inputs, completely determines the value at its output. In the two-value logic system, the **non-controlling value** of a gate Z, $\overline{\boldsymbol{CV}}^{\mathbf{Z}}$, is the complement of its controlling value. The **to-controlling transition** at an input of Z is denoted as a sequence of values $<\overline{CV}^{Z}, CV^{Z}>$. If to-controlling transitions occurs at one or more inputs of a gate, and the gate's non-controlling value is applied to its remaining inputs, then the transition at the gate output is called a **to-controlling response**. **To-non-controlling transition** and **response** are defined similarly. The **transition time ($\mathbf{T^X}_{tr}$)** of a transition *tr*, where $tr \in \{R, F\}$, on line X is the time required for a rising transition (**R**) to go from 0.1Vdd to 0.9Vdd, and from 0.9Vdd to 0.1Vdd for a falling transition (**F**). The **arrival time ($\mathbf{A^X}_{tr}$)** of a transition *tr* on line X is the time when the voltage at the output reaches 0.5 V_{dd}. The **skew** ($\delta^{\mathbf{X,Y}}$) between transitions on lines X and Y is $A^Y_{tr} - A^X_{tr}$. The **to-controlling gate delay function $\mathbf{d^Z}_{tr}$** , defined as $A^Z_{tr} - \min(A^X_{\overline{tr}}, A^Y_{\overline{tr}})$, is the gate delay of *Z*, where the output transition $tr \in \{R, F\}$ is a to-controlling response, $R = \overline{F}$ and $F = \overline{R}$. The **pin-to-pin delay** from X to Z is the gate delay of Z when Y is steady at the non-controlling value and a transition is applied on X. $\mathbf{d^{Z,X}}_{\boldsymbol{tr}}$ is the pin-to-pin delay function from X to *Z*, where the output transition is *tr*. The **to-non-controlling gate delay** is defined as $A^Z_{tr} - \max(A^X_{\overline{tr}}, A^Y_{\overline{tr}})$, where A^Z_{tr} , the latest output arrival time computed through pin-to-pin delay, is $\max(A^X_{\overline{tr}} + d^{Z,X}_{tr}, A^Y_{\overline{tr}} + d^{Z,Y}_{tr})$. **To-controlling transition time function $\mathbf{t^Z}_{\boldsymbol{tr}}$** and **to-non-controlling transition time function $\mathbf{t^{Z,X}}_{\boldsymbol{tr}}$** are defined similarly.

3.1 Delay Phenomena

3.1.1 Simultaneous Switching

SDF [5], which is commonly used for STA, uses pin-to-pin delays and hence is not accurate for modeling simultaneous transitions with small skew values. For a two-input NAND gate, the delay when a single input has a falling transition is larger than that when both inputs have simultaneous falling transitions, since in the latter case the output is charged via multiple PMOS transistors (Figure 1) [6]. We have developed a delay model to capture this phenomenon. Given the input skews and transition times of a gate, our model computes the gate delay and output transition time by formulating timing functions using empirical results.

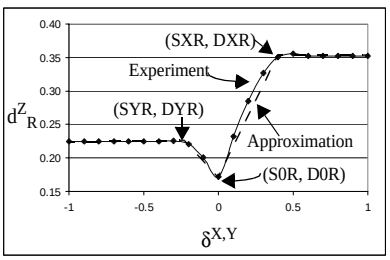

FIGURE 2. Rising delay of two-input NAND gate as a function of $\delta^{\mathbf{X,Y}}$ and its linear approximation.

To explain the speed-up caused by simultaneous falling transitions in Figure 1, we plot the to-controlling gate delay as a function of $\delta^{X,Y}$ for some fixed T^X_F and T^Y_F , where $\delta^{X,Y} = A^Y_F - A^X_F$ (Figure 2). The speed-up caused by the simultaneous switches is significant only when $|\delta^{X,Y}|$ is small. When $|\delta^{X,Y}|$ is large, the delay is the same as the pin-to-pin delay. A linear approximation is also shown in Figure 2 along with the coordinators of the three points that define this approximation. Two transitions in the same direction on X and Y are called **δ-simultaneous** if $SYR \le \delta^{X,Y} \le SXR$. The output transition occurs earlier if input transitions are δ-simultaneous.

3.1.2 Input Positions

Let **n** be the number of inputs to a NAND gate and $\mathbf{p^X}$ the position of input X in the serial chain (Figure 3). The position of the input closest to the output is defined as 0. According to SPICE simulations, the pin-to-pin rise delay of a 5-input NAND gate from input 4 to the output may be 50% larger than that from input 0. This occurs because in the former case, the pull-up transistor also needs to charge the source/drain capacitances of many transistors in the pulldown network.

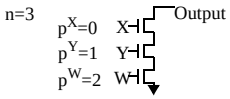

FIGURE 3. Number of inputs and input positions.

3.2 Timing Functions (for a Two-input NAND)

During test generation, all circuit parameters (e.g., device sizes and loads) remain fixed. In contrast, timing parameters (e.g., arrival times, transition times) may change from vector to vector. So the delay and transition times for a two-input NAND gate can be represented by functions of timing variables.

Given the *arrival times* and *transition times* of transitions at a gate's inputs, we compute the *gate delay* and *output transition time*. The *output arrival time* of a gate is computed using the input arrival times and gate delay.

Consider only the cases where all inputs of a gate have either non-controlling values or transitions to the same value. The gate delay and output transition time of a two-input NAND gate is represented by the following timing functions (Figure 4): (a) fall delay function (from input pin X), $\mathbf{d^{Z,X}_F}(T^X_R)$; (b) fall transition time function (from input pin X), $\mathbf{t^{Z,X}_F}(T^X_R)$; (c) rise delay function for two simultaneous input switching, $\mathbf{d^Z_R}(T^X_F , T^Y_F , \delta^{X,Y})$; and (d) rise transition time function for two simultaneous input switching, $\mathbf{t^Z_R}(T^X_F , T^Y_F , \delta^{X,Y})$.

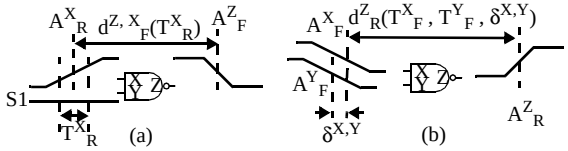

FIGURE 4. (a) Fall delay function and (b) rise delay function.

3.3 Trends with Respect to Single Variables

The relations between output variables and each input variable for a two-input NAND gate (Figure 1) are further detailed in Figure 5.

Based on extensive simulations, we have identified that for fixed $\delta^{X,Y}$ and T^Y_{tr} , the gate delay as a function of T^X_{tr} (Figure 5 (a), (b)) may be either (1) monotonically increasing or (2) bi-tonic (monotonically increasing and then monotonically decreasing in this case). In case (2), the pin-to-pin delay may become negative for large T^X_{tr} . In such case, this bi-tonicity is due to the fact that

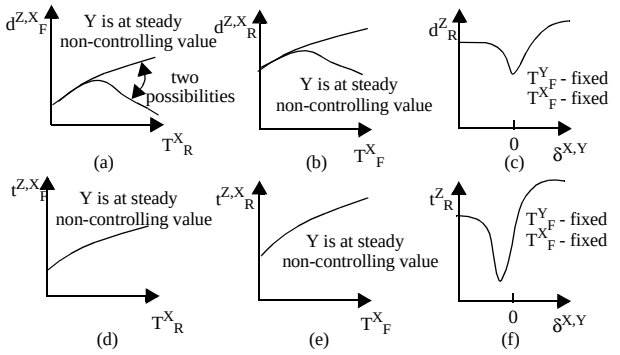

FIGURE 5. Timing functions vs. input variables.

the input transition starts to pull up (down) the output voltage before the actual arrival time of the input transition, i.e., the time it reaches 0.5Vdd. The effective βn/βp ratio determines which shape the T^X_{tr} - $d^{Z,X}_{tr}$ curves take. Below we only treat case (2), because (1) is a special case of (2) with the curve's peak at T^X_{tr} = infinity. The output transition time will always increase as T^X_{tr} increases (Figure 5 (d), (e)). Delay and output transition times have similar shapes with respect to skew (Figure 5 (c), (f)). The minimal delay always occurs at $\delta^{X,Y} = 0$, but the minimal transition time does not.

3.4 Finding Empirical Formulas

$\mathbf{d^{Z,X}_F}$, $\mathbf{d^{Z,X}_R}$, $\mathbf{t^{Z,X}_F}$, and $\mathbf{t^{Z,X}_R}$ are derived as they are in the pin-to-pin model (SDF [5]). For small skew, $\mathbf{d^Z_R}$ and $\mathbf{t^Z_R}$ are constructed via simulation and curve fitting. $\mathbf{d^Z_R}$ is constructed as a function of input skew by fixing other variables. This function is represented by a V-shape function that has three important points (S0R, D0R), (SXR, DXR), and (SYR, DYR) shown in Figure 2. Here **S0R** = 0, and **D0R** is the minimal delay caused by simultaneous transitions at X and Y. **SXR** is the minimum skew $\delta^{X,Y}$ such that a transition on Y does not affect the gate delay for fixed T^X_F and T^Y_F. **DXR** is the delay caused by a single transition on X with transition time T^X_F. SYR and DYR are defined similarly. Here D0R and SXR are functions of T^X_F and T^Y_F. DXR is a function of T^X_F. We determined the general forms of D0R, SXR, and DXR from the experimental data, and performed curve fitting to find the best coefficients and powers. The expressions obtained are listed below.

$DXR(T^X_F) = K_{10}*(T^X_F)^2 + K_{11}*T^X_F + K_{12}$,

$D0R(T^X_F, T^Y_F) = (K_{20}*(T^X_F)^{1/3}+K_{21})*(K_{22}*(T^Y_F)^{1/3}+K_{23}) + K_{24}$, and

$SXR(T^X_F, T^Y_F) = K_{30}*(T^X_F)^2 + K_{31}*(T^Y_F)^2 + K_{32}*T^X_F*T^Y_F + K_{33}*T^X_F + K_{34}*T^Y_F + K_{35}$.

Here the gate delay is defined with respect to the arrival time of the earliest transition at the gate input. $\mathbf{t^Z_R}$ and $\mathbf{d^Z_R}$ are constructed in a similar way, except that S0R for $\mathbf{t^Z_R}$ may be non-zero. Note that $\mathbf{d^Z_R}$ is a function of input transition times and input skew.

3.5 Validation of the Approximation

We validated the assumption of the V-shape approximations for the d^Z_R-$\delta^{X,Y}$ curves. The arguments substantiating the following claims can be found in [9].

Claim 1: The minimal delay point in function $d^Z_R(T^X_F, T^Y_F, \delta^{X,Y})$ for NAND gate Z is always at $\delta^{X,Y} = 0$ for any given values of T^X_F, T^Y_F.

Claim 2: The V-shape approximation in Figure 2 accurately captures the general shape of $d^Z_R(T^X_F, T^Y_F, \delta^{X,Y})$ for all fixed values of T^X_F and T^Y_F.

3.6 Extended Model

We treat the delay as increasing linearly as load increases. The proposed model has been extended to handle different numbers of inputs, input positions, and more than two simultaneous to-controlling transitions. More details can be found in [9]. Considering the effect of pre-initialization [7], we are currently developing a delay model for simultaneous to-non-controlling transitions for STA and ITR based on the simplified model of [19].

3.7 Characterization Efforts

Computation of pin-to-pin delay is usually a part of timing characterization of library cells [5]. For each NAND/NOR gates with different transistor sizes in a cell library, formulas for DXR, D0R, and SXR need to be determined in pre-characterization. Note that this is an one-time effort required for building up the new delay model for a given set of cells.

4. STATIC TIMING ANALYSIS

Static timing analysis provides min-max timing ranges for each line in a circuit for both rising and falling transitions. The ranges represent bounds on minimum and maximum delay values over all possible pairs of vectors. In timing analysis (Figure 6) arrival times (**A**) and transition times (**T**) at a gate's output are calculated based on these values at gate inputs. These values are computed via a forward traversal starting at the primary inputs. Similarly, the required times (**Q**) are computed via a backward traversal starting at primary outputs. If the arrival time range does not overlap with the required time range for the rising/falling transitions at a line, then the given timing requirements cannot be satisfied and a delay error is found. Delay transfer functions for forward and backward calculations in timing analysis are defined for the proposed model. If all input values are specified, timing ranges become points.

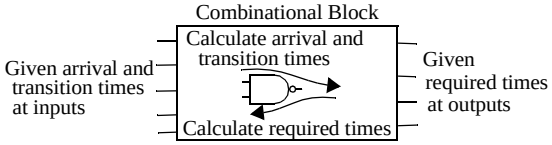

FIGURE 6. Overall structure of timing analysis.

4.1 Timing Information

In our min-max range representation, the timing windows in [8] are used (Figure 7). The earliest/latest arrival times and the shortest/longest transition times of rise/fall transitions are recorded for calculating the timing information for the next stage. The smallest (largest) arrival time of falling (rising) transition on line X is represented as $\mathbf{A^X_{F,S}}$ ($\mathbf{A^X_{R,L}}$). Transition and required times are represented similarly.

- Arrival time (A) and transition time (T) -- Rise/Fall Smallest/Largest
- Required time (Q) for timing analysis-- Rise/Fall Smallest/Largest

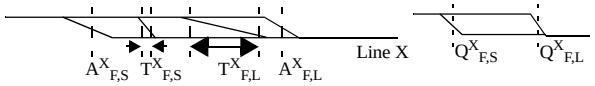

FIGURE 7. Timing information used in our method.

4.2 Calculating Arrival and Transition Times

Given the arrival and transition times at a gate's inputs, we calculate the corresponding quantities for the gate's outputs. The relations between output variables and input variables in Section 3.3 help identify the input A/T combinations that possibly induce worst case values on the computed output quantities. A/T calculations for an output falling transition use the pin-to-pin delay and

have been shown in [10]. Calculation of A/T for output rising transition (Figure 8) using our new delay model is shown and explained below.

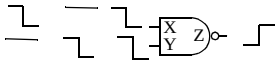

FIGURE 8. Possible input combinations for output rising transition.

$$A^Z_{R,S} = \min [A^X_{F,S}, A^Y_{F,S}] + \min_{\beta,\gamma \in \{S,L\}} [\mathbf{d^Z_R}(T^X_{F,\beta}, T^Y_{F,\gamma}, A^Y_{F,S} - A^X_{F,S})].$$

$$A^Z_{R,L} = \max [A^X_{F,L} + \mathbf{[d^{Z,X}_R}(T^{X*}_F)], A^Y_{F,L} + [\mathbf{d^{Z,Y}_R}(T^{Y*}_F)]]$$

$$\text{where } T^{X*}_F = \begin{cases} T^X_{F,\max}, & \text{if } T^X_{F,\max} \in (T^X_{F,S}, T^X_{F,L}); \\ T^X_{F,S}, & \text{else if } \mathbf{d^{Z,X}_R}(T^X_{F,S}) > \mathbf{d^{Z,X}_R}(T^X_{F,L}); \\ T^X_{F,L}, & \text{otherwise.} \end{cases}$$

$\mathbf{T^X_{F,\,max}}$ is the value of T^X_F that maximizes $\mathbf{d^{Z,X}_R}(T^X_F)$, for T^{Y*}_F is defined similarly.

$$T^Z_{R,S} = \begin{cases} \mathbf{t^Z_R}(T^X_{F,S}, T^Y_{F,S}, SK_{t,R,min}) \\ \quad \text{if } (A^X_{F,S} + SK_{t,R,min}, A^X_{F,L} + SK_{t,R,min}) \cap (A^Y_{F,S}, A^Y_{F,L}) \neq \varnothing; \\ \min[\mathbf{t^Z_R}(T^X_{F,S}, T^Y_{F,S}, A^Y_{F,S} - A^X_{F,L}), \\ \quad \mathbf{t^Z_R}(T^X_{F,S}, T^Y_{F,S}, A^Y_{F,L} - A^X_{F,S})], \quad \text{otherwise.} \end{cases}$$

Here, $SK_{t,R,min}$ is the value of skew $\delta^{X,Y}$ that minimizes $\mathbf{t^Z_R}(T^X_F, T^Y_F, \delta^{X,Y})$ for a given T^X_F, T^Y_F. Similarly, $T^Z_{R,L} = \max[\mathbf{t^{Z,X}_R}(T^X_{F,L}), \mathbf{t^{Z,X}_R}(T^Y_{F,L})]$.

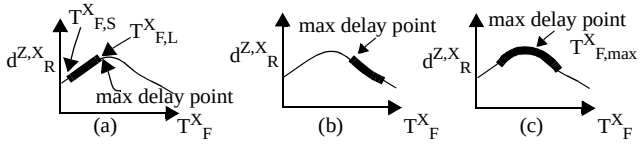

FIGURE 9. Possible transition time min-max range in $\mathbf{T^X_F}$-$\mathbf{d^{Z,X}_R}$ curve.

For an output rising transition to arrive as early as possible, we expect all input transitions to arrive as early as possible. The transition times at both X and Y should be either minimal or maximal, depending on which one causes shorter pin-to-pin delay on X and Y, since the shortest delay may be caused by the shortest (Figure 9.(a)) or longest transition time (Figure 9.(b)), but not at any other time in between.

Since simultaneous to-controlling transitions may speed up the output transition, A^Z_R is maximized either when only one input transition occurs, or when the lagging input transition does not affect A^Z_R in the case where transitions occur at both inputs. The maximal gate delay may occur when the input transition times are (a) maximal, (b) minimal, or (c) at some values in between. These three scenarios correspond respectively to the three cases shown in Figure 9, where the min-max range is to the left of the peak, to the right of the peak, or straddles the peak.

Although minimal gate delay always occurs when $\delta^{X,Y} = 0$, minimal output rising transition time may occur when $\delta^{X,Y} = SK_{t,R,min} \neq 0$. $\delta^{X,Y}$ may be equal to $SK_{t,R,min}$ if $(A^X_{F,S} + SK_{t,R,min}, A^X_{F,L} + SK_{t,R,min}) \cap (A^Y_{F,S}, A^Y_{F,L}) \neq \varnothing$. Otherwise, either minimal or maximal $\delta^{X,Y}$ closest to $SK_{t,R,min}$ will cause a minimal output transition time. A minimal output transition time occurs when both input transition times are minimal since it monotonically increases with T^X_F and T^Y_F.

Calculation of the required times for STA can be found in [9].

5. INCREMENTAL TIMING REFINEMENT

STA provides vector-independent min-max timing ranges for rising and falling transitions on each line. It can be used to provide initial timing information for test generation since a test generator starts with all unknown values. But as more specific values are assigned during the test generation process, the min-max ranges will become narrower due to (1) the increased specificity of the input vector pair, and (2) the logic and timing dependencies between lines. Thus, worst case corners used during STA may become impossible after some input values are specified.

Incremental timing refinement (ITR) [10] uses the min-max timing ranges computed from static timing analysis as the initial timing information. At each test generation step, a more specific value is assigned to one or more circuit lines. The min-max ranges for timing parameters shrink due to re-calculation of arrival, transition, and required times. The shrinking of timing ranges helps timing oriented test generator eliminate invalid choices. We will demonstrate how to perform ITR on our new delay model by identifying worst case corners used in ITR.

5.1 Logic Value System

For timing simulation and test generation, two-pattern tests are needed to create transitions carrying timing information. In addition to the timing information, a sequence of two values, (v_1, v_2), is used to record the logic information for each line. The values in each time-frame could be 0, 1, or x, where **x** represents the unspecified value for a primary input, and the unknown value for any other line. As the value at a line is further specified, forward and backward logic implications may refine the values at other lines. The required implication procedure can be obtained by extending a basic implication method ([20]) to two timeframes.

Among the nine logic values, {00, 01, 0x, 10, 11, 1x, x0, x1, xx}, for two-frame logic, 01 specifies a rising transition. 0x, x1, and xx specify a potential rising transition. According to the analysis for transitions on the nine logic values, we define the **state** of a transition *tr* on line Z, $\mathbf{S^Z_{tr}}$, as follows:

$$S^Z_{tr} = \begin{cases} 1, \text{ if line Z has a transition } tr; \\ 0, \text{ if line Z potentially has a transition } tr; \\ -1, \text{ if line Z definitely does not have a transition } tr. \end{cases}$$

S^Z_{tr} can be computed according to the logic value on Z, where $tr \in \{R, F\}$. When S^Z_{tr} is -1, none of the timing values pertaining to the transition *tr* at line Z are meaningful; each timing value may hence be left undefined. (Verifying the state of a line before using this line's timing values will avoid these undefined values from being used incorrectly.) In the other two cases ($S^Z_{tr} = 1$ or 0), the timing fields are identical to those in STA. STA is a special case of ITR where $S_{tr} = 0$ for every line. A method to calculate the timing values at each line is illustrated next.

5.2 Calculating Arrival and Transition Times

Again a NAND gate with output Z and two inputs X and Y is used for illustrating ITR. An **optimization target** ($\mathbf{OPT^Z_{tr,\,extreme}}$) is an assignment of *OPT* to line Z whose *extreme* value is desired for transition *tr*, where $OPT \in \{T, A\}$, $tr \in \{R, F\}$ and $extreme \in \{S, L\}$. S^Z_{tr} is only related to $S^X_{\overline{tr}}$ and $S^Y_{\overline{tr}}$, where $\overline{tr}$ is R(F) when *tr* is F(R). In the following, this mapping is always assumed if we do not specify the transition direction.

If the output Z has a rising transition, then the first frame of X and Y should each be 1. During ITR, we temporarily perform backward implication to reduce the number of combinations considered at the gate inputs and hence to obtain tighter timing ranges.

To find the extreme value for an optimization target on Z, we need to determine (1) if this extreme value prefers single or multiple input transitions, (2) given the current logic values on X and Y (we lose some choices at X if $S^X = 1$ or -1, similarly for Y), do we prefer to have transitions on X and Y if $S^X = 0$ ($S^Y = 0$), and (3) for the inputs with transitions, which corner to pick (minimal, maximal, or peak) for A and T to excite the extreme value on the optimization target.

For exciting the extreme value on an optimization target, the line with potential transitions ($S^X = 0$) will be set as either the transition occurs ($S^X = 1$) or it does not occur ($S^X = -1$), depending on the optimization target. That is, the zero value S^X should be set to 1 or -1 depending on S^Y. Rules of setting zero-value S^X for minimizing arrival time at Z (for both rising and falling) are shown below:

1. $S^Y = -1$: set S^X to 1 for creating a transition at Z.
2. $S^Y = 1$ and to-controlling transition occurs at Y: set S^X to 1 because additional input transition may speed up the output transition.
3. $S^Y = 1$ and to-non-controlling transition occurs at Y: set S^X to -1 because additional input transition may slow down the output transition.
4. $S^Y = 0$ and possible to-controlling transition at Y: set (S^X, S^Y) to (1, 1), because simultaneous switches in this direction speed up the output transition.
5. $S^Y = 0$ and possible to-non-controlling transition at Y: try two possibilities, namely (S^X, S^Y) = (1, -1) and (-1, 1), because simultaneous switches are not desired but at least one input transition is required to create a transition at the output.

TABLE 1. The implied values of S for obtaining the extreme cases for optimization target.

original input state		Optimization target															
		$A^Z_{F,S}$		$A^Z_{F,L}$		$A^Z_{R,S}$		$A^Z_{R,L}$		$T^Z_{F,S}$		$T^Z_{F,L}$		$T^Z_{R,S}$		$T^Z_{R,L}$	
S^X	S^Y	S^X_R	S^Y_R	S^X_R	S^Y_R	S^X_F	S^Y_F	S^X_F	S^Y_F	S^X_F	S^Y_F	S^X_F	S^Y_F	S^X_F	S^Y_F	S^X_F	S^Y_F
0	-1	**1**	-1	**1**	-1	**1**	-1	**1**	-1	**1**	-1	**1**	-1	**1**	-1	**1**	-1
0	0	**1**	**-1**	**1**	**1**	**1**	**1**	**1**	**-1**	**1**	**-1**	**1**	**-1**	**1**	**1**	**1**	**-1**
		-1	**1**					**-1**	**1**	**-1**	**1**	**-1**	**1**	**1**	**-1**	**-1**	**1**
														-1	**1**		
0	1	**-1**	1	**1**	1	**1**	1	**-1**	1	**1**	1	**1**	1	**1**	1	**1**	1
										-1	1	**-1**	1	**-1**	1	**-1**	1

According to these five rules, we obtain the zero-value setting of S^X and S^Y for $A^Z_{F,S}$ and $A^Z_{R,S}$. The setting for extreme values on optimization targets are shown in Table 1. Only the cases where S^X is 0 are shown (the cases where S^Y is 0 are symmetric to the shown cases). Non-zero S^X and S^Y will not be changed, so the cases with both S^X and S^Y as non-zero are not shown. There are cases where worst case corners can not be covered by one set of zero-value setting. In these cases, all input setting listed in Table 1 need to be tried, and the worst case values among them are picked. After S^X and S^Y are found, the calculation of the arrival times, transition times, and required times for ITR can be carried out [9]. Extension of this approach for gates with more than two inputs is also described in [9].

6. EXPERIMENT RESULTS

6.1 Delay Model

The proposed delay model has been implemented and compared with HSPICE and with the inverter-collapsing methods of Jun [6] and Nabavi [18]. The improved input mapping method for simultaneous switching at more than two inputs proposed in [19] is also integrated into Jun's approach. Empirical data are obtained from HSPICE simulation using SPICE LEVEL 3 model and 0.5 µm technology. NAND gates with minimum-size transistors are used for comparison. Each gate drives a minimum-size inverter as a load. To-controlling transitions are applied to some gate inputs. The non-controlling value is applied to the remaining inputs.

Figure 10 shows the pin-to-pin delay at position 4 of a five-input NAND gate. Since current inverter-collapsing methods [6][18] do not consider input position, the error rate may be high even for a single input transition. When the same transition is applied at the position 0 of a five-input NAND gate, all these approaches match HSPICE results.

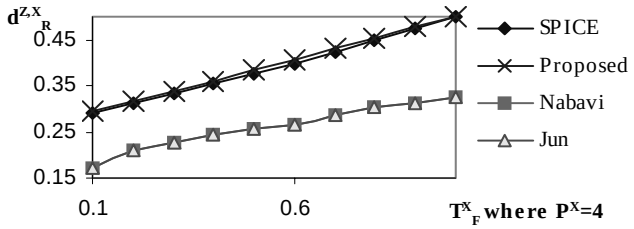

FIGURE 10. Single transition on position 4 of NAND5.

Figure 11 shows the result of simultaneous transitions at the NAND gate in Figure 1 when $\delta^{X,Y} = 0$ and $T^X_F = 0.5$ns. Jun's and our methods perform well but Nabavi's method performs well only when the transition times of the two inputs are close to each other. The reason is that this approach mainly considers simultaneous transitions with the same start time, but the formula obtained for this case does not extend well to the general case.

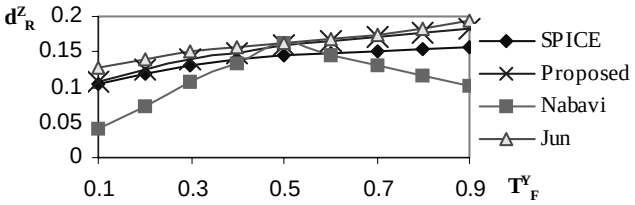

FIGURE 11. Simultaneous switch on NAND2 with single input transition time change.

For a two-input NAND gate with fixed T^X_F and T^Y_F, Figure 12 shows the delay as $\delta^{X,Y}$ changes. Our approach matches with HSPICE. Jun's approach fails to capture the delay for large skew. Nabavi's approach is the least accurate.

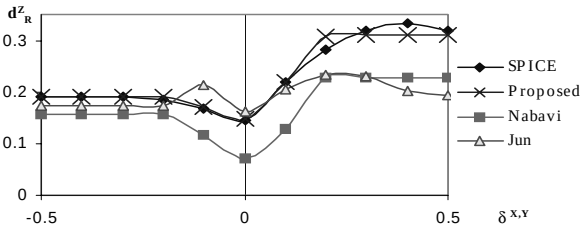

FIGURE 12. Vary $\delta^{X,Y}$ on simultaneous switch at NAND2.

Similar results for three simultaneous transitions are shown in [9]. As some timing variables are not captured in the models of Jun and Nabavi, these methods work well only when some timing conditions are satisfied. In contrast, our approach works for more general cases.

In addition to the improved accuracy, our model can be used as the timing model for STA and ITR where the worst case corners need to be identified. These corners are difficult to identify not only for equation solving models like [12] and table lookup models like [14][15][16][17] but also for some empirical models like [6][18]. For a model to be used in our method to find the worst case corners, a sufficient condition is that all timing functions of this model are monotonic or bi-tonic respect to each input variable.

6.2 Static Timing Analysis

Computing accurate min-delay for STA is important. For example, in advanced microprocessor designs, min-delay violation is treated as a serious potential problem, and a lot of buffers are inserted into the design to avoid this violation.

We compare STA results based on pin-to-pin delay model and our delay model on ISCAS85 benchmark circuits. Compared with pin-to-pin delay, STA that uses our delay model obtains the same max-delay on circuit blocks, but smaller min-delays. The min-

delay values shown in Table 2 are obtained from the union of primary outputs' timing ranges, which determines if potential hold/ setup time violations exist according to STA. Among the nine benchmarks used, the pin-to-pin model causes 5 to 31% error on min-delay in the six benchmarks listed. These two models give the same min-max ranges for three other benchmarks which are not listed. These percentages show that, in STA, the effects of simultaneous to-controlling transition may not be ignored, even in large circuits.

TABLE 2. Min-delay at outputs of ISCAS85 benchmarks.

Circuit	c17	c880	c1355	c1908	c3540	c7552
Pin-to-pin delay	352	716	736	322	463	154
Our model	268	590	723	305	440	147
Ratio	1.31	1.21	1.08	1.09	1.05	1.05

7. TIMING BASED ATPG

A crosstalk delay fault [8] is used to illustrated how our delay model, along with STA and ITR, can be used in a timing based ATPG.

To generate a test for a crosstalk delay fault, one (or more) crosstalk fault site(s) should be identified first (see Figure 13). For a site, a crosstalk fault *excitation criteria* is needed to determine the required two-frame logic values and arrival/transition times at the inputs of the fault site (A and B). A crosstalk *fault model* is needed to determine the logic/timing information at the outputs of the fault site (C and D). Then traditional *two-pattern ATPG methods* and *ITR* can be used to determine if there exists a test that can excite the fault and propagate its effect to a primary output or a flip-flop with setup time violation. The required times at A and B should be within the min-max ranges with relative arrival time constraints on these two lines. This timing information is compatible with STA and ITR, so both techniques can be used by the ATPG. The required timing ranges at A and B imply that the crosstalk delay model should have the capability to deal with min-max ranges (mainly worst case corner identification technique), so the timing-based ATPG can incorporate the fault mode, STA, and ITR.

We suggest that a timing-based ATPG must contain components (1) a delay model (both fault model and fault-free models need to have the capability to deal with timing ranges), (2) fault excitation conditions at the faulty sites, and propagation conditions in the fault-free sites, (3) a search engine to implicitly enumerate the logic search space, and (4) ITR that computes more accurate timing ranges when logic values or timing ranges at lines are further specified (timing violation should be checked after the new timing ranges are calculated).

This framework has been implemented in a crosstalk fault ATPG [10], where ITR improved ATPG efficiency (% of targeted faults that are detected or identified undetectable) from 39.63% to 82.75%.

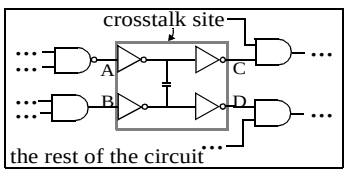

FIGURE 13. A view of the circuit during crosstalk test generation

8. CONCLUSIONS

We have developed a new model to capture the delay of simultaneous to-controlling transitions and input positions. By simplifying a linear approximation to the skew-delay relation of a two-input NAND gate and using curve fitting to the empirical results, general forms of delay equations have been developed. The experiments show that this model provides higher accuracy than other delay models developed for the same purpose. They also show that the simultaneous to-controlling transition effects should not be ignored. The model has been extended to more general cases.

Based on our new delay model, we have developed the delay transfer functions for static timing analysis and incremental timing refinement. Through worst case corner identification, we guarantee the correct propagation of min-max timing ranges for this delay model. A sufficient condition for adopting the worst case corner identification method for STA and ITR for a delay model is reported. Approaches for constructing timing-based ATPG utilizing the ITR framework are proposed.

9. REFERENCES

[1] R. B. Hitchcock, "Timing verification and timing analysis program", Proc. of 19th ACM/IEEE DAC, pp. 594-604, 1982.
[2] C. Visweswariah and R. A. Rohrer, "Piecewise approximate circuit simulation", IEEE Trans. on CAD, vol. 10, pp. 861-870, July 1991.
[3] Y. H. Shih, Y. Leblebici, and S. M. Kang, "ILLIADS: A fast timing and reliability simulator for digital MOS circuits", IEEE Trans. on CAD, vol. 12, pp. 1387-1402, Sept. 1993.
[4] L. W. Nagel, "SPICE2, A computer program to simulate semiconductor circuits", Memo UCB / ERL M520, Univ. Cal., Berkeley, May 1975.
[5] IEEE DASC standard delay format (SDF) - web page http://vhdl.org/vi/sdf/.
[6] Y. H. Jun, K. Jun, and S. B. Park, "An accurate and efficient delay time modeling for MOS logic circuits using polynomial approximation", IEEE Trans. on CAD, vol. 8, pp. 1027-1032, Sept. 1989.
[7] P. Franco and E. J. McCluskey, "Three-pattern tests for delay faults", Proc. of VLSI Test Symposium, pp. 452-456, 1994.
[8] W. Y. Chen, S. K. Gupta, and M. A. Breuer, "Test generation for crosstalk-induced delay in integrated circuits", Proc. of International Test Conference, pp. 191-299, 1999.
[9] L. C. Chen, S. K. Gupta, and M. A. Breuer, "Incremental timing refinement on a min-max delay model", Computer Engineer technical report No. 00-01, Electrical Engineer - System Dept., University of Southern California, April 2000.
[10] L. C. Chen, S. K. Gupta, and M. A. Breuer, "A new framework for static timing analysis, incremental timing analysis, and timing simulation", Proc. of Ninth Asia Test Symposium, pp.102 -107, 2000.
[11] J. Rubenstein, P. Penfield, Jr., and M. A. Horowitz, "Signal delay in RC networks", IEEE Trans. on CAD, vol. CAD-2, pp.202-211, July 1983.
[12] L. Bisdounis, S. Nikolaidis, and O. Koufopavlou, "Analytical Transient Response and Propagation Delay Evaluation of the CMOS Inverter for Short-Channel Devices", IEEE J. Solid-State Circuit, Vol.33, pp.302-306, Feb. 1998.
[13] T. Sakurai and A. R. Newton, "Alpha-power law MOSFET model and its applications to CMOS inverter delay and other formulas", IEEE J. Solid-State Circuit, Vol.25, pp.584-594, Apr. 1990.
[14] V. B. Rao, T. N. Trick, and I. N. Hajj, "A table-driven delay-operator approach to timing simulation of MOS VLSI circuits", Proc. of IEEE ICCD, pp.445-448, Nov. 1983.
[15] F. C. Chang, C. F. Chen, and P. Subramaniam, "An accurate and efficient gate level delay calculator for MOS circuits", Proc. of 25th ACM / IEEE DAC, pp. 282-287, 1988.
[16] D. Overhauser and I. Hajj, "A tabular macromodeling approach to fast timing simulation including parasitics", Proc. of IEEE ICCAD, pp. 70-73, 1988.
[17] V. Chandramouli and K. A. Sakallah, "Modeling the effects of temporal proximity of input transitions on gate propagation delay and transition time", Proc. of 32th ACM/IEEE DAC, pp. 617-622, 1996.
[18] A. Nabavi-Lishi and N. C. Rumin, "Inverter models of CMOS gates for supply current and delay evaluation", IEEE Trans. on CAD, vol. 13, pp. 1271-1279, Oct. 1994.
[19] A. Chatzigeorgiou, S. Nikolaidis, and I. Tsoukalas, "A modeling technique for CMOS gates", IEEE Trans. on CAD, vol. 18, pp. 557-575, May 1999.
[20] M. Abramovici, M. Breuer, and A. Friedman, *Digital Systems Testing and Testable Design*, IEEE Press, 1995.